aws

Datenbankentwicklerhandbuch

Amazon Redshift

Amazon Redshift: Datenbankentwicklerhandbuch

Table of Contents

Einführung

Willkommen beim Datenbankentwicklerhandbuch zu Amazon Redshift. Amazon Redshift ist ein vollständig verwalteter Data-Warehouse-Service in Petabytegröße in der Cloud. Mit Amazon Redshift Serverless können Sie auf Daten zugreifen und diese analysieren, ohne die üblichen Konfigurationen wie bei einem bereitgestellten Data Warehouse vornehmen zu müssen. Ressourcen werden automatisch bereitgestellt und die Data-Warehouse-Kapazität wird intelligent skaliert, um eine schnelle Leistung selbst für anspruchsvollste und unvorhersehbare Workloads zu erzielen. Es fallen keine Kosten an, wenn das Data Warehouse inaktiv ist, Sie zahlen also nur für das, was Sie tatsächlich nutzen. Unabhängig von der Größe des Datensatzes können Sie Daten laden und sofort mit der Abfrage beginnen. Hierfür können Sie Amazon Redshift Query Editor v2 oder Ihr bevorzugtes Business Intelligence (BI)-Tool nutzen. Genießen Sie das beste Preis-Leistungs-Verhältnis und die vertrauten SQL-Funktionen in einer easy-to-use Umgebung ohne Verwaltungsaufwand.

Dieser Leitfaden konzentriert sich auf die Verwendung von Amazon Redshift zur Erstellung und Verwaltung eines Data Warehouse. Wenn Sie als Designer, Softwareentwickler oder Administrator mit Datenbanken arbeiten, finden Sie hier die Informationen, die Sie benötigen, um Ihr Data Warehouse zu entwerfen, zu erstellen, abzufragen und zu pflegen.

Themen

Voraussetzungen

Bevor Sie dieses Handbuch verwenden, sollten Sie den Abschnitt Amazon Redshift Serverless lesen, in dem die Ausführung der folgenden Aufgaben beschrieben wird.

- Erstellen Sie ein Data Warehouse mit Amazon Redshift Serverless.
- Laden von Beispieldaten mit Amazon Redshift Query Editor v2
- Laden von Daten aus Amazon S3

Sie sollten auch wissen, wie Sie Ihren SQL-Client verwenden und über ein grundlegendes Verständnis der SQL-Sprache verfügen.

Sind Sie Datenbankentwickler?

Wenn Sie Amazon Redshift zum ersten Mal verwenden, empfehlen wir Ihnen, zunächst den Abschnitt Amazon Redshift Serverless mit Informationen für den Einstieg zu lesen.

Wenn Sie Datenbankbenutzer, Datenbankdesigner, Datenbankentwickler oder Datenbankadministrator sind, hilft Ihnen die folgende Tabelle bei Ihrer Suche.

Wenn Sie …	Wir empfehlen...
Erfahren Sie mehr über die interne Architektur des Amazon-Redshift-Data-Warehouse.	Die System- und Architekturübersicht bietet eine allgemeine Übersicht über die interne Architektur von Amazon Redshift. Eine umfassendere Übersicht über den Amazon-Redshift-Webservice finden Sie auf der Amazon-Redshift-Produktdetailseite.
Erstellen Sie Datenbanken, Tabellen, Benutzer und andere Datenbankobjekte.	Common Database Tasks bietet eine kurze Einführung in die Grundlagen der SQL-Entwicklung. Amazon-Redshift-SQL stellt Syntax und Beispiele für Amazon-Redshift-SQL-Befehle und -Funktionen und andere SQL-Elemente bereit. Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift bietet eine Zusammenfassung unserer Empfehlungen für die Auswahl von Sortierschlüsseln, Verteilungsschlüsseln und Kompressionskodierungen.
Erfahren Sie, wie Sie Tabellen so gestalten, dass sie optimale Leistungen erbringen.	Arbeiten mit automatischer Tabellenoptimierung erläutert die Überlegungen zur Verwendung der Kompression für die Daten in Tabellenspalten sowie für die Auswahl von Verteilungs- und Sortierschlüsseln.
Daten laden.	Laden von Daten beschreibt die Verfahren beim Laden großer Datensätze aus Amazon-DynamoDB-Tabellen oder einfachen, in Amazon-S3-Buckets gespeicherten Dateien.

Wenn Sie …	Wir empfehlen...
	Bewährte Methoden für Amazon Redshift zum Laden von Daten enthält Tipps, um Ihre Daten schnell und effizient zu laden.
Verwalten Sie Benutzer, Gruppen und die Datenbank sicherheit.	Verwalten der Datenbanksicherheit behandelt Themen zur Datenbank sicherheit.
Überwachen und optimieren Sie die Systemleistung.	Referenz zu Systemtabellen und Ansichten erläutert Systemtabellen und Ansichten, die Sie nach dem Status der Datenbank abfragen, und mit denen Sie Abfragen und Prozesse überwachen können. Konsultieren Sie auch das Amazon-Redshift-Verwaltungshandbuch, um zu erfahren, wie Sie über die AWS Management Console den Systemzustand prüfen, Messwerte überwachen sowie Cluster sichern und wiederherstellen können.
Analysieren und melden Sie Informati onen aus sehr großen Datensätzen.	Viele beliebte Softwareanbieter zertifizieren Amazon Redshift mit ihren Angeboten, damit Sie die Toolkits, die Sie heute nutzen, auch weiterhin verwenden können. Weitere Informationen finden Sie auf der Seite Amazon-Redshift-Partner. SQL-Referenz enthält alle Details zu den SQL-Ausdrücken, -Befehlen und -Funktionen, die Amazon Redshift unterstützt.
Interagieren Sie mit Amazon-Redshift-Ressourcen und -Tabellen.	Lesen Sie den Amazon-Redshift-Serverless-API-Leitfaden, den Amazon-Redshift-API-Leitfaden und den Amazon-Redshift-Data-API-Leitfaden, um mehr darüber zu erfahren, wie Sie programmgesteuert mit Ressourcen interagieren und Operationen ausführen können.
Folgen Sie einem Tutorial, um besser mit Amazon Redshift vertraut zu werden.	Folgen Sie einem Tutorial unter Tutorials für Amazon Redshift, um mehr über die Funktionen von Amazon Redshift zu erfahren.

System- und Architekturübersicht

Das Data Warehouse von Amazon Redshift ist eine relationale Datenbank und ein Verwaltungssystem für Unternehmen.

Amazon Redshift unterstützt Client-Verbindungen mit vielen Arten von Anwendungen, einschließlich Business Intelligence (BI), Berichterstellung, Daten und Analysetools.

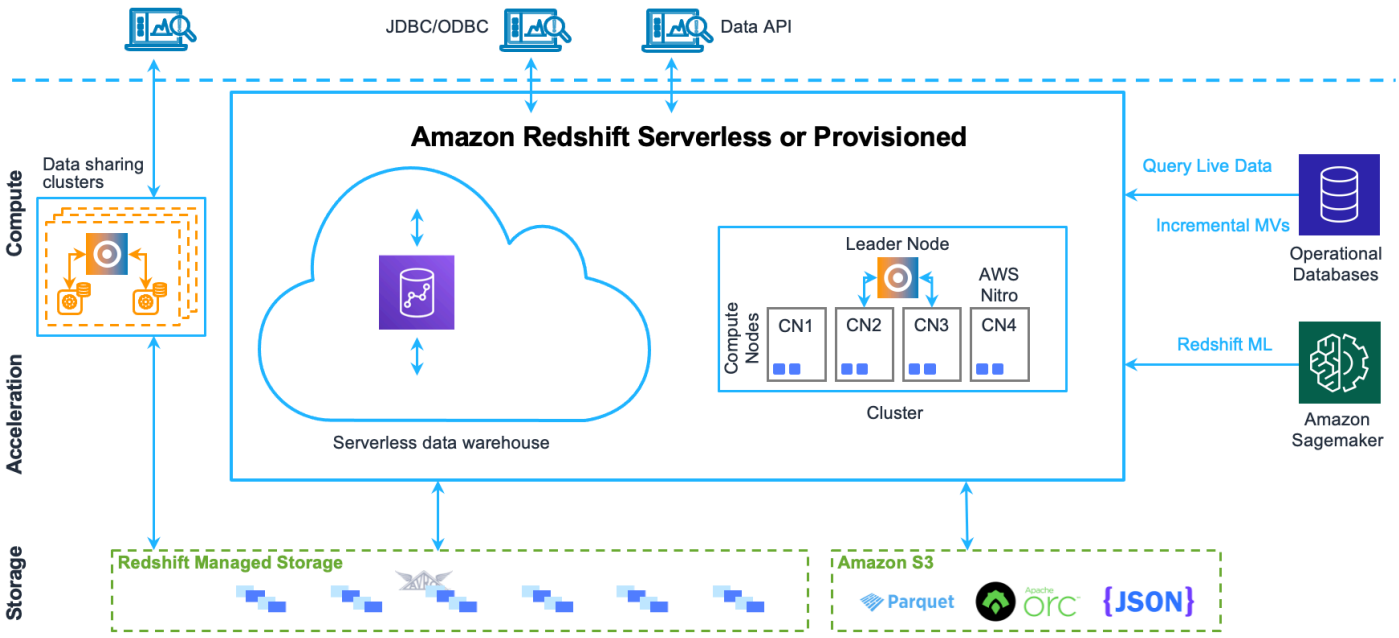
Bei Analyseabfragen werden große Datenmengen in mehrphasigen Operationen abgerufen, verglichen und bewertet, um ein Endergebnis zurückzugeben.

Durch eine Kombination von massiver paralleler Verarbeitung, Spaltendatenspeicherung und sehr effizienten, zielgerichteten Kodierungsschemata für Spaltendatenkomprimierung erzielt Amazon Redshift eine effiziente Speicherung und optimale Abfrageleistung. Dieser Abschnitt bietet eine Einführung in die Architektur des Amazon-Redshift-Systems.

Themen

Architektur des Data Warehouse-Systems

Dieser Abschnitt stellt die Elemente der Architektur des Data Warehouse von Amazon Redshift vor wie in der folgenden Abbildung gezeigt.



Clientanwendungen

Amazon Redshift lässt sich in verschiedene Datenlade- und ETL-Tools (Extrahieren, Transformieren und Laden) sowie in Business-Intelligence (BI)-Berichterstellungs-, Data-Mining- und Analysetools integrieren. Amazon Redshift basiert auf dem offenen Standard PostgreSQL, sodass die meisten vorhandenen SQL-Client-Anwendungen mit nur wenigen Änderungen funktionieren. Weitere Informationen zu wichtigen Unterschieden zwischen Amazon-Redshift-SQL und PostgreSQL finden Sie unter Amazon Redshift und PostgreSQL.

Cluster

Zentraler Bestandteil der Infrastruktur des Data Warehouse von Amazon Redshift ist ein Cluster.

Ein Cluster besteht aus einem oder mehreren Datenverarbeitungsknoten. Wird ein Cluster mit zwei oder mehr Datenverarbeitungsknoten bereitgestellt, koordiniert ein zusätzlicher Führungsknoten die Datenverarbeitungsknoten und verarbeitet die externe Kommunikation. Ihre Client-Anwendung interagiert nur mit dem Führungsknoten direkt. Die Datenverarbeitungsknoten sind für externe Anwendungen transparent.

Führungsknoten

Der Führungsknoten verwaltet die Kommunikation mit Client-Programmen sowie die gesamte Kommunikation mit Datenverarbeitungsknoten. Er analysiert und entwickelt Ausführungspläne für Datenbankoperationen, insbesondere die erforderlichen Schritte zum Abrufen von Ergebnissen

vor komplexe Abfragen. Auf der Grundlage des Ausführungsplans kompiliert der Führungsknoten Code, verteilt den kompilierten Code auf die Datenverarbeitungsknoten und weist jedem Datenverarbeitungsknoten einen Teil der Daten zu.

Der Führungsknoten verteilt SQL-Anweisungen nur dann an den Datenverarbeitungsknoten, wenn eine Abfrage auf Tabellen verweist, die auf dem Datenverarbeitungsknoten gespeichert sind. Alle anderen Abfragen werden ausschließlich auf dem Führungsknoten ausgeführt. Amazon Redshift wurde entwickelt, um bestimmte SQL-Funktionen nur auf dem Führungsknoten zu implementieren. Eine Abfrage, die eine dieser Funktionen verwendet, gibt einen Fehler zurück, wenn sie auf Tabellen verweist, die sich auf dem Datenverarbeitungsknoten befinden. Weitere Informationen finden Sie unter SQL-Funktionen, die auf dem Führungsknoten unterstützt werden.

Datenverarbeitungsknoten

Der Führungsknoten kompiliert Code für einzelne Elemente des Ausführungsplans und weist den Code einzelnen Datenverarbeitungsknoten zu. Die Datenverarbeitungsknoten führen den kompilierten Code aus und senden Zwischenergebnisse zur endgültigen Aggregation an den Führungsknoten zurück.

Jeder Datenverarbeitungsknoten verfügt über eine eigene dedizierte CPU und eigenen Arbeitsspeicher, die von dem Knotentyp bestimmt werden. Bei zunehmendem Workload können Sie die Rechenkapazität eines Clusters steigern, indem sie die Anzahl der Knoten erhöhen und/oder ein Upgrade des Knotentyps ausführen.

Amazon Redshift bietet verschiedene Knotentypen für Ihre Rechenanforderungen. Details zu den einzelnen Knotentypen finden Sie unter Amazon-Redshift-Cluster im Amazon-Redshift-Verwaltungshandbuch.

Redshift Managed Storage

Data-Warehouse-Daten werden in einer separaten Speicherschicht gespeichert – Redshift Managed Storage (RMS). RMS bietet die Möglichkeit einer Skalierung Ihres Speichers auf Petabyte mithilfe von Amazon-S3-Speicher. Mit RMS können Sie Rechenleistung und Speicher unabhängig voneinander skalieren und bezahlen. Somit können Sie die Größe des Clusters ausschließlich basierend auf Ihren Datenverarbeitungsanforderungen festlegen. Als Tier-1-Cache wird automatisch ein leistungsstarker lokaler SSD-basierter Speicher verwendet. Durch Optimierungen der Temperatur und des Alters von Datenblöcken sowie der Workload-Muster wird zudem eine hohe Leistung erzielt. Bei Bedarf wird der Speicher automatisch auf Amazon S3 skaliert, ohne dass ein Eingreifen erforderlich ist.

Knoten-Slices

Ein Datenverarbeitungsknoten ist in Slices aufgeteilt. Jedem Slice wird ein Teil des Arbeitsspeichers und des Festplattenspeichers des Knoten zugeordnet. Dort verarbeitet er einen Teil des dem Knoten zugewiesenen Workloads. Der Führungsknoten verwaltet die Verteilung von Daten an die Slices und teilt den Workload für Abfragen oder sonstige Datenbankoperationen auf die Slices auf. Die Slices arbeiten dann parallel, um die Operation abzuschließen.

Die Anzahl an Slices pro Knoten wird durch die Knotengröße des Clusters bestimmt. Weitere Informationen zur Anzahl der Slices für die einzelnen Knotengrößen finden Sie unter About clusters and nodes (Informationen zu Clustern und Knoten) im Amazon-Redshift-Verwaltungshandbuch.

Wenn Sie eine Tabelle erstellen, können Sie optional eine Spalte als Verteilungsschlüssel angeben. Wenn die Tabelle mit Daten geladen wird, werden die Zeilen entsprechend dem für die Tabelle definierten Verteilungsschlüssel an die Knoten-Slices verteilt. Ein guter Verteilungsschlüssel ermöglicht es Amazon Redshift, die parallele Verarbeitung zu verwenden, um effektiv Daten zu laden und Abfragen auszuführen. Informationen zur Auswahl eines Verteilungsschlüssels finden Sie unter Auswahl des besten Verteilungsstils.

Internes Netzwerk

Amazon Redshift nutzt Verbindungen mit hoher Bandbreite, räumliche Nähe und benutzerdefinierte Kommunikationsprotokolle für die Bereitstellung privater Netzwerkkommunikation in Ultrahochgeschwindigkeit zwischen dem Führungsknoten und den Datenverarbeitungsknoten. Die Datenverarbeitungsknoten werden auf einem getrennten, isolierten Netzwerk ausgeführt, auf das Client-Anwendungen niemals direkt zugreifen.

Datenbanken

Ein Cluster umfasst einen oder mehrere Datenbanken. Die Benutzerdaten werden auf den Datenverarbeitungsknoten gespeichert. Ihr SQL-Client kommuniziert mit dem Führungsknoten, der wiederum die Abfrageausführung mit den Datenverarbeitungsknoten koordiniert.

Amazon Redshift ist ein relationales Datenbankmanagementsystem (RDBMS), sodass es mit anderen RDBMS-Anwendungen kompatibel ist. Obwohl es dieselben Funktionen wie ein typisches RDBMS bereitstellt, einschließlich Funktionen zur Online-Transaktionsverarbeitung (Online Transaction Processing, OLTP), wie beispielsweise das Einsetzen und Löschen von Daten, ist Amazon Redshift optimiert für hochperformante Analysen und die Berichterstellung zu sehr großen Datensätzen.

Amazon Redshift basiert auf PostgreSQL. Zwischen Amazon Redshift und PostgreSQL gibt es eine Reihe sehr wichtiger Unterschiede, die Sie berücksichtigen müssen, wenn Sie Ihre Data-Warehouse-

Anwendungen entwerfen und entwickeln. Weitere Informationen zu den Unterschieden zwischen Amazon-Redshift-SQL und PostgreSQL finden Sie in Amazon Redshift und PostgreSQL.

Leistung

Amazon Redshift erzielt eine extrem schnelle Abfrageausführung mithilfe der folgenden Leistungsfunktionen.

Themen

- Massiv parallele Verarbeitung
- Spaltenweise Datenspeicherung
- Datenkompression
- Abfragenoptimierer
- Ergebnis-Zwischenspeicherung
- Kompilierter Code

Massiv parallele Verarbeitung

Die massiv parallele Verarbeitung (Massively Parallel Processing, MPP) ermöglicht die schnelle Ausführung äußerst komplexer Abfragen für sehr große Datenmengen. Mehrere Datenverarbeitungsknoten führen die gesamte Abfrageverarbeitung bis zur abschließenden Aggregation der Ergebnisse aus. Jeder Kern eines jeden Knotens führt dabei dieselben kompilierten Abfragesegmente auf Teilen der gesamten Daten aus.

Amazon Redshift verteilt die Zeilen einer Tabelle an die Datenverarbeitungsknoten, sodass die Daten parallel verarbeitet werden können. Durch die Auswahl eines geeigneten Verteilungsschlüssels für die einzelnen Tabellen können Sie die Verteilung der Daten optimieren, um den Workload auszugleichen und die Bewegung der Daten von Knoten zu Knoten zu verringern. Weitere Informationen finden Sie unter Auswahl des besten Verteilungsstils.

Das Laden von Daten aus Flat-Dateien nutzt die parallele Verarbeitung, indem der Workload auf mehrere Knoten verteilt und gleichzeitig aus mehreren Dateien gelesen wird. Weitere Informationen zum Laden von Daten in Tabellen finden Sie unter Bewährte Methoden für Amazon Redshift zum Laden von Daten.

Spaltenweise Datenspeicherung

Die spaltenweise Speicherung für Datenbanktabellen sorgt für eine drastische Reduzierung der allgemeinen Datenträger-I/O-Anforderungen und stellt einen wichtigen Faktor im Zusammenhang mit der Optimierung der Leistung analytischer Abfragen dar. Die spaltenweise Speicherung von Datenbanktabellen-Daten reduziert die Anzahl von Datenträger-I/O-Anfragen sowie die Menge an Daten, die Sie von der Festplatte laden müssen. Das Laden von weniger Daten in den Arbeitsspeicher ermöglicht Amazon Redshift, beim Ausführen von Abfragen weniger In-Memory-Verarbeitung auszuführen. Eine detaillierte Erläuterung finden Sie unter Spaltenweise Speicherung.

Wenn Spalten angemessen sortiert werden, kann der Abfrageverarbeiter schnell einen großen Teil von Datenblöcken herausfiltern. Weitere Informationen finden Sie unter Auswahl des besten Sortierschlüssels.

Datenkompression

Datenkompression reduziert die Speicheranforderungen und somit die Datenträger-I/O, wodurch die Abfrageleistung verbessert wird. Wenn Sie eine Abfrage ausführen, werden die komprimierten Daten in den Arbeitsspeicher gelesen und anschließend während der Abfrageausführung dekomprimiert. Das Laden von weniger Daten in den Arbeitsspeicher ermöglicht Amazon Redshift, mehr Arbeitsspeicher zum Analysieren der Daten zuzuordnen. Da bei der spaltenweisen Speicherung ähnliche Daten sequenziell gespeichert werden, kann Amazon Redshift Codierungen für adaptive Komprimierungen anwenden, die speziell mit spaltenweisen Datentypen verbunden sind. Die beste Möglichkeit, Datenkomprimierung in Tabellenspalten zu aktivieren, besteht darin, Amazon Redshift zu gestatten, optimale Komprimierungskodierungen anzuwenden, wenn die Tabelle mit Daten geladen wird. Weitere Informationen zur Verwendung der automatischen Datenkompression erhalten Sie unter Laden von Tabellen mit automatischer Kompression.

Abfragenoptimierer

Die Abfrageausführungs-Engine von Amazon Redshift enthält einen Abfragenoptimierer, der MPP-fähig ist und die spaltenweise Datenspeicherung nutzt. Der Abfragenoptimierer von Amazon Redshift implementiert wesentliche Verbesserungen und Erweiterungen für die Verarbeitungen komplexer Analyseabfragen, die oft Mehrtabellen-Joins, Unterabfragen und Aggregation beinhalten. Weitere Informationen zur Optimierung von Abfragen finden Sie unter Optimieren der Abfrageleistung.

Ergebnis-Zwischenspeicherung

Um die Laufzeit von Abfragen zu reduzieren und die Systemleistung zu verbessern, stellt Amazon Redshift die Ergebnisse bestimmter Abfragetypen in den Speicher auf dem Führungsknoten.

Wenn ein Benutzer eine Abfrage stellt, durchsucht Amazon Redshift den Ergebnis-Cache nach einer gültigen, zwischengespeicherten Kopie der Abfrageergebnisse. Wenn im Ergebnis-Cache ein passender Datensatz gefunden wird, verwendet Amazon Redshift die zwischengespeicherten Ergebnisse und führt die Abfrage nicht aus. Die Ergebnis-Zwischenspeicherung ist transparent für den Benutzer.

Die Ergebnis-Zwischenspeicherung ist standardmäßig aktiviert. Um die Ergebnis-Zwischenspeicherung für die aktuelle Sitzung zu deaktivieren, setzen Sie den Parameter enable_result_cache_for_session auf `off`.

Amazon Redshift verwendet für eine neue Abfrage zwischengespeicherte Ergebnisse, wenn die folgenden Dinge zutreffen:

- Der Benutzer, der die Abfrage absetzt, besitzt Zugriffsberechtigung für die in der Abfrage verwendeten Objekte.
- Die Tabelle oder die Ansichten in der Abfrage wurden nicht verändert.
- Die Abfrage verwendet keine Funktion, die bei jeder Ausführung überprüft werden muss, wie beispielsweise GETDATE.
- Die Abfrage verweist nicht auf externe Tabellen von Amazon Redshift Spectrum.
- Konfigurationsparameter, die sich auf die Abfrageparameter auswirken könnten, sind unverändert.
- Die Abfrage stimmt syntaktisch mit der zwischengespeicherten Abfrage überein.

Um die Effektivität der Zwischenspeicherung und die effiziente Nutzung der Ressourcen zu maximieren, speichert Amazon Redshift einige große Datensätze von Abfrageergebnissen nicht zwischen. Amazon Redshift stellt anhand verschiedener Faktoren fest, ob Abfrageergebnisse zwischengespeichert werden sollen. Zu diesen Faktoren zählen unter anderem die Anzahl der Einträge im Cache sowie der Instance-Typ Ihres Amazon-Redshift-Clusters.

Um festzustellen, ob eine Abfrage die Ergebnis-Zwischenspeicherung genutzt hat, rufen Sie die Systemansicht SVL_QLOG auf. Wenn eine Abfrage die Ergebniszwischenspeicherung verwendet hat, gibt die Spalte source_query die Abfrage-ID der Quellabfrage zurück. Wenn keine Ergebniszwischenspeicherung verwendet wurde, ist der Wert der Spalte source_query gleich NULL.

Das folgende Beispiel zeigt, dass die von der userid 104 und der userid 102 gestellten Abfragen die Abfragen von der userid 100 aus dem Ergebnis-Zwischenspeicher verwenden.

```
select userid, query, elapsed, source_query from svl_qlog
where userid > 1
```

```
order by query desc;

userid | query  | elapsed  | source_query
-------+--------+----------+-------------
   104 | 629035 |       27 |       628919
   104 | 629034 |       60 |       628900
   104 | 629033 |       23 |       628891
   102 | 629017 |  1229393 |
   102 | 628942 |       28 |       628919
   102 | 628941 |       57 |       628900
   102 | 628940 |       26 |       628891
   100 | 628919 | 84295686 |
   100 | 628900 | 87015637 |
   100 | 628891 | 58808694 |
```

Kompilierter Code

Der Führungsknoten verteilt vollständig optimierten kompilierten Code auf die Knoten eines Clusters. Durch das Kompilieren der Abfrage verringert sich der mit einem Interpreter verbundene Verwaltungsaufwand, wodurch insbesondere bei komplexen Abfragen eine höhere Laufzeitgeschwindigkeit erzielt wird. Der kompilierte Code wird im Cache gespeichert und von mehreren Sitzungen auf demselben Cluster gemeinsam verwendet. Zukünftige Ausführungen derselben Abfrage können somit schneller erfolgen – häufig sogar mit verschiedenen Parametern.

Die Abfrageausführungs-Engine kompiliert unterschiedlichen Code für die JDBC- und ODBC-Verbindungsprotokolle, sodass für zwei Clients, die verschiedene Protokolle verwenden, jeweils Kosten anfallen, wenn zum ersten Mal Code kompiliert wird. Clients, die dasselbe Protokoll verwenden, profitieren dann jedoch wieder von dem im Cache verfügbaren Code.

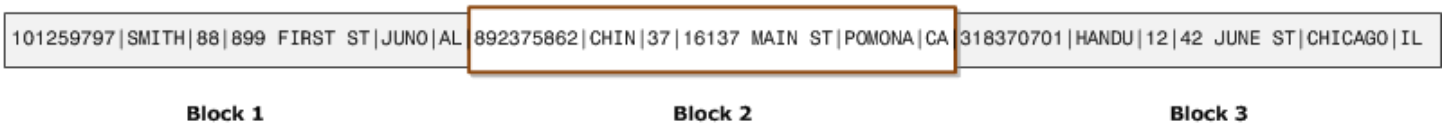
Spaltenweise Speicherung

Die spaltenweise Speicherung für Datenbanktabellen trägt wesentlich zur Optimierung der Leistung analytischer Abfragen bei, da sie die allgemeinen Datenträger-I/O-Anforderungen drastisch reduziert. Die Menge an Daten, die von der Festplatte geladen werden müssen, verringert sich.

Die folgenden Abbildungen zeigen, wie durch die spaltenweise Datenspeicherung Effizienzen erzielt werden und wie sich diese in Effizienzen beim Abrufen von Daten in den Arbeitsspeicher niederschlagen.

Die erste Abbildung zeigt, wie Datensätze aus Datenbanktabellen in der Regel zeilenweise in Datenträgerblöcken gespeichert werden.

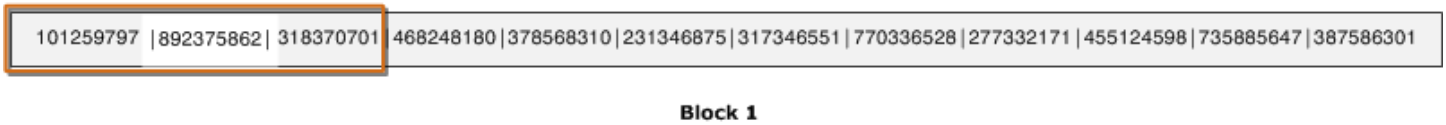
SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



In einer typischen relationalen Datenbanktabelle enthält jede Zeile Feldwerte für einen einzelnen Datensatz. Bei der zeilenweisen Datenbankspeicherung speichern Datenblöcke Werte sequenziell für alle aufeinanderfolgenden Spalten der gesamten Zeile. Wenn die Blockgröße kleiner ist als die Größe eines Datensatzes, nimmt die Speicherung eines vollständigen Datensatzes möglicherweise mehr als einen Block ein. Wenn die Blockgröße größer ist als die Größe eines Datensatzes, nimmt die Speicherung eines vollständigen Datensatzes möglicherweise weniger als einen Block ein, was eine ineffiziente Nutzung von Speicherplatz bedeutet. Bei Anwendungen für die Onlineverarbeitung von Transaktionen (Online Transaction Processing, OLTP) beinhalten die meisten Transaktionen häufiges Lesen und Schreiben aller Werte für ganze Datensätze – in der Regel ein Datensatz oder eine kleine Anzahl an Datensätzen gleichzeitig. Folglich ist die zeilenweise Speicherung optimal für OLTP-Datenbanken geeignet.

Die nächste Abbildung zeigt, wie bei der spaltenweise Speicherung die Werte für alle Spalten sequenziell in Datenträgerblöcken gespeichert werden.

SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



Bei der spaltenweisen Speicherung speichert jeder Datenblock die Werte einer einzelnen Spalte für mehrere Zeilen. Wenn Datensätze im System eintreffen, konvertiert Amazon Redshift die Daten transparent in die spaltenweise Speicherung für alle Spalten.

In diesem vereinfachten Beispiel für die spaltenweise Speicherung enthält jeder Datenblock Spaltenfeldwerte für bis zu dreimal so viele Datensätze wie bei der zeilenbasierten Speicherung. Dies bedeutet, dass zum Lesen derselben Anzahl an Spaltenfeldwerten für dieselbe Anzahl an Datensätzen verglichen mit der zeilenweisen Speicherung ein Drittel der I/O-Operationen erforderlich ist. In der Praxis ist die Speichereffizienz bei der Verwendung von Tabellen mit einer sehr hohen Anzahl an Spalten und Zeilen noch größer.

Ein weiterer Vorteil besteht darin, dass Blockdaten ein speziell für den Spaltendatentyp ausgewähltes Komprimierungsschema verwenden können, da alle Blöcke denselben Datentyp beinhalten. Dadurch werden der benötigte Speicherplatz und I/O weiter reduziert. Weitere Informationen zu Komprimierungsverschlüsselungen basierend auf Datentypen erhalten Sie unter Kompressionskodierungen.

Die Einsparungen an Speicherplatz für Daten auf der Festplatte gelten auch für das Abrufen und Speichern dieser Daten im Arbeitsspeicher. Da bei vielen Datenbankoperationen der Zugriff auf bzw. die Arbeit mit nur einer kleinen Anzahl an Spalten gleichzeitig erforderlich ist, lässt sich Speicherplatz einsparen, indem nur Blöcke für Spalten abgerufen werden, die Sie für eine Abfrage tatsächlich benötigen. Während OLTP-Transaktionen in der Regel die meisten Spalten in einer Zeile für eine kleine Anzahl an Datensätzen umfassen, lesen Data Warehouse-Abfragen in der Regel nur einige wenige Spalten für eine sehr große Anzahl an Zeilen. Dies bedeutet, dass zum Lesen derselben Anzahl an Spaltenfeldwerten für dieselbe Anzahl an Zeilen ein Bruchteil der I/O-Operationen erforderlich ist. Es wird nur ein Bruchteil des Arbeitsspeichers verwendet, der für die Verarbeitung zeilenweiser Blöcke erforderlich wäre. In der Praxis sind die Effizienzgewinne bei der Verwendung von Tabellen mit einer sehr hohen Anzahl an Spalten und Zeilen proportional größer. Nehmen wir beispielsweise an, eine Tabelle hat 100 Spalten. Eine Abfrage, die fünf Spalten verwendet, muss nur etwa fünf Prozent der in der Tabelle enthaltenen Daten lesen. Diese Einsparungen wiederholen sich bei großen Datenbanken für möglicherweise Milliarden oder sogar Billionen an Datensätzen. Im Gegensatz dazu würde eine zeilenweise Datenbank ebenfalls die Blöcke mit den 95 nicht benötigten Spalten lesen.

Typische Datenbankblock-Größen reichen von 2 KB bis 32 KB. Amazon Redshift verwendet eine Blockgröße von 1 MB. Dies ist effizienter und sorgt für eine weitere Reduzierung der Zahl der I/O-Anforderungen, die für das Laden von Datenbanken oder andere Operationen im Zusammenhang mit einer Abfrageausführung erforderlich sind.

Workload-Management

Mit dem Workload-Management (WLM) in Amazon Redshift können Benutzer Prioritäten innerhalb von Workloads flexibel verwalten, sodass kurze, schnelle Abfragen nicht hinter lange dauernden Abfragen in Warteschlangen hängen bleiben.

Amazon Redshift WLM erstellt Abfragewarteschlangen zur Laufzeit gemäß Service-Klassen, welche die Konfigurationsparameter für verschiedene Arten an Warteschlangen definieren, einschließlich interner Systemwarteschlangen und für Benutzer zugängliche Warteschlangen. Aus einer Benutzerperspektive sind eine für Benutzer zugängliche Service-Klasse und eine Warteschlange funktional vergleichbar. Aus Konsistenzgründen wird in dieser Dokumentation der Begriff Warteschlange sowohl für eine für Benutzer zugängliche Service-Klasse als auch für eine Warteschlange zur Laufzeit verwendet.

Wenn eine Abfrage ausgeführt wird, weist WLM die Abfrage entweder entsprechend der Benutzergruppe des Benutzers einer Warteschlange zu oder indem eine in der Warteschlangenkonfiguration mit einer Abfragegruppenbeschriftung, die der Benutzer zur Laufzeit festlegt, aufgeführte Abfragegruppe zugeordnet wird.

> Derzeit wird das automatische WLM standardmäßig für Cluster verwendet, die Standard-Parametergruppen nutzen. Das automatische WLM verwaltet die Abfragegleichzeitigkeit sowie die Speicherzuweisung. Weitere Informationen finden Sie unter Implementieren von automatischem WLM.

Mit manuellem WLM konfiguriert Amazon Redshift eine Warteschlange mit der Gleichzeitigkeitsstufe Fünf, wodurch bis zu fünf Abfragen gleichzeitig ausgeführt werden können, plus eine vordefinierte Superuser-Warteschlange mit der Gleichzeitigkeitsstufe Eins. Sie können bis zu acht Warteschlangen definieren. Jede Warteschlange kann mit einer maximalen Gleichzeitigkeitsstufe von 50 konfiguriert werden. Die maximale Gleichzeitigkeitsstufe für alle benutzerdefinierten Warteschlangen (ohne die Superuser-Warteschlange) ist 50.

Die einfachste Möglichkeit zur Modifizierung der WLM-Konfiguration besteht in der Verwendung der Amazon-Redshift-Managementkonsole. Sie können auch die Amazon-Redshift-Befehlszeilenschnittstelle (Command Line Interface, CLI) oder die Amazon-Redshift-API verwenden.

Weitere Informationen zur Implementierung und Verwendung von Workload-Management finden Sie unter Implementierung von Workload Management.

Verwenden von Amazon Redshift mit sonstigen Services

Amazon Redshift lässt sich in andere AWS Services integrieren, sodass Sie Ihre Daten mithilfe von Datensicherheitsfunktionen schnell und zuverlässig verschieben, transformieren und laden können.

Verschieben von Daten zwischen Amazon Redshift und Amazon S3

Amazon Simple Storage Service (Amazon S3) ist ein Webservice, der Daten in der Cloud speichert. Amazon Redshift nutzt die parallele Verarbeitung zum Lesen und Laden von Daten aus mehreren in Amazon-S3-Buckets gespeicherten Dateien. Weitere Informationen finden Sie unter So laden Sie Daten aus Amazon S3.

Sie können die parallele Verarbeitung auch zum Exportieren von Daten aus Ihrem Data Warehouse von Amazon Redshift in mehrere Datendateien auf Amazon S3 verwenden. Weitere Informationen finden Sie unter Entfernen von Daten.

Verwenden von Amazon Redshift mit Amazon DynamoDB

Amazon DynamoDB ist ein vollständig verwalteter NoSQL-Datenbankservice. Sie können den COPY-Befehl verwenden, um eine Amazon-Redshift-Tabelle mit Daten aus einer einzelnen Amazon-DynamoDB-Tabelle zu laden. Weitere Informationen finden Sie unter Laden von Daten aus einer Amazon-DynamoDB-Tabelle.

Importieren von Daten aus Remote-Hosts über SSH

Sie können den Befehl COPY in Amazon Redshift verwenden, um Daten aus einem oder mehreren Remote-Hosts wie Amazon-EMR-Clustern, Amazon-EC2-Instances oder anderen Computern zu laden. COPY stellt über SSH eine Verbindung zu den Remote-Hosts her und führt Befehle auf den Remote-Hosts aus, um Daten zu generieren. Amazon Redshift unterstützt mehrere gleichzeitige Verbindungen. Der COPY-Befehl liest und lädt die Ausgabe von mehreren Host-Quellen in paralleler Weise. Weitere Informationen finden Sie unter Laden von Daten aus Remote-Hosts.

Automatisieren von Datenladungen mit AWS Data Pipeline

Sie können AWS Data Pipeline damit die Übertragung und Transformation von Daten in und aus Amazon Redshift automatisieren. Mithilfe der integrierten Planungsfunktionen von können Sie wiederkehrende Jobs planen und ausführen AWS Data Pipeline, ohne Ihre eigene komplexe Datenübertragungs- oder Transformationslogik schreiben zu müssen. Beispielsweise können Sie

einrichten, dass ein wiederkehrender Auftrag automatisch Daten von Amazon DynamoDB in Amazon Redshift kopiert. Ein Tutorial, das Sie durch den Prozess der Erstellung einer Pipeline führt, die regelmäßig Daten von Amazon S3 nach Amazon Redshift verschiebt, finden Sie unter Daten nach Amazon Redshift kopieren AWS Data Pipeline im AWS Data Pipeline Entwicklerhandbuch.

Daten mithilfe von () migrieren AWS Database Migration ServiceAWS DMS

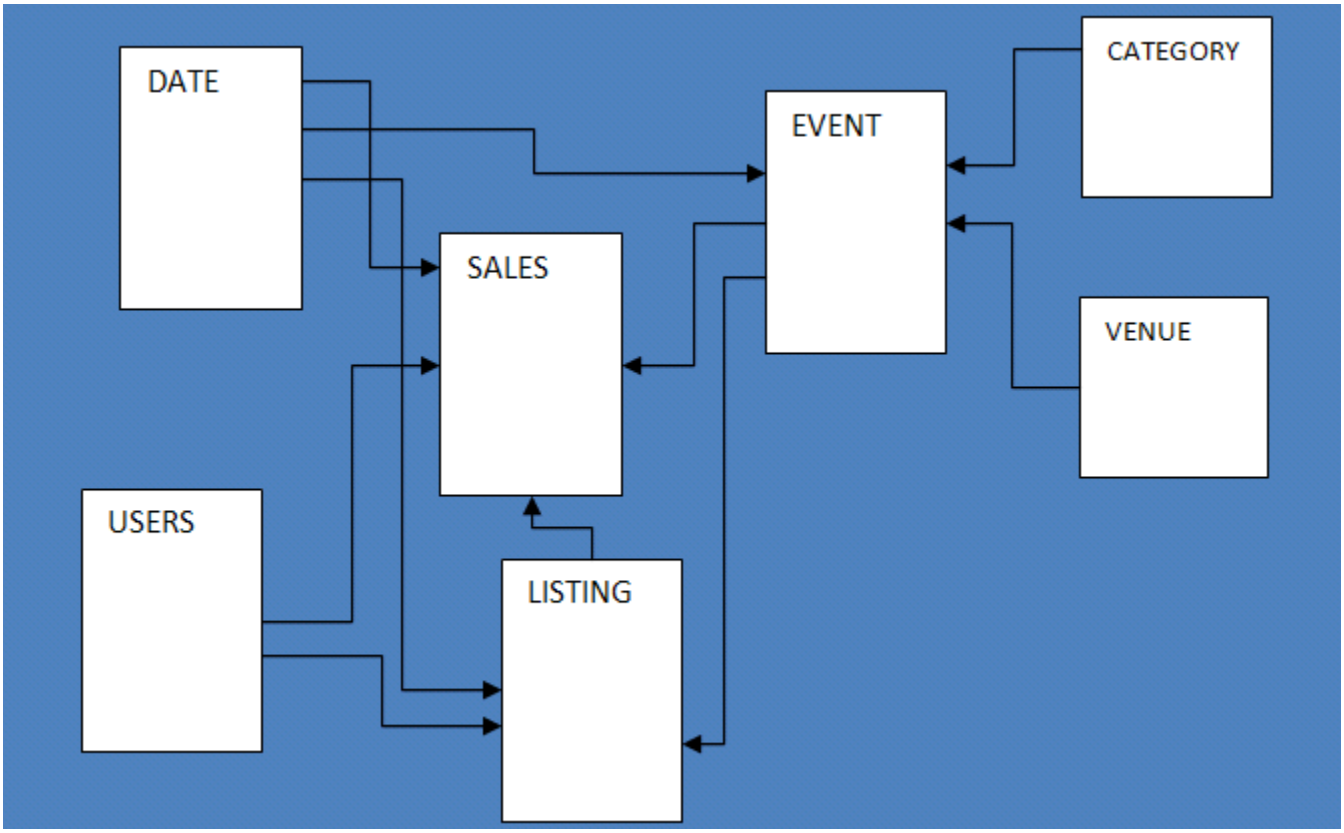
Sie können Daten mit AWS Database Migration Service zu Amazon Redshift migrieren. AWS DMS kann Ihre Daten zu und von den am häufigsten verwendeten kommerziellen und Open-Source-Datenbanken wie Oracle, PostgreSQL, Microsoft SQL Server, Amazon Redshift, Aurora DB-Cluster, DynamoDB, Amazon S3, MariaDB und MySQL migrieren. Weitere Informationen finden Sie unter Verwenden einer Amazon-Redshift-Datenbank als Ziel für AWS Database Migration Service.

Beispieldatenbank

Themen

- Tabelle CATEGORY
- Tabelle DATE
- Tabelle EVENT
- Tabelle VENUE
- Tabelle USERS
- Tabelle LISTING
- Tabelle SALES

Die meisten Beispiele in der Amazon-Redshift-Dokumentation verwenden eine Beispieldatenbank mit dem Namen TICKIT. Diese kleine Datenbank enthält sieben Tabellen, zwei Faktentabellen und fünf Dimensionen. Sie können den TICKIT-Datensatz laden, indem Sie den Schritten in Schritt 4: Daten von Amazon S3 nach Amazon Redshift laden im Amazon Redshift Getting Started Guide folgen.



Diese Beispieldatenbankanwendung unterstützt Analysten dabei, die Verkaufsaktivitäten auf einer fiktionalen TICKIT-Website nachzuverfolgen, auf der Benutzer online Tickets für Sportereignisse, Shows und Konzerte kaufen und verkaufen. Analysten können damit insbesondere die Bewegungen von Tickets über die Zeit identifizieren, sowie Abschlussraten für Verkäufer und die Ereignisse, Austragungsorte und Spielzeiten ermitteln, die sich am besten verkaufen. Analysten können anhand dieser Informationen Käufern und Verkäufern auf dieser Website Anreize geben, um neue Benutzer anzuziehen und Anzeigen und Werbeaktionen zu fördern.

Die folgende Abfrage ermittelt beispielsweise die Top 5-Verkäufer in San Diego, ausgehend von der Anzahl der 2008 verkauften Tickets:

```
select sellerid, username, (firstname ||' '|| lastname) as name,
city, sum(qtysold)
from sales, date, users
where sales.sellerid = users.userid
and sales.dateid = date.dateid
and year = 2008
and city = 'San Diego'
group by sellerid, username, name, city
order by 5 desc
```

```
limit 5;

sellerid | username |       name        |   city    | sum
---------+----------+-------------------+-----------+-----
49977 | JJK84WTE | Julie Hanson      | San Diego |  22
19750 | AAS23BDR | Charity Zimmerman | San Diego |  21
29069 | SVL81MEQ | Axel Grant        | San Diego |  17
43632 | VAG08HKW | Griffin Dodson    | San Diego |  16
36712 | RXT40MKU | Hiram Turner      | San Diego |  14
(5 rows)
```

Die für die Beispiele in diesem Handbuch verwendete Datenbank enthält nur kleine Datenbestände. Die beiden Faktentabellen enthalten jeweils weniger als 200.000 Zeilen, und die Dimensionen bewegen sich in einem Bereich von 11 Zeilen in der Tabelle CATEGORY bis etwa 50.000 Zeilen in der Tabelle USERS.

Die Datenbankbeispiele in diesem Handbuch sollen insbesondere zentrale Funktionen beim Entwurf von Amazon-Redshift-Tabellen demonstrieren:

- Datenverteilung
- Datensortierung
- Spaltenkompression

Tabelle CATEGORY

Spaltenname	Datentyp	Beschreibung
CATID	SMALLINT	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jede Zeile stellt eine spezifische Art von Event dar, für die Tickets gekauft und verkauft werden.
CATGROUP	VARCHAR(10)	Beschreibender Name für eine Gruppe von Events, beispielsweise **`Shows`** oder **`Sports`**.
CATNAME	VARCHAR(10)	Kurzer beschreibender Name für eine Art von Events innerhalb einer Gruppe, beispielsweise **`Opera`** oder **`Musicals`**.

Spaltenname	Datentyp	Beschreibung
CATDESC	VARCHAR(50)	Längerer beschreibender Name für eine Art von Events, beispielsweise **`Musical theatre`**.

Tabelle DATE

Spaltenname	Datentyp	Beschreibung
DATEID	SMALLINT	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jedes Zeile steht für einen Tag im Kalenderjahr.
CALDATE	DATUM	Kalenderdatum, beispielsweise **`2008-06-24`** .
DAY	CHAR(3)	Wochentag (abgekürzt), beispielsweise **`SA`**.
WEEK	SMALLINT	Nummer der Kalenderwoche, beispielsweise **`26`**.
MONTH	CHAR(5)	Monatsname (abgekürzt), beispielsweise **`JUN`**.
QTR	CHAR(5)	Quartalsnummer (**`1`** bis **`4`**).
YEAR	SMALLINT	Das vierstellige Jahr (**`2008`**).
HOLIDAY	BOOLEAN	Flag zur Angabe, ob der Tag ein öffentlicher Feiertag (in den USA) ist.

Tabelle EVENT

Spaltenname	Datentyp	Beschreibung
EVENTID	INTEGER	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jede Zeile stellt ein eigenes Event dar, das an einem bestimmten Ort und zu einer bestimmten Zeit stattfindet.
VENUEID	SMALLINT	Fremdschlüsselverweis auf die VENUE-Tabelle.

Spaltenname	Datentyp	Beschreibung
CATID	SMALLINT	Fremdschlüsselverweis auf die CATEGORY-Tabelle.
DATEID	SMALLINT	Fremdschlüsselverweis auf die DATE-Tabelle.
EVENTNAME	VARCHAR(200)	Name des Ereignisses, etwa **`Hamlet`** oder **`La Traviata`**.
STARTTIME	TIMESTAMP	Vollständige Angabe von Datum und Uhrzeit, beispielsweise **`2008-10-10 19:30:00`** .

Tabelle VENUE

Spaltenname	Datentyp	Beschreibung
VENUEID	SMALLINT	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jede Zeile stellt einen eigenen Veranstaltungsort dar, an dem Events stattfinden.
VENUENAME	VARCHAR(100)	Exakter Name des Ortes, beispielsweise **`Cleveland Browns Stadium`**.
VENUECITY	VARCHAR(30)	Name des Ortes, beispielsweise **`Cleveland`** .
VENUESTATE	CHAR(2)	Abkürzung des Bundesstaats bzw. der Provinz (in den USA bzw. in Kanada), beispielsweise **`OH`**.
VENUESEATS	INTEGER	Maximale Anzahl verfügbarer Plätze an dem Veranstaltungsort (falls bekannt), beispielsweise **`73200`**. Diese Spalte enthält zur Illustration von Sonderfällen auch Nullwerte und Nullen.

Tabelle USERS

Spaltenname	Datentyp	Beschreibung
USERID	INTEGER	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jede Zeile stellt einen registrierten Benutzer (Käufer, Verkäufer oder beides) dar, der Tickets für mindestens ein Event angeboten oder gekauft hat.
Die Datei „snowball-adapter.config“ enthält die Konfigura tionseins tellungen für den Adapter.	CHAR(8)	Ein 8 Zeichen langer, alphanumerischer Benutzern ame, beispielsweise **PGL08LJI**.
FIRSTNAME	VARCHAR(30)	Der Vorname des Benutzers, beispielsweise **Victor**.
LASTNAME	VARCHAR(30)	Der Nachname des Benutzers, beispielsweise **Hernandez** .
CITY	VARCHAR(30)	Der Heimatort Benutzers, beispielsweise **Napervill e** .
STATE	CHAR(2)	Der Bundesstaat aus der Heimatanschrift des Benutzers, beispielsweise **GA**.
EMAIL	VARCHAR(100)	Die E-Mail-Adresse des Benutzers; diese Spalte enthält zufällige lateinschriftliche Werte, wie etwa **turpis@accumsanlaoreet.org** .
PHONE	CHAR(14)	Die 14 Zeichen lange Telefonnummer des Benutzers, beispielsweise **(818) 765-4255**.

Spaltenname	Datentyp	Beschreibung
LIKESPORTS, ...	BOOLEAN	Eine Reihe von 10 verschiedenen Spalten zur Angabe der Benutzerpräferenzen bei Events, mit möglichen Werten **`true`** oder **`false`**,

Tabelle LISTING

Spaltenname	Datentyp	Beschreibung
LISTID	INTEGER	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jede Zeile stellt ein Listing für ein Batch an Tickets für ein spezifisches Event dar.
SELLERID	INTEGER	Fremdschlüsselverweis auf die USERS-Tabelle, über den der Benutzer identifiziert wird, der die Tickets verkauft.
EVENTID	INTEGER	Fremdschlüsselverweis auf die EVENT-Tabelle.
DATEID	SMALLINT	Fremdschlüsselverweis auf die DATE-Tabelle.
NUMTICKETS	SMALLINT	Die Anzahl der zum Verkauf stehenden Tickets, beispielsweise **`2`** oder **`20`**.
PRICEPERTICKET	DECIMAL(8,2)	Der Festpreis für ein einzelnes Ticket, beispielsweise **`27.00`** oder **`206.00`**.
TOTALPRICE	DECIMAL(8,2)	Der Gesamtwert für dieses Listing (NUMTICKETS*PRICEPERTICKET).
LISTTIME	TIMESTAMP	Vollständige Angabe von Datum und Uhrzeit, zu der das Listing gebucht wurde, beispielsweise **`2008-03-18 07:19:35`** .

Tabelle SALES

Spaltenname	Datentyp	Beschreibung
SALESID	INTEGER	Primärschlüssel, eine eindeutige Kennung für jede Zeile. Jede Zeile stellt eine einen Verkauf von einem oder mehreren Tickets für ein bestimmtes Event dar, wie in einem spezifischen Listing angeboten.
LISTID	INTEGER	Fremdschlüsselverweis auf die LISTING-Tabelle.
SELLERID	INTEGER	Fremdschlüsselverweis auf die USERS-Tabelle (der Benutzer, der die Tickets verkauft hat).
BUYERID	INTEGER	Fremdschlüsselverweis auf die USERS-Tabelle (der Benutzer, der die Tickets gekauft hat).
EVENTID	INTEGER	Fremdschlüsselverweis auf die EVENT-Tabelle.
DATEID	SMALLINT	Fremdschlüsselverweis auf die DATE-Tabelle.
QTYSOLD	SMALLINT	Die Anzahl der verkauften Tickets, ein Wert zwischen **1** und **8**. (Es können pro Transaktion maximal 8 Tickets verkauft werden).
PRICEPAID	DECIMAL(8,2)	Der Gesamtpreis, der für die Tickets bezahlt wurde, beispielsweise **75.00** oder **488.00**. Der Preis für ein einzelnes Ticket berechnet sich als PRICEPAID/ QTYSOLD.
COMMISSION	DECIMAL(8,2)	Der Betrag der 15 % Kommission, den das Geschäft aus dem Verkauf einnimmt, beispielsweise **11.25** oder **73.20**. Der Verkäufer erhält 85 % des Betrags in PRICEPAID.
SALETIME	TIMESTAMP	Vollständige Angabe von Datum und Uhrzeit, zu der der Verkauf abgeschlossen wurde, beispielsweise **2008-05-24 06:21:47** .

Bewährte Methoden für Amazon Redshift

Anschließend können Sie bewährte Methoden zur Planung eines Machbarkeitsnachweises, zum Gestalten von Tabellen, Laden von Daten in Tabellen und Verfassen von Abfragen für Amazon Redshift sowie eine Erörterung der Arbeit mit Amazon Redshift Advisor suchen.

Amazon Redshift unterscheidet sich von anderen SQL-Datenbanksystemen. Um alle Vorteile der Amazon-Redshift-Architektur vollständig zu nutzen, müssen Sie Ihre Tabellen dafür gestalten, erstellen und laden, dass sie die massive parallele Verarbeitung, die Spaltendatenspeicherung und die Spaltendatenkomprimierungen nutzen können. Wenn Ihre Datenlade- und Abfrageausführungszeiten länger sind als erwartet oder erwünscht, kann es sein, dass Sie wichtige Informationen übersehen.

Wenn Sie ein erfahrener SQL-Datenbankentwickler sind, sollten Sie unbedingt dieses Thema lesen, bevor Sie mit der Entwicklung Ihres Data Warehouse von Amazon Redshift beginnen.

Wenn Sie Anfänger bei der Entwicklung von SQL-Datenbanken sind, sollten Sie nicht mit diesem Thema beginnen. Wir empfehlen Ihnen, zunächst Allgemeine Datenbankaufgaben zu lesen und die Beispiele selbst auszuprobieren.

In diesem Thema finden Sie eine Übersicht über die wichtigsten Entwicklungsprinzipien und spezielle Tipps, Beispiele und bewährte Methoden für deren Implementierung. Keine einzelne Methode gilt für jede Anwendung. Prüfen Sie alle Ihre Optionen, bevor Sie ein Datenbankdesign abschließen. Weitere Informationen finden Sie unter Arbeiten mit automatischer Tabellenoptimierung, Laden von Daten, Optimieren der Abfrageleistung und in den Referenzkapiteln.

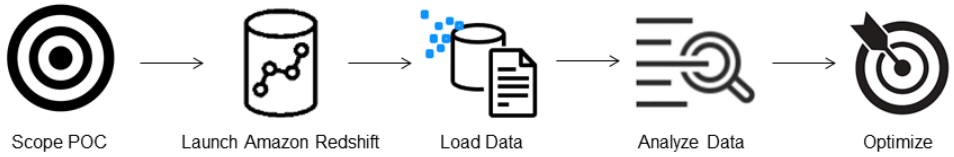
Themen

Führen Sie einen Machbarkeitsnachweis (POC) für Amazon Redshift durch

Amazon Redshift ist ein beliebtes Cloud-Data Warehouse, das einen vollständig verwalteten Cloud-basierten Service bietet, der in den Amazon Simple Storage Service Data Lake eines Unternehmens, Echtzeit-Streams, maschinelles Lernen (ML), Transaktionsworkflows und vieles mehr integriert werden kann. Die folgenden Abschnitte führen Sie durch den Prozess der Durchführung eines Machbarkeitsnachweises (PoC) auf Amazon Redshift. Die Informationen hier helfen Ihnen bei der Festlegung von Zielen für Ihren POC und nutzen die Vorteile von Tools, mit denen Sie die Bereitstellung und Konfiguration von Services für Ihren POC automatisieren können.

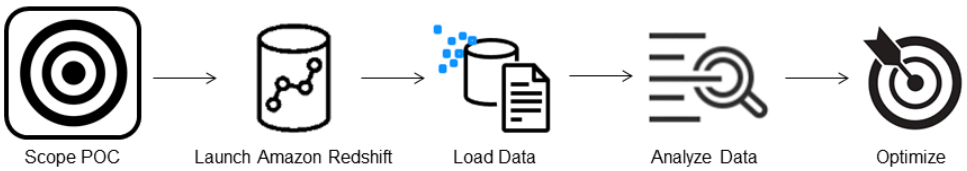
> **Note**
>
> Um eine Kopie dieser Informationen als PDF zu erhalten, wählen Sie auf der Amazon Redshift-Ressourcenseite den Link Run your own Redshift POC.

Wenn Sie einen POC von Amazon Redshift durchführen, testen, testen und übernehmen Sie Funktionen, die von best-in-class Sicherheitsfunktionen über elastische Skalierung, einfache Integration und Aufnahme bis hin zu flexiblen dezentralen Datenarchitekturoptionen reichen.



Folgen Sie diesen Schritten, um einen erfolgreichen Machbarkeitsnachweis durchzuführen.

Schritt 1: Definieren Sie Ihren POC



Bei der Durchführung eines POC können Sie entweder Ihre eigenen Daten oder Benchmarking-Datensätze verwenden. Wenn Sie Ihre eigenen Daten auswählen, führen Sie Ihre eigenen Abfragen anhand der Daten durch. Bei Benchmarking-Daten werden Beispielabfragen zusammen mit dem Benchmark bereitgestellt. Weitere Informationen finden Sie unter Verwenden von Beispieldatensätzen, falls Sie noch nicht bereit sind, einen POC mit Ihren eigenen Daten durchzuführen.

Im Allgemeinen empfehlen wir, Daten von zwei Wochen für einen Amazon Redshift Redshift-POC zu verwenden.

Gehen Sie zunächst wie folgt vor:

1. Identifizieren Sie Ihre geschäftlichen und funktionalen Anforderungen und arbeiten Sie dann rückwärts. Typische Beispiele sind: schnellere Leistung, geringere Kosten, Testen eines neuen Workloads oder Features oder Vergleich zwischen Amazon Redshift und einem anderen Data Warehouse.
2. Legen Sie spezifische Ziele fest, die zu den Erfolgskriterien für den POC werden. Überlegen Sie sich beispielsweise anhand einer schnelleren Leistung eine Liste der fünf wichtigsten Prozesse, die Sie beschleunigen möchten, und geben Sie die aktuellen Laufzeiten zusammen mit der erforderlichen Laufzeit an. Dabei kann es sich um Berichte, Abfragen, ETL-Prozesse, Datenerfassung oder was auch immer Ihre aktuellen Probleme sind, handeln.
3. Identifizieren Sie den spezifischen Umfang und die Artefakte, die für die Durchführung der Tests erforderlich sind. Welche Datensätze müssen Sie migrieren oder kontinuierlich in Amazon Redshift aufnehmen, und welche Abfragen und Prozesse sind erforderlich, um die Tests durchzuführen, um sie anhand der Erfolgskriterien zu messen? Es gibt zwei Möglichkeiten dafür:

   Bringen Sie Ihre eigenen Daten mit

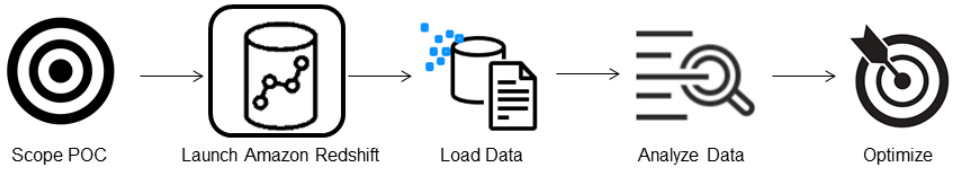
   - Um Ihre eigenen Daten zu testen, erstellen Sie die mindestens praktikable Liste von Datenartefakten, die zum Testen Ihrer Erfolgskriterien erforderlich ist. Wenn Ihr aktuelles Data Warehouse beispielsweise über 200 Tabellen verfügt, die Berichte, die Sie testen möchten, jedoch nur 20 benötigen, kann Ihr POC schneller ausgeführt werden, wenn Sie nur die kleinere Teilmenge von Tabellen verwenden.

   Verwenden Sie Beispieldatensätze

   - Wenn Sie keine eigenen Datensätze bereit haben, können Sie trotzdem mit der Durchführung eines POC auf Amazon Redshift beginnen, indem Sie die branchenüblichen Benchmark-

Datensätze wie TPC-DS oder TPC-H verwenden und Beispiel-Benchmarking-Abfragen ausführen, um die Leistungsfähigkeit von Amazon Redshift zu nutzen. Auf diese Datensätze kann von Ihrem Amazon Redshift Data Warehouse aus zugegriffen werden, nachdem es erstellt wurde. Detaillierte Anweisungen zum Zugriff auf diese Datensätze und Beispielabfragen finden Sie unter. Schritt 2: Amazon Redshift starten

Schritt 2: Amazon Redshift starten



Amazon Redshift sorgt mit schnellem, einfachem und sicherem Cloud-Data Warehousing in großem Maßstab dafür, dass Sie schneller Erkenntnisse gewinnen. Sie können schnell beginnen, indem Sie Ihr Warehouse auf der Redshift Serverless-Konsole starten und innerhalb von Sekunden von Daten zu Erkenntnissen gelangen. Mit Redshift Serverless können Sie sich darauf konzentrieren, Ihre Geschäftsergebnisse zu erzielen, ohne sich Gedanken über die Verwaltung Ihres Data Warehouse machen zu müssen.

Amazon Redshift Serverless einrichten

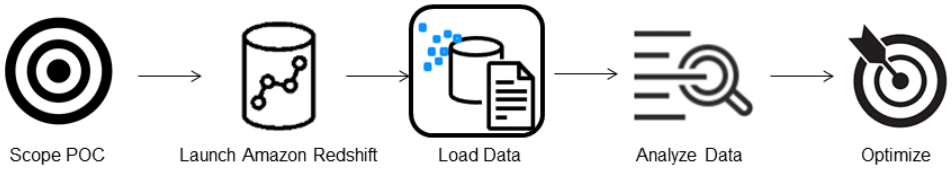
Wenn Sie Redshift Serverless zum ersten Mal verwenden, führt Sie die Konsole durch die Schritte, die zum Starten Ihres Warehouse erforderlich sind. Möglicherweise haben Sie auch Anspruch auf eine Gutschrift für Ihre Redshift Serverless-Nutzung in Ihrem Konto. Weitere Informationen zur Auswahl einer kostenlosen Testversion finden Sie unter Kostenlose Testversion von Amazon Redshift. Folgen Sie den Schritten unter Erstellen eines Data Warehouse mit Redshift Serverless im Amazon Redshift Getting Started Guide, um ein Data Warehouse mit Redshift Serverless zu erstellen. Wenn Sie keinen Datensatz haben, den Sie laden möchten, enthält der Leitfaden auch Schritte zum Laden eines Beispieldatensatzes.

Wenn Sie Redshift Serverless zuvor in Ihrem Konto gestartet haben, folgen Sie den Schritten unter Erstellen einer Arbeitsgruppe mit einem Namespace im Amazon Redshift Management Guide. Sobald Ihr Warehouse verfügbar ist, können Sie sich dafür entscheiden, die in Amazon Redshift verfügbaren Beispieldaten zu laden. Informationen zur Verwendung des Amazon Redshift Query

Editors v2 zum Laden von Daten finden Sie unter Laden von Beispieldaten im Amazon Redshift Management Guide.

Wenn Sie Ihre eigenen Daten mitbringen, anstatt den Beispieldatensatz zu laden, finden Sie unter. Schritt 3: Laden Sie Ihre Daten

Schritt 3: Laden Sie Ihre Daten



Nach dem Start von Redshift Serverless besteht der nächste Schritt darin, Ihre Daten für den POC zu laden. Ganz gleich, ob Sie eine einfache CSV-Datei hochladen, halbstrukturierte Daten aus S3 aufnehmen oder Daten direkt streamen, Amazon Redshift bietet die Flexibilität, die Daten schnell und einfach von der Quelle in Amazon Redshift Redshift-Tabellen zu verschieben.

Wählen Sie eine der folgenden Methoden, um Ihre Daten zu laden.

Laden Sie eine lokale Datei hoch

Für eine schnelle Aufnahme und Analyse können Sie den Amazon Redshift Query Editor v2 verwenden, um Datendateien einfach von Ihrem lokalen Desktop zu laden. Es ist in der Lage, Dateien in verschiedenen Formaten wie CSV, JSON, AVRO, PARQUET, ORC und mehr zu verarbeiten. Damit Ihre Benutzer als Administrator Daten mit dem Abfrage-Editor v2 von einem lokalen Desktop laden können, müssen Sie einen gemeinsamen Amazon S3 S3-Bucket angeben und das Benutzerkonto muss mit den entsprechenden Berechtigungen konfiguriert sein. Sie können das einfache und sichere Laden von Daten in Amazon Redshift verfolgen, indem Sie Query Editor V2 als step-by-step Anleitung verwenden.

Eine Amazon S3 S3-Datei laden

Um Daten aus einem Amazon S3 S3-Bucket in Amazon Redshift zu laden, verwenden Sie zunächst den Befehl COPY und geben Sie den Amazon S3 S3-Quellspeicherort und die Amazon Redshift Redshift-Zieltabelle an. Stellen Sie sicher, dass die IAM-Rollen und -Berechtigungen ordnungsgemäß konfiguriert sind, sodass Amazon Redshift auf den angegebenen Amazon S3 S3-Bucket zugreifen

kann. Folgen Sie der Anleitung Tutorial: Daten aus Amazon S3 laden. step-by-step Sie können auch die Option Daten laden im Abfrage-Editor v2 wählen, um Daten direkt aus Ihrem S3-Bucket zu laden.

Kontinuierliche Datenaufnahme

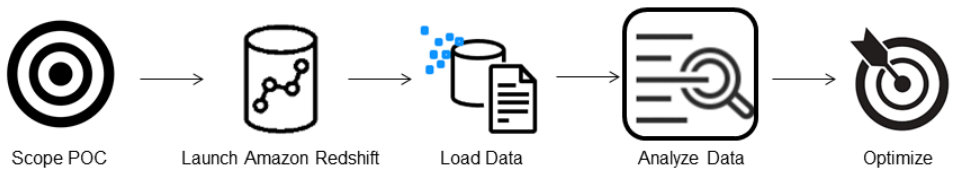
Autocopy (in der Vorschauversion) ist eine Erweiterung des COPY-Befehls und automatisiert das kontinuierliche Laden von Daten aus Amazon S3 S3-Buckets. Wenn Sie einen Kopierauftrag erstellen, erkennt Amazon Redshift, wenn neue Amazon S3 S3-Dateien in einem angegebenen Pfad erstellt werden, und lädt sie dann automatisch, ohne dass Sie eingreifen müssen. Amazon Redshift verfolgt die geladenen Dateien, um sicherzustellen, dass sie nur einmal geladen werden. Anweisungen zum Erstellen von Kopieraufträgen finden Sie unter COPY JOB (Vorschau)

> **Note**
>
> Autocopy befindet sich derzeit in der Vorschauversion und wird nur in bestimmten bereitgestellten Clustern unterstützt. AWS-Regionen Informationen zum Erstellen eines Vorschau-Clusters für Autocopy finden Sie unter. Kontinuierliche Dateierfassung von Amazon S3 (Vorschau)

Laden Sie Ihre Streaming-Daten

Die Streaming-Aufnahme ermöglicht die Aufnahme von Stream-Daten aus Amazon Kinesis Data Streams und Amazon Managed Streaming for Apache Kafka mit niedriger Latenz und hoher Geschwindigkeit in Amazon Redshift. Die Amazon Redshift Redshift-Streaming-Aufnahme verwendet eine materialisierte Ansicht, die mithilfe der auto Aktualisierung direkt aus dem Stream aktualisiert wird. Die materialisierte Ansicht wird der Stream-Datenquelle zugeordnet. Sie können die Stream-Daten als Teil der Definition der materialisierten Ansicht filtern und aggregieren. step-by-step Anleitungen zum Laden von Daten aus einem Stream finden Sie unter Erste Schritte mit Amazon Kinesis Data Streams oder Erste Schritte mit Amazon Managed Streaming for Apache Kafka.

Schritt 4: Analysieren Sie Ihre Daten



Nachdem Sie Ihre Redshift Serverless-Arbeitsgruppe und Ihren Namespace erstellt und Ihre Daten geladen haben, können Sie Abfragen sofort ausführen, indem Sie den Abfrage-Editor v2 im Navigationsbereich der Redshift Serverless-Konsole öffnen. Sie können den Abfrage-Editor v2 verwenden, um die Abfragefunktionalität oder die Abfrageleistung anhand Ihrer eigenen Datensätze zu testen.

Abfragen mit dem Amazon Redshift Redshift-Abfrage-Editor v2

Sie können über die Amazon Redshift Redshift-Konsole auf den Abfrage-Editor v2 zugreifen. Eine vollständige Anleitung zur Konfiguration, Verbindung und Ausführung von Abfragen mit dem Abfrage-Editor v2 finden Sie unter Vereinfachen Sie Ihre Datenanalyse mit dem Amazon Redshift Query Editor v2.

Wenn Sie einen Auslastungstest als Teil Ihres POC ausführen möchten, können Sie dies alternativ mit den folgenden Schritten tun, um Apache JMeter zu installieren und auszuführen.

Führen Sie einen Auslastungstest mit Apache JMeter durch

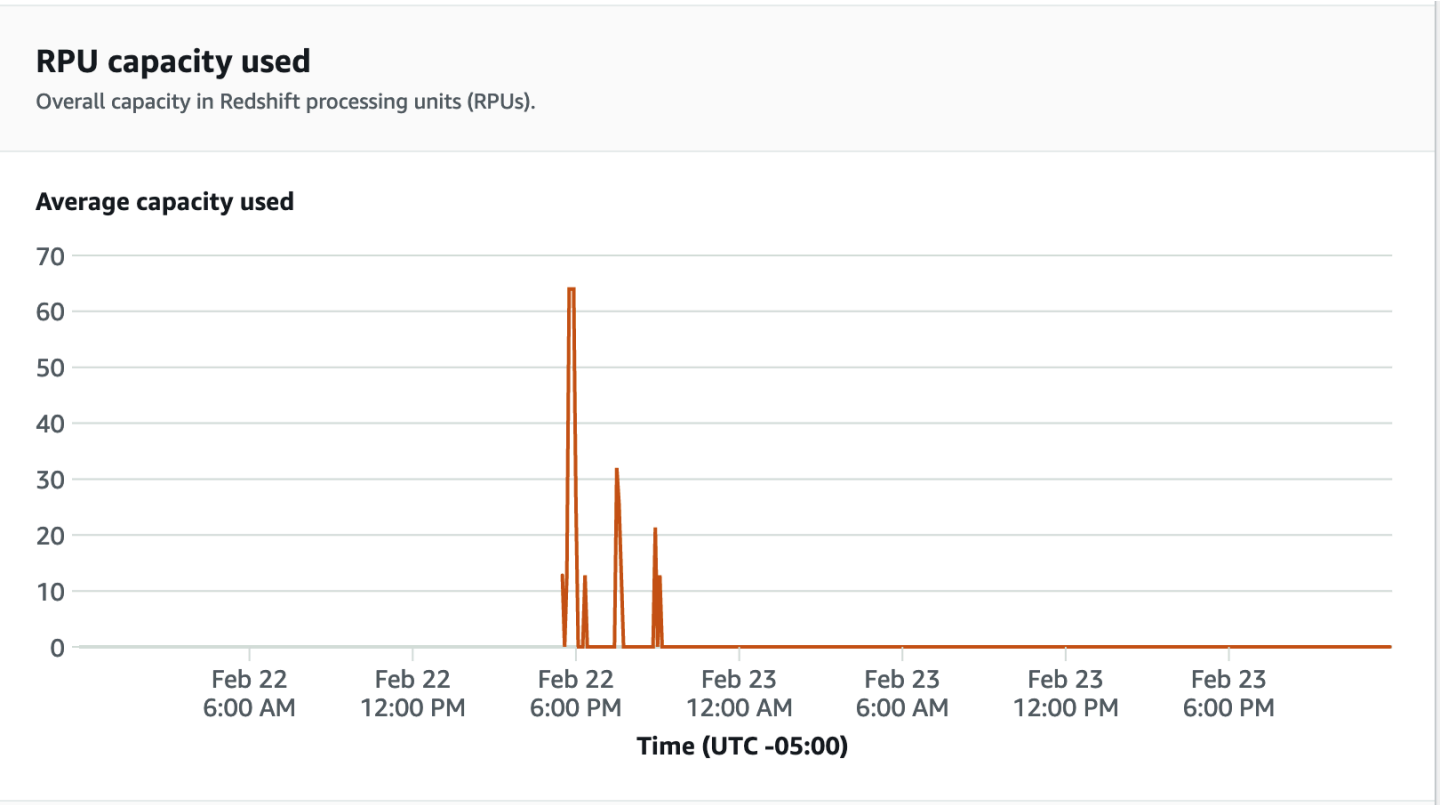
Um einen Auslastungstest durchzuführen, um „N“ Benutzer zu simulieren, die gleichzeitig Anfragen an Amazon Redshift senden, können Sie Apache JMeter verwenden, ein Open-Source-Tool auf Java-Basis.

Um Apache JMeter für die Ausführung in Ihrer Redshift Serverless-Arbeitsgruppe zu installieren und zu konfigurieren, folgen Sie den Anweisungen unter Automatisieren von Amazon Redshift Redshift-Lasttests mit dem Analytics Automation Toolkit. AWS Es verwendet das AWS Analytics Automation Toolkit (AAA), ein Open-Source-Hilfsprogramm für die dynamische Bereitstellung von Redshift-Lösungen, um diese Ressourcen automatisch zu starten. Wenn Sie Ihre eigenen Daten in Amazon Redshift geladen haben, stellen Sie sicher, dass Sie die Option Schritt #5 — SQL anpassen ausführen, um sicherzustellen, dass Sie die entsprechenden SQL-Anweisungen angeben, die Sie anhand Ihrer Tabellen testen möchten. Testen Sie jede dieser SQL-Anweisungen einmal mit dem Abfrage-Editor v2, um sicherzustellen, dass sie fehlerfrei ausgeführt werden.

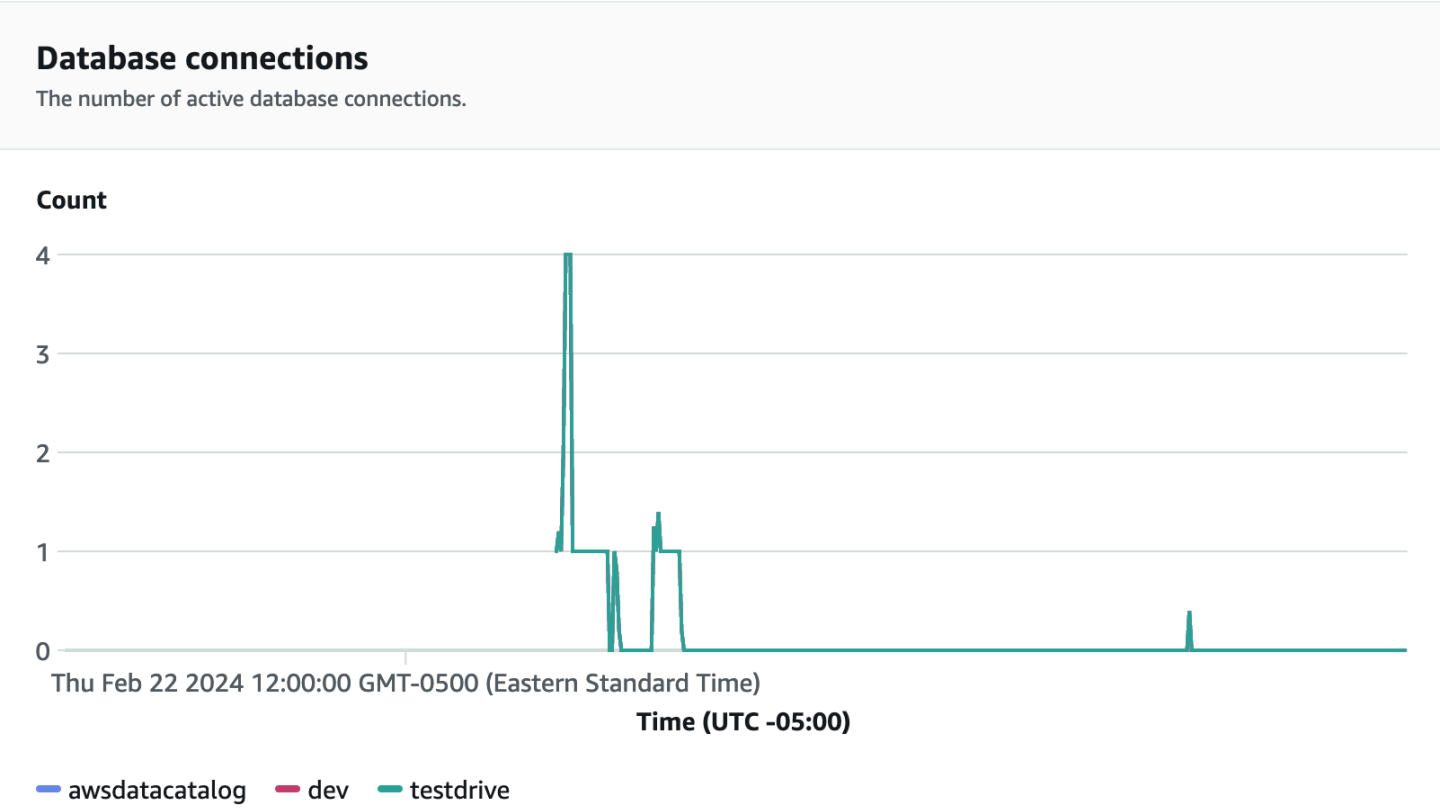
Nachdem Sie die Anpassung Ihrer SQL-Anweisungen und die Fertigstellung Ihres Testplans abgeschlossen haben, speichern Sie Ihren Testplan und führen Sie ihn für Ihre Redshift Serverless-Arbeitsgruppe aus. Um den Fortschritt Ihres Tests zu überwachen, öffnen Sie die Redshift Serverless-Konsole, navigieren Sie zu Abfrage- und Datenbanküberwachung, wählen Sie die Registerkarte Abfrageverlauf und sehen Sie sich Informationen zu Ihren Abfragen an.

Wählen Sie für Leistungsmetriken die Registerkarte Datenbankleistung auf der Redshift Serverless-Konsole, um Metriken wie Datenbankverbindungen und CPU-Auslastung zu überwachen. Hier

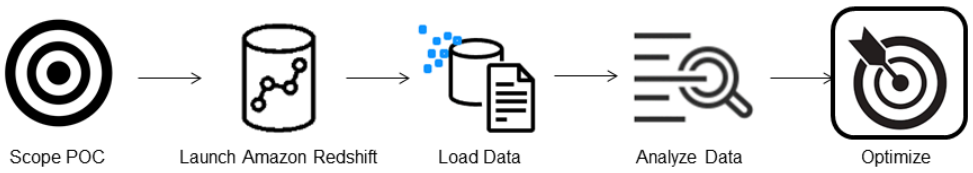
können Sie sich ein Diagramm ansehen, um die verwendete RPU-Kapazität zu überwachen und zu beobachten, wie Redshift Serverless automatisch skaliert, um gleichzeitigen Workload-Anforderungen gerecht zu werden, während der Auslastungstest für Ihre Arbeitsgruppe ausgeführt wird.



Datenbankverbindungen sind eine weitere nützliche Metrik, die Sie während der Ausführung des Auslastungstests überwachen können, um zu sehen, wie Ihre Arbeitsgruppe zahlreiche gleichzeitige Verbindungen zu einem bestimmten Zeitpunkt verarbeitet, um den steigenden Arbeitslastanforderungen gerecht zu werden.



Schritt 5: Optimieren



Amazon Redshift ermöglicht es Zehntausenden von Benutzern, täglich Exabyte an Daten zu verarbeiten und ihre Analyse-Workloads zu optimieren, indem es eine Vielzahl von Konfigurationen und Funktionen zur Unterstützung individueller Anwendungsfälle bietet. Bei der Wahl zwischen diesen Optionen suchen Kunden nach Tools, mit denen sie die optimale Data Warehouse-Konfiguration zur Unterstützung ihrer Amazon Redshift Redshift-Workloads ermitteln können.

Probefahrt

Sie können Test Drive verwenden, um Ihren vorhandenen Workload anhand potenzieller Konfigurationen automatisch abzuspielen und die entsprechenden Ergebnisse zu analysieren, um das optimale Ziel für die Migration Ihres Workloads zu ermitteln. Informationen zur Verwendung von

Test Drive zur Evaluierung verschiedener Amazon Redshift Redshift-Konfigurationen finden Sie unter Finden der besten Amazon Redshift-Konfiguration für Ihren Workload mithilfe von Redshift Test Drive.

Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift

Wenn Sie Ihre Datenbank planen, müssen Sie bestimmte wichtige Tabellengestaltungsentscheidungen treffen, die erheblichen Einfluss auf die allgemeine Abfrageleistung haben. Diese Entscheidungen haben auch große Auswirkungen auf die Speicheranforderungen, die wiederum die Abfrageanforderungen beeinflussen, indem sie die Anzahl der I/O-Operationen reduzieren und den für die Verarbeitung von Abfragen erforderlichen Speicherplatz minimieren.

In diesem Abschnitt finden Sie eine Übersicht der wichtigsten Gestaltungsentscheidungen und bewährte Methoden für die Optimierung der Abfrageleistung. Arbeiten mit automatischer Tabellenoptimierung bietet ausführlichere Erläuterungen und Beispiele für die Tabellendesignoptionen.

Themen

- Auswahl des besten Sortierschlüssels
- Auswahl des besten Verteilungsstils
- Lassen Sie COPY die Kompressionskodierungen auswählen
- Definition von Primär- und Fremdschlüsseleinschränkungen
- Verwendung der geringstmöglichen Spaltengröße
- Verwenden von Datum-/Uhrzeit-Datentypen für Datumsspalten

Auswahl des besten Sortierschlüssels

Amazon Redshift speichert die Daten auf der Festplatte in sortierter Form gemäß dem Sortierschlüssel. Der Amazon-Redshift-Abfrageoptimierer verwendet die Sortierfolge bei der Bestimmung optimaler Abfragepläne.

> **Note**
>
> Wenn Sie die automatische Tabellenoptimierung verwenden, müssen Sie den Sortierschlüssel Ihrer Tabelle nicht auswählen. Weitere Informationen finden Sie unter Arbeiten mit automatischer Tabellenoptimierung.

Einige Vorschläge für den besten Ansatz folgen:

- Damit Amazon Redshift die entsprechende Sortierreihenfolge auswählen kann, geben Sie `AUTO` für den Sortierschlüssel an.
- Wenn aktuelle Daten am häufigsten abgefragt werden, geben Sie die Zeitstempelspalte als führende Spalte für den Sortierschlüssel an.

  Die Abfragen sind effizienter, weil sie gesamte Blöcke überspringen können, die außerhalb des Zeitraums liegen.
- Wenn Sie häufig Bereichsfilterungen oder Gleichheitsfilterungen für eine Spalte durchführen, geben Sie diese Spalte als Sortierschlüssel an.

  Amazon Redshift kann das Lesen gesamter Datenblöcke für diese Spalte überspringen. Dies ist möglich, weil es die minimalen und maximalen in jedem Block gespeicherten Spaltenwerte verfolgt und Blöcke überspringen kann, die dem Prädikatsbereich nicht entsprechen.
- Wenn Sie häufig eine Tabelle zusammenführen (Join), geben Sie die Join-Tabelle als Sortierschlüssel und Verteilungsschlüssel an.

  Dadurch kann der Abfrageoptimierer eine Sortier-/Zusammenführungs-Join-Operation anstelle einer langsameren Hash-Join-Operation wählen. Da die Daten bereits auf dem Join-Schlüssel sortiert sind, kann der Abfrageoptimierer die Sortierungsphase der Sortier-/Zusammenführungs-Join-Operation übergehen.

Auswahl des besten Verteilungsstils

Wenn sie eine Abfrage ausführen, führt der Abfrageoptimierer nach Bedarf eine Neuverteilung der Zeilen zu den Datenverarbeitungsknoten durch, um Join- oder Aggregierungsoperationen durchführen zu können. Das Ziel der Auswahl eines Tabellenverteilungsstils besteht darin, die Auswirkungen des Neuverteilungsschritts dadurch zu minimieren, dass die Daten dort platziert werden, wo sie benötigt werden, bevor die Abfrage ausgeführt wird.

> **Note**
>
> Wenn Sie die automatische Tabellenoptimierung verwenden, müssen Sie den Verteilungsstil Ihrer Tabelle nicht auswählen. Weitere Informationen finden Sie unter Arbeiten mit automatischer Tabellenoptimierung.

Einige Vorschläge für den besten Ansatz folgen:

1. Verteilen der Faktentabelle und einer Dimensionstabelle auf ihren gemeinsamen Spalten.

   Ihre Faktentabelle kann nur einen Verteilungsschlüssel haben. Alle Tabellen, die auf einem anderen Schlüssel verbunden werden, werden nicht mit der Faktentabelle zusammengestellt. Wählen Sie eine Dimension für die Zusammenstellung auf der Grundlage der Join-Häufigkeit und der Größe der Joining-Zeilen. Bezeichnen Sie den Primärschlüssel der Dimensionstabelle und den entsprechenden Fremdschlüssel der Faktentabelle als DISTKEY.
2. Wählen Sie die größte Dimension auf der Grundlage der Größe des gefilterten Datasets.

   Nur die in dem Join verwendeten Zeilen müssen verteilt werden. Berücksichtigen Sie daher die Größe des Datensatzes nach der Filterung und nicht die Größe der Tabelle.
3. Wählen Sie eine Spalte mit hoher Kardinalität im gefilterten Ergebnissatz.

   Wenn Sie beispielsweise eine Vertriebstabelle auf einer Datumsspalte verteilen, erhalten Sie wahrscheinlich eine ziemlich gleichmäßige Datenverteilung, wenn nicht der Großteil Ihres Vertriebs saisongebunden ist. Wenn Sie jedoch üblicherweise ein bereichseingeschränktes Prädikat für die Filterung für einen engen Zeitbereich verwenden, befinden sich die meisten der gefilterten Zeilen auf einem begrenzten Satz von Slices, und der Abfrageworkload wird verzerrt.
4. Ändern einiger Dimensionstabellen, um die ALL-Verteilung zu verwenden.

   Wenn eine Dimensionstabelle nicht mit der Faktentabelle oder anderen wichtigen Joining-Tabellen zusammengestellt werden kann, können Sie die Abfrageleistung dadurch erheblich verbessern, dass Sie die gesamte Tabelle zu allen Knoten verteilen. Die Verwendung der ALL-Verteilung vervielfacht die Speicheranforderungen, verlängert Ladezeiten und erhöht den Aufwand für Wartungsoperationen. Sie sollten daher alle Faktoren sorgfältig abwägen, bevor Sie die ALL-Verteilung wählen.

Wenn Amazon Redshift den geeigneten Verteilungsstil auswählt, geben Sie AUTO für den Verteilungsstil an.

Für weitere Informationen zur Auswahl von Verteilungsstilen vgl. Arbeiten mit Datenverteilungsstilen.

Lassen Sie COPY die Kompressionskodierungen auswählen

Sie können Kompressionskodierungen angeben, wenn Sie eine Tabelle erstellen, in den meisten Fällen führt jedoch die automatische Kompression zu den besten Ergebnissen.

ENCODE AUTO ist die Standardeinstellung für Tabellen. Wenn für eine Tabelle ENCODE AUTO festgelegt wird, verwaltet Amazon Redshift automatisch die Kompressionskodierung für alle Spalten in der Tabelle. Weitere Informationen erhalten Sie unter CREATE TABLE und ALTER TABLE.

Der COPY-Befehl analysiert Ihre Daten und wendet die Kompressionskodierungen automatisch auf eine leere Tabelle im Rahmen der Ladeoperation an.

Die automatische Kompression sorgt für den Ausgleich der allgemeinen Leistung bei der Auswahl von Kompressionskodierungen. Bereichseingeschränkte Scans können eine geringe Leistung zeigen, wenn Sortierschlüsselspalten stärker komprimiert sind als andere Spalten in der selben Abfrage. Daher wählt die automatische Kompression eine weniger effiziente Kompressionskodierung, um den Ausgleich zwischen der Sortierschlüsselspalte und den anderen Spalten zu wahren.

Angenommen, der Sortierschlüssel Ihrer Tabelle ist ein Datum oder ein Zeitstempel und die Tabelle verwendet viele große varchar-Spalten. In diesem Fall erzielen Sie möglicherweise eine bessere Leistung, wenn Sie die Sortierschlüsselspalte überhaupt nicht komprimieren. Führen Sie den Befehl ANALYZE COMPRESSION für die Tabelle aus, und verwenden Sie dann die Kodierungen zur Erstellung einer neuen Tabelle, lassen Sie jedoch die Kompressionskodierung für den Sortierschlüssel weg.

Die automatische Kompressionskodierung ist mit Leistungseinbußen verbunden, aber nur, wenn die Tabelle leer ist und nicht bereits über eine Kompressionskodierung verfügt. Bei kurzlebigen und häufig erstellten Tabellen wie Staging-Tabellen laden Sie die Tabelle einmal mit automatischer Kompression oder führen den Befehl ANALYZE COMPRESSION aus. Verwenden Sie dann diese Kodierungen, um neue Tabellen zu erstellen. Sie können diese Kodierungen zu der CREATE TABLE-Anweisung hinzufügen oder CREATE TABLE LIKE verwenden, um eine neue Tabelle mit der gleichen Kodierung zu erstellen.

Weitere Informationen finden Sie unter Laden von Tabellen mit automatischer Kompression.

Definition von Primär- und Fremdschlüsseleinschränkungen

Definieren Sie nach Bedarf Primär- und Fremdschlüsseleinschränkungen zwischen Tabellen. Obwohl diese nur Informationscharakter haben, nutzt der Abfrageoptimierer diese Einschränkungen zur Generierung effizienterer Abfragepläne.

Definieren Sie Primär- und Fremdschlüsseleinschränkungen nur, wenn Ihre Anwendung die Einschränkungen durchsetzt. Amazon Redshift setzt keine Einschränkungen für eindeutige, Primär- und Fremdschlüssel durch.

Weitere Informationen darüber, wie Amazon Redshift Einschränkungen verwendet, finden Sie unter Definieren von Tabelleneinschränkungen.

Verwendung der geringstmöglichen Spaltengröße

Gewöhnen Sie sich nicht an, aus Bequemlichkeit die maximale Spaltengröße zu verwenden.

Berücksichtigen Sie stattdessen die größten Werte, die wahrscheinlich in Ihren Spalten gespeichert werden, und dimensionieren Sie sie entsprechend. Beispielsweise muss eine CHAR-Spalte zum Speichern von Abkürzungen für US-Bundesstaaten und Territorien, die von der Post verwendet werden, nur CHAR (2) lauten.

Verwenden von Datum-/Uhrzeit-Datentypen für Datumsspalten

Amazon Redshift speichert DATE- und TIMESTAMP-Daten effizienter als CHAR oder VARCHAR, was die Abfrageleistung verbessert. Verwenden Sie den Datentyp DATE oder TIMESTAMP, je nach der Auflösung, die Sie benötigen, anstelle eines Zeichentyps beim Speichern von Datums-/Uhrzeitinformationen. Weitere Informationen finden Sie unter Datum-/Uhrzeittypen.

Bewährte Methoden für Amazon Redshift zum Laden von Daten

Themen

- Verwenden einer Mehrzeileneinfügung
- Verwenden einer Masseneinfügung
- Laden von Daten in Sortierschlüsselreihenfolge
- Laden von Daten in sequenziellen Blöcken
- Verwenden von Zeitreihentabellen
- Planen rund um Wartungszeitfenster

Das Laden sehr großer Datasets kann sehr lange dauern und die Datenverarbeitungsressourcen stark belasten. Wie Ihre Daten geladen werden, kann sich auch auf die Abfrageleistung auswirken. Dieser Abschnitt erläutert bewährte Methoden für das effiziente Laden von Daten mit COPY-Befehlen, Masseneinfügungen und Staging-Tabellen.

Tutorial: Laden von Daten

Tutorial: So laden Sie Daten aus Amazon S3 erläutert alle Schritte für das Hochladen von Daten zu einem Amazon S3 Bucket und die Verwendung des COPY-Befehls zum Laden der Daten in Ihre Tabellen. Das Tutorial bietet Hilfe bei der Behebung von Ladefehlern und vergleicht die Leistungsunterschiede zwischen dem Laden aus einer einzelnen und aus mehreren Dateien.

Verwenden eines COPY-Befehls zum Laden von Daten

Der COPY-Befehl lädt Daten parallel von Amazon S3, Amazon EMR, Amazon DynamoDb oder mehreren Datenquellen bzw. Remote-Hosts. COPY lädt große Mengen von Daten viel effizienter als INSERT-Anweisungen und speichert die Daten auch effektiver.

Weitere Informationen zur Verwendung des COPY-Befehls finden Sie unter So laden Sie Daten aus Amazon S3 und Laden von Daten aus einer Amazon-DynamoDB-Tabelle.

Verwenden eines einzelnen COPY-Befehls zum Laden aus mehreren Dateien

Amazon Redshift kann Daten automatisch parallel aus mehreren Datendateien laden. Sie können die Dateien, die geladen werden sollen, durch Verwendung eines Amazon-S3-Objektpräfixes oder einer Manifestdatei angeben.

Wenn Sie jedoch mehrere gleichzeitige COPY-Befehle verwenden, um eine Tabelle aus mehreren Dateien zu laden, ist Amazon Redshift gezwungen, einen serialisierten Ladevorgang durchzuführen. Dieser Ladetyp ist viel langsamer und erfordert einen VACUUM-Prozess am Ende, falls für die

Tabelle eine Sortierspalte definiert ist. Weitere Informationen zur Verwendung des Befehls COPY zum parallelen Laden von Daten finden Sie unter So laden Sie Daten aus Amazon S3.

Laden von Datendateien

Quelldatendateien weisen verschiedene Formate auf und verwenden unterschiedliche Komprimierungsalgorithmen. Beim Laden von Daten mit dem COPY-Befehl lädt Amazon Redshift alle Dateien, auf die mit dem Amazon-S3-Bucket-Präfix verwiesen wird. (Bei dem Präfix handelt es sich um eine Zeichenfolge am Anfang des Objektschlüsselnamens.) Wenn sich das Präfix auf mehrere Dateien oder auf aufteilbare Dateien bezieht, lädt Amazon Redshift die Daten parallel und nutzt dabei die MPP-Architektur von Amazon Redshift. Dadurch wird der Workload auf die Knoten im Cluster verteilt. Wenn Sie dagegen Daten aus einer Datei laden, die nicht aufgeteilt werden kann, muss Amazon Redshift einen serialisierten Ladevorgang durchführen, was viel langsamer ist. In den folgenden Abschnitten wird die empfohlene Methode zum Laden verschiedener Dateitypen in Amazon Redshift je nach Format und Komprimierung beschrieben.

Laden von Daten aus Dateien, die aufgeteilt werden können

Die folgenden Dateien können automatisch aufgeteilt werden, wenn ihre Daten geladen werden:

- unkomprimierte CSV-Dateien
- mit BZIP komprimierte CSV-Dateien
- spaltenbasierte Dateien (Parquet/ORC)

Amazon Redshift teilt Dateien mit mindestens 128 MB automatisch in Blöcke auf. Spaltenbasierte Dateien, insbesondere Parquet und ORC, werden nicht aufgeteilt, wenn sie weniger als 128 MB umfassen. Redshift verwendet zum Laden der Daten parallel arbeitende Slices. Dadurch ergibt sich beim Laden eine hohe Leistung.

Laden von Daten aus Dateien, die nicht aufgeteilt werden können

Dateitypen wie JSON oder CSV werden, wenn sie mit anderen Komprimierungsalgorithmen wie z. B. GZIP komprimiert wurden, nicht automatisch aufgeteilt. In diesen Fällen empfehlen wir, die Daten manuell in mehrere kleinere Dateien von ungefähr gleicher Größe zwischen 1 MB und 1 GB nach der Komprimierung aufzuteilen. Nutzen Sie zudem als Anzahl der Dateien ein Vielfaches der Anzahl der Slices in Ihrem Cluster. Weitere Informationen zum Aufteilen Ihrer Daten auf mehrere Dateien und Beispiele zur Verwendung von COPY zum Laden von Daten finden Sie unter So laden Sie Daten aus Amazon S3.

Komprimieren Ihrer Datendateien

Wir empfehlen für das Komprimieren großer Ladedateien die Verwendung von gzip, lzop, bzip2 oder Zstandard, um sie zu komprimieren und die Daten in mehrere, kleinere Dateien aufzuteilen.

Geben Sie die Option GZIP, LZOP, BZIP2 oder ZSTD mit dem COPY-Befehl an. Dieses Beispiel lädt die TIME-Tabelle aus einer pipe-begrenzten lzop-Datei.

```
copy time
from 's3://mybucket/data/timerows.lzo'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
lzop
delimiter '|';
```

Es gibt Fälle, in denen Sie unkomprimierte Datendateien nicht aufteilen müssen. Weitere Informationen zum Aufteilen Ihrer Daten und Beispiele zur Verwendung von COPY zum Laden von Daten finden Sie unter So laden Sie Daten aus Amazon S3.

Prüfen der Datendateien vor und nach einem Ladevorgang

Wenn Sie Daten von Amazon S3 laden, bestätigen Sie zunächst, dass Ihr Amazon-S3-Bucket alle korrekten Dateien nur diese enthält. Weitere Informationen finden Sie unter Überprüfen, ob sich die korrekten Dateien in Ihrem Bucket befinden.

Fragen Sie nach Abschluss der Ladeoperation die Systemtabelle STL_LOAD_COMMITS ab, um sicherzustellen, dass die erwarteten Dateien geladen wurden. Weitere Informationen finden Sie unter Überprüfung, ob die Daten korrekt geladen wurden.

Verwenden einer Mehrzeileneinfügung

Wenn ein COPY-Befehl nicht verwendet werden kann, und Sie SQL-Einfügungen benötigen, verwenden Sie nach Möglichkeit eine Mehrzeileneinfügung. Die Datenkompression ist ineffizient, wenn Sie nur jeweils eine oder wenige Zeilen von Daten hinzufügen.

Mehrzeileneinfügungen verbessern die Leistung durch die Zusammenfassung einer Reihe von Einfügungen. Im folgenden Beispiel werden drei Zeilen in eine Tabelle mit vier Spalten unter Verwendung einer einzelnen INSERT-Anweisung eingefügt. Dies ist immer noch eine kleine Einfügung, die lediglich die Syntax einer Mehrzeileneinfügung illustrieren soll.

```
insert into category_stage values
(default, default, default, default),
```

```
(20, default, 'Country', default),
(21, 'Concerts', 'Rock', default);
```

Weitere Details und Beispiele finden Sie unter INSERT.

Verwenden einer Masseneinfügung

Verwenden Sie eine Masseneinfügungsoperation mit einer SELECT-Klausel für Hochleistungs-Dateneinfügungen.

Verwenden Sie die Befehle INSERT und CREATE TABLE AS, wenn Sie Daten oder eine Teilmenge der Daten von einer Tabelle zu einer anderen verschieben müssen.

Die folgende INSERT-Anweisung wählt beispielsweise alle Zeilen aus der Tabelle CATEGORY aus und fügt sie in die Tabelle CATEGORY_STAGE ein.

```
insert into category_stage
(select * from category);
```

Das folgende Beispiel erstellt CATEGORY_STAGE als Kopie von CATEGORY und fügt alle Zeilen in CATEGORY in CATEGORY_STAGE ein.

```
create table category_stage as
select * from category;
```

Laden von Daten in Sortierschlüsselreihenfolge

Laden Sie Ihre Daten in Sortierschlüsselreihenfolge, damit keine Bereinigung erforderlich ist.

Falls der jeweilige Batch neuer Daten nach den vorhandenen Zeilen in Ihrer Tabelle eingefügt wird, werden Ihre Daten korrekt in der Sortierreihenfolge gespeichert und Sie müssen keine Bereinigung durchführen. Sie müssen die Zeilen in jeder Ladung nicht vorab sortieren, da COPY jedes Batch eingehender Daten beim Laden sortiert.

Nehmen Sie beispielsweise an, Sie laden jeden Tag Daten auf Grundlage der Aktivitäten des jeweiligen Tages. Wenn es sich bei Ihrem Sortierschlüssel um eine Zeitstempelspalte handelt, werden Ihre Daten in der Sortierreihenfolge gespeichert. Diese Reihenfolge entsteht, weil die Daten des aktuellen Tages am Ende der Daten des vorherigen Tages angehängt werden. Weitere Informationen finden Sie unter Laden von Daten in der Reihenfolge des Sortierschlüssels. Weitere Informationen zu Vakuumvorgängen finden Sie unter Vakuumiertabellen.

Laden von Daten in sequenziellen Blöcken

Wenn Sie große Mengen von Daten laden müssen, laden Sie die Daten in sequenziellen Blöcken gemäß der Sortierreihenfolge, damit keine Bereinigung erforderlich ist.

Nehmen Sie beispielsweise an, Sie müssen eine Tabelle mit Ereignissen von Januar 2017 bis Dezember 2017 laden. Wenn sich jeder Monat in einer einzigen Datei befindet, laden Sie die Zeilen für Januar, dann Februar usw. Ihre Tabelle ist nach Abschluss des Ladevorgangs vollständig sortiert und Sie müssen keine Bereinigung durchführen. Weitere Informationen finden Sie unter Verwenden von Zeitreihentabellen.

Beim Laden sehr großer Datasets kann der für die Sortierung erforderliche Platz den gesamten verfügbaren Platz überschreiten. Durch das Laden in kleineren Blöcken verwenden Sie sehr viel weniger temporären Sortierplatz bei jedem Ladevorgang. Darüber hinaus erleichtert das Laden kleinerer Blöcke den Neustart, wenn COPY fehlschlägt und zurückgefahren wird.

Verwenden von Zeitreihentabellen

Falls Ihre Daten einen festen Aufbewahrungszeitraum besitzen, können Sie die Daten als Sequenz von Zeitreihentabellen organisieren. In einer solchen Sequenz sind alle Tabellen identisch, enthalten aber jeweils Daten für verschiedene Zeiträume.

Sie können alte Daten einfach entfernen, indem Sie den DROP TABLE-Befehl für die entsprechenden Tabellen ausführen. Dieser Ansatz ist viel schneller als das Ausführen eines umfangreichen DELETE-Prozesses und erspart Ihnen das anschließende Ausführen eines VACUUM-Prozesses, um den Speicherplatz wieder zurückzugewinnen. Sie können eine UNION ALL-Ansicht erstellen, um zu verbergen, dass die Daten in unterschiedlichen Tabellen gespeichert sind. Wenn Sie alte Daten löschen, verfeinern Sie Ihre UNION ALL-Ansicht, um die abgelegten Tabellen zu entfernen. Fügen Sie ähnlich beim Laden neuer Zeiträume in neue Tabellen die neuen Tabellen der Ansicht hinzu. Die Ansichtsdefinition filtert nach dem Datumsbereich für die jeweilige Tabelle, um den Optimierer anzuweisen, die Tabellen bei der Durchsuchung auszulassen, die nicht dem Abfragefilter entsprechenden.

Die UNION ALL-Ansicht sollte nach Möglichkeit nicht zu viele Tabellen enthalten. Jede zusätzliche Tabelle fügt der Abfrage eine kurze Verarbeitungszeit hinzu. Tabellen müssen nicht denselben Zeitrahmen verwenden. Sie können beispielsweise Tabellen für unterschiedliche Zeiträume verwenden, wie täglich, monatlich und jährlich.

Wenn Sie Zeitreihentabellen mit einer Zeitstempelspalte als Sortierschlüssel verwenden, können Sie Ihre Daten effektiv in Sortierschlüsselreihenfolge laden. So vermeiden Sie, dass Sie zur erneuten

Sortierung der Daten eine Bereinigung durchführen müssen. Weitere Informationen finden Sie unter Laden von Daten in der Reihenfolge des Sortierschlüssels.

Planen rund um Wartungszeitfenster

Wenn eine geplante Wartung stattfindet, während eine Abfrage ausgeführt wird, wird die Abfrage beendet und zurückgefahren: Sie müssen sie dann erneut starten. Planen Sie längere Operationen, wie etwa große Datenladevorgänge oder eine VACUUM-Operation, so, dass Wartungszeitfenster vermieden werden. Sie können das Risiko verringern und Neustarts bei Bedarf erleichtern, indem Sie Datenladevorgänge in kleineren Inkrementen durchführen und die Größe Ihrer VACUUM-Operationen verwalten. Weitere Informationen erhalten Sie unter Laden von Daten in sequenziellen Blöcken und Bereinigen von Tabellen.

Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift

Befolgen Sie diese Empfehlungen beim Erstellen von Abfragen, um die Abfrageleistung zu maximieren:

- Gestalten Sie die Tabellen unter Berücksichtigung der bewährten Methoden, um eine solide Grundlage für die Abfrageleistung zu schaffen. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift.
- Vermeiden Sie die Verwendung von `select *`. Schließen Sie nur die Spalten ein, die Sie wirklich benötigen.
- Verwenden Sie einen Der bedingte Ausdruck CASE zur Durchführung komplexer Aggregierungen, anstatt mehrmals aus der selben Tabelle auszuwählen.
- Verwenden Sie Cross-Joins nur wenn unbedingt erforderlich. Diese Joins ohne Join-Bedingung ergeben das cartesische Produkt zweier Tabellen. Cross-Joins werden typischerweise als verschachtelte Loop–Joins durchgeführt; dies ist der langsamste der möglichen Join-Typen.
- Verwenden Sie Unterabfragen, wenn eine Tabelle in der Abfrage nur für Prädikatbedingungen verwendet wird, und die Unterabfrage eine kleine Zahl von Zeilen (weniger als etwa 200) zurückgibt. Das folgende Beispiel verwendet eine Unterabfrage, um das Joining der LISTING-Tabelle zu vermeiden.

```
select sum(sales.qtysold)
from sales
```

```
where salesid in (select listid from listing where listtime > '2008-12-26');
```

- Verwenden Sie Prädikate, um den Datensatz so weit wie möglich einzuschränken.
- Verwenden Sie in den Prädikaten die Operatoren mit den geringstmöglichen Kosten. Vergleichsbedingung-Operatoren sind gegenüber LIKE-Operatoren vorzuziehen. LIKE-Operatoren sind weiterhin gegenüber SIMILAR TO oder POSIX-Operatoren vorzuziehen.
- Vermeiden Sie die Verwendung von Funktionen in Abfrageprädikaten. Deren Verwendung kann die Kosten der Abfrage dadurch erhöhen, dass große Zahlen von Zeilen die Zwischenschritte der Abfrage auflösen müssen.
- Verwenden Sie nach Möglichkeit eine WHERE-Klausel, um den Datensatz einzuschränken. Der Abfrageplaner kann dann die Zeilenreihenfolge verwenden, um zu bestimmen, welche Datensätze den Kriterien entsprechen, und entsprechend das Scannen großer Zahlen von Festplattenblöcken überspringen. Ohne dies muss die Abfrageausführung die beteiligten Spalten vollständig scannen.
- Fügen Sie Prädikate zu Filtertabellen hinzu, die an Joins beteiligt sind, selbst wenn die Prädikate die gleichen Filter anwenden. Die Abfrage gibt den gleichen Ergebnissatz zurück, Amazon Redshift kann jedoch die Join-Tabellen vor dem Scan-Schritt filtern und dann das Scannen von Blöcken aus diesen Tabellen effektiv übergehen. Redundante Filter werden nicht benötigt, wenn Sie auf einer Spalte filtern, die in der Join-Bedingung verwendet wird.

  Nehmen Sie zum Beispiel an, Sie möchten `SALES` und `LISTING` verbinden, um Ticketverkäufe für nach dem Dezember gelistete Tickets, gruppiert nach Verkäufer, zu finden. Beide Tabellen werden nach Datum sortiert. Die folgende Abfrage verbindet die Tabellen auf ihrem gemeinsamen Schlüssel und filtert nach `listing.listtime`-Werten nach dem 1. Dezember.

  ```
  select listing.sellerid, sum(sales.qtysold)
  from sales, listing
  where sales.salesid = listing.listid
  and listing.listtime > '2008-12-01'
  group by 1 order by 1;
  ```

  Die WHERE-Klausel enthält kein Prädikat für `sales.saletime`, die Ausführungsengine muss daher die gesamte `SALES`-Tabelle filtern. Wenn Sie wissen, dass der Filter dazu führt, dass weniger Zeilen an dem Join beteiligt sind, fügen Sie auch diesen Filter hinzu. Beim folgenden Beispiel ist die Ausführungszeit erheblich verkürzt.

  ```
  select listing.sellerid, sum(sales.qtysold)
  from sales, listing
  where sales.salesid = listing.listid
  ```

```
and listing.listtime > '2008-12-01'
and sales.saletime > '2008-12-01'
group by 1 order by 1;
```

- Verwenden Sie Sortierschlüssel in der GROUP BY-Klausel, so dass der Abfrageplaner eine effizientere Aggregierung verwenden kann. Eine Abfrage kann für eine einphasige Aggregierung in Frage kommen, wenn ihre GROUP BY-Liste Sortierschlüsselspalten enthält, von denen eine auch der Verteilungsschlüssel ist. Die Sortierschlüsselspalten in der GROUP BY-Liste müssen den ersten Sortierschlüssel und dann weitere Sortierschlüssel, die Sie verwenden möchten, in der Sortierschlüsselreihenfolge enthalten. So ist es beispielsweise korrekt, den ersten Sortierschlüssel, den ersten und den zweiten Sortierschlüssel, den ersten, zweiten und dritten Sortierschlüssel zu verwenden usw. Es ist nicht möglich, den ersten und den dritten Sortierschlüssel zu verwenden.

  Sie können die Verwendung der einphasigen Aggregierung bestätigen, indem Sie den Befehl EXPLAIN verwenden und nach `XN GroupAggregate` im Aggregierungsschritt der Abfrage suchen.
- Wenn Sie die GROUP BY- und die ORDER BY-Klausel verwenden, müssen Sie sicherstellen, dass die Spalten in beiden die gleiche Reihenfolge haben. Gehen Sie also einfach wie folgt vor:

  ```
  group by a, b, c
  order by a, b, c
  ```

  Gehen Sie nicht wie folgt vor.

  ```
  group by b, c, a
  order by a, b, c
  ```

Arbeiten mit Empfehlungen von Amazon Redshift Advisor

Um Ihnen bei der Verbesserung der Leistung und bei der Senkung der Betriebskosten für Ihren Amazon-Redshift-Cluster zu helfen, bietet Amazon Redshift Advisor Ihnen spezifische Empfehlungen über vorzunehmende Änderungen. Advisor entwickelt seine angepassten Empfehlungen durch Analyse der Leistung und Nutzungsmetriken für Ihr Cluster. Diese individuellen Empfehlungen beziehen sich auf Einstellungen für Operationen und Cluster. Advisor ordnet Empfehlungen in der Reihenfolge ihrer Auswirkungen ein, damit Sie Ihre Optimierungen priorisieren können.

Advisor basiert die Empfehlungen auf Beobachtungen im Hinblick auf Leistungsstatistiken oder Operationsdaten. Advisor entwickelt Beobachtungen durch Ausführen von Tests an Ihren Clustern,

um zu bestimmen, ob ein Testwert innerhalb eines festgelegten Bereichs liegt. Falls das Testergebnis außerhalb dieses Bereichs liegt, generiert Advisor eine Beobachtung für Ihr Cluster. Zur selben Zeit erstellt Advisor eine Empfehlung, wie Sie den beobachteten Wert zurück in den Bereich der bewährten Methode bringen. Der Advisor zeigt nur Empfehlungen an, die erhebliche Auswirkungen auf die Leistung und die Operationen haben sollten. Wenn Advisor ermittelt, dass eine Empfehlung angenommen wurde, wird diese aus Ihrer Empfehlungsliste entfernt.

Nehmen Sie beispielsweise an, Ihr Data Warehouse enthält viele unkomprimierte Tabellenspalten. In diesem Fall können Sie Cluster-Speicherkosten sparen, indem Sie die Tabelle mit dem `ENCODE`-Parameter neu erstellen und dabei festzulegen, dass die Spalten komprimiert sind. Ein weiteres Beispiel: Nehmen Sie an, der Advisor erkennt, dass Ihr Cluster eine große Menge an Daten in Tabellen enthält, die nicht komprimiert sind. In diesem Fall stellt Ihnen der Advisor den SQL-Codeblock bereit, mit dem die Tabellenspalten, die Kandidaten für eine Komprimierung sind, gefunden werden können, sowie Ressourcen, die beschreiben, wie diese Spalten komprimiert werden.

Amazon-Redshift-Regionen

Die Amazon Redshift Advisor-Funktion ist nur in den folgenden AWS Regionen verfügbar:

- Region USA Ost (Nord-Virginia) (us-east-1)
- Region USA Ost (Ohio) (us-east-2)
- Region USA West (Nordkalifornien) (us-west-1)
- Region USA West (Oregon) (us-west-2)
- Region Afrika (Kapstadt) (af-south-1)
- Region Asien-Pazifik (Hongkong) (ap-east-1)
- Region Asien-Pazifik (Hyderabad) (ap-south-2)
- Region Asien-Pazifik (Jakarta) (ap-southeast-3)
- Region Asien-Pazifik (Melbourne) (ap-southeast-4)
- Region Asien-Pazifik (Mumbai) (ap-south-1)
- Region Asien-Pazifik (Osaka) (ap-northeast-3)
- Region Asien-Pazifik (Seoul) (ap-northeast-2)
- Region Asien-Pazifik (Singapur) (ap-southeast-1)
- Region Asien-Pazifik (Sydney) (ap-southeast-2)
- Region Asien-Pazifik (Tokio) (ap-northeast-1)

- Region Kanada (Zentral) (ca-central-1)
- Region Kanada West (Calgary) (ca-west-1)
- Region China (Peking) (cn-north-1)
- Region China (Ningxia) (cn-northwest-1)
- Region Europa (Frankfurt) (eu-central-1)
- Region Europa (Irland) (eu-west-1)
- Region Europa (London) (eu-west-2)
- Region Europa (Mailand) (eu-south-1)
- Region Europa (Paris) (eu-west-3)
- Region Europa (Spanien) (eu-south-2)
- Region Europa (Stockholm) (eu-north-1)
- Region Europa (Zürich) (eu-central-2)
- Region Israel (Tel Aviv) (il-central-1)
- Region Naher Osten (Bahrain) (me-south-1)
- Region Naher Osten (VAE) (me-central-1)
- Region Südamerika (São Paulo) (sa-east-1)

Themen

Empfehlungen von Amazon Redshift Advisor anzeigen

Sie können über die Amazon Redshift-Konsole, die Amazon Redshift Redshift-API oder auf Amazon Redshift Advisor-Empfehlungen zugreifen. AWS CLI Um auf Empfehlungen zugreifen zu können, müssen Sie mit Ihrer `redshift:ListRecommendations` IAM-Rolle oder Identität über eine entsprechende Berechtigung verfügen.

Amazon Redshift Advisor-Empfehlungen auf der von Amazon Redshift bereitgestellten Konsole anzeigen

Sie können die Empfehlungen von Amazon Redshift Advisor auf der AWS Management Console einsehen.

Um Amazon Redshift Advisor-Empfehlungen für Amazon Redshift Redshift-Cluster auf der Konsole anzuzeigen

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Advisor aus.
3. Erweitern Sie die jeweilige Empfehlung, um weitere Details anzuzeigen. Auf dieser Seite können Sie Empfehlungen sortieren und gruppieren.

Amazon Redshift Advisor-Empfehlungen mithilfe von Amazon Redshift Redshift-API-Vorgängen anzeigen

Sie können Amazon Redshift Advisor-Empfehlungen für Amazon Redshift Redshift-Cluster mithilfe der Amazon Redshift Redshift-API auflisten. In der Regel entwickeln Sie eine Anwendung in der Programmiersprache Ihrer Wahl, um die `redshift:ListRecommendations` API mithilfe eines SDK aufzurufen. AWS Weitere Informationen finden Sie ListRecommendationsin der Amazon Redshift API-Referenz.

Amazon Redshift Advisor-Empfehlungen mithilfe von AWS Command Line Interface Operationen anzeigen

Sie können Amazon Redshift Advisor-Empfehlungen für Amazon Redshift Redshift-Cluster mithilfe der auflisten. AWS Command Line InterfaceWeitere Informationen finden Sie unter list-recommendations in der Befehlsreferenz.AWS CLI

Empfehlungen für Amazon Redshift Advisor

Amazon Redshift Advisor bietet Empfehlungen über die Optimierung Ihres Amazon-Redshift-Clusters zum Steigern der Leistung und Sparen von Betriebskosten. Sie finden Erläuterungen für jede Empfehlung in der Konsole, wie vorstehend beschrieben. Sie finden weitere Details zu diesen Empfehlungen in den folgenden Abschnitten.

Themen

- Komprimieren von Amazon-S3-Objekten, die über COPY geladen wurden
- Isolierung mehrerer aktiver Datenbanken
- Neuzuordnung des Workload-Management- (WLM) Arbeitsspeichers
- Überspringen der Komprimierungsanalyse während des COPY-Vorgangs

- Teilen Sie Amazon-S3-Objekte mit dem Befehl COPY auf
- Aktualisieren der Tabellenstatistik
- Aktivieren von Short Query Acceleration
- Ändern der Verteilungsschlüssel in Tabellen
- Ändern der Sortierschlüssel für Tabellen
- Ändern der Komprimierungskodierungen für Spalten
- Datentyp-Empfehlungen

Komprimieren von Amazon-S3-Objekten, die über COPY geladen wurden

Der Befehl COPY nutzt die massive Parallelverarbeitungsarchitektur (Massively Parallel Processing, MPP) in Amazon Redshift, um Daten parallel zu lesen und zu laden. Der Befehl liest Dateien aus Amazon S3, DynamoDB-Tabellen und Textausgaben von einem oder mehreren Remotehosts.

Beim Laden von großen Datenmengen empfehlen wir dringend die Verwendung des COPY-Befehls zum Laden komprimierter Datendateien aus S3. Wenn Sie große Datensätze komprimieren, können Sie die Dateien schneller in Amazon S3 hochladen. COPY kann auch den Ladeprozess durch Dekomprimierung der Dateien während des Lesens beschleunigen.

Analyse

Lange laufende COPY-Befehle, die große, nicht komprimierte Datasets laden, bieten häufig die Möglichkeit einer erheblichen Leistungsverbesserung. Die Advisor-Analyse erkennt COPY-Befehle, die große, nicht komprimierte Datasets laden. In einem solchen Fall generiert Advisor eine Empfehlung zur Implementierung einer Komprimierung in den Quelldateien in Amazon S3.

Empfehlung

Stellen Sie sicher, dass jeder COPY-Vorgang, der eine erhebliche Datenmenge lädt oder über einen längeren Zeitraum ausgeführt wird, komprimierte Datenobjekte aus Amazon S3 übernimmt. Sie können herausfinden, welche COPY-Befehle große, unkomprimierte Datensätze aus Amazon S3 laden, indem Sie den folgenden SQL-Befehl als Superuser ausführen.

```
SELECT
    wq.userid, query, exec_start_time AS starttime, COUNT(*) num_files,
    ROUND(MAX(wq.total_exec_time/1000000.0),2) execution_secs,
```

```
    ROUND(SUM(transfer_size)/(1024.0*1024.0),2) total_mb,
    SUBSTRING(querytxt,1,60) copy_sql
FROM stl_s3client s
JOIN stl_query q USING (query)
JOIN stl_wlm_query wq USING (query)
WHERE s.userid>1 AND http_method = 'GET'
    AND POSITION('COPY ANALYZE' IN querytxt) = 0
    AND aborted = 0 AND final_state='Completed'
GROUP BY 1, 2, 3, 7
HAVING SUM(transfer_size) = SUM(data_size)
AND SUM(transfer_size)/(1024*1024) >= 5
ORDER BY 6 DESC, 5 DESC;
```

Falls die bereitgestellten Daten nach dem Laden in Amazon S3 verbleiben, was in Data-Lake-Architekturen üblich ist, können durch Speichern der Daten in komprimierter Form Ihre Speicherkosten gesenkt werden.

Implementierungstipps

- Die ideale Objektgröße beträgt nach der Komprimierung 1 bis 128 MB.
- Sie können Dateien mit dem Format gzip, lzop oder bzip2 komprimieren.

Isolierung mehrerer aktiver Datenbanken

Als bewährte Methode empfehlen wir die Isolierung von Datenbanken in Amazon Redshift von einander. Abfragen werden in einer spezifischen Datenbank ausgeführt und können keine Daten von einer anderen Datenbank im Cluster aufrufen. Die Abfragen, die Sie in allen Datenbanken eines Clusters ausführen, verwenden jedoch denselben zugrunde liegenden Cluster-Speicherplatz und dieselben Datenverarbeitungsressourcen. Wenn ein einzelnes Cluster mehrere aktive Datenbanken enthält, sind ihre Workloads in der Regel nicht miteinander verbunden.

Analyse

Die Advisor-Analyse prüft alle Datenbanken im Cluster auf aktive Workloads, die gleichzeitig ausgeführt werden. Falls aktive Workloads gleichzeitig ausgeführt werden, generiert Advisor eine Empfehlung, über die Migration von Datenbanken in separate Amazon-Redshift-Cluster nachzudenken.

Empfehlung

Denken Sie über das Verschieben jeder aktiv abgefragten Datenbank in ein separates dediziertes Cluster nach. Die Verwendung eines separaten Clusters kann Ressourcenkonflikte reduzieren und die Abfrageleistung verbessern. Dies ist möglich, da Sie die Größe jedes Clusters entsprechend den Speicher-, Kosten- und Leistungsanforderungen des jeweiligen Workloads festlegen können. Zudem profitieren nicht verbundene Workloads häufig von verschiedenen Workload-Management-Konfigurationen.

Sie können den folgenden SQL-Befehl als Superuser ausführen, um zu ermitteln, welche Datenbanken aktiv verwendet werden.

```
SELECT database,
  COUNT(*) as num_queries,
  AVG(DATEDIFF(sec,starttime,endtime)) avg_duration,
  MIN(starttime) as oldest_ts,
  MAX(endtime) as latest_ts
FROM stl_query
WHERE userid > 1
GROUP BY database;
```

Implementierungstipps

- Da ein Benutzer spezifisch eine Verbindung zur jeweiligen Datenbank herstellen muss und Abfragen nur eine einzelne Datenbank aufrufen können, hat das Verschieben von Datenbanken in separate Cluster geringfügige Auswirkungen für Benutzer.
- Eine Option zum Verschieben einer Datenbank sind folgende Schritte:
  1. Vorübergehendes Wiederherstellen eines Snapshots des aktuellen Clusters in einem Cluster derselben Größe.
  2. Löschen aller Datenbanken aus dem neuen Cluster außer der zu verschiebenden Zieldatenbank.
  3. Ändern der Größe des Clusters in einen entsprechenden Knotentyp und eine entsprechende Knotenanzahl für den Workload der Datenbank.

Neuzuordnung des Workload-Management- (WLM) Arbeitsspeichers

Amazon Redshift leitet Benutzerabfragen zur Verarbeitung an Implementieren von manuellem WLM weiter. Workload Management (WLM) legt fest, wie diese Abfragen zu den Warteschlangen geleitet

werden. Amazon Redshift weist jeder Warteschlange einen Teil des verfügbaren Speicherplatzes des Clusters zu. Der Speicher einer Warteschlange wird unter den Abfrageslots der Warteschlange aufgeteilt.

Wenn eine Warteschlange mit mehr Slots konfiguriert ist, als der Workload benötigt, ist der Arbeitsspeicher, der diesen nicht verwendeten Slots zugewiesen ist, nicht ausgelastet. Durch Reduzieren der konfigurierten Slots und Anpassen an die Anforderungen der Spitzenworkloads wird jeglicher nicht ausgelasteter Arbeitsspeicher neu verteilt. Dies kann zu einer verbesserten Abfrageleistung führen.

Analyse

Die Advisor-Analyse prüft die Anforderungen an die Nebenläufigkeit des Workloads, um Abfragewarteschlangen mit ungenutzten Slots zu erkennen. Advisor generiert eine Empfehlung, um die Anzahl der Slots in einer Warteschlange zu reduzieren, wenn Folgendes gefunden wird:

- Eine Warteschlange mit Slots, die während der gesamten Analyse vollständig inaktiv waren
- Eine Warteschlange mit mehr als vier Slots, von denen mindestens zwei Slots während der gesamten Analyse inaktiv waren

Empfehlung

Durch die Reduzierung der konfigurierten Slots und Anpassung an die Anforderungen des Spitzenworkloads wird ungenutzter Arbeitsspeicher an aktive Slots neu verteilt. Denken Sie über eine Reduzierung der konfigurierten Slot-Anzahl für Warteschlangen nach, bei denen die Slots nie vollständig verwendet wurden. Um diese Warteschlangen zu ermitteln, können Sie die stündlichen Spitzenanforderungen an Slots für jede Warteschlange vergleichen, indem Sie den folgenden SQL-Befehl als Superuser ausführen.

```
WITH
 generate_dt_series AS (select sysdate - (n * interval '5 second') as dt from (select
 row_number() over () as n from stl_scan limit 17280)),
 apex AS (
     SELECT iq.dt, iq.service_class, iq.num_query_tasks, count(iq.slot_count) as
 service_class_queries, sum(iq.slot_count) as service_class_slots
     FROM
         (select gds.dt, wq.service_class, wscc.num_query_tasks, wq.slot_count
         FROM stl_wlm_query wq
```

```
        JOIN stv_wlm_service_class_config wscc ON (wscc.service_class =
 wq.service_class AND wscc.service_class > 5)
        JOIN generate_dt_series gds ON (wq.service_class_start_time <= gds.dt AND
 wq.service_class_end_time > gds.dt)
        WHERE wq.userid > 1 AND wq.service_class > 5) iq
    GROUP BY iq.dt, iq.service_class, iq.num_query_tasks),
    maxes as (SELECT apex.service_class, trunc(apex.dt) as d, date_part(h,apex.dt) as
 dt_h, max(service_class_slots) max_service_class_slots
                    from apex group by apex.service_class, apex.dt,
 date_part(h,apex.dt))
SELECT apex.service_class - 5 AS queue, apex.service_class, apex.num_query_tasks AS
 max_wlm_concurrency, maxes.d AS day, maxes.dt_h || ':00 - ' || maxes.dt_h || ':59' as
 hour, MAX(apex.service_class_slots) as max_service_class_slots
FROM apex
JOIN maxes ON (apex.service_class = maxes.service_class AND apex.service_class_slots =
 maxes.max_service_class_slots)
GROUP BY  apex.service_class, apex.num_query_tasks, maxes.d, maxes.dt_h
ORDER BY apex.service_class, maxes.d, maxes.dt_h;
```

Die `max_service_class_slots`-Spalte steht für die maximale Anzahl der WLM-Abfrageslots in der Abfragewarteschlange für diese Stunde. Falls nicht ausgelastete Warteschlangen vorhanden sind, implementieren Sie die Optimierung der Slot-Reduzierung durch Ändern einer Parametergruppe wie im Amazon-Redshift-Verwaltungshandbuch beschrieben.

Implementierungstipps

- Falls der Workload-Umfang stark variiert, stellen Sie sicher, dass die Analyse einen Zeitraum mit Spitzenauslastung erfasst hat. Falls dies nicht der Fall ist, führen Sie den vorstehenden SQL-Befehl wiederholt aus, um die Anforderungen an die Spitzengleichläufigkeit zu überwachen.
- Weitere Informationen zur Interpretation der Abfrageergebnisse aus dem vorherigen SQL-Code finden Sie im Skript wlm_apex_hourly.sql unter. GitHub

Überspringen der Komprimierungsanalyse während des COPY-Vorgangs

Beim Laden von Daten in eine leere Tabelle mit Komprimierungskodierung, die mit dem Befehl COPY deklariert war, übernimmt Amazon Redshift die Speicherkomprimierung. Diese Optimierung stellt sicher, dass Daten in Ihrem Cluster sogar effizient gespeichert werden, wenn sie von Endbenutzern geladen werden. Die erforderliche Analyse zur Anwendung der Komprimierung kann sehr zeitraubend sein.

Analyse

Die Advisor-Analyse prüft COPY-Operationen, die verzögert wurden, anhand einer automatischen Komprimierungsanalyse. Die Analyse bestimmt die Komprimierungskodierungen durch Analysieren der Daten, während sie geladen werden. Diese Analyse ist ähnlich wie die, die mit dem Befehl ANALYZE COMPRESSION ausgeführt wird.

Wenn Sie Daten als Teil eines strukturierten Prozesses laden, wie etwa bei einem ETL-Batch (Extrahieren, Transformieren, Laden) über Nacht, können Sie die Komprimierung vorab definieren. Sie können Ihre Tabellendefinitionen so optimieren, dass diese Phase dauerhaft ohne negative Auswirkungen übersprungen wird.

Empfehlung

Um die Reaktionsfähigkeit des COPY-Vorgangs durch Überspringen der Komprimierungsanalyse zu verbessern, implementieren Sie eine der folgenden beiden Optionen:

- Verwenden Sie den Spaltenparameter `ENCODE` beim Erstellen von Tabellen, die Sie mithilfe des COPY-Befehls laden.
- Schalten Sie die Komprimierung insgesamt durch Angeben des `COMPUPDATE OFF`-Parameters im COPY-Befehl aus.

Die beste Lösung allgemein ist die Verwendung der Spaltenkodierung während des Erstellens der Tabelle, da dieser Ansatz auch den Vorteil des Speicherns komprimierter Daten auf dem Datenträger beibehält. Sie können den Befehl ANALYZE COMPRESSION verwenden, um Komprimierungskodierungen vorzuschlagen, aber müssen die Tabelle neu erstellen, um diese Kodierungen zu übernehmen. Um diesen Prozess zu automatisieren, können Sie das AWSColumnEncodingHilfsprogramm verwenden, das Sie unter finden GitHub.

Um die aktuellen COPY-Operationen zu ermitteln, die die automatische Komprimierungsanalyse ausgelöst haben, führen Sie den folgenden SQL-Befehl aus.

```
  WITH xids AS (
    SELECT xid FROM stl_query WHERE userid>1 AND aborted=0
    AND querytxt = 'analyze compression phase 1' GROUP BY xid
    INTERSECT SELECT xid FROM stl_commit_stats WHERE node=-1)
SELECT a.userid, a.query, a.xid, a.starttime, b.complyze_sec,
    a.copy_sec, a.copy_sql
```

```
FROM (SELECT q.userid, q.query, q.xid, date_trunc('s',q.starttime)
    starttime, substring(querytxt,1,100) as copy_sql,
    ROUND(datediff(ms,starttime,endtime)::numeric / 1000.0, 2) copy_sec
    FROM stl_query q JOIN xids USING (xid)
    WHERE (querytxt ilike 'copy %from%' OR querytxt ilike '% copy %from%')
    AND querytxt not like 'COPY ANALYZE %') a
LEFT JOIN (SELECT xid,
    ROUND(sum(datediff(ms,starttime,endtime))::numeric / 1000.0,2) complyze_sec
    FROM stl_query q JOIN xids USING (xid)
    WHERE (querytxt like 'COPY ANALYZE %'
    OR querytxt like 'analyze compression phase %')
    GROUP BY xid ) b ON a.xid = b.xid
WHERE b.complyze_sec IS NOT NULL ORDER BY a.copy_sql, a.starttime;
```

Implementierungstipps

- Stellen Sie sicher, dass alle Tabellen von erheblicher Größe, die während Ihrer ETL-Prozesse erstellt wurden (Beispiel: Staging-Tabellen und temporäre Tabellen), eine Komprimierungskodierung für alle Spalten außer dem ersten Sortierschlüssel deklarieren.
- Schätzen Sie die erwartete Lebenszeitgröße der Tabelle, die für jeden COPY-Befehl geladen wird, der vom vorstehenden SQL-Befehl erkannt wird. Falls Sie sicher sind, dass die Tabelle extrem klein bleibt, schalten Sie die Komprimierung zusammen mit dem `COMPUPDATE OFF`-Parameter aus. Erstellen Sie die Tabelle anderenfalls mit einer expliziten Komprimierung, bevor Sie sie mit dem COPY-Befehl laden.

Teilen Sie Amazon-S3-Objekte mit dem Befehl COPY auf

Der Befehl COPY nutzt die massive Parallelverarbeitungsarchitektur (Massively Parallel Processing, MPP) in Amazon Redshift, um Daten aus Dateien in Amazon S3 zu lesen und zu laden. Der COPY-Befehl lädt die Daten in paralleler Weise aus mehreren Dateien und teilt dabei den Workload über die Knoten in Ihrem Cluster auf. Um einen optimalen Durchsatz zu erzielen, wird nachdrücklich empfohlen, die Daten in mehrere Dateien aufzuteilen, um die Vorteile der Parallelverarbeitung nutzen zu können.

Analyse

Bei der Advisor-Analyse werden COPY-Befehle erkannt, die große Datensätze laden, die in einer kleinen Anzahl von Dateien enthalten sind, die in Amazon S3 bereitgestellt wurden. Lange laufende COPY-Befehle, die große Datasets aus wenigen Dateien laden, bieten häufig die Möglichkeit

einer erheblichen Leistungsverbesserung. Wenn Advisor erkennt, dass diese COPY-Befehle eine erhebliche Menge Zeit in Anspruch nehmen, wird die Empfehlung erstellt, die parallele Ausführung durch Aufteilen der Daten in zusätzliche Dateien in Amazon S3 zu verstärken.

Empfehlung

In diesem Fall empfehlen wir die folgenden Aktionen, aufgelistet in Reihenfolge ihrer Priorität:

1. Optimieren der COPY-Befehle, die weniger Dateien als die Anzahl der Cluster-Knoten laden.
2. Optimieren der COPY-Befehle, die weniger Dateien als die Anzahl der Cluster-Slices laden.
3. Optimieren der COPY-Befehle, bei denen die Anzahl der Dateien kein Vielfaches der Anzahl der Cluster-Slices ist.

Bestimmte COPY-Befehle laden eine erhebliche Datenmenge oder laufen sehr lang. Wir empfehlen, die Anzahl der aus Amazon S3 zu ladenden Datenobjekte so zu wählen, dass sie einem Vielfachen der Anzahl der Slices im Cluster entspricht. Um zu ermitteln, wie viele S3-Objekte der jeweilige COPY-Befehl geladen hat, führen Sie den folgenden SQL-Code als Superuser aus.

```
SELECT
    query, COUNT(*) num_files,
    ROUND(MAX(wq.total_exec_time/1000000.0),2) execution_secs,
    ROUND(SUM(transfer_size)/(1024.0*1024.0),2) total_mb,
    SUBSTRING(querytxt,1,60) copy_sql
FROM stl_s3client s
JOIN stl_query q USING (query)
JOIN stl_wlm_query wq USING (query)
WHERE s.userid>1 AND http_method = 'GET'
    AND POSITION('COPY ANALYZE' IN querytxt) = 0
    AND aborted = 0 AND final_state='Completed'
GROUP BY query, querytxt
HAVING (SUM(transfer_size)/(1024*1024))/COUNT(*) >= 2
ORDER BY CASE
WHEN COUNT(*) < (SELECT max(node)+1 FROM stv_slices) THEN 1
WHEN COUNT(*) < (SELECT COUNT(*) FROM stv_slices WHERE node=0) THEN 2
ELSE 2+((COUNT(*) % (SELECT COUNT(*) FROM stv_slices))/(SELECT COUNT(*)::DECIMAL FROM
 stv_slices))
END, (SUM(transfer_size)/(1024.0*1024.0))/COUNT(*) DESC;
```

Implementierungstipps

- Die Anzahl der Slices in einem Knoten ist von der Knotengröße des Clusters abhängig. Weitere Informationen zur Anzahl der Slices für die einzelnen Knotengrößen finden Sie unter Clusters and Nodes in Amazon Redshift (Cluster und Knoten in Amazon Redshift) im Amazon-Redshift-Verwaltungshandbuch.
- Sie können mehrere Dateien durch die Angabe eines gemeinsamen Präfixes bzw. Präfixschlüssels für den Satz oder durch die explizite Auflistung der Dateien in einer Manifestdatei laden. Weitere Informationen zum Laden von Dateien finden Sie unter Laden von Daten aus komprimierten und unkomprimierten Dateien.
- Amazon Redshift berücksichtigt die Dateigröße beim Aufteilen des Workloads nicht. Teilen Sie Ihre Ladedatendateien, so dass die Dateien etwa gleich groß sind, zwischen 1 MB und 1 GB nach der Kompression.

Aktualisieren der Tabellenstatistik

Amazon Redshift verwendet eine kostenbasierte Abfrageoptimierung zur Auswahl eines optimalen Ausführungsplans für Abfragen. Die Kostenschätzungen basieren auf Tabellenstatistiken, die mit dem Befehl ANALYZE erfasst wurden. Wenn die Statistiken nicht mehr aktuell sind oder fehlen, wählt die Datenbank möglicherweise einen weniger effizienten Plan für die Abfrageausführung, insbesondere bei komplexen Abfragen. Wenn die Statistiken aktuell sind, hilft dies dabei, komplexe Abfragen in der kürzestmöglichen Zeit auszuführen.

Analyse

Die Advisor-Analyse verfolgt Tabellen, deren Statistiken vorhanden sind out-of-date oder fehlen. Der Advisor untersucht Metadaten zu Tabellenzugriffen bei komplexen Abfragen. Wenn der Advisor feststellt, dass Tabellen mit komplexen Zugriffsmustern keine Statistiken haben, gibt er eine kritische Empfehlung aus, ANALYZE auszuführen. Wenn Tabellen, auf die häufig mit komplexen Mustern zugegriffen wird, out-of-date Statistiken enthalten, schlägt Advisor vor, ANALYZE auszuführen.

Empfehlung

Immer wenn sich der Inhalt einer Tabelle erheblich ändert, sollten Sie die Statistiken mit ANALYZE aktualisieren. Wir empfehlen die Ausführung von ANALYZE, wenn eine große Anzahl neuer Datenzeilen mit dem COPY- oder dem INSERT-Befehl in eine vorhandene Tabelle geladen wird. Wir empfehlen zudem die Ausführung von ANALYZE, wenn eine erhebliche Anzahl von Zeilen mit dem UPDATE- oder dem DELETE-Befehl modifiziert wird. Um Tabellen mit fehlenden oder out-of-date

statistischen Daten zu identifizieren, führen Sie den folgenden SQL-Befehl als Superuser aus. Die Ergebnisse werden von der größten zur kleinsten Tabelle sortiert.

Um Tabellen mit fehlenden oder out-of-date statistischen Daten zu identifizieren, führen Sie den folgenden SQL-Befehl als Superuser aus. Die Ergebnisse werden von der größten zur kleinsten Tabelle sortiert.

```
SELECT
   ti.schema||'.'||ti."table" tablename,
   ti.size table_size_mb,
   ti.stats_off statistics_accuracy
 FROM svv_table_info ti
 WHERE ti.stats_off > 5.00
 ORDER BY ti.size DESC;
```

Implementierungstipps

Der Standardschwellenwert für ANALYZE ist 10 Prozent. Dieser Standardwert bedeutet, dass der ANALYZE-Befehl eine vorhandene Tabelle überspringt, wenn weniger als 10 Prozent der Tabellenzeilen seit dem letzten ANALYZE-Vorgang geändert wurden. Daher können Sie am Ende jedes ETL-Prozesses wählen, ob ANALYZE-Befehle ausgeführt werden sollen. Dieser Ansatz beinhaltet, dass der Befehl ANALYZE häufig übersprungen wird, aber sichergestellt ist, dass er bei Bedarf ausgeführt wird.

ANALYZE-Statistiken haben die größten Auswirkungen auf Spalten, die in Joins verwendet werden (Beispiel: `JOIN tbl_a ON col_b`) oder als Prädikate (Beispiel: `WHERE col_b = 'xyz'`). Standardmäßig erfasst ANALYZE Statistiken für alle in der Tabelle angegebenen Spalten. Bei Bedarf können Sie die erforderliche Zeit zum Ausführen von ANALYZE reduzieren, indem Sie ANALYZE nur für die Spalten mit den größten Auswirkungen ausführen. Sie können den folgenden SQL-Befehl ausführen, um Spalten zu ermitteln, die als Prädikate verwendet werden. Sie können auch Amazon Redshift wählen lassen, welche Spalten analysiert werden sollen, indem Sie angeben `ANALYZE PREDICATE COLUMNS`.

```
WITH predicate_column_info as (
SELECT ns.nspname AS schema_name, c.relname AS table_name, a.attnum as col_num,
 a.attname as col_name,
        CASE
            WHEN 10002 = s.stakind1 THEN array_to_string(stavalues1, '||')
```

```
            WHEN 10002 = s.stakind2 THEN array_to_string(stavalues2, '||')
            WHEN 10002 = s.stakind3 THEN array_to_string(stavalues3, '||')
            WHEN 10002 = s.stakind4 THEN array_to_string(stavalues4, '||')
            ELSE NULL::varchar
        END AS pred_ts
   FROM pg_statistic s
   JOIN pg_class c ON c.oid = s.starelid
   JOIN pg_namespace ns ON c.relnamespace = ns.oid
   JOIN pg_attribute a ON c.oid = a.attrelid AND a.attnum = s.staattnum)
SELECT schema_name, table_name, col_num, col_name,
       pred_ts NOT LIKE '2000-01-01%' AS is_predicate,
       CASE WHEN pred_ts NOT LIKE '2000-01-01%' THEN (split_part(pred_ts,
 '||',1))::timestamp ELSE NULL::timestamp END as first_predicate_use,
       CASE WHEN pred_ts NOT LIKE '%||2000-01-01%' THEN (split_part(pred_ts,
 '||',2))::timestamp ELSE NULL::timestamp END as last_analyze
FROM predicate_column_info;
```

Weitere Informationen finden Sie unter Analysieren von Tabellen.

Aktivieren von Short Query Acceleration

Wenn Sie Short Query Acceleration (SQA) verwenden, werden ausgewählte, kurze Abfragen gegenüber Abfragen mit einer höheren Dauer priorisiert. SQA führt kurze Abfragen an einer dedizierten Stelle aus, sodass SQA-Abfragen nicht hinter längeren Abfragen in Warteschlangen eingereiht werden. SQA priorisiert nur Warteschlangen, die über eine kurze Zeit ausgeführt werden und sich in einer benutzerdefinierten Warteschlange befinden. Mit SQA werden kürzere Abfragen schneller ausgeführt und der Benutzer sieht schneller Ergebnisse.

Wenn Sie SQA aktivieren, können Sie dedizierte Workload Management (WLM)-Warteschlangen für kurze Anfragen reduzieren oder ganz abschaffen. Außerdem konkurrieren lange Abfragen nicht mit kurzen Abfragen um Slots in einer Warteschlange, daher können Sie Ihre WLM-Warteschlangen mit weniger Abfrage-Slots konfigurieren. Wenn Sie einen niedrigeren Nebenläufigkeitswert verwenden, wird bei den meisten Arbeitslasten der Abfragedurchsatz erhöht und die gesamte Systemleistung verbessert. Weitere Informationen finden Sie unter Arbeiten mit Short Query Acceleration.

Analyse

Der Advisor sucht nach Workloadmustern und meldet die Anzahl kürzlich ausgeführter Abfragen, bei denen SQA die Latenz und die Tages-Warteschlangendauer für SQA-qualifizierte Abfragen reduzieren würde.

Empfehlung

Ändern Sie die WLM-Konfiguration, um SQA zu aktivieren. Amazon Redshift verwendet einen Machine-Learning-Algorithmus, um berechtigte Abfragen zu analysieren. Voraussagen werden in dem Maße besser, in dem SQA aus den Abfragemustern lernt. Weitere Informationen finden Sie unter Workload-Management-Konfiguration.

Wenn Sie SQA aktivieren, legt WLM die maximale Ausführungszeit für kurze Abfragen standardmäßig als „dynamisch“ fest. Sie sollten die dynamische Einstellung für die maximale SQA-Laufzeit beibehalten.

Implementierungstipps

Führen Sie die folgende Abfrage aus, um zu überprüfen, ob SQA aktiviert ist. Wenn für die Abfrage eine Zeile zurückgegeben wird, ist SQA aktiviert.

```
select * from stv_wlm_service_class_config
where service_class = 14;
```

Weitere Informationen finden Sie unter SQA-Überwachung.

Ändern der Verteilungsschlüssel in Tabellen

Amazon Redshift verteilt Tabellenzeilen im Cluster gemäß des Verteilungsstils der Tabelle. In Tabellen mit KEY-Verteilung muss eine Spalte als Verteilungsschlüssel (DISTKEY) gekennzeichnet sein. Eine Tabellenzeile wird einem Knoten-Slice eines Clusters basierend auf dessen DISTKEY-Spaltenwert zugewiesen.

Ein passender DISTKEY platziert eine vergleichbare Anzahl an Zeilen in jedem Knoten-Slice und wird regelmäßig in Join-Bedingungen referenziert. Ein optimierter Join tritt auf, wenn Tabellen anhand ihrer DISTKEY-Spalten verbunden werden, um die Abfrageleistung zu erhöhen.

Analyse

Advisor analysiert den Workload Ihres Clusters, um den geeignetsten Verteilungsschlüssel für die Tabellen zu finden, der erheblich von einem KEY-Verteilungsstil profitieren kann.

Empfehlung

Advisor stellt ALTER TABLE Anweisungen bereit, die die Werte DISTSTYLE und DISTKEY einer Tabelle basierend auf deren Analyse anpassen. Um einen erheblichen Leistungsvorteil zu erzielen, müssen alle SQL-Anweisungen innerhalb einer Empfehlungsgruppe implementiert werden.

Die Neuverteilung großer Tabellen mit ALTER TABLE verbraucht Cluster-Ressourcen, außerdem muss die Tabelle an verschiedenen Zeitpunkten kurzfristig gesperrt werden. Implementieren Sie die jeweiligen Empfehlungsgruppen, wenn der sonstige Workload im Cluster gering ist. Weitere Informationen zur Optimierung der Tabellenverteilungseigenschaften finden Sie unter Amazon Redshift Engineering's Advanced Table Design Playbook: Distribution Styles and Distribution Keys.

Weitere Informationen zu ALTER DISTSTYLE und DISTKEY finden Sie unter ALTER TABLE.

> **Note**
>
> Wenn keine Empfehlung angezeigt wird, bedeutet dies nicht unbedingt, dass die aktuellen Verteilungsstile am besten geeignet sind. Advisor gibt keine Empfehlungen, wenn nicht genügend Daten vorhanden sind oder der erwartete Nutzen der Umverteilung gering ist. Advisor-Empfehlungen beziehen sich auf eine bestimmte Tabelle und können nicht auf Tabellen mit identischem Spaltennamen übertragen werden. Tabellen, die denselben Spaltennamen verwenden, können andere Eigenschaften für diese Spalten haben, sofern sich die darin enthaltenen Daten unterscheiden.
> Wenn Sie Empfehlungen für Staging-Tabellen sehen, die von ETL-Aufgaben erstellt oder verworfen wurden, passen Sie Ihre ETL-Prozesse an, um von Advisor empfohlene Verteilungsschlüssel zu verwenden.

Ändern der Sortierschlüssel für Tabellen

Amazon Redshift sortiert Tabellenzeilen nach dem Tabellensortierschlüssel. Die Sortierung der Tabellenzeilen basiert auf den Sortierschlüsselspaltenwerten.

Das Sortieren einer Tabelle mit einem geeigneten Sortierschlüssel kann die Leistung von Abfragen beschleunigen, insbesondere solche mit bereichsbeschränkten Prädikaten, da weniger Tabellenblöcke von der Festplatte gelesen werden müssen.

Analyse

Advisor analysiert die Workload Ihres Clusters über mehrere Tage, um einen vorteilhaften Sortierschlüssel für Ihre Tabellen zu identifizieren.

Empfehlung

Advisor stellt zwei Gruppen von ALTER TABLE-Anweisungen bereit, die den Sortierschlüssel einer Tabelle basierend auf ihrer Analyse ändern:

- Anweisungen, die eine Tabelle ändern, die derzeit keinen Sortierschlüssel zum Hinzufügen eines COMPOUND-Sortierschlüssels hat.
- Anweisungen, die einen Sortierschlüssel von INTERLEAVED zu COMPOUND oder keinen Sortierschlüssel verändern.

  Die Verwendung zusammengesetzter Sortierschlüssel verringert den Verwaltungsaufwand bei der Wartung deutlich. Tabellen mit zusammengesetzten Sortierschlüsseln benötigen nicht die teuren VACUUM REINDEX-Operationen, die bei überlappenden Sortierschlüsseln erforderlich sind. In der Praxis sind für die große Mehrheit der Amazon-Redshift-Workloads zusammengesetzte Sortierschlüssel effizienter als überlappende. Wenn eine Tabelle jedoch klein ist, ist der Verzicht auf einen Sortierschlüssel effizienter, um den Speicheraufwand für Sortierschlüssel zu vermeiden.

Beim Sortieren einer großen Tabelle mit ALTER TABLE werden Clusterressourcen belegt und zu unterschiedlichen Zeitpunkten sind Tabellensperren erforderlich. Implementieren Sie jede Empfehlung, wenn die Workload eines Clusters moderat ist. Weitere Details zur Optimierung von Tabellensortierschlüsselkonfigurationen finden Sie im Amazon Redshift Engineering's Advanced Table Design Playbook: Compound and Interleaved Sort Keys.

Weitere Informationen zu ALTER SORTKEY finden Sie im Abschnitt ALTER TABLE.

> **Note**
>
> Wenn Sie keine Empfehlung für eine Tabelle sehen, bedeutet dies nicht unbedingt, dass die aktuelle Konfiguration die beste ist. Advisor gibt keine Empfehlungen, wenn nicht genügend Daten vorhanden sind oder der erwartete Nutzen der Sortierung gering ist. Advisor-Empfehlungen gelten für eine bestimmte Tabelle und gelten nicht unbedingt für eine Tabelle, die eine Spalte mit demselben Namen und Datentyp enthält. Für Tabellen, die Spaltennamen gemeinsam nutzen, können basierend auf den Daten in den Tabellen und der Workload unterschiedliche Empfehlungen gelten.

Ändern der Komprimierungskodierungen für Spalten

Komprimierung ist eine Operation auf Spaltenebene, die die Größe von Daten reduziert, wenn sie gespeichert werden. Die Komprimierung wird in Amazon Redshift verwendet, um Speicherplatz zu sparen und die Abfrageleistung zu verbessern, indem die Menge an Festplatten-I/O reduziert wird. Wir empfehlen eine optimale Komprimierungskodierung für jede Spalte basierend auf ihrem Datentyp

und auf Abfragemustern. Bei optimaler Komprimierung können Abfragen effizienter ausgeführt werden und die Datenbank beansprucht nur wenig Speicherplatz.

Analyse

Advisor führt kontinuierlich eine Analyse der Workload und des Datenbankschemas Ihres Clusters durch, um die optimale Komprimierungskodierung für jede Tabellenspalte zu ermitteln.

Empfehlung

Advisor stellt ALTER-TABLE-Anweisungen bereit, die die Komprimierungskodierung bestimmter Spalten basierend auf ihrer Analyse ändern.

Ändern der Spaltenkomprimierungskodierungen mit ALTER TABLE verbraucht Clusterressourcen und erfordert Tabellensperren zu verschiedenen Zeitpunkten. Implementieren Sie am besten Empfehlungen, wenn der Workload im Cluster leicht ausfällt.

Als Referenz zeigt Beispiele für ALTER TABLE mehrere Anweisungen an, die die Kodierung für eine Spalte ändern.

> **Note**
>
> Advisor gibt keine Empfehlungen, wenn nicht genügend Daten vorhanden sind oder der erwartete Nutzen der Verschlüsselung gering ist.

Datentyp-Empfehlungen

Amazon Redshift verfügt über eine Bibliothek mit SQL-Datentypen für verschiedene Anwendungsfälle. Dazu gehören ganzzahlige Typen wie `INT` und Typen zum Speichern von Zeichen wie `VARCHAR`. Redshift speichert Typen optimiert, um einen schnelleren Zugriff und eine gute Abfrageleistung zu ermöglichen. Redshift bietet außerdem Funktionen für bestimmte Typen, mit denen Sie Berechnungen für Abfrageergebnisse formatieren oder durchführen können.

Analyse

Advisor führt kontinuierlich eine Analyse der Workload und des Datenbankschemas Ihres Clusters durch, um Spalten zu ermitteln, die erheblich von einer Datentypänderung profitieren können.

Empfehlung

Advisor bietet eine `ALTER TABLE`-Anweisung, die eine neue Spalte mit dem vorgeschlagenen Datentyp hinzufügt. Eine begleitende `UPDATE`-Anweisung kopiert Daten aus der vorhandenen Spalte in die neue Spalte. Nachdem Sie die neue Spalte erstellt und die Daten geladen haben, ändern Sie Ihre Abfragen und Erfassungsskripte, um auf die neue Spalte zuzugreifen. Nutzen Sie dann Ressourcen und Funktionen, die auf den neuen Datentyp spezialisiert sind, in SQL-Funktionsreferenz.

Das Kopieren vorhandener Daten in die neue Spalte kann einige Zeit dauern. Wir empfehlen Ihnen, jede Advisor-Empfehlung zu implementieren, wenn die Workload des Clusters leicht ist. Verweisen Sie auf die Liste der verfügbaren Datentypen unter Datentypen.

Beachten Sie, Advisor gibt keine Empfehlungen, wenn nicht genügend Daten vorhanden sind oder der erwartete Nutzen der Veränderung des Datentyps gering ist.

Tutorials für Amazon Redshift

Befolgen Sie die Schritte in diesen Tutorials, um mehr über Amazon-Redshift-Funktionen zu erfahren:

- Tutorial: So laden Sie Daten aus Amazon S3
- Tutorial: Abfragen verschachtelter Daten mit Amazon Redshift Spectrum
- Tutorial: Konfigurieren von manuellen Workload-Management(WLM)-Warteschlangen
- Tutorial: Verwenden von Geo-SQL-Funktionen mit Amazon Redshift
- Tutorials für Amazon Redshift ML

Arbeiten mit automatischer Tabellenoptimierung

Die automatische Tabellenoptimierung ist eine Selbstoptimierungsfunktion, die automatisch das Design von Tabellen optimiert, indem Sortier- und Verteilungsschlüssel ohne Eingreifen eines Administrators angewendet werden. Durch die Automatisierung zur Optimierung des Designs von Tabellen können Sie loslegen und die schnellste Leistung erzielen, ohne Zeit investieren zu müssen, um Tabellenoptimierungen manuell zu erstellen und zu implementieren.

Die automatische Tabellenoptimierung beobachtet kontinuierlich, wie Abfragen mit Tabellen interagieren. Es verwendet fortschrittliche Methoden der künstlichen Intelligenz, um Sortier- und Verteilungsschlüssel auszuwählen, um die Leistung für den Workload des Clusters zu optimieren. Wenn Amazon Redshift feststellt, dass die Anwendung eines Schlüssels die Clusterleistung verbessert, werden Tabellen automatisch innerhalb von Stunden ab der Erstellung des Clusters geändert. Dies hat minimale Auswirkungen auf Abfragen.

Um diese Automatisierung nutzen zu können, erstellt ein Amazon-Redshift-Administrator eine neue Tabelle oder ändert eine vorhandene Tabelle, um die automatische Optimierung zu ermöglichen. Bestehende Tabellen mit einem Verteilungsstil oder einem AUTO-Sortierschlüssel sind bereits für die Automatisierung aktiviert. Wenn Sie Abfragen in diesen Tabellen ausführen, ermittelt Amazon Redshift, ob ein Sortierschlüssel oder ein Verteilungsschlüssel die Leistung verbessert. Wenn dies der Fall ist, ändert Amazon Redshift die Tabelle automatisch, ohne dass der Administrator eingreifen muss. Wenn eine Mindestanzahl von Abfragen ausgeführt wird, werden Optimierungen innerhalb von Stunden nach dem Start des Clusters angewendet.

Wenn Amazon Redshift feststellt, dass ein Verteilungsschlüssel die Leistung von Abfragen verbessert, kann der Verteilungsstil in Tabellen, in denen der Verteilungsstil AUTO lautet, zu KEY geändert werden.

Themen

- Definieren von Tabelleneinschränkungen

Aktivieren automatischer Tabellenoptimierung

Standardmäßig wird für Tabellen, die ohne explizite Definition von Sortierschlüsseln oder Verteilungsschlüsseln erstellt wurden, AUTO festgelegt. Zum Zeitpunkt der Tabellenerstellung können Sie auch explizit eine Sortierung oder einen Verteilungsschlüssel manuell festlegen. Wenn Sie den Sortier- oder Verteilungsschlüssel festlegen, wird die Tabelle nicht automatisch verwaltet.

Um eine vorhandene Tabelle automatisch zu optimieren, verwenden Sie die ALTER-Anweisungsoptionen, um die Tabelle zu AUTO zu ändern. Sie können die Automatisierung für Sortierschlüssel definieren, aber nicht für Verteilungsschlüssel (und umgekehrt). Wenn Sie eine ALTER-Anweisung ausführen, um eine Tabelle in eine automatisierte Tabelle zu konvertieren, werden vorhandene Sortierschlüssel und Verteilungsstile beibehalten.

```
ALTER TABLE table_name ALTER SORTKEY AUTO;
```

```
ALTER TABLE table_name ALTER DISTSTYLE AUTO;
```

Weitere Informationen finden Sie unter ALTER TABLE.

Anfangs hat eine Tabelle keinen Verteilungsschlüssel oder Sortierschlüssel. Der Verteilungsstil wird entweder auf EVEN oder ALL festgelegt, je nach Tischgröße. Mit zunehmender Größe der Tabelle wendet Amazon Redshift die optimalen Verteilungsschlüssel und Sortierschlüssel an. Optimierungen werden innerhalb von Stunden nach dem Ausführen einer minimalen Anzahl von Abfragen angewendet. Bei der Ermittlung von Sortierschlüsseloptimierungen versucht Amazon Redshift, die Datenblöcke zu optimieren, die während eines Tabellenscans von der Festplatte gelesen wurden. Bei der Bestimmung des Verteilungsstils versucht Amazon Redshift, die Anzahl der zwischen Clusterknoten übertragenen Bytes zu optimieren.

Entfernen der automatischen Tabellenoptimierung aus einer Tabelle

Sie können eine Tabelle aus der automatischen Optimierung entfernen. Beim Entfernen einer Tabelle aus der Automatisierung müssen Sie einen Sortierschlüssel oder einen Verteilungsstil auswählen. Geben Sie zum Ändern des Verteilungsstils einen bestimmten Verteilungsstil an.

```
ALTER TABLE table_name ALTER DISTSTYLE EVEN;
```

```
ALTER TABLE table_name ALTER DISTSTYLE ALL;
```

```
ALTER TABLE table_name ALTER DISTSTYLE KEY DISTKEY c1;
```

Um einen Sortierschlüssel zu ändern, können Sie einen Sortierschlüssel definieren oder keinen auswählen.

```
ALTER TABLE table_name ALTER SORTKEY(c1, c2);
```

```
ALTER TABLE table_name ALTER SORTKEY NONE;
```

Überwachung von Aktionen der automatischen Tabellenoptimierung

Die Systemansicht SVV_ALTER_TABLE_RECOMMENDATIONS zeichnet die aktuellen Empfehlungen von Amazon Redshift Advisor für Tabellen auf. Diese Ansicht zeigt Empfehlungen für alle Tabellen, für diejenigen, die für die automatische Optimierung definiert sind und für diejenigen, die es nicht sind.

Um anzuzeigen, ob eine Tabelle für die automatische Optimierung definiert ist, führen Sie eine Abfrage für die Systemansicht SVV_TABLE_INFO aus. Einträge werden nur für Tabellen angezeigt, die in der Datenbank der aktuellen Sitzung sichtbar sind. Empfehlungen werden zweimal täglich innerhalb von Stunden ab der Erstellung des Clusters in die Ansicht eingefügt. Nachdem eine Empfehlung verfügbar ist, wird sie innerhalb einer Stunde gestartet. Nachdem eine Empfehlung (entweder von Amazon Redshift oder von Ihnen) angewendet wurde, wird sie nicht mehr in der Ansicht angezeigt.

Die Systemansicht SVL_AUTO_WORKER_ACTION zeigt ein Überwachungsprotokoll aller von Amazon Redshift durchgeführten Aktionen und den vorherigen Status der Tabelle an.

Die Systemansicht SVV_TABLE_INFO listet alle Tabellen im System zusammen mit einer Spalte auf, die angibt, ob der Sortierschlüssel und der Verteilungsstil der Tabelle auf AUTO festgelegt ist.

Weitere Informationen zur Verwendung dieser Systemansichten finden Sie unter Systemüberwachung (nur bereitgestellt).

Arbeiten mit Spaltenkomprimierung

Komprimierung ist eine Operation auf Spaltenebene, die die Speichergröße von Daten reduziert. Die Komprimierung bewahrt Speicherplatz und reduziert die Größe der Daten, die aus dem Speicher gelesen werden. Dies reduziert die Menge der Datenträger-I/O und verbessert damit die Abfrageleistung.

ENCODE AUTO ist die Standardeinstellung für Tabellen. Wenn für eine Tabelle ENCODE AUTO festgelegt wird, verwaltet Amazon Redshift automatisch die Kompressionskodierung für alle Spalten in der Tabelle. Weitere Informationen erhalten Sie unter CREATE TABLE und ALTER TABLE.

Wenn Sie jedoch die Komprimierungskodierung für eine Spalte in der Tabelle angeben, wird die Tabelle nicht mehr auf ENCODE AUTO festgelegt. Amazon Redshift verwaltet nicht mehr automatisch die Komprimierungskodierung für alle Spalten in der Tabelle.

Sie können einen Komprimierungstyp oder encoding manuell auf die Tabellen anwenden, wenn Sie sie erstellen. Sie können auch den Befehl COPY verwenden, um die Komprimierung automatisch zu analysieren und anzuwenden. Weitere Informationen finden Sie unter Lassen Sie COPY die Kompressionskodierungen auswählen. Details zur Anwendung der automatischen Kompression finden Sie unter Laden von Tabellen mit automatischer Kompression.

> **Note**
>
> Es wird nachdrücklich empfohlen, den COPY-Befehl zu verwenden, um die automatische Kompression anzuwenden.

Sie können die Komprimierungskodierung manuell anwenden, wenn die neue Tabelle dieselben Datenmerkmale wie eine andere Tabelle aufweist. Sie können dies auch tun, wenn Sie während Tests feststellen, dass die während der automatischen Komprimierung angewendeten Kompressionskodierungen für Ihre Daten nicht optimal geeignet sind. Wenn Sie Kompressionskodierungen manuell anwenden, können Sie den Befehl ANALYZE COMPRESSION für eine bereits ausgefüllte Tabelle ausführen und die Ergebnisse verwenden, um die Kompressionskodierungen zu wählen.

Um die Kompression manuell anzuwenden, geben Sie die Kompressionskodierungen für einzelne Spalten als Teil der Anweisung CREATE TABLE an. Die Syntax ist wie folgt.

```
CREATE TABLE table_name (column_name
```

```
data_type ENCODE encoding-type)[, ...]
```

Wobei encoding-type der Schlüsselworttabelle im folgenden Abschnitt entnommen ist.

Die folgende Anweisung erstellt beispielsweise eine Tabelle mit zwei Spalten, PRODUCT. Wenn Daten in die Tabelle geladen werden, wird nicht die Spalte PRODUCT_ID komprimiert, sondern die Spalte PRODUCT_NAME. Dabei wird die Byte-Verzeichnis-Kodierung (BYTEDICT) verwendet.

```
create table product(
product_id int encode raw,
product_name char(20) encode bytedict);
```

Sie können die Kodierung für eine Spalte angeben, wenn sie einer Tabelle mittels des Befehls ALTER TABLE hinzugefügt wird.

```
ALTER TABLE table-name ADD [ COLUMN ] column_name column_type ENCODE encoding-type
```

Themen

- Kompressionskodierungen
- Testen der Kompressionskodierungen
- Beispiel: Wahl der Kompressionskodierungen für die Tabelle CUSTOMER

Kompressionskodierungen

Eine Komprimierungskodierung gibt die Art der Komprimierung an, die auf eine Spalte von Datenwerten angewendet wird, während einer Tabelle Zeilen hinzugefügt werden.

ENCODE AUTO ist die Standardeinstellung für Tabellen. Wenn für eine Tabelle ENCODE AUTO festgelegt wird, verwaltet Amazon Redshift automatisch die Kompressionskodierung für alle Spalten in der Tabelle. Weitere Informationen erhalten Sie unter CREATE TABLE und ALTER TABLE.

Wenn Sie jedoch die Komprimierungskodierung für eine Spalte in der Tabelle angeben, wird die Tabelle nicht mehr auf ENCODE AUTO festgelegt. Amazon Redshift verwaltet nicht mehr automatisch die Komprimierungskodierung für alle Spalten in der Tabelle.

Bei Verwendung von CREATE AUTO wird ENCODE AUTO deaktiviert, wenn Sie für eine Spalte in der Tabelle die Kompressionskodierung festlegen. Wenn ENCODE AUTO deaktiviert ist, weist

Amazon Redshift den Spalten, für die Sie keinen ENCODE-Typ angeben, automatisch wie folgt eine anfängliche Kompressionskodierung zu:

- Spalten, die als Sortierschlüssel definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als die Datentypen BOOLEAN, REAL oder DOUBLE PRECISION definiert sind, wird die RAW-Kodierung zugewiesen.
- Spalten, die als Datentypen SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP oder TIMESTAMPTZ definiert sind, wird die AZ64-Komprimierung zugewiesen.
- Spalten, die als CHAR- oder VARCHAR-Datentypen definiert sind, wird LZO-Komprimierung zugewiesen.

Sie können die Kodierung einer Tabelle nach der Erstellung ändern, indem Sie ALTER TABLE verwenden. Wenn Sie ENCODE AUTO mithilfe von ALTER TABLE deaktivieren, verwaltet Amazon Redshift die Kompressionskodierungen für Ihre Spalten nicht mehr automatisch. Alle Spalten behalten so lange die Kompressionskodierungstypen bei, die sie bei der Deaktivierung von ENCODE AUTO aufwiesen, bis Sie sie ändern oder ENCODE AUTO erneut aktivieren.

In der folgenden Tabelle werden die unterstützten Kompressionskodierungen und die Datentypen, die die Kodierung unterstützen, aufgelistet.

Kodierungstyp	Schlüsselwort in CREATE TABLE und ALTER TABLE	Datentypen
Roh (keine Kompression)	RAW	Alle
AZ64	AZ64	SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP, TIMESTAMPTZ
Byte-Verzeichnis	BYTEDICT	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMP TZ
Delta	DELTA DELTA32K	SMALLINT, INT, BIGINT, DATE, TIMESTAMP, DECIMAL

Kodierungstyp	Schlüsselwort in CREATE TABLE und ALTER TABLE	Datentypen
		INT, BIGINT, DATE, TIMESTAMP, DECIMAL
LZO	LZO	SMALLINT, INTEGER, BIGINT, DECIMAL, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMP TZ, SUPER
Mostlyn	MOSTLY8	SMALLINT, INT, BIGINT, DECIMAL
	MOSTLY16	INT, BIGINT, DECIMAL
	MOSTLY32	BIGINT, DECIMAL
Run-length	RUNLENGTH	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMPTZ
Text	TEXT255	Nur VARCHAR
	TEXT32K	Nur VARCHAR
Zstandard	ZSTD	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMPTZ, SUPER

Rohkodierung

Die Rohkodierung ist die Standardkodierung für Spalten, die als Sortierschlüssel identifiziert sind, und für Spalten, die als die Datentypen BOOLEAN, REAL oder DOUBLE PRECISION definiert sind. Bei einer Rohkodierung werden die Daten in roher, nicht komprimierter Form gespeichert.

AZ64-Kodierung

AZ64 ist Amazons eigener Komprimierungscodierungsalgorithmus, der dazu dient, ein hohes Komprimierungsverhältnis zu erzielen und die Abfrageleistung zu verbessern. Im Kern komprimiert der AZ64-Algorithmus kleinere Gruppen von Datenwerten und verwendet SIMD- (Single Instruction, Multiple Data) Anweisungen für die parallele Verarbeitung. Verwenden Sie AZ64 für deutliche Speicherplatzeinsparungen und hohe Leistung bei numerischen, Datums- und Uhrzeitdaten.

Sie können AZ64 als Komprimierungscodierung bei der Definition von Spalten mit den Anweisungen CREATE TABLE und ALTER TABLE für die folgenden Datentypen verwenden:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- DATUM
- TIMESTAMP
- TIMESTAMPTZ

Byte-Verzeichnis-Kodierung

Bei einer Byte-Verzeichnis-Kodierung wird für jeden Block von Spaltenwerten auf dem Datenträger ein getrenntes Verzeichnis eindeutiger Werte erstellt. (Ein Amazon-Redshift-Datenträgerblock belegt 1 MB.) Das Verzeichnis enthält bis zu 256 Einzelbyte-Werten, die als Indizes für die ursprünglichen Datenwerte gespeichert werden. Wenn in einem einzelnen Block mehr als 256 Werte gespeichert werden, werden die zusätzlichen Werte in roher, nicht komprimierter Form in den Block geschrieben. Dieser Vorgang wird für jeden Datenträgerblock wiederholt.

Diese Kodierung ist bei Zeichenfolgenspalten mit niedriger Kardinalität sehr effektiv. Diese Kodierung ist optimal, wenn die Datendomäne einer Spalte weniger als 256 eindeutige Werte enthält.

Für Spalten mit dem Zeichenfolgendatentyp (CHAR und VARCHAR), die mit BYTEDICT codiert sind, führt Amazon Redshift vektorisierte Scans und Prädikatauswertungen durch, die direkt über komprimierte Daten arbeiten. Diese Scans verwenden hardwarespezifische Anweisungen des Typs Single Instruction and Multiple Data (SIMD) für die parallele Verarbeitung. Dadurch wird das Scannen von Zeichenfolgenspalten erheblich beschleunigt. Die Byte-Verzeichnis-Kodierung ist besonders platzeffizient, wenn eine CHAR/VARCHAR-Spalte lange Zeichenfolgen enthält.

Angenommen eine Tabelle besitzt eine Spalte COUNTRY mit einem Datentyp CHAR(30). Während die Daten geladen werden, erstellt Amazon Redshift das Verzeichnis und füllt die Spalte COUNTRY mit dem Indexwert aus. Das Verzeichnis enthält die indizierten eindeutigen Werte. Die Tabelle selbst enthält nur die Einzelbyte-Subskripts der entsprechenden Werte.

> **Note**
>
> Leerzeichen am Ende werden im Fall von Zeichenspalten mit fester Länge gespeichert. Daher spart in einer CHAR(30)-Spalte jeder komprimierte Wert 29 Bytes an Speicher, wenn Sie die Byte-Verzeichnis-Kodierung verwenden.

Die folgende Tabelle stellt das Verzeichnis für die Spalte COUNTRY dar.

Eindeutiger Datenwert	Verzeichnisindex	Größe (feste Länge, 30 Bytes pro Wert)
England	0	30
United States of America	1	30
Venezuela	2	30
Sri Lanka	3	30
Argentina	4	30
Japan	5	30
Gesamtsumme		180

Die folgende Tabelle stellt die Werte in der Spalte COUNTRY dar.

Ursprünglicher Datenwert	Ursprüngliche Größe (feste Länge, 30 Bytes pro Wert)	Komprimierter Wert (Index)	Neue Größe (Bytes)
England	30	0	1

Ursprünglicher Datenwert	Ursprüngliche Größe (feste Länge, 30 Bytes pro Wert)	Komprimierter Wert (Index)	Neue Größe (Bytes)
England	30	0	1
United States of America	30	1	1
United States of America	30	1	1
Venezuela	30	2	1
Sri Lanka	30	3	1
Argentina	30	4	1
Japan	30	5	1
Sri Lanka	30	3	1
Argentina	30	4	1
Gesamtsumme	300		10

Die gesamte komprimierte Größe in diesem Beispiel wird wie folgt berechnet: Im Verzeichnis sind 6 verschiedene Einträge gespeichert (6 * 30 = 180) und die Tabelle enthält 10 komprimierte Werte mit 1 Byte. Dies sind insgesamt 190 Bytes.

Delta-Kodierung

Delta-Kodierungen sind für Datum-/Uhrzeitspalten sehr nützlich.

Bei der Delta-Kodierung werden Daten komprimiert, indem der Unterschied zwischen Werten aufgezeichnet wird, die in der Spalte aufeinander folgen. Dieser Unterschied wird für jeden Block von Spaltenwerten auf dem Datenträger in einem getrennten Verzeichnis aufgezeichnet. (Ein Amazon-Redshift-Datenträgerblock belegt 1 MB.) Angenommen, die Spalte enthält 10 ganze Zahlen in der Reihenfolge zwischen 1 und 10. Die ersten werden als 4-Byte-Ganzzahl (plus ein 1-Byte-Flag)

gespeichert. Die nächsten neun werden jeweils als Byte mit dem Wert 1 gespeichert, was darauf hinweist, dass es um eins größer als der vorherige Wert ist.

Die Delta-Kodierung besitzt zwei Varianten:

- DELTA zeichnet die Unterschiede als 1-Byte-Werte auf (8-Bit-Ganzzahlen).
- DELTA32K zeichnet die Unterschiede als 2-Byte-Werte auf (16-Bit-Ganzzahlen).

Wenn die meisten Werte in der Spalte unter Verwendung eines einzelnen Bytes komprimiert werden könnten, ist die 1-Byte-Variante sehr effektiv. Wenn die Unterschiede größer werden, ist diese Kodierung schlimmstenfalls etwas weniger effektiv als das Speichern der nicht komprimierten Daten. Eine ähnliche Logik gilt für die 16-Bit-Version.

Wenn der Unterschied zwischen zwei Werten den 1-Byte-Bereich (DELTA) oder den 2-Byte-Bereich () überschreitet, wird der ursprüngliche Wert vollständig wiederhergestellt. Ihm wird ein 1-Byte-Flag vorangestellt. Der 1-Byte-Bereich liegt zwischen -127 und 127. Der 2-Byte-Bereich liegt zwischen -32000 und 32000.

Die folgende Tabelle zeigt, wie eine Delta-Kodierung für eine numerische Spalte funktioniert:

Ursprünglicher Datenwert	Ursprüngliche Größe (Bytes)	Unterschied (Delta)	Komprimierter Wert	Komprimierte Größe (Bytes)
1	4		1	1+4 (Flag + tatsächlicher Wert)
5	4	4	4	1
50	4	45	45	1
200	4	150	150	1+4 (Flag + tatsächlicher Wert)
185	4	-15	-15	1
220	4	35	35	1

Ursprünglicher Datenwert	Ursprüngliche Größe (Bytes)	Unterschied (Delta)	Komprimierter Wert	Komprimierte Größe (Bytes)
221	4	1	1	1
Gesamt	28			15

LZO-Kodierung

Die LZO-Kodierung stellt ein sehr hohes Kompressionsverhältnis mit guter Leistung bereit. Die LZO-Kodierung funktioniert besonders gut für CHAR- und VARCHAR-Spalten, die sehr lange Zeichenfolgen speichern. Sie eignet sich besonders gut für Freiformtexte wie Produktbeschreibungen, Benutzerkommentare oder JSON-Zeichenfolgen.

Mostly-Kodierung

Mostly-Kodierungen sind nützlich, wenn der Datentyp für eine Spalte größer ist, als es die meisten gespeicherten Werte erfordern. Durch die Angabe einer Mostly-Kodierung für diese Art von Spalten können Sie die Mehrzahl der Werte in der Spalte zu einer kleineren Standardspeichergröße komprimieren. Die übrigen Werte, die nicht komprimiert werden können, werden in ihrer rohen Form gespeichert. Sie können beispielsweise eine 16-Bit-Spalte wie eine INT2-Spalte zu einer 8-Bit-Speichergröße komprimieren.

Im Allgemeinen funktionieren Mostly-Kodierungen mit den folgenden Datentypen:

- SMALLINT/INT2 (16-Bit)
- INTEGER/INT (32-Bit)
- BIGINT/INT8 (64-Bit)
- DECIMAL/NUMERIC (64-Bit)

Wählen Sie die Variante der Mostly-Kodierung, die der Größe des Datentyps für die Spalte am besten entspricht. Sie können beispielsweise MOSTLY8 auf eine Spalte anwenden, die als 16-Bit-Ganzzahlspalte definiert ist. Die Anwendung von MOSTLY16 auf eine Spalte mit einem 16-Bit-Datentyp oder MOSTLY32 auf eine Spalte mit einem 32-Bit-Datentyp ist nicht zulässig.

Mostly-Kodierungen sind möglicherweise weniger effektiv als keine Kompression, wenn eine vergleichsweise große Zahl der Werte in der Spalte nicht komprimiert werden kann. Bevor Sie eine

dieser Kodierungen auf eine Spalte anwenden, führen Sie eine Überprüfung durch. Die meisten Werte, die Sie jetzt (und wahrscheinlich in der Zukunft) laden, sollten in die in der folgenden Tabelle gezeigten Bereiche passen.

Codierung	Komprimierte Speicherg rröße	Bereich von Werten, die komprimie rt werden können (Werte außerhalb des Bereichs werden nicht komprimiert gespeichert).
MOSTLY8	1 Byte (8 Bits)	-128 bis +127
MOSTLY16	2 Bytes (16 Bits)	-32768 bis +32767
MOSTLY32	4 Bytes (32 Bits)	-2147483648 bis +2147483647

> **Note**
>
> Ignorieren Sie bei Dezimalwerten das Dezimalzeichen, um zu ermitteln, ob der Wert dem Bereich entspricht. Beispielsweise wird 1.234,56 als 123.456 behandelt und kann in einer MOSTLY32-Spalte komprimiert werden.

Die Spalte VENUEID in der Tabelle VENUE ist beispielsweise als nicht komprimierte Ganzzahlspalte definiert. Das bedeutet, dass ihre Werte 4 Bytes Speicher verbrauchen. Der aktuelle Bereich von Werten in der Spalte ist **`0`** bis **`309`**. Daher würde der Speicherplatz für jeden Wert in dieser Spalte auf 2 Bytes reduziert werden, wenn diese Tabelle mit MOSTLY16-Kodierung für VENUEID neu erstellt und geladen würde.

Wenn die VENUEID-Werte, die in einer anderen Tabelle referenziert werden, überwiegend zwischen 0 und 127 liegen, wäre es sinnvoll, diese Fremdschlüsselspalte als MOSTLY8 zu kodieren. Bevor Sie diese Wahl treffen, führen Sie mehrere Abfragen für die referenzierenden Tabellendaten aus, um festzustellen, ob die Werte überwiegend im 8-Bit-, 16-Bit- oder 32-Bit-Bereich liegen.

Die folgende Tabelle zeigt komprimierte Größen für spezifische numerische Werte, wenn die MOSTLY8-, MOSTLY16- und MOSTLY32-Kodierungen verwendet werden:

Ursprünglicher Wert	Ursprüngliche INT- oder BIGINT-Größe (Bytes)	Größe mit MOSTLY8-Komprimierung (Bytes)	Größe mit MOSTLY16-Komprimierung (Bytes)	Größe mit MOSTLY32-Komprimierung (Bytes)
1	4	1	2	4
10	4	1	2	4
100	4	1	2	4
1000	4	Identisch mit der Größe der nicht komprimierten Daten	2	4
10000	4		2	4
20000	4		2	4
40000	8		Identisch mit der Größe der nicht komprimierten Daten	4
100000	8			4
2000000000	8			4

Kodierung der Run-length

Die Run-length-Kodierung ersetzt einen Wert, der in Folge wiederholt wird, durch ein Token, das aus dem Wert und einer Zahl für die Anzahl der Wiederholungen in Folge (der Länge des Laufs) besteht. Für jeden Block von Spaltenwerten auf dem Datenträger wird ein getrenntes Verzeichnis eindeutiger Werte erstellt. (Ein Amazon-Redshift-Datenträgerblock belegt 1 MB.) Diese Kodierung ist am besten für eine Tabelle geeignet, in der Datenwerte häufig in Folge wiederholt werden, beispielsweise, wenn die Tabelle nach diesen Werten sortiert ist.

Angenommen, eine Spalte in einer großen Dimensionstabelle hat eine vorhersehbare kleine Domäne, wie eine COLOR-Spalte mit weniger als 10 möglichen Werten. Diese Werte werden wahrscheinlich in langen Sequenzen in der gesamten Tabelle angezeigt, auch wenn die Daten nicht sortiert sind.

Es wird nicht empfohlen, die Run-length-Kodierung auf eine Spalte anzuwenden, die als Sortierschlüssel definiert ist. Scans mit eingeschränkten Bereichen funktionieren besser, wenn Blöcke

ähnliche Zahlen von Zeilen enthalten. Wenn Sortierschlüsselspalten sehr viel stärker als andere Spalten in derselben Abfrage komprimiert werden, zeigen Scans mit eingeschränkten Bereichen möglicherweise eine schlechte Leistung.

In der folgenden Tabelle wird die Spalte COLOR verwendet, um zu zeigen, wie die Run-length-Kodierung funktioniert:

Ursprünglicher Datenwert	Ursprüngliche Größe (Bytes)	Komprimierter Wert (Token)	Komprimierte Größe (Bytes)
Blue	4	{2,Blau}	5
Blue	4		0
Green	5	{3,Grün}	6
Green	5		0
Green	5		0
Blue	4	{1,Blue}	5
Yellow	6	{4,Yellow}	7
Yellow	6		0
Yellow	6		0
Yellow	6		0
Gesamtsumme	51		23

Text255- und Text32k-Kodierungen

Text255- und Text32k-Kodierungen sind für die Komprimierung von VARCHAR-Spalten nützlich, in denen dieselben Wörter häufig wiederholt werden. Für jeden Block von Spaltenwerten auf dem Datenträger wird ein getrenntes Verzeichnis eindeutiger Wörter erstellt. (Ein Amazon-Redshift-Datenträgerblock belegt 1 MB.) Das Verzeichnis enthält die ersten 245 eindeutigen Wörter in der Spalte. Diese Wörter werden auf dem Datenträger durch einen Einzelbyte-Indexwert ersetzt, der einen der 245 Werte darstellt. Wörter, die im Verzeichnis nicht dargestellt werden,

werden nicht komprimiert gespeichert. Dieser Vorgang wird für jeden Datenträgerblock von 1 MB wiederholt. Wenn die indizierten Wörter in der Spalte häufig vorkommen, führt dies zu einem hohen Komprimierungsverhältnis für die Spalte.

Für die Text32k-Kodierung gilt dasselbe Prinzip. Das Verzeichnis für die einzelnen Blöcke erfasst jedoch keine bestimmte Anzahl von Wörtern. Stattdessen indiziert das Verzeichnis jedes gefundene eindeutige Wort, bis die kombinierten Einträge eine Länge von 32K abzüglich etwas Overhead erreichen. Die indizierten Werte werden in zwei Bytes gespeichert.

Betrachten Sie beispielsweise die Spalte VENUENAME in der Tabelle VENUE. Wörter wie **`Arena`**, **`Center`** und **`Theatre`** wiederholen sich in dieser Spalte und befinden sich wahrscheinlich unter den ersten 245 Wörtern in jedem Block, wenn die Text255-Kompression angewendet wird. Wenn ja, profitiert diese Spalte von der Komprimierung. Jedes Mal, wenn diese Wörter erscheinen, belegen sie nur 1 Byte Speicher (anstatt 5, 6 bzw. 7 Bytes).

Zstandard-Kodierung

Die Zstandard (ZSTD)-Kodierung stellt ein hohes Kompressionsverhältnis mit sehr guter Leistung über unterschiedliche Datensätze hinweg bereit. ZSTD funktioniert besonders gut für CHAR- und VARCHAR-Spalten, die eine große Vielzahl langer und kurzer Zeichenfolgen speichern, wie Produktbeschreibungen, Benutzerkommentare, Protokolle und JSON-Zeichenfolgen. Während Algorithmen wie die Delta-Kodierung oder die Mostly-Kodierung möglicherweise mehr Speicherplatz belegen als die entsprechenden nicht komprimierten Daten, ist es sehr unwahrscheinlich, dass ZSTD die Datenträgernutzung steigert.

ZSTD unterstützt die Datentypen SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP und TIMESTAMPTZ.

Testen der Kompressionskodierungen

Wenn Sie die Spaltenkodierungen manuell angeben, sollten Sie die verschiedenen Kodierungen mit Ihren Daten testen.

> **Note**
>
> Es wird empfohlen, den COPY-Befehl zu verwenden, um Daten wann immer möglich zu laden, und dem COPY-Befehl zu gestatten, die optimalen Kodierungen basierend auf Ihren Daten zu wählen. Alternativ können Sie den ANALYZE COMPRESSION-Befehl verwenden, um die für vorhandene Daten vorgeschlagenen Kodierungen anzuzeigen. Details

zur Anwendung der automatischen Kompression finden Sie unter Laden von Tabellen mit automatischer Kompression.

Um einen relevanten Test der Datenkomprimierung ausführen zu können, müssen Sie über eine große Zahl von Zeilen verfügen. Für die Zwecke dieses Beispiels erstellen wir eine Tabelle, in die unter Verwendung einer Anweisung, die Daten aus den beiden Tabellen VENUE und LISTING auswählt, Zeilen eingefügt werden. Wir lassen die WHERE-Klausel aus, die normalerweise die beiden Tabellen verbinden würde. Daher wird für jede Zeile in der Tabelle VENUE ein Join für alle Zeilen in der Tabelle LISTING ausgeführt, insgesamt mehr als 32 Millionen Zeilen. Diese Art von Join ist als kartesischer Join bekannt und wird normalerweise nicht empfohlen. Für die vorliegenden Zwecke ist dies jedoch eine praktische Methode, um eine große Zahl von Zeilen zu erstellen. Wenn Sie bereits eine Tabelle mit Daten besitzen, die Sie testen möchten, können Sie diesen Schritt überspringen.

Nachdem wir eine Tabelle mit Beispieldaten erhalten, erstellen wir eine Tabelle mit sieben Spalten. Jede Komprimierungskodierung hat eine andere Komprimierungskodierung: raw, bytedict, lzo, run length, text255, text32k und zstd. Jede Spalte wird mit genau den gleichen Daten ausgefüllt, indem ein INSERT-Befehl ausgeführt wird, der die Daten aus der ersten Tabelle auswählt.

Führen Sie zum Testen von Komprimierungskodierungen folgende Schritte aus:

1. (Optional) Verwenden Sie einen kartesischen Join, um eine Tabelle mit einer großen Zahl von Zeilen zu erstellen. Sie können diesen Schritt überspringen wenn Sie eine bereits vorhandene Tabelle testen möchten.

```
create table cartesian_venue(
venueid smallint not null distkey sortkey,
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer);

insert into cartesian_venue
select venueid, venuename, venuecity, venuestate, venueseats
from venue, listing;
```

2. Erstellen Sie als Nächstes eine Tabelle mit den Kodierungen, die Sie vergleichen möchten.

```
create table encodingvenue (
```

```
venueraw varchar(100) encode raw,
venuebytedict varchar(100) encode bytedict,
venuelzo varchar(100) encode lzo,
venuerunlength varchar(100) encode runlength,
venuetext255 varchar(100) encode text255,
venuetext32k varchar(100) encode text32k,
venuezstd varchar(100) encode zstd);
```

3. Fügen Sie die gleichen Daten in alle Spalten mithilfe einer INSERT-Anweisungen mit einer SELECT-Klausel ein.

```
insert into encodingvenue
select venuename as venueraw, venuename as venuebytedict, venuename as venuelzo,
 venuename as venuerunlength, venuename as  venuetext32k, venuename as  venuetext255,
 venuename as venuezstd
from cartesian_venue;
```

4. Überprüfen Sie die Anzahl der Zeilen in der neuen Tabelle.

```
select count(*) from encodingvenue

  count
----------
 38884394
(1 row)
```

5. Führen Sie eine Abfrage für die Systemtabelle STV_BLOCKLIST aus, um die Zahl der Datenträgerblöcke mit 1 MB zu vergleichen, die von den einzelnen Spalten verwendet werden.

   Die MAX-Aggregationsfunktion gibt für jede Spalte die höchste Blockzahl zurück. Die Tabelle STV_BLOCKLIST enthält Details für drei systemgenerierte Spalten. Dieses Beispiel verwendet `col < 6` in der WHERE-Klausel, um die systemgenerierten Spalten auszuschließen.

```
select col, max(blocknum)
from stv_blocklist b, stv_tbl_perm p
where (b.tbl=p.id) and name ='encodingvenue'
and col < 7
group by name, col
order by col;
```

   Die Abfrage gibt die folgenden Ergebnisse zurück. Die Spalten sind nummeriert, beginnend mit null. Je nach Konfiguration Ihres Clusters unterscheiden sich die Zahlen in Ihrem Ergebnis

möglicherweise. Die relativen Größen sollten jedoch ähnlich sein. Sie sehen, dass die BYTEDICT-Kodierung in der zweiten Spalte die besten Ergebnisse für diesen Datensatz generiert. Dieser Ansatz hat ein Komprimierungsverhältnis von mehr als 20:1. Die LZO und ZSTD-Kodierungen führten ebenfalls zu herausragenden Ergebnissen. Andere Datensätze liefern natürlich andere Ergebnisse. Wenn eine Spalte längere Textzeichenfolgen enthält, generiert LZO häufig die besten Kompressionsergebnisse.

```
 col | max
-----+-----
   0 | 203
   1 |  10
   2 |  22
   3 | 204
   4 |  56
   5 |  72
   6 |  20
(7 rows)
```

Wenn Sie über Daten in einer vorhandenen Tabelle verfügen, können Sie den ANALYZE COMPRESSION-Befehl verwenden, um die für die Tabelle vorgeschlagenen Kodierungen anzuzeigen. Die folgende Tabelle zeigt beispielsweise die empfohlene Kodierung für eine Kopie der Tabelle VENUE mit dem Namen CARTESIAN_VENUE, die 38 Millionen Zeilen enthält. Beachten Sie, dass ANALYZE COMPRESSION die LZO-Kodierung für die Spalte VENUENAME empfiehlt. ANALYZE COMPRESSION wählt die optimale Kompression auf der Basis mehrerer Faktoren aus. Zu diesen gehört der Prozentsatz der Reduktion. In diesem spezifischen Fall bietet BYTEDICT eine bessere Kompression, LZO führt jedoch ebenfalls zu einer Kompression von mehr als 90 Prozent.

```
analyze compression cartesian_venue;

Table          | Column     | Encoding | Est_reduction_pct
---------------+------------+----------+------------------
reallybigvenue | venueid    | lzo      | 97.54
reallybigvenue | venuename  | lzo      | 91.71
reallybigvenue | venuecity  | lzo      | 96.01
reallybigvenue | venuestate | lzo      | 97.68
reallybigvenue | venueseats | lzo      | 98.21
```

Beispiel: Wahl der Kompressionskodierungen für die Tabelle CUSTOMER

Die folgende Anweisung erstellt die Tabelle CUSTOMER, die Spalten mit verschiedenen Datentypen besitzt. Diese CREATE TABLE-Anweisung zeigt eine der zahlreichen möglichen Kombinationen von Kompressionskodierungen für diese Spalten.

```
create table customer(
custkey int encode delta,
custname varchar(30) encode raw,
gender varchar(7) encode text255,
address varchar(200) encode text255,
city varchar(30) encode text255,
state char(2) encode raw,
zipcode char(5) encode bytedict,
start_date date encode delta32k);
```

Die folgende Tabelle zeigt die Spaltenkodierungen, die für die Tabelle CUSTOMER gewählt wurden, und erklärt, warum die betreffende Kodierungen gewählt wurden:

Spalte	Datentyp	Codierung	Erklärung
CUSTKEY	int	DELTA	CUSTKEY besteht aus eindeutigen Ganzzahlwerten in Folge. Da die Unterschiede nur ein Byte betragen, stellt DELTA eine gute Wahl dar.
CUSTNAME	varchar(30)	RAW	CUSTNAME ist eine große Domäne, in der nur wenige Werte wiederholt werden. Jede Kompressi onskodierung wäre wahrscheinlich ineffektiv.

Spalte	Datentyp	Codierung	Erklärung
GENDER	varchar(7)	Text255	GENDER ist eine sehr kleine Domäne, in der zahlreiche Werte wiederholt werden. Text255 funktioniert gut mit VARCHAR-Spalten, in denen dieselben Wörter wiederholt werden.
ADDRESS	varchar(200)	Text255	ADDRESS ist eine große Domain, enthält jedoch zahlreiche Wörter, die sich wiederholen, wie Straße, Nord, Süd usw. Text255 und Text32k sind für die Komprimierung von VARCHAR-Spalten nützlich, in denen dieselben Wörter wiederholt werden. Die Spaltenlänge ist kurz. Daher ist Text255 eine gute Wahl.

Spalte	Datentyp	Codierung	Erklärung
CITY	varchar(30)	Text255	CITY ist eine große Domäne, in der einige Werte wiederholt werden. Bestimmte Namen von Städten werden häufiger als andere verwendet. Text255 ist aus den gleichen Gründen wie bei ADDRESS eine gute Wahl.
STATE	char(2)	RAW	In den Vereinigten Staaten ist STATE eine präzise Domäne mit 50 Werten, die aus zwei Zeichen bestehen. Eine Bytedict-Kodierung würde zu etwas Kompression führen. Da die Größe der Spalte jedoch nur zwei Zeichen beträgt, ist die Kompressi on wahrscheinlich den Aufwand nicht wert, der durch die Entkomprimierung der Daten entsteht.

Spalte	Datentyp	Codierung	Erklärung
ZIPCODE	char(5)	Bytedict	ZIPCODE ist eine bekannte Domäne mit weniger als 50.000 eindeutigen Werten. Bestimmte Postleitzahlen treten sehr viel häufiger auf als andere. Die Bytedict-Kodierung ist sehr effektiv, wenn eine Spalte eine begrenzte Zahl eindeutiger Werte enthält.
START_DATE	date	Delta32K	Delta-Kodierungen sind für Datum-/Uhrzeitspalten sehr nützlich, besonders, wenn die Zeilen in der Reihenfolge des Datums geladen werden.

Arbeiten mit Datenverteilungsstilen

Wenn Sie Daten in eine Tabelle laden, verteilt Amazon Redshift die Zeilen der Tabelle entsprechend dem Verteilungsstil der Tabelle an die einzelnen Datenverarbeitungsknoten. Wenn sie eine Abfrage ausführen, führt der Abfrageoptimierer nach Bedarf eine Neuverteilung der Zeilen zu den Datenverarbeitungsknoten durch, um Join- oder Aggregierungsoperationen durchführen zu können. Das Ziel der Auswahl eines Tabellenverteilungsstils besteht darin, die Auswirkungen des Neuverteilungsschritts dadurch zu minimieren, dass die Daten dort platziert sind, wo sie benötigt werden, bevor die Abfrage ausgeführt wird.

> **Note**
>
> In diesem Abschnitt werden die Grundsätze der Datenverteilung in einer Amazon-Redshift-Datenbank vorgestellt. Wir empfehlen, Ihre Tabellen mit `DISTSTYLE AUTO` zu erstellen. Wenn Sie dies tun, verwendet Amazon Redshift die automatische Tabellenoptimierung, um den Datenverteilungsstil auszuwählen. Weitere Informationen finden Sie unter Arbeiten mit automatischer Tabellenoptimierung. Der Rest dieses Abschnitts enthält Details zu Verteilungsstilen.

Themen

Datenverteilungskonzepte

Einige Datenverteilungskonzepte für Amazon Redshift folgen.

Knoten und Slices

Ein Amazon-Redshift-Cluster besteht aus einem Satz von Knoten. Jeder Knoten im Cluster besitzt ein eigenes Betriebssystem, einen eigenen dedizierten Arbeitsspeicher und einen eigenen dedizierten Datenträgerplatz. Ein Knoten ist der Führungsknoten, der die Verteilung von Daten- und Abfrageverarbeitungsaufgaben an die Datenverarbeitungsknoten verwaltet. Die Datenverarbeitungsknoten stellen Ressourcen bereit, um diese Aufgaben zu erledigen.

Der Datenträgerplatz für einen Datenverarbeitungsknoten ist in verschiedene Slices aufgeteilt. Die Anzahl der Slices pro Knoten ist von der Knotengröße des Clusters abhängig. Die Knoten sind alle an der parallelen Abfrageausführung beteiligt und arbeiten mit Daten, die so gleichmäßig wie möglich über die Slices verteilt sind. Weitere Informationen zur Anzahl der Slices für die einzelnen

Knotengrößen finden Sie unter About Clusters and Nodes (Informationen zu Clustern und Knoten) im Amazon-Redshift-Verwaltungshandbuch.

Datenumverteilung

Wenn Sie Daten in eine Tabelle laden, verteilt Amazon Redshift die Zeilen der Tabelle entsprechend dem Verteilungsstil der Tabelle an die einzelnen Knoten-Slices. Als Teil des Abfrageplans ermittelt der Optimierer die Stellen, an denen die Datenblöcke platziert werden müssen, um die Abfrage optimal ausführen zu können. Die Daten werden anschließend während der Ausführung der Abfrage physisch verschoben bzw. umverteilt. Während der Umverteilung können einzelne Zeilen an Knoten gesendet werden, um einen Join auszuführen, oder eine ganze Tabelle an alle Knoten gesendet werden.

Die Datenumverteilung kann einen wesentlichen Anteil an den Kosten eines Abfrageplans haben und der generierte Netzwerkdatenverkehr kann sich auf andere Datenbankoperationen auswirken und die Systemleistung insgesamt beeinträchtigen. Wenn Sie bereits zu Beginn überlegen, wo die Daten am besten platziert werden sollten, können Sie die Auswirkungen der Datenumverteilung minimieren.

Ziele der Datenverteilung

Wenn Sie Daten in eine Tabelle laden, verteilt Amazon Redshift die Zeilen der Tabelle entsprechend dem Verteilungsstil, den Sie während der Erstellung der Tabelle gewählt haben, an die Datenverarbeitungsknoten und Slices. Die Datenverteilung hat zwei primäre Ziele:

- Gleichmäßige Verteilung des Workloads auf die Knoten im Cluster. Eine ungleichmäßige Verteilung oder Verteilungsverzerrung zwingt einige Knoten, mehr Arbeit zu leisten als andere, was die Abfrageleistung beeinträchtigt.
- Minimierung der Datenverschiebungen während einer Abfrage. Wenn die Zeilen, die an Joins oder Aggregationen beteiligt sind, auf den Knoten bereits zusammen mit den entsprechenden Join-Zeilen in anderen Tabellen platziert werden, muss der Optimierer während der Abfrage weniger Daten umverteilen.

Die von Ihnen für Ihre Datenbank gewählte Verteilungsstrategie hat wichtige Folgen für die Abfrageleistung, die Speicheranforderungen, das Laden von Daten und die Wartung. Indem Sie für jede Tabelle den optimalen Verteilungsstil auswählen, können Sie Ihre Daten gleichmäßig verteilen und die allgemeine Systemleistung deutlich verbessern.

Verteilungsstile

Wenn Sie eine Tabelle erstellen, können Sie einen der folgenden Verteilungsstile festlegen: AUTO, EVEN, KEY oder ALL.

Falls Sie keinen Verteilungsstil angeben, verwendet Amazon Redshift die AUTO-Verteilung.

AUTO-Verteilung

Bei der AUTO-Verteilung weist Amazon Redshift auf der Grundlage der Größe der Tabellendaten den optimalen Verteilungsstil aus. Wenn beispielsweise der Verteilungsstil AUTO angegeben ist, weist Amazon Redshift kleinen Tabellen zunächst den Verteilungsstil ALL zu. Wenn die Tabelle größer wird, ändert Amazon Redshift den Verteilungsstil möglicherweise in KEY und wählt den Primärschlüssel (oder eine Spalte des zusammengesetzten Primärschlüssels) als Verteilungsschlüssel. Wenn die Tabelle größer wird und keine der Spalten als Verteilungsschlüssel geeignet ist, ändert Amazon Redshift den Verteilungsstil in EVEN. Die Änderung des Verteilungsstils erfolgt im Hintergrund mit minimalen Auswirkungen auf die Benutzerabfragen.

Informationen zum Anzeigen von Aktionen, die Amazon Redshift automatisch zum Ändern eines Tabellenverteilungsschlüssels ausgeführt hat, finden Sie unter SVL_AUTO_WORKER_ACTION. Informationen zum Anzeigen aktueller Empfehlungen zum Ändern eines Tabellenverteilungsschlüssels finden Sie unter SVV_ALTER_TABLE_RECOMMENDATIONS.

Um den Verteilungsstil einer Tabelle anzuzeigen, führen Sie eine Abfrage für die Systemkatalogansicht PG_CLASS_INFO aus. Weitere Informationen finden Sie unter Anzeigen von Verteilungsstilen. Falls Sie die CREATE TABLE-Anweisung ohne Verteilungsstil angeben, verwendet Amazon Redshift die AUTO-Verteilung.

EVEN-Verteilung

Der Führungsknoten verteilt die Zeilen nacheinander über die Slices, unabhängig von den Werten in einer bestimmten Spalte. EVEN-Verteilung ist angemessen, wenn eine Tabelle nicht an Joins beteiligt ist. Sie ist auch angemessen, wenn keine klare Entscheidung zwischen der KEY-Verteilung und der ALL-Verteilung getroffen werden kann.

KEY-Verteilung

Die Zeilen werden entsprechend den Werten in einer einzelnen Spalte verteilt. Der Führungsknoten platziert übereinstimmende Werte auf demselben Knoten-Slice. Wenn Sie ein Paar von Tabellen anhand der Joining-Schlüssel verteilen, platziert der Führungsknoten entsprechend den Werten in

den Joining-Spalten die Zeilen in den Slices. Auf diese Weise werden übereinstimmende Werte aus den gemeinsamen Spalten physisch zusammen gespeichert.

ALL-Verteilung

An jeden Knoten wird eine Kopie der gesamten Tabelle verteilt. Während die EVEN-Verteilung oder die KEY-Verteilung nur einen Teil der Zeilen einer Tabelle auf den einzelnen Knoten platzieren, stellt die ALL-Verteilung sicher, dass alle Zeilen für alle Joins, an denen die Tabelle beteiligt ist, gemeinsam gespeichert werden.

Die ALL-Verteilung multipliziert den erforderlichen Speicherplatz mit der Zahl der Knoten im Cluster. Daher dauert es sehr viel länger, Daten zu laden, zu aktualisieren oder in mehrere Tabellen einzufügen. Die ALL-Verteilung ist nur für Tabellen geeignet, die sich vergleichsweise wenig verändern, d. h. Tabellen, die nicht häufig oder umfassend aktualisiert werden. Da die Kosten für die Umverteilung kleiner Tabellen während einer Abfrage gering sind, gibt es keinen signifikanten Vorteil, kleine Dimensionstabellen als DISTSTYLE ALL zu definieren.

> **Note**
>
> Nachdem Sie für eine Spalte einen Verteilungsstil angegeben haben, verarbeitet Amazon Redshift die Datenverteilung auf Cluster-Ebene. Amazon Redshift erfordert keine Partitionierung von Daten innerhalb von Datenbankobjekten oder unterstützt dieses Konzept. Sie müssen keine Tabellenräume erstellen oder Partitionierungsschemen für Tabellen definieren.

In bestimmten Szenarien können Sie den Verteilungsstil einer Tabelle nach der Erstellung ändern. Weitere Informationen finden Sie unter ALTER TABLE. In Szenarien, in denen Sie den Verteilungsstil nach der Erstellung nicht ändern können, können Sie die Tabelle neu erstellen und die neue Tabelle mit einer Deep Copy-Operation füllen. Weitere Informationen finden Sie unter Ausführen einer Deep Copy-Operation

Anzeigen von Verteilungsstilen

Um den Verteilungsstil einer Tabelle anzuzeigen, führen Sie eine Abfrage für die Systemkatalogansichten PG_CLASS_INFO oder SVV_TABLE_INFO aus.

Die Spalte RELEFFECTIVEDISTSTYLE in PG_CLASS_INFO zeigt den Verteilungsstil für die Tabelle an. Wenn die Tabelle die automatische Verteilung verwendet, ist RELEFFECTIVEDISTSTYLE auf 10, 11 oder 12 festgelegt. Diese Werte bezeichnen die verwendeten Verteilungsstile AUTO (ALL), AUTO

(EVEN) oder AUTO (KEY). Wenn die Tabelle die automatische Verteilung verwendet, ändert sich der Verteilungsstile von anfänglich AUTO (ALL) in AUTO (EVEN) oder AUTO (KEY), wenn die Tabelle wächst.

In der folgenden Tabelle wird der Verteilungsstil für die einzelnen Werte in der Spalte RELEFFECTIVEDISTSTYLE angegeben:

RELEFFECTIVEDISTSTYLE	Aktueller Verteilungsstil
0	EVEN
1	KEY
8	ALL
10	AUTO (ALL)
11	AUTO (EVEN)
12	AUTO (KEY)

Die Spalte DISTSTYLE in SVV_TABLE_INFO zeigt den aktuellen Verteilungsstil für die Tabelle an. Wenn die Tabelle die automatische Verteilung verwendet, ist für DISTSTYLE entweder AUTO (ALL), AUTO (EVEN) oder AUTO (KEY) festgelegt.

Im folgenden Beispiel werden vier Tabellen erstellt, die die drei Verteilungsstile und die automatische Verteilung verwenden, anschließend wird eine Abfrage für PG_TABLE_INFO ausgeführt, um die Verteilungsstile anzuzeigen.

```
create table public.dist_key (col1 int)
diststyle key distkey (col1);

insert into public.dist_key values (1);

create table public.dist_even (col1 int)
diststyle even;

insert into public.dist_even values (1);

create table public.dist_all (col1 int)
```

```
diststyle all;

insert into public.dist_all values (1);

create table public.dist_auto (col1 int);

insert into public.dist_auto values (1);

select "schema", "table", diststyle from SVV_TABLE_INFO
where "table" like 'dist%';

        schema   |    table        | diststyle
     ------------+-----------------+------------
      public     | dist_key        | KEY(col1)
      public     | dist_even       | EVEN
      public     | dist_all        | ALL
      public     | dist_auto       | AUTO(ALL)
```

Auswerten von Abfragemustern

Die Wahl des Verteilungsstils stellt nur einen Aspekt des Datenbankdesigns dar. Betrachten Sie den Verteilungsstil im Kontext des gesamten Systems und wägen Sie die Verteilung mit anderen wichtigen Faktoren wie Clustergröße, Methoden der Kompressionskodierung, Sortierschlüsseln und Tabelleneinschränkungen ab.

Testen Sie Ihr System mit Daten, die so nahe an den realen Daten wie möglich sind.

Um eine gute Wahl in Bezug auf den Verteilungsstil treffen zu können, müssen Sie die Abfragemuster für Ihre Amazon-Redshift-Anwendung verstehen. Identifizieren Sie aufwändigsten Abfragen in Ihrem System und bauen Sie bei Ihrem anfänglichen Datenbankdesign auf den Anforderungen dieser Abfragen auf. Zu den Faktoren, die die Gesamtkosten einer Abfrage bestimmen, gehören die Dauer der Ausführung der Abfrage und die Zahl der beanspruchten Datenverarbeitungsressourcen. Andere Faktoren, die die Abfragekosten bestimmen, sind die Häufigkeit der Ausführung und die Unterbrechung anderer Abfragen und Datenbankvorgänge.

Identifizieren Sie die Tabellen, die von den kostenintensivsten Abfragen verwendet werden, und werten sie deren Rolle bei der Abfragelaufzeit aus. Betrachten Sie die Joins und Aggregationen für diese Tabellen.

Wählen Sie anhand der Anleitungen in diesem Abschnitt für jede Tabelle einen Verteilungsstil aus. Erstellen Sie anschließend die Tabellen, landen Sie sie mit Daten, die den realen Daten so

ähnlich wie möglich sind. Testen Sie dann die Tabellen auf die Abfragetypen, die Sie verwenden möchten. Sie können die Abfragebeschreibungspläne auswerten, um Optimierungsmöglichkeiten zu identifizieren. Vergleichen Sie Ladezeiten, Speicheranforderungen und Abfragelaufzeiten, um die Anforderungen Ihres Systems insgesamt auszugleichen.

Bezeichnen von Verteilungsstilen

Die Überlegungen und Empfehlungen für die Bezeichnung von Verteilungsstilen in diesem Abschnitt verwenden ein Sternschema als Beispiel. Ihr Datenbankdesign könnte auf einem Sternschema, einer Sternschemavariante oder einem ganz anderen Schema beruhen. Amazon Redshift wurde entwickelt, um effektiv mit jedem Schemadesign zu arbeiten, das Sie wählen. Die Grundsätze in diesem Abschnitt können auf jedes Schemadesign angewendet werden.

1. Geben Sie die Primär- und Fremdschlüssel für alle Ihre Tabellen an.

   Amazon Redshift setzt keine Einschränkungen für Primär- und Fremdschlüssel durch. Der Abfrageoptimierer verwendet sie jedoch beim Generieren von Abfrageplänen. Wenn Sie Primär- und Fremdschlüssel festlegen, muss Ihre Anwendung die Gültigkeit der Schlüssel wahren.
2. Verteilen Sie die Faktentabelle und ihre größte Dimensionstabelle anhand ihrer gemeinsamen Spalten.

   Wählen Sie die größte Dimension basierend auf der Größe des Datensatzes aus, der am häufigsten Join beteiligt ist, nicht nur basierend auf der Größe der Tabelle. Wenn eine Tabelle unter Verwendung einer WHERE-Klausel gefiltert ist, ist nur ein Teil ihrer Zeilen am Join beteiligt. Eine solche Tabelle wirkt sich weniger auf die Umverteilung aus als eine kleinere Tabelle, die mehr Daten beiträgt. Bezeichnen Sie den Primärschlüssel der Dimensionstabelle und den entsprechenden Fremdschlüssel der Faktentabelle als DISTKEY. Wenn mehrere Tabellen denselben Verteilungsschlüssel verwenden, werden sie ebenfalls zusammen mit der Faktentabelle platziert. Ihre Faktentabelle kann nur einen Verteilungsschlüssel haben. Alle Tabellen, die auf einem anderen Schlüssel verbunden werden, werden nicht mit der Faktentabelle zusammengestellt.
3. Bezeichnen Sie die Verteilungsschlüssel für die anderen Dimensionstabellen.

   Verteilt die Tabellen anhand ihrer Primär- oder Fremdschlüssel, abhängig davon wie sie am häufigsten Joins mit anderen Tabellen ausführen.
4. Überprüfen Sie, ob die Verteilung einiger Dimensionstabellen in eine ALL-Verteilung geändert werden sollte.

Wenn eine Dimensionstabelle nicht mit der Faktentabelle oder anderen wichtigen Joining-Tabellen zusammengestellt werden kann, können Sie die Abfrageleistung dadurch erheblich verbessern, dass Sie die gesamte Tabelle zu allen Knoten verteilen. Die Verwendung der ALL-Verteilung vervielfacht die Speicheranforderungen, verlängert Ladezeiten und erhöht den Aufwand für Wartungsoperationen. Sie sollten daher alle Faktoren sorgfältig abwägen, bevor Sie die ALL-Verteilung wählen. Im folgenden Abschnitt wird beschrieben, wie Sie Kandidaten für die ALL-Verteilung durch Auswerten des EXPLAIN-Plans identifizieren.

5. Verwenden Sie die AUTO-Verteilung für die übrigen Tabellen.

   Wenn eine Tabelle zum großen Teil denormalisiert ist und nicht an Joins beteiligt ist oder es keine klare Wahl für einen anderen Verteilungsstil gibt, verwenden Sie die AUTO-Verteilung.

Wenn Sie möchten, dass Amazon Redshift einen geeigneten Verteilungsstil auswählt, geben Sie keinen Verteilungsstil an.

Auswerten des Abfrageplans

Sie können Abfragepläne verwenden, um Kandidaten für die Optimierung des Verteilungsstils zu identifizieren.

Nachdem Sie Ihre anfänglichen Designentscheidungen getroffen haben, erstellen Sie Ihre Tabellen, laden Daten in sie und testen sie. Verwenden Sie einen Testdatensatz, der so nahe an den realen Daten wie möglich ist. Messen Sie die Ladezeiten, um sie als Ausgangspunkt für Vergleiche verwenden zu können.

Werten Sie Abfragen aus, die repräsentativ für die aufwändigsten Abfragen sind, die Sie voraussichtlich ausführen werden, insbesondere Abfragen, die Joins und Aggregationen verwenden. Vergleichen Sie die Laufzeiten für verschiedene Designoptionen. Wenn Sie die Laufzeiten vergleichen, sollten Sie die erste Ausführung der Abfrage nicht berücksichtigen, da die erste Laufzeit die Kompilierungszeit enthält.

DS_DIST_NONE

Es ist keine Umverteilung erforderlich, da korrespondierende Slices auf den Datenverarbeitungsknoten zusammen platziert werden. In der Regel gibt es nur einen DS_DIST_NONE-Schritt, den Join zwischen der Faktentabelle und einer einzelnen Dimensionstabelle.

DS_DIST_ALL_NONE

Es ist keine Umverteilung erforderlich, da die innere Join-Tabelle bereits DISTSTYLE ALL verwendet hat. Die gesamte Tabelle befindet sich auf jedem Knoten.

DS_DIST_INNER

Die innere Tabelle wird umverteilt.

DS_DIST_OUTER

Die äußere Tabelle wird umverteilt.

DS_BCAST_INNER

Eine Kopie der gesamten inneren Tabelle wird an alle Verarbeitungsknoten gesendet.

DS_DIST_ALL_INNER

Die gesamte innere Tabelle wird an eine einzige Slice umverteilt wurde, weil die äußere Tabelle DISTSTYLE ALL verwendet.

DS_DIST_BOTH

Beide Tabellen werden umverteilt.

DS_DIST_NONE und DS_DIST_ALL_NONE sind gut. Sie zeigen an, dass für diesen Schritt keine Verteilung erforderlich war, da alle Joins zusammen platziert waren.

DS_DIST_INNER bedeutet, dass der Schritt wahrscheinlich relativ hohe Kosten verursacht, da die innere Tabelle an die Knoten umverteilt wird. DS_DIST_INNER zeigt an, dass die äußere Tabelle bereits korrekt anhand des Join-Schlüssels verteilt wurde. Legen Sie den Verteilungsschlüssel der inneren Tabelle auf den Join-Schlüssel fest, um dies in DS_DIST_NONE zu konvertieren. In einigen Fällen ist eine Verteilung der inneren Tabelle auf den Join-Schlüssel nicht möglich, da die äußere Tabelle nicht auf den Join-Schlüssel verteilt ist. Wenn dies der Fall ist, prüfen Sie, ob die ALL-Verteilung für die innere Tabelle verwendet werden soll. Wenn die Tabelle nicht häufig oder umfassend aktualisiert wird, und groß genug ist, um zu hohen Umverteilungskosten zu führen, ändern Sie den Verteilungsschlüssel in ALL und führen den Test erneut aus. Die ALL-Verteilung verursacht höhere Ladezeiten. Berücksichtigen Sie daher die Ladezeit bei Ihrer Auswertung, wenn Sie den Test erneut ausführen.

DS_DIST_ALL_INNER ist nicht gut. Dies bedeutet, dass die gesamte innere Tabelle an einen einzelnen Slice umverteilt wird, da die äußere Tabelle DISTSTYLE ALL verwendet, sodass sich auf

jedem Knoten eine Kopie der gesamten äußeren Tabelle befindet. Dies führt zu einer ineffizienten seriellen Laufzeit des Joins auf einem einzelnen Knoten, anstatt die Vorteile einer parallelen Laufzeit unter Verwendung aller Knoten zu nutzen. DISTSTYLE ALL ist nur für die Verwendung für die innere Join-Tabelle vorgesehen. Geben Sie stattdessen einen Verteilungsschlüssel an oder verwenden Sie für die äußere Tabelle die EVEN-Verteilung.

DS_BCAST_INNER und DS_DIST_BOTH sind nicht gut. In der Regel erfolgen diese Umverteilungen, da für die Tabellen kein Join anhand ihrer Verteilungsschlüssel ausgeführt wurde. Wenn die Faktentabelle noch nicht über einen Verteilungsschlüssel verfügt, geben Sie für beide Tabellen die Join-Spalte als Verteilungsschlüssel an. Wenn die Faktentabelle bereits über einen Verteilungsschlüssel anhand einer anderen Spalte verfügt, überprüfen Sie, ob eine Änderung des Verteilungsschlüssels, damit dieser Join zusammen platziert werden kann, die Leistung insgesamt verbessert. Wenn die Änderung des Verteilungsschlüssels der äußeren Tabelle keine optimale Wahl darstellt, können Sie eine gemeinsame Platzierung erzielen, indem Sie für die innere Tabelle DISTSTYLE ALL angeben.

Im folgenden Beispiel wird ein Teil eines Abfrageplans mit den Bezeichnungen DS_BCAST_INNER und DS_DIST_NONE gezeigt.

```
->  XN Hash Join DS_BCAST_INNER  (cost=112.50..3272334142.59 rows=170771 width=84)
        Hash Cond: ("outer".venueid = "inner".venueid)
        ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..3167290276.71 rows=172456
 width=47)
              Hash Cond: ("outer".eventid = "inner".eventid)
              ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6286.47 rows=172456 width=30)
                    Merge Cond: ("outer".listid = "inner".listid)
                    ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497
 width=14)
                    ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=24)
```

Nach der Änderung des Verteilungsstils der Dimensionstabellen in DISTSTYLE ALL zeigt der Abfrageplan für dieselbe Abfrage DS_DIST_ALL_NONE anstelle von DS_BCAST_INNER. Es gibt darüber hinaus eine wesentliche Änderung bei den relativen Kosten für die Join-Schritte. Die Gesamtkosten sind `14142.59` verglichen mit `3272334142.59` in der vorherigen Abfrage.

```
->  XN Hash Join DS_DIST_ALL_NONE  (cost=112.50..14142.59 rows=170771 width=84)
        Hash Cond: ("outer".venueid = "inner".venueid)
        ->  XN Hash Join DS_DIST_ALL_NONE  (cost=109.98..10276.71 rows=172456 width=47)
              Hash Cond: ("outer".eventid = "inner".eventid)
              ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6286.47 rows=172456 width=30)
```

```
                    Merge Cond: ("outer".listid = "inner".listid)
                    ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497
 width=14)
                    ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=24)
```

Beispiel für einen Abfrageplan

In diesem Beispiel wird gezeigt, wie Sie einen Abfrageplan auswerten, um Möglichkeiten für die Optimierung der Distribution zu identifizieren.

Führen Sie die folgende Abfrage mit einem EXPLAIN-Befehl aus, um einen Abfrageplan zu generieren.

```
explain
select lastname, catname, venuename, venuecity, venuestate, eventname,
month, sum(pricepaid) as buyercost, max(totalprice) as maxtotalprice
from category join event on category.catid = event.catid
join venue on venue.venueid = event.venueid
join sales on sales.eventid = event.eventid
join listing on sales.listid = listing.listid
join date on sales.dateid = date.dateid
join users on users.userid = sales.buyerid
group by lastname, catname, venuename, venuecity, venuestate, eventname, month
having sum(pricepaid)>9999
order by catname, buyercost desc;
```

SALES ist in der Datenbank TICKIT eine Faktentabelle und LISTING ist ihre größte Dimension. Um die Tabellen gemeinsam zu platzieren, wird SALES anhand der LISTID verteilt. Dies ist der Fremdschlüssel für LISTING. LISTING wird anhand ihres Primärschlüssels LISTID verteilt. Im folgenden Beispiel werden die CREATE TABLE-Befehle für SALES und LISTING gezeigt.

```
create table sales(
 salesid integer not null,
 listid integer not null distkey,
 sellerid integer not null,
 buyerid integer not null,
 eventid integer not null encode mostly16,
 dateid smallint not null,
 qtysold smallint not null encode mostly8,
 pricepaid decimal(8,2) encode delta32k,
```

```
 commission decimal(8,2) encode delta32k,
 saletime timestamp,
 primary key(salesid),
 foreign key(listid) references listing(listid),
 foreign key(sellerid) references users(userid),
 foreign key(buyerid) references users(userid),
 foreign key(dateid) references date(dateid))
        sortkey(listid,sellerid);

create table listing(
 listid integer not null distkey sortkey,
 sellerid integer not null,
 eventid integer not null encode mostly16,
 dateid smallint not null,
 numtickets smallint not null encode mostly8,
 priceperticket decimal(8,2) encode bytedict,
 totalprice decimal(8,2) encode mostly32,
 listtime timestamp,
 primary key(listid),
 foreign key(sellerid) references users(userid),
 foreign key(eventid) references event(eventid),
 foreign key(dateid) references date(dateid));
```

Im folgenden Abfrageplan zeigt der Zusammenführungs-Join-Schritt für den Join für SALES und LISTING DS_DIST_NONE. Dies zeigt, dass für den Schritt keine Umverteilung erforderlich ist. Weiter oben im Abfrageplan sehen Sie jedoch, dass die übrigen inneren Joins DS_BCAST_INNER zeigen. Dies zeigt an, dass die innere Tabelle als Teil der Abfrageausführung rundgesendet wird. Da nur ein Paar von Tabellen gemeinsam anhand der Schlüsselverteilung platziert werden kann, müssen fünf Tabellen erneut rundgesendet werden.

```
QUERY PLAN
XN Merge  (cost=1015345167117.54..1015345167544.46 rows=1000 width=103)
  Merge Key: category.catname, sum(sales.pricepaid)
  ->  XN Network  (cost=1015345167117.54..1015345167544.46 rows=170771 width=103)
        Send to leader
        ->  XN Sort  (cost=1015345167117.54..1015345167544.46 rows=170771 width=103)
              Sort Key: category.catname, sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=15345150568.37..15345152276.08 rows=170771
 width=103)
                    Filter: (sum(pricepaid) > 9999.00)
                     ->  XN Hash Join DS_BCAST_INNER  (cost=742.08..15345146299.10
 rows=170771 width=103)
```

```
                           Hash Cond: ("outer".catid = "inner".catid)
                           ->  XN Hash Join DS_BCAST_INNER
(cost=741.94..15342942456.61 rows=170771 width=97)
                                 Hash Cond: ("outer".dateid = "inner".dateid)
                                 ->  XN Hash Join DS_BCAST_INNER
(cost=737.38..15269938609.81 rows=170766 width=90)
                                       Hash Cond: ("outer".buyerid = "inner".userid)
                                       ->  XN Hash Join DS_BCAST_INNER
(cost=112.50..3272334142.59 rows=170771 width=84)
                                             Hash Cond: ("outer".venueid =
"inner".venueid)
                                             ->  XN Hash Join DS_BCAST_INNER
(cost=109.98..3167290276.71 rows=172456 width=47)
                                                   Hash Cond: ("outer".eventid =
"inner".eventid)
                                                   ->  XN Merge Join DS_DIST_NONE
(cost=0.00..6286.47 rows=172456 width=30)
                                                         Merge Cond: ("outer".listid =
"inner".listid)
                                                         ->  XN Seq Scan on listing
(cost=0.00..1924.97 rows=192497 width=14)
                                                         ->  XN Seq Scan on sales
(cost=0.00..1724.56 rows=172456 width=24)
                                                   ->  XN Hash  (cost=87.98..87.98
rows=8798 width=25)
                                                         ->  XN Seq Scan on event
(cost=0.00..87.98 rows=8798 width=25)
                                             ->  XN Hash  (cost=2.02..2.02 rows=202
width=41)
                                                   ->  XN Seq Scan on venue
(cost=0.00..2.02 rows=202 width=41)
                                       ->  XN Hash  (cost=499.90..499.90 rows=49990
width=14)
                                             ->  XN Seq Scan on users
(cost=0.00..499.90 rows=49990 width=14)
                                 ->  XN Hash  (cost=3.65..3.65 rows=365 width=11)
                                       ->  XN Seq Scan on date  (cost=0.00..3.65
rows=365 width=11)
                           ->  XN Hash  (cost=0.11..0.11 rows=11 width=10)
                                 ->  XN Seq Scan on category  (cost=0.00..0.11 rows=11
width=10)
```

Eine Lösung besteht darin, die Tabellen zu ändern, sodass sie DISTSTYLE ALL aufweisen.

```
ALTER TABLE users ALTER DISTSTYLE ALL;
ALTER TABLE venue ALTER DISTSTYLE ALL;
ALTER TABLE category ALTER DISTSTYLE ALL;
ALTER TABLE date ALTER DISTSTYLE ALL;
ALTER TABLE event ALTER DISTSTYLE ALL;
```

Führen Sie denselben Abfrageplan mit EXPLAIN erneut aus und untersuchen Sie den neuen Abfrageplan. Die Joins zeigen nun DS_DIST_ALL_NONE. Dies zeigt an, dass keine Umverteilung erforderlich ist, da die Daten mittels DISTSTYLE ALL an alle Knoten verteilt wurden.

```
QUERY PLAN
XN Merge  (cost=1000000047117.54..1000000047544.46 rows=1000 width=103)
  Merge Key: category.catname, sum(sales.pricepaid)
  ->  XN Network  (cost=1000000047117.54..1000000047544.46 rows=170771 width=103)
        Send to leader
        ->  XN Sort  (cost=1000000047117.54..1000000047544.46 rows=170771 width=103)
              Sort Key: category.catname, sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=30568.37..32276.08 rows=170771 width=103)
                    Filter: (sum(pricepaid) > 9999.00)
                    ->  XN Hash Join DS_DIST_ALL_NONE  (cost=742.08..26299.10
 rows=170771 width=103)
                          Hash Cond: ("outer".buyerid = "inner".userid)
                          ->  XN Hash Join DS_DIST_ALL_NONE  (cost=117.20..21831.99
 rows=170766 width=97)
                                Hash Cond: ("outer".dateid = "inner".dateid)
                                ->  XN Hash Join DS_DIST_ALL_NONE
 (cost=112.64..17985.08 rows=170771 width=90)
                                      Hash Cond: ("outer".catid = "inner".catid)
                                      ->  XN Hash Join DS_DIST_ALL_NONE
 (cost=112.50..14142.59 rows=170771 width=84)
                                            Hash Cond: ("outer".venueid =
 "inner".venueid)
                                            ->  XN Hash Join DS_DIST_ALL_NONE
 (cost=109.98..10276.71 rows=172456 width=47)
                                                  Hash Cond: ("outer".eventid =
 "inner".eventid)
                                                  ->  XN Merge Join DS_DIST_NONE
 (cost=0.00..6286.47 rows=172456 width=30)
                                                        Merge Cond: ("outer".listid =
 "inner".listid)
                                                        ->  XN Seq Scan on listing
 (cost=0.00..1924.97 rows=192497 width=14)
```

```
                                                        ->  XN Seq Scan on sales
 (cost=0.00..1724.56 rows=172456 width=24)
                                                  ->  XN Hash  (cost=87.98..87.98
 rows=8798 width=25)
                                                        ->  XN Seq Scan on event
 (cost=0.00..87.98 rows=8798 width=25)
                                              ->  XN Hash  (cost=2.02..2.02 rows=202
 width=41)
                                                    ->  XN Seq Scan on venue
 (cost=0.00..2.02 rows=202 width=41)
                                        ->  XN Hash  (cost=0.11..0.11 rows=11 width=10)
                                              ->  XN Seq Scan on category
 (cost=0.00..0.11 rows=11 width=10)
                                  ->  XN Hash  (cost=3.65..3.65 rows=365 width=11)
                                        ->  XN Seq Scan on date  (cost=0.00..3.65
 rows=365 width=11)
                            ->  XN Hash  (cost=499.90..499.90 rows=49990 width=14)
                                  ->  XN Seq Scan on users  (cost=0.00..499.90 rows=49990
 width=14)
```

Verteilungsbeispiele

In den folgenden Beispielen wird gezeigt, wie Daten anhand der Optionen verteilt werden, die Sie in der CREATE TABLE-Anweisung definieren.

Beispiele für DISTKEY

Betrachten Sie das Schema der Tabelle USERS in der Datenbank TICKIT. USERID ist als SORTKEY-Spalte und DISTKEY-Spalte definiert:

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'users';

    column     |          type          | encoding | distkey | sortkey
---------------+------------------------+----------+---------+---------
 userid        | integer                | none     | t       |       1
 username      | character(8)           | none     | f       |       0
 firstname     | character varying(30)  | text32k  | f       |       0

...
```

USERID ist eine gute Wahl als Verteilungsspalte für diese Tabelle. Wenn Sie eine Abfrage für die Systemansicht SVV_DISKUSAGE ausführen, können Sie sehen, dass die Tabelle sehr gleichmäßig verteilt ist. Die Spaltennummern sind nullbasiert. Daher ist USERID Spalte 0.

```
select slice, col, num_values as rows, minvalue, maxvalue
from svv_diskusage
where name='users' and col=0 and rows>0
order by slice, col;

slice| col | rows  | minvalue | maxvalue
-----+-----+-------+----------+----------
0    | 0   | 12496 | 4        | 49987
1    | 0   | 12498 | 1        | 49988
2    | 0   | 12497 | 2        | 49989
3    | 0   | 12499 | 3        | 49990
(4 rows)
```

Die Tabelle enthält 49.990 Zeilen. Die Spalte für Zeilen (num_values) zeigt, dass jeder Slice ungefähr dieselbe Zahl von Zeilen enthält. Die Spalten für den Mindest- und Maximalwert (minvalue und maxvalue) zeigen den Bereich der Werte in jedem Slice. Jeder Slice enthält beinahe den gesamten Bereich von Werten. Daher ist die Chance groß, dass jeder Slice an der Ausführung einer Abfrage beteiligt ist, die nach einem Bereich von Benutzer-IDs filtert.

Dieses Beispiel zeigt eine Verteilung für ein kleines Testsystem. Die Gesamtzahl der Slices ist in der Regel sehr viel höher.

Wenn Sie häufig Joins oder Gruppierungen anhand der Spalte STATE ausführen, sollten Sie vielleicht eine Verteilung anhand der Spalte STATE wählen. Das folgende Beispiel zeigt das Erstellen einer neuen Tabelle mit denselben Daten wie die Tabelle USERS, bei der der DISTKEY jedoch auf die Spalte STATE festgelegt wird. In diesem Fall ist die Verteilung nicht so gleichmäßig. Slice 0 (13.587 Zeilen) enthält ungefähr 30 % mehr Zeilen als Slice 3 (10.150 Zeilen). In einer sehr viel größeren Tabelle könnte sich diese Verteilungsverzerrung nachteilig auf die Abfrageverarbeitung auswirken.

```
create table userskey distkey(state) as select * from users;

select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'userskey' and col=0 and rows>0
order by slice, col;

slice | col | rows  | minvalue | maxvalue
```

```
------+-----+-------+----------+----------
    0 |   0 | 13587 |        5 |    49989
    1 |   0 | 11245 |        2 |    49990
    2 |   0 | 15008 |        1 |    49976
    3 |   0 | 10150 |        4 |    49986
(4 rows)
```

Beispiel für DISTSTYLE EVEN

Wenn Sie eine neue Tabelle mit denselben Daten wie die Tabelle USERS erstellen, DISTSTYLE jedoch auf EVEN festlegen, werden die Zeilen stets gleichmäßig über die Slices verteilt.

```
create table userseven diststyle even as
select * from users;

select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'userseven' and col=0 and rows>0
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 12497 |        4 |    49990
    1 |   0 | 12498 |        8 |    49984
    2 |   0 | 12498 |        2 |    49988
    3 |   0 | 12497 |        1 |    49989
(4 rows)
```

Da die Verteilung jedoch nicht auf einer spezifischen Spalte basiert, kann die Leistung der Abfrageverarbeitung nachlassen, besonders, wenn für die Tabelle ein Join mit anderen Tabellen ausgeführt wurde. Eine fehlende Verteilung anhand einer Join-Spalte wirkt sich häufig auf den Typ der Join-Operation aus, der effizient ausgeführt werden kann. Join-, Aggregations- und Gruppierungsoperationen sind optimiert, wenn beide Tabellen anhand der jeweiligen Join-Spalten verteilt und sortiert sind.

Beispiel für DISTSTYLE ALL

Wenn Sie eine neue Tabelle mit denselben Daten wie die Tabelle USERS erstellen, DISTSTYLE jedoch auf ALL festlegen, werden alle Zeilen an den ersten Slice jedes Knotens verteilt.

```
select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'usersall' and col=0 and rows > 0
```

```
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 49990 |        4 |    49990
    2 |   0 | 49990 |        2 |    49990

(4 rows)
```

Arbeiten mit Sortierschlüsseln

> **Note**
>
> Wir empfehlen, Ihre Tabellen mit `SORTKEY AUTO` zu erstellen. Wenn Sie dies tun, verwendet Amazon Redshift die automatische Tabellenoptimierung, um den Sortierschlüssel auszuwählen. Weitere Informationen finden Sie unter Arbeiten mit automatischer Tabellenoptimierung. Der Rest dieses Abschnitts enthält Details zur Sortierreihenfolge.

Wenn Sie eine Tabelle erstellen, können Sie alternativ eine oder mehrere ihrer Spalten als Sortierschlüssel definieren. Wenn Daten zum ersten Mal in die leere Tabelle geladen werden, werden die Zeilen auf dem Datenträger in sortierter Reihenfolge gespeichert. Die Informationen zu Sortierschlüsselspalten werden an den Abfrageplaner übergeben. Der Planer verwendet diese Informationen zum Erstellen von Plänen, die die Sortierung der Daten nutzen. Weitere Informationen finden Sie unter CREATE TABLE. Informationen zu bewährten Methoden beim Erstellen eines Sortierschlüssels finden Sie unter Auswahl des besten Sortierschlüssels.

Die Sortierung ermöglicht eine effiziente Verarbeitung von Prädikaten mit eingeschränkten Bereichen. Amazon Redshift speichert Spaltendaten in 1-MB-Blöcken. Die Mindest- und Maximalwerte für jeden Block werden als Teil der Metadaten gespeichert. Wenn eine Abfrage ein Prädikat mit eingeschränktem Bereich verwendet, kann der Abfrageverarbeiter die Mindest- und Maximalwerte verwenden, um während Tabellenscans große Mengen von Blöcken schnell zu überspringen. Angenommen, eine Tabelle speichert Daten aus fünf Jahren, die nach Datum sortiert sind, und eine Abfrage gibt einen Datumsbereich von einem Monat an. In diesem Fall können Sie bis zu 98 Prozent der Festplattenblöcke aus dem Scan entfernen. Wenn die Daten nicht sortiert sind, müssen mehr Datenträgerblöcke (möglicherweise alle) gescannt werden.

Sie können einen zusammengesetzten oder einen überlappenden Sortierschlüssel angeben. Ein zusammengesetzter Sortierschlüssel ist effizienter, wenn Abfrageprädikate ein Präfix verwenden,

das ein Teilbereich der Sortierschlüsselspalten ihrer Reihenfolge nach ist. Ein überlappender Sortierschlüssel gewichtet jede Spalte im Sortierschlüssel gleich, sodass Abfrageprädikate jeden Teilbereich der Spalten, aus denen der Sortierschlüssel besteht, in jeder beliebigen Reihenfolge verwenden können.

Um die Auswirkungen des gewählten Sortierschlüssels auf die Abfrageleistung zu verstehen, verwenden Sie den Befehl EXPLAIN. Weitere Informationen finden Sie unter Workflow der Abfrageplanung und -ausführung.

Um einen Sortiertyp zu definieren, verwenden Sie die Schlüsselwörter INTERLEAVED oder COMPOUND mit Ihrer CREATE TABLE- oder CREATE TABLE AS-Anweisung. Der Standard ist COMPOUND. COMPOUND wird empfohlen, wenn Ihre Tabellen regelmäßig mit INSERT-, UPDATE- oder DELETE-Operationen aktualisiert werden. Ein INTERLEAVED-Sortierschlüssel kann maximal acht Spalten verwenden. Je nach Daten und Cluster-Größe benötigt VACUUM REINDEX deutlich mehr Zeit als VACUUM FULL, da ein zusätzlicher Schritt zur Analyse der überlappenden Sortierschlüssel ausgeführt wird. Die Sortier- und Zusammenführungsoperation für überlappende Tabellen kann länger dauern, weil die überlappende Sortierung möglicherweise mehr Zeilen als eine zusammengesetzte Sortierung neu anordnen muss.

Um die Sortierschlüssel für eine Tabelle anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_TABLE_INFO aus.

Themen

- Mehrdimensionale Datenlayout-Sortierung (Vorschau)
- Zusammengesetzter Sortierschlüssel
- Überlappender Sortierungsschlüssel

Mehrdimensionale Datenlayout-Sortierung (Vorschau)

Im Folgenden finden Sie eine Dokumentation zur Vorschauversion für die mehrdimensionale Datenlayout-Sortierung von Tabellen. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

> **Note**
>
> Dieses Feature ist nur bei Verwendung von Vorschau-Clustern und Vorschau-Arbeitsgruppen verfügbar. Weitere Informationen zum Einrichten von Vorschau-Clustern finden Sie unter Erstellen eines Vorschau-Clusters im Amazon-Redshift-Verwaltungshandbuch. Weitere Informationen zum Einrichten von Vorschau-Arbeitsgruppen finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe im Amazon-Redshift-Verwaltungshandbuch.

Ein mehrdimensionaler Datenlayout-Sortierschlüssel ist eine Art AUTO-Sortierschlüssel, der auf sich wiederholenden Prädikaten basiert, die in einem Workload vorkommen. Wenn Ihr Workload sich wiederholende Prädikate enthält, kann Amazon Redshift die Tabellen-Scan-Leistung verbessern, indem Datenzeilen, die den sich wiederholenden Prädikaten entsprechen, gekoppelt werden. Anstatt die Daten einer Tabelle in strikter Spaltenreihenfolge zu speichern, speichert ein mehrdimensionaler Datenlayout-Sortierschlüssel Daten, indem er sich wiederholende Prädikate analysiert, die in einem Workload vorkommen. In einem Workload kann es mehr als ein sich wiederholendes Prädikat geben. Je nach Workload kann diese Art von Sortierschlüssel die Leistung vieler Prädikate verbessern. Amazon Redshift bestimmt automatisch, ob diese Sortierschlüsselmethode für Tabellen verwendet werden soll, die mit einem AUTO-Sortierschlüssel definiert sind.

Angenommen, Sie haben eine Tabelle, deren Daten in Spaltenreihenfolge sortiert sind. Möglicherweise müssen viele Datenblöcke untersucht werden, um festzustellen, ob sie den Prädikaten Ihres Workloads entsprechen. Wenn die Daten jedoch in Prädikatreihenfolge auf der Festplatte gespeichert sind, müssen weniger Blöcke gescannt werden, um die Abfrage durchzuführen. In diesem Fall ist die Verwendung eines mehrdimensionalen Datenlayout-Sortierschlüssels von Vorteil.

Informationen dazu, ob eine Abfrage einen mehrdimensionalen Datenlayoutschlüssel verwendet, finden Sie in der `step_attribute`-Spalte der SYS_QUERY_DETAIL-Ansicht. Wenn der Wert `multi-dimensional` ist, wurde ein mehrdimensionales Datenlayout für die Abfrage verwendet. Informationen dazu, ob eine mit dem AUTO-Sortierschlüssel definierte Tabelle ein mehrdimensionales Datenlayout verwendet, finden Sie in der `sortkey1`-Spalte der SVV_TABLE_INFO-Ansicht. Wenn der Wert `padb_internal_mddl_key_col` ist, wurde ein mehrdimensionales Datenlayout für den Tabellensortierschlüssel verwendet.

Um zu verhindern, dass Amazon Redshift einen mehrdimensionalen Sortierschlüssel für das Datenlayout verwendet, wählen Sie eine andere Tabellen-Sortierschlüsseloption als `SORTKEY AUTO`. Weitere Informationen zu SORTKEY-Optionen finden Sie unter CREATE TABLE.

Zusammengesetzter Sortierschlüssel

Ein zusammengesetzter Sortierschlüssel besteht aus allen in der Sortierschlüsseldefinition aufgelisteten Spalten in der Reihenfolge ihrer Auflistung. Ein zusammengesetzter Sortierschlüssel ist dann am nützlichsten, wenn der Filter einer Abfrage Bedingungen wie Filter und Joins anwendet, die ein Präfix der Sortierschlüssel verwenden. Die Leistungsvorteile einer Sortierung mit einem zusammengesetzten Schlüssel nehmen ab, wenn Abfragen ausschließlich von sekundären Sortierspalten abhängig sind, ohne dass die primären Spalten referenziert werden. Der Standardsortiertyp ist COMPOUND.

Zusammengesetzte Sortierschlüssel können Joins, GROUP BY- und ORDER BY-Operationen sowie Fensterfunktionen, die PARTITION BY und ORDER BY verwenden, beschleunigen. Beispielsweise kann ein Zusammenführungs-Join, der häufig schneller als ein Jash-Join ist, ausgeführt werden, wenn die Daten anhand der Join-Spalten verteilt und vorsortiert sind. Zusammengesetzte Sortierschlüssel können auch zur Verbesserung der Kompression beitragen.

Während Sie einer sortierten Tabelle Zeilen hinzufügen, die bereits Daten enthält, wächst die nicht sortierte Region. Dies wirkt sich deutlich auf die Leistung aus. Der Effekt ist größer, wenn die Tabelle eine überlappende Sortierung verwendet, besonders, wenn die Sortierspalten Daten enthalten, die monoton zunehmen, beispielsweise Datums- oder Zeitstempelspalten. Führen Sie regelmäßig eine VACUUM-Operation aus, besonders nach dem Laden großer Mengen von Daten, um die Daten neu zu sortieren und zu analysieren. Weitere Informationen finden Sie unter Verwalten der Größe der nicht sortierten Region. Nach der Bereinigung der Daten, um sie neu zu sortieren, stellt es eine bewährte Methode dar, einen -Befehl auszuführen, um die statistischen Metadaten für den Abfrageplaner zu aktualisieren. Weitere Informationen finden Sie unter Analysieren von Tabellen.

Überlappender Sortierungsschlüssel

Eine überlappende Sortierung gewichtet alle Spalten oder Teilbereiche von Spalten im Sortierschlüssel gleich. Wenn verschiedene Abfragen verschiedene Spalten als Filter verwenden, können Sie die Leistung dieser Abfragen häufig verbessern, indem Sie einen überlappenden Sortierstil verwenden. Wenn eine Abfrage einschränkende Prädikate für sekundäre Sortierspalten verwendet, wird die Abfrageleistung durch die überlappende Sortierung im Vergleich zur zusammengesetzten Sortierung deutlich verbessert.

> **Important**
>
> Verwenden Sie keinen überlappenden Sortierschlüssel in Spalten mit monoton ansteigenden Attributen, wie Identitätsspalten, Daten oder Zeitstempel.

Die Leistungsverbesserungen, die Sie durch die Implementierung eines überlappenden Sortierschlüssels erzielen, sollten mit den längeren Lade- und Bereinigungszeiten abgewogen werden.

Überlappende Sortierungen sind in Verbindung mit hoch selektiven Abfragen, die nach einem oder mehreren Sortierschlüsselspalten in der WHERE-Klausel filtern, am effektivsten, beispielsweise `select c_name from customer where c_region = 'ASIA'`. Die Vorteile der überlappenden Sortierung nehmen mit der Zahl der sortierten Spalten zu, die eingeschränkt wurden.

Eine überlappende Sortierung ist für große Tabellen effektiver. Die Sortierung wird auf jedes Slice angewendet. Daher ist eine überlappende Sortierung am effektivsten, wenn eine Tabelle groß genug ist, um mehrere 1-MB-Blöcke pro Slice zu beanspruchen. Hier kann der Abfrageprozessor einen erheblichen Teil der Blöcke mit restriktiven Prädikaten überspringen. Um die Zahl der von einer Tabelle verwendeten Blöcke anzuzeigen, führen Sie eine Abfrage für die Systemansicht STV_BLOCKLIST aus.

Beim Sortieren einer einzelnen Spalte kann eine überlappende Sortierung zu einer besseren Leistung als eine zusammengesetzte Sortierung führen, wenn die Spalten ein langes gemeinsames Präfix haben. Beispielsweise beginnen URLs in der Regel mit „http://www“. Zusammengesetzte Sortierschlüssel verwenden eine begrenzte Zahl von Zeichen aus dem Präfix, was zu zahlreichen duplizierten Schlüsseln führt. Überlappende Sortierungen verwenden ein internes Kompressionsschema für Zonenzuweisungswerte. Daher können sie Spaltenwerte besser unterscheiden, die ein langes gemeinsames Präfix haben.

Bei der Migration von bereitgestellten Amazon-Redshift-Clustern zu Amazon Redshift Serverless konvertiert Redshift Tabellen mit überlappenden Sortierschlüsseln und DISTSTYLE KEY in zusammengesetzte Sortierschlüssel. Der DISTSTYLE ändert sich nicht. Weitere Informationen zu Verteilungsstilen finden Sie unter Arbeiten mit Datenverteilungsstilen.

VACUUM REINDEX

Während Sie einer sortierten Tabelle Zeilen hinzufügen, die bereits Daten enthält, kann die Leistung mit der Zeit nachlassen. Diese Verschlechterung tritt sowohl für zusammengesetzte als auch für

überlappende Sortierungen auf, wirkt sich jedoch auf überlappende Tabellen stärker aus. Eine VACUUM-Operation stellt die Sortierreihenfolge wieder her. Die Operation kann jedoch im Fall von überlappenden Tabellen länger dauern, da die Zusammenführung neuer überlappender Daten bedeuten kann, dass jeder Datenblock geändert werden muss.

Wenn Tabellen zum ersten Mal geladen werden, analysiert Amazon Redshift die Verteilung der Werte in den Sortierschlüsselspalten und verwendet diese Informationen, um eine optimale Überlappung der Sortierschlüsselspalten zu erzielen. Mit zunehmender Größe der Tabelle kann sich die Verteilung der Werte in den Sortierschlüsselspalten verändern oder verzerren, besonders im Fall von Datums- oder Zeitstempelspalten. Wenn die Verzerrung zu groß wird, wirkt sich dies möglicherweise auf die Leistung aus. Um die Sortierschlüssel neu zu indizieren und die Leistung wiederherzustellen, führen Sie den Befehl VACUUM mit dem Schlüsselwort REINDEX aus. Da ein zusätzlicher Analysedurchgang für die Daten notwendig ist, kann VACUUM REINDEX im Fall überlappender Tabellen länger als eine VACUUM-Standardoperation dauern. Um Informationen zu Verzerrungen der Schlüsselverteilung und zum letzten Neuindizierungszeitpunkt anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_INTERLEAVED_COLUMNS aus.

Weitere Informationen dazu, wie häufig VACUUM ausgeführt werden sollte und wann VACUUM REINDEX ausgeführt werden sollte, finden Sie unter Entscheidung über die Neuindizierung.

Definieren von Tabelleneinschränkungen

Einschränkungen hinsichtlich Eindeutigkeit, Primärschlüssel und Fremdschlüssel dienen lediglich Informationszwecken. Sie werden von Amazon Redshift nicht erzwungen, wenn Sie eine Tabelle ausfüllen. Wenn Sie beispielsweise Daten in eine Tabelle mit Abhängigkeiten einfügen, kann der Einfügevorgang erfolgreich sein, auch wenn er gegen die Einschränkung verstößt. Dennoch werden Primärschlüssel und Fremdschlüssel als Planungshilfen verwendet und sollten deklariert werden, wenn Ihr ETL-Prozess oder ein anderer Prozess in Ihrer Anwendung ihre Integrität erzwingt.

Beispielsweise verwendet der Abfrageplaner Primär- und Fremdschlüssel in bestimmten statistischen Berechnungen. Dadurch sollen Eindeutigkeit und referentielle Beziehungen abgeleitet werden, die die Entkorrelierung von Unterabfragen beeinflussen. Somit können eine große Anzahl von Joins in Auftrag gegeben und redundante Joins entfernt werden.

Der Planer nutzt diese Schlüsselbeziehungen, nimmt jedoch an, dass alle Schlüssel in Amazon-Redshift-Tabellen wie geladen gültig sind. Wenn Ihre Anwendung ungültige Fremd- oder Primärschlüssel zulässt, könnten einige Abfragen falsche Ergebnisse zurückgeben. Beispielsweise kann eine SELECT DISTINCT-Abfrage duplizierte Zeilen zurückgeben, wenn der Primärschlüssel

nicht eindeutig ist. Definieren Sie keine Schlüsseleinschränkungen für Ihre Tabellen, wenn Sie sich hinsichtlich ihrer Gültigkeit nicht sicher sind. Deklarieren Sie Einschränkungen hinsichtlich Primärschlüsseln, Fremdschlüsseln und Eindeutigkeit jedoch immer, wenn Sie wissen, dass sie gültig sind.

Amazon Redshift erzwingt NOT NULL-Spalteneinschränkungen.

Weitere Informationen zu diesen Tabelleneinschränkungen finden Sie unter CREATE TABLE. Hinweise zum Entfernen einer Tabelle mit Abhängigkeiten finden Sie unter DROP TABLE.

Laden von Daten

Themen

Ein COPY-Befehl ist die effizienteste Methode für das Laden einer Tabelle. Sie können Ihren Tabellen Daten auch mittels des Befehls INSERT hinzufügen. Dies ist jedoch sehr viel weniger effizient als die Verwendung von COPY. Der Befehl COPY ist in der Lage, mehrere Datendateien oder mehrere Datenströme gleichzeitig zu lesen. Amazon Redshift teilt den Workload den Clusterknoten zu und führt die Ladeoperationen parallel aus, einschließlich des Sortierens der Zeilen und der Verteilung von Daten auf die Knoten-Slices.

> **Note**
>
> Externe Tabellen von Amazon Redshift Spectrum sind schreibgeschützt. Sie können die Befehle COPY oder INSERT nicht für externe Tabellen ausführen.

Um auf Daten auf anderen AWS Ressourcen zugreifen zu können, muss Ihr Cluster berechtigt sein, auf diese Ressourcen zuzugreifen und die erforderlichen Aktionen für den Zugriff auf die Daten durchzuführen. Sie können AWS Identity and Access Management (IAM) verwenden, um den Zugriff der Benutzer auf Ihre Clusterressourcen und -daten einzuschränken.

Wenn Sie nach dem ersten Laden von Daten eine erhebliche Menge von Daten hinzufügen, verändern oder löschen, sollten Sie anschließend einen VACUUM-Befehl ausführen, um Ihre

Daten neu zu organisieren und Platz zurückzugewinnen. Sie sollten auch einen ANALYZE-Befehl ausführen, um die Tabellenstatistiken zu aktualisieren.

In diesem Abschnitt wird beschrieben, wie Daten geladen und Fehler bei Datenladevorgängen behoben werden. Außerdem werden bewährte Methoden für das Laden von Daten vorgestellt.

Verwenden eines COPY-Befehls zum Laden von Daten

Themen

Der Befehl COPY nutzt die massive Parallelverarbeitungsarchitektur (Massively Parallel Processing, MPP) von Amazon Redshift, um Daten parallel aus Dateien in Amazon S3, DynamoDB-Tabellen oder Textausgaben von einem oder mehreren Remote-Hosts zu lesen und zu laden.

> **Note**
>
> Es wird nachdrücklich empfohlen, den Befehl COPY zu verwenden, um große Mengen von Daten zu laden. Die Verwendung einzelner INSERT-Anweisungen, um eine Tabelle auszufüllen, kann äußerst langsam sein. Wenn Ihre Daten in anderen Amazon-Redshift-Datenbanktabellen bereits vorhanden sind, können Sie alternativ den Befehl INSERT INTO ... verwenden. SELECT oder CREATE TABLE AS verwenden, um die Leistung zu verbessern. Weitere Informationen finden Sie unter INSERT oder CREATE TABLE AS.

Um Daten aus einer anderen AWS Ressource zu laden, muss Ihr Cluster über die Berechtigung verfügen, auf die Ressource zuzugreifen und die erforderlichen Aktionen auszuführen.

Um das Recht zum Laden von Daten in eine Tabelle mittels eines COPY-Befehls zu gewähren oder zu widerrufen, gewähren oder widerrufen Sie das INSERT-Recht.

Ihre Daten müssen das korrekte Format für das Laden in Ihre Amazon-Redshift-Tabelle besitzen. In diesem Abschnitt werden Richtlinien für die Vorbereitung und Verifizierung Ihrer Daten vor dem Laden und für die Validierung einer COPY-Anweisung vor der Ausführung beschrieben.

Um die Informationen in Ihren Dateien zu schützen, können Sie die Datendateien vor dem Hochladen zu Ihrem Amazon S3 Bucket verschlüsseln. COPY entschlüsselt die Daten während des Ladevorgangs. Sie können auch den Zugriff auf die Ladedaten einschränken, indem Sie Benutzern temporäre Sicherheitsanmeldeinformationen bereitstellen. Temporäre Sicherheitsanmeldedaten bieten erweiterte Sicherheit, da sie nur kurzlebig sind und nach ihrem Ablauf nicht erneut verwendet werden können.

Amazon Redshift verfügt über Funktionen, die in COPY integriert sind, damit unkomprimierte, getrennte Daten schnell geladen werden können. Sie können aber die Dateien mittels gzip, lzop, oder bzip2 komprimieren, um während des Hochladens der Dateien Zeit zu sparen.

Wenn die folgenden Schlüsselwörter in der COPY-Abfrage enthalten sind, wird das automatische Aufteilen unkomprimierter Daten nicht unterstützt: ESCAPE, REMOVEQUOTES und FIXEDWIDTH. Das CSV-Schlüsselwort wird jedoch unterstützt.

Um die Sicherheit Ihrer Daten bei der Übertragung innerhalb der AWS Cloud zu gewährleisten, verwendet Amazon Redshift hardwarebeschleunigtes SSL für die Kommunikation mit Amazon S3 oder Amazon DynamoDB für KOPIER-, ENTLADEN-, Sicherungs- und Wiederherstellungsvorgänge.

Wenn Sie Ihre Tabelle direkt aus einer Amazon-DynamoDB-Tabelle laden, können Sie die Menge des von Amazon DynamoDB bereitgestellten Durchsatzes steuern, den Sie nutzen.

Sie können optional mittels COPY die Eingabedaten analysieren und als Teil des Ladeprozesses automatisch optimierte Kompressionskodierungen auf Ihre Tabelle anwenden.

Anmeldeinformationen und Zugriffsberechtigungen

Um Daten mithilfe einer anderen AWS Ressource wie Amazon S3, Amazon DynamoDB, Amazon EMR oder Amazon EC2 zu laden oder zu entladen, muss Ihr Cluster über die Berechtigung verfügen, auf die Ressource zuzugreifen und die erforderlichen Aktionen auszuführen, um auf die Daten

zuzugreifen. Um beispielsweise Daten aus Amazon S3 zu laden, muss COPY über LIST-Zugriff auf den Bucket und GET-Zugriff auf die Bucket-Objekte verfügen.

Um die Autorisierung für den Zugriff auf eine Ressource zu erhalten, muss Ihr Cluster authentifiziert werden. Sie können eine rollenbasierte Zugriffssteuerung oder eine schlüsselbasierte Zugriffssteuerung wählen. Dieser Abschnitt bietet eine Übersicht über die beiden Methoden. Weitere Details und Beispiele finden Sie unter Berechtigungen für den Zugriff auf andere AWS -Ressourcen.

Rollenbasierte Zugriffskontrolle

Mit einer rollenbasierten Zugriffssteuerung übernimmt Ihr Cluster vorübergehend in Ihrem Namen eine AWS Identity and Access Management (IAM)-Rolle. Anschließend kann Ihr Cluster auf der Grundlage der der Rolle erteilten Berechtigungen auf die erforderlichen Ressourcen zugreifen. AWS

Wir empfehlen die Verwendung der rollenbasierten Zugriffskontrolle, da sie zusätzlich zum Schutz Ihrer Anmeldeinformationen eine sicherere und detailliertere Steuerung des Zugriffs auf AWS Ressourcen und vertrauliche Benutzerdaten ermöglicht. AWS

Um die rollenbasierte Zugriffssteuerung zu verwenden, müssen Sie zunächst unter Verwendung des Amazon-Redshift-Servicerollentyps eine IAM-Rolle erstellen und die Rolle anschließend Ihrem Cluster hinzufügen. Die Rolle muss mindestens die in aufgelisteten Berechtigungen besitzen IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY. Schritte zum Erstellen einer IAM-Rolle und zum Anhängen dieser Rolle an Ihren Cluster finden Sie unter Creating an IAM Role to Allow Your Amazon Redshift Cluster to Access AWS Services im Amazon Redshift Management Guide.

Sie können über die Amazon-Redshift-Managementkonsole, die CLI oder eine API einem Cluster eine Rolle hinzufügen oder die Rollen anzeigen, die mit einem Cluster verknüpft sind. Weitere Informationen finden Sie unter Authorizing COPY and UNLOAD Operations Using IAM Roles (Autorisierung von COPY- und UNLOAD-Operationen mit IAM-Rollen) im Amazon-Redshift-Verwaltungshandbuch.

Beim Erstellen einer IAM-Rolle gibt IAM einen Amazon-Ressourcennamen (ARN) für die Rolle zurück. Um unter Verwendung einer IAM-Rolle einen COPY-Befehl auszuführen, geben Sie unter Verwendung der Parameter IAM_ROLE oder CREDENTIALS den ARN der Rolle an.

Im folgenden Beispiel für den COPY-Befehl wird der Parameter IAM_ROLE mit der Rolle `MyRedshiftRole` zur Authentifizierung verwendet.

```
copy customer from 's3://mybucket/mydata'
iam_role 'arn:aws:iam::12345678901:role/MyRedshiftRole';
```

Der AWS Benutzer muss mindestens über die unter aufgeführten Berechtigungen verfügen. IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY

Schlüsselbasierte Zugriffssteuerung

Bei der schlüsselbasierten Zugriffskontrolle geben Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel für einen Benutzer an, der berechtigt ist, auf die AWS Ressourcen zuzugreifen, die die Daten enthalten.

> **Note**
>
> Es wird nachdrücklich empfohlen, eine IAM-Rolle für die Authentifizierung zu verwenden, statt eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel im Textformat bereitzustellen. Wenn Sie sich für die schlüsselbasierte Zugriffskontrolle entscheiden, verwenden Sie niemals Ihre AWS Kontoanmeldeinformationen (Root). Erstellen Sie stets einen IAM-Benutzer und geben Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel für diesen Benutzer an. Schritte zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS -Konto.

Vorbereiten der Eingabedaten

Wenn die Eingabedaten nicht mit den Tabellenspalten kompatibel sind, die diese erhalten, wird der COPY-Befehl fehlschlagen.

Wenden Sie die folgenden Richtlinien an, um sicherzustellen, dass die Eingabedaten gültig sind:

- Die Daten dürfen nur UTF-8-Zeichen mit höchstens vier Bytes enthalten.
- Überprüfen Sie, ob CHAR- und VARCHAR-Zeichenfolgen die Länge der entsprechenden Spalten nicht überschreiten. VARCHAR-Zeichenfolgen werden in Bytes und nicht in Zeichen gemessen. Daher erfordert eine Zeichenfolge mit vier chinesischen Zeichen, die jeweils vier Bytes enthalten, eine VARCHAR(16)-Spalte.
- Multibyte-Zeichen können nur mit VARCHAR-Spalten verwendet werden. Überprüfen Sie, ob Multibyte-Zeichen nicht mehr als vier Bytes enthalten.
- Überprüfen Sie, ob Daten für CHAR-Spalten nur Einzelbyte-Zeichen enthalten.
- Verwenden Sie keine Sonderzeichen oder eine spezielle Syntax, um das letzte Feld in einem Datensatz anzuzeigen. Dieses Feld kann ein Trennzeichen sein.

- Wenn Ihre Daten Null-Terminatoren enthalten, auch als NUL (UTF-8 0000) oder Binär-Null (0x000) bezeichnet, können Sie diese Zeichen als NULL-Werte in CHAR- oder VARCHAR-Spalten laden, indem Sie die Option NULL AS im COPY-Befehl verwenden: `null as '\0'` oder `null as '\000'`. Wenn Sie NULL AS nicht verwenden, schlägt COPY fehl, wenn Null-Terminatoren vorhanden sind.
- Wenn Ihre Zeichenfolgen Sonderzeichen wie Trennzeichen und eingebettete neue Zeilen enthalten, verwenden Sie die ESCAPE-Option des COPY-Befehls.
- Überprüfen Sie, ob alle einfachen und doppelten Anführungszeichen korrekt übereinstimmen.
- Überprüfen Sie, ob Gleitkomma-Zeichenfolgen das Gleitkomma-Standardformat, beispielsweise 12.123, oder ein exponentielles Format, beispielsweise 1.0E4, aufweisen.
- Überprüfen Sie, ob alle Zeitstempel- und Datumszeichenfolgen die Spezifikationen für befolgen DATEFORMAT- und TIMEFORMAT-Zeichenfolgen. Das Standardzeitstempelformat ist JJJJ-MM-TT hh:mm:ss. Das Standarddatumsformat ist JJJJ-MM-TT.
- Weitere Informationen zu Grenzen und Einschränkungen für einzelne Datentypen finden Sie unter Datentypen. Weitere Informationen zu Fehlern für Multibyte-Zeichen finden Sie unter . Fehler beim Laden von Multibyte-Zeichen

So laden Sie Daten aus Amazon S3

Themen

- Laden von Daten aus komprimierten und unkomprimierten Dateien
- Hochladen von Dateien in Amazon S3
- Verwenden des COPY-Befehls zum Laden aus Amazon S3

Der Befehl COPY nutzt die massive Parallelverarbeitungsarchitektur (Massively Parallel Processing, MPP) von Amazon Redshift, um Daten parallel aus einer Datei oder mehreren Dateien in einem Amazon S3 Bucket zu lesen und zu laden. Sie können maximal von der Parallelverarbeitung profitieren, indem Sie Ihre Daten in mehrere Dateien aufteilen, wenn die Dateien komprimiert sind. (Es gibt Ausnahmen von dieser Regel. Diese werden unter Laden von Datendateien ausführlich beschrieben.) Sie können auch maximal von der Parallelverarbeitung profitieren, indem Sie für Ihre Tabellen Verteilungsschlüssel festlegen. Weitere Informationen zu Verteilungsschlüsseln finden Sie unter Arbeiten mit Datenverteilungsstilen.

Die Daten werden Zeile für Zeile in die Zieltabelle geladen. Die Felder in der Datendatei stimmen mit den Tabellenspalten von links nach rechts überein. Die Felder in den Datendateien können eine

feste Breite oder durch Zeichen abgetrennt sein. Das Standardtrennzeichen ist das Pipe-Zeichen (|). Standardmäßig werden alle Tabellenspalten geladen. Sie können jedoch optional eine durch Komma getrennte Liste von Spalten definieren. Wenn eine Tabellenspalte nicht in der Liste der Spalten enthalten ist, die im COPY-Befehl angegeben ist, wird sie mit einem Standardwert geladen. Weitere Informationen finden Sie unter Laden von Standardspaltenwerten.

Laden von Daten aus komprimierten und unkomprimierten Dateien

Wenn Sie komprimierte Daten laden, empfehlen wir, die Daten für jede Tabelle in mehrere Dateien aufzuteilen. Wenn Sie unkomprimierte, getrennte Daten laden, verwendet der COPY-Befehl die massiv parallele Verarbeitung (MPP) und Scanbereiche, um Daten aus großen Dateien in einen Amazon S3 Bucket zu laden.

Laden von Daten aus mehreren komprimierten Dateien

In Fällen, in denen Sie komprimierte Daten haben, empfehlen wir, die Daten für jede Tabelle in mehrere Dateien aufzuteilen. Der COPY-Befehl kann Daten aus mehreren Dateien parallel laden. Sie können mehrere Dateien durch die Angabe eines gemeinsamen Präfixes bzw. Präfixschlüssels für den Satz oder durch die explizite Auflistung der Dateien in einer Manifestdatei laden.

Teilen Sie die Daten so in Dateien auf, dass die Anzahl der Dateien ein Vielfaches der Anzahl der Slices in Ihrem Cluster ist. Auf diese Weise kann Amazon Redshift die Daten gleichmäßig über die Slices verteilen. Die Anzahl der Slices pro Knoten ist von der Knotengröße des Clusters abhängig. Beispielsweise hat jeder dc2.large Compute Node zwei Slices und jeder dc2.8xlarge Compute Node hat 16 Slices. Weitere Informationen zur Anzahl der Slices für die einzelnen Knotengrößen finden Sie unter About Clusters and Nodes (Informationen zu Clustern und Knoten) im Amazon-Redshift-Verwaltungshandbuch.

Die Knoten sind alle an der parallelen Abfrageausführung beteiligt und arbeiten mit Daten, die so gleichmäßig wie möglich über die Slices verteilt sind. Wenn Sie einen Cluster mit zwei dc2.large-Knoten haben, können Sie Ihre Daten in vier Dateien oder ein Vielfaches von vier aufteilen. Amazon Redshift berücksichtigt die Dateigröße beim Aufteilen des Workloads nicht. Deshalb müssen Sie dafür sorgen, dass die Dateien ungefähr die gleiche Größe haben, nach der Komprimierung von 1 MB bis 1 GB.

Wenn Sie Objektpräfixe zur Bezeichnung der Ladedateien verwenden möchten, benennen Sie die einzelnen Dateien mit einem gemeinsamen Präfix. Beispielsweise können Sie die Datei `venue.txt` wie folgt in vier Dateien aufteilen.

```
venue.txt.1
```

```
venue.txt.2
venue.txt.3
venue.txt.4
```

Wenn Sie mehrere Dateien in einen Ordner in Ihrem Bucket ablegen und den Ordnernamen als Präfix angeben, lädt COPY alle Dateien in diesem Ordner. Wenn Sie die Dateien, die geladen werden sollen, explizit in einer Manifestdatei auflisten, können sich die Dateien in verschiedenen Buckets oder Ordnern befinden.

Weitere Informationen zu Manifestdateien finden Sie unter Example: COPY from Amazon S3 using a manifest.

Laden von Daten aus unkomprimierten, getrennten Dateien

Wenn Sie unkomprimierte, getrennte Daten laden, verwendet der COPY-Befehl die Architektur für die massiv parallele Verarbeitung (Massively Parallel Processing, MPP) in Amazon Redshift. Amazon Redshift verwendet automatisch parallel arbeitende Slices, um Datenbereiche aus einer großen Datei in einen Amazon S3 Bucket zu laden. Die Datei muss getrennt sein, damit ein paralleles Laden stattfindet. Zum Beispiel durch Pipe-Zeichen (|) getrennt. Das automatische, parallele Laden von Daten mit dem Befehl COPY ist auch für CSV-Dateien verfügbar. Sie können auch von der Parallelverarbeitung profitieren, indem Sie für Ihre Tabellen Verteilungsschlüssel festlegen. Weitere Informationen zu Verteilungsschlüsseln finden Sie unter Arbeiten mit Datenverteilungsstilen.

Das automatische, parallele Laden von Daten wird nicht unterstützt, wenn die COPY-Abfrage eines der folgenden Schlüsselwörter enthält: ESCAPE, REMOVEQUOTES und FIXEDWIDTH.

Die Daten aus der Datei oder den Dateien werden Zeile für Zeile in die Zieltabelle geladen. Die Felder in der Datendatei stimmen mit den Tabellenspalten von links nach rechts überein. Die Felder in den Datendateien können eine feste Breite oder durch Zeichen abgetrennt sein. Das Standardtrennzeichen ist das Pipe-Zeichen (|). Standardmäßig werden alle Tabellenspalten geladen. Sie können jedoch optional eine durch Komma getrennte Liste von Spalten definieren. Wenn eine Tabellenspalte nicht in der Liste der Spalten enthalten ist, die im COPY-Befehl angegeben ist, wird sie mit einem Standardwert geladen. Weitere Informationen finden Sie unter Laden von Standardspaltenwerten.

Befolgen Sie für das Laden von Daten aus Amazon S3 diesen allgemeinen Prozess, wenn Ihre Daten unkomprimiert und getrennt sind:

1. Laden Sie Ihre Dateien in Amazon S3 hoch.
2. Führen Sie einen COPY-Befehl aus, um die Tabelle zu laden.

3. Überprüfen Sie, ob die Daten korrekt geladen wurden.

Beispiele für COPY-Befehle finden Sie unter Beispiele für COPY. Weitere Informationen zu Daten, die in Amazon Redshift geladen wurden, finden Sie in den STL_LOAD_COMMITS- und STL_LOAD_ERRORS-Systemtabellen.

Weitere Informationen zu Knoten und den darin enthaltenen Slices finden Sie unter Informationen zu Clustern und Knoten im Amazon-Redshift-Verwaltungshandbuch.

Hochladen von Dateien in Amazon S3

Themen

- Verwalten der Datenkonsistenz
- Hochladen verschlüsselter Daten in Amazon S3
- Überprüfen, ob sich die korrekten Dateien in Ihrem Bucket befinden

Beim Hochladen von Textdateien auf Amazon S3 gibt es mehrere verschiedene Ansätze:

- Wenn Sie komprimierte Dateien haben, wird empfohlen, große Dateien aufzuteilen, um von den Vorteilen der Parallelverarbeitung in Amazon Redshift zu profitieren.
- Andererseits teilt COPY automatisch große, unkomprimierte, durch Text getrennte Dateidaten auf, um die Parallelität zu erleichtern und die Daten effektiv aus großen Dateien zu verteilen.

Erstellen Sie einen Amazon S3 Bucket für Ihre Datendateien und laden Sie anschließend die Datendateien zum Bucket hoch. Weitere Informationen zum Erstellen von Buckets und Hochladen von Dateien finden Sie unter Arbeiten mit Amazon S3 Buckets im Benutzerhandbuch von Amazon Simple Storage Service.

> **Important**
>
> Der Amazon-S3-Bucket, der die Datendateien enthält, muss in derselben AWS -Region wie Ihr Cluster erstellt werden, es sei denn, Sie verwenden die Option REGION, um die Region anzugeben, in der sich der Amazon-S3-Bucket befindet.

Stellen Sie sicher, dass die S3-IP-Bereiche zu Ihrer Zulassungsliste hinzugefügt werden. Weitere Informationen zu den erforderlichen S3-IP-Bereichen finden Sie unter Netzwerkisolierung.

Sie können einen Amazon-S3-Bucket in einer bestimmten Region erstellen, indem Sie entweder bei der Erstellung des Buckets die Region über die Amazon-S3-Konsole auswählen oder indem Sie einen Endpunkt angeben, wenn Sie den Bucket mithilfe der Amazon-S3-API oder -CLI erstellen.

Überprüfen Sie nach dem Laden der Daten, ob sich die korrekten Dateien in Amazon S3 befinden.

Verwalten der Datenkonsistenz

Amazon S3 bietet eine hohe read-after-write Konsistenz für COPY-, UNLOAD-, INSERT- (externe Tabelle), CREATE EXTERNAL TABLE AS- und Amazon Redshift Spectrum Spectrum-Operationen auf Amazon S3 S3-Buckets in allen Regionen. AWS Darüber hinaus sind Lesevorgänge in Amazon S3 Select, Amazon-S3-Zugriffskontrolllisten, Amazon-S3-Objekt-Markierungen und Objekt-Metadaten (z. B. HEAD-Objekt) stark konsistent. Ausführlichere Informationen zur Datenkonsistenz finden Sie unter Amazon-S3- Datenkonsistenzmodell im Benutzerhandbuch von Amazon Simple Storage Service.

Hochladen verschlüsselter Daten in Amazon S3

Amazon S3 unterstützt sowohl die serverseitige als auch die clientseitige Verschlüsselung. In diesem Thema werden die Unterschiede zwischen serverseitiger und clientseitiger Verschlüsselung behandelt und die Schritte beschrieben, die Sie ausführen müssen, um für Amazon Redshift die clientseitige Verschlüsselung zu verwenden. Die serverseitige Verschlüsselung ist für Amazon Redshift transparent.

Server-side encryption

Die serverseitige Verschlüsselung betrifft die Datenverschlüsselung in Ruhe , das heißt, Amazon S3 verschlüsselt Ihre Daten, während es sie in seine Rechenzentren schreibt, und entschlüsselt sie für Sie, wenn Sie darauf zugreifen. Wenn Sie Tabellen mittels eines COPY-Befehls laden, gibt es in Bezug auf die Art und Weise des Ladens aus serverseitig verschlüsselten oder nicht verschlüsselten Objekten in Amazon S3 keinen Unterschied. Weitere Informationen zur Datenverschlüsselung mittels serverseitiger Verschlüsselung finden Sie unter Verwendung serverseitiger Verschlüsselung im Benutzerhandbuch von Amazon Simple Storage Service.

Clientseitige Verschlüsselung

Bei der clientseitigen Verschlüsselung verwaltet Ihre Clientanwendung die Verschlüsselung Ihrer Daten, die Verschlüsselungsschlüssel und die entsprechenden Tools. Sie können Daten unter Verwendung einer clientseitigen Verschlüsselung zu einem Amazon S3 Bucket hochladen und die Daten anschließend mittels des Befehls COPY mit der Option ENCRYPTED und eines privaten Verschlüsselungsschlüssels laden, um größere Sicherheit zu erzielen.

Sie verschlüsseln Ihre Daten mittels einer Envelope-Verschlüsselung. Mittels einer Envelope-Verschlüsselung verarbeitet ausschließlich Ihre Anwendung die gesamte Verschlüsselung. Ihre privaten Verschlüsselungsschlüssel und Ihre unverschlüsselten Daten werden niemals an Dritte gesendet. Daher ist es sehr wichtig AWS, dass Sie Ihre Verschlüsselungsschlüssel sicher verwalten. Wenn Sie Ihre Verschlüsselungsschlüssel verlieren, können Sie Ihre Daten nicht entschlüsseln und Sie können Ihre Verschlüsselungsschlüssel von nicht wiederherstellen. AWS Die Envelope-Verschlüsselung kombiniert die Leistung einer schnellen symmetrischen Verschlüsselung mit der größeren Sicherheit, die eine Schlüsselverwaltung mit asymmetrischen Schlüsseln bietet. Ein one-time-use symmetrischer Schlüssel (der symmetrische Envelope-Schlüssel) wird von Ihrem Amazon S3-Verschlüsselungsclient generiert, um Ihre Daten zu verschlüsseln. Anschließend wird dieser Schlüssel mit Ihrem Root-Schlüssel verschlüsselt und zusammen mit Ihren Daten in Amazon S3 gespeichert. Wenn Amazon Redshift während eines Ladevorgangs auf Ihre Daten zugreift, wird der verschlüsselte symmetrische Schlüssel abgerufen und mit Ihrem echten Schlüssel entschlüsselt. Anschließend werden die Daten entschlüsselt.

Um in Amazon Redshift mit clientseitig verschlüsselten Amazon-S3-Daten zu arbeiten, befolgen Sie die in Schutz von Daten mittels clientseitiger Verschlüsselung im Benutzerhandbuch von Amazon Simple Storage Service beschriebenen Schritte. Zusätzlich müssen Sie Folgendes verwenden:

- Symmetrische Verschlüsselung Die Klasse AWS SDK for Java `AmazonS3EncryptionClient` verwendet die zuvor beschriebene Envelope-Verschlüsselung, die auf einer symmetrischen Schlüsselverschlüsselung basiert. Verwenden Sie diese Klasse, um einen Amazon-S3-Client für das Hochladen von clientseitig verschlüsselten Daten zu erstellen.
- Symmetrischer 256-Bit-AES-Root-Schlüssel Ein Root-Schlüssel verschlüsselt den Envelope-Schlüssel. Sie übergeben den Root-Schlüssel an Ihre Instance der Klasse `AmazonS3EncryptionClient`. Speichern Sie diesen Schlüssel, da Sie diesen benötigen, um Daten zu Amazon Redshift zu kopieren.
- Objektmetadaten zum Speichern des verschlüsselten Envelope-Schlüssels Standardmäßig speichert Amazon S3 den Envelope-Schlüssel in Form von Objektmetadaten für die `AmazonS3EncryptionClient`-Klasse. Der in Form von Objektmetadaten gespeicherte verschlüsselte Envelope-Schlüssel wird während des Entschlüsselungsprozesses verwendet.

> **Note**
>
> Wenn Sie während der ersten Verwendung der Verschlüsselungs-API eine Verschlüsselungsfehlermeldung erhalten, enthält Ihre Version des JDK möglicherweise eine Java Cryptography Extension (JCE)-Jurisdiktionsrichtlinie, die die maximale Schlüssellänge

für Verschlüsselungs- und Entschlüsselungstransformationen auf 128 Bits begrenzt. Informationen zur Behebung dieses Problems finden Sie unter Spezifizierung der clientseitigen Verschlüsselung mithilfe des AWS SDK for Java im Amazon Simple Storage Service-Benutzerhandbuch.

Weitere Informationen zum Laden von clientseitig verschlüsselten Dateien in Ihre Amazon-Redshift-Tabellen unter Verwendung des Befehls COPY finden Sie unter Laden verschlüsselter Datendateien aus Amazon S3.

Beispiel: Hochladen von clientseitig verschlüsselten Daten

Ein Beispiel für die Verwendung des AWS SDK for Java zum Hochladen von clientseitig verschlüsselten Daten finden Sie unter Schützen von Daten mit clientseitiger Verschlüsselung im Amazon Simple Storage Service-Benutzerhandbuch.

In der zweiten Option werden die Entscheidungen gezeigt, die Sie während der clientseitigen Verschlüsselung treffen müssen, damit die Daten in Amazon Redshift geladen werden können. Insbesondere zeigt das Beispiel die Verwendung von Objektmetadaten für das Speichern des verschlüsselten Envelope-Schlüssels und die Verwendung eines symmetrischen 256-Bit-AES-Root-Schlüssels.

Dieses Beispiel enthält Beispielcode, der das AWS SDK for Java verwendet, um einen symmetrischen 256-Bit-AES-Stammschlüssel zu erstellen und ihn in einer Datei zu speichern. Anschließend wird mittels eines S3-Verschlüsselungsclients, der die Beispieldaten zunächst clientseitig verschlüsselt, ein Objekt zu Amazon S3 hochgeladen. Das Beispiel zeigt auch den Download des Objekts und die Überprüfung, ob die Daten identisch ist.

Überprüfen, ob sich die korrekten Dateien in Ihrem Bucket befinden

Nachdem Sie Ihre Dateien zu Ihrem Amazon S3 Bucket hochgeladen haben, sollten Sie die Inhalte des Buckets auflisten, um zu überprüfen, ob alle korrekten Dateien vorhanden sind und keine unerwünschten Dateien vorhanden sind. Wenn beispielsweise der Bucket `mybucket` eine Datei namens `venue.txt.back` enthält, wird diese Datei (vielleicht unbeabsichtigt) durch den folgenden Befehl geladen:

```
copy venue from 's3://mybucket/venue' … ;
```

Wenn Sie spezifisch kontrollieren möchten, welche Dateien geladen werden, können Sie eine Manifestdatei verwenden, um die Datendateien explizit aufzulisten. Weitere Informationen zur

Verwendung einer Manifestdatei finden Sie in der Beschreibung der copy_from_s3_manifest_file-Option für den COPY-Befehl und unter Example: COPY from Amazon S3 using a manifest in den Beispielen für COPY.

Weitere Informationen zum Auflisten der Inhalte des Buckets finden Sie unter Listing Object Keys (Auflisten von Objektschlüsseln) im Amazon-S3-Entwicklerhandbuch.

Verwenden des COPY-Befehls zum Laden aus Amazon S3

Themen

- Verwenden eines Manifests für die Angabe von Datendateien
- Laden komprimierter Datendateien aus Amazon S3
- Laden von Daten mit fester Breite aus Amazon S3
- Laden von Multibyte-Daten aus Amazon S3
- Laden verschlüsselter Datendateien aus Amazon S3

Verwenden Sie den COPY-Befehl, um eine Tabelle parallel aus Datendateien in Amazon S3 zu laden. Sie können die Dateien, die geladen werden sollen, durch Verwendung eines Amazon-S3-Objektpräfixes oder einer Manifestdatei angeben.

Die Syntax für die Angabe der Dateien, die unter Verwendung eines Präfixes geladen werden sollen, ist wie folgt:

```
copy <table_name> from 's3://<bucket_name>/<object_prefix>'
authorization;
```

Die Manifestdatei ist eine Datei im JSON-Format, die die Datendateien auflistet, die geladen werden sollen. Die Syntax für die Angabe der Dateien, die unter Verwendung einer Manifestdatei geladen werden sollen, ist wie folgt:

```
copy <table_name> from 's3://<bucket_name>/<manifest_file>'
authorization
manifest;
```

Die Tabelle, die geladen werden soll, muss in der Datenbank bereits vorhanden sein. Informationen zum Erstellen einer Tabelle finden Sie unter CREATE TABLE in der SQL-Referenz.

Die Werte für die Autorisierung geben die AWS Autorisierung an, die Ihr Cluster für den Zugriff auf die Amazon S3 S3-Objekte benötigt. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY. Die bevorzugte Methode für die Authentifizierung besteht in der Angabe des Parameters IAM_ROLE und des Amazon-Ressourcennamens (ARN) für eine IAM-Rolle mit den notwendigen Berechtigungen. Weitere Informationen finden Sie unter Rollenbasierte Zugriffskontrolle .

Um die Authentifizierung mittels des Parameters IAM_ROLE durchzuführen, ersetzen Sie *<aws-account-id>* und *<role-name>* wie in der folgenden Syntax gezeigt.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Das folgende Beispiel zeigt die Authentifizierung unter Verwendung einer IAM-Rolle.

```
copy customer
from 's3://mybucket/mydata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Weitere Informationen zu anderen Autorisierungsoptionen finden Sie unter . Autorisierungsparameter

Wenn Sie Ihre Daten validieren möchten, ohne die Tabelle tatsächlich zu laden, verwenden Sie die Option NOLOAD mit dem Befehl COPY.

Das folgende Beispiel zeigt die ersten wenigen Zeilen mit durch Pipe-Zeichen getrennten Daten in einer Datei namens `venue.txt`.

```
1|Toyota Park|Bridgeview|IL|0
2|Columbus Crew Stadium|Columbus|OH|0
3|RFK Stadium|Washington|DC|0
```

Bevor Sie die Datei zu Amazon S3 hochladen, teilen Sie die Datei in mehrere Dateien auf, sodass der Befehl COPY diese unter Verwendung der Parallelverarbeitung hochladen kann. Die Anzahl der Dateien sollte ein Vielfaches der Anzahl der Slices in Ihrem Cluster sein. Teilen Sie Ihre Ladedatendateien, so dass die Dateien etwa gleich groß sind, zwischen 1 MB und 1 GB nach der Kompression. Weitere Informationen finden Sie unter Laden von Daten aus komprimierten und unkomprimierten Dateien.

Beispielsweise kann die Datei `venue.txt` wie folgt in vier Dateien aufgeteilt werden:

```
venue.txt.1
```

```
venue.txt.2
venue.txt.3
venue.txt.4
```

Der folgende COPY-Befehl lädt die Tabelle VENUE mittels der durch Pipe-Zeichen getrennten Daten in den Datendateien mit dem Präfix „venue“ im Amazon-S3-Bucket `mybucket`.

> **Note**
>
> Der Amazon-S3-Bucket `mybucket` in den folgenden Beispiele ist nicht vorhanden. Beispiele für COPY-Befehle, die echte Daten in einem vorhandenen Amazon-S3-Bucket verwenden, finden Sie unter Load sample data (Beispieldateien laden).

```
copy venue from 's3://mybucket/venue'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Wenn keine Amazon-S3-Objekte mit dem Schlüsselpräfix „venue“ vorhanden sind, schlägt der Ladevorgang fehl.

Verwenden eines Manifests für die Angabe von Datendateien

Sie können ein Manifest verwenden, um sicherzustellen, dass bei Verwendung des COPY-Befehls während eines Datenladevorgangs alle erforderlichen Dateien und nur diese geladen werden. Sie können ein Manifest verwenden, um Dateien aus verschiedenen Buckets hochzuladen oder Dateien, die nicht das gleiche Präfix verwenden. Statt einen Objektpfad für den COPY-Befehl anzugeben, stellen Sie den Namen einer Textdatei im JSON-Format bereit, in der die Dateien, die hochgeladen werden sollen, explizit aufgelistet werden. Die URL im Manifest muss den Bucket-Namen und den vollständigen Objektpfad für die Datei angeben, nicht nur ein Präfix.

Weitere Informationen zu Manifestdateien finden Sie im COPY-Beispiel unter Verwenden eines Manifests für die Angabe von Datendateien

Das folgende Beispiel zeigt das JSON-Format, um Dateien aus verschiedenen Buckets hochzuladen, deren Namen mit Datumsstempeln beginnen.

```
{
  "entries": [
```

```
    {"url":"s3://mybucket-alpha/2013-10-04-custdata", "mandatory":true},
    {"url":"s3://mybucket-alpha/2013-10-05-custdata", "mandatory":true},
    {"url":"s3://mybucket-beta/2013-10-04-custdata", "mandatory":true},
    {"url":"s3://mybucket-beta/2013-10-05-custdata", "mandatory":true}
  ]
}
```

Das optionale `mandatory`-Flag gibt an, ob COPY einen Fehler zurückgeben soll, wenn die Datei nicht gefunden wird. Der Standardwert von `mandatory` ist `false`. Unabhängig von obligatorischen Einstellungen wird COPY beendet, wenn keine Dateien gefunden werden.

Im folgenden Beispiel wird der COPY-Befehl mit dem Manifest aus dem vorherigen Beispiel ausgeführt, dessen Name ist `cust.manifest`.

```
copy customer
from 's3://mybucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Verwenden eines mit UNLOAD erstellten Manifests

Ein mit einem UNLOAD-Vorgang und dem MANIFEST-Parameter erstelltes Manifest kann Schlüssel enthalten, die für den COPY-Vorgang nicht benötigt werden. Das folgende UNLOAD-Manifest enthält beispielsweise einen `meta`-Schlüssel, der von einer externen Amazon-Redshift-Spectrum-Tabelle und zum Laden von Dateien im Dateiformat ORC oder `Parquet` benötigt wird. Der `meta`-Schlüssel enthält einen `content_length`-Schlüssel mit einem Wert, der der tatsächlichen Größe der Datei in Bytes entspricht. Für den COPY-Vorgang sind jedoch nur der `url`-Schlüssel und ein optionaler `mandatory`-Schlüssel erforderlich.

```
{
  "entries": [
    {"url":"s3://mybucket/unload/manifest_0000_part_00", "meta": { "content_length":
 5956875 }},
    {"url":"s3://mybucket/unload/unload/manifest_0001_part_00", "meta":
 { "content_length": 5997091 }}
 ]
}
```

Weitere Informationen zu Manifestdateien finden Sie unter Example: COPY from Amazon S3 using a manifest.

Laden komprimierter Datendateien aus Amazon S3

Um Datendateien zu laden, die mittels gzip, lzop oder bzip2 komprimiert wurden, verwenden Sie die entsprechende Option: GZIP, LZOP oder BZIP2.

Der folgende Befehl lädt beispielsweise Daten aus Dateien, die mittels lzop komprimiert wurden.

```
copy customer from 's3://mybucket/customer.lzo'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' lzop;
```

> **Note**
>
> Wenn Sie eine Datendatei mit LZop-Komprimierung komprimieren und die Option --filter verwenden, unterstützt der COPY-Befehl dies nicht.

Laden von Daten mit fester Breite aus Amazon S3

Die Datenspalten in Datendateien mit fester Breite haben einheitliche Längen. Jedes Feld in einer Datendatei mit fester Breite besitzt genau dieselbe Länge und Position. Im Fall von Zeichendaten (CHAR und VARCHAR) in einer Datendatei mit fester Breite müssen Sie am Anfang und am Ende Leerzeichen als Platzhalter verwenden, um die einheitliche Breite einzuhalten. Im Fall von Ganzzahlen müssen Sie am Anfang Nullen als Platzhalter verwenden. Eine Datendatei mit fester Breite verwendet keine Trennzeichen, um Spalten zu trennen.

Um eine Datendatei mit fester Breite in eine vorhandene Tabelle zu laden, verwenden Sie den Parameter FIXEDWIDTH im COPY-Befehl. Die Tabellenspezifikationen müssen dem Wert von fixedwidth_spec entsprechen, damit die Daten korrekt geladen werden.

Um Daten mit fester Breite aus einer Datei in eine Tabelle zu laden, geben Sie den folgenden Befehl aus:

```
copy table_name from 's3://mybucket/prefix'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth 'fixedwidth_spec';
```

Der Parameter fixedwidth_spec ist eine Zeichenfolge, die einen Bezeichner für jede Spalte und die Breite jeder Spalte enthält, getrennt durch einen Punkt. Die **`column:width`**-Paare werden durch

Komma getrennt. Der Bezeichner kann von Ihnen frei gewählt werden: Zahlen, Buchstaben oder eine Kombination aus diesen. Der Bezeichner hat keine Beziehung zur Tabelle selbst. Daher muss die Spezifikation die Spalten in derselben Reihenfolge wie die Tabelle enthalten.

Die folgenden beiden Beispiele zeigen dieselbe Spezifikation, wobei die erste numerische Bezeichner und die zweite Zeichenfolgenbezeichner verwendet:

```
'0:3,1:25,2:12,3:2,4:6'
```

```
'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6'
```

Im folgenden Beispiel sind Beispieldaten mit fester Breite zu sehen, die unter Verwendung der oben aufgeführten Spezifikationen in die Tabelle VENUE geladen werden könnten:

```
1  Toyota Park               Bridgeview  IL0
2  Columbus Crew Stadium     Columbus    OH0
3  RFK Stadium               Washington  DC0
4  CommunityAmerica Ballpark Kansas City KS0
5  Gillette Stadium          Foxborough  MA68756
```

Der folgende COPY-Befehl lädt diesen Datensatz in die Tabelle VENUE:

```
copy venue
from 's3://mybucket/data/venue_fw.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth 'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6';
```

Laden von Multibyte-Daten aus Amazon S3

Wenn Ihre Daten Multibyte-Zeichen enthalten, die andere als ASCII-Zeichen verwenden (beispielsweise chinesische oder kyrillische Zeichen), müssen Sie die Daten in VARCHAR-Spalten laden. Der VARCHAR-Datentyp unterstützt UTF-8-Zeichen mit vier Bytes. Der CHAR-Datentyp unterstützt jedoch nur ASCII-Zeichen mit einem Byte. Sie können keine Zeichen mit fünf Bytes oder mehr in Amazon-Redshift-Tabellen laden. Weitere Informationen zu CHAR und VARCHAR finden Sie unter Datentypen.

Um zu prüfen, welche Kodierung von einer Eingabedatei verwendet wird, verwenden Sie den Linux-Befehl *file*.

```
$ file ordersdata.txt
ordersdata.txt: ASCII English text
$ file uni_ordersdata.dat
uni_ordersdata.dat: UTF-8 Unicode text
```

Laden verschlüsselter Datendateien aus Amazon S3

Sie können den COPY-Befehl verwenden, um Datendateien zu laden, die unter Verwendung der serverseitigen Verschlüsselung, der clientseitigen Verschlüsselung oder beider Verschlüsselungen zu Amazon S3 hochgeladen wurden.

Der COPY-Befehl unterstützt die folgenden Amazon S3-Verschlüsselungsarten:

- Serverseitige Verschlüsselung mit von Amazon S3 verwalteten Schlüsseln (SSE-S3)
- Serverseitige Verschlüsselung mit (SSE-KMS) AWS KMS keys
- Clientseitige Verschlüsselung mittels eines clientseitigen symmetrischen Root-Schlüssels

Der COPY-Befehl unterstützt nicht die folgenden Amazon S3-Verschlüsselungsarten:

- Serverseitige Verschlüsselung mit vom Kunden bereitgestellten Schlüsseln (SSE-C)
- Clientseitige Verschlüsselung mit einem AWS KMS key
- Clientseitige Verschlüsselung mittels eines vom Kunden bereitgestellten asymmetrischen Root-Schlüssels

Weitere Informationen zur Amazon-S3-Verschlüsselung finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung und Schützen von Daten mithilfe clientseitiger Verschlüsselung im Benutzerhandbuch von Amazon Simple Storage Service.

Der Befehl UNLOAD verschlüsselt Dateien automatisch mittels SSE-S3. Sie können zum Entladen auch SSE-KMS oder eine clientseitige Verschlüsselung mit einem vom Kunden verwalteten symmetrischen Schlüssel verwenden. Weitere Informationen finden Sie unter Entladen verschlüsselter Datendateien

Der COPY-Befehl erkennt automatisch Dateien, die mit SSE-S3 und SSE-KMS verschlüsselt wurden, und lädt diese. Sie können Dateien, die mit einem clientseitigen symmetrischen Root-Schlüssel verschlüsselt wurden, durch Angabe der Option ENCRYPTED und des Schlüssel-Werts laden. Weitere Informationen finden Sie unter Hochladen verschlüsselter Daten in Amazon S3.

Um clientseitig verschlüsselte Datendateien zu laden, stellen Sie unter Verwendung des Parameters MASTER_SYMMETRIC_KEY den Root-Schlüsselwert bereit und schließen die Option ENCRYPTED ein.

```
copy customer from 's3://mybucket/encrypted/customer'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted
delimiter '|';
```

Um verschlüsselte Datendateien zu laden, die mit gzip, lzop oder bzip2 komprimiert wurden, verwenden Sie die Option GZIP, LZOP oder BZIP2 zusammen mit dem Root-Schlüsselwert und der Option ENCRYPTED.

```
copy customer from 's3://mybucket/encrypted/customer'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted
delimiter '|'
gzip;
```

So laden Sie Daten aus Amazon EMR:

Sie können den COPY-Befehl verwenden, um Daten parallel aus einem Amazon-EMR-Cluster zu laden, der für das Schreiben von Textdateien in das Hadoop Distributed File System (HDFS) des Clusters in Form von Dateien mit fester Breite, zeichenbegrenzten Dateien, CSV-Dateien oder JSON-Dateien konfiguriert wurde.

Prozess zum Laden von Daten aus Amazon EMR

In diesem Abschnitt wird das Laden von Daten aus einem Amazon-EMR-Cluster beschrieben. In den folgenden Abschnitten finden Sie detaillierte Anweisungen für die einzelnen Schritte.

- Schritt 1: Konfigurieren von IAM-Berechtigungen

  Die Benutzer, die den Amazon-EMR-Cluster erstellen und den COPY-Befehl von Amazon Redshift ausführen, müssen die notwendigen Berechtigungen besitzen.
- Schritt 2: Erstellen eines Amazon-EMR-Clusters

Konfigurieren Sie den Cluster für die Ausgabe von Textdateien zum Hadoop Distributed File System (HDFS). Sie benötigen die ID des Amazon-EMR-Clusters und das öffentliche Haupt-DNS des Clusters (den Endpunkt für die Amazon-EC2-Instance, die den Cluster hostet).

- Schritt 3: Abrufen des öffentlichen Schlüssels des Amazon-Redshift-Clusters und der IP-Adressen der Cluster-Knoten

  Der öffentliche Schlüssel ermöglicht den Knoten des Amazon-Redshift-Clusters die Herstellung von SSH-Verbindungen zu den Hosts. Sie verwenden die IP-Adresse der einzelnen Cluster-Knoten, um die Host-Sicherheitsgruppen zu konfigurieren, damit diese den Zugriff von Ihrem Amazon-Redshift-Cluster unter Verwendung dieser IP-Adressen gestatten.

- Schritt 4: Hinzufügen des öffentlichen Schlüssels des Amazon-Redshift-Clusters zur Datei mit den autorisierten Schlüsseln der einzelnen Amazon-EC2-Hosts

  Sie fügen den öffentlichen Schlüssel des Amazon-Redshift-Clusters zur Datei des Hosts hinzu, die die autorisierten Schlüssel enthält, damit der Host den Amazon-Redshift-Cluster erkennt und die SSH-Verbindung akzeptiert.

- Schritt 5: Konfigurieren der Hosts, sodass sie alle IP-Adressen des Amazon-Redshift-Clusters akzeptieren

  Ändern Sie die Sicherheitsgruppen der Amazon-EMR-Instance, indem Sie Eingaberegeln für die Akzeptierung der Amazon-Redshift-IP-Adressen hinzufügen.

- Schritt 6: Ausführen des COPY-Befehls, um die Daten zu laden

  Führen Sie den COPY-Befehl aus einer Amazon-Redshift-Datenbank aus, um die Daten in eine Amazon-Redshift-Tabelle zu laden.

Schritt 1: Konfigurieren von IAM-Berechtigungen

Die Benutzer, die den Amazon-EMR-Cluster erstellen und den COPY-Befehl von Amazon Redshift ausführen, müssen die notwendigen Berechtigungen besitzen.

So konfigurieren Sie IAM-Berechtigungen

1. Fügen Sie dem Benutzer, der den Amazon-EMR-Cluster erstellt, die folgenden Berechtigungen hinzu.

```
ec2:DescribeSecurityGroups
```

```
ec2:RevokeSecurityGroupIngress
ec2:AuthorizeSecurityGroupIngress
redshift:DescribeClusters
```

2. Fügen Sie der IAM-Rolle oder dem Benutzer, die bzw. der den COPY-Befehl ausführt, die folgende Berechtigung hinzu.

```
elasticmapreduce:ListInstances
```

3. Fügen Sie der IAM-Rolle des Amazon-EMR-Clusters die folgende Berechtigung hinzu.

```
redshift:DescribeClusters
```

Schritt 2: Erstellen eines Amazon-EMR-Clusters

Der COPY-Befehl lädt Daten aus Dateien im Hadoop Distributed File System (HDFS) von Amazon EMR. Konfigurieren Sie beim Erstellen des Amazon-EMR-Clusters den Cluster für die Ausgabe von Datendateien an das HDFS des Clusters.

So erstellen Sie einen Amazon-EMR-Cluster:

1. Erstellen Sie einen Amazon EMR-Cluster in derselben AWS Region wie der Amazon Redshift Redshift-Cluster.

   Wenn sich der Amazon-Redshift-Cluster in einer VPC befindet, muss sich der Amazon-EMR-Cluster in derselben VPC-Gruppe befinden. Wenn der Amazon-Redshift-Cluster den EC2-Classic-Modus verwendet (d. h., sich nicht in einer VPC befindet), muss der Amazon-EMR-Cluster ebenfalls den EC2-Classic-Modus verwenden. Weitere Informationen finden Sie unter Managing Clusters in Virtual Private Cloud (VPC) (Verwalten von Clustern in Virtual Private Cloud (VPC)) im Amazon-Redshift-Verwaltungshandbuch.
2. Konfigurieren Sie den Cluster für die Ausgabe von Datendateien an das HDFS des Clusters. Die HDFS-Dateinamen dürfen keine Sternchen (*) oder Fragezeichen (?) enthalten.

> **Important**
>
> Die Dateinamen dürfen keine Sternchen (*) oder Fragezeichen (?) enthalten.

3. Geben Sie No (Nein) für die Option Auto-terminate (Automatisches Beenden) in der Amazon-EMR-Clusterkonfiguration an, sodass der Cluster während der Ausführung des COPY-Befehls verfügbar bleibt.

   > **Important**
   >
   > Wenn eine der Datendateien vor Abschluss von COPY geändert oder gelöscht wird, kann dies zu unerwarteten Ergebnissen führen. Es ist auch möglich, dass die COPY-Operation fehlschlägt.

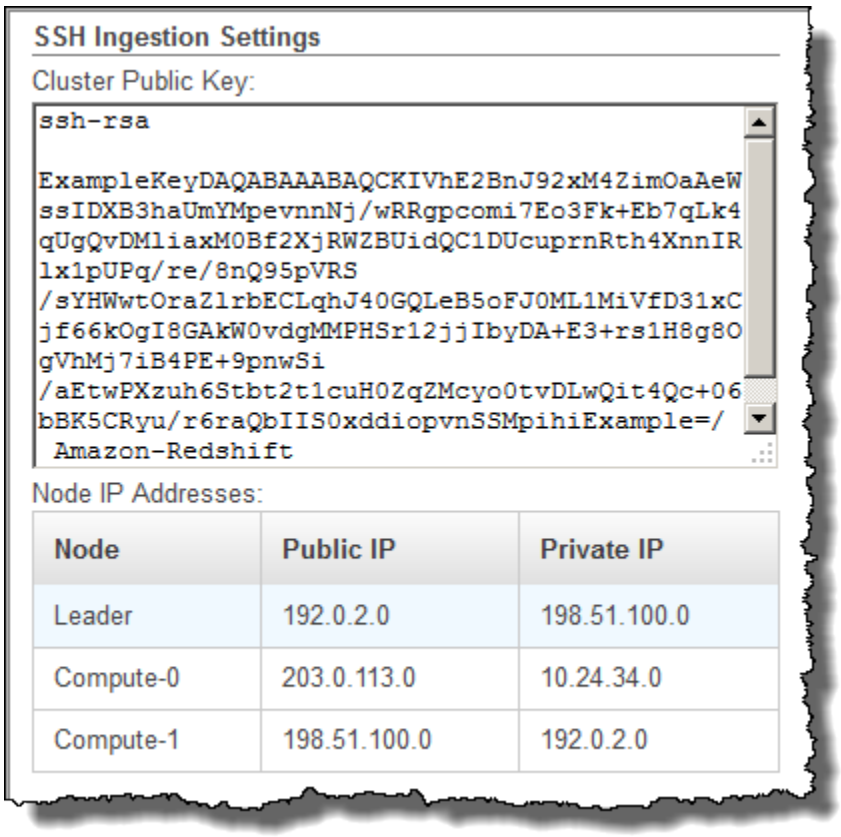
4. Notieren Sie die Cluster-ID und das öffentliche Haupt-DNS (den Endpunkt für die Amazon-EC2-Instance, die den Cluster hostet). Diese Informationen werden in späteren Schritten benötigt.

Schritt 3: Abrufen des öffentlichen Schlüssels des Amazon-Redshift-Clusters und der IP-Adressen der Cluster-Knoten

So rufen Sie den öffentlichen Schlüssel des Amazon-Redshift-Clusters und die IP-Adressen der Cluster-Knoten für Ihren Cluster über die Konsole ab:

1. Greifen Sie auf die Managementkonsole von Amazon Redshift zu.
2. Wählen Sie im Navigationsbereich den Link Clusters (Cluster) aus.
3. Wählen Sie Ihren Cluster aus der Liste aus.
4. Suchen Sie die Gruppe SSH Ingestion Settings (SSH-Eingangseinstellungen).

   Notieren Sie sich die Informationen unter Cluster Public Key (Öffentlicher Schlüssel des Clusters) und Node IP addresses (Knoten-IP-Adressen). Diese Informationen werden in späteren Schritten verwendet.



Sie werden die privaten IP-Adressen in Schritt 3 verwenden, um den Amazon-EC2-Host so zu konfigurieren, dass er die Verbindung von Amazon Redshift akzeptiert.

Um den öffentlichen Schlüssel des Clusters und die IP-Adressen der Cluster-Knoten für Ihren Cluster über die Amazon-Redshift-CLI abzurufen, führen Sie den Befehl describe-clusters aus. Beispielsweise:

```
aws redshift describe-clusters --cluster-identifier <cluster-identifier>
```

Die Antwort enthält einen ClusterPublicKey Wert und die Liste der privaten und öffentlichen IP-Adressen, ähnlich der folgenden:

```
{
    "Clusters": [
        {
            "VpcSecurityGroups": [],
            "ClusterStatus": "available",
            "ClusterNodes": [
                {
```

```
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "LEADER",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-0",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-1",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                }
            ],
            "AutomatedSnapshotRetentionPeriod": 1,
            "PreferredMaintenanceWindow": "wed:05:30-wed:06:00",
            "AvailabilityZone": "us-east-1a",
            "NodeType": "dc2.large",
            "ClusterPublicKey": "ssh-rsa AAAABexamplepublickey...Y3TAl Amazon-
Redshift",
             ...
             ...
}
```

Um den öffentlichen Schlüssel des Clusters und die IP-Adressen der Cluster-Knoten für Ihren Cluster über die Amazon-Redshift-API abzurufen, verwenden Sie die Aktion `DescribeClusters`. Weitere Informationen finden Sie unter describe-clusters im Amazon Redshift CLI Guide oder DescribeClustersim Amazon Redshift API Guide.

Schritt 4: Hinzufügen des öffentlichen Schlüssels des Amazon-Redshift-Clusters zur Datei mit den autorisierten Schlüsseln der einzelnen Amazon-EC2-Hosts

Sie fügen den öffentlichen Schlüssel des Clusters der Datei mit den autorisierten Schlüsseln der einzelnen Hosts für alle Knoten des Amazon-EMR-Clusters hinzu, damit die Hosts Amazon Redshift erkennen und die SSH-Verbindung akzeptieren.

So fügen Sie den öffentlichen Schlüssel des Amazon-Redshift-Clusters zur Datei des Hosts mit den autorisierten Schlüsseln hinzu:

1. Greifen Sie über eine SSH-Verbindung auf den Host zu.

Informationen zum Herstellen von Verbindungen mit Instances über SSH finden Sie unter Connect to Your Instance (Herstellen einer Verbindung mit Ihrer Instance) im Amazon-EC2-Benutzerhandbuch.

2. Kopieren Sie den öffentlichen Amazon-Redshift-Schlüssel aus der Konsole oder aus dem CLI-Antworttext.
3. Kopieren Sie den Inhalt des öffentlichen Schlüssels in die Datei `/home/<ssh_username>/.ssh/authorized_keys` auf dem Host. Kopieren Sie die gesamte Zeichenfolge einschließlich des Präfix `ssh-rsa` und des Suffix `Amazon-Redshift`. Beispiel:

```
ssh-rsa AAAACTP3isxgGzVWoIWpbVvRCOzYdVifMrh… uA70BnMHCaMiRdmvsDOedZDOedZ Amazon-Redshift
```

Schritt 5: Konfigurieren der Hosts, sodass sie alle IP-Adressen des Amazon-Redshift-Clusters akzeptieren

Um eingehenden Datenverkehr für die Host-Instances zuzulassen, bearbeiten Sie die Sicherheitsgruppe und fügen jedem Amazon-Redshift-Clusterknoten eine Inbound-Regel hinzu. Wählen Sie für Type SSH mit TCP-Protokoll auf Port 22 aus. Geben Sie für Source (Quelle) die privaten IP-Adressen der Amazon-Redshift-Cluster-Knoten ein, die Sie in Schritt 3: Abrufen des öffentlichen Schlüssels des Amazon-Redshift-Clusters und der IP-Adressen der Cluster-Knoten abgerufen haben. Informationen zum Hinzufügen von Regeln zu einer Amazon-EC2-Sicherheitsgruppe finden Sie unter Authorizing Inbound Traffic for Your Instances (Autorisieren von eingehendem Datenverkehr für Instances) im Amazon-EC2-Benutzerhandbuch.

Schritt 6: Ausführen des COPY-Befehls, um die Daten zu laden

Führen Sie einen COPY-Befehl aus, um eine Verbindung zum Amazon-EMR-Cluster herzustellen und die Daten in eine Amazon-Redshift-Tabelle zu laden. Der Amazon-EMR-Cluster muss weiter ausgeführt werden, bis der COPY-Befehl abgeschlossen ist. Der Cluster darf beispielsweise nicht für die automatische Beendigung konfiguriert sein.

> **Important**
>
> Wenn eine der Datendateien vor Abschluss von COPY geändert oder gelöscht wird, kann dies zu unerwarteten Ergebnissen führen. Es ist auch möglich, dass die COPY-Operation fehlschlägt.

Geben Sie im COPY-Befehl die ID des Amazon-EMR-Clusters und den Dateipfad und Dateinamen in HDFS an.

```
copy sales
from 'emr://myemrclusterid/myoutput/part*' credentials
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Sie können die Platzhalterzeichen Sternchen (*) und Fragezeichen (?) als Teil des Dateinamenarguments verwenden. Beispielsweise lädt `part*` die Dateien `part-0000`, `part-0001` usw. Wenn Sie nur einen Ordnernamen angeben, versucht COPY, alle Dateien im Ordner zu laden.

> **Important**
>
> Wenn Sie Platzhalterzeichen oder nur den Ordnernamen verwenden, müssen Sie überprüfen, ob unerwünschte Dateien geladen werden. Andernfalls schlägt der COPY-Befehl fehl. Einige Prozesse schreiben beispielsweise eine Protokolldatei in den Ausgabeordner.

Laden von Daten aus Remote-Hosts

Sie können den Befehl COPY verwenden, um Daten parallel Daten aus einem oder mehreren Remote-Hosts wie Amazon-EC2-Instances oder anderen Computern zu laden. COPY stellt über SSH eine Verbindung zu den Remote-Hosts her und führt Befehle auf den Remote-Hosts aus, um Textausgaben zu generieren.

Beim Remote-Host kann es sich um eine Amazon-EC2-Linux-Instance oder einen anderen Unix- oder Linux-Computer handeln, der für die Annahme von SSH-Verbindungen konfiguriert wurde. In diesem Handbuch wird angenommen, dass es sich bei Ihrem Remote-Host um eine Amazon-EC2-Instance handelt. Wenn sich das Verfahren für eine andere Art von Computer unterscheidet, wird im Handbuch auf den Unterschied hingewiesen.

Amazon Redshift kann eine Verbindung zu mehreren Hosts herstellen und für jeden Host mehrere SSH-Verbindungen öffnen. Amazon Redshift sendet über jede Verbindung einen eindeutigen Befehl, um die Textausgabe an die Standardausgabe des Hosts zu generieren. Amazon Redshift liest diese dann wie eine Textdatei.

Bevor Sie beginnen

Folgendes sollte vorhanden sein, bevor Sie den Vorgang starten:

- Ein oder mehrere Host-Computer, beispielsweise Amazon-EC2-Instances, mit denen Sie über SSH eine Verbindung herstellen können.
- Datenquellen auf den Hosts.

  Sie stellen Befehle bereit, die der Amazon-Redshift-Cluster auf den Hosts ausführt, um die Textausgabe zu generieren. Nachdem der Cluster eine Verbindung mit einem Host hergestellt hat, führt der COPY-Befehl die Befehle aus, liest den Text aus der Standardausgabe des Hosts und lädt die Daten parallel in eine Amazon-Redshift-Tabelle. Die Textausgabe muss ein Format aufweisen, das der COPY-Befehl verarbeiten kann. Weitere Informationen finden Sie unter Vorbereiten der Eingabedaten
- Zugriff auf die Hosts von Ihrem Computer aus.

  Im Fall einer Amazon-EC2-Instance verwenden Sie eine SSH-Verbindung, um auf den Host zuzugreifen. Sie müssen auf den Host zugreifen, um den öffentlichen Schlüssel des Amazon-Redshift-Clusters zu der Datei des Hosts mit den autorisierten Schlüsseln hinzuzufügen.
- Ein aktiver Amazon Redshift Cluster.

  Weitere Informationen zum Starten eines Clusters finden Sie im Handbuch Erste Schritte mit Amazon Redshift.

Prozess für das Laden von Daten

In diesem Abschnitt wird das Laden von Daten aus Remote-Hosts beschrieben. In den folgenden Abschnitten finden Sie detaillierte Anweisungen für die einzelnen Schritte.

- Schritt 1: Abrufen des öffentlichen Schlüssels des Clusters und der IP-Adressen der Cluster-Knoten

  Der öffentliche Schlüssel ermöglicht den Knoten des Amazon-Redshift-Clusters die Herstellung von SSH-Verbindungen zu den Remote-Hosts. Sie verwenden die IP-Adresse der einzelnen Cluster-Knoten, um die Host-Sicherheitsgruppen oder die Host-Firewall zu konfigurieren, damit diese den Zugriff von Ihrem Amazon-Redshift-Cluster unter Verwendung dieser IP-Adressen gestattet/gestatten.
- Schritt 2: Hinzufügen des öffentlichen Schlüssels des Amazon-Redshift-Clusters zur Datei des Hosts mit den autorisierten Schlüsseln

  Sie fügen den öffentlichen Schlüssel des Amazon-Redshift-Clusters zur Datei des Hosts hinzu, die die autorisierten Schlüssel enthält, damit der Host den Amazon-Redshift-Cluster erkennt und die SSH-Verbindung akzeptiert.

- Schritt 3: Konfigurieren des Hosts, sodass er alle IP-Adressen des Amazon-Redshift-Clusters akzeptiert

  Ändern Sie bei Amazon EC2 die Sicherheitsgruppen der Instance, indem Sie Eingangsregeln für die Akzeptierung der Amazon-Redshift-IP-Adressen hinzufügen. Im Fall anderer Hosts ändern Sie die Firewall, sodass Ihre Amazon-Redshift-Knoten SSH-Verbindungen zum Remote-Host herstellen können.
- Schritt 4: Abrufen des öffentlichen Schlüssels für den Host

  Sie können optional angeben, dass Amazon Redshift den öffentlichen Schlüssel angeben soll, um den Host zu identifizieren. Sie müssen den öffentlichen Schlüssel suchen und den Text in Ihre Manifestdatei kopieren.
- Schritt 5: Erstellen einer Manifestdatei

  Das Manifest ist eine Textdatei im JSON-Format, das die Details enthält, die Amazon Redshift benötigt, um eine Verbindung zu den Hosts herzustellen und die Daten abzurufen.
- Schritt 6: Hochladen der Manifestdatei in einen Amazon S3 Bucket

  Amazon Redshift liest das Manifest und verwendet diese Informationen, um eine Verbindung zum Remote-Host herzustellen. Wenn sich der Amazon-S3-Bucket nicht in derselben Region wie Ihr Amazon-Redshift-Cluster befindet, müssen Sie die Option REGION verwenden, um die Region anzugeben, in der sich die Daten befinden.
- Schritt 7: Ausführen des COPY-Befehls, um die Daten zu laden

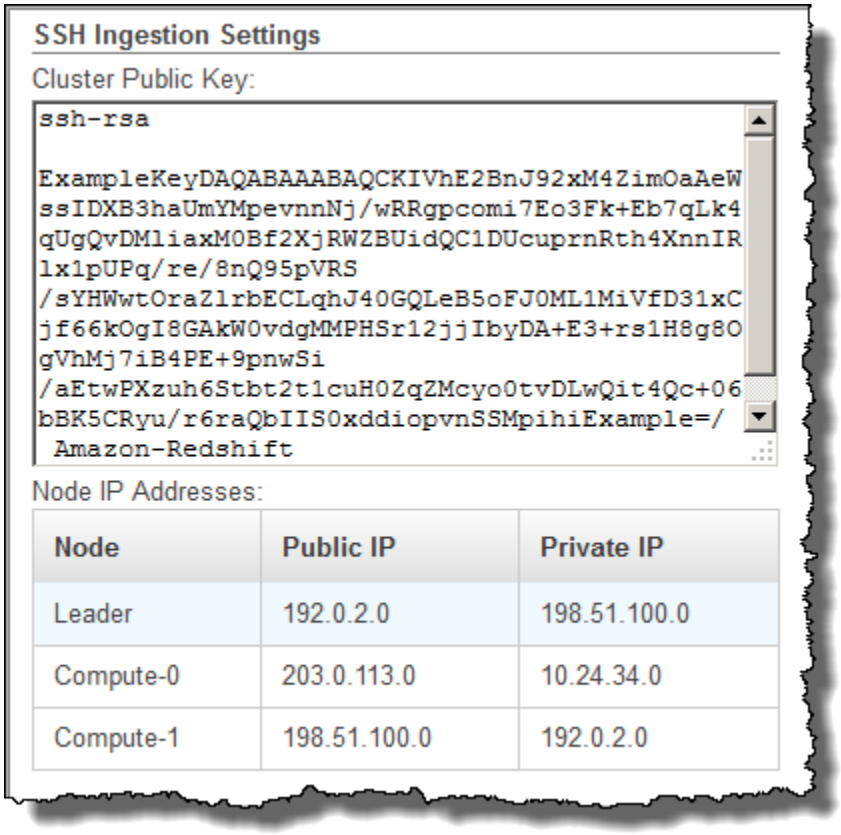
  Führen Sie den COPY-Befehl aus einer Amazon-Redshift-Datenbank aus, um die Daten in eine Amazon-Redshift-Tabelle zu laden.

Schritt 1: Abrufen des öffentlichen Schlüssels des Clusters und der IP-Adressen der Cluster-Knoten

So rufen Sie den öffentlichen Schlüssel des Clusters und die IP-Adressen der Cluster-Knoten für Ihren Cluster über die Konsole ab

1. Greifen Sie auf die Managementkonsole von Amazon Redshift zu.
2. Wählen Sie im Navigationsbereich den Link Clusters (Cluster) aus.
3. Wählen Sie Ihren Cluster aus der Liste aus.
4. Suchen Sie die Gruppe SSH Ingestion Settings (SSH-Eingangseinstellungen).

Notieren Sie sich die Informationen unter Cluster Public Key (Öffentlicher Schlüssel des Clusters) und Node IP addresses (Knoten-IP-Adressen). Diese Informationen werden in späteren Schritten verwendet.



Sie verwenden die IP-Adressen in Schritt 3, um den Host so zu konfigurieren, dass er die Verbindung von Amazon Redshift akzeptiert. Abhängig von der Art des Hosts, zu dem Sie eine Verbindung herzustellen, und davon, ob er sich in einer VPC befindet, verwenden Sie entweder die öffentlichen IP-Adressen oder die privaten IP-Adressen.

Um den öffentlichen Schlüssel des Clusters und die IP-Adressen der Cluster-Knoten für Ihren Cluster über die Amazon-Redshift-CLI abzurufen, führen Sie den Befehl describe-clusters aus.

Beispielsweise:

```
aws redshift describe-clusters --cluster-identifier <cluster-identifier>
```

Die Antwort wird die ClusterPublicKey und die Liste der privaten und öffentlichen IP-Adressen enthalten, ähnlich der folgenden:

```
{
    "Clusters": [
        {
            "VpcSecurityGroups": [],
            "ClusterStatus": "available",
            "ClusterNodes": [
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "LEADER",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-0",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-1",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                }
            ],
            "AutomatedSnapshotRetentionPeriod": 1,
            "PreferredMaintenanceWindow": "wed:05:30-wed:06:00",
            "AvailabilityZone": "us-east-1a",
            "NodeType": "dc2.large",
            "ClusterPublicKey": "ssh-rsa AAAABexamplepublickey...Y3TAl Amazon-
Redshift",
             ...
             ...
}
```

Verwenden Sie die DescribeClusters Aktion, um den öffentlichen Clusterschlüssel und die Clusterknoten-IP-Adressen für Ihren Cluster mithilfe der Amazon Redshift Redshift-API abzurufen. Weitere Informationen finden Sie unter describe-clusters im Amazon Redshift CLI Guide oder DescribeClustersim Amazon Redshift API Guide.

Schritt 2: Hinzufügen des öffentlichen Schlüssels des Amazon-Redshift-Clusters zur Datei des Hosts mit den autorisierten Schlüsseln

Sie fügen den öffentlichen Schlüssel des Clusters zur Datei mit den autorisierten Schlüsseln der einzelnen Hosts hinzu, damit der Host den Amazon-Redshift-Cluster erkennt und die SSH-Verbindung akzeptiert.

So fügen Sie den öffentlichen Schlüssel des Amazon-Redshift-Clusters zur Datei des Hosts mit den autorisierten Schlüsseln hinzu:

1. Greifen Sie über eine SSH-Verbindung auf den Host zu.

   Informationen zum Herstellen von Verbindungen mit Instances über SSH finden Sie unter Connect to Your Instance (Herstellen einer Verbindung mit Ihrer Instance) im Amazon-EC2-Benutzerhandbuch.
2. Kopieren Sie den öffentlichen Amazon-Redshift-Schlüssel aus der Konsole oder aus dem CLI-Antworttext.
3. Kopieren Sie den Inhalt des öffentlichen Schlüssels in die Datei `/home/<ssh_username>/.ssh/authorized_keys` auf dem Remote-Host. Der `<ssh_username>` muss mit dem Wert für das Benutzernamenfeld in der Manifestdatei übereinstimmen. Kopieren Sie die gesamte Zeichenfolge einschließlich des Präfix `ssh-rsa` und des Suffix `Amazon-Redshift`. Beispiel:

   ```
   ssh-rsa AAAACTP3isxgGzVWoIWpbVvRCOzYdVifMrh… uA70BnMHCaMiRdmvsDOedZDOedZ Amazon-Redshift
   ```

Schritt 3: Konfigurieren des Hosts, sodass er alle IP-Adressen des Amazon-Redshift-Clusters akzeptiert

Wenn Sie mit einer Amazon-EC2-Instance oder einem Amazon-EMR-Cluster arbeiten, fügen Sie der Sicherheitsgruppe des Hosts Regeln für eingehenden Datenverkehr hinzu, um Datenverkehr von den einzelnen Amazon-Redshift-Clusterknoten zuzulassen. Wählen Sie für Type SSH mit TCP-Protokoll auf Port 22 aus. Geben Sie für Source die IP-Adressen der Amazon-Redshift-Clusterknoten ein, die Sie in Schritt 1: Abrufen des öffentlichen Schlüssels des Clusters und der IP-Adressen der Cluster-Knoten abgerufen haben. Informationen zum Hinzufügen von Regeln zu einer Amazon-EC2-Sicherheitsgruppe finden Sie unter Authorizing Inbound Traffic for Your Instances (Autorisieren von eingehendem Datenverkehr für Instances) im Amazon-EC2-Benutzerhandbuch.

Verwenden Sie die privaten IP-Adressen in folgenden Fällen:

- Sie haben einen Amazon Redshift Redshift-Cluster, der sich nicht in einer Virtual Private Cloud (VPC) befindet, und eine Amazon EC2 EC2-Classic-Instance, die sich beide in derselben Region befinden. AWS
- Sie haben einen Amazon Redshift Redshift-Cluster, der sich in einer VPC befindet, und eine Amazon EC2 EC2-VPC-Instance, die sich beide in derselben AWS Region und in derselben VPC befinden.

Verwenden Sie andernfalls die öffentlichen IP-Adressen.

Weitere Informationen zur Nutzung von Amazon Redshift finden Sie unter Managing Clusters in Virtual Private Cloud (VPC) (Verwalten von Clustern in Virtual Private Cloud (VPC)) im Amazon-Redshift-Verwaltungshandbuch.

Schritt 4: Abrufen des öffentlichen Schlüssels für den Host

Sie können optional den öffentlichen Schlüssel des Hosts in der Manifestdatei angeben, sodass Amazon Redshift den Host identifizieren kann. Der öffentliche Schlüssel des Hosts ist für den COPY-Befehl nicht erforderlich. Aus Sicherheitsgründen wird jedoch nachdrücklich empfohlen, einen öffentlichen Schlüssel zu verwenden, um „Man-In-The-Middle“-Angriffe verhindern zu helfen.

Sie finden den öffentlichen Schlüssel des Hosts an der folgenden Stelle, wobei `<ssh_host_rsa_key_name>` der eindeutige Name für den öffentlichen Schlüssel des Hosts ist:

```
:  /etc/ssh/<ssh_host_rsa_key_name>.pub
```

> **Note**
>
> Amazon Redshift unterstützt nur RSA-Schlüssel. DSA-Schlüssel werden nicht unterstützt.

Wenn Sie in Schritt 5 Ihre Manifestdatei erstellen, fügen Sie den Text des öffentlichen Schlüssels in das Feld „Public Key“ im Manifestdateieintrag ein.

Schritt 5: Erstellen einer Manifestdatei

Der COPY-Befehl kann über SSH Verbindungen zu mehreren Hosts herstellen und für jeden Host mehrere SSH-Verbindungen erstellen. COPY führt über jede Hostverbindung einen Befehl aus und

lädt anschließend die Ausgabe der Befehle parallel in die Tabelle. Die Manifestdatei ist eine Textdatei im JSON-Format, die Amazon Redshift zum Herstellen der Verbindung zum Host verwendet. Die Manifestdatei gibt die Endpunkte des SSH-Hosts und die Befehle an, die auf den Hosts ausgeführt werden, um Daten an Amazon Redshift zurückzugeben. Optional können Sie den öffentlichen Schlüssel des Hosts, den Anmeldebenutzernamen und ein obligatorisches Flag für die einzelnen Einträge einschließen.

Erstellen Sie die Manifestdatei auf Ihrem lokalen Computer. In einem späteren Schritt laden Sie die Datei zu Amazon S3 hoch.

Die Manifestdatei hat folgendes Format:

```
{
   "entries": [
     {"endpoint":"<ssh_endpoint_or_IP>",
       "command": "<remote_command>",
       "mandatory":true,
       "publickey": "<public_key>",
       "username": "<host_user_name>"},
     {"endpoint":"<ssh_endpoint_or_IP>",
       "command": "<remote_command>",
       "mandatory":true,
       "publickey": "<public_key>",
       "username": "host_user_name"}
    ]
}
```

Die Manifestdatei enthält für jede SSH-Verbindung jeweils ein "entries"-Konstrukt. Jeder Eintrag stellt eine einzelne SSH-Verbindung dar. Es kann mehrere Verbindungen zu einem einzelnen Host oder mehrere Verbindungen zu mehreren Hosts geben. Die doppelten Anführungszeichen sind wie gezeigt erforderlich, sowohl für die Feldnamen als auch die Werte. Der einzige Wert, der keine doppelten Anführungszeichen benötigt, sind die booleschen Werte **`true`** oder **`false`** für das obligatorische Feld.

Nachfolgend werden die Felder in der Manifestdatei beschrieben.

endpoint

Die URL- oder IP-Adresse des Hosts. Beispiele sind `ec2-111-222-333.compute-1.amazonaws.com` oder `22.33.44.56`.

command

Der Befehl, der durch den Host ausgeführt wird, um eine Textausgabe oder eine Binärausgabe (gzip, lzop oder bzip2) zu generieren. Bei diesem Befehl kann es sich um jeden Befehl handeln, zu dessen Ausführung der Benutzer host_user_name berechtigt ist. Beim Befehl kann es sich einfach um einen Befehl zum Drucken einer Datei, um eine Abfrage einer Datenbank oder um das Starten eines Skripts handeln. Die Ausgabe (Textdatei, binäre gzip-Datei, binäre lzop-Datei oder binäre bzip2-Datei) muss ein Format aufweisen, das der Amazon-Redshift-Befehl COPY verarbeiten kann. Weitere Informationen finden Sie unter Vorbereiten der Eingabedaten.

publickey

(Optional) Der öffentliche Schlüssel des Hosts. Wenn angegeben, verwendet Amazon Redshift den öffentlichen Schlüssel, um den Host zu identifizieren. Wenn der öffentliche Schlüssel nicht angegeben ist, versucht Amazon Redshift nicht, den Host zu identifizieren. Wenn beispielsweise der öffentliche Schlüssel des Remote-Hosts `ssh-rsa AbcCbaxxx…xxxDHKJ root@amazon.com` ist, geben Sie den folgenden Text in das Feld für den öffentlichen Schlüssel ein: `AbcCbaxxx…xxxDHKJ`.

mandatory

(Optional) Zeigt an, ob der COPY-Befehl fehlschlagen soll, wenn keine Verbindung hergestellt wird. Der Standardwert ist `false`. Wenn Amazon Redshift nicht mindestens eine Verbindung herstellt, schlägt der COPY-Befehl fehl.

username

(Optional) Der Benutzername, der für die Anmeldung am Hostsystem und die Ausführung des Remotebefehls verwendet wird. Der Benutzeranmeldename muss mit dem Anmeldenamen identisch sein, der zum Hinzufügen des öffentlichen Schlüssels zur Datei des Hosts mit den autorisierten Schlüsseln in Schritt 2 verwendet wurde. Der Standardbenutzername ist „redshift“.

Im folgenden Beispiel wird gezeigt, wie ein abgeschlossenes Manifest vier Verbindungen mit demselben Host herstellt und über jede Verbindung einen anderen Befehl ausführt:

```
{
  "entries": [
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata1.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
```

```
           "username": "ec2-user"},
        {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
           "command": "cat loaddata2.txt",
           "mandatory":true,
           "publickey": "ec2publickeyportionoftheec2keypair",
           "username": "ec2-user"},
        {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
           "command": "cat loaddata3.txt",
           "mandatory":true,
           "publickey": "ec2publickeyportionoftheec2keypair",
           "username": "ec2-user"},
        {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
           "command": "cat loaddata4.txt",
           "mandatory":true,
           "publickey": "ec2publickeyportionoftheec2keypair",
           "username": "ec2-user"}
     ]
}
```

Schritt 6: Hochladen der Manifestdatei in einen Amazon S3 Bucket

Laden Sie die Manifestdatei zu einem Amazon S3 Bucket hoch. Wenn sich der Amazon S3 S3-Bucket nicht in derselben AWS Region wie Ihr Amazon Redshift Redshift-Cluster befindet, müssen Sie die REGION Option verwenden, um die AWS Region anzugeben, in der sich das Manifest befindet. Informationen zum Erstellen eines Amazon S3 Buckets und zum Hochladen einer Datei finden Sie im Benutzerhandbuch zu Amazon Simple Storage Service.

Schritt 7: Ausführen des COPY-Befehls, um die Daten zu laden

Führen Sie einen COPY-Befehl aus, um eine Verbindung zum Host herzustellen und die Daten in eine Amazon-Redshift-Tabelle zu laden. Geben Sie im COPY-Befehl den expliziten Amazon-S3-Objektpfad für die Manifestdatei an und schließen Sie die SSH-Option ein. Beispiel,

```
copy sales
from 's3://mybucket/ssh_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|'
ssh;
```

> **Note**
>
> Wenn Sie die automatische Kompression verwenden, führt der COPY-Befehl zwei Datenlesevorgänge aus. Dies bedeutet, dass der Remotebefehl zweimal ausgeführt wird. Der erste Lesevorgang dient dazu, eine Stichprobe zur Kompressionsanalyse bereitzustellen. Im zweiten Lesevorgang werden die Dateien tatsächlich geladen. Wenn die zweimalige Ausführung des Remotebefehls aufgrund möglicher Nebeneffekte Probleme verursachen könnte, sollten Sie die automatische Kompression deaktivieren. Um die automatische Kompression zu deaktivieren, führen Sie den COPY-Befehl aus, wobei Sie die Option COMPUPDATE auf OFF setzen. Weitere Informationen finden Sie unter Laden von Tabellen mit automatischer Kompression.

Laden von Daten aus einer Amazon-DynamoDB-Tabelle

Sie können den COPY-Befehl verwenden, um eine Tabelle mit Daten aus einer einzelnen Amazon-DynamoDB-Tabelle zu laden.

> **Important**
>
> Die Amazon DynamoDB-Tabelle, die die Daten bereitstellt, muss in derselben AWS Region wie Ihr Cluster erstellt werden, es sei denn, Sie verwenden die REGION Option, um die AWS Region anzugeben, in der sich die Amazon DynamoDB-Tabelle befindet.

Der COPY-Befehl nutzt die Architektur für die massiv parallele Verarbeitung (Massively Parallel Processing, MPP) von Amazon Redshift, um Daten parallel aus einer Amazon-DynamoDB-Tabelle zu lesen und zu laden. Sie können die Parallelverarbeitung maximal nutzen, indem Sie für Ihre Amazon-Redshift-Tabellen Verteilungsstile festlegen. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

> **Important**
>
> Wenn der COPY-Befehl Daten aus der Amazon-DynamoDB-Tabelle liest, ist die entsprechende Datenübertragung Teil des bereitgestellten Durchsatzes dieser Tabelle.

Um die übermäßige Nutzung des bereitgestellten Lesedurchsatzes zu vermeiden, wird empfohlen, keine Daten aus Amazon-DynamoDB-Tabellen zu laden, die sich in Produktionsumgebungen befinden. Wenn Sie Daten aus Produktionstabellen laden, wird empfohlen, die Option READRATIO auf einen sehr viel niedrigeren Wert als den durchschnittlichen Prozentsatz ungenutzten bereitgestellten Durchsatzes festzulegen. Eine niedrige READRATIO-Einstellung hilft, Ablehnungsprobleme zu minimieren. Um den gesamten bereitgestellten Durchsatz einer Amazon-DynamoDB-Tabelle zu nutzen, legen Sie READRATIO auf 100 fest.

Der COPY-Befehl gleicht mittels folgender Regeln Attributnamen in den Elementen, die aus der DynamoDB-Tabelle abgerufen wurden, mit Spaltennamen in einer vorhandenen Amazon-Redshift-Tabelle ab:

- Amazon-Redshift-Tabellenspalten werden ohne Berücksichtigung von Groß- und Kleinschreibung mit Amazon-DynamoDB-Elementattributen abgeglichen. Wenn ein Element in der DynamoDB-Tabelle mehrere Attribute enthält, die sich nur nach Groß- und Kleinschreibung unterscheiden, wie „price“ und „PRICE“, schlägt der COPY-Befehl fehl.
- Amazon-Redshift-Tabellenspalten, die mit keinem Attribut in der Amazon-DynamoDB-Tabelle übereinstimmen, werden entweder als NULL oder leer geladen, abhängig von dem Wert, der mit der Option EMPTYASNULL im Befehl COPY angegeben ist.
- Amazon-DynamoDB-Attribute, die mit keiner Spalte in der Amazon-Redshift-Tabelle übereinstimmen, werden verworfen. Attribute werden gelesen, bevor sie abgeglichen werden. Daher verbrauchen auch verworfene Attribute einen Teil des bereitgestellten Durchsatzes dieser Tabelle.
- Es werden nur Amazon-DynamoDB-Attribute mit skalaren STRING- und NUMBER-Datentypen unterstützt. Die Amazon-DynamoDB-Datentypen BINARY und SET werden nicht unterstützt. Wenn ein COPY-Befehl versucht, ein Attribut zu laden, dessen Datentyp nicht unterstützt wird, schlägt der Befehl fehl. Wenn das Attribut mit keiner Amazon-Redshift-Tabellenspalte übereinstimmt, versucht COPY nicht, es zu laden, und es tritt kein Fehler auf.

Der COPY-Befehl verwendet die folgende Syntax, um Daten aus einer Amazon-DynamoDB-Tabelle zu laden:

```
copy <redshift_tablename> from 'dynamodb://<dynamodb_table_name>'
authorization
readratio '<integer>';
```

Die Werte für die Autorisierung sind die AWS Anmeldeinformationen, die für den Zugriff auf die Amazon DynamoDB-Tabelle benötigt werden. Wenn diese Anmeldeinformationen einem Benutzer entsprechen, muss dieser Benutzer SCAN- und DESCRIBE-Berechtigungen für die Amazon-DynamoDB-Tabelle besitzen, die geladen wird.

Die Werte für die Autorisierung geben die AWS Autorisierung an, die Ihr Cluster für den Zugriff auf die Amazon DynamoDB-Tabelle benötigt. Die Berechtigung muss SCAN und DESCRIBE für die Amazon-DynamoDB-Tabelle einschließen, die geladen wird. Weitere Informationen zu erforderlichen Berechtigungen finden Sie unter IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY. Die bevorzugte Methode für die Authentifizierung besteht in der Angabe des Parameters IAM_ROLE und des Amazon-Ressourcennamens (ARN) für eine IAM-Rolle mit den notwendigen Berechtigungen. Weitere Informationen finden Sie unter Rollenbasierte Zugriffskontrolle.

Um die Authentifizierung mittels des Parameters IAM_ROLE durchzuführen, ersetzen Sie *<aws-account-id>* und *<role-name>* wie in der folgenden Syntax gezeigt.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Das folgende Beispiel zeigt die Authentifizierung unter Verwendung einer IAM-Rolle.

```
copy favoritemovies
from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Weitere Informationen zu anderen Autorisierungsoptionen finden Sie unter . Autorisierungsparameter

Wenn Sie Ihre Daten validieren möchten, ohne die Tabelle tatsächlich zu laden, verwenden Sie die Option NOLOAD mit dem Befehl COPY.

Im folgenden Beispiel wird die FAVORITEMOVIES-Tabelle mit Daten aus der DynamoDB-Tabelle geladen. my-favorite-movies-table Die Leseaktivität kann bis zu 50 % des bereitgestellten Durchsatzes verbrauchen.

```
copy favoritemovies from 'dynamodb://my-favorite-movies-table'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Um den Durchsatz zu maximieren, lädt der COPY-Befehl Daten aus einer Amazon-DynamoDB-Tabelle aus allen Datenverarbeitungsknoten im Cluster parallel.

Bereitgestellter Durchsatz mit automatischer Kompression

Standardmäßig wendet der COPY-Befehl eine automatische Kompression an, wenn Sie eine leere Tabelle ohne Kompressionskodierung angeben. Die automatische Komprimierungsanalyse untersucht zunächst stichprobenartig eine große Zahl von Zeilen aus der Amazon-DynamoDB-Tabelle. Die Größe der Stichprobe basiert auf dem Wert des Parameters COMPROWS. Der Standardwert ist 100.000 Zeilen pro Slice.

Nach der Untersuchung der Stichprobe werden die Stichprobenzeilen verworfen und die gesamte Tabelle wird geladen. Daher werden zahlreiche Zeilen zweimal gelesen. Weitere Informationen zur Funktionsweise der automatischen Kompression finden Sie unter Laden von Tabellen mit automatischer Kompression.

> **Important**
>
> Wenn der COPY-Befehl Daten aus der Amazon-DynamoDB-Tabelle liest, einschließlich der Stichprobenzeilen, ist die entsprechende Datenübertragung Teil des bereitgestellten Durchsatzes dieser Tabelle.

Laden von Multibyte-Daten aus Amazon DynamoDB

Wenn Ihre Daten Multibyte-Zeichen enthalten, die andere als ASCII-Zeichen verwenden (beispielsweise chinesische oder kyrillische Zeichen), müssen Sie die Daten in VARCHAR-Spalten laden. Der VARCHAR-Datentyp unterstützt UTF-8-Zeichen mit vier Bytes. Der CHAR-Datentyp unterstützt jedoch nur ASCII-Zeichen mit einem Byte. Sie können keine Zeichen mit fünf Bytes oder mehr in Amazon-Redshift-Tabellen laden. Weitere Informationen zu CHAR und VARCHAR finden Sie unter Datentypen.

Überprüfung, ob die Daten korrekt geladen wurden

Fragen Sie nach Abschluss der Ladeoperation die Systemtabelle STL_LOAD_COMMITS ab, um sicherzustellen, dass die erwarteten Dateien geladen wurden. Führen Sie den COPY-Befehl und die Verifizierung des Ladevorgangs innerhalb derselben Transaktion aus. Wenn es Probleme mit dem Ladevorgang gibt, können Sie so ein Rollback für die gesamte Transaktion ausführen.

Die folgende Abfrage gibt Einträge für das Laden der Tabellen in der Datenbank TICKIT zurück:

```
select query, trim(filename) as filename, curtime, status
```

```
from stl_load_commits
where filename like '%tickit%' order by query;

 query |         filename          |          curtime           | status
-------+---------------------------+----------------------------+--------
 22475 | tickit/allusers_pipe.txt  | 2013-02-08 20:58:23.274186 |      1
 22478 | tickit/venue_pipe.txt     | 2013-02-08 20:58:25.070604 |      1
 22480 | tickit/category_pipe.txt  | 2013-02-08 20:58:27.333472 |      1
 22482 | tickit/date2008_pipe.txt  | 2013-02-08 20:58:28.608305 |      1
 22485 | tickit/allevents_pipe.txt | 2013-02-08 20:58:29.99489  |      1
 22487 | tickit/listings_pipe.txt  | 2013-02-08 20:58:37.632939 |      1
 22489 | tickit/sales_tab.txt      | 2013-02-08 20:58:37.632939 |      1
(6 rows)
```

Validieren von Eingabedaten

Um die Daten in den Amazon-S3-Eingabedateien oder in der Amazon-DynamoDB-Tabelle zu validieren, bevor sie tatsächlich geladen werden, verwenden Sie die Option NOLOAD mit dem Befehl COPY. Verwenden Sie NOLOAD mit den COPY-Befehlen und Optionen, die Sie auch zum Laden der Daten verwenden. NOLOAD prüft die Integrität aller Daten, ohne sie in die Datenbank zu laden. Die NOLOAD-Option zeigt alle Fehler an, die beim Versuch, die Daten zu laden, auftreten.

Wenn Sie beispielsweise einen falschen Amazon-S3-Pfad für die Eingabedatei angegeben haben, zeigt Amazon Redshift den folgenden Fehler an:

```
ERROR:  No such file or directory
DETAIL:
-----------------------------------------------
Amazon Redshift error:  The specified key does not exist
code:      2
context:   S3 key being read :
location:  step_scan.cpp:1883
process:   xenmaster [pid=22199]
-----------------------------------------------
```

Informationen zur Fehlerbehebung bei Fehlermeldungen finden Sie unter Ladefehlerreferenz.

Ein Beispiel mit der Option NOLOAD finden Sie unter COPY-Befehl mit der Option NOLOAD.

Laden von Tabellen mit automatischer Kompression

Themen

Sie können Komprimierungskodierungen manuell auf Spalten in Tabellen anwenden, basierend auf Ihrer eigenen Auswertung der Daten. Sie können auch den Befehl COPY verwenden, bei dem COMPUPDATE auf ON gesetzt ist, um die Komprimierung automatisch basierend auf Beispieldaten zu analysieren und anzuwenden.

Sie können die automatische Kompression beim Erstellen und Laden einer völlig neuen Tabelle verwenden. Mit dem Befehl COPY führen Sie eine Komprimierungsanalyse durch. Sie können eine Komprimierungsanalyse auch durchführen, ohne Daten zu laden oder die Komprimierung einer Tabelle zu ändern, indem Sie den Befehl ANALYZE COMPRESSION für eine bereits ausgefüllte Tabelle ausführen. Beispielsweise können Sie ANALYZE COMPRESSION ausführen, wenn Sie die Komprimierung für eine Tabelle für die zukünftige Verwendung analysieren und gleichzeitig die vorhandenen DDL-Anweisungen (Data Definition Language) beibehalten möchten.

Die automatische Kompression sorgt für den Ausgleich der allgemeinen Leistung bei der Auswahl von Kompressionskodierungen. Bereichseingeschränkte Scans können eine geringe Leistung zeigen, wenn Sortierschlüsselspalten stärker komprimiert sind als andere Spalten in der selben Abfrage. Daher überspringt die automatische Komprimierung die Datenanalysephase in den Sortierschlüsselspalten und behält die benutzerdefinierten Kodierungstypen bei.

Bei der automatischen Komprimierung wird die RAW-Kodierung ausgewählt, wenn Sie nicht explizit eine Art der Kodierung definiert haben. ANALYZE COMPRESSION verhält sich gleich. Um eine optimale Abfrageleistung zu erzielen, sollten Sie RAW für Sortierschlüssel verwenden.

So funktioniert die automatische Kompression

Wenn der COMPUPDATE-Parameter auf ON eingestellt ist, führt der COPY-Befehl jedes Mal eine automatische Kompression aus, wenn Sie den COPY-Befehl mit einer leeren Zieltabelle ausführen und alle Tabellenspalten entweder eine RAW-Kodierung oder keine Kodierung aufweisen.

Um die automatische Kompression auf eine leere Tabelle unabhängig von ihren aktuellen Kompressionskodierungen anzuwenden, führen Sie den COPY-Befehl aus, wobei Sie die Option COMPUPDATE auf ON festlegen. Um die automatische Kompression zu deaktivieren, führen Sie den COPY-Befehl aus, wobei Sie die Option COMPUPDATE auf OFF setzen.

Sie können keine automatische Kompression auf eine Tabelle anwenden, die bereits Daten enthält.

> **Note**
>
> Um die automatische Kompression analysieren zu können, müssen in den geladenen Daten genügend Zeilen vorhanden sein (mindestens 100.000 Zeilen pro Slice), um eine relevante Stichprobe darzustellen.

Die automatische Kompression führt im Hintergrund die folgenden Operationen als Teil der Ladetransaktion aus:

1. Aus der Eingabedatei wird eine anfängliche Stichprobe von Zeilen geladen. Die Größe der Stichprobe basiert auf dem Wert des Parameters COMPROWS. Der Standardwert ist 100,000.
2. Für jede Spalte werden Kompressionsoptionen ausgewählt.
3. Die Stichprobenzeilen werden aus der Tabelle entfernt.
4. Die Tabelle wird erneut mit den gewählten Kompressionskodierungen erstellt.
5. Die gesamte Eingabedatei wird geladen und unter Verwendung der neuen Kodierungen komprimiert.

Nach der Ausführung des COPY-Befehls wird die Tabelle vollständig geladen und komprimiert und kann verwendet werden. Wenn Sie später weitere Daten laden, werden die angefügten Zeilen entsprechend der vorhandenen Kodierung komprimiert.

Wenn Sie nur eine Kompressionsanalyse ausführen möchten, führen Sie ANALYZE COMPRESSION aus. Dies ist effizienter als eine vollständige COPY-Operation. Anschließend können Sie die Ergebnisse auswerten, um zu entscheiden, ob Sie die automatische Kompression verwenden oder die Tabelle manuell neu erstellen sollten.

Die automatische Kompression wird nur für den COPY-Befehl unterstützt. Alternativ können Sie beim Erstellen der Tabelle manuell eine Kompressionskodierung anwenden. Informationen zur manuellen Kompressionskodierung finden Sie unter Arbeiten mit Spaltenkomprimierung.

Beispiel für eine automatische Kompression

In diesem Beispiel wird angenommen, dass die Datenbank TICKIT eine Kopie der LISTING-Tabelle mit dem Namen BIGLIST enthält und Sie auf diese Tabelle eine automatische Kompression anwenden möchten, wenn sie mit ungefähr 3 Millionen Zeilen geladen wird.

So laden Sie die Tabelle und komprimieren sie automatisch

1. Stellen Sie sicher, dass die Tabelle leer ist. Sie können die automatische Kompression nur auf leere Tabellen anwenden:

```
truncate biglist;
```

2. Laden Sie die Tabelle mit einem einzelnen COPY-Befehl. Auch wenn die Tabelle leer ist, wurde möglicherweise früher bereits eine Kodierung angegeben. Um eine Kompressionsanalyse durch Amazon Redshift zu ermöglichen, setzen Sie den Parameter COMPUPDATE auf ON.

```
copy biglist from 's3://mybucket/biglist.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' COMPUPDATE ON;
```

   Da die Option COMPROWS nicht angegeben ist, wird die standardmäßige und empfohlene Größe von 100.000 Zeilen pro Slice verwendet.

3. Betrachten Sie das neue Schema für die Tabelle BIGLIST, um die automatisch gewählten Kodierungsschemas zu überprüfen.

```
select "column", type, encoding
from pg_table_def where tablename = 'biglist';

     Column     |            Type             | Encoding
----------------+-----------------------------+----------
 listid         | integer                     | az64
 sellerid       | integer                     | az64
 eventid        | integer                     | az64
 dateid         | smallint                    | none
 numtickets     | smallint                    | az64
 priceperticket | numeric(8,2)                | az64
 totalprice     | numeric(8,2)                | az64
 listtime       | timestamp without time zone | az64
```

4. Überprüfen Sie, ob die erwartete Zahl von Zeilen geladen wurde:

```
select count(*) from biglist;

count
---------
3079952
```

```
(1 row)
```

Wenn dieser Tabelle später mittels der Anweisungen COPY oder INSERT Zeilen angefügt werden, werden die gleichen Kompressionskodierungen angewendet.

Optimieren des Speichers für enge Tabellen

Wenn Sie eine Tabelle mit sehr wenigen Spalten, jedoch einer sehr großen Zahl von Zeilen haben, nehmen die drei ausgeblendeten Metadatenidentitätsspalten (INSERT_XID, DELETE_XID, ROW_ID) einen unverhältnismäßig großen Anteil am Festplattenplatz für die Tabelle in Anspruch.

Um die Kompression der ausgeblendeten Spalten zu optimieren, laden Sie die Tabelle in einer einzigen COPY-Transaktion, wenn möglich. Wenn Sie die Tabelle mit mehreren getrennten COPY-Befehlen laden, wird die Spalte INSERT_XID nicht gut komprimiert. Sie müssen eine Bereinigungsoperation ausführen, wenn Sie mehrere COPY-Befehle verwenden. Dies verbessert die Kompression von INSERT_XID jedoch nicht.

Laden von Standardspaltenwerten

Sie können in Ihrem COPY-Befehl optional eine Spaltenliste definieren. Wenn eine Spalte in der Tabelle aus der Spaltenliste ausgelassen wird, lädt COPY die Spalte entweder mit dem Wert, der von der im Befehl CREATE TABLE angegebenen Option DEFAULT bereitgestellt wird, oder mit NULL, wenn die Option DEFAULT nicht angegeben wurde.

Wenn COPY versucht, einer Spalte NULL zuzuweisen, die als NOT NULL definiert ist, schlägt der COPY-Befehl fehl. Weitere Informationen zum Zuweisen der Option DEFAULT finden Sie unter CREATE TABLE.

Wenn Sie Daten aus Datendateien in Amazon S3 laden, müssen sich die Spalten in der Spaltenliste in derselben Reihenfolge wie die Felder in der Datendatei befinden. Wenn es für ein Feld in der Datendatei keine entsprechende Spalte in der Spaltenliste gibt, schlägt der COPY-Befehl fehl.

Beim Laden aus einer Amazon-DynamoDB-Tabelle spielt die Reihenfolge keine Rolle. Felder in den Amazon-DynamoDB-Attributen, die mit keiner Spalte in der Amazon-Redshift-Tabelle übereinstimmen, werden verworfen.

Bei der Verwendung des COPY-Befehls zum Laden von DEFAULT-Werten in eine Tabelle gelten die folgenden Einschränkungen:

- Wenn eine IDENTITY-Spalte in der Spaltenliste enthalten ist, muss auch die Option EXPLICIT_IDS im Befehl COPY angegeben werden. Andernfalls schlägt der COPY-Befehl fehl. Wenn eine IDENTITY-Spalte in der Spaltenliste ausgelassen wird und die Option EXPLICIT_IDS angegeben ist, schlägt der COPY-Befehl ebenfalls fehl.
- Da der ausgewertete DEFAULT-Ausdruck für eine bestimmte Spalte für alle geladenen Zeilen derselbe ist, weist ein DEFAULT-Ausdruck, der eine RANDOM()-Funktion verwendet, allen Zeilen denselben Wert zu.
- DEFAULT-Ausdrücke, die CURRENT_DATE oder SYSDATE enthalten, sind auf den Zeitstempel der aktuellen Transaktion festgelegt.

Ein Beispiel hierfür finden Sie unter „Laden von Daten aus einer Datei mit Standardwerten“ in Beispiele für COPY.

Fehlerbehebung bei Datenladevorgängen

Themen

- ServiceException S3-Fehler
- Systemtabellen für die Behebung von Fehlern beim Laden von Daten
- Fehler beim Laden von Multibyte-Zeichen
- Ladefehlerreferenz

In diesem Abschnitt finden Sie Informationen zum Identifizieren und Beheben von Fehlern beim Laden von Daten.

ServiceException S3-Fehler

Die häufigsten ServiceException S3-Fehler werden durch eine falsch formatierte oder falsche Zeichenfolge für Anmeldeinformationen verursacht, wenn sich Ihr Cluster und Ihr Bucket in verschiedenen AWS Regionen befinden und die Amazon S3 S3-Berechtigungen nicht ausreichen.

In diesem Abschnitt werden Informationen zur Fehlerbehebung für die einzelnen Fehlertypen bereitgestellt.

Ungültige Anmeldeinformationszeichenfolgen

Wenn die Anmeldeinformationszeichenfolge nicht korrekt formatiert ist, erhalten Sie die folgende Fehlermeldung:

```
ERROR: Invalid credentials. Must be of the format: credentials
'aws_access_key_id=<access-key-id>;aws_secret_access_key=<secret-access-key>
[;token=<temporary-session-token>]'
```

Überprüfen Sie, ob die Anmeldeinformationszeichenfolge keine Leerzeichen oder Zeilenumbrüche enthält und in einfachen Anführungszeichen eingeschlossen ist.

Ungültige Zugriffsschlüssel-ID

Wenn Ihre Zugriffsschlüssel-ID nicht vorhanden ist, erhalten Sie folgende Fehlermeldung:

```
[Amazon](500310) Invalid operation: S3ServiceException:The AWS Access Key Id you
 provided does not exist in our records.
```

Dies geht häufig auf einen Fehler beim Kopieren und Einfügen zurück. Überprüfen Sie, ob die Zugriffsschlüssel-ID korrekt eingegeben wurde. Wenn Sie temporäre Sitzungsschlüssel verwenden, überprüfen Sie außerdem, ob der Wert für `token` festgelegt wurde.

Ungültiger geheimer Zugriffsschlüssel

Wenn der geheime Zugriffsschlüssel falsch ist, erhalten Sie folgende Fehlermeldung:

```
[Amazon](500310) Invalid operation: S3ServiceException:The request signature we
 calculated does not match the signature you provided.
Check your key and signing method.,Status 403,Error SignatureDoesNotMatch
```

Dies geht häufig auf einen Fehler beim Kopieren und Einfügen zurück. Überprüfen Sie, ob der geheime Zugriffsschlüssel korrekt eingegeben wurde und ob es sich um den korrekten Schlüssel für die Zugriffsschlüssel-ID handelt.

Bucket befindet sich in einer anderen Region

Der im COPY-Befehl angegebene Amazon S3 S3-Bucket muss sich in derselben AWS Region wie der Cluster befinden. Wenn sich Amazon-S3-Bucket und -Cluster in verschiedenen Regionen befinden, erhalten Sie eine Fehlermeldung wie die folgende:

```
ERROR: S3ServiceException:The bucket you are attempting to access must be addressed
 using the specified endpoint.
```

Sie können einen Amazon-S3-Bucket in einer bestimmten Region erstellen, indem Sie entweder bei der Erstellung des Buckets die Region über die Amazon-S3-Managementkonsole auswählen oder

indem Sie einen Endpunkt angeben, wenn Sie den Bucket mithilfe der Amazon-S3-API oder -CLI erstellen. Weitere Informationen finden Sie unter Hochladen von Dateien in Amazon S3.

Weitere Informationen zu Amazon-S3-Regionen finden Sie unter Zugriff auf einen Bucket im Benutzerhandbuch für Amazon Simple Storage Service.

Alternativ können Sie die Region mithilfe der Option REGION mit dem Befehl COPY angeben.

Zugriff verweigert

Wenn der Benutzer keine ausreichenden Berechtigungen besitzt, erhalten Sie folgende Fehlermeldung:

```
ERROR: S3ServiceException:Access Denied,Status 403,Error AccessDenied
```

Eine mögliche Ursache dafür ist, dass der durch die Anmeldeinformationen identifizierte Benutzer nicht über LIST- und GET-Zugriff für den Amazon-S3-Bucket verfügt. Weitere mögliche Ursachen finden Sie unter Beheben von Fehlern aufgrund einer Zugriffsverweigerung (403 Forbidden) in Amazon S3 im Benutzerhandbuch zu Amazon Simple Storage Service.

Informationen zur Verwaltung des Benutzerzugriffs auf Buckets finden Sie unter Identity and Access Management in Amazon S3 im Benutzerhandbuch zu Amazon Simple Storage Service.

Systemtabellen für die Behebung von Fehlern beim Laden von Daten

Die folgenden Amazon-Redshift-Systemtabellen können beim Beheben von Fehlern, die beim Laden von Daten auftreten, nützlich sein:

- Führen Sie eine Abfrage für STL_LOAD_ERRORS aus, um Fehler zu entdecken, die während spezifischer Ladevorgänge aufgetreten sind.
- Führen Sie eine Abfrage für STL_FILE_SCAN aus, um die Ladezeiten für spezifische Dateien anzuzeigen oder um festzustellen, ob eine spezifische Datei überhaupt gelesen wurde.
- Führen Sie eine Abfrage für STL_S3CLIENT_ERROR aus, um Details zu Fehlern zu ermitteln, die während der Übertragung von Daten aus Amazon S3 aufgetreten sind.

So finden und diagnostizieren Sie Ladefehler

1. Erstellen Sie eine Ansicht oder definieren Sie eine Abfrage, die Details zu Ladefehlern zurückgibt. Im folgenden Beispiel wird ein Join der Tabellen STL_LOAD_ERRORS und

STV_TBL_PERM ausgeführt, um die Tabellen-IDs mit den tatsächlichen Tabellennamen abzugleichen.

```
create view loadview as
(select distinct tbl, trim(name) as table_name, query, starttime,
trim(filename) as input, line_number, colname, err_code,
trim(err_reason) as reason
from stl_load_errors sl, stv_tbl_perm sp
where sl.tbl = sp.id);
```

2. Legen Sie die Option MAXERRORS in Ihrem COPY-Befehl auf einen Wert fest, der groß genug ist, dass COPY nützliche Informationen zu Ihren Daten zurückgeben kann. Wenn für den COPY-Befehl Fehler auftreten, weist Sie eine Fehlermeldung an, in der Tabelle STL_LOAD_ERRORS nach Details zu suchen.
3. Führen Sie eine Abfrage für die Ansicht LOADVIEW aus, um Fehlerdetails anzuzeigen. Beispiel:

```
select * from loadview where table_name='venue';
```

```
  tbl   | table_name | query |         starttime
--------+------------+-------+---------------------------
 100551 | venue      | 20974 | 2013-01-29 19:05:58.365391

|     input      | line_number | colname | err_code |       reason
+----------------+-------------+-------+----------+--------------------
| venue_pipe.txt |           1 |     0 |     1214 | Delimiter not found
```

4. Beheben Sie das Problem in der Eingabedatei oder im Ladeskript basierend auf den Informationen, die von der Ansicht zurückgegeben werden. Typische Ladefehler, auf die Sie achten sollten, sind:

   - Fehlende Übereinstimmung zwischen Datentypen in der Tabelle und Werten in den Eingabedatenfeldern.
   - Fehlende Übereinstimmung zwischen der Anzahl der Spalten in der Tabelle und der Anzahl der Felder in den Eingabedatenfeldern.
   - Nicht übereinstimmende Anführungszeichen. Amazon Redshift unterstützt sowohl einfache als auch doppelte Anführungszeichen. Diese Anführungszeichen müssen jedoch entsprechend ausgeglichen werden.
   - Falsches Format für Datum-/Uhrzeitdaten in Eingabedateien.

- ut-of-range O-Werte in Eingabedateien (für numerische Spalten).
- Anzahl der unterschiedlichen Werte für eine Spalte überschreitet die Begrenzung für ihre Kompressionskodierung.

Fehler beim Laden von Multibyte-Zeichen

Spalten mit einem CHAR-Datentyp akzeptieren nur UTF-8-Einzelbyte-Zeichen bis zum Bytewert 127 oder 7F Hex. Dies entspricht dem ASCII-Zeichensatz. VARCHAR-Spalten akzeptieren UTF-8-Multibyte-Zeichen bis zu einer Länge von vier Bytes. Weitere Informationen finden Sie unter Zeichentypen.

Wenn eine Zeile in den geladenen Daten ein für den Datentyp der Spalte ungültiges Zeichen enthält, gibt COPY einen Fehler zurück und protokolliert eine Zeile mit der Fehlernummer 1220 in der Systemprotokolltabelle STL_LOAD_ERRORS. Das Feld ERR_REASON enthält die Byte-Folge in Hex für das ungültige Zeichen.

Alternativ zur Korrektur von ungültigen Zeichen in den geladenen Daten können Sie die ungültigen Zeichen auch während des Ladeprozesses ersetzen. Um ungültige UTF-8-Zeichen zu ersetzen, geben Sie mit dem COPY-Befehl die Option ACCEPTINVCHARS an. Wenn die Option ACCEPTINVCHARS festgelegt ist, ersetzt das von Ihnen angegebene Zeichen den Codepunkt. Wenn die Option ACCEPTINVCHARS nicht festgelegt ist, akzeptiert Amazon Redshift die Zeichen als gültiges UTF-8. Weitere Informationen finden Sie unter ACCEPTINVCHARS.

Die folgende Liste von Codepunkten ist gültiges UTF-8, COPY-Operationen geben keinen Fehler zurück, wenn die ACCEPTINVCHARS-Option nicht festgelegt wurde. Diese Codepunkte sind jedoch keine gültigen Zeichen. Sie können die Option ACCEPTINVCHARS verwenden, um einen Codepunkt durch ein von Ihnen festgelegtes Zeichen zu ersetzen. Diese Codepunkte beinhalten den Wertebereich von `0xFDD0` bis `0xFDEF` und Werte bis `0x10FFFF`, endend mit `FFFE` oder `FFFF`:

- `0xFFFE`, `0x1FFFE`, `0x2FFFE`, ..., `0xFFFFE`, `0x10FFFE`
- `0xFFFF`, `0x1FFFF`, `0x2FFFF`, ..., `0xFFFFF`, `0x10FFFF`

Im folgenden Beispiel ist der Fehlergrund angegeben, wenn COPY versucht, das UTF-8-Zeichen `e0 a1  c7a4` in eine CHAR-Spalte zu laden.

```
Multibyte character not supported for CHAR
(Hint: Try using  VARCHAR). Invalid char: e0 a1 c7a4
```

Wenn sich der Fehler auf einen VARCHAR-Datentyp bezieht, beinhaltet der Fehlergrund einen Fehlercode sowie die ungültige UTF-8-Hex-Sequenz. Im folgenden Beispiel ist der Fehlergrund angegeben, wenn COPY versucht, das UTF-8-Zeichen a4 in ein VARCHAR-Feld zu laden:

```
String contains invalid or unsupported UTF-8 codepoints.
Bad UTF-8 hex sequence: a4 (error 3)
```

In der folgenden Tabelle werden die Beschreibungen und vorgeschlagenen Umgehungen für VARCHAR-Ladefehler aufgelistet. Wenn einer dieser Fehler auftritt, ersetzen Sie das Zeichen durch eine gültige UTF-8-Codefolge oder entfernen das Zeichen.

Fehlercode	Beschreibung
1	Die UTF-8-Bytefolge überschreitet das Maximum von vier Bytes, das von VARCHAR unterstützt wird.
2	Die UTF-8-Bytefolge ist unvollständig. COPY konnte nicht die erwartete Anzahl von Fortsetzungsbytes für ein Multibyte-Zeichen vor dem Ende der Zeichenfolge finden.
3	Das UTF-8-Einzelbyte-Zeichen befindet sich außerhalb des Bereichs. Das Startbyte darf nicht 254, 255 oder ein Zeichen zwischen 128 und 191 (einschli eßlich) sein.
4	Der Wert des Bytes am Ende der Bytefolge befindet sich außerhalb des Bereichs. Das Fortsetzungsbyte muss zwischen 128 und 191 (einschließlich) liegen.
5	Das UTF-8-Zeichen ist als Surrogat reserviert. Surrogat-Codepunkte (U+D800 bis U+DFFF) sind nicht gültig.
8	Die Bytefolge überschreitet den maximal zulässigen UTF-8-Codepunkt.
9	Die UTF-8-Bytefolge besitzt keinen übereinstimmenden Codepunkt.

Ladefehlerreferenz

Wenn während des Ladens von Daten aus einer Datei Fehler auftreten, führen Sie eine Abfrage für die Tabelle STL_LOAD_ERRORS aus, um den Fehler zu identifizieren und die mögliche Erklärung zu ermitteln. In der folgenden Tabelle werden alle Fehlercodes aufgelistet, die beim Laden von Daten auftreten können:

Ladefehlercodes

Fehlercode	Beschreibung
1200	Unbekannter Fehler beim Parsen. Nehmen Sie Kontakt mit dem Support auf.
1201	In der Eingabedatei wurde kein Feldtrennzeichen gefunden.
1202	Die Eingabedaten hatten mehr Spalten als in der DDL definiert.
1203	Die Eingabedaten hatten weniger Spalten als in der DDL definiert.
1204	Die Eingabedaten überschritten den für den Datentyp akzeptablen Bereich.
1205	Das Datumsformat ist nicht gültig. Informationen zu gültigen Formaten finden Sie unter DATEFORMAT- und TIMEFORMAT-Zeichenfolgen.
1206	Das Zeitstempelformat ist nicht gültig. Informationen zu gültigen Formaten finden Sie unter DATEFORMAT- und TIMEFORMAT-Zeichenfolgen.
1207	Die Daten enthalten einen Wert außerhalb des erwarteten Bereichs von 0 bis 9.
1208	Formatierungsfehler für den Datentyp FLOAT.
1209	Formatierungsfehler für den Datentyp DECIMAL.
1210	Formatierungsfehler für den Datentyp BOOLEAN.
1211	Die Eingabezeile enthält keine Daten.
1212	Die Ladedatei wurde nicht gefunden.
1213	Ein als NOT NULL angegebenes Feld enthält keine Daten.
1214	Das Trennzeichen wurde nicht gefunden,
1215	Fehler für das CHAR-Feld.
1 216	Die Eingabezeile ist nicht gültig.
1217	Der Wert der Identitätsspalte ist nicht gültig.

Fehlercode	Beschreibung
1218	Bei Verwendung von NULL AS '\0' enthielt ein Feld mit einem Null-Terminator (NUL oder UTF-8 0000) mehr als ein Byte.
1219	UTF-8-Hexadezimal enthält eine ungültige Ziffer.
1220	Zeichenfolge enthält ungültige oder nicht unterstützte UTF-8-Codepunkte.
1221	Die Kodierung der Datei ist nicht mit der Kodierung identisch, die im COPY-Befehl angegeben wurde.
1222	Ganzzahlüberlauf-Fehler
1223	Der Datentyp ist nicht gültig.
1224	Eingabedaten weder im ordnungsgemäßen JSON-Format noch im Super-Datentyp
8001	COPY mit MANIFEST-Parameter erfordert den vollständigen Pfad eines Amazon-S3-Objekts.
9005	Ungültiger Endschlüssel angegeben.

Kontinuierliche Dateierfassung von Amazon S3 (Vorschau)

Dies ist eine Vorabversion der Dokumentation für Autocopy (SQL COPY JOB), die sich in der Vorabversion befindet. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur in Test- und nicht in Produktionsumgebungen zu verwenden. Die öffentliche Vorversion endet am 31. Juli 2024. Vorschau-Cluster werden zwei Wochen nach dem Ende der Vorschauversion automatisch entfernt. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Note

Sie können einen Amazon-Redshift-Cluster in der Vorschau erstellen, um neue Funktionen von Amazon Redshift zu testen. Sie haben nicht die Möglichkeit, diese Funktionen in der Produktion zu verwenden oder Ihren Vorschau-Cluster in einen Produktionscluster oder

einen Cluster auf einem anderen Pfad zu verschieben. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS - Servicebedingungen.

Erstellen eines Clusters in der Vorschau

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Provisioned clusters dashboard (Dashboard für bereitgestellte Cluster) und dann Clusters (Cluster) aus. Die aktuellen Cluster für Ihr Konto AWS-Region sind aufgeführt. Eine Teilmenge der Eigenschaften jedes Clusters wird in den Spalten der Liste angezeigt.
3. Auf der Seite mit der Clusterliste (Clusters) wird ein Banner angezeigt, das die Vorschau vorstellt. Wählen Sie die Schaltfläche Create preview cluster (Vorschau-Cluster erstellen) aus, um die Seite zum Erstellen von Clustern zu öffnen.
4. Geben Sie Eigenschaften für Ihren Cluster ein. Wählen Sie den Vorschaupfad (Preview track) aus, der die zu testenden Funktionen enthält. Wir empfehlen, einen Namen für den Cluster zu verwenden, der darauf hinweist, dass sich dieser auf einem Vorschaupfad befindet. Wählen Sie Optionen für Ihren Cluster, einschließlich Optionen mit der Bezeichnung -preview (Vorschau), für die zu testenden Funktionen. Allgemeine Informationen zum Erstellen von Clustern finden Sie unter Erstellen eines Clusters im Amazon-Redshift-Verwaltungshandbuch.
5. Wählen Sie Vorschau-Cluster erstellen aus, um einen Cluster in der Vorschau zu erstellen.
6. Wenn Ihr Vorschau-Cluster verfügbar ist, verwenden Sie Ihren SQL-Client, um Daten zu laden und abzufragen.

Ihr Cluster muss mit dem Vorschaupfad mit folgendem Namen erstellt werden: `preview_2023`. Verwenden Sie zum Testen einen neuen Cluster. Das Wiederherstellen eines Clusters in diesem Pfad wird nicht unterstützt. Die Funktion zum automatischen Kopieren ist in der Amazon-Redshift-Serverless-Arbeitsgruppe nicht verfügbar.
Diese Vorschau ist in den folgenden Sprachen verfügbar AWS-Regionen:

- Region USA Ost (Ohio) (us-east-2)
- Region USA Ost (Nord-Virginia) (us-east-1)
- Region USA West (Oregon) (us-west-2)

- Region Asien-Pazifik (Tokio) (ap-northeast-1)
- Region Europa (Stockholm) (eu-north-1)
- Region Europa (Irland) (eu-west-1)

Mithilfe von COPY JOB können Sie Daten aus Dateien, die in Amazon S3 gespeichert sind, in Ihre Amazon-Redshift-Tabellen zu laden. Amazon Redshift erkennt, wenn neue Amazon-S3-Dateien zu dem in Ihrem COPY-Befehl angegebenen Pfad hinzugefügt werden. Ein COPY-Befehl wird dann automatisch ausgeführt, ohne dass Sie eine externe Datenerfassungspipeline erstellen müssen. Amazon Redshift verfolgt, welche Dateien geladen wurden. Amazon Redshift bestimmt die Anzahl der Dateien, die für jeden COPY-Befehl zusammengefasst werden. Sie können die resultierenden COPY-Befehle in Systemansichten sehen.

Sie definieren einen COPY JOB einmalig. Für zukünftige Ausführungen werden dieselben Parameter verwendet.

Sie verwalten die Ladevorgänge mithilfe der Optionen CREATE, LIST, SHOW, DROP, ALTER und RUN. Weitere Informationen finden Sie unter COPY JOB (Vorschau).

Sie können Systemansichten abfragen, um den Status und Fortschritt von COPY JOB anzuzeigen. Die Ansichten werden wie folgt bereitgestellt:

- SYS_COPY_JOB (Vorschau) – enthält eine Zeile für jeden derzeit definierten COPY JOB.
- STL_LOAD_ERRORS – enthält Fehler von COPY-Befehlen.
- STL_LOAD_COMMITS – enthält Informationen, die zur Behebung von Fehlern beim Laden von COPY-Befehlsdaten verwendet werden.
- SYS_LOAD_HISTORY – enthält Details zu COPY-Befehlen.
- SYS_LOAD_ERROR_DETAIL – enthält Details zu Fehlern bei COPY-Befehlen.

Um die Liste der über COPY JOB geladenen Dateien abzurufen, führen Sie das folgende Beispiel aus und ersetzen Sie dabei *<job_id>*:

```
SELECT job_id, job_name, data_source, copy_query,filename,status, curtime
FROM sys_copy_job copyjob
JOIN stl_load_commits loadcommit
ON copyjob.job_id = loadcommit.copy_job_id
```

```
WHERE job_id = <job_id>;
```

Aktualisieren von Tabellen mit DML-Befehlen

Amazon Redshift unterstützt Data Manipulation Language (DML)-Standardbefehle (INSERT, UPDATE und DELETE), die Sie verwenden können, um Zeilen in Tabellen zu modifizieren. Sie können auch den Befehl TRUNCATE verwenden, um schnell eine große Zahl von Löschvorgängen auszuführen.

> **Note**
>
> Es wird nachdrücklich empfohlen, den Befehl COPY zu verwenden, um große Mengen von Daten zu laden. Die Verwendung einzelner INSERT-Anweisungen, um eine Tabelle auszufüllen, kann äußerst langsam sein. Wenn Ihre Daten in anderen Amazon-Redshift-Datenbanktabellen bereits vorhanden sind, können Sie alternativ den Befehl INSERT INTO ... verwenden. SELECT FROM oder CREATE TABLE AS verwenden, um die Leistung zu verbessern. Weitere Informationen finden Sie unter INSERT oder CREATE TABLE AS.

Wenn Sie im Vergleich zur Anzahl der Zeilen vor den Änderungen eine große Zahl von Zeilen in einer Tabelle einfügen, aktualisieren oder löschen, führen Sie für die Tabelle die Befehle ANALYZE und VACUUM aus, wenn Sie fertig sind. Wenn sich über die Zeit kleinere Änderungen in Ihrer Anwendung ansammeln, sollten Sie für die Befehle ANALYZE und VACUUM eine Ausführung in regelmäßigen Abständen planen. Weitere Informationen erhalten Sie unter Analysieren von Tabellen und Bereinigen von Tabellen.

Aktualisieren von Daten und Einfügen neuer Daten

Mit dem Befehl MERGE können Sie einer vorhandenen Tabelle effizient neue Daten hinzufügen. Führen Sie einen Zusammenführungsvorgang durch, indem Sie eine Staging-Tabelle erstellen und dann eine der in diesem Abschnitt beschriebenen Methoden verwenden, um die Zieltabelle aus der Staging-Tabelle zu aktualisieren. Weitere Informationen zum MERGE-Befehl finden Sie unter MERGE.

Themen

- Zusammenführungsmethode 1: Ersetzung vorhandener Zeilen

In Beispiele für Zusammenführungen wird ein Beispieldatensatz für Amazon Redshift verwendet, der sogenannte TICKIT-Datensatz. Als Voraussetzung können Sie die TICKIT-Tabellen und -Daten einrichten, indem Sie den Anweisungen unter Erste Schritte mit gängigen Datenbankaufgaben folgen. Weitere Informationen zum Beispieldatensatz finden Sie unter Beispieldatenbank.

Zusammenführungsmethode 1: Ersetzung vorhandener Zeilen

Wenn Sie alle Spalten in der Zieltabelle überschreiben, besteht die schnellste Methode der Zusammenführung darin, die vorhandenen Zeilen zu ersetzen. Bei dieser Methode wird die Zieltabelle nur einmal gescannt. Dabei wird ein innerer Join verwendet, um Zeilen zu löschen, die aktualisiert werden. Nach dem Löschen werden die Zeilen in einer einzigen Einfügungsoperation basierend auf der Staging-Tabelle durch neue Zeilen ersetzt.

Verwenden Sie diese Methode, wenn alle folgenden Bedingungen zutreffen:

- Zieltabelle und Staging-Tabelle enthalten dieselben Spalten.
- Sie möchten alle Daten in den Spalten der Zieltabelle durch alle Spalten der Staging-Tabelle ersetzen.
- Sie verwenden alle Zeilen in der Staging-Tabelle bei der Zusammenführung.

Wenn eines dieser Kriterien nicht zutrifft, verwenden Sie die Zusammenführungsmethode 2: Angabe einer Spaltenliste ohne den Befehl MERGE, wie im folgenden Abschnitt beschrieben.

Wenn Sie nicht alle Zeilen in der Staging-Tabelle verwenden, filtern Sie die Anweisungen DELETE und INSERT mittels einer WHERE-Klausel, um Zeilen auszulassen, die nicht geändert werden. Wenn jedoch die meisten Zeilen in der Staging-Tabelle nicht an der Zusammenführung beteiligt sind, wird die Ausführung einer UPDATE- und INSERT-Operation in getrennten Schritte empfohlen wie später in diesem Abschnitt beschrieben.

Zusammenführungsmethode 2: Angabe einer Spaltenliste ohne den Befehl MERGE

Verwenden Sie diese Methode, um spezifische Spalten in der Zieltabelle zu aktualisieren, statt ganze Zeilen zu überschreiben. Diese Methode dauert länger als die vorherige Methode, da sie einen zusätzlichen Update-Schritt erfordert und den Befehl MERGE nicht nutzt. Verwenden Sie diese Methode, wenn eine der folgenden Bedingungen zutrifft:

- Es müssen nicht alle Spalten in der Zieltabelle aktualisiert werden.
- Die meisten Zeilen in der Staging-Tabelle werden nicht für Aktualisierung verwendet.

Erstellen einer temporären Staging-Tabelle

Die Staging-Tabelle ist eine temporäre Tabelle, die alle Daten enthält, die für Änderungen der Zieltabelle verwendet werden, einschließlich Aktualisierungen und Einfügungen.

Eine Zusammenführungsoperation erfordert einen Join zwischen der Staging-Tabelle und der Zieltabelle. Um die Join-Zeilen zusammenzufassen, legen Sie den Verteilungsschlüssel der Staging-Tabelle auf dieselbe Spalte wie den Verteilungsschlüssel der Zieltabelle fest. Wenn die Zieltabelle beispielsweise eine Fremdschlüsselspalte als Verteilungsschlüssel verwendet, verwenden Sie dieselbe Spalte für den Verteilungsschlüssel der Staging-Tabelle. Wenn Sie die Staging-Tabelle mittels einer CREATE TABLE LIKE-Anweisung erstellen, erbt die Staging-Tabelle den Verteilungsschlüssel aus der übergeordneten Tabelle. Wenn Sie eine CREATE TABLE AS-Anweisung verwenden, erbt die neue Tabelle den Verteilungsschlüssel nicht. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen

Wenn der Verteilungsschlüssel nicht mit dem primären Schlüssel identisch ist und der Verteilungsschlüssel im Rahmen der Zusammenführungsoperation nicht aktualisiert wird, fügen Sie für die Verteilungsschlüsselspalten ein redundantes Join-Prädikat hinzu, um einen zusammengefassten Join zu ermöglichen. Beispiel:

```
where target.primarykey = stage.primarykey
and target.distkey = stage.distkey
```

Um zu überprüfen, ob die Abfrage einen zusammengefassten Join verwendet, führen Sie die Abfrage mit EXPLAIN aus und suchen für alle Joins nach DS_DIST_NONE. Weitere Informationen finden Sie unter Auswerten des Abfrageplans

Ausführen einer Zusammenführungsoperation durch Ersetzung vorhandener Zeilen

Wenn Sie die im Verfahren beschriebene Zusammenführungsoperation ausführen, führen Sie alle Schritte abgesehen vom Erstellen und Entfernen der temporären Staging-Tabelle in einer einzigen Transaktion aus. Die Transaktion wird rückgängig gemacht, falls ein Schritt fehlschlägt. Die Verwendung einer einzelnen Transaktion reduziert auch die Anzahl der Commit-Vorgänge, was Zeit und Ressourcen spart.

So führen Sie eine Zusammenführungsoperation durch Ersetzung vorhandener Zeilen aus

1. Erstellen Sie eine Staging-Tabelle und füllen Sie sie mit Daten aus, die zusammengeführt werden sollen, wie im folgenden Pseudocode gezeigt.

   ```
   create temp table stage (like target);

   insert into stage
   select * from source
   where source.filter = 'filter_expression';
   ```

2. Verwenden Sie MERGE, um einen Inner Join mit der Staging-Tabelle durchzuführen und die Zeilen aus der Zieltabelle zu aktualisieren, die der Staging-Tabelle entsprechen. Fügen Sie dann alle verbleibenden Zeilen in die Zieltabelle ein, die nicht mit der Staging-Tabelle übereinstimmen.

   Wir empfehlen, die Aktualisierungs- und Einfügevorgänge in einem einzigen MERGE-Befehl auszuführen.

   ```
   MERGE INTO target
   USING stage [optional alias] on (target.primary_key = stage.primary_key)
   WHEN MATCHED THEN
   UPDATE SET col_name1 = stage.col_name1 , col_name2= stage.col_name2, col_name3 =
    {expr}
   WHEN NOT MATCHED THEN
   INSERT (col_name1 , col_name2, col_name3) VALUES (stage.col_name1, stage.col_name2,
    {expr});
   ```

3. Entfernen Sie die Staging-Tabelle.

   ```
   drop table stage;
   ```

Ausführen einer Zusammenführungsoperation durch Angabe einer Spaltenliste ohne den Befehl MERGE

Wenn Sie die im Verfahren beschriebene Zusammenführungsoperation ausführen, führen Sie alle Schritte in einer einzigen Transaktion aus. Die Transaktion wird rückgängig gemacht, falls ein Schritt fehlschlägt. Die Verwendung einer einzelnen Transaktion reduziert auch die Anzahl der Commit-Vorgänge, was Zeit und Ressourcen spart.

So führen Sie eine Zusammenführungsoperation durch Angabe einer Spaltenliste aus

1. Platzieren Sie die gesamte Operation in einen einzigen Transaktionsblock.

```
begin transaction;
…
end transaction;
```

2. Erstellen Sie eine Staging-Tabelle und füllen Sie sie mit Daten aus, die zusammengeführt werden sollen, wie im folgenden Pseudocode gezeigt.

```
create temp table stage (like target);
insert into stage
select * from source
where source.filter = 'filter_expression';
```

3. Aktualisieren Sie die Zieltabelle mittels eines internen Joins mit der Staging-Tabelle.

   - Listen Sie in der UPDATE-Klausel explizit die Spalten auf, die aktualisiert werden sollen.
   - Führen Sie einen internen Join mit der Staging-Tabelle aus.
   - Wenn sich der Verteilungsschlüssel vom primären Schlüssel unterscheidet und der Verteilungsschlüssel nicht aktualisiert wird, fügen Sie für den Verteilungsschlüssel einen redundanten Join hinzu. Um zu überprüfen, ob die Abfrage einen zusammengefassten Join verwendet, führen Sie die Abfrage mit EXPLAIN aus und suchen für alle Joins nach DS_DIST_NONE. Weitere Informationen finden Sie unter Auswerten des Abfrageplans
   - Wenn Ihre Zieltabelle nach Zeitstempel sortiert ist, fügen Sie ein Prädikat hinzu, um Scans mit eingeschränkten Bereichen für die Zieltabelle zu nutzen. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift.
   - Wenn Sie nicht alle Zeilen in der Zusammenführung verwenden, fügen Sie eine Klausel hinzu, um die Zeilen herauszufiltern, die Sie ändern möchten. Sie können beispielsweise einen

Ungleichheitsfilter zu einer oder mehreren Zeilen hinzufügen, um Zeilen auszuschließen, die nicht geändert wurden.

- Platzieren Sie die Operationen zum Aktualisieren, Löschen und Einfügen in einem einzigen Transaktionsblock, damit ein Rollback ausgeführt werden kann, wenn es Probleme gibt.

Beispiel:

```
begin transaction;

update target
set col1 = stage.col1,
col2 = stage.col2,
col3 = 'expression'
from stage
where target.primarykey = stage.primarykey
and target.distkey = stage.distkey
and target.col3 > 'last_update_time'
and (target.col1 != stage.col1
or target.col2 != stage.col2
or target.col3 = 'filter_expression');
```

4. Löschen Sie nicht benötigte Zeilen aus der Staging-Tabelle, indem Sie einen internen Join mit der Zieltabelle verwenden. Einige Zeilen in der Zieltabelle stimmen bereits mit den entsprechenden Zeilen in der Staging-Tabelle überein. Andere Zeilen wurden im vorherigen Schritt aktualisiert. In beiden Fällen werden diese Zeilen für die Einfügungsoperation nicht benötigt.

```
delete from stage
using target
where stage.primarykey = target.primarykey;
```

5. Fügen Sie die verbleibenden Zeilen aus der Staging-Tabelle ein. Verwenden Sie hierfür dieselbe Spaltenliste in der VALUES-Klausel, die Sie in Schritt zwei in der UPDATE-Anweisung verwendet haben.

```
insert into target
(select col1, col2, 'expression'
from stage);

end transaction;
```

6. Entfernen Sie die Staging-Tabelle.

```
drop table stage;
```

Beispiele für Zusammenführungen

In den folgenden Beispielen werden Zusammenführungen ausgeführt, um die Tabelle SALES zu aktualisieren. Im ersten Beispiel wird die einfachere Methode verwendet, bei der alle Zeilen in der Zieltabelle gelöscht werden und anschließend alle Zeilen aus der Staging-Tabelle eingefügt werden. Im zweiten Beispiel müssen ausgewählte Spalten in der Zieltabelle aktualisiert werden. Daher enthält es einen zusätzlichen Aktualisierungsschritt.

In Beispiele für Zusammenführungen wird ein Beispieldatensatz für Amazon Redshift verwendet, der sogenannte TICKIT-Datensatz. Als Voraussetzung können Sie die TICKIT-Tabellen und -Daten einrichten, indem Sie den Anweisungen in der Anleitung Erste Schritte mit gängigen Datenbankaufgaben folgen. Weitere Informationen zum Beispieldatensatz finden Sie unter Beispieldatenbank.

Beispieldatenquelle für eine Zusammenführung

Die Beispiele in diesem Abschnitt benötigen eine Beispieldatenquelle, die sowohl Aktualisierungen als auch Einfügungen enthält. Zu diesem Zweck wird eine Beispieltabelle namens SALES_UPDATE erstellt, die Daten aus der Tabelle SALES verwendet. Die neue Tabelle wird mit zufälligen Daten ausgefüllt, die neue Vertriebsaktivitäten für den Dezember darstellen. Die Beispieltabelle SALES_UPDATE wird verwendet, um die Staging-Tabelle in den folgenden Beispielen zu erstellen.

```
-- Create a sample table as a copy of the SALES table.

create table tickit.sales_update as
select * from tickit.sales;

-- Change every fifth row to have updates.

update tickit.sales_update
set qtysold = qtysold*2,
pricepaid = pricepaid*0.8,
commission = commission*1.1
where saletime > '2008-11-30'
and mod(sellerid, 5) = 0;
```

```
-- Add some new rows to have inserts.
-- This example creates a duplicate of every fourth row.

insert into tickit.sales_update
select (salesid + 172456) as salesid, listid, sellerid, buyerid, eventid, dateid,
 qtysold, pricepaid, commission, getdate() as saletime
from tickit.sales_update
where saletime > '2008-11-30'
and mod(sellerid, 4) = 0;
```

Beispiel für eine Zusammenführung, bei der vorhandene Zeilen basierend auf übereinstimmenden Schlüsseln ersetzt werden

Im folgenden Skript wird die Tabelle SALES_UPDATE verwendet, um für die Tabelle SALES eine Zusammenführungsoperation mit neuen Daten für Vertriebsaktivitäten im Dezember auszuführen. In diesem Beispiel werden Zeilen in der SALES-Tabelle ersetzt, die aktualisiert wurden. In diesem Beispiel aktualisieren wir die Spalten QTYSOLD und PRICEPAID. Die Spalten COMMISSION und SALETIME bleiben unverändert.

```
MERGE into tickit.sales
USING tickit.sales_update sales_update
on ( sales.salesid = sales_update.salesid
and sales.listid = sales_update.listid
and sales_update.saletime > '2008-11-30'
and (sales.qtysold != sales_update.qtysold
or sales.pricepaid != sales_update.pricepaid))
WHEN MATCHED THEN
update SET qtysold = sales_update.qtysold,
pricepaid = sales_update.pricepaid
WHEN NOT MATCHED THEN
INSERT (salesid, listid, sellerid, buyerid, eventid, dateid, qtysold , pricepaid,
 commission, saletime)
values (sales_update.salesid, sales_update.listid, sales_update.sellerid,
 sales_update.buyerid, sales_update.eventid,
sales_update.dateid, sales_update.qtysold , sales_update.pricepaid,
 sales_update.commission, sales_update.saletime);

-- Drop the staging table.
drop table tickit.sales_update;

-- Test to see that commission and salestime were not impacted.
SELECT sales.salesid, sales.commission, sales.salestime, sales_update.commission,
 sales_update.salestime
```

```
FROM tickit.sales
INNER JOIN tickit.sales_update sales_update
ON
sales.salesid = sales_update.salesid
AND sales.listid = sales_update.listid
AND sales_update.saletime > '2008-11-30'
AND (sales.commission != sales_update.commission
OR sales.salestime != sales_update.salestime);
```

Beispiel für eine Zusammenführung, bei der eine Spaltenliste angegeben wird, ohne den Befehl MERGE zu verwenden

Im folgenden Beispiel wird eine Zusammenführungsoperation ausgeführt, um SALES durch neue Daten für Vertriebsaktivitäten im Dezember zu aktualisieren. Die Beispieldaten müssen sowohl Aktualisierungen als auch Einfügungen sowie nicht geänderte Zeilen enthalten. In diesem Beispiel sollen die Spalten QTYSOLD und PRICEPAID aktualisiert werden. Die Spalten COMMISSION und SALETIME sollen dagegen nicht verändert werden. Im folgenden Skript wird die Tabelle SALES_UPDATE verwendet, um für die Tabelle SALES eine Zusammenführungsoperation auszuführen.

```
-- Create a staging table and populate it with rows from SALES_UPDATE for Dec
create temp table stagesales as select * from sales_update
where saletime > '2008-11-30';

-- Start a new transaction
begin transaction;

-- Update the target table using an inner join with the staging table
-- The join includes a redundant predicate to collocate on the distribution key -- A
 filter on saletime enables a range-restricted scan on SALES

update sales
set qtysold = stagesales.qtysold,
pricepaid = stagesales.pricepaid
from stagesales
where sales.salesid = stagesales.salesid
and sales.listid = stagesales.listid
and stagesales.saletime > '2008-11-30'
and (sales.qtysold != stagesales.qtysold
or sales.pricepaid != stagesales.pricepaid);

-- Delete matching rows from the staging table
```

```
-- using an inner join with the target table

delete from stagesales
using sales
where sales.salesid = stagesales.salesid
and sales.listid = stagesales.listid;

-- Insert the remaining rows from the staging table into the target table
insert into sales
select * from stagesales;

-- End transaction and commit
end transaction;

-- Drop the staging table
drop table stagesales;
```

Ausführen einer Deep Copy-Operation

In einer Deep Copy-Operation wird eine Tabelle mittels einer Masseneinfügung neu erstellt und ausgefüllt. Dabei wird die Tabelle automatisch sortiert. Wenn eine Tabelle über eine große, nicht sortierte Region verfügt, ist eine Deep Copy-Operation sehr viel schneller als eine Bereinigung. Wir empfehlen, während einer Deep-Copy-Operation nur dann gleichzeitige Aktualisierungen auszuführen, wenn Sie diese nachverfolgen können. Verschieben Sie nach Abschluss des Vorgangs die Delta-Updates in die neue Tabelle. Eine VACUUM-Operation unterstützt gleichzeitige Updates automatisch.

Für das Erstellen einer Kopie der ursprünglichen Tabelle steht Ihnen eine der folgenden Methoden zur Verfügung:

- Verwendung der ursprünglichen Tabellen-DDL.

  Wenn CREATE TABLE DDL verfügbar ist, ist dies die schnellste und bevorzugte Methode. Wenn Sie eine neue Tabelle erstellen, können Sie alle Tabellen- und Spaltenattribute angeben, einschließlich primärer Schlüssel und Fremdschlüssel. Sie können die ursprüngliche DDL mithilfe der Funktion SHOW TABLE suchen.
- Verwendung von CREATE TABLE LIKE.

  Wenn die ursprüngliche DDL nicht verfügbar ist, können Sie CREATE TABLE LIKE verwenden, um die ursprüngliche Tabelle neu zu erstellen. Die neue Tabelle erbt die Kodierung, den

Verteilungsschlüssel, den Sortierschlüssel und die NOT NULL-Attribute der übergeordneten Tabelle. Die neue Tabelle erbt die Primärschlüssel- und Fremdschlüsselattribute der übergeordneten Tabelle nicht. Sie können diese jedoch mittels hinzufügen ALTER TABLE.

- Erstellen Sie eine temporäre Tabelle und verkürzen Sie die ursprüngliche Tabelle.

  Wenn Sie die Primärschlüssel- und Fremdschlüsselattribute der übergeordneten Tabelle beibehalten müssen. Wenn die übergeordnete Tabelle Abhängigkeiten hat, können Sie CREATE TABLE ... AS (CTAS) verwenden, um eine temporäre Tabelle zu erstellen. Verkürzen Sie anschließend die ursprüngliche Tabelle und füllen Sie sie mit Daten aus der temporären Tabelle.

  Durch die Verwendung einer temporären Tabelle wird die Leistung im Vergleich zur Verwendung einer permanenten Tabelle deutlich verbessert. Es besteht jedoch das Risiko, dass Daten verloren gehen. Eine temporäre Tabelle wird am Ende der Sitzung, in der sie erstellt wurde, automatisch entfernt. TRUNCATE führt sofort einen Commit aus, auch wenn er innerhalb eines Transaktionsblocks ausgeführt wird. Wenn TRUNCATE erfolgreich ist, die Sitzung jedoch beendet wird, bevor die nachfolgende INSERT-Operation abgeschlossen ist, gehen die Daten verloren. Wenn ein Datenverlust nicht akzeptabel ist, verwenden Sie eine permanente Tabelle.

Nachdem Sie eine Kopie einer Tabelle erstellt haben, müssen Sie möglicherweise Zugriff auf die neue Tabelle gewähren. Sie können GRANT verwenden, um Zugriffsberechtigungen zu definieren. Um alle Zugriffsberechtigungen für eine Tabelle anzeigen und erteilen zu können, müssen Sie einer der folgenden Kategorien angehören:

- Superuser
- Besitzer der Tabelle, die Sie kopieren möchten.
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE, um die Berechtigungen der Tabelle anzuzeigen, und mit dem Recht zum Erteilen aller relevanten Berechtigungen.

Darüber hinaus müssen Sie möglicherweise eine Nutzungsberechtigung für das Schema erteilen, in dem sich Ihre Deep Copy befindet. Eine Nutzungsberechtigung muss erteilt werden, wenn sich das Schema Ihrer Deep Copy vom Schema der Originaltabelle unterscheidet und es sich nicht um das `public`-Schema handelt. Um Nutzungsrechte anzeigen und gewähren zu können, müssen Sie einer der folgenden Kategorien angehören:

- Superuser
- Benutzer, der die USAGE-Berechtigung für das Schema der Deep Copy erteilen kann.

Ausführen einer Deep Copy-Operation mittels der DDL der ursprünglichen Tabelle

1. (Optional) Erstellen Sie die Tabellen-DDL durch Ausführen eines Skripts namens neu `v_generate_tbl_ddl`.
2. Erstellen Sie mittels der ursprünglichen CREATE TABLE DDL eine Kopie der Tabelle.
3. Verwenden Sie eine INSERT INTO … SELECT-Anweisung, um die Kopie mit Daten aus der ursprünglichen Tabelle auszufüllen.
4. Suchen Sie nach Berechtigungen, die für die alte Tabelle erteilt wurden. Sie können diese Berechtigungen in der Systemansicht SVV_RELATION_PRIVILEGES sehen.
5. Falls erforderlich, gewähren Sie der neuen Tabelle die Berechtigungen der alten Tabelle.
6. Erteilen Sie jeder Gruppe und jedem Benutzer, die/der über Berechtigungen in der ursprünglichen Tabelle verfügt, eine Nutzungsberechtigung. Dieser Schritt ist nicht erforderlich, wenn sich Ihre Deep-Copy-Tabelle im `public`-Schema oder in demselben Schema wie die Originaltabelle befindet.
7. Entfernen Sie die ursprüngliche Tabelle.
8. Verwenden Sie eine ALTER TABLE-Anweisung, um die Kopie mit dem Namen der ursprünglichen Tabelle umzubenennen.

Im folgenden Beispiel wird für die Tabelle SAMPLE unter Verwendung eines Duplikats von SAMPLE namens sample_copy eine Deep-Copy-Operation ausgeführt.

```
--Create a copy of the original table in the sample_namespace namespace using the
 original CREATE TABLE DDL.
create table sample_namespace.sample_copy ( … );

--Populate the copy with data from the original table in the public namespace.
insert into sample_namespace.sample_copy (select * from public.sample);

--Check SVV_RELATION_PRIVILEGES for the original table's privileges.
select * from svv_relation_privileges where namespace_name = 'public' and relation_name
 = 'sample' order by identity_type, identity_id, privilege_type;

--Grant the original table's privileges to the copy table.
grant DELETE on table sample_namespace.sample_copy to group group1;
grant INSERT, UPDATE on table sample_namespace.sample_copy to group group2;
grant SELECT on table sample_namespace.sample_copy to user1;
grant INSERT, SELECT, UPDATE on table sample_namespace.sample_copy to user2;
```

```
--Grant usage permission to every group and user that has privileges in the original
 table.
grant USAGE on schema sample_namespace to group group1, group group2, user1, user2;

--Drop the original table.
drop table public.sample;

--Rename the copy table to match the original table's name.
alter table sample_namespace.sample_copy rename to sample;
```

Ausführen einer Deep Copy-Operation mittels CREATE TABLE LIKE

1. Erstellen Sie mittels CREATE TABLE LIKE eine neue Tabelle.
2. Verwenden Sie eine INSERT INTO … SELECT-Anweisung, um die Zeilen aus der aktuellen Tabelle in die neue Tabelle zu kopieren.
3. Suchen Sie nach Berechtigungen, die für die alte Tabelle erteilt wurden. Sie können diese Berechtigungen in der Systemansicht SVV_RELATION_PRIVILEGES sehen.
4. Falls erforderlich, gewähren Sie der neuen Tabelle die Berechtigungen der alten Tabelle.
5. Erteilen Sie jeder Gruppe und jedem Benutzer, die/der über Berechtigungen in der ursprünglichen Tabelle verfügt, eine Nutzungsberechtigung. Dieser Schritt ist nicht erforderlich, wenn sich Ihre Deep-Copy-Tabelle im `public`-Schema oder in demselben Schema wie die Originaltabelle befindet.
6. Entfernen Sie die aktuelle Tabelle.
7. Verwenden Sie eine ALTER TABLE-Anweisung, um die neue Tabelle mit dem Namen der ursprünglichen Tabelle umzubenennen.

Im folgenden Beispiel wird für die Tabelle SAMPLE mittels CREATE TABLE LIKE eine Deep-Copy-Operation ausgeführt.

```
--Create a copy of the original table in the sample_namespace namespace using CREATE
 TABLE LIKE.
create table sameple_namespace.sample_copy (like public.sample);

--Populate the copy with data from the original table.
insert into sample_namespace.sample_copy (select * from public.sample);

--Check SVV_RELATION_PRIVILEGES for the original table's privileges.
```

```
select * from svv_relation_privileges where namespace_name = 'public' and relation_name
 = 'sample' order by identity_type, identity_id, privilege_type;

--Grant the original table's privileges to the copy table.
grant DELETE on table sample_namespace.sample_copy to group group1;
grant INSERT, UPDATE on table sample_namespace.sample_copy to group group2;
grant SELECT on table sample_namespace.sample_copy to user1;
grant INSERT, SELECT, UPDATE on table sample_namespace.sample_copy to user2;

--Grant usage permission to every group and user that has privileges in the original
 table.
grant USAGE on schema sample_namespace to group group1, group group2, user1, user2;

--Drop the original table.
drop table public.sample;

--Rename the copy table to match the original table's name.
alter table sample_namespace.sample_copy rename to sample;
```

Ausführen einer Deep Copy-Operation durch Erstellen einer temporären Tabelle und Verkürzen der ursprünglichen Tabelle

1. Verwenden Sie CREATE TABLE AS, um eine temporäre Tabelle mit den Zeilen aus der ursprünglichen Tabelle zu erstellen.
2. Verkürzen Sie die aktuelle Tabelle.
3. Verwenden Sie eine INSERT INTO … SELECT-Anweisung, um die Zeilen aus der temporären Tabelle in die ursprüngliche Tabelle zu kopieren.
4. Entfernen Sie die temporäre Tabelle.

Im folgenden Beispiel wird für die Tabelle SALES eine Deep-Copy-Operation durch Erstellen einer temporären Tabelle und Verkürzen der ursprünglichen Tabelle ausgeführt. Da die ursprüngliche Tabelle erhalten bleibt, müssen Sie keine Berechtigungen für die kopierte Tabelle erteilen.

```
--Create a temp table copy using CREATE TABLE AS.
create temp table salestemp as select * from sales;

--Truncate the original table.
truncate sales;

--Copy the rows from the temporary table to the original table.
```

```
insert into sales (select * from salestemp);

--Drop the temporary table.
drop table salestemp;
```

Analysieren von Tabellen

Verwenden Sie die ANALYZE-Operation, um die statistischen Metadaten zu aktualisieren, die vom Abfrageplaner verwendet werden, um optimale Pläne zu wählen.

In den meisten Fällen brauchen Sie den Befehl ANALYZE nicht explizit auszuführen. Amazon Redshift überwacht Änderungen an Ihrer Workload und aktualisiert die Statistik im Hintergrund automatisch. Außerdem führt der COPY-Befehl nach dem Laden von Daten in eine leere Tabelle automatisch eine Analyse durch.

Wenn Sie eine Tabelle oder die gesamte Datenbank explizit analysieren möchten, führen Sie den Befehl ANALYZE aus.

Themen

- Automatische Analyse
- Analyse neuer Tabellendaten
- Verlauf für den Befehl ANALYZE

Automatische Analyse

Amazon Redshift überwacht fortlaufend Ihre Datenbank und führt automatisch Analyseoperationen im Hintergrund durch. Um die Auswirkungen auf die Systemperformance zu minimieren, wird die automatischen Analyse in Zeiten ausgeführt, in denen der Workload gering ist.

Die automatische Analyse ist standardmäßig aktiviert. Um die automatische Analyse zu deaktivieren, legen Sie für den Parameter `auto_analyze` den Wert **`false`** fest, indem Sie die Parametergruppe Ihres Clusters entsprechend ändern.

Zur Verkürzung der Verarbeitungszeit und zur Verbesserung der allgemeinen Systemleistung überspringt Amazon Redshift die automatische Analyse bei Tabellen mit nur geringen Änderungen.

Bei einem Analysevorgang werden Tabellen mit up-to-date Statistiken übersprungen. Wenn ANALYZE im Rahmen des ETL-Workflows (Extrahieren, Transformieren und Laden) ausgeführt

wird, werden Tabellen, deren Statistik aktuell ist, bei der automatischen Analyse übersprungen. Aus entsprechenden Gründen werden bei einem expliziten Aufruf von ANALYZE Tabellen ausgelassen, deren Statistik bereits bei einer automatischen Analyse aktualisiert wurde.

Analyse neuer Tabellendaten

Standardmäßig ruft der COPY-Befehl nach dem Laden von Daten in eine leere Tabelle den ANALYZE-Befehl auf. Sie können unabhängig davon, ob eine Tabelle leer ist, den Aufruf von ANALYZE erzwingen, indem Sie STATUPDATE auf ON festlegen. Wenn Sie für STATUPDATE den Wert OFF angeben, wird keine Analyse ausgeführt. Nur der Tabellenbesitzer oder ein Superuser können den Befehl ANALYZE oder den Befehl COPY mit STATUPDATE auf ON ausführen.

Amazon Redshift analysiert auch automatisch neue Tabellen, die mit den folgenden Befehlen erstellt werden:

- CREATE TABLE AS (CTAS)
- CREATE TEMP TABLE AS
- SELECT INTO

Wenn Sie eine Abfrage für eine neue Tabelle ausführen, die nach dem anfänglichen Laden der Daten noch nicht analysiert wurde, gibt Amazon Redshift eine Warnung zurück. Wenn Sie jedoch dieselbe Tabelle nach einem anschließenden Aktualisierungs- oder Ladevorgang eine Abfrage ausführen, wird keine Warnung angezeigt. Dieselbe Warnmeldung wird auch zurückgegeben, wenn Sie den Befehl EXPLAIN für eine Abfrage ausführen, die auf Tabellen verweist, die noch nicht analysiert wurden.

Wenn Sie einer nichtleeren Tabelle eine bedeutende Menge von Daten hinzufügen, können Sie die Statistik explizit aktualisieren. Sie können dies mit dem Befehl ANALYZE oder durch Festlegen von STATUPDATE auf ON erreichen. Um Details zur Anzahl der Zeilen anzuzeigen, die seit der letzten ANALYZE-Operation eingefügt oder gelöscht wurden, führen Sie eine Abfrage für die PG_STATISTIC_INDICATOR-Systemkatalogtabelle aus.

Sie können den Geltungsbereich für den Befehl ANALYZE auf eine der folgenden Einstellungen festlegen:

- Die gesamte aktuelle Datenbank
- Eine einzelne Tabelle
- Eine oder mehrere spezifische Spalten in einer einzelnen Tabelle

- Spalten, die in Abfragen wahrscheinlich als Prädikate verwendet werden

Der ANALYZE-Befehl ruft eine Zeilenstichprobe aus der Tabelle ab, führt einige Berechnungen aus und speichert die sich daraus ergebenden Spaltenstatistiken. Standardmäßig führt Amazon Redshift einen Stichprobendurchgang für die Spalte DISTKEY und einen weiteren Stichprobendurchgang für alle anderen Spalten in der Tabelle aus. Wenn Sie Statistiken für einen Teilsatz von Spalten generieren möchten, können Sie eine durch Komma getrennte Spaltenliste angeben. Wenn Sie ANALYZE mit der PREDICATE COLUMNS-Klausel ausführen, werden dabei nur Prädikatspalten berücksichtigt.

ANALYZE-Operationen sind ressourcenintensiv. Sie sollten sie daher nur für Tabellen und Spalten ausführen, für die Statistikaktualisierungen tatsächlich erforderlich sind. Sie müssen nicht regelmäßig oder zu gleichen Termin alle Spalten in allen Tabellen analysieren. Wenn sich die Daten wesentlich verändern, analysieren Sie die Spalten, die häufig bei Folgendem verwendet werden:

- Sortierungs- und Gruppierungsoperationen
- Joins
- Abfrageprädikate

Zur Verkürzung der Verarbeitungszeit und zur Verbesserung der allgemeinen Systemleistung überspringt Amazon Redshift die ANALYZE-Operation für alle Tabellen mit einem niedrigen Prozentsatz geänderter Zeilen, wie durch den Parameter analyze_threshold_percent angegeben. Standardmäßig ist der Analyseschwellenwert auf 10 Prozent festgelegt. Sie können den Analyseschwellenwert für die aktuelle Sitzung durch die Ausführung eines SET-Befehls ändern.

Spalten, die weniger wahrscheinlich häufige Analysen erfordern, sind Spalten, die Fakten und Messungen und zugehörige Attribute enthalten, die niemals tatsächlich abgefragt werden, beispielsweise große VARCHAR-Spalten. Betrachten Sie beispielsweise die Tabelle LISTING in der TICKIT-Datenbank.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'listing';

column         |        type        | encoding | distkey | sortkey
---------------+--------------------+----------+---------+---------
listid         | integer            | none     | t       | 1
sellerid       | integer            | none     | f       | 0
eventid        | integer            | mostly16 | f       | 0
```

```
dateid         | smallint           | none     | f       | 0
numtickets     | smallint           | mostly8  | f       | 0
priceperticket | numeric(8,2)       | bytedict | f       | 0
totalprice     | numeric(8,2)       | mostly32 | f       | 0
listtime       | timestamp with...  | none     | f       | 0
```

Wenn diese Tabelle jeden Tag mit einer großen Zahl neuer Datensätze geladen wird, muss die Spalte LISTID, die häufig als Join-Schlüssel in Abfragen verwendet wird, regelmäßig analysiert werden. Wenn TOTALPRICE und LISTTIME die häufig verwendeten Einschränkungen in Abfragen sind, können Sie diese Spalten und den Verteilungsschlüssel jeden Wochentag analysieren.

```
analyze listing(listid, totalprice, listtime);
```

Angenommen, die Verkäufer und Veranstaltungen in der Anwendung sind sehr viel statischer und die Datums-IDs geben einen festen Satz von Tagen an, allerdings nur aus zwei oder drei Jahren. In diesem Fall ändern sich die jeweiligen Werte für diese Spalten nicht wesentlich. Die Anzahl der Instances der einzelnen spezifischen Werte wird jedoch stetig steigen.

Betrachten Sie zusätzlich Fälle, in denen die Messungen NUMTICKETS und PRICEPERTICKET im Vergleich zur Spalte TOTALPRICE selten abgefragt werden. In diesem Fall können Sie den ANALYZE-Befehl einmal an jedem Wochenende für die gesamte Tabelle ausführen, um die Statistiken für die fünf Spalten zu aktualisieren, die nicht täglich analysiert werden:

Prädikatsspalten

Eine bequeme Alternative zur Angabe einer Spaltenliste stellt die Analyse nur der Spalten dar, die wahrscheinlich als Prädikate verwendet werden. Wenn Sie eine Abfrage ausführen, werden alle Spalten, die in einem Join, einer Filterbedingung oder einer GROUP BY-Klausel verwendet werden, im Systemkatalog als Prädikatsspalten markiert. Wenn Sie ANALYZE mit der PREDICATE COLUMNS-Klausel ausführen, betrifft die Analyseoperation nur Spalten, die den folgenden Kriterien entsprechen:

- Die Spalte ist als Prädikatsspalte markiert.
- Die Spalte ist ein Verteilungsschlüssel.
- Die Spalte ist Teil eines Sortierschlüssels.

Wenn keine Spalten der Tabelle als Prädikate markiert sind, schließt ANALYZE alle Spalten ein, auch wenn PREDICATE COLUMNS angegeben ist. Wenn keine Spalten als Prädikatspalten markiert sind, kann dies daran liegen, dass für die Tabelle noch keine Abfrage ausgeführt wurde.

Sie sollten PREDICATE COLUMNS verwenden, wenn das Abfragemuster Ihres Workloads vergleichsweise stabil ist. Wenn das Abfragemuster variabel ist und unterschiedliche Spalten häufig als Prädikate verwendet werden, kann die Verwendung von PREDICATE COLUMNS vorübergehend zu veralteten Statistiken führen. Veraltete Statistiken können zu nicht optimalen Abfrageausführungszeitplänen und langen Ausführungszeiten führen. Wenn Sie ANALYZE das nächste Mal mit PREDICATE COLUMNS ausführen, sind die neuen Prädikatspalten jedoch enthalten.

Um Details für Prädikatspalten anzuzeigen, verwenden Sie den folgenden SQL-Code, um eine Ansicht namens PREDICATE_COLUMNS zu erstellen.

```
CREATE VIEW predicate_columns AS
WITH predicate_column_info as (
SELECT ns.nspname AS schema_name, c.relname AS table_name, a.attnum as col_num,
 a.attname as col_name,
        CASE
            WHEN 10002 = s.stakind1 THEN array_to_string(stavalues1, '||')
            WHEN 10002 = s.stakind2 THEN array_to_string(stavalues2, '||')
            WHEN 10002 = s.stakind3 THEN array_to_string(stavalues3, '||')
            WHEN 10002 = s.stakind4 THEN array_to_string(stavalues4, '||')
            ELSE NULL::varchar
        END AS pred_ts
   FROM pg_statistic s
   JOIN pg_class c ON c.oid = s.starelid
   JOIN pg_namespace ns ON c.relnamespace = ns.oid
   JOIN pg_attribute a ON c.oid = a.attrelid AND a.attnum = s.staattnum)
SELECT schema_name, table_name, col_num, col_name,
       pred_ts NOT LIKE '2000-01-01%' AS is_predicate,
       CASE WHEN pred_ts NOT LIKE '2000-01-01%' THEN (split_part(pred_ts,
 '||',1))::timestamp ELSE NULL::timestamp END as first_predicate_use,
       CASE WHEN pred_ts NOT LIKE '%||2000-01-01%' THEN (split_part(pred_ts,
 '||',2))::timestamp ELSE NULL::timestamp END as last_analyze
FROM predicate_column_info;
```

Angenommen, Sie führen die folgende Abfrage für die Tabelle LISTING aus. Beachten Sie, dass LISTID, LISTTIME und EVENTID in den Join-, Filter- und Group by-Klauseln verwendet werden.

```
select s.buyerid,l.eventid, sum(l.totalprice)
from listing l
join sales s on l.listid = s.listid
where l.listtime > '2008-12-01'
```

```
group by l.eventid, s.buyerid;
```

Wenn Sie eine Abfrage für die Ansicht PREDICATE_COLUMNS ausführen, wie im folgenden Beispiel gezeigt, sehen Sie, dass LISTID, EVENTID und LISTTIME als Prädikatspalten markiert sind.

```
select * from predicate_columns
where table_name = 'listing';
```

```
schema_name | table_name | col_num | col_name       | is_predicate |
 first_predicate_use | last_analyze
------------+------------+---------+----------------+--------------
+---------------------+--------------------
public      | listing    |       1 | listid         | true         | 2017-05-05
 19:27:59 | 2017-05-03 18:27:41
public      | listing    |       2 | sellerid       | false        |
  | 2017-05-03 18:27:41
public      | listing    |       3 | eventid        | true         | 2017-05-16
 20:54:32 | 2017-05-03 18:27:41
public      | listing    |       4 | dateid         | false        |
  | 2017-05-03 18:27:41
public      | listing    |       5 | numtickets     | false        |
  | 2017-05-03 18:27:41
public      | listing    |       6 | priceperticket | false        |
  | 2017-05-03 18:27:41
public      | listing    |       7 | totalprice     | false        |
  | 2017-05-03 18:27:41
public      | listing    |       8 | listtime       | true         | 2017-05-16
 20:54:32 | 2017-05-03 18:27:41
```

Wenn die Statistiken aktuell sind, kann dies die Leistung bei Abfragen verbessern, weil die Abfrageplanung dann optimale Pläne auswählen kann. Amazon Redshift aktualisiert die Statistik im Hintergrund automatisch, und Sie können außerdem den Befehl ANALYZE explizit ausführen. Wenn Sie ANALYZE explizit aufrufen möchten, gehen Sie wie folgt vor:

- Führen Sie den Befehl ANALYZE aus, bevor Sie Abfragen ausführen.
- Führen Sie am Ende jedes regulären Lade- oder Aktualisierungszyklus routinemäßig den Befehl ANALYZE über der Datenbank aus.
- Führen Sie den Befehl ANALYZE über allen neuen Tabellen aus, die Sie erstellen, aber auch über allen vorhandenen Tabellen oder Spalten mit wesentlichen Änderungen.

- Sie sollten ANALYZE-Operationen für verschiedene Arten von Tabellen und Spalten zu unterschiedlichen Zeitpunkten ausführen, abhängig von ihrer Verwendung in Abfragen und der Wahrscheinlichkeit von Änderungen.
- Um Zeit und Cluster-Ressourcen zu sparen, sollten Sie die PREDICATE COLUMNS-Klausel verwenden, wenn Sie ANALYZE ausführen.

Sie müssen den Befehl ANALYZE nicht explizit ausführen, nachdem Sie einen Snapshot in einem bereitgestellten Cluster oder Serverless-Namespace wiederhergestellt oder einen pausierten bereitgestellten Cluster fortgesetzt haben. Amazon Redshift behält in diesen Fällen die Systemtabelleninformationen bei, sodass manuelle ANALYZE-Befehle nicht erforderlich sind. Amazon Redshift wird bei Bedarf weiterhin automatische Analysevorgänge ausführen.

Bei einem Analysevorgang werden Tabellen mit Statistiken übersprungen. up-to-date Wenn ANALYZE im Rahmen des ETL-Workflows (Extrahieren, Transformieren und Laden) ausgeführt wird, werden Tabellen, deren Statistik aktuell ist, bei der automatischen Analyse übersprungen. Aus entsprechenden Gründen werden bei einem expliziten Aufruf von ANALYZE Tabellen ausgelassen, deren Statistik bereits bei einer automatischen Analyse aktualisiert wurde.

Verlauf für den Befehl ANALYZE

Es ist nützlich, zu wissen, wann der Befehl ANALYZE zum letzten Mal für eine Tabelle oder Datenbank ausgeführt wurde. Während der Ausführung des Befehls ANALYZE führt Amazon Redshift mehrere Abfragen aus, die wie folgt aussehen:

```
padb_fetch_sample: select * from table_name
```

Führen Sie eine Abfrage für STL_ANALYZE aus, um den Verlauf von ANALZYE-Operationen anzuzeigen. Wenn Amazon Redshift eine Tabelle automatisch analysiert, wird für die Spalte `is_background` der Wert `t` (true) festgelegt. Andernfalls lautet der Wert `f` (false). Im folgende Beispiel wird STV_TBL_PERM verknüpft, um den Tabellennamen und Details zur Ausführungszeit anzuzeigen.

```
select distinct a.xid, trim(t.name) as name, a.status, a.rows, a.modified_rows,
 a.starttime, a.endtime
from stl_analyze a
join stv_tbl_perm t  on t.id=a.table_id
where name = 'users'
```

```
order by starttime;

xid    | name  | status          | rows  | modified_rows | starttime           |
 endtime
-------+-------+-----------------+-------+---------------+---------------------
+---------------------
  1582 | users | Full            | 49990 |         49990 | 2016-09-22 22:02:23 |
 2016-09-22 22:02:28
244287 | users | Full            | 24992 |         74988 | 2016-10-04 22:50:58 |
 2016-10-04 22:51:01
244712 | users | Full            | 49984 |         24992 | 2016-10-04 22:56:07 |
 2016-10-04 22:56:07
245071 | users | Skipped         | 49984 |             0 | 2016-10-04 22:58:17 |
 2016-10-04 22:58:17
245439 | users | Skipped         | 49984 |          1982 | 2016-10-04 23:00:13 |
 2016-10-04 23:00:13
(5 rows)
```

Alternativ können Sie eine komplexere Abfrage ausführen, die alle Anweisungen zurückgibt, die in allen abgeschlossenen Transaktionen ausgeführt wurden und den Befehl ANALYZE enthielten:

```
select xid, to_char(starttime, 'HH24:MM:SS.MS') as starttime,
datediff(sec,starttime,endtime ) as secs, substring(text, 1, 40)
from svl_statementtext
where sequence = 0
and xid in (select xid from svl_statementtext s where s.text like 'padb_fetch_sample
%' )
order by xid desc, starttime;

xid  |  starttime   | secs |                 substring
-----+--------------+------+-------------------------------------------
1338 | 12:04:28.511 |    4 | Analyze date
1338 | 12:04:28.511 |    1 | padb_fetch_sample: select count(*) from
1338 | 12:04:29.443 |    2 | padb_fetch_sample: select * from date
1338 | 12:04:31.456 |    1 | padb_fetch_sample: select * from date
1337 | 12:04:24.388 |    1 | padb_fetch_sample: select count(*) from
1337 | 12:04:24.388 |    4 | Analyze sales
1337 | 12:04:25.322 |    2 | padb_fetch_sample: select * from sales
1337 | 12:04:27.363 |    1 | padb_fetch_sample: select * from sales
...
```

Bereinigen von Tabellen

Amazon Redshift kann im Hintergrund automatisch eine VACUUM DELETE-Operation für Tabellen sortieren und ausführen. Um Tabellen nach einem Laden oder einer Reihe von inkrementellen Updates zu bereinigen, können Sie auch den VACUUM-Befehl ausführen (entweder gegen die gesamte Datenbank oder gegen einzelne Tabellen).

> **Note**
>
> Nur Benutzer mit den erforderlichen Tabellenberechtigungen können eine Tabelle effektiv bereinigen. Wenn VACUUM ohne die notwendigen Tabellenberechtigungen ausgeführt, wird die Operation zwar erfolgreich abgeschlossen, hat jedoch keine Wirkung. Eine Liste der gültigen Tabellenberechtigungen zum effektiven Ausführen von VACUUM finden Sie unter VACUUM.
>
> Aus diesem Grund empfehlen wir, das Vacuuming einzelner Tabellen bei Bedarf durchzuführen. Wir empfehlen ebenfalls diesen Ansatz, da das Vacuuming der gesamten Datenbank ein teurer Vorgang sein kann.

Automatische Tabellensortierung

Amazon Redshift sortiert automatisch die Daten im Hintergrund, um die Tabellendaten in der Reihenfolge ihres Sortierschlüssels zu sortieren. Amazon Redshift verfolgt Ihre Scan-Abfragen, um festzustellen, welche Abschnitte der Tabelle von der Sortierung profitieren.

Abhängig von der Auslastung des Systems leitet Amazon Redshift die Sortierung automatisch ein. Diese automatische Sortierung erübrigt die Ausführung des Befehls VACUUM, um die Daten in der Reihenfolge der Sortierkriterien zu halten. Wenn Sie Daten vollständig nach dem Sortkey sortiert benötigen, z. B. nach einer großen Datenmenge, können Sie den Befehl VACUUM trotzdem manuell ausführen. Um festzustellen, ob Ihre Tabelle von der Ausführung von VACUUM SORT profitieren wird, überwachen Sie die Spalte `vacuum_sort_benefit` in SVV_TABLE_INFO.

Amazon Redshift verfolgt Scan-Abfragen, die den Sortierschlüssel für jede Tabelle verwenden. Amazon Redshift schätzt den maximalen Prozentsatz der Verbesserung beim Scannen und Filtern von Daten für die Tabelle (wenn die Tabelle vollständig sortiert war). Diese Schätzung ist in der Spalte `vacuum_sort_benefit` in SVV_TABLE_INFO sichtbar. Sie können diese Spalte zusammen mit der Spalte `unsorted` verwenden, um festzulegen, wann Abfragen davon profitieren können,

VACUUM SORT manuell für eine Tabelle auszuführen. Die Spalte `unsorted` spiegelt die physische Sortierreihenfolge einer Tabelle wider. Die Spalte `vacuum_sort_benefit` gibt die Auswirkung der Sortierung einer Tabelle durch manuelles Ausführen von VACUUM SORT an.

Nehmen Sie beispielsweise die folgende Abfrage:

```
select "table", unsorted,vacuum_sort_benefit from svv_table_info order by 1;
```

```
 table | unsorted | vacuum_sort_benefit
-------+----------+---------------------
 sales |    85.71 |                5.00
 event |    45.24 |               67.00
```

Obwohl die Tabelle "sales" ~86 % physisch unsortiert ist, hat dies nur einen Einfluss 5 % auf die Abfrageleistung. Dies kann entweder daran liegen, dass nur ein kleiner Teil der Tabelle von Abfragen angesprochen wird oder dass nur sehr wenige Abfragen auf die Tabelle zugegriffen haben. Die Tabelle "event" ist ~45 % physisch unsortiert. Aber die Änderung der Abfrageleistung von 67 % zeigt auf, dass entweder ein größerer Teil der Tabelle von Abfragen abgerufen wurde oder dass die Anzahl der auf die Tabelle zugreifenden Abfragen groß war. Die Tabelle "event" kann potentiell von der Ausführung von VACUUM SORT profitieren.

Automatisches Aufrufen von VACUUM DELETE

Wenn Sie einen Löschvorgang ausführen, werden die Zeilen zum Löschen markiert, jedoch nicht entfernt. Amazon Redshift führt automatisch einen VACUUM DELETE-Vorgang im Hintergrund aus, der auf der Anzahl der gelöschten Zeilen in Datenbanktabellen basiert. Amazon Redshift legt für VACUUM DELETE eine zeitplangesteuerte Ausführung so fest, dass diese Ausführung in Zeiten fällt, in denen der Workload gering ist, und hält die Ausführung bei hoher Auslastung an.

Themen

- Häufigkeit von Bereinigungen (VACUUM)
- Sortierphase und Zusammenführungsphase
- Schwellenwert für die Bereinigung
- Arten von Bereinigungen
- Verwalten der Bereinigungszeiten

Häufigkeit von Bereinigungen (VACUUM)

Sie sollten Bereinigungen so häufig wie notwendig ausführen, um eine konsistente Abfrageleistung aufrechtzuerhalten. Bei der Festlegung der Häufigkeit, mit der Sie den Befehl VACUUM ausführen, sollten Sie folgende Faktoren berücksichtigen:

- Führen Sie VACUUM während Zeiten aus, in denen Sie für den Cluster nur minimale Aktivität erwarten, beispielsweise abends oder während festgelegter Zeitfenster für die Datenbankadministration.
- Führen Sie VACUUM-Befehle außerhalb von Wartungsfenstern aus. Weitere Informationen finden Sie unter Zeitplanung um Wartungsfenster.
- Eine große, nicht sortierte Region führt zu längeren Bereinigungszeiten. Wenn Sie die Bereinigung verzögern, dauert sie länger, da mehr Daten neu organisiert werden müssen.
- VACUUM ist eine I/O-intensive Operation. Je länger es also dauert, bis die Bereinigung abgeschlossen ist, desto größer sind die Auswirkungen auf gleichzeitig ausgeführte Abfragen und andere Datenbankoperationen, die für Ihren Cluster ausgeführt werden.
- VACUUM benötigt für Tabellen, die eine überlappende Sortierung verwenden, länger. Um zu beurteilen, ob überlappende Tabellen neu sortiert werden müssen, fragen Sie die Ansicht SVV_INTERLEAVED_COLUMNS ab.

Sortierphase und Zusammenführungsphase

Amazon Redshift führt Bereinigungsoperationen in zwei Phasen aus. Zunächst werden die Zeilen in der nicht sortierten Region sortiert. Anschließend werden die neu sortierten Zeilen am Ende der Tabelle mit den vorhandenen Zeilen zusammengeführt, wenn notwendig. Beim Bereinigen einer großen Tabelle wird die Bereinigungsoperation in einer Reihe von Schritten ausgeführt, die aus inkrementellen Sortierungen gefolgt von Zusammenführungen bestehen. Wenn die Operation fehlschlägt oder Amazon Redshift während der Bereinigung offline geht, befindet sich die teilweise bereinigte Tabelle oder Datenbank in einem konsistenten Zustand. Sie müssen jedoch die Bereinigungsoperation manuell neu starten. Inkrementelle Sortierungen gehen verloren. Die zusammengeführten Zeilen, für die vor dem Fehler ein Commit ausgeführt wurde, müssen jedoch nicht erneut bereinigt werden. Wenn die nicht sortierte Region groß ist, kann erhebliche Zeit verloren gehen. Weitere Informationen zu den Sortierungs- und Zusammenführungsphasen finden Sie unter Verwalten des Volumens zusammengeführter Zeilen.

Benutzer können auf Tabellen zugreifen, während sie bereinigt werden. Sie können Abfragen und Schreiboperationen ausführen, während eine Tabelle bereinigt wird. Wenn jedoch DML und eine Bereinigung gleichzeitig ausgeführt werden, dauern beide Vorgänge möglicherweise länger. Wenn Sie während einer Bereinigung UPDATE- und DELETE-Anweisungen ausführen, wird die Systemleistung möglicherweise reduziert. Inkrementelle Zusammenführungen blockieren gleichzeitig ausgeführte UPDATE- und DELETE-Operationen. UPDATE- und DELETE-Operationen wiederum blockieren inkrementelle Zusammenführungen für die betroffenen Tabellen. DDL-Operationen wie ALTER TABLE werden blockiert, bis die Bereinigungsoperation für die Tabelle abgeschlossen ist.

> **Note**
>
> Verschiedene Modifikatoren für VACUUM steuern die Funktionsweise. Sie können sie verwenden, um die Bereinigung an den aktuellen Bedarf anzupassen. Beispielsweise verkürzt die Verwendung von VACUUM RECLUSTER die Bereinigung, indem keine vollständige Verschmelzung durchgeführt wird. Weitere Informationen finden Sie unter VACUUM.

Schwellenwert für die Bereinigung

Standardmäßig überspringt VACUUM die Sortierungsphase für alle Tabellen, in denen mehr als 95 Prozent der Tabellenzeilen bereits sortiert sind. Das Überspringen der Sortierungsphase kann die Leistung von VACUUM deutlich verbessern. Um den Standardschwellenwert für die Sortierung für eine einzelne Tabelle zu ändern, verwenden Sie den Tabellennamen und den Parameter TO threshold PERCENT, wenn Sie den Befehl VACUUM ausführen.

Arten von Bereinigungen

Weitere Informationen zu den verschiedenen Arten von Bereinigungstypen finden Sie unter VACUUM.

Verwalten der Bereinigungszeiten

Abhängig von der Art Ihrer Daten wird empfohlen, die in diesem Abschnitt beschriebenen Verfahren zu befolgen, um die Bereinigungszeiten zu minimieren.

Themen

- Entscheidung über die Neuindizierung

- Verwalten der Größe der nicht sortierten Region
- Verwalten des Volumens zusammengeführter Zeilen
- Laden von Daten in der Reihenfolge des Sortierschlüssels
- Verwenden von Zeitreihentabellen

Entscheidung über die Neuindizierung

Häufig können Sie die Abfrageleistung deutlich verbessern, indem Sie einen überlappenden Sortierstil verwenden. Mit der Zeit verschlechtert sich die Leistung jedoch möglicherweise, wenn die Verteilung der Werte in den Sortierschlüsselspalten geändert wird.

Wenn Sie eine leere, überlappende Tabelle mittels COPY oder CREATE TABLE AS laden, erstellt Amazon Redshift den überlappenden Index automatisch. Wenn Sie eine überlappende Tabelle mittels INSERT laden, müssen Sie im Anschluss VACUUM REINDEX ausführen, um den überlappenden Index zu initialisieren.

Während Sie Zeilen mit neuen Sortierschlüsselwerten hinzufügen, kann sich die Leistung verschlechtern, wenn die Verteilung der Werte in den Sortierschlüsselspalten geändert wird. Wenn Ihre neuen Zeilen primär innerhalb des Bereichs der vorhandenen Sortierschlüsselwerte liegen, müssen Sie keine Neuindizierung ausführen. Führen Sie VACUUM SORT ONLY oder VACUUM FULL aus, um die Sortierreihenfolge wiederherzustellen.

Das Abfragemodul kann die Sortierreihenfolge verwenden, um effizient festzulegen, welche Datenblöcke gescannt werden müssen, um eine Abfrage zu verarbeiten. Im Fall einer überlappenden Sortierung analysiert Amazon Redshift die Sortierschlüsselspaltenwerte, um die optimale Sortierreihenfolge zu ermitteln. Wenn aufgrund hinzugefügter Zeilen die Verteilung der Schlüsselwerte geändert oder verschoben wird, ist die Sortierstrategie nicht mehr optimal und der Vorteil, den die Sortierung für die Leistung hat, nimmt ab. Um die Sortierschlüsselverteilung neu zu analysieren, können Sie eine VACUUM REINDEX-Operation ausführen. Die Neuindizierungsoperation ist zeitaufwändig. Um festzustellen, ob eine Tabelle von einer Neuindizierung profitiert, führen Sie eine Abfrage für die Ansicht SVV_INTERLEAVED_COLUMNS aus.

Beispielsweise zeigt die folgende Abfrage Details für Tabellen an, die überlappende Sortierschlüssel verwenden.

```
select tbl as tbl_id, stv_tbl_perm.name as table_name,
```

```
col, interleaved_skew, last_reindex
from svv_interleaved_columns, stv_tbl_perm
where svv_interleaved_columns.tbl = stv_tbl_perm.id
and interleaved_skew is not null;

 tbl_id | table_name | col | interleaved_skew | last_reindex
--------+------------+-----+------------------+---------------------
 100048 | customer   |   0 |             3.65 | 2015-04-22 22:05:45
 100068 | lineorder  |   1 |             2.65 | 2015-04-22 22:05:45
 100072 | part       |   0 |             1.65 | 2015-04-22 22:05:45
 100077 | supplier   |   1 |             1.00 | 2015-04-22 22:05:45
(4 rows)
```

Der Wert für `interleaved_skew` ist ein Verhältnis, das die Menge der Verschiebung angibt. Ein Wert von 1 bedeutet, dass es keine Verschiebung gegeben hat. Wenn die Verschiebung größer als 1,4 ist, verbessert eine VACUUM REINDEX -Operation in der Regel die Leistung, es sei denn, die Verschiebung ist ein Merkmal des zugrundeliegenden Satzes.

Sie können den Datumswert in `last_reindex` verwenden, um festzustellen, wie viel Zeit seit der letzten Neuindizierung verstrichen ist.

Verwalten der Größe der nicht sortierten Region

Die nicht sortierte Region nimmt an Größe zu, wenn Sie große Mengen neuer Daten in Tabellen laden, die bereits Daten enthalten, oder wenn Sie Tabellen nicht als Teil der routinemäßigen Wartungsoperationen bereinigen. Um lange Ausführungszeiten von Bereinigungsoperationen zu vermeiden, verwenden Sie die folgenden Verfahren:

- Führen Sie Bereinigungsoperationen regelmäßig aus.

  Wenn Sie Ihre Tabellen in kleinen Inkrementen laden (beispielsweise täglichen Updates, die einen kleinen Prozentsatz der gesamten Zeilenzahl in der Tabelle darstellen), hilft die regelmäßige Ausführung von VACUUM, eine schnelle Ausführung der einzelnen Bereinigungsoperationen sicherzustellen.
- Führen Sie den größten Ladevorgang zuerst aus.

  Wenn Sie eine neue Tabelle mit mehreren COPY-Operationen laden müssen, führen Sie den größten Ladevorgang zuerst aus. Wenn Sie einen Ladevorgang zum ersten Mal in eine neue oder verkürzte Tabelle ausführen, werden alle Daten direkt in die sortierte Region geladen, sodass keine Bereinigung erforderlich ist.

- Verkürzen Sie eine Tabelle, statt alle Zeilen zu löschen.

  Durch das Löschen von Zeilen aus einer Tabelle wird der Platz nicht zurückgewonnen, den die Zeilen vor der Ausführung der Bereinigungsoperation belegt haben. Durch das Verkürzen einer Tabelle werden jedoch die Tabelle geleert und der Festplattenplatz zurückgewonnen; daher ist keine Bereinigung erforderlich. Alternativ können Sie die Tabelle entfernen und neu erstellen.
- Verkürzen oder entfernen Sie Testtabellen.

  Wenn Sie zu Testzwecken eine kleine Zahl von Zeilen in eine Tabelle laden, sollten Sie die Zeilen nach Abschluss des Vorgangs nicht löschen. Verkürzen Sie stattdessen die Tabelle und laden Sie diese Zeilen als Teil der anschließenden Produktionsladeoperation neu.
- Führen Sie eine Deep Copy-Operation aus.

  Wenn eine Tabelle mit einer zusammengesetzten Sortierschlüsseltabelle einen großen, nicht sortierten Bereich besitzt, ist eine Deep Copy-Operation sehr viel schneller als eine Bereinigung. Eine Deep-Kopie erstellt eine Tabelle neu und füllt diese, indem sie einen Bulk-Insert verwendet, der die Tabelle automatisch neu sortiert. Wenn eine Tabelle einen großen, nicht sortierten Bereich besitzt, ist eine Deep Copy-Operation sehr viel schneller als eine Bereinigung. Der Kompromiss besteht darin, dass Sie während einer Deep Copy-Operation anders als bei einer Bereinigung nicht gleichzeitig Aktualisierungen ausführen können. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift.

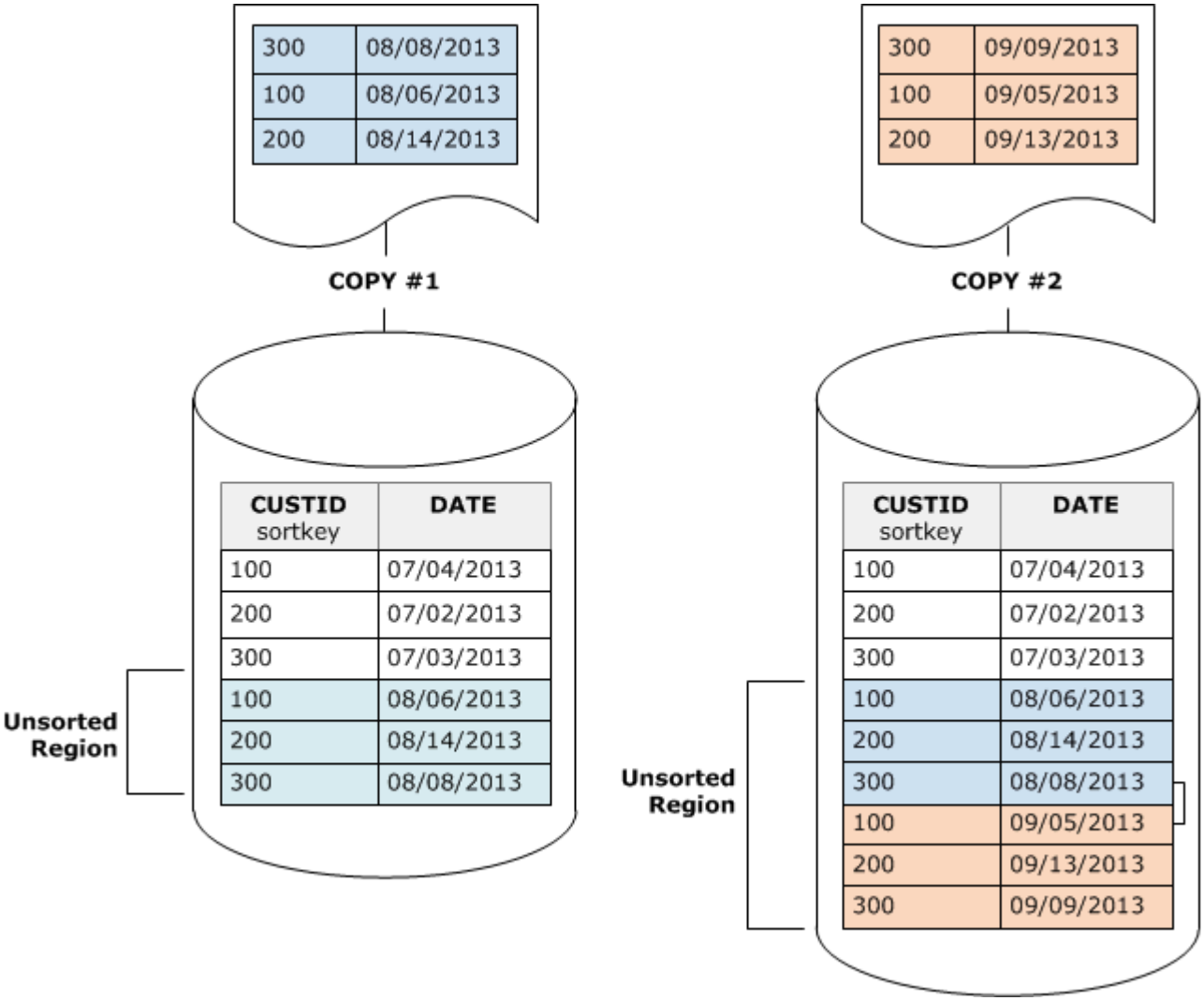
Verwalten des Volumens zusammengeführter Zeilen

Wenn eine Bereinigungsoperation neue Zeilen mit der sortierten Region einer Tabelle zusammenführen muss, nimmt der für die Bereinigung benötigte Zeitraum zu, wenn die Tabelle größer wird. Sie können die Leistung der Bereinigung verbessern, indem Sie die Zahl der Zeilen reduzieren, die zusammengeführt werden müssen.

Vor einer Bereinigung besteht eine Tabelle aus einer sortierten Region zu Beginn der Tabelle, gefolgt von einer nicht sortierten Region, deren Größe jedes Mal zunimmt, wenn Zeilen hinzugefügt oder aktualisiert werden. Wenn ein Satz von Zeilen durch eine COPY-Operation hinzugefügt wird, wird der neue Satz von Zeilen anhand des Sortierschlüssels sortiert, wenn er der nicht sortierten Region am Ende der Tabelle hinzugefügt wird. Die neuen Zeilen werden innerhalb des eigenen Satzes, jedoch nicht innerhalb der nicht sortierten Region sortiert.

Im folgenden Diagramm wird die nicht sortierte Region nach zwei aufeinanderfolgenden COPY-Operationen gezeigt. Der Sortierschlüssel ist CUSTID. Aus Gründen der Einfachheit zeigt dieses

Beispiel einen zusammengesetzten Sortierschlüssel. Für überlappende Sortierschlüssel gelten jedoch dieselben Grundsätze, abgesehen davon, dass die Auswirkungen der nicht sortierten Region bei überlappenden Tabellen größer sind.



Eine Bereinigung stellt die Sortierreihenfolge der Tabelle in zwei Phasen wieder her:

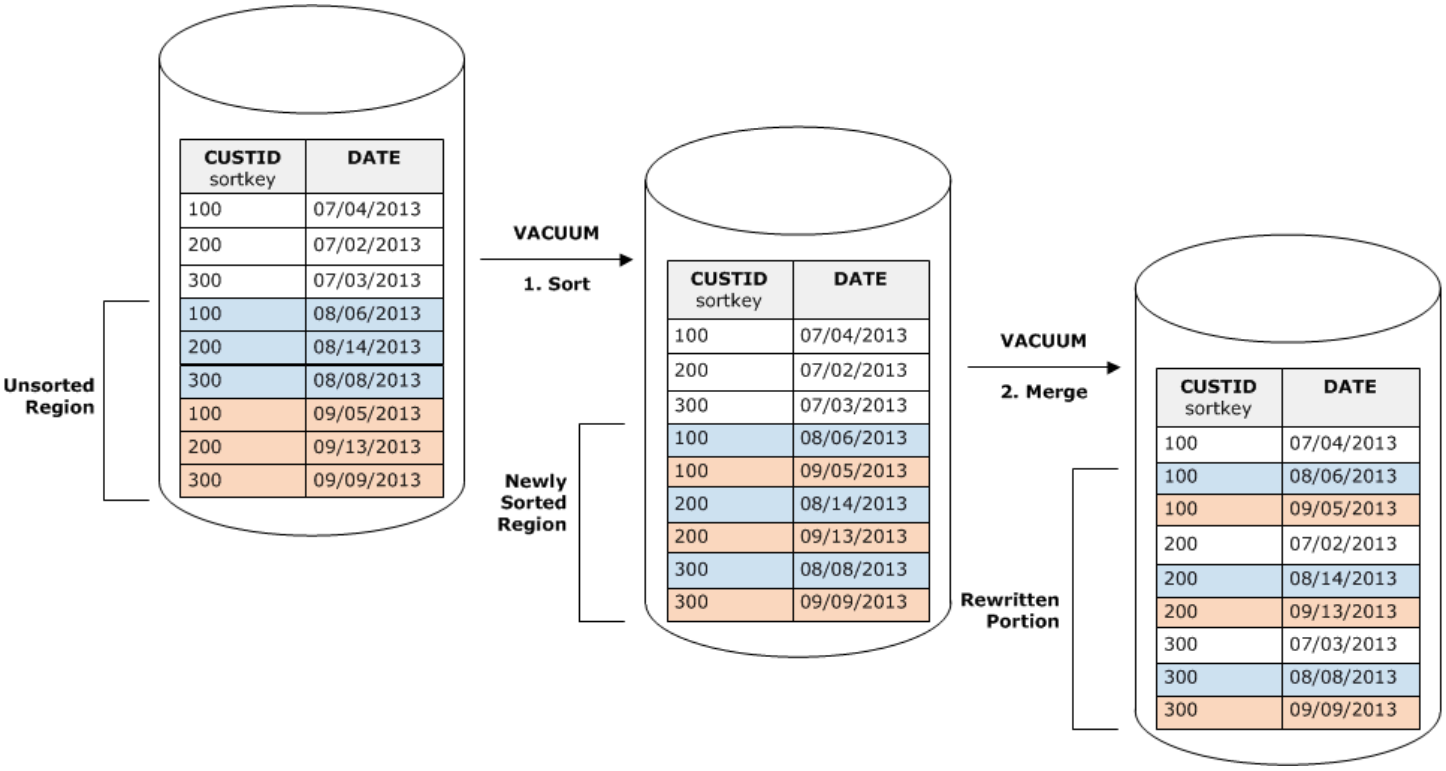
1. Die nicht sortierte Region wird zu einer neu sortierten Region sortiert.

   Die erste Phase ist vergleichsweise kostengünstig, da nur die nicht sortierte Region neu geschrieben wird. Wenn der Bereich der Sortierschlüsselwerte der neu sortierten Region größer als der vorhandene Bereich ist, müssen nur die neuen Zeilen neu geschrieben werden und die Bereinigung ist abgeschlossen. Wenn die sortierte Region beispielsweise ID-Werte von 1 bis 500 enthält und nachfolgende COPY-Operationen Schlüsselwerte hinzufügen, die größer als 500 sind, dann muss nur die nicht sortierte Region neu geschrieben werden.
2. Führen Sie die neu sortierte Region mit der zuvor sortierten Region zusammen.

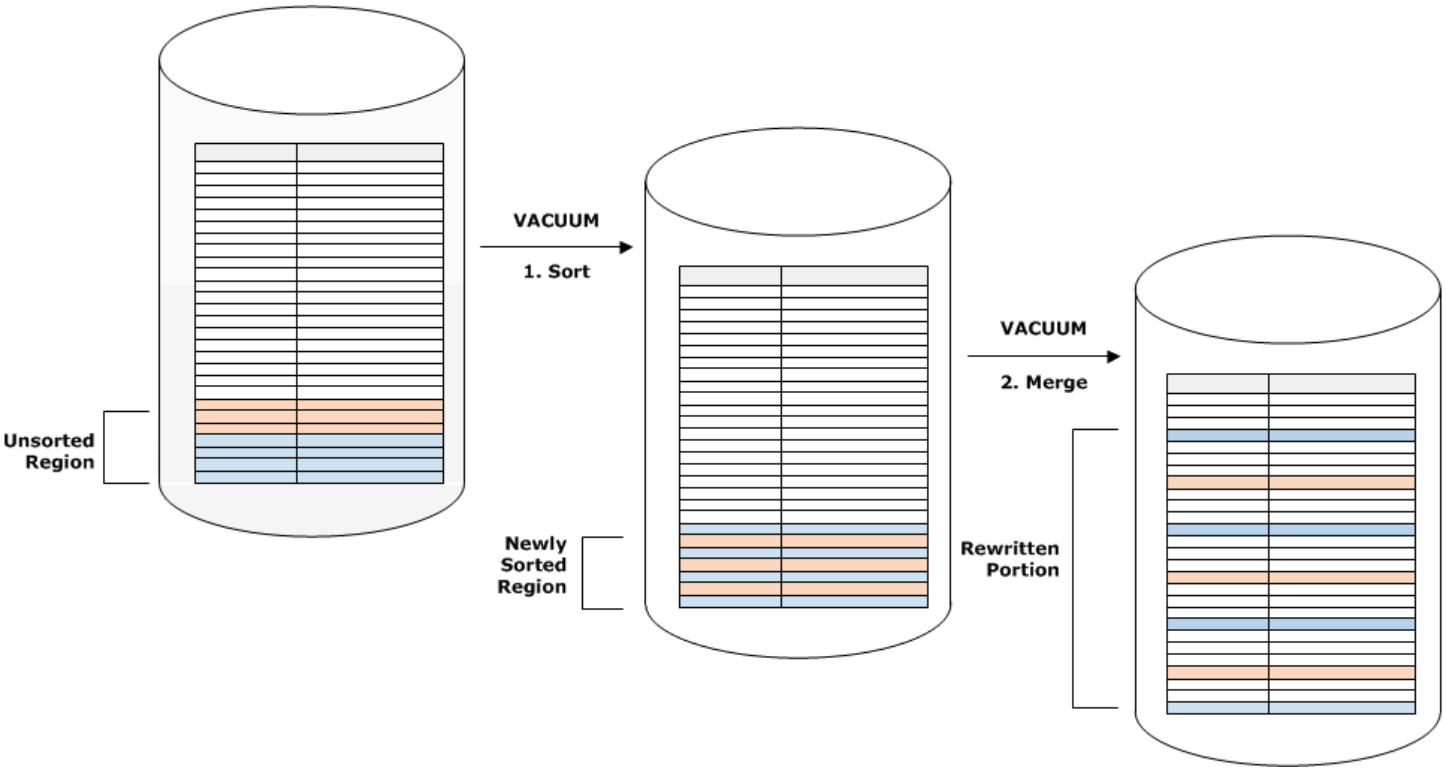
Wenn die Schlüssel in der neu sortierten Region mit den Schlüsseln in der sortierten Region überlappen, muss VACUUM die Zeilen zusammenführen. Beginnend mit dem Anfang der neu sortierten Region (beim niedrigsten Sortierschlüssel) schreibt die Bereinigungsoperation die zusammengeführten Zeilen aus der zuvor sortierten Region und der neu sortierten Region in einen neuen Satz von Blöcken.

Der Umfang, mit dem der neue Sortierschlüsselbereich mit den vorhandenen Sortierschlüsseln überlappt, legt den Umfang fest, in dem die zuvor sortierte Region neu geschrieben werden muss. Wenn die nicht sortierten Schlüssel über den vorhandenen Sortierbereich verstreut sind, muss eine Bereinigungsoperation möglicherweise vorhandene Teile der Tabelle neu schreiben.

Im folgenden Diagramm wird gezeigt, wie eine Bereinigungsoperation Zeilen sortierten und zusammenführen würde, die einer Tabelle mit CUSTID als Sortierschlüssel hinzugefügt wurden. Da jede Kopieroperation einen neuen Satz von Zeilen mit Schlüsselwerten hinzufügt, die die vorhandenen Schlüssel überlappen, muss beinahe die gesamte Tabelle neu geschrieben werden. Das Diagramm zeigt einen einzelnen Sortierungs- und Zusammenführungsvorgang. In der Praxis bestehen große Bereinigungen jedoch aus einer Reihe inkrementeller Sortierungs- und Zusammenführungsvorgänge.



Wenn der Bereich von Sortierschlüsseln in einem Satz neuer Zeilen mit dem Bereich vorhandener Schlüssel überlappt, nehmen die Kosten der Zusammenführungsphase entsprechend der Tabellengröße zu, während die Tabelle an Größe zunimmt. Die Kosten der Sortierphase entsprechen jedoch weiter der Größe der nicht sortierten Region. In diesem Fall sind die Kosten der Zusammenführungsphase größer als die Kosten der Sortierphase, wie das folgende Diagramm zeigt.



Um den Anteil einer Tabelle zu ermitteln, der neu zusammengeführt wurde, führen Sie eine Abfrage für SVV_VACUUM_SUMMARY aus, nachdem die Bereinigungsoperation abgeschlossen ist. Die folgende Abfrage zeigt die Auswirkungen von sechs aufeinanderfolgenden Bereinigungsoperationen, während CUSTSALES mit der Zeit an Größe zunahm.

```
select * from svv_vacuum_summary
where table_name = 'custsales';

 table_name | xid  | sort_      | merge_     | elapsed_   | row_  | sortedrow_ | block_
  | max_merge_
            |      | partitions | increments | time       | delta | delta      | delta
  | partitions
 -----------+------+------------+------------+------------+-------+------------
+---------+---------------
  custsales | 7072 |          3 |          2 |  143918314 |     0 |   88297472 |   1524
  |      47
```

```
 custsales | 7122 |          3 |          3 |  164157882 |     0 |   88297472 |    772
 |      47
 custsales | 7212 |          3 |          4 |  187433171 |     0 |   88297472 |    767
 |      47
 custsales | 7289 |          3 |          4 |  255482945 |     0 |   88297472 |    770
 |      47
 custsales | 7420 |          3 |          5 |  316583833 |     0 |   88297472 |    769
 |      47
 custsales | 9007 |          3 |          6 |  306685472 |     0 |   88297472 |    772
 |      47
(6 rows)
```

Die Spalte merge_increments zeigt die Menge der Daten an, die für die einzelnen Bereinigungsoperationen zusammengeführt wurden. Wenn die Zahl der Zusammenführungsinkremente über aufeinanderfolgende Bereinigungen entsprechend dem Wachstum der Tabellengröße zunimmt, ist das ein Anzeichen dafür, dass jede Bereinigungsoperation eine zunehmende Zahl von Zeilen in der Tabelle neu zusammenführt, da die vorhandenen und neu sortierten Regionen überlappen.

Laden von Daten in der Reihenfolge des Sortierschlüssels

Wenn Sie Ihre Daten in der Reihenfolge des Sortierschlüssels mithilfe eines COPY-Befehls laden, können Sie die Notwendigkeit einer Bereinigung verringern oder die Bereinigung sogar ganz überflüssig machen.

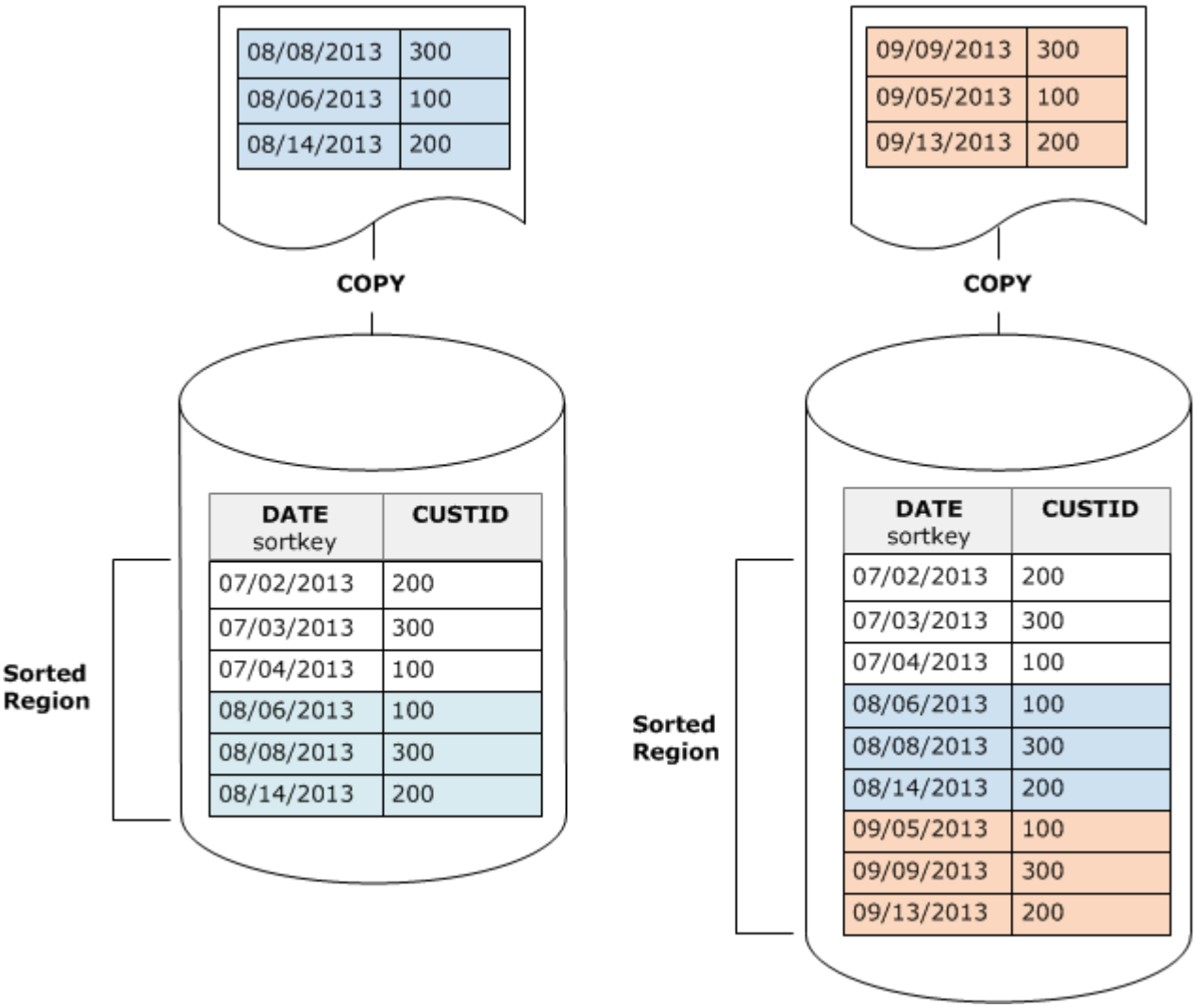
COPY fügt automatisch der sortierten Region der Tabelle neue Zeilen hinzu, wenn alle folgenden Bedingungen zutreffen:

- Die Tabelle verwendet einen zusammengesetzten Sortierschlüssel mit nur einer Sortierspalte.
- Die Sortierspalte ist NOT NULL.
- Die Tabelle ist zu 100 Prozent sortiert oder leer.
- Alle neuen Zeilen liegen in der Sortierreihenfolge höher als die vorhandenen Zeilen, einschließlich Zeilen, die zum Löschen markiert sind. In diesem Beispiel verwendet Amazon Redshift die ersten acht Bytes des Sortierschlüssels, um die Sortierreihenfolge festzulegen.

Angenommen, Sie verfügen beispielsweise über eine Tabelle, die Kundenveranstaltungen mittels einer Kunden-ID und eines Zeitpunkts aufzeichnet. Wenn Sie die Tabelle nach der Kunden-ID sortieren, überlappt der Sortierschlüsselbereich der neuen, durch inkrementelle Ladevorgänge

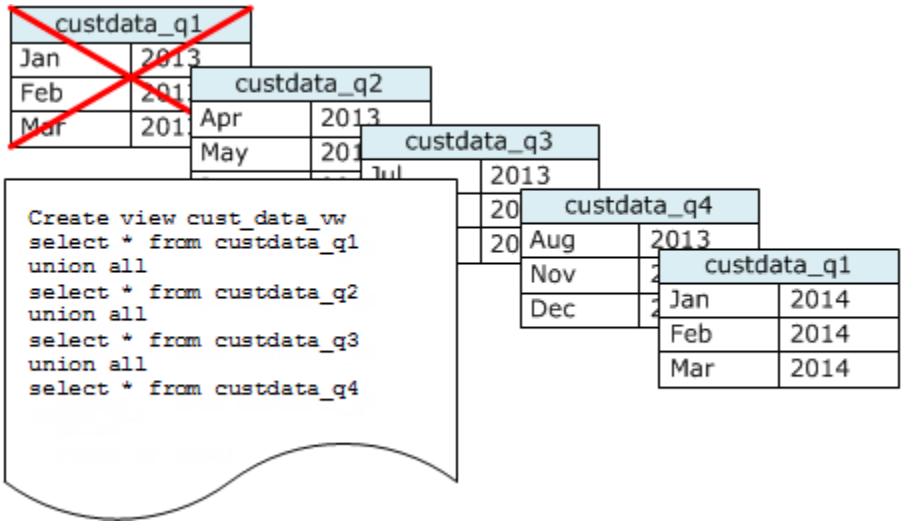
hinzugefügten Zeilen wahrscheinlich den vorhandenen Bereich, wie im vorherigen Beispiel gezeigt. Dies führt zu einer kostspieligen Bereinigungsoperation.

Wenn Sie den Sortierschlüssel auf eine Zeitstempelspalte festlegen, werden die neuen Zeilen in Sortierreihenfolge am Ende der Tabelle angefügt, wie im folgenden Diagramm dargestellt. Dies reduziert die Notwendigkeit einer Bereinigung oder macht diese sogar überflüssig.



Verwenden von Zeitreihentabellen

Wenn Sie Daten für einen rollierenden Zeitraum warten, verwenden Sie eine Reihe von Tabellen wie im folgenden Diagramm gezeigt.



Erstellen Sie jedes Mal, wenn Sie einen Satz von Daten hinzufügen, eine neue Tabelle. Löschen Sie anschließend die älteste Tabelle in der Reihe. Sie erzielen einen doppelten Vorteil:

- Sie vermeiden den zusätzlichen Aufwand für das Löschen von Zeilen, da die DROP TABLE-Operation sehr viel effizienter als eine DELETE-Massenoperation ist.
- Wenn die Tabellen nach Zeitstempel sortiert sind, wird keine Bereinigung benötigt. Wenn jede Tabelle die Daten für einen Monat enthält, muss eine Bereinigung höchstens die Daten eines Monats neu schreiben, auch wenn die Tabellen nicht nach Zeitstempel sortiert sind.

Sie können eine UNION ALL-Ansicht erstellen, die von Berichtsabfragen verwendet wird, die Tatsache verbirgt, dass die Dateien in mehreren Tabellen gespeichert sind. Wenn eine Abfrage nach dem Sortierschlüssel filtert, kann der Abfrageplaner effizient alle nicht verwendeten Tabellen überspringen. Eine UNION ALL-Ansicht kann für andere Arten von Abfragen weniger effizient sein. Daher sollten Sie die Abfrageleistung im Zusammenhang mit allen Abfragen bewerten, die die Tabellen verwenden.

Verwalten gleichzeitiger Schreiboperationen

Themen

Amazon Redshift ermöglicht das Lesen von Tabellen, während sie inkrementell geladen oder geändert werden.

In einigen herkömmlichen Data Warehousing- und Business Intelligence-Anwendungen ist die Datenbank nur dann für Benutzer verfügbar, wenn der nächtliche Ladevorgang abgeschlossen wurde. In diesen Fällen sind während der regelmäßigen Geschäftszeiten keine Updates zulässig, wenn analytische Abfragen ausgeführt und Berichte erstellt werden. Eine zunehmende Zahl von Anwendungen ist jedoch während langer Zeiträume am Tag oder sogar während des ganzen Tages aktiv, wodurch das Konzept eines Ladefensters obsolet wird.

Amazon Redshift unterstützt diese Arten von Anwendungen, da Tabellen gelesen werden können, während sie inkrementell geladen oder geändert werden. Abfragen steht einfach die letzte übergegebene Version bzw. der letzte Snapshot der Daten bereit. Sie müssen nicht darauf warten, dass für die nächste Version ein Commit ausgeführt wird. Wenn Sie möchten, dass eine bestimmte Abfrage auf einen Commit aus einer anderen Schreiboperation wartet, müssen Sie dies entsprechend planen.

In den folgenden Themen werden einige der wesentlichen Konzepte und Anwendungsfälle beschrieben, die Transaktionen, Datenbank-Snapshots, Updates und Gleichzeitigkeitsverhalten umfassen.

Serialisierbare Isolierung

Einige Anwendungen erfordern nicht nur gleichzeitiges Abfragen und Laden, sondern auch die Fähigkeit, gleichzeitig zu mehreren Tabellen oder zur selben Tabelle zu schreiben. In diesem Zusammenhang bedeutet gleichzeitig überlappend, nicht die Ausführung zu exakt demselben Zeitpunkt. Zwei Transaktionen werden als gleichzeitig betrachtet, wenn die zweite Transaktion gestartet wird, bevor für die erste Transaktion ein Commit ausgeführt wird. Gleichzeitige Operationen können ihren Ursprung in verschiedenen Sitzungen haben, die vom selben Benutzer oder von verschiedenen Benutzern kontrolliert werden.

> **Note**
>
> Amazon Redshift unterstützt standardmäßig automatische Commits, bei denen für jeden getrennt ausgeführten SQL-Befehl ein einzelnes Commit ausgeführt wird. Wenn Sie in einem Transaktionsblock einen Satz von Befehlen einschließen (definiert durch die Anweisungen BEGIN und END), wird der Block als eine einzige Transaktion übergeben, sodass Sie ein Rollback für ihn ausführen können, wenn notwendig. Ausnahmen von diesem Verhalten sind

die Befehle TRUNCATE und VACUUM, die automatische alle ausstehenden Änderungen, die in der aktuellen Transaktion vorgenommen wurden, übergeben.
Einige SQL-Clients geben automatisch BEGIN- und COMMIT-Befehle aus, sodass der Client steuert, ob eine Gruppe von Anweisungen als Transaktion oder jede einzelne Anweisung als eigene Transaktion ausgeführt wird. Konsultieren Sie die Dokumentation zu der von Ihnen verwendeten Schnittstelle. Wenn Sie zum Beispiel den JDBC-Treiber von Amazon Redshift verwenden, führt ein JDBC `PreparedStatement` mit einer Abfragezeichenfolge, die mehrere (durch Semikolon getrennte) SQL-Befehle enthält, alle Anweisungen als eine einzige Transaktion aus. Wenn Sie dagegen SQL Workbench/J verwenden und AUTO COMMIT ON festlegen, wird beim Ausführen mehrer Anweisungen jede Anweisung als eigene Transaktion ausgeführt.

Gleichzeitige Schreiboperationen werden in Amazon Redshift auf geschützte Art unterstützt, indem Schreibsperren für Tabellen und der Grundsatz der serialisierbaren Isolierung angewendet werden. Die serialisierbare Isolierung bewahrt die Illusion, dass eine Transaktion, die für eine Tabelle ausgeführt wird, die einzige Transaktion ist, die für diese Tabelle ausgeführt wird. Beispielsweise müssen zwei gleichzeitig ausgeführte Transaktionen T1 und T2 in mindestens einer der folgenden Varianten dieselben Ergebnisse produzieren:

- T1 und T2 werden seriell in dieser Reihenfolge ausgeführt.
- T2 und T1 werden seriell in dieser Reihenfolge ausgeführt.

Gleichzeitige Transaktionen können sich gegenseitig nicht sehen, d. h., sie können die Änderungen der jeweils anderen Transaktionen nicht erkennen. Jede gleichzeitige Transaktion erstellt zu Beginn der Transaktion einen Snapshot der Datenbank. Innerhalb der Transaktion wird beim ersten Auftreten der meisten SELECT-Anweisungen, von DML-Befehlen wie COPY, DELETE, INSERT, UPDATE und TRUNCATE und der folgenden DDL-Befehle ein Datenbank-Snapshot erstellt:

- ALTER TABLE (Hinzufügen oder Entfernen von Spalten)
- CREATE TABLE
- DROP TABLE
- TRUNCATE TABLE

Wenn eine beliebige serielle Ausführung der gleichzeitigen Transaktionen dieselben Ergebnisse wie ihre gleichzeitige Ausführung produzieren würde, gelten diese Transaktionen als „serialisierbar“ und

können sicher ausgeführt werden. Wenn keine serielle Ausführung dieser Transaktionen dieselben Ergebnisse produziert, wird die Transaktion abgebrochen, die eine Anweisung ausführt, die die Serialisierbarkeit verhindern würde. Für diese Transaktion wird ein Rollback ausgeführt.

Systemkatalogtabellen (PG) und andere Amazon-Redshift-Systemtabellen (STL und STV) sind in einer Transaktion nicht gesperrt. Daher sind Änderungen an Datenbankobjekten, die aufgrund von DDL- und TRUNCATE-Transaktionen entstanden sind, beim Commit für alle gleichzeitigen Transaktionen erkennbar.

Angenommen, Tabelle A ist in der Datenbank vorhanden, wenn zwei gleichzeitige Transaktionen, T1 und T2, gestartet werden. Angenommen, T2 gibt eine Liste von Tabellen zurück, indem Sie aus der Katalogtabelle PG_TABLES auswählen. Dann löscht T1 Tabelle A und Commits, und dann listet T2 die Tabellen erneut auf. Tabelle A ist jetzt nicht mehr aufgeführt. Wenn T2 versucht, eine Abfrage für die entfernte Tabelle auszuführen, gibt Amazon Redshift den Fehler „Beziehung nicht vorhanden“ zurück. Die Katalogabfrage, die die Liste von Tabellen an T2 zurückgibt oder überprüft, ob Tabelle A vorhanden ist, unterliegt nicht denselben Isolierungsregeln wie Operationen in Benutzertabellen.

Transaktionen für Aktualisierungen dieser Tabellen werden im Isolierungsmodus read committed ausgeführt. PG-prefix-Katalogtabellen unterstützen die Snapshot-Isolierung nicht.

Serialisierbare Isolierung für Systemtabellen und Katalogtabellen

Ein Datenbank-Snapshot wird auch in einer Transaktion für eine SELECT-Abfrage erstellt, die eine benutzererstellte Tabelle oder eine Amazon-Redshift-Systemtabelle (STL oder STV) referenziert. SELECT-Abfragen, die keine Tabellen referenzieren, erstellen keinen neuen Transaktionsdatenbank-Snapshot. Die Anweisungen INSERT, DELETE und UPDATE, die ausschließlich für Systemkatalogtabellen (PG) ausgeführt werden, erstellen ebenfalls keinen neuen Transaktionsdatenbank-Snapshot.

Beheben von Fehlern für die serialisierbare Isolierung

FEHLER: 1023 DETAIL: Serialisierbare Isolationsverletzung für eine Tabelle in Redshift

Wenn Amazon Redshift einen Fehler für die serialisierbare Isolierung erkennt, wird Ihnen eine Fehlermeldung ähnlich der folgenden Fehlermeldung angezeigt.

```
ERROR:1023 DETAIL: Serializable isolation violation on table in Redshift
```

Um einen Fehler für die serialisierbare Isolierung zu beheben, können Sie die folgenden Methoden anwenden:

- Wiederholen der abgebrochenen Transaktion.

  Amazon Redshift hat festgestellt, dass ein gleichzeitiger Workload nicht serialisierbar ist. Dies weist auf Lücken in der Anwendungslogik hin, die normalerweise bearbeitet werden können, indem die Transaktion erneut versucht wird, bei der der Fehler aufgetreten ist. Wenn das Problem weiterhin besteht, wenden Sie eine der anderen Methoden an.

- Verschieben Sie alle Operationen, die sich nicht in derselben atomaren Transaktion befinden müssen, aus der Transaktion.

  Diese Methode kann angewendet werden, wenn sich einzelne Operationen innerhalb von zwei Transaktion gegenseitig so referenzieren, dass das Ergebnis der jeweils anderen Transaktion beeinflusst werden kann. Die folgenden beiden Sitzungen starten beispielsweise jeweils eine Transaktion.

```
Session1_Redshift=# begin;
```

```
Session2_Redshift=# begin;
```

  Das Ergebnis einer SELECT-Anweisung in einer der beiden Transaktionen könnte durch eine INSERT-Anweisung in der jeweils anderen Transaktion beeinflusst werden. Angenommen, Sie führen die folgenden Anweisungen seriell in beliebiger Reihenfolge aus. In jedem Fall besteht das Ergebnis darin, dass eine der SELECT-Anweisungen eine Zeile mehr zurückgibt, als dies bei einer gleichzeitigen Ausführung der Transaktionen der Fall wäre. Es gibt keine Reihenfolge, in der die Operationen seriell ausgeführt werden können und zum gleichen Ergebnis wie bei einer gleichzeitigen Ausführung führen. Daher führt die zuletzt ausgeführte Operation zu einem Fehler für die serialisierbare Isolierung.

```
Session1_Redshift=# select * from tab1;
Session1_Redshift=# insert into tab2 values (1);
```

```
Session2_Redshift=# insert into tab1 values (1);
Session2_Redshift=# select * from tab2;
```

  In vielen Fällen ist das Ergebnis der SELECT-Anweisungen nicht wichtig. Mitanderen Worten: Die Atomizität der Operationen in den Transaktionen ist nicht wichtig. In diesen Fällen verschieben Sie die SELECT-Anweisungen aus den Transaktionen wie in den folgenden Beispielen gezeigt.

```
Session1_Redshift=# begin;
Session1_Redshift=# insert into tab1 values (1)
Session1_Redshift=# end;
Session1_Redshift=# select * from tab2;
```

```
Session2_Redshift # select * from tab1;
Session2_Redshift=# begin;
Session2_Redshift=# insert into tab2 values (1)
Session2_Redshift=# end;
```

In diesen Beispielen gibt es keine Querbezüge in den Transaktionen. Die beiden INSERT-Anweisungen beeinflussen sich nicht gegenseitig. In diesen Beispielen gibt es mindestens eine Reihenfolge, in der die Transaktionen seriell ausgeführt werden können und zum gleichen Ergebnis wie bei einer gleichzeitigen Ausführung führen. Das bedeutet, dass die Transaktionen serialisierbar sind.

- Sie können die Serialisierung erzwingen, indem Sie alle Tabellen in jeder Sitzung sperren.

  Der Befehl LOCK blockiert Operationen, die zu Fehlern für die serialisierbare Isolierung führen können. Wenn Sie den Befehl LOCK verwenden, müssen Sie Folgendes ausführen:

  - Sperren Sie alle Tabellen, die von der Transaktion betroffen sind, einschließlich Tabellen, die von schreibgeschützten SELECT-Anweisungen innerhalb der Transaktion betroffen sind.
  - Sperren Sie die Tabellen in derselben Reihenfolge, unabhängig von der Reihenfolge, in der die Operationen ausgeführt werden.
  - Sperren Sie alle Tabellen zu Beginn der Transaktion, bevor Operationen ausgeführt werden.
- Verwenden Sie die Snapshot-Isolation für gleichzeitige Transaktionen

  Verwenden Sie einen ALTER DATABASE-Befehl mit Snapshot-Isolation. Weitere Informationen zum SNAPSHOT-Parameter für ALTER DATABASE finden Sie unter Parameter.

FEHLER: 1018 DETAIL: Beziehung existiert nicht

Wenn Sie Amazon-Redshift-Vorgänge in verschiedenen Sitzungen ausführen, wird Ihnen eine Fehlermeldung ähnlich der folgenden Fehlermeldung angezeigt.

```
ERROR: 1018 DETAIL: Relation does not exist.
```

Transaktionen in Amazon Redshift folgen der Snapshot-Isolation. Nachdem eine Transaktion gestartet wurde, erstellt Amazon Redshift einen Snapshot der Datenbank. Während des gesamten Lebenszyklus der Transaktion arbeitet die Transaktion mit dem Status der Datenbank, wie sie im Snapshot wiedergegeben wird. Wenn die Transaktion aus einer Tabelle liest, die nicht im Snapshot vorhanden ist, wird die zuvor angezeigte Fehlermeldung 1018 ausgelöst. Selbst wenn eine andere gleichzeitige Transaktion eine Tabelle erstellt, nachdem die Transaktion den Snapshot erstellt hat, kann die Transaktion nicht aus der neu erstellten Tabelle lesen.

Um diesen Serialisierungsisolationsfehler zu beheben, können Sie versuchen, den Start der Transaktion an einen Punkt zu verschieben, an dem Sie wissen, dass die Tabelle existiert.

Wenn die Tabelle von einer anderen Transaktion erstellt wird, liegt dieser Punkt nach dem Commit dieser Transaktion. Stellen Sie außerdem sicher, dass gleichzeitig für keine Transaktion ein Commit durchgeführt wurde, das die Tabelle möglicherweise gelöscht hat.

```
session1 = # BEGIN;
session1 = # DROP TABLE A;
session1 = # COMMIT;
```

```
session2 = # BEGIN;
```

```
session3 = # BEGIN;
session3 = # CREATE TABLE A (id INT);
session3 = # COMMIT;
```

```
session2 = # SELECT * FROM A;
```

Daher führt die zuletzt von session2 als Lesevorgang ausgeführte Operation zu einem serialisierbaren Isolierungsfehler. Dieser Fehler tritt auf, wenn session2 einen Snapshot erstellt und die Tabelle bereits von einer festgeschriebenen session1 gelöscht wurde. Mit anderen Worten, obwohl eine gleichzeitige session3 die Tabelle erstellt hat, sieht session2 die Tabelle nicht, da sie sich nicht im Snapshot befindet.

Um diesen Fehler zu beheben, können Sie die Sitzungen wie folgt neu anordnen.

```
session1 = # BEGIN;
session1 = # DROP TABLE A;
session1 = # COMMIT;
```

```
session3 = # BEGIN;
session3 = # CREATE TABLE A (id INT);
session3 = # COMMIT;
```

```
session2 = # BEGIN;
session2 = # SELECT * FROM A;
```

Wenn nun session2 seinen Snapshot erstellt, wurde für session3 bereits ein Commit ausgeführt, und die Tabelle befindet sich in der Datenbank. Session2 kann fehlerfrei aus der Tabelle lesen.

Schreib- und Lese-Schreib-Operationen

Sie können das spezifische Verhalten gleichzeitiger Schreib- und Lese-Schreib-Operationen verwalten, indem Sie entscheiden, wann und wie verschiedene Arten von Befehlen ausgeführt werden. Die folgenden Befehle sind für dieses Thema relevant:

- COPY-Befehle, die Ladevorgänge ausführen (anfänglich oder inkrementell)
- INSERT-Befehle, die jeweils mindestens eine Zeile anfügen
- UPDATE-Befehle, die vorhandene Zeilen ändern
- DELETE-Befehle, die vorhandene Zeilen entfernen

COPY- und INSERT-Operationen sind reine Schreiboperationen. DELETE- und UPDATE-Operationen sind hingegen Lese-Schreib-Operationen. (Damit Zeilen gelöscht oder aktualisiert werden können, müssen sie zuerst gelesen werden.) Die Ergebnisse gleichzeitiger Schreibvorgänge sind von den spezifischen Befehlen abhängig, die gleichzeitig ausgeführt werden. COPY- und INSERT-Operationen für dieselbe Tabelle bleiben in einem Wartezustand, bis die Sperre aufgehoben wird. Anschließend werden sie wie normal verarbeitet.

UPDATE- und DELETE-Operationen verhalten sich anders, da sie von einem anfänglichen Tabellenlesevorgang abhängig sind, bevor Schreibvorgänge ausgeführt werden können. Da sich gleichzeitige Transaktionen gegenseitig nicht erkennen können, müssen UPDATE- und DELETE-Operationen einen Snapshot der Daten aus dem letzten Commit lesen. Wenn die erste UPDATE- oder DELETE-Operation die Sperre aufhebt, muss die zweite UPDATE- oder DELETE-Operation feststellen, ob die Daten, mit denen sie arbeitet, möglicherweise veraltet sind. Sie sind nicht veraltet, da die zweite Transaktion den Daten-Snapshot erst erhält, nachdem die erste Transaktion die Sperre aufgehoben hat.

Potenzielle Deadlock-Situation für gleichzeitige Schreibtransaktionen

Wenn Transaktionen Aktualisierungen von mehr als einer Tabelle umfassen, besteht stets die Möglichkeit, dass gleichzeitig ausgeführte Transaktionen in eine Deadlock-Situation geraten, wenn beide versuchen, zum gleichen Satz von Tabellen zu schreiben. Transaktionen heben alle ihre Tabellensperren auf einmal auf, wenn sie ein Commit oder ein Rollback ausführen. Sie heben Sperren nicht einzeln auf.

Angenommen, die Transaktionen T1 und T2 werden ungefähr zur gleichen Zeit gestartet. Wenn T1 einen Schreibvorgang für Tabelle A startet und T2 einen Schreibvorgang für Tabelle B startet, können beide Transaktionen konfliktfrei ausgeführt werden. Wenn T1 jedoch den Schreibvorgang für Tabelle A beendet und einen Schreibvorgang für Tabelle V starten muss, kann die Transaktion nicht fortgesetzt werden, da T2 Tabelle B noch nicht freigegeben hat. Wenn umgekehrt T2 den Schreibvorgang für Tabelle B beendet und einen Schreibvorgang für Tabelle A starten muss, kann die Transaktion auch nicht fortgesetzt werden, da T1 Tabelle A noch nicht freigegeben hat. Da keine der beiden Transaktionen ihre Sperren aufheben kann, solange nicht für alle ihre Schreibvorgänge ein Commit ausgeführt wurde, kann keine der beiden Transaktionen fortgesetzt werden.

Um diese Art von Deadlock zu vermeiden, müssen Sie gleichzeitige Schreibvorgänge sorgfältig planen. Beispielsweise sollten Sie in Transaktionen Tabellen stets in derselben Reihenfolge aktualisieren und bei Angabe von Sperren Tabellen auch in derselben Reihenfolge sperren, bevor Sie DML-Operationen ausführen.

Beispiele für gleichzeitige Schreibvorgänge

In den folgenden Pseudocode-Beispielen wird gezeigt, wie Transaktionen entweder fortgesetzt werden oder warten, wenn sie gleichzeitig ausgeführt werden.

Gleichzeitige COPY-Operationen für dieselbe Tabelle

Transaktion 1 kopiert Zeilen in die Tabelle LISTING:

```
begin;
copy listing from ...;
end;
```

Transaktion 2 wird gleichzeitig in einer getrennten Sitzung gestartet und versucht, mehr Zeilen in die Tabelle LISTING zu kopieren. Transaktion 2 muss warten, bis Transaktion 1 die Schreibsperre für die Tabelle LISTING aufhebt. Anschließend kann sie fortgesetzt werden.

```
begin;
[waits]
copy listing from ;
end;
```

Dasselbe Verhalten würde auftreten, wenn eine oder beide Transaktionen anstelle eines COPY-Befehls einen INSERT-Befehl enthalten.

Gleichzeitige DELETE-Operationen für dieselbe Tabelle

Transaktion 1 löscht Zeilen aus einer Tabelle:

```
begin;
delete from listing where ...;
end;
```

Transaktion 2 wird gleichzeitig gestartet und versucht, in derselben Tabelle Zeilen zu löschen. Die Transaktion ist erfolgreich, da sie auf den Abschluss von Transaktion 1 wartet, bevor sie versucht, Zeilen zu löschen.

```
begin
[waits]
delete from listing where ;
end;
```

Dasselbe Verhalten würde auftreten, wenn eine oder beide Transaktionen anstelle eines DELETE-Befehls einen UPDATE-Befehl für dieselbe Tabelle enthalten.

Gleichzeitige Transaktionen mit einer Mischung aus Lese- und Schreiboperationen

In diesem Beispiel löscht Transaktion 1 Zeilen aus der Tabelle USERS, lädt die Tabelle neu und führt eine COUNT(*)-Abfrage und anschließend ANALYZE aus, bevor sie einen Commit ausführt:

```
begin;
delete one row from USERS table;
copy ;
select count(*) from users;
analyze ;
end;
```

In der Zwischenzeit wird Transaktion 2 gestartet. Diese Transaktion versucht, zusätzliche Zeilen in die Tabelle USERS zu kopieren, die Tabelle zu analysieren und anschließend dieselbe COUNT(*)-Abfrage wie die erste Transaktion auszuführen:

```
begin;
[waits]
copy users from ...;
select count(*) from users;
analyze;
end;
```

Die zweite Transaktion ist erfolgreich, da sie auf den Abschluss der ersten Transaktion warten muss. Die COUNT-Abfrage gibt die Zahl basierend auf dem abgeschlossenen Ladevorgang zurück.

Tutorial: So laden Sie Daten aus Amazon S3

In diesem Tutorial durchlaufen Sie den kompletten Prozess des Ladens von Daten in Ihre Amazon-Redshift-Datenbanktabellen aus Datendateien in einem Amazon-S3-Bucket.

In diesem Tutorial führen Sie folgende Aufgaben aus:

- Laden Sie Datendateien als kommagetrennte Werte (CSV), als durch bestimmte Zeichen getrennte Werte und Formate mit fester Breite herunter.
- Erstellen eines Amazon-S3-Buckets und Upload der Datendateien zu diesem Bucket.
- Starten eines Amazon-Redshift-Clusters und Erstellen von Datenbanktabellen.
- Verwenden von COPY-Befehlen zum Laden der Tabellen aus den Datendateien auf Amazon S3.
- Beheben von Ladefehlern und Modifizieren Ihrer COPY-Befehle zur Behebung der Fehler.

Geschätzte Zeit: 60 Minuten

Geschätzte Kosten 1,00 USD pro Stunde für den Cluster

Voraussetzungen

Sie benötigen die folgenden Voraussetzungen:

- Ein AWS Konto zum Starten eines Amazon Redshift Redshift-Clusters und zum Erstellen eines Buckets in Amazon S3.

- Ihre AWS Anmeldeinformationen (IAM-Rolle) zum Laden von Testdaten aus Amazon S3. Wenn Sie eine neue IAM-Rolle benötigen, wechseln Sie zu Erstellen von IAM-Rollen.
- Ein SQL-Client, z. B. der Konsolenabfrage-Editor von Amazon Redshift.

Dieses Tutorial kann unabhängig von anderen absolviert werden. Zusätzlich zu diesem Tutorial empfehlen wir die folgenden Tutorials, um ein umfassenderes Verständnis vom Entwurf und von der Verwendung von Amazon-Redshift-Datenbanken zu erhalten:

- Das Handbuch Erste Schritte mit Amazon Redshift begleitet Sie beim Erstellen eines Amazon-Redshift-Clusters und dem Laden von Beispieldaten.

Übersicht

Sie können zum Hinzufügen von Daten zu Ihren Amazon-Redshift-Tabellen einen INSERT-Befehl oder einen COPY-Befehl verwenden. Der COPY-Befehl bietet den Umfang und die Geschwindigkeit eines Data Warehouse von Amazon Redshift und ist damit um ein Vielfaches schneller und effizienter als INSERT-Befehle.

Der COPY-Befehl nutzt die massive Parallelverarbeitungsarchitektur (Massively Parallel Processing, MPP) von Amazon Redshift, um Daten parallel aus mehreren Datenquellen zu lesen und zu laden. Sie können aus Datendateien in Amazon S3, Amazon EMR oder auf jedem Remote-Host laden, der über eine Secure Shell (SSH)-Verbindung erreichbar ist. Oder Sie können direkt aus einer Amazon-DynamoDB-Tabelle laden.

In diesem Tutorial verwenden Sie den Befehl COPY, um Daten aus Amazon S3 zu laden. Viele der hier vorgestellten Prinzipien gelten auch für das Laden aus anderen Datenquellen.

Weitere Informationen zur Verwendung des COPY-Befehls finden Sie in diesen Ressourcen:

- Bewährte Methoden für Amazon Redshift zum Laden von Daten
- So laden Sie Daten aus Amazon EMR:
- Laden von Daten aus Remote-Hosts
- Laden von Daten aus einer Amazon-DynamoDB-Tabelle

Schritte

- Schritt 1: Erstellen eines Clusters

Schritt 1: Erstellen eines Clusters

Wenn Sie bereits einen Cluster haben, den Sie verwenden möchten, können Sie diesen Schritt überspringen.

Verwenden Sie für die Übungen in diesem Tutorial einen Vierknoten-Cluster.

So erstellen Sie einen Cluster

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.

   Wählen Sie über das Navigationsmenü Dashboard für bereitgestellte Cluster aus.

   > **Important**
   >
   > Stellen Sie sicher, dass Sie über die erforderlichen Berechtigungen verfügen, um die Cluster-Operationen durchzuführen. Informationen zur Erteilung der erforderlichen Berechtigungen finden Sie unter Autorisieren von Amazon Redshift für den Zugriff auf AWS Dienste.

2. Wählen Sie oben rechts die AWS Region aus, in der Sie den Cluster erstellen möchten. Wählen Sie für die Zwecke dieses Tutorials USA West (Oregon) aus.
3. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Create cluster (Cluster erstellen) aus. Die Seite Create Cluster (Cluster erstellen) wird angezeigt.
4. Auf der Seite Cluster erstellen geben Sie die Parameter für Ihren Cluster ein. Wählen Sie Ihre eigenen Werte für die Parameter aus, außer die folgenden Werte zu ändern:

   - Wählen Sie **`dc2.large`** als Knotentyp.
   - Klicken Sie für die Anzahl der Knoten auf **`4`**.

- Wählen Sie im Abschnitt Cluster permissions (Clusterberechtigungen) eine IAM-Rolle aus Available IAM roles (Verfügbare IAM-Rollen) aus. Diese Rolle sollte eine sein, die Sie zuvor erstellt haben und die Zugriff auf Amazon S3 hat. Wählen Sie dann Associate IAM role (IAM-Rolle zuordnen) aus, um sie der Liste der Associated IAM roles (Zugeordneten IAM-Rollen) für den Cluster hinzuzufügen.

5. Wählen Sie Create cluster (Cluster erstellen).

Befolgen Sie die Schritte im Handbuch Erste Schritte mit Amazon Redshift zur Verbindung zu Ihrem Cluster von einem SQL-Client aus sowie zum Testen einer Verbindung. Sie müssen die verbleibenden Schritte aus "Erste Schritte" nicht durchführen, um Tabellen zu erstellen, Daten hochzuladen und Beispielabfragen auszuprobieren.

Nächster Schritt

Schritt 2: Herunterladen der Datendateien

Schritt 2: Herunterladen der Datendateien

In diesem Schritt laden Sie einen Satz Beispieldatendateien auf Ihren Computer herunter. Im nächsten Schritt laden Sie die Dateien in einen Amazon S3 Bucket hoch.

So laden Sie die Datendateien herunter:

1. Laden Sie die komprimierte Datei herunter: LoadingDataSampleFiles.zip.
2. Extrahieren Sie die Dateien in einen Ordner auf Ihrem Computer.
3. Prüfen Sie, ob Ihr Ordner die folgenden Dateien enthält.

```
customer-fw-manifest
customer-fw.tbl-000
customer-fw.tbl-000.bak
customer-fw.tbl-001
customer-fw.tbl-002
customer-fw.tbl-003
customer-fw.tbl-004
customer-fw.tbl-005
customer-fw.tbl-006
customer-fw.tbl-007
customer-fw.tbl.log
dwdate-tab.tbl-000
dwdate-tab.tbl-001
```

```
dwdate-tab.tbl-002
dwdate-tab.tbl-003
dwdate-tab.tbl-004
dwdate-tab.tbl-005
dwdate-tab.tbl-006
dwdate-tab.tbl-007
part-csv.tbl-000
part-csv.tbl-001
part-csv.tbl-002
part-csv.tbl-003
part-csv.tbl-004
part-csv.tbl-005
part-csv.tbl-006
part-csv.tbl-007
```

Nächster Schritt

Schritt 3: Hochladen der Datei in einen Amazon S3 Bucket

Schritt 3: Hochladen der Datei in einen Amazon S3 Bucket

In diesem Schritt erstellen Sie einen Amazon S3 Bucket und laden die Datendateien in diesen Bucket.

Hochladen der Dateien in einen Amazon S3 Bucket

1. Erstellen eines Buckets in Amazon S3.

   Weitere Informationen zum Erstellen eines Buckets finden Sie unter Erstellen von Buckets im Benutzerhandbuch für Amazon Simple Storage Service.

   a. Melden Sie sich bei der Amazon S3 S3-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/s3/.

   b. Wählen Sie Bucket erstellen aus.

   c. Wählen Sie eine AWS-Region.

      Erstellen Sie den Bucket in derselben Region, in der sich auch Ihr Cluster befindet. Wenn sich Ihr Cluster in der Region USA West (Oregon) befindet, wählen Sie USA West (Oregon) Region (us-west-2) aus.

d. Geben Sie im Feld Bucket-Name des Dialogfelds Bucket erstellen einen Bucket-Namen ein.

   Der von Ihnen gewählte Bucket-Name muss unter allen in Amazon S3 vorhandenen Bucket-Namen eindeutig sein. Eine Möglichkeit, für Eindeutigkeit zu sorgen, besteht darin, vor den Namen von Buckets den Namen Ihres Unternehmens zu setzen. Bucket-Namen müssen bestimmten Regeln folgen. Weitere Informationen finden Sie unter Bucket-Einschränkungen und -Limits im Benutzerhandbuch zu Amazon Simple Storage Service.

e. Wählen Sie die empfohlenen Standardwerte für die restlichen Optionen.

f. Wählen Sie Bucket erstellen aus.

   Wenn Amazon S3 Ihren Bucket erfolgreich erstellt hat, wird der leere Bucket in der Konsole im Feld Buckets angezeigt.

2. Erstellen Sie einen Ordner.

   a. Wählen Sie den Namen des neuen Buckets.

   b. Wählen Sie die Schaltfläche Ordner erstellen aus.

   c. Geben Sie als Namen für den neuen Ordner ein **`load`**.

   > **Note**
   >
   > Der von Ihnen erstellte Bucket befindet sich nicht in einer Sandbox. In dieser Übung fügen Sie Objekte zu einem echten Bucket hinzu. Für die Zeit, in der Sie die Objekte im Bucket speichern, wird Ihnen ein Nominalbetrag berechnet. Weitere Informationen zu Amazon-S3-Preisen finden Sie unter Amazon-S3-Preise.

3. Laden Sie die Datendateien in den neuen Amazon S3 Bucket.

   a. Wählen Sie auf den Namen des Datenordners aus.

   b. Wählen Sie im Assistenten für das Hochladen die Option Dateien hinzufügen.

      Befolgen Sie die Amazon-S3-Konsolenanweisungen, um alle heruntergeladenen und extrahierten Dateien hochzuladen.

   c. Klicken Sie auf Hochladen.

Benutzeranmeldeinformationen

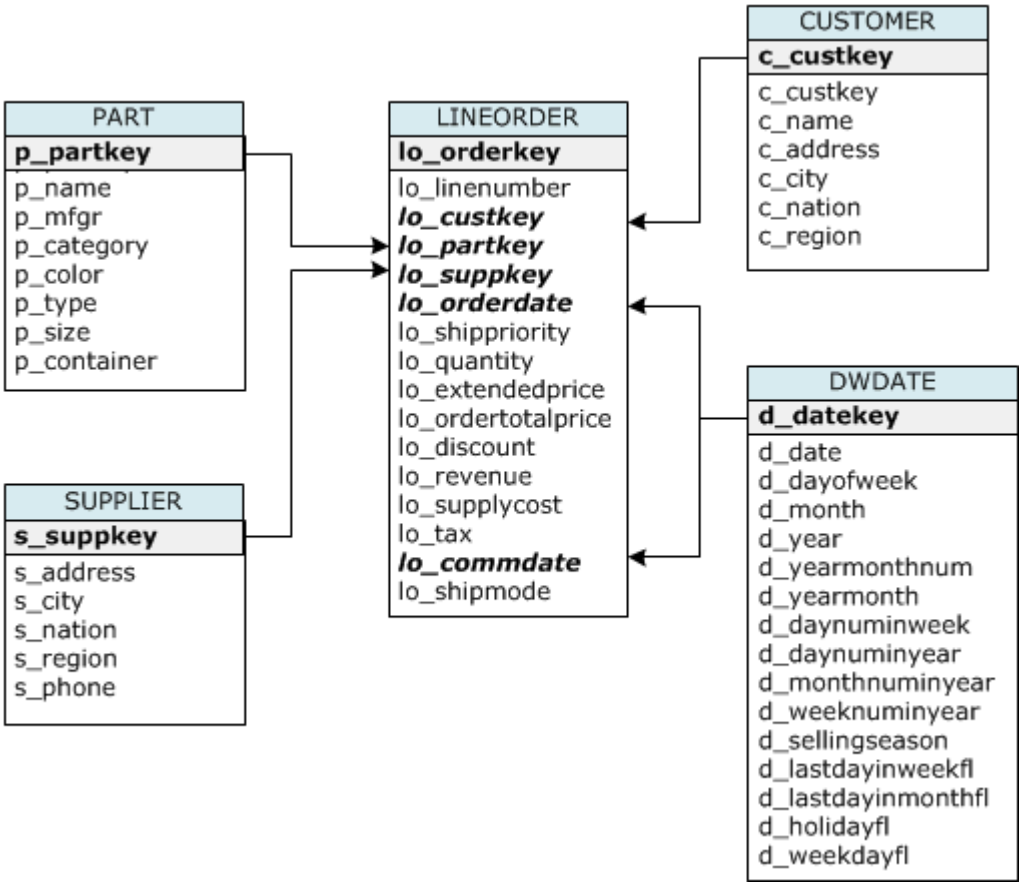
Der COPY-Befehl in Amazon Redshift muss einen Lesezugriff auf Dateiobjekte im Amazon-S3-Bucket besitzen. Wenn Sie dieselben Benutzer-Anmeldeinformationen verwenden, um den Amazon-S3-Bucket zu erstellen und den Amazon-Redshift-Befehl COPY auszuführen, verfügt der Befehl COPY über alle erforderlichen Berechtigungen. Wenn Sie andere Benutzeranmeldeinformationen verwenden möchten, können Sie den Zugriff über die Amazon-S3-Zugriffssteuerungselemente gewähren. Der Amazon Redshift COPY-Befehl erfordert mindestens ListBucket und GetObject Berechtigungen für den Zugriff auf die Dateiobjekte im Amazon S3 S3-Bucket. Weitere Informationen über Zugriffsrichtlinien für Amazon-S3-Ressourcen finden Sie unter Managing access permissions to your Amazon S3 resources (Verwaltung der Zugriffsberechtigungen zu Ihren Amazon-S3-Ressourcen).

Nächster Schritt

Schritt 4: Erstellen der Beispieltabellen

Schritt 4: Erstellen der Beispieltabellen

Für dieses Tutorial verwenden Sie einen Satz von fünf Tabellen, die auf dem Schema Star Schema Benchmark (SSB) basieren. Das folgende Diagramm zeigt das SSB-Datenmodell.



Die SSB-Tabellen sind möglicherweise bereits in der aktuellen Datenbank vorhanden. Wenn ja, führen Sie einen Drop für die Tabellen aus, um sie aus der Datenbank zu entfernen, bevor Sie sie im nächsten Schritt mit den Befehlen CREATE TABLE erstellen. Die in diesem Tutorial verwendeten Tabellen haben möglicherweise andere Attribute als die vorhandenen Tabellen.

So erstellen Sie die Beispieltabellen:

1. Um die SSB-Tabellen zu löschen, führen Sie die folgenden Befehle in Ihrem SQL-Client aus.

```
drop table part cascade;
drop table supplier;
drop table customer;
drop table dwdate;
drop table lineorder;
```

2. Führen Sie die folgenden CREATE TABLE-Befehle in Ihrem SQL-Client aus.

```
CREATE TABLE part
(
  p_partkey     INTEGER NOT NULL,
  p_name        VARCHAR(22) NOT NULL,
  p_mfgr        VARCHAR(6),
  p_category    VARCHAR(7) NOT NULL,
  p_brand1      VARCHAR(9) NOT NULL,
  p_color       VARCHAR(11) NOT NULL,
  p_type        VARCHAR(25) NOT NULL,
  p_size        INTEGER NOT NULL,
  p_container   VARCHAR(10) NOT NULL
);

CREATE TABLE supplier
(
  s_suppkey   INTEGER NOT NULL,
  s_name      VARCHAR(25) NOT NULL,
  s_address   VARCHAR(25) NOT NULL,
  s_city      VARCHAR(10) NOT NULL,
  s_nation    VARCHAR(15) NOT NULL,
  s_region    VARCHAR(12) NOT NULL,
  s_phone     VARCHAR(15) NOT NULL
);

CREATE TABLE customer
(
```

```
  c_custkey      INTEGER NOT NULL,
  c_name         VARCHAR(25) NOT NULL,
  c_address      VARCHAR(25) NOT NULL,
  c_city         VARCHAR(10) NOT NULL,
  c_nation       VARCHAR(15) NOT NULL,
  c_region       VARCHAR(12) NOT NULL,
  c_phone        VARCHAR(15) NOT NULL,
  c_mktsegment   VARCHAR(10) NOT NULL
);

CREATE TABLE dwdate
(
  d_datekey            INTEGER NOT NULL,
  d_date               VARCHAR(19) NOT NULL,
  d_dayofweek          VARCHAR(10) NOT NULL,
  d_month              VARCHAR(10) NOT NULL,
  d_year               INTEGER NOT NULL,
  d_yearmonthnum       INTEGER NOT NULL,
  d_yearmonth          VARCHAR(8) NOT NULL,
  d_daynuminweek       INTEGER NOT NULL,
  d_daynuminmonth      INTEGER NOT NULL,
  d_daynuminyear       INTEGER NOT NULL,
  d_monthnuminyear     INTEGER NOT NULL,
  d_weeknuminyear      INTEGER NOT NULL,
  d_sellingseason      VARCHAR(13) NOT NULL,
  d_lastdayinweekfl    VARCHAR(1) NOT NULL,
  d_lastdayinmonthfl   VARCHAR(1) NOT NULL,
  d_holidayfl          VARCHAR(1) NOT NULL,
  d_weekdayfl          VARCHAR(1) NOT NULL
);
CREATE TABLE lineorder
(
  lo_orderkey          INTEGER NOT NULL,
  lo_linenumber        INTEGER NOT NULL,
  lo_custkey           INTEGER NOT NULL,
  lo_partkey           INTEGER NOT NULL,
  lo_suppkey           INTEGER NOT NULL,
  lo_orderdate         INTEGER NOT NULL,
  lo_orderpriority     VARCHAR(15) NOT NULL,
  lo_shippriority      VARCHAR(1) NOT NULL,
  lo_quantity          INTEGER NOT NULL,
  lo_extendedprice     INTEGER NOT NULL,
  lo_ordertotalprice   INTEGER NOT NULL,
  lo_discount          INTEGER NOT NULL,
```

```
  lo_revenue           INTEGER NOT NULL,
  lo_supplycost        INTEGER NOT NULL,
  lo_tax               INTEGER NOT NULL,
  lo_commitdate        INTEGER NOT NULL,
  lo_shipmode          VARCHAR(10) NOT NULL
);
```

Nächster Schritt

Schritt 5: Ausführen der COPY-Befehle

Schritt 5: Ausführen der COPY-Befehle

Sie führen COPY-Befehle aus, um jede der Tabellen im SSB-Schema zu laden. Die Beispiele für COPY-Befehle demonstrieren das Laden aus unterschiedlichen Dateiformaten unter Verwendung verschiedener COPY-Befehlsoptionen sowie die Behebung von Ladefehlern.

Themen

- COPY-Befehlssyntax
- Laden der SSB-Tabellen

COPY-Befehlssyntax

Die grundlegende COPY-Befehlssyntax ist wie folgt.

```
COPY table_name [ column_list ] FROM data_source CREDENTIALS access_credentials
 [options]
```

Zur Ausführung eines COPY-Befehls geben Sie die folgenden Werte an.

Tabellenname

Die Zieltabelle für den COPY-Befehl. Die Tabelle muss in der Datenbank bereits vorhanden sein. Die Tabelle kann temporär oder persistent sein. Der COPY-Befehl fügt die neuen Eingabedaten den vorhandenen Zeilen in der Tabelle an.

Spaltenliste

Standardmäßig lädt COPY Felder aus den Quelldaten in ihrer Reihenfolge in die Tabellenspalten. Sie können optional eine Spaltenliste angeben, d. h. eine durch Kommas getrennte Liste von

Spaltennamen, um Datenfelder bestimmten Spalten zuzuordnen. In diesem Tutorial verwenden Sie keine Spaltenlisten. Weitere Informationen finden Sie unter Column List in der Referenz zum COPY-Befehl.

Datenquelle

Sie können den COPY-Befehl verwenden, um Daten aus einem Amazon-S3-Bucket, aus einem Amazon-EMR-Cluster, über eine SSH-Verbindung aus einem Remote-Host oder aus einer Amazon-DynamoDB-Tabelle zu laden. Für dieses Tutorial laden Sie Datendateien in einen Amazon-S3-Bucket. Beim Laden von Amazon S3 müssen Sie den Namen des Buckets und den Speicherort der Datendateien angeben. Dazu geben Sie entweder einen Objektpfad für die Datendateien oder den Speicherort einer Manifestdatei an, die jede Datendatei und ihren Speicherort explizit auflistet.

- Schlüsselpräfix

  Ein in Amazon S3 gespeichertes Objekt wird durch einen Objektschlüssel eindeutig definiert, der den Bucketnamen, eventuelle Ordnernamen sowie den Objektnamen enthält. Ein Schlüsselpräfix bezieht sich auf eine Reihe von Objekten, die das gleiche Präfix haben. Der Objektpfad ist ein Schlüsselpräfix, das der COPY-Befehl verwendet, um alle Objekte mit dem gleichen Schlüsselpräfix zu laden. Beispielsweise kann sich das Schlüsselpräfix `custdata.txt` auf eine einzelne Datei oder auf einen Satz von Dateien einschließlich `custdata.txt.001`, `custdata.txt.002` und so weiter beziehen.
- Manifestdatei

  In einigen Fällen müssen Sie möglicherweise Dateien mit unterschiedlichen Präfixen laden, z. B. aus mehreren Buckets oder Ordnern. In anderen Fällen müssen Sie möglicherweise Dateien ausschließen, die ein Präfix verwenden. In diesen Fällen können Sie eine Manifestdatei verwenden. Eine Manifestdatei führt alle Ladedateien und ihre eindeutigen Objektschlüssel explizit auf. Sie verwenden eine Manifestdatei, um die PART-Tabelle später in diesem Tutorial zu laden.

Anmeldeinformationen

Um auf die AWS Ressourcen zuzugreifen, die die zu ladenden Daten enthalten, müssen Sie die AWS Zugangsdaten für einen Benutzer mit ausreichenden Rechten angeben. Diese Anmeldeinformationen enthalten einen Amazon-Ressourcennamen (ARN) für die IAM-Rolle. Um Daten aus Amazon S3 zu laden, müssen die Anmeldeinformationen ListBucket und GetObject Berechtigungen enthalten. Weitere Anmeldeinformationen sind erforderlich, wenn Ihre Daten verschlüsselt sind. Weitere Informationen finden Sie unter Autorisierungsparameter in der Referenz zum COPY-Befehl. Weitere

Informationen zur Verwaltung des Zugriffs finden Sie unter Managing access permissions to your Amazon S3 resources (Verwaltung von Zugriffsberechtigungen für Ihre Amazon-S3-Ressourcen).

Optionen

Sie können mit dem COPY-Befehl eine Reihe von Parametern angeben, um Dateiformate anzugeben, Datenformate zu verwalten, mit Fehlern umzugehen und andere Features zu steuern. In diesem Tutorial verwenden Sie die folgenden COPY-Befehlsoptionen und -Funktionen:

- Schlüsselpräfix

  Informationen zum Laden von mehreren Dateien durch Angabe eines Schlüsselpräfixes finden Sie unter Laden der Tabelle PART mit NULL AS.
- CSV-Format

  Informationen zum Laden von Daten im CSV-Format finden Sie unter Laden der Tabelle PART mit NULL AS.
- NULL AS

  Hinweise zum Laden von PART mit der Option NULL AS finden Sie unter Laden der Tabelle PART mit NULL AS.
- Zeichengetrenntes Format

  Hinweise zur Verwendung der Option DELIMITER finden Sie unter Laden der Tabelle SUPPLIER mit REGION.
- REGION

  Informationen zur Verwendung der Option REGION finden Sie unter Laden der Tabelle SUPPLIER mit REGION.
- Breite des festen Formats

  Informationen zum Laden der Tabelle CUSTOMER aus Daten mit fester Breite finden Sie unter Laden der Tabelle CUSTOMER mit MANIFEST.
- MAXERROR

  Hinweise zur Verwendung der Option MAXERROR finden Sie unter Laden der Tabelle CUSTOMER mit MANIFEST.
- ACCEPTINVCHARS

Hinweise zur Verwendung der Option ACCEPTINVCHARS finden Sie unter Laden der Tabelle CUSTOMER mit MANIFEST.

- MANIFEST

  Hinweise zur Verwendung der Option MANIFEST finden Sie unter Laden der Tabelle CUSTOMER mit MANIFEST.

- DATEFORMAT

  Hinweise zur Verwendung der Option DATEFORMAT finden Sie unter Laden der Tabelle DWDATE mit DATEFORMAT.

- GZIP, LZOP und BZIP2

  Informationen zum Komprimieren von Dateien finden Sie unter Laden der Tabelle LINEORDER unter Verwendung mehrerer Tabellen.

- COMPUPDATE

  Hinweise zur Verwendung der Option COMPUPDATE finden Sie unter Laden der Tabelle LINEORDER unter Verwendung mehrerer Tabellen.

- Mehrere Dateien

  Informationen zum Laden mehrerer Dateien finden Sie unter Laden der Tabelle LINEORDER unter Verwendung mehrerer Tabellen.

Laden der SSB-Tabellen

Mit den folgenden COPY-Befehlen laden Sie jede der Tabellen im SSB-Schema. Der Befehl für jede Tabelle demonstriert unterschiedliche COPY-Optionen und Fehlerbehebungstechniken.

Gehen Sie zum Laden der SSB-Tabellen wie folgt vor:

1. Ersetzen Sie den Bucket-Namen und die AWS Anmeldeinformationen
2. Laden der Tabelle PART mit NULL AS
3. Laden der Tabelle SUPPLIER mit REGION
4. Laden der Tabelle CUSTOMER mit MANIFEST
5. Laden der Tabelle DWDATE mit DATEFORMAT
6. Laden der Tabelle LINEORDER unter Verwendung mehrerer Tabellen

Ersetzen Sie den Bucket-Namen und die AWS Anmeldeinformationen

Die COPY-befehle in diesem Tutorial werden im folgenden Format angegeben.

```
copy table from 's3://<your-bucket-name>/load/key_prefix'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
options;
```

Gehen Sie für jeden COPY-Befehl wie folgt vor:

1. Ersetzen Sie *<your-bucket-name>* durch den Namen eines Buckets in derselben Region, in der sich auch Ihr Cluster befindet.

   Für diesen Schritt wird davon ausgegangen, dass Bucket und Cluster sich in derselben Region befinden. Alternativ können Sie die Region mittels der Option REGION mit dem Befehl COPY angeben.
2. Ersetzen Sie <aws-account-id>und <role-name>durch Ihre eigene AWS-Konto und IAM-Rolle. Das Segment der Anmeldedatenzeichenfolge, das in einfachen Anführungszeichen eingeschlossen ist, darf keine Leerzeichen oder Zeilenumbrüche enthalten. Beachten Sie, dass sich das Format des ARNs geringfügig von dem im Beispiel verwendeten Format unterscheiden kann. Am besten kopieren Sie den ARN für die Rolle aus der IAM-Konsole, um sicherzustellen, dass er korrekt ist, wenn Sie die COPY-Befehle ausführen.

Laden der Tabelle PART mit NULL AS

In diesem Schritt verwenden Sie die Optionen CSV und NULL AS, um die PART-Tabelle zu laden.

Der COPY-Befehl kann Daten aus mehreren Dateien parallel laden, was viel schneller ist als das Laden aus einer einzelnen Datei. Zur Demonstration dieses Prinzips sind die Daten für jede Tabelle in diesem Tutorial in acht Dateien unterteilt, obwohl die Dateien sehr klein sind. In einem späteren Schritt vergleichen Sie den Zeitunterschied zwischen dem Laden aus einer einzelnen Datei und dem Laden aus mehreren Dateien. Weitere Informationen finden Sie unter Laden von Datendateien.

Schlüsselpräfix

Sie können aus mehreren Dateien durch die Angabe eines Schlüsselpräfixes den Dateiensatz oder durch die explizite Auflistung der Dateien in einer Manifestdatei laden. In diesem Schritt verwenden Sie ein Schlüsselpräfix. In einem späteren Schritt verwenden Sie eine Manifestdatei. Das Schlüsselpräfix `'s3://mybucket/load/part-csv.tbl'` lädt den folgenden Satz der Dateien im Ordner `load`.

```
part-csv.tbl-000
part-csv.tbl-001
part-csv.tbl-002
part-csv.tbl-003
part-csv.tbl-004
part-csv.tbl-005
part-csv.tbl-006
part-csv.tbl-007
```

CSV-Format

CSV (Comma Separated Values) ist ein für den Import und den Export von Spreadsheet-Daten häufig verwendetes Format. CSV ist flexibler als das Comma-Delimited Format, da es den Einschluss von Zeichenfolgen mit Anführungszeichen in Feldern erlaubt. Das Standard-Anführungszeichen für COPY aus dem CSV-Format ist das doppelte Anführungszeichen ("), Sie können aber mit der Option QUOTE AS ein anderes Zeichen angeben. Wenn Sie das Anführungszeichen innerhalb des Feldes verwenden, verwenden Sie ein weiteres Anführungszeichen als Escape-Zeichen.

Der folgende Auszug aus einer CSV-formatierten Datendatei für die PART-Tabelle zeigt Zeichenfolgen, die in doppelte Anführungszeichen () eingeschlossen sind. (`"LARGE ANODIZED BRASS"`). Er zeigt außerdem eine Zeichenfolge, die in zwei doppelte Anführungszeichen innerhalb einer Zeichenfolge mit Anführungszeichen eingeschlossen () ist. (`"MEDIUM ""BURNISHED"" TIN"`).

```
15,dark sky,MFGR#3,MFGR#47,MFGR#3438,indigo,"LARGE ANODIZED BRASS",45,LG CASE
22,floral beige,MFGR#4,MFGR#44,MFGR#4421,medium,"PROMO, POLISHED BRASS",19,LG DRUM
23,bisque slate,MFGR#4,MFGR#41,MFGR#4137,firebrick,"MEDIUM ""BURNISHED"" TIN",42,JUMBO
 JAR
```

Die Daten für die PART-Tabelle enthalten Zeichen, die dazu führen, dass COPY fehlschlägt. In dieser Übung finden Sie die Fehler und beheben sie.

Um Daten im CSV-Format zu laden, fügen Sie Ihrem COPY-Befehl `csv` hinzu. Führen Sie den folgenden Befehl aus, um die PART-Tabelle zu laden.

```
copy part from 's3://<your-bucket-name>/load/part-csv.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
csv;
```

Sie könnten eine Fehlermeldung ähnlich der folgenden erhalten.

```
An error occurred when executing the SQL command:
copy part from 's3://mybucket/load/part-csv.tbl'
credentials' ...

ERROR: Load into table 'part' failed.  Check 'stl_load_errors' system table for
 details. [SQL State=XX000]

Execution time: 1.46s

1 statement(s) failed.
1 statement(s) failed.
```

Um mehr Informationen zu dem Fehler zu erhalten, fragen Sie die Tabelle STL_LOAD_ERRORS ab. Die folgende Abfrage verwendet die Funktion SUBSTRING zum Kürzen von Spalten zur besseren Lesbarkeit sowie LIMIT 10, um die Anzahl der ausgegebenen Zeilen zu reduzieren. Sie können die Werte in `substring(filename,22,25)` an die Länge Ihres Bucketnamens anpassen.

```
select query, substring(filename,22,25) as filename,line_number as line,
substring(colname,0,12) as column, type, position as pos, substring(raw_line,0,30) as
 line_text,
substring(raw_field_value,0,15) as field_text,
substring(err_reason,0,45) as reason
from stl_load_errors
order by query desc
limit 10;
```

```
 query  |    filename      | line |  column   |    type    | pos |
--------+--------------------------+-----------+------------+------------+-----+----
 333765 | part-csv.tbl-000 |    1 |           |            |   0 |

 line_text        | field_text |                   reason
------------------+------------+-----------------------------------------------
 15,NUL next,     |            | Missing newline: Unexpected character 0x2c f
```

NULL AS

Die `part-csv.tbl`-Datendateien verwenden das NUL-Begrenzungszeichen (`\x000` oder `\x0`) zur Anzeige von NULL-Werten.

> **Note**
>
> Trotz ihrer sehr ähnlichen Schreibweise sind NUL und NULL nicht identisch. NUL ist ein UTF-8-Zeichen mit Codepunkt `x000`, das oft zur Kennzeichnen des Datensatzendes (End of Record, EOR) verwendet wird. NULL ist ein SQL-Wert, der für die Abwesenheit von Daten steht.

Standardmäßig behandelt COPY das Begrenzungszeichen NUL als EOR-Zeichen und beendet den Datensatz, was oft zu unerwarteten Ergebnissen oder einem Fehler führt. Es gibt keine einzige Standardmethode, um NULL in Textdaten anzuzeigen. Mit der Befehlsoption NULL AS COPY können Sie also angeben, welches Zeichen beim Laden der Tabelle durch NULL ersetzt werden soll. In diesem Beispiel soll COPY das NUL-Begrenzungszeichen als NULL-Wert behandeln.

> **Note**
>
> Die Tabellenspalte, die den NULL-Wert erhält, muss als nullfähig konfiguriert sein. Das bedeutet, dass sie die NOT NULL-Einschränkung in der CREATE TABLE-Spezifikation nicht enthalten darf.

Führen Sie den folgenden COPY-Befehl aus, um PART mit der Option NULL AS zu laden.

```
copy part from 's3://<your-bucket-name>/load/part-csv.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
csv
null as '\000';
```

Um zu prüfen, ob COPY NULL-Werte geladen hat, führen Sie den folgenden Befehl aus, um nur die Zeilen auszuwählen, die NULL enthalten.

```
select p_partkey, p_name, p_mfgr, p_category from part where p_mfgr is null;
```

```
 p_partkey |  p_name  | p_mfgr | p_category
-----------+----------+--------+------------
        15 | NUL next |        | MFGR#47
        81 | NUL next |        | MFGR#23
       133 | NUL next |        | MFGR#44
```

```
(2 rows)
```

Laden der Tabelle SUPPLIER mit REGION

In diesem Schritt verwenden Sie die Optionen DELIMITER und REGION, um die Tabelle SUPPLIER zu laden.

> **Note**
>
> Die Dateien zum Laden der SUPPLIER-Tabelle befinden sich in einem AWS Muster-Bucket. Für diesen Schritt müssen Sie keine Daten hochladen.

Zeichengetrenntes Format

Die Felder in einer zeichengetrennten Datei werden von einem speziellen Teichen, etwa einem senkrechten Strich (|), einem Komma (,) oder einem Tabulatorzeichen (\t) abgetrennt. Dafür können alle einzelnen ASCII-Zeichen verwendet werden, auch nicht gedruckte ASCII-Zeichen. Sie geben das Trennzeichen mit der Option DELIMITER an. Das Standardtrennzeichen ist der senkrechte Strich (|).

Der folgende Auszug aus den Daten für die Tabelle SUPPLIER verwendet den senkrechten Strich als Trennzeichen.

```
1|1|257368|465569|41365|19950218|2-HIGH|0|17|2608718|9783671|4|2504369|92072|2|
19950331|TRUCK
1|2|257368|201928|8146|19950218|2-HIGH|0|36|6587676|9783671|9|5994785|109794|6|
19950416|MAIL
```

REGION

Wann immer möglich, sollten Sie Ihre Ladedaten in derselben AWS Region wie Ihr Amazon Redshift Redshift-Cluster lokalisieren. Wenn sich Ihre Daten und Ihr Cluster in derselben Region befinden, reduzieren Sie die Latenz und vermeiden Kosten für den regionenübergreifenden Datentransfer. Weitere Informationen finden Sie unter Bewährte Methoden für Amazon Redshift zum Laden von Daten

Wenn Sie Daten aus einer anderen AWS Region laden müssen, verwenden Sie die Option REGION, um die AWS Region anzugeben, in der sich die Ladedaten befinden. Wenn Sie eine Region angeben, müssen sich alle Ladedaten, einschließlich der Manifestdateien, in der benannten Region befinden. Weitere Informationen finden Sie unter REGION.

Wenn sich Ihr Cluster in der Region USA Ost (Nord-Virginia) befindet, führen Sie den folgenden Befehl aus, um die Tabelle SUPPLIER aus mit | getrennten Daten in einem Amazon S3 Bucket in der Region USA West (Oregon) zu laden. Ändern Sie für dieses Beispiel den Namen des Buckets nicht.

```
copy supplier from 's3://awssampledbuswest2/ssbgz/supplier.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
delimiter '|'
gzip
region 'us-west-2';
```

Wenn sich Ihr Cluster nicht in der Region USA Ost (Nord-Virginia) befindet, führen Sie den folgenden Befehl aus, um die Tabelle SUPPLIER aus durch | getrennten Daten in einem Amazon S3 Bucket zu laden, der sich in der Region USA Ost (Nord-Virginia) befindet. Ändern Sie für dieses Beispiel den Namen des Buckets nicht.

```
copy supplier from 's3://awssampledb/ssbgz/supplier.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
delimiter '|'
gzip
region 'us-east-1';
```

Laden der Tabelle CUSTOMER mit MANIFEST

In diesem Schritt laden Sie die Tabelle CUSTOMER mit den Optionen FIXEDWIDTH, MAXERROR, ACCEPTINVCHARS und MANIFEST.

Die Beispieldaten für diese Übung enthalten Zeichen, die beim Laden durch COPY Fehler verursachen. Mit der Option MAXERRORS und der Systemtabelle STL_LOAD_ERRORS können Sie die Ladefehler ausfindig machen und mit den Optionen ACCEPTINVCHARS und MANIFEST die Fehler beheben.

Format mit fester Breite

Das Format mit fester Breite definiert jedes Feld mit einer festen Anzahl von Zeichen, anstatt die Felder durch Trennzeichen voneinander zu scheiden. Der folgende Auszug aus den Daten für die Tabelle CUSTOMER verwendet das Format mit fester Breite.

```
1   Customer#000000001   IVhzIApeRb            MOROCCO  0MOROCCO  AFRICA      25-705
2   Customer#000000002   XSTf4,NCwDVaWNe6tE    JORDAN   6JORDAN   MIDDLE EAST 23-453
3   Customer#000000003   MG9kdTD               ARGENTINA5ARGENTINAAMERICA     11-783
```

Die Reihenfolge der Paare aus Bezeichnung/Breite muss exakt mit der Reihenfolge der Tabellenspalten übereinstimmen. Weitere Informationen finden Sie unter FIXEDWIDTH.

Die Spezifikationszeichenfolge für die feste Breite der Daten für die Tabelle CUSTOMER ist die folgende.

```
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
c_region :12, c_phone:15,c_mktsegment:10'
```

Um die Tabelle CUSTOMER aus Daten mit fester Breite zu laden, führen Sie den folgenden Befehl aus.

```
copy customer
from 's3://<your-bucket-name>/load/customer-fw.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10';
```

Sie sollten eine Fehlermeldung ähnlich der folgenden erhalten.

```
An error occurred when executing the SQL command:
copy customer
from 's3://mybucket/load/customer-fw.tbl'
credentials'...

ERROR: Load into table 'customer' failed.  Check 'stl_load_errors' system table for
 details. [SQL State=XX000]

Execution time: 2.95s

1 statement(s) failed.
```

MAXERROR

Standardmäßig schlägt der COPY-Befehl beim ersten Auftreten eines Fehlers fehl und gibt eine Fehlermeldung aus. Um beim testen Zeit zu sparen, können Sie die Option MAXERROR verwenden, damit COPY eine bestimmte Anzahl von Fehlern übergeht, bevor der Befehl fehlschlägt. Da wir beim ersten Test des Ladens der Daten der Tabelle CUSTOMER Fehler erwarten, fügen Sie dem COPY-Befehl `maxerror 10` hinzu.

Führen Sie zum Test mit den Optionen FIXEDWIDTH und MAXERROR den folgenden Befehl aus.

```
copy customer
from 's3://<your-bucket-name>/load/customer-fw.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10'
maxerror 10;
```

Diesmal erhalten Sie statt einer Fehlermeldung eine Warnmeldung, ähnlich der folgenden.

```
Warnings:
Load into table 'customer' completed, 112497 record(s) loaded successfully.
Load into table 'customer' completed, 7 record(s) could not be loaded.  Check
 'stl_load_errors' system table for details.
```

Die Warnung gibt an, dass COPY auf sieben Fehler gestoßen ist. Fragen Sie zur Prüfung der Fehler die Tabelle STL_LOAD_ERRORS ab, wie im folgenden Beispiel gezeigt.

```
select query, substring(filename,22,25) as filename,line_number as line,
substring(colname,0,12) as column, type, position as pos, substring(raw_line,0,30) as
 line_text,
substring(raw_field_value,0,15) as field_text,
substring(err_reason,0,45) as error_reason
from stl_load_errors
order by query desc, filename
limit 7;
```

Das Ergebnis der Abfrage von STL_LOAD_ERRORS sollte ähnlich wie folgt aussehen.

```
 query  |         filename          | line |  column   |    type    | pos |
 line_text           | field_text |              error_reason
--------+---------------------------+------+-----------
+------------+-----+-------------------------------+------------
+--------------------------------------------
 334489 | customer-fw.tbl.log       |    2 | c_custkey | int4       |  -1 | customer-
fw.tbl              | customer-f | Invalid digit, Value 'c', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    6 | c_custkey | int4       |  -1 | Complete
                   | Complete   | Invalid digit, Value 'C', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    3 | c_custkey | int4       |  -1 | #Total rows
                   | #Total row | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    5 | c_custkey | int4       |  -1 | #Status
                   | #Status    | Invalid digit, Value '#', Pos 0, Type: Integ
```

```
 334489 | customer-fw.tbl.log       |    1 | c_custkey | int4       |  -1 | #Load file
                  | #Load file | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl000        |    1 | c_address | varchar    |  34 | 1
 Customer#000000001  | .Mayag.ezR | String contains invalid or unsupported UTF8
 334489 | customer-fw.tbl000        |    1 | c_address | varchar    |  34 | 1
 Customer#000000001  | .Mayag.ezR | String contains invalid or unsupported UTF8
(7 rows)
```

Bei der Untersuchung der Ergebnisse sehen Sie zwei Meldungen in der Spalte `error_reasons`:

- ```
  Invalid digit, Value '#', Pos 0, Type: Integ
  ```

  Diese Fehler wurden von der Datei `customer-fw.tbl.log` verursacht. Das Problem besteht darin, dass es sich um eine Protokolldatei und nicht um eine Datendatei handelt, die daher nicht geladen werden sollte. Sie können eine Manifestdatei verwenden, um das Laden falscher Dateien zu vermeiden.

- ```
  String contains invalid or unsupported UTF8
  ```

  Der Datentyp VARCHAR unterstützt Multibyte-UTF-8-Zeichen mit bis zu drei Byte. Wenn die Ladedaten nicht unterstützte oder ungültige Zeichen enthalten, können Sie jedes ungültige Zeichen mit der Option ACCEPTINVCHARS gegen ein angegebenes Alternativzeichen austauschen.

Ein weiteres Problem mit der Last ist schwieriger zu erkennen – die Last führte zu unerwarteten Ergebnissen. Fragen Sie mit dem folgenden Befehl die Tabelle CUSTOMER ab, um dieses Problem zu untersuchen.

```
select c_custkey, c_name, c_address
from customer
order by c_custkey
limit 10;
```

```
 c_custkey |          c_name           |         c_address
-----------+---------------------------+---------------------------
         2 | Customer#000000002        | XSTf4,NCwDVaWNe6tE
         2 | Customer#000000002        | XSTf4,NCwDVaWNe6tE
         3 | Customer#000000003        | MG9kdTD
         3 | Customer#000000003        | MG9kdTD
         4 | Customer#000000004        | XxVSJsL
```

```
         4 | Customer#000000004        | XxVSJsL
         5 | Customer#000000005        | KvpyuHCplrB84WgAi
         5 | Customer#000000005        | KvpyuHCplrB84WgAi
         6 | Customer#000000006        | sKZz0CsnMD7mp4Xd0YrBvx
         6 | Customer#000000006        | sKZz0CsnMD7mp4Xd0YrBvx
(10 rows)
```

Die Zeilen sollten eindeutig sein, es gibt jedoch Duplikate.

Eine weitere Möglichkeit zur Untersuchung unerwarteter Ergebnisse besteht darin, die Anzahl der geladenen Zeilen zu prüfen. In unserem Fall sollten 100000 Zeilen geladen werden, die Lademeldung gab 112497 geladene Datensätze an. Die zusätzlichen Zeilen wurden geladen, weil COPY eine überzählige Datei, , geladen hat., `customer-fw.tbl0000.bak`.

In dieser Übung verwenden Sie eine Manifestdatei, um das Laden der falschen Dateien zu vermeiden.

ACCEPTINVCHARS

Standardmäßig gilt, dass COPY beim Treffen auf eine von dem Datentyp der Spalte nicht unterstütztes Zeichen die Zeile überspringt und einen Fehler ausgibt. Informationen zu ungültigen UTF-8-Zeichen finden Sie unter Fehler beim Laden von Multibyte-Zeichen.

Sie können die Option MAXERRORS verwenden, um Fehler zu ignorieren und den Ladevorgang fortzusetzen, dann STL_LOAD_ERRORS abfragen, um die ungültigen Zeichen zu finden und anschließend die Datendateien korrigieren. MAXERRORS wird jedoch sinnvollerweise für die Behebung von Ladeproblemen verwendet und sollte generell nicht in einer Produktionsumgebung genutzt werden.

Die Option ACCEPTINVCHARS ist normalerweise die bessere Wahl für den Umgang mit ungültigen Zeichen. ACCEPTINVCHARS instruiert COPY dazu, jedes ungültige Zeichen durch ein festgelegtes gültiges Zeichen zu ersetzen und den Ladevorgang fortzusetzen. Sie können jedes gültige ASCII-Zeichen, ausgenommen NULL, als Austauschzeichen festlegen. Das Standard-Austauschzeichen ist ein Fragezeichen (?). COPY ersetzt Multibyte-Zeichen durch eine Austauschzeichenfolge gleicher Länge. Ein 4-Byte-Zeichen wird etwa durch ersetzt `'????'`.

COPY gibt die Anzahl der Zeilen zurück, die ungültige UTF-8-Zeichen enthielten. Außerdem wird für jede betroffene Zeile ein Eintrag in die Systemtabelle STL_REPLACEMENTS hinzugefügt (bis zu maximal 100 Zeilen pro Knotenebene). Zusätzliche ungültige UTF-8-Zeichen werden ebenfalls ersetzt. Diese Ersetzungsereignisse werden jedoch nicht aufgezeichnet.

ACCEPTINVCHARS ist nur für VARCHAR-Spalten gültig.

Für diesen Schritt fügen Sie die ACCEPTINVCHARS mit dem Ersetzungszeichen `'^'` hinzu.

MANIFEST

Wenn Sie von Amazon S3 mit einem Schlüsselpräfix kopieren, besteht die Gefahr, dass Sie unerwünschte Tabellen laden. Beispielsweise enthält der Ordner `'s3://mybucket/load/` acht Datendateien, die das Schlüsselpräfix `customer-fw.tbl` gemeinsam haben: `customer-fw.tbl0000`, `customer-fw.tbl0001` usw. Dieser Ordner enthält aber auch die überschüssigen Dateien `customer-fw.tbl.log` und `customer-fw.tbl-0001.bak`.

Um sicherzustellen, dass Sie alle und nur die korrekten Dateien laden, verwenden Sie eine Manifestdatei. Das Manifest ist eine Textdatei im JSON-Format, die den eindeutigen Objektschlüssel für jede zu ladende Quelldatei ausdrücklich aufführt. Die Dateiobjekte können sich in verschiedenen Ordnern oder Buckets, müssen sich aber in derselben Region befinden. Weitere Informationen finden Sie unter MANIFEST.

Nachfolgend sehen Sie den `customer-fw-manifest`-Text.

```
{
  "entries": [
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-000"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-001"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-002"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-003"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-004"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-005"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-006"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-007"}
    ]
}
```

So laden Sie die Daten für die Tabelle CUSTOMER mit der Manifestdatei:

1. Öffnen Sie die Datei `customer-fw-manifest` in einem Text-Editor.
2. Ersetzen Sie *<your-bucket-name>* durch den Namen Ihres Buckets.
3. Speichern Sie die Datei.
4. Laden Sie die Datei in den Ladeordner in Ihrem Bucket.
5. Führen Sie den folgenden COPY-Befehl aus.

```
copy customer from 's3://<your-bucket-name>/load/customer-fw-manifest'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10'
maxerror 10
acceptinvchars as '^'
manifest;
```

Laden der Tabelle DWDATE mit DATEFORMAT

In diesem Schritt verwenden Sie die Optionen DELIMITER und DATEFORMAT, um die Tabelle DWDATE zu laden.

Beim Laden der Spalten DATE und TIMESTAMP erwartet COPY das Standardformat, s. h. JJJJ-MM-TT für Datumsangaben und JJJJ-MM-TT-HH:MI:SS für Zeitstempel. Wenn die Ladedaten das Standardformat nicht verwenden, können Sie das Format mithilfe von DATEFORMAT und TIMEFORMAT angeben.

Der folgende Auszug zeigt Datumsformate in der Tabelle DWDATE. Beachten Sie, dass die Datumsformate in Spalte zwei nicht konsistent sind.

```
19920104 1992-01-04          Sunday  January 1992 199201 Jan1992 1 4 4 1...
19920112 January 12, 1992 Monday  January 1992 199201 Jan1992 2 12 12 1...
19920120 January 20, 1992 Tuesday     January 1992 199201 Jan1992 3 20 20 1...
```

DATEFORMAT

Sie können nur ein Datumsformat angeben. Wenn die Ladedaten inkonsistente Formate enthalten, möglicherweise in verschiedenen Spalten, oder wenn das Format zum Zeitpunkt des Ladevorgangs nicht bekannt ist, verwenden Sie DATEFORMAT mit dem Argument `'auto'`. Wenn `'auto'` angegeben ist, erkennt COPY jedes gültige Datums- oder Zeitformat und konvertiert es in das Standardformat. Die Option `'auto'` erkennt verschiedene Formate, die bei der Verwendung einer DATEFORMAT- und TIMEFORMAT-Zeichenfolge nicht unterstützt werden. Weitere Informationen finden Sie unter Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT.

Führen Sie den folgenden COPY-Befehl aus, um die Tabelle DWDATE zu laden.

```
copy dwdate from 's3://<your-bucket-name>/load/dwdate-tab.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
```

```
delimiter '\t'
dateformat 'auto';
```

Laden der Tabelle LINEORDER unter Verwendung mehrerer Tabellen

Dieser Schritt verwendet die Optionen GZIP und COMPUPDATE zum Laden der Tabelle LINEORDER.

In dieser Übung laden Sie die Tabelle LINEORDER aus einer einzigen Datendatei und laden sie dann erneut aus mehreren Dateien. Auf diese Weise können Sie die Ladezeiten der beiden Methoden vergleichen.

> **Note**
>
> Die Dateien zum Laden der LINEORDER-Tabelle werden in einem AWS Beispiel-Bucket bereitgestellt. Für diesen Schritt müssen Sie keine Daten hochladen.

GZIP, LZOP und BZIP2

Sie können Ihre Dateien mit den Kompressionsformaten gzip, lzop und bzip2 komprimieren. Beim laden aus komprimierten Dateien dekomprimiert COPY diese während des Ladevorgangs. Die Komprimierung Ihrer Dateien spart Speicherplatz und verkürzt die Ladezeiten.

COMPUPDATE

Wenn COPY eine leere Tabelle ohne Kompressionskodierungen lädt, analysiert der Befehl die Ladedaten, um die optimalen Kodierungen zu bestimmen. Anschließend ändert er die Tabelle, um diese Kodierungen vor dem beginn des Ladevorgangs anzuwenden. Diese Analyse nimmt Zeit in Anspruch, findet aber höchstens einmal pro Tabelle statt. Um Zeit zu sparen, können Sie diesen Schritt überspringen, indem Sie COMPUPDATE ausschalten. Um eine genaue Auswertung der COPY-Zeiten zu ermöglichen, schalten Sie COMPUPDATE für diesen Schritt aus.

Mehrere Dateien

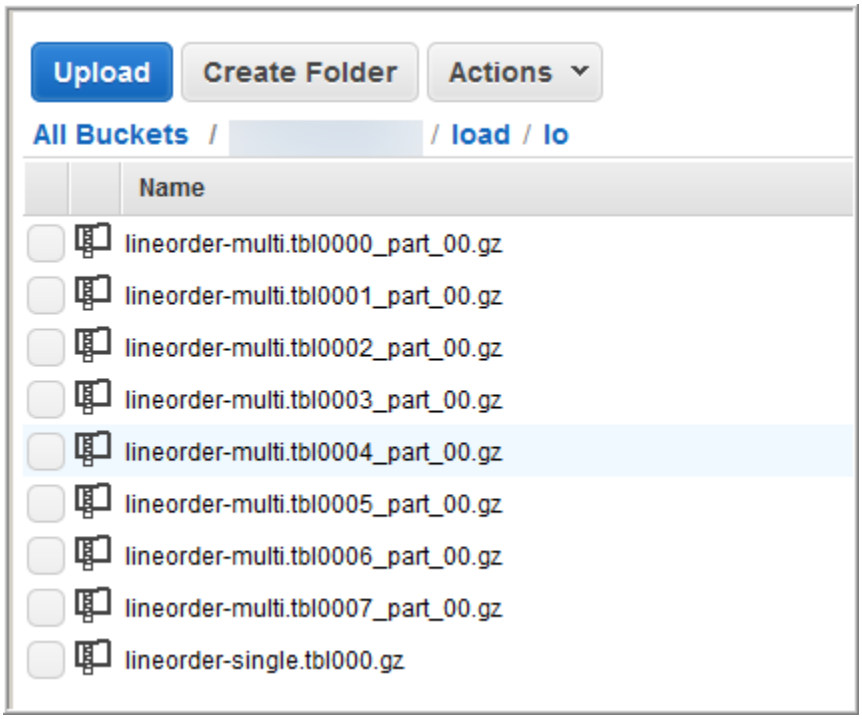
Der Befehl COPY kann Daten sehr effizient laden, wenn er aus mehreren Dateien parallel statt aus einer einzigen Datei lädt. Sie können Ihre Daten in Dateien aufteilen, sodass die Anzahl der Dateien ein Vielfaches der Anzahl der Schichten in Ihrem Cluster beträgt. In diesem Fall teilt Amazon Redshift den Workload auf und verteilt die Daten gleichmäßig auf die Schichten. Die Anzahl der Slices pro Knoten ist von der Knotengröße des Clusters abhängig. Weitere Informationen zur Anzahl

der Slices für die einzelnen Knotengrößen finden Sie unter About Clusters and Nodes (Informationen zu Clustern und Knoten) im Amazon-Redshift-Verwaltungshandbuch.

Beispiel: Die in diesem Tutorial verwendeten Datenverarbeitungsknoten der Größe dc2.large haben jeweils zwei Slices, der aus vier Knoten bestehende Cluster hat also acht Slices. In früheren Schritten waren die Ladedaten in acht Dateien enthalten, obwohl diese sehr klein waren. In diesem Schritt vergleichen Sie den Zeitunterschied zwischen dem Laden aus einer einzigen großen Datei und dem Laden aus mehreren Dateien.

Die Dateien, die Sie für dieses Tutorial verwenden, enthalten etwa 15 Millionen Datensätze und belegen etwa 1,2 GB. Für Amazon Redshift sind dies sehr kleine Dateien, sie reichen jedoch aus, um den Leistungsvorteil des Ladens aus mehreren Dateien zu illustrieren. Die Dateien sind so groß, dass zu viel Zeit erforderlich ist, um sie im Rahmen dieses Tutorials herunterzuladen und dann zu Amazon S3 hochzuladen. Somit laden Sie die Dateien direkt aus einem AWS Beispiel-Bucket.

Das folgende Bildschirmfoto zeigt die Datendateien für LINEORDER.



So evaluieren Sie die Leistung von COPY mit mehreren Dateien:

1. Führen Sie den folgenden COPY-Befehl aus, um aus einer einzelnen Datei zu kopieren. Ändern Sie den Namen des Buckets nicht.

```
copy lineorder from 's3://awssampledb/load/lo/lineorder-single.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
```

```
gzip
compupdate off
region 'us-east-1';
```

2. Ihre Ergebnisse sollten wie folgt aussehen. Beachten Sie die Ausführungszeit.

```
Warnings:
Load into table 'lineorder' completed, 14996734 record(s) loaded successfully.

0 row(s) affected.
copy executed successfully

Execution time: 51.56s
```

3. Führen Sie den folgenden COPY-Befehl aus, um aus mehreren Dateien zu kopieren. Ändern Sie den Namen des Buckets nicht.

```
copy lineorder from 's3://awssampledb/load/lo/lineorder-multi.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
gzip
compupdate off
region 'us-east-1';
```

4. Ihre Ergebnisse sollten wie folgt aussehen. Beachten Sie die Ausführungszeit.

```
Warnings:
Load into table 'lineorder' completed, 14996734 record(s) loaded successfully.

0 row(s) affected.
copy executed successfully

Execution time: 17.7s
```

5. Vergleichen Sie die Ausführungszeiten.

   In unserem Beispiel sank die Zeit für das Ladern von 15 Millionen Datensätzen von 51,56 Sekunden auf 17,7 Sekunden, d. h. um 65,7 Prozent.

   Diese Ergebnisse basieren auf der Verwendung eines Clusters mit vier Knoten. Wenn Ihr Cluster mehr Knoten hat, vervielfachen sich die Zeiteinsparungen. Bei typischen Amazon-Redshift-Clustern mit manchmal Hunderten von Knoten ist der Unterschied noch viel deutlicher. Wenn Sie

einen Cluster mit nur einem Knoten haben, besteht nur ein geringer Unterschied zwischen den Ausführungszeiten.

Nächster Schritt

Schritt 6: Bereinigen und Analysieren der Datenbank

Schritt 6: Bereinigen und Analysieren der Datenbank

Immer wenn Sie eine größere Zahl von Zeilen hinzufügen, löschen oder modifizieren, sollten Sie einen VACUUM- und dann einen ANALYZE-Befehl ausführen. Eine Bereinigung gewinnt den Speicherplatz gelöschter Zeilen zurück und stellt die Sortierfolge wieder her. Der ANALYZE-Befehl aktualisiert die Statistik-Metadaten, wodurch der Abfrageoptimierer korrektere Abfragepläne erstellen kann. Weitere Informationen finden Sie unter Bereinigen von Tabellen.

Wenn Sie die Daten in der Reihenfolge des Sortierschlüssels laden, geht die Bereinigung sehr schnell. In diesem Tutorial haben Sie eine große Zahl von Zeilen hinzugefügt, jedoch in leere Tabellen. Daher muss die Sortierung nicht wiederhergestellt werden; weiterhin haben Sie keine Zeilen gelöscht. COPY aktualisiert die Statistiken automatisch, nachdem eine leere Tabelle geladen wurde. Das sollten auch Ihre Statistiken sein up-to-date. Aus Gründen der guten Organisation schließen Sie dieses Tutorial jedoch ab, indem Sie ein Vacuuming und eine Analyse Ihrer Datenbank durchführen.

Führen Sie zum Bereinigen und Analysieren der Datenbank die folgenden Befehle aus.

```
vacuum;
analyze;
```

Nächster Schritt

Schritt 7: Bereinigen Ihrer Ressourcen

Schritt 7: Bereinigen Ihrer Ressourcen

Solange dieser ausgeführt wird, fallen Gebühren für Ihren Cluster an. Wenn Sie dieses Tutorial abgeschlossen haben, sollten Sie Ihre Umgebung wieder auf den ursprünglichen Zustand zurücksetzen, indem Sie die Schritte in Schritt 5: Aufheben des Zugriffs und Löschen Ihres Beispielclusters im Handbuch Erste Schritte in Amazon Redshift ausführen.

Wenn Sie den Cluster behalten möchten, den von den SSB-Tabellen beanspruchten Speicherplatz jedoch zurückgewinnen möchten, führen Sie die folgenden Befehle aus.

```
drop table part;
drop table supplier;
drop table customer;
drop table dwdate;
drop table lineorder;
```

Next

Übersicht

Übersicht

In diesem Tutorial haben Sie Datendateien zu Amazon S3 hochgeladen und dann die Daten mit dem COPY-Befehl aus den Dateien in Amazon-Redshift-Tabellen geladen.

Sie haben Daten unter Verwendung der folgenden Formate geladen:

- Zeichengetrennt
- CSV
- feste Breite

Mit der Systemtabelle STL_LOAD_ERRORS haben Sie Ladefehler ermittelt und dann mit den Optionen REGION, MANIFEST, MAXERROR, ACCEPTINVCHARS, DATEFORMAT und NULL AS beseitigt.

Zum Laden der Daten haben Sie die folgenden bewährten Verfahren verwendet:

- Verwenden eines COPY-Befehls zum Laden von Daten
- Laden von Datendateien
- Verwenden eines einzelnen COPY-Befehls zum Laden aus mehreren Dateien
- Komprimieren Ihrer Datendateien
- Prüfen der Datendateien vor und nach einem Ladevorgang

Für weitere Informationen zu bewährten Verfahren für Amazon Redshift vgl. die folgenden Links:

- Bewährte Methoden für Amazon Redshift zum Laden von Daten
- Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift
- Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift

Entfernen von Daten

Themen

Um Daten aus Datentabellen zu entfernen und in einen Dateiensatz in einem Amazon-S3-Bucket zu übertragen, können Sie den Befehl UNLOAD mit einer SELECT-Anweisung verwenden. Sie können Textdaten in einem abgeteilten Format oder in einem Format mit fester Breite entladen, unabhängig von dem Datenformat, das zum laden verwendet wurde. Sie können auch angeben, ob komprimierte GZIP-Dateien erstellt werden sollen.

Sie können den Zugriff Ihrer Benutzer auf Ihren Amazon-S3-Bucket mithilfe temporärer Sicherheitsanmeldedaten beschränken.

Entladen von Daten aus Amazon S3

Amazon Redshift teilt die Ergebnisse einer SELECT-Anweisung auf einen Satz von Dateien auf (jeweils eine oder mehrere Dateien pro Knoten-Slice), um das parallele erneute Laden der Daten zu vereinfachen. Sie können auch angeben, dass UNLOAD die Ergebnisse sequenziell in eine oder mehrere Dateien schreibt, indem Sie die Option PARALLEL OFF wählen. Sie können die Größe der Dateien in Amazon S3 durch Angabe des Parameters MAXIFILESIZE begrenzen. UNLOAD verschlüsselt Datendateien automatisch mit der serverseitigen Amazon-S3-Verschlüsselung (SSE-S3).

Sie können jede SELECT-Anweisung in dem UNLOAD-Befehl verwenden, die Amazon Redshift unterstützt, mit Ausnahme von SELECT-Anweisungen mit einer LIMIT-Klausel in der äußeren Auswahl. So können Sie beispielsweise eine SELECT-Anweisung verwenden, die bestimmte Spalten einschließt, oder eine, die eine WHERE-Klausel verwendet, um mehrere Tabellen zu verbinden. Wenn Ihre Abfrage Anführungszeichen (beispielsweise für Literalwerte) oder umgekehrte Schrägstriche (\) enthält, müssen Sie für diese im Abfragetext Escape-Zeichen verwenden (\'). Weitere Informationen finden Sie in der SELECT-Befehlsreferenz. Weitere Informationen zur Verwendung einer LIMIT-Klausel finden Sie unter Nutzungshinweise für den UNLOAD-Befehl.

Beispielsweise sendet der folgende UNLOAD-Befehl die Inhalte der Tabelle VENUE an den Amazon-S3-Bucket `s3://mybucket/tickit/unload/`.

```
unload ('select * from venue')
to 's3://mybucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Die vom vorherigen Beispiel erstellten Dateinamen enthalten das Präfix „„. `'venue_'`.

```
venue_0000_part_00
venue_0001_part_00
venue_0002_part_00
venue_0003_part_00
```

Standardmäßig schreibt UNLOAD die Daten parallel in mehrere Dateien, je nach der Anzahl der Slices in dem Cluster. Um Daten in eine einzelne Datei zu schreiben, geben Sie PARALLEL OFF an. UNLOAD schreibt die Daten seriell, absolut nach der ORDER BY-Klausel sortiert, falls eine solche verwendet wird. Die maximale Größe für eine Datendatei ist 6,2 GB. Wenn der Umfang der Daten die maximale Dateigröße von 6,2 GB überschreitet, erstellt UNLOAD zusätzliche Dateien, jeweils bis zu 6,2 GB.

Das folgende Beispiel schreibt die Inhalte von VENUE in eine einzelne Datei. Es ist nur eine Datei erforderlich, da die Dateigröße unter 6,2 GB liegt.

```
unload ('select * from venue')
to 's3://mybucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off;
```

> **Note**
>
> Der UNLOAD-Befehl ist zur Verwendung der parallelen Verarbeitung gedacht. Wir empfehlen, in den meisten Fällen PARALLEL aktiviert zu lassen, besonders, wenn die Dateien zum Laden von Tabellen mit einem COPY-Befehl verwendet werden sollen.

Unter der Annahme eines Gesamtdatenvolumens für VENUE von 5 GB schreibt das folgende Beispiel den Inhalt von VENUE in 50 Dateien mit jeweils 100 MB.

```
unload ('select * from venue')
to 's3://mybucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off
maxfilesize 100 mb;
```

Wenn Sie ein Präfix in die Amazon-S3-Pfad-Zeichenfolge einschließen, verwendet UNLOAD dieses Präfix für die Dateinamen.

```
unload ('select * from venue')
to 's3://mybucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Sie können eine Manifestdatei erstellen, die die Entladedateien auflistet, indem Sie die Option MANIFEST in dem UNLOAD-Befehl angeben. Das Manifest ist eine Textdatei im JSON-Format, die explizit die URL jeder einzelnen Datei auflistet, die zu Amazon S3 geschrieben wurde.

Das folgende Beispiel enthält die MANIFEST-Option.

```
unload ('select * from venue')
to 's3://mybucket/tickit/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Das folgende Beispiel zeigt ein Manifest für vier Entladedateien.

```
{
  "entries": [
    {"url":"s3://mybucket/tickit/venue_0000_part_00"},
    {"url":"s3://mybucket/tickit/venue_0001_part_00"},
    {"url":"s3://mybucket/tickit/venue_0002_part_00"},
    {"url":"s3://mybucket/tickit/venue_0003_part_00"}
  ]
}
```

Die Manifestdatei kann mit COPY und der Option MANIFEST zum Laden derselben Dateien verwendet werden. Weitere Informationen finden Sie unter Verwenden eines Manifests für die Angabe von Datendateien.

Stellen Sie nach Abschluss einer UNLOAD-Operation sicher, dass die Daten korrekt entladen wurden, indem Sie zu dem Amazon-S3-Bucket navigieren, in dem UNLOAD die Dateien geschrieben

hat. Sie sehen dort eine oder mehrere nummerierte Dateien pro Slice, beginnend mit der Nummer 0. Wenn Sie die MANIFEST-Option angegeben haben, sehen Sie auch eine Datei, die mit „“ endet. '`manifest`'. Beispiel:

```
mybucket/tickit/venue_0000_part_00
mybucket/tickit/venue_0001_part_00
mybucket/tickit/venue_0002_part_00
mybucket/tickit/venue_0003_part_00
mybucket/tickit/venue_manifest
```

Sie können eine Liste der in Amazon S3 geschriebenen Dateien programmgesteuert erhalten, indem Sie nach Abschluss des UNLOAD-Vorgangs eine Amazon-S3-Listenoperation aufrufen. Sie können auch STL_UNLOAD_LOG abfragen.

Die folgende Abfrage gibt den Pfadnamen für Dateien aus, die durch einen UNLOAD-Befehl erstellt wurden. Die Funktion PG_LAST_QUERY_ID gibt die jüngste Abfrage zurück.

```
select query, substring(path,0,40) as path
from stl_unload_log
where query=2320
order by path;

query |             path
-------+---------------------------------------
  2320 | s3://my-bucket/venue0000_part_00
  2320 | s3://my-bucket/venue0001_part_00
  2320 | s3://my-bucket/venue0002_part_00
  2320 | s3://my-bucket/venue0003_part_00
(4 rows)
```

Wenn der Umfang der Daten sehr groß ist, kann Amazon Redshift die Dateien in mehrere Teile pro Slice trennen. Beispiel:

```
venue_0000_part_00
venue_0000_part_01
venue_0000_part_02
venue_0001_part_00
venue_0001_part_01
venue_0001_part_02
...
```

Der folgende UNLOAD-Befehl enthält eine Zeichenfolge in Anführungszeichen in der SELECT-Anweisung, für die Anführungszeichen wird daher ein Escape-Zeichen () verwendet. (=\'OH\' ').

```
unload ('select venuename, venuecity from venue where venuestate=\'OH\' ')
to 's3://mybucket/tickit/venue/ '
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Standardmäßig schlägt UNLOAD fehl, anstatt vorhandene Dateien zu überschreiben, die im Ziel-Bucket vorhanden sind. Um die vorhandenen Dateien, einschließlich der Manifestdatei, zu überschreiben, geben Sie die Option ALLOWOVERWRITE an.

```
unload ('select * from venue')
to 's3://mybucket/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
allowoverwrite;
```

Entladen verschlüsselter Datendateien

UNLOAD erstellt die Dateien automatisch mit der serverseitigen Amazon-S3-Verschlüsselung und AWS-verwalteten Verschlüsselungsschlüsseln (SSE-S3). Sie können auch eine serverseitige Verschlüsselung mit einem AWS Key Management Service-Schlüssel (SSE-KMS) oder eine clientseitige Verschlüsselung mit einem vom Kunden verwalteten Schlüssel angeben. UNLOAD unterstützt keine serverseitige Amazon-S3-Verschlüsselung mit einem vom Kunden verwalteten Schlüssel. Weitere Informationen finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung.

Um Daten zu Amazon S3 unter Verwendung der serverseitigen Verschlüsselung mittels eines AWS KMS-Schlüssels zu übertragen, verwenden Sie den Parameter KMS_KEY_ID zur Bereitstellung der Schlüssel-ID wie im folgenden Beispiel gezeigt.

```
unload ('select venuename, venuecity from venue')
to 's3://mybucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
KMS_KEY_ID '1234abcd-12ab-34cd-56ef-1234567890ab'
encrypted;
```

Wenn Sie Ihren eigenen Verschlüsselungsschlüssel bereitstellen möchten, können Sie clientseitig verschlüsselte Datendateien in Amazon S3 erstellen, indem Sie den UNLOAD-Befehl mit der

Option ENCRYPTED verwenden. UNLOAD verwendet denselben Verschlüsselungsprozess wie die clientseitige Amazon-S3-Verschlüsselung. Anschließend können Sie den COPY-Befehl mit der Option ENCRYPTED verwenden, um die verschlüsselten Dateien zu laden.

Der Prozess funktioniert wie folgt:

1. Sie erstellen einen base64-kodierten 256-Bit-AES-Schlüssel, den Sie als Ihren privaten Verschlüsselungsschlüssel bzw. symmetrischen Root-Schlüssel verwenden.
2. Sie geben einen UNLOAD-Befehl aus, der Ihren symmetrischen Root-Schlüssel und die Option ENCRYPTED enthält.
3. UNLOAD generiert einen symmetrischen Schlüssel zur einmaligen Verwendung (den so genannten Envelope Symmetric Key) und einen Initialisierungsvektor (IV), mit dem Ihre Daten verschlüsselt werden.
4. UNLOAD verschlüsselt den Envelope Symmetric Key unter Verwendung Ihres symmetrischen Root-Schlüssels.
5. Dann speichert UNLOAD die verschlüsselten Datendateien in Amazon S3 und den verschlüsselten Envelope Key und IV als Objektmetadaten mit jeder Datei. Der verschlüsselte Envelope Key wird als Objektmetadaten `x-amz-meta-x-amz-key` und der IV als Objektmetadaten `x-amz-meta-x-amz-iv` gespeichert.

Weitere Informationen zum Envelope-Verschlüsselungsprozess finden Sie im Artikel Clientseitige Datenverschlüsselung mit dem AWS SDK for Java und Amazon S3.

Fügen Sie zum Entladen verschlüsselter Datendateien den Root-Schlüssel-Wert der Anmeldedatenzeichenfolge hinzu und verwenden Sie die Option ENCRYPTED. Wenn Sie die MANIFEST-Option verwenden, wird die Manifestdatei ebenfalls verschlüsselt.

```
unload ('select venuename, venuecity from venue')
to 's3://mybucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
manifest
encrypted;
```

Um verschlüsselte Datendateien mit GZIP-Komprimierung zu entladen, verwenden Sie die Option GZIP zusammen mit dem Root-Schlüssel-Wert und der Option ENCRYPTED.

```
unload ('select venuename, venuecity from venue')
```

```
to 's3://mybucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted gzip;
```

Um die verschlüsselten Datendateien zu laden, fügen Sie den Parameter MASTER_SYMMETRIC_KEY mit dem gleichen Root-Schlüssel-Wert hinzu und verwenden die Option ENCRYPTED.

```
copy venue from 's3://mybucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted;
```

Entladen von Daten im getrennten Format oder im Format mit fester Breite

Sie können Daten im getrennten Format oder im Format mit fester Breite entladen. Die Standardausgabe ist im mit dem Zeichen „|“ getrennten Format.

Das folgende Beispiel verwendet ein Komma als Trennzeichen:

```
unload ('select * from venue')
to 's3://mybucket/tickit/venue/comma'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter ',';
```

Die Ausgabe sieht wie folgt aus:

```
20,Air Canada Centre,Toronto,ON,0
60,Rexall Place,Edmonton,AB,0
100,U.S. Cellular Field,Chicago,IL,40615
200,Al Hirschfeld Theatre,New York City,NY,0
240,San Jose Repertory Theatre,San Jose,CA,0
300,Kennedy Center Opera House,Washington,DC,0
...
```

Geben Sie den folgenden Befehl aus, um den gleichen Ergebnissatz in eine tabulatorgetrennte Datei auszugeben:

```
unload ('select * from venue')
to 's3://mybucket/tickit/venue/tab'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t';
```

Sie können auch die Angabe FIXEDWIDTH verwenden. Diese besteht aus einer Kennung für jede Tabellenspalte sowie der Breite der Spalte (Anzahl von Zeichen). Der UNLOAD-Befehl schlägt fehl, und kürzt die Daten nicht, geben Sie daher eine Breite an, die mindestens der des längsten Eintrags in der Spalte entspricht. Die Verwendung von Daten mit fester Breite funktioniert ähnlich wie bei getrennten Daten, mit der Ausnahme dass die Ausgabe keine Trennzeichen enthält. Beispiel:

```
unload ('select * from venue')
to 's3://mybucket/tickit/venue/fw'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth '0:3,1:100,2:30,3:2,4:6';
```

Die Ausgabe mit fester Breite sieht etwa so aus:

```
20 Air Canada Centre         Toronto      ON0
60 Rexall Place              Edmonton     AB0
100U.S. Cellular Field       Chicago      IL40615
200Al Hirschfeld Theatre     New York CityNY0
240San Jose Repertory TheatreSan Jose     CA0
300Kennedy Center Opera HouseWashington   DC0
```

Für weitere Einzelheiten zur FIXEDWIDTH-Angabe vgl. den Befehl UNLOAD.

Erneutes Laden entladener Daten

Zum erneuten Laden der Ergebnisse einer Entladeoperation können Sie den COPY-Befehl verwenden.

Das folgende Beispiel zeigt einen einzelnen Fall, in dem die Tabelle VENUE mit einer Manifestdatei entladen, gekürzt und erneut geladen wird.

```
unload  ('select * from venue order by venueid')
to 's3://mybucket/tickit/venue/reload_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
manifest
delimiter '|';

truncate venue;

copy venue
from 's3://mybucket/tickit/venue/reload_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
delimiter '|';
```

nach dem erneuten Laden sieht die Tabelle VENUE wie folgt aus:

```
select * from venue order by venueid limit 5;

 venueid |         venuename          |  venuecity  | venuestate | venueseats
---------+----------------------------+-------------+------------+-----------
       1 | Toyota Park                | Bridgeview  | IL         |          0
       2 | Columbus Crew Stadium      | Columbus    | OH         |          0
       3 | RFK Stadium                | Washington  | DC         |          0
       4 | CommunityAmerica Ballpark | Kansas City | KS         |          0
       5 | Gillette Stadium           | Foxborough  | MA         |      68756
(5 rows)
```

Erstellung benutzerdefinierter Funktionen

Sie können eine benutzerdefinierte Skalar-UDF (user-defined function) erstellen, indem Sie entweder eine SQL SELECT-Klausel oder ein Python-Programm verwenden. Die neue Funktion ist in der Datenbank gespeichert und steht jedem Benutzer mit ausreichenden Berechtigungen zur Ausführung zur Verfügung. Sie führen eine benutzerdefinierte skalare UDF genauso aus wie bestehende Amazon-Redshift-Funktionen.

Dabei sind Sie hinsichtlich Python-UDFs nicht auf die Standardfunktionalität von Python beschränkt, sondern Sie können auch benutzerdefinierte Python-Module importieren. Weitere Informationen finden Sie unter Python-Sprachunterstützung für UDFs. Beachten Sie, dass Python 3 für Python-UDFs nicht verfügbar ist. Um Python 3-Unterstützung für Amazon-Redshift-UDFs zu erhalten, verwenden Sie stattdessen . Erstellen einer skalaren Lambda-UDF

Sie können auch AWS Lambda UDFs erstellen, die benutzerdefinierte Funktionen verwenden, die in Lambda als Teil Ihrer SQL-Abfragen definiert sind. Lambda-UDFs ermöglichen es Ihnen, komplexe UDFs zu schreiben und in Komponenten von Drittanbietern zu integrieren. Sie können auch helfen, einige der Einschränkungen aktueller Python- und SQL-UDFs zu überwinden. So können Sie beispielsweise auf Netzwerk- und Speicherressourcen zugreifen und vollwertige SQL-Anweisungen schreiben. Sie können Lambda-UDFs in jeder der von Lambda unterstützten Programmiersprachen erstellen, z. B. Java, Go PowerShell, Node.js, C#, Python und Ruby. Sie können auch eine benutzerdefinierte Laufzeit verwenden.

Standardmäßig können alle Benutzer UDFs ausführen. Weitere Informationen zu Rechten finden Sie in UDF-Sicherheit und Rechte.

Themen

UDF-Sicherheit und Rechte

Zum Erstellen einer UDF müssen Sie eine Berechtigung für die Nutzung in der Sprache für SQL oder plpythonu (Python) besitzen. Standardmäßig wird USAGE ON LANGUAGE SQL für PUBLIC gewährt. Sie müssen jedoch USAGE ON LANGUAGE PLPYTHONU spezifischen Benutzern oder Gruppen explizit gewähren.

Um die Nutzung für SQL zu widerrufen, widerrufen Sie zuerst die Nutzung von PUBLIC. Erlauben Sie dann die Nutzung in SQL nur spezifischen Benutzern oder Gruppen, die auch die Erlaubnis zum Erstellen von SQL-UDFs besitzen. Beim folgenden Beispiel wird die Nutzung von SQL durch Benutzer mit Berechtigungen der Gruppe „PUBLIC“ untersagt. Daraufhin wird die Nutzung der Benutzergruppe erlaubt `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Um eine UDF auszuführen, benötigen Sie für die jeweilige Funktion eine Ausführungsberechtigung. Standardmäßig wird PUBLIC die Ausführungsberechtigung für neue UDFs gewährt. Um die Nutzung einzuschränken, widerrufen Sie diese Berechtigung für die Funktion von PUBLIC. Gewähren Sie die Berechtigung anschließend den spezifischen Einzelpersonen oder Gruppen.

Beim folgenden Beispiel wird die Ausführung der Funktion `f_py_greater` durch Benutzer mit Berechtigungen der Gruppe „PUBLIC“ untersagt. Daraufhin wird die Nutzung der Benutzergruppe erlaubt `udf_devs`.

```
revoke execute on function f_py_greater(a float, b float) from PUBLIC;
grant execute on function f_py_greater(a float, b float) to group udf_devs;
```

Superuser haben standardmäßig alle Berechtigungen.

Weitere Informationen erhalten Sie unter GRANT und REVOKE.

Erstellung einer skalaren SQL-UDF

Eine skalare SQL-UDF enthält eine SQL-SELECT-Klausel, die ausgeführt wird, wenn die Funktion aufgerufen wird, und einen einzelnen Wert zurückgibt. Der Befehl CREATE FUNCTION wird mit den folgenden Parametern aufgerufen:

- (Optional) Eingabeparameter, Jedes Argument muss einen Datentyp aufweisen.
- Ein Rückgabedatentyp
- Eine SQL-SELECT-Klausel. Nutzen Sie in Übereinstimmung mit der Reihenfolge der Argumente in der Funktionsdefinition die Elemente „$1“, „$2“ usw. zur Benennung der Eingabeargumente in der SELECT-Klausel.

Ein- und Rückgabedaten können jeden standardmäßigen Amazon-Redshift-Datentyp verwenden.

Die SELECT-Klausel darf keine FROM-Klausel enthalten. Bauen Sie die FROM-Klausel stattdessen in die SQL-Anweisung ein, die die SQL-UDF aufruft.

Die SELECT-Klausel darf keinen der folgenden Klausel-Typen enthalten:

- FROM
- INTO
- WHERE
- GROUP BY
- ORDER BY
- LIMIT

Beispiel für eine skalare SQL-Funktion

Im folgenden Beispiel wird eine Funktion erstellt, die zwei Zahlen vergleicht und die größere zurückgibt. Weitere Informationen finden Sie unter CREATE FUNCTION.

```
create function f_sql_greater (float, float)
  returns float
stable
as $$
  select case when $1 > $2 then $1
    else $2
  end
$$ language sql;
```

In der folgenden Abfrage wird die neue Funktion „f_sql_greater“ aufgerufen, um eine Abfrage über der Tabelle „SALES“ auszuführen und entweder „COMMISSION“ oder 20 Prozent von „PRICEPAID“ zurückzugeben, je nachdem, welcher Wert größer ist.

```
select f_sql_greater(commission, pricepaid*0.20) from sales;
```

Benennung von UDFs

Sie können Namenskonflikte und unerwartete Ergebnisse verhindern, wenn Sie sich bei der Implementation an Namenskonventionen halten. Da Funktionsnamen überladen sein können, besteht die Möglichkeit, dass Namenskonflikte mit anderen Amazon-Redshift-Funktionen auftreten oder sich zu einem späteren Zeitpunkt ergeben können. In diesem Thema werden überladene Funktionen und eine Strategie zur Vermeidung von Namenskonflikten besprochen.

Überladen von Funktionsnamen

Eine Funktion wird anhand Ihres Namens und ihrer Signatur identifiziert, wobei die Signatur im Wesentlichen die Anzahl und die Datentypen der Eingabeparameter ist. Solange sich zwei Funktionen in demselben Schema bezüglich ihrer Signatur unterscheiden, können sie denselben Namen haben. Mit anderen Worten, die Funktionsnamen können überladen sein.

Wenn Sie eine Abfrage ausführen, identifiziert die Abfrage-Engine auf der Grundlage des übergebenen Namens und der Datentypen der Parameter, welche Funktion aufgerufen werden soll. Sie können das Überladen nutzen, um Funktionen mit einer variablen Anzahl von Argumenten bis zur maximal durch den Befehl CREATE FUNCTION zugelassen Anzahl zu simulieren.

Verhindern von UDF-Namenskonflikten

Es wird empfohlen, dass Sie alle UDFs mit dem Präfix `f_` benennen. Amazon Redshift reserviert das Präfix `f_` ausschließlich für UDFs. Wenn Sie Ihren UDF-Namen `f_` voranstellen, stellen Sie daher sicher, dass Ihre UDF-Namen nicht im Widerspruch zu den Namen vorhandener oder zukünftiger, in Amazon Redshift integrierter SQL-Funktionen stehen. Wenn Sie eine neue UDF beispielsweise `f_sum` benennen, vermeiden Sie einen Konflikt mit der SUM-Funktion von Amazon Redshift. Wenn Sie eine neue Funktion mit `f_fibonacci` benennen, vermeiden Sie einen Konflikt, wenn Amazon Redshift in einer zukünftigen Version eine Funktion mit dem Namen FIBONACCI hinzufügt.

Sie können eine UDF mit demselben Namen und derselben Signatur wie eine vorhandene integrierte Amazon-Redshift-SQL-Funktion erstellen, ohne dass der Funktionsname überladen ist, wenn sich die UDF und die integrierte Funktion in verschiedenen Schemata befinden. Integrierte Funktionen befinden sich im Systemschema „pg_catalog“, daher können Sie UDFs mit demselben Namen wie integrierte Funktionen in einem anderen Schema erstellen, beispielsweise in einem öffentlichen oder

in einem benutzerdefinierten Schema. In einigen Fällen können Sie möglicherweise eine Funktion aufrufen, die nicht explizit mit einem Schemanamen qualifiziert ist. Wenn ja, durchsucht Amazon Redshift standardmäßig zuerst das Schema pg_catalog. Daher läuft eine eingebaute Funktion vor einer neuen UDF mit dem gleichen Namen.

Sie können dieses Verhalten ändern, indem Sie den Suchpfad so angeben, dass „pg_catalog“ ans Ende gestellt ist. Wenn Sie dies tun, haben Ihre UDFs Vorrang vor integrierten Funktionen. Diese Vorgehensweise kann jedoch zu nicht vorhergesehenen Ergebnissen führen. Es wird stattdessen geraten, eine Namenskonvention zu verwenden, beispielsweise das reservierte Präfix `f_`, weil diese Vorgehensweise zuverlässiger ist. Weitere Informationen erhalten Sie unter SET und search_path.

Erstellung einer skalaren Python-UDF

Eine skalare Python-UDF enthält ein Python-Programm, das ausgeführt wird, wenn die Funktion aufgerufen wird, und die Funktion gibt dann einen einzelnen Wert zurück. Der Befehl CREATE FUNCTION wird mit den folgenden Parametern aufgerufen:

- (Optional) Eingabeparameter, jeweils mit Namen und Datentyp
- Ein Rückgabedatentyp
- Ein ausführbares Python-Programm

Als Eingabe- und Rückgabedatentypen kommen SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE oder TIMESTAMP infrage. Zusätzlich können Python-UDFs den Datentyp ANYELEMENT verwenden. Dieser wird von Amazon Redshift basierend auf den zur Laufzeit bereitgestellten Argumenten automatisch in einen Standarddatentyp umgewandelt. Weitere Informationen finden Sie unter Datentyp ANYELEMENT

Wenn eine Amazon-Redshift-Abfrage eine skalare UDF aufruft, wird zur Laufzeit Folgendes durchgeführt.

1. Die Funktion wandelt die Eingabeparameter werden in Python-Datentypen um.

   Ein Mapping von Amazon-Redshift-Datentypen zu Python-Datentypen finden Sie in Python-UDF-Datentypen.
2. Die Funktion führt das Python-Programm mit den umgewandelten Eingabeparametern aus.
3. Das Python-Programm gibt einen einzelnen Wert zurück. Der Datentyp des Rückgabewerts muss dem RETURNS-Datentyp in der Funktionsdefinition entsprechen.

4. Die Funktion wandelt den Python-Rückgabewert in den angegebenen Amazon-Redshift-Datentyp um und gibt diesen Wert an die Abfrage zurück.

> **Note**
>
> Python 3 ist für Python-UDFs nicht verfügbar. Um Python 3-Unterstützung für Amazon-Redshift-UDFs zu erhalten, verwenden Sie stattdessen . Erstellen einer skalaren Lambda-UDF

Beispiel für eine skalare Python-UDF

Im folgenden Beispiel wird eine Funktion erstellt, die zwei Zahlen vergleicht und die größere zurückgibt. Beachten Sie, dass es sich bei der Einrückung des Codes zwischen den doppelten Dollarzeichen ($$) um eine Python-Anforderung handelt. Weitere Informationen finden Sie unter CREATE FUNCTION.

```
create function f_py_greater (a float, b float)
  returns float
stable
as $$
  if a > b:
    return a
  return b
$$ language plpythonu;
```

In der folgenden Abfrage wird die neue Funktion `f_greater` aufgerufen, um eine Abfrage über der Tabelle SALES auszuführen und entweder COMMISSION oder 20 Prozent von PRICEPAID zurückzugeben, je nachdem, welcher Wert größer ist.

```
select f_py_greater (commission, pricepaid*0.20) from sales;
```

Python-UDF-Datentypen

Python-UDFs können alle als Eingabeparameter und als Rückgabewert alle Standard-Amazon-Redshift-Datentypen verwenden. Zusätzlich zu den Standarddatentypen unterstützen UDFs den Datentyp ANYELEMENT, den Amazon Redshift automatisch in einen Standarddatentyp konvertiert, abhängig von den zur Ausführungszeit bereitgestellten Argumenten. Skalare UDFs können den

Datentyp ANYELEMENT auch zurückgeben. Weitere Informationen finden Sie unter Datentyp ANYELEMENT.

Während der Ausführung konvertiert Amazon Redshift die Argumente von Amazon-Redshift-Datentypen zu Python-Datentypen zur Verarbeitung. Anschließend wird der Rückgabewert vom Python-Datentyp in den entsprechenden Amazon-Redshift-Datentyp konvertiert. Weitere Informationen zu Amazon-Redshift-Datentypen finden Sie unter Datentypen.

Die folgende Tabelle enthält eine Zuordnung von Amazon-Redshift-Datentypen und Python-Datentypen.

Amazon-Redshift-Datentyp	Python-Datentyp
smallint	int
integer	
bigint	
short	
long	
decimal oder numeric	Dezimalwert
double	float
real	
Boolean	bool
char	string
varchar	
timestamp	datetime

Datentyp ANYELEMENT

ANYELEMENT ist ein polymorpher Datentyp. Wenn eine Funktion unter Verwendung von ANYELEMENT für den Datentyp eines Arguments deklariert wird, kann die Funktion alle Amazon-

Redshift-Standarddatentypen als Eingabe für das betreffende Argument akzeptieren, wenn die Funktion aufgerufen wird. Der ANYELEMENT-Parameter wird auf den übergebenen Datentyp gesetzt, wenn die Funktion aufgerufen wird.

Wenn eine Funktion mehrere ANYELEMENT-Datentypen verwendet, werden sie beim Funktionsaufruf alle in denselben Datentyp aufgelöst. Alle ANYELEMENT-Parameter werden auf den Datentyp gesetzt, in den der erste übergebene ANYELEMENT-Parameter aufgelöst wird. Wenn eine Funktion beispielsweise als `f_equal(anyelement, anyelement)` deklariert ist, kann sie zwei beliebige Eingabewerte entgegennehmen, solange diese vom selben Datentyp sind.

Wenn Sie eine Funktion selbst als ANYELEMENT deklarieren, muss mindestens einer der Eingabeparameter vom Typ ANYELEMENT sein. Der tatsächliche Datentyp für den Rückgabewert ist identisch mit dem tatsächlichen Datentyp, der für das ANYELEMENT-Eingabeargument bereitgestellt wird.

Python-Sprachunterstützung für UDFs

Sie können auf der Basis von Python benutzerdefinierte Funktionen (UDFs, User-Defined Functions) erstellen. Sie können die Python 2.7-Standardbibliothek in UDFs verwenden, ausgenommen folgende Module:

- ScrolledText
- Tix
- Tkinter
- tk
- turtle
- smtpd

Neben der Python-Standardbibliothek sind in die Amazon-Redshift-Implementation die folgenden Module integriert:

- numpy 1.8.2
- pandas 0.14.1
- python-dateutil 2.2
- pytz 2014.7
- scipy 0.12.1

- six 1.3.0
- wsgiref 0.1.2

Sie können über den Befehl CREATE LIBRARY auch eigene Python-Module importieren und zur Verwendung in UDFs zur Verfügung stellen. Weitere Informationen finden Sie unter Importieren von benutzerdefinierten Python-Bibliotheksmodulen.

> **Important**
>
> Amazon Redshift blockiert den gesamten Netzwerkzugriff und den Schreibzugriff auf das Dateisystem durch UDFs.

> **Note**
>
> Python 3 ist für Python-UDFs nicht verfügbar. Um Python 3-Unterstützung für Amazon-Redshift-UDFs zu erhalten, verwenden Sie stattdessen . Erstellen einer skalaren Lambda-UDF

Importieren von benutzerdefinierten Python-Bibliotheksmodulen

Skalare Funktionen werden entsprechend der Syntax von Python definiert. Sie können Module der Python-Standardbibliothek und vorinstallierte Amazon-Redshift-Module verwenden. Sie können auch eigene benutzerdefinierte Python-Bibliotheksmodule erstellen und zusammen mit anderen Bibliotheken von Python oder Drittanbietern in Ihre Cluster importieren.

Die erstellten Bibliotheken dürfen jedoch nicht genau so heißen wie die integrierten Python-Standardbibliotheksmodule oder die vorinstallierten Amazon-Redshift-Python-Module. Wenn eine vorhandene, benutzerinstallierte Bibliothek dasselbe Python-Paket wie eine von Ihnen erstellte Bibliothek verwendet, müssen Sie die vorhandene Bibliothek entfernen, bevor Sie die neue Bibliothek installieren.

Sie müssen Superuser sein oder über die Berechtigung `USAGE ON LANGUAGE plpythonu` verfügen, um benutzerdefinierte Bibliotheken installieren zu können. Die installierten Bibliotheken können anschließend von allen Benutzern mit der Berechtigung zum Erstellen von Funktionen verwendet werden. Um Informationen zu den in Ihrem Cluster installierten Bibliotheken anzuzeigen, führen Sie eine Abfrage über dem Systemkatalog PG_LIBRARY aus.

So importieren Sie ein benutzerdefiniertes Python-Modul in Ihren Cluster:

Dieser Abschnitt enthält ein Beispiel zum Importieren eines benutzerdefiniertes Python-Moduls in Ihren Cluster. Für die Durchführung der Schritte in diesem Abschnitt benötigen Sie einen Amazon S3 Bucket, in den Sie das Bibliothekspaket hochladen. Dann installieren Sie das Paket in Ihrem Cluster. Weitere Informationen zum Erstellen von Buckets finden Sie unter Erstellen von Buckets im Benutzerhandbuch zu Amazon Simple Storage Service.

Hintergrund dieses Beispiels ist, dass UDFs erstellt werden sollen, um mit Positionen und Entfernungen in Ihren Daten zu arbeiten. Stellen Sie mit einem SQL-Client-Tool eine Verbindung zu Ihrem Amazon-Redshift-Cluster her und führen die folgenden Befehle aus, um die Funktionen zu erstellen.

```
CREATE FUNCTION f_distance (x1 float, y1 float, x2 float, y2 float) RETURNS float
 IMMUTABLE as $$
    def distance(x1, y1, x2, y2):
        import math
        return math.sqrt((y2 - y1) ** 2 + (x2 - x1) ** 2)

    return distance(x1, y1, x2, y2)
$$ LANGUAGE plpythonu;

CREATE FUNCTION f_within_range (x1 float, y1 float, x2 float, y2 float) RETURNS bool
 IMMUTABLE as $$
    def distance(x1, y1, x2, y2):
        import math
        return math.sqrt((y2 - y1) ** 2 + (x2 - x1) ** 2)

    return distance(x1, y1, x2, y2) < 20
$$ LANGUAGE plpythonu;
```

Beachten Sie, dass in den Funktionen oben einige Codezeilen mehrmals vorkommen. Dies ist notwendig, weil UDFs nicht den Inhalt anderer UDFs referenzieren können, aber beide Funktionen dieselbe Funktionalität benötigen. Natürlich können Sie auch eine benutzerdefinierte Bibliothek mit dem mehrfach verwendeten Code erstellen und Ihre Funktionen so schrieben, dass sie diese Bibliothek verwenden.

Dazu erstellen Sie zuerst das Bibliothekspaket, indem Sie die folgenden Schritte ausführen:

1. Erstellen Sie einen Ordner mit dem Namen geometry. Dieser Ordner ist das Bibliothekspaket der obersten Ebene.

2. Erstellen Sie im Ordner geometry eine Datei mit dem Namen `__init__.py`. Beachten Sie, dass die Unterstriche in dem Dateinamen jeweils doppelte Unterstriche sind. Diese Datei weist Python darauf hin, dass das Paket initialisiert werden kann.
3. Erstellen Sie im Ordner geometry außerdem einen Ordner mit dem Namen trig. Dieser Ordner ist das Unterpaket der Bibliothek.
4. Erstellen Sie im Ordner trig eine Datei mit dem Namen `__init__.py` und eine Datei mit dem Namen `line.py`. Mit der Datei `__init__.py` wird Python wieder darauf hingewiesen, dass das Unterpaket initialisiert werden kann, und dass die Datei `line.py` den Bibliothekscode enthält.

   Ihr Ordner sollte nun eine Dateistruktur wie folgt enthalten:

   ```
   geometry/
      __init__.py
      trig/
         __init__.py
         line.py
   ```

   Weitere Informationen über die Struktur von Paketen finden Sie unter Modules im Python-Tutorial auf der Python-Website.
5. Der folgende Quellcode enthält eine Klasse und zugehörige Mitgliedsfunktionen für die Bibliothek. Kopieren Sie den Code und fügen Sie ihn in ein `line.py`.

   ```
   class LineSegment:
     def __init__(self, x1, y1, x2, y2):
       self.x1 = x1
       self.y1 = y1
       self.x2 = x2
       self.y2 = y2
     def angle(self):
       import math
       return math.atan2(self.y2 - self.y1, self.x2 - self.x1)
     def distance(self):
       import math
       return math.sqrt((self.y2 - self.y1) ** 2 + (self.x2 - self.x1) ** 2)
   ```

Nachdem Sie das Paket erstellt haben, gehen Sie wie folgt vor, um das Paket vorzubereiten und in Amazon S3 hochzuladen.

1. Komprimieren Sie den Inhalt des Ordners geometry in einer ZIP-Datei mit dem Namen geometry.zip. Der Ordner geometry selbst darf nicht in der ZIP-Datei enthalten sein, sondern nur der Inhalt des Ordners wie im Folgenden gezeigt:

```
geometry.zip
   __init__.py
   trig/
      __init__.py
      line.py
```

2. Laden Sie geometry.zip in Ihren Amazon S3 Bucket hoch.

> **Important**
>
> Wenn sich der Amazon-S3-Bucket nicht in derselben Region wie Ihr Amazon-Redshift-Cluster befindet, müssen Sie die Option REGION verwenden, um die Region anzugeben, in der sich die Daten befinden. Weitere Informationen finden Sie unter CREATE LIBRARY.

3. Führen Sie in Ihrem SQL-Client-Tool den folgenden Befehl aus, um die Bibliothek zu installieren. Ersetzen Sie *<bucket_name>* durch den Namen Ihres Buckets und ersetzen Sie *<access key id>* und *<secret key>* durch einen Zugriffsschlüssel und einen geheimen Zugriffsschlüssel aus Ihren AWS Identity and Access Management (IAM)-Benutzeranmeldeinformationen.

```
CREATE LIBRARY geometry LANGUAGE plpythonu FROM 's3://<bucket_name>/geometry.zip'
 CREDENTIALS 'aws_access_key_id=<access key id>;aws_secret_access_key=<secret key>';
```

Nach der Installation der Bibliothek in Ihrem Cluster müssen Sie die Funktionen so konfigurieren, dass sie die Bibliothek verwenden. Führen Sie dazu die folgenden Befehle aus.

```
CREATE OR REPLACE FUNCTION f_distance (x1 float, y1 float, x2 float, y2 float) RETURNS
 float IMMUTABLE as $$
    from trig.line import LineSegment

    return LineSegment(x1, y1, x2, y2).distance()
$$ LANGUAGE plpythonu;

CREATE OR REPLACE FUNCTION f_within_range (x1 float, y1 float, x2 float, y2 float)
 RETURNS bool IMMUTABLE as $$
    from trig.line import LineSegment
```

```
    return LineSegment(x1, y1, x2, y2).distance() < 20
$$ LANGUAGE plpythonu;
```

In den Befehlen oben können durch die Anweisung `import trig/line` die doppelten Vorkommen des Codes wie in der ersten Version dieser Funktionen am Anfang dieses Abschnitts vermieden werden. Sie können durch Bibliotheken bereitgestellte Funktionalitäten in mehreren UDFs verwenden. Beachten Sie, dass Sie beim Importieren des Moduls lediglich den Pfad zu dem Unterpaket und den Modulnamen angeben (`trig/line`).

Einschränkungen für UDFs

Solange Sie die in diesem Thema aufgelisteten Einschränkungen berücksichtigen, können Sie UDFs überall dort verwenden, wo Sie auch die integrierten skalaren Funktionen von Amazon Redshift einsetzen können. Weitere Informationen finden Sie unter SQL-Funktionsreferenz.

Für Amazon-Redshift-Python-UDFs gelten die folgenden Einschränkungen:

- Python-UDFs können weder auf das Netzwerk zugreifen noch im Dateisystem lesen oder schreiben.
- Die Gesamtgröße von Python-Bibliotheken, die von Benutzern installiert werden, darf 100 MB nicht überschreiten.
- Die Anzahl der Python-UDFs, die gleichzeitig pro Cluster ausgeführt werden können, ist auf ein Viertel der gesamten Parallelausführungsstufe (Gleichzeitigkeitsstufe) für den Cluster begrenzt. Wenn der Cluster beispielsweise mit einem Gleichzeitigkeitswert von 15 konfiguriert ist, dürfen maximal drei UDFs gleichzeitig ausgeführt werden. Wenn dieses Limit erreicht ist, werden weitere UDFs zur Ausführung in die Workload Management-Warteschlangen eingestellt. SQL-UDFs weisen keine Gleichzeitigkeitsbeschränkung auf. Weitere Informationen finden Sie unter Implementierung von Workload Management.
- Bei Verwendung von Python-UDFs unterstützt Amazon Redshift die Datentypen SUPER und HLLSKETCH nicht.

Protokollieren von Fehlern und Warnungen in UDFs

Die können das Protokollierungsmodul von Python verwenden, um in Ihren UDFs benutzerdefinierte Fehler- und Warnmeldungen zu erstellen. Nach der Abfrageausführung können Sie die Systemansicht SVL_UDF_LOG abfragen, um protokollierte Meldungen abzurufen.

> **Note**
>
> Beachten Sie, dass die Protokollierung von UDFs Clusterressourcen verbraucht und die Systemleistung beeinträchtigen kann. Wir empfehlen, die Protokollierung nur in der Entwicklung und zur Problembehandlung zu implementieren.

Wenn eine Abfrage ausgeführt wird, schreibt der Protokollhandler Meldungen in die Systemansicht SVL_UDF_LOG, jeweils mit dem betreffenden Funktionsnamen, dem Knoten und der Slice. Der Protokollhandler schreibt jeweils pro Meldung und Slice eine Zeile in SVL_UDF_LOG. Meldungen werden jeweils auf eine maximale Länge von 4096 Bytes gekürzt. Das UDF-Protokoll ist auf 500 Zeilen pro Slice begrenzt. Wenn das Protokoll voll ist, werden alte Meldungen gelöscht und eine entsprechende Warnmeldung in SVL_UDF_LOG hinzugefügt.

> **Note**
>
> Der Amazon-Redshift-UDF-Protokollhandler kodiert Zeilenenden (\n), Pipe-Zeichen (|) und umgekehrte Schrägstriche (\ ) mit einem umgekehrten Schrägstrich (\ ).

Standardmäßig ist die Protokollierungsstufe für UDFs auf WARNING festgelegt. Dies bedeutet, dass Meldungen mit den Protokollierungsstufen WARNING, ERROR und CRITICAL protokolliert werden. Meldungen mit geringerem Schweregrad wie INFO, DEBUG und NOTSET werden ignoriert. Sie können die Protokollierungsstufe für UDF mittels der Python-Methode „logger“ einstellen. Mit dem folgenden Aufruf können Sie beispielsweise die Protokollierungsstufe auf INFO festlegen.

```
logger.setLevel(logging.INFO)
```

Weitere Informationen zur Verwendung des Python-Protokollierungsmoduls finden unter Logging facility for Python in der Python-Dokumentation.

In dem folgenden Beispiel wird eine Funktion namens „f_pyerror“ erstellt, die das Python-Modul zur Protokollierung importiert, das „logger“-Objekt instanziiert und einen Fehler protokolliert.

```
CREATE OR REPLACE FUNCTION f_pyerror()
RETURNS INTEGER
VOLATILE AS
$$
import logging
```

```
logger = logging.getLogger()
logger.setLevel(logging.INFO)
logger.info('Your info message here')
return 0
$$ language plpythonu;
```

Das folgende Beispiel ist eine SVL_UDF_LOG-Abfrage zur Anzeige der in dem vorangehenden Beispiel protokollierten Meldung.

```
select funcname, node, slice, trim(message) as message
from svl_udf_log;

  funcname  | query | node | slice |   message
------------+-------+------+-------+-----------------
  f_pyerror | 12345 |     1|     1 | Your info message here
```

Erstellen einer skalaren Lambda-UDF

Amazon Redshift kann benutzerdefinierte Funktionen verwenden, die in AWS Lambda als Teil von SQL-Abfragen definiert sind. Sie können skalare Lambda-UDFs in allen von Lambda unterstützten Programmiersprachen schreiben, z. B. Java, Go, PowerShell, Node.js, C#, Python und Ruby. Sie können auch eine benutzerdefinierte Laufzeit verwenden.

Lambda-UDFs werden in Lambda definiert und verwaltet, und Sie können die Zugriffsberechtigungen zum Aufrufen dieser UDFs in Amazon Redshift steuern. Sie können mehrere Lambda-Funktionen in derselben Abfrage aufrufen oder die gleiche Funktion mehrmals aufrufen.

Verwenden Sie Lambda-UDFs in allen Klauseln der SQL-Anweisungen, in denen skalare Funktionen unterstützt werden. Sie können Lambda-UDFs auch in jeder SQL-Anweisung wie SELECT, UPDATE, INSERT oder DELETE verwenden.

> **Note**
>
> Die Verwendung von Lambda-UDFs kann zusätzliche Gebühren für den Lambda-Service verursachen. Ob dies geschieht, hängt von Faktoren wie der Anzahl der Lambda-Anforderungen (UDF-Aufrufe) und der Gesamtdauer der Lambda-Programmausführung ab. Es gibt jedoch keine zusätzlichen Gebühren für die Verwendung von Lambda-UDFs in Amazon Redshift. Weitere Informationen zu AWS Lambda-Preisen finden Sie unter -AWS Lambda Preise.

Die Anzahl der Lambda-Anforderungen hängt von der spezifischen SQL-Anweisungsklausel ab, in der die Lambda-UDF verwendet wird. Angenommen, die Funktion wird in einer WHERE-Klausel wie der folgenden verwendet.

```
SELECT a, b FROM t1 WHERE lambda_multiply(a, b) = 64; SELECT a, b
FROM t1 WHERE a*b = lambda_multiply(2, 32)
```

In diesem Fall ruft Amazon Redshift die erste SELECT-Anweisung für jede und ruft die zweite SELECT-Anweisung nur einmal auf.

Wenn Sie jedoch ein UDF im Projektionsteil der Abfrage verwenden, wird die Lambda-Funktion möglicherweise nur einmal für jede qualifizierte oder aggregierte Zeile in der Ergebnismenge aufgerufen.

Registrieren eine Lambda-UDF

Der Befehl CREATE EXTERNAL FUNCTION erstellt die folgenden Parameter:

- (Optional) Eine Liste von Argumenten mit Datentyp.
- Ein Rückgabedatentyp
- Ein Funktionsname der externen Funktion, die von Amazon Redshift aufgerufen wird.
- Eine IAM-Rolle, die der Amazon-Redshift-Cluster übernehmen und Lambda aufrufen darf.
- Ein Lambda-Funktionsname, den die Lambda-UDF aufruft.

Weitere Hinweise zu CREATE EXTERNAL FUNCTION finden Sie unter CREATE EXTERNAL FUNCTION.

Ein- und Rückgabedaten für diese Funktion können jeden standardmäßigen Amazon-Redshift-Datentyp verwenden.

Amazon Redshift stellt sicher, dass die externe Funktion Batch-Argumente und -Ergebnisse senden und empfangen kann.

Verwaltung von Lambda-UDF-Sicherheit und -Berechtigungen

Um eine Lambda-UDF zu erstellen, stellen Sie sicher, dass Sie über Berechtigungen für die Verwendung auf LANGUAGE EXFUNC verfügen. Sie müssen USAGE ON LANGUAGE EXFUNC explizit gewähren oder USAGE ON LANGUAGE EXFUNC für bestimmte Benutzer, Gruppen oder öffentliche Benutzer widerrufen.

Im folgenden Beispiel wird PUBLIC die Verwendung auf EXFUNC gewährt.

```
grant usage on language exfunc to PUBLIC;
```

Im folgenden Beispiel wird die Verwendung auf exfunc von PUBLIC widerrufen. Anschließend wird die Nutzung der Benutzergruppe lambda_udf_devs erlaubt.

```
revoke usage on language exfunc from PUBLIC;
grant usage on language exfunc to group lambda_udf_devs;
```

Stellen Sie zum Ausführen einer Lambda-UDF sicher, dass Sie die Berechtigung für jede aufgerufene Funktion haben. Standardmäßig wird PUBLIC die Ausführungsberechtigung für neue Lambda-UDFs gewährt. Um die Nutzung einzuschränken, widerrufen Sie diese Berechtigung für die Funktion von PUBLIC. Gewähren Sie die Berechtigung anschließend den spezifischen Benutzern oder Gruppen.

Beim folgenden Beispiel wird die Ausführung der Funktion exfunc_sum durch Benutzer mit Berechtigungen der Gruppe PUBLIC untersagt. Daraufhin wird die Nutzung der Benutzergruppe lambda_udf_devs erlaubt.

```
revoke execute on function exfunc_sum(int, int) from PUBLIC;
grant execute on function exfunc_sum(int, int) to group lambda_udf_devs;
```

Superuser haben standardmäßig alle Berechtigungen.

Weitere Informationen zum Erteilen und Widerrufen von Berechtigungen finden Sie unter GRANT und REVOKE.

Konfigurieren des Autorisierungsparameters für Lambda-UDFs

Der Befehl CREATE EXTERNAL FUNCTION erfordert die Autorisierung, Lambda-Funktionen in AWS Lambda aufzurufen. Um die Autorisierung zu starten, geben Sie eine AWS Identity and Access Management (IAM)-Rolle an, wenn Sie den Befehl CREATE EXTERNAL FUNCTION ausführen. Weitere Informationen zu IAM-Rollen finden Sie unter IAM-Rollen im IAM-Benutzerhandbuch.

Wenn bereits eine IAM-Rolle mit Berechtigungen zum Aufrufen von Lambda-Funktionen vorhanden ist, können Sie den Befehl durch Ihre Rolle Amazon-Ressourcenname (ARN) im Parameter IAM_ROLE ersetzen. Die folgenden Abschnitte beschreiben die Schritte zur Verwendung einer IAM-Rolle im Befehl CREATE EXTERNAL FUNCTION.

Erstellen einer IAM-Rolle für Lambda

Die IAM-Rolle erfordert die Berechtigung zum Aufrufen von Lambda-Funktionen. Geben Sie beim Erstellen der IAM-Rolle die Berechtigung auf eine der folgenden Arten an:

- Fügen Sie die `AWSLambdaRole`-Richtlinie auf der Seite Berechtigungsrichtlinie anfügen an, während Sie eine IAM-Rolle erstellen. Die`AWSLambdaRole`-Richtlinie erteilt Berechtigungen zum Aufrufen von Lambda-Funktionen, was die minimale Anforderung ist. Weitere Informationen und andere Richtlinien finden Sie unter Identitätsbasierte IAM-Richtlinien für AWS Lambda im AWS Lambda -Entwicklerhandbuch.
- Erstellen Sie Ihre eigene benutzerdefinierte Richtlinie, die Sie Ihrer IAM-Rolle mit der `lambda:InvokeFunction`-Berechtigung entweder für alle Ressourcen oder eine bestimmten Lambda-Funktion mit dem ARN dieser Funktion anfügen. Weitere Informationen zum Erstellen von Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

  Die folgende Beispielrichtlinie ermöglicht das Aufrufen von Lambda für eine bestimmte Lambda-Funktion.

  ```
  {
      "Version": "2012-10-17",
      "Statement": [
          {
              "Sid": "Invoke",
              "Effect": "Allow",
              "Action": [
                  "lambda:InvokeFunction"
              ],
              "Resource": "arn:aws:lambda:us-west-2:123456789012:function:my-function"
          }
      ]
  }
  ```

  Weitere Informationen zu Ressourcen für Lambda Funktionen finden Sie unter Ressourcen und Bedingungen für Lambda-Aktionen in der IAM-API-Referenz.

  Nachdem Sie Ihre benutzerdefinierte Richtlinie mit den erforderlichen Berechtigungen erstellt haben, können Sie Ihre Richtlinie der IAM-Rolle auf der Registerkarte Berechtigungsrichtlinie anfügen anfügen, während Sie eine IAM-Rolle erstellen.

Schritte zum Erstellen einer IAM-Rolle finden Sie unter Amazon Redshift autorisieren, in Ihrem Namen auf andere - AWS Services zuzugreifen im Amazon-Redshift-Verwaltungshandbuch.

Wenn Sie keine neue IAM-Rolle erstellen möchten, können Sie die zuvor genannten Berechtigungen zu Ihrer vorhandenen IAM-Rolle hinzufügen.

Zuweisen einer IAM-Rolle zum Cluster

Fügen Sie die IAM-Rolle an Ihren Cluster an. Sie können über die Amazon-Redshift-Managementkonsole, die CLI oder eine API einem Cluster eine Rolle hinzufügen oder die Rollen anzeigen, die mit einem Cluster verknüpft sind. Weitere Informationen finden Sie unter Verknüpfen einer IAM-Rolle mit einem Cluster im Amazon-Redshift-Verwaltungshandbuch.

Einschließen der IAM-Rolle in den Befehl

Schließen Sie die IAM-Rolle ARN in den Befehl CREATE EXTERNAL FUNCTION ein. Beim Erstellen einer IAM-Rolle gibt IAM einen Amazon-Ressourcennamen (ARN) für die Rolle zurück. Um eine IAM-Rolle anzugeben, geben Sie für den ARN der Rolle den Parameter `IAM_ROLE` an. Im Folgenden wird die Syntax für den Parameter `IAM_ROLE` gezeigt.

```
IAM_ROLE 'arn:aws:iam::aws-account-id:role/role-name'
```

Informationen zum Aufrufen von Lambda-Funktionen, die sich in anderen Konten innerhalb derselben Region befinden, finden Sie unter Verketten von IAM-Rollen in Amazon Redshift.

Verwenden der JSON-Schnittstelle zwischen Amazon Redshift und AWS Lambda

Amazon Redshift verwendet eine gemeinsame Schnittstelle für alle Lambda-Funktionen, mit denen Amazon Redshift kommuniziert.

Die folgende Tabelle zeigt die Liste der Eingabefelder, die die designierten Lambda-Funktionen für die JSON-Payload erwarten können.

Feldname	Beschreibung	Wertebereich
request_id	Eine UUID (Universally Unique Identifier), die	Eine gültige UUID.

Feldname	Beschreibung	Wertebereich
	jede Aufrufanforderung eindeutig identifiziert.	
Cluster	Der vollständige Amazon-Ressourcenname (ARN) des Clusters.	Ein gültiger Cluster-ARN.
user	Der Name des Benutzers, der den Aufruf ausführt.	Ein gültiger Benutzername.
Datenbank	Der Name der -Datenbank, auf der die Abfrage ausgeführt wird.	Ein gültiger Datenbankname.
external_function	Der vollqualifizierte Name der externen Funktion, die den Aufruf ausführt.	Ein gültiger vollqualifizierter Funktionsname.
query_id	Die Abfrage-ID der Abfrage, die den Aufruf ausführt.	Eine gültige Abfrage-ID.
num_records	Die Anzahl der Argumente in der Nutzlast.	Ein Wert von 1 - 2^64.
arguments	Die Datennutzlast im angegebenen Format.	Die Daten im Array-Format müssen ein JSON-Array sein. Jedes Element ist ein Datensatz, der ein Array ist, wenn die Anzahl der Argumente größer als 1 ist. Durch die Verwendung eines Arrays behält Amazon Redshift die Reihenfolge der Datensätze in der Nutzlast bei.

Die Reihenfolge des JSON-Arrays bestimmt die Reihenfolge der Batch-Verarbeitung. Die Lambda Funktion muss die Argumente iterativ wiederholen und die genaue Anzahl von Datensätzen erzeugen. Im Folgenden sehen Sie ein Beispiel für eine Nutzlast.

```
{
  "request_id" : "23FF1F97-F28A-44AA-AB67-266ED976BF40",
  "cluster" : "arn:aws:redshift:xxxx",
  "user" : "adminuser",
  "database" : "db1",
  "external_function": "public.foo",
  "query_id" : 5678234,
  "num_records" : 4,
  "arguments" : [
     [ 1, 2 ],
     [ 3, null],
     null,
     [ 4, 6]
   ]
 }
```

Die Ausgabe der Lambda-Funktion sollte folgende Felder umfassen.

Feldname	Beschreibung	Wertebereich
Erfolg	Der Hinweis auf Erfolg oder Fehlschlag der Funktion.	Ein Wert von `"true"` oder `"false"`.
error_msg	Die Fehlermeldung, wenn der Erfolgswert `"false"` ist (wenn die Funktion fehlschlägt); andernfalls wird dieses Feld ignoriert.	Eine gültige Nachricht.
num_records	Die Anzahl der Datensätze in der Nutzlast.	Ein Wert von 1 - 2^64.

Feldname	Beschreibung	Wertebereich
results	Die Ergebnisse des Aufrufs im angegebenen Format.	N/A

Im Folgenden sehen Sie ein Beispiel für die Ausgabe der Lambda-Funktion.

```
{
  "success": true,   // true indicates the call succeeded
  "error_msg" : "my function isn't working",  // shall only exist when success != true
  "num_records": 4,      // number of records in this payload
  "results" : [
     1,
     4,
     null,
     7
   ]
}
```

Wenn Sie Lambda-Funktionen aus SQL-Abfragen aufrufen, stellt Amazon Redshift die Sicherheit der Verbindung mit den folgenden Überlegungen sicher:

- GRANT- und REVOKE-Berechtigungen. Weitere Informationen zur Sicherheit und zu Berechtigungen von UDFs finden Sie unter UDF-Sicherheit und Rechte.
- Amazon Redshift sendet nur den Mindestdatensatz an die dafür vorgesehene Lambda-Funktion.
- Amazon Redshift ruft nur die designierte Lambda-Funktion mit der angegebenen IAM-Rolle auf.

Beispiel für Verwendungszwecke von benutzerdefinierten Funktionen (UDFs)

Sie können benutzerdefinierte Funktionen verwenden, um Geschäftsprobleme zu lösen, indem Sie Amazon Redshift mit anderen Komponenten integrieren. Nachfolgend finden Sie einige Beispiel, wie andere UDFs für ihre Anwendungsfälle verwendet haben:

- Zugriff auf externe Komponenten mit Amazon Redshift Lambda UDFs – beschreibt, wie Amazon Redshift Lambda UDFs funktionieren und durch die Erstellung eines Lambda-UDFs geht.

- Übersetzen und analysieren Sie Text mithilfe von SQL-Funktionen mit Amazon Redshift, Amazon Translate und Amazon Comprehend – bietet vorgefertigte Amazon Redshift-Lambda-UDFs, die Sie mit wenigen Klicks installieren können, um Textfelder zu übersetzen, zu redigieren und zu analysieren.
- Greifen Sie von Amazon Redshift auf Amazon Location Service zu – beschreibt, wie Amazon Redshift Lambda UDFs zur Integration in Amazon Location Service verwendet wird.
- Datenaufgliederung in Token mit Amazon Redshift und Protegrity – beschreibt, wie Amazon Redshift Lambda UDFs in das Protegrity-Serverless-Produkt integriert werden kann.
- Amazon Redshift UDFs – eine Sammlung von Amazon-Redshift-SQL-, Lambda- und Python-UDFs.

Erstellen von gespeicherten Prozeduren in Amazon Redshift

Sie können eine in Amazon Redshift gespeicherte Prozedur mit der PostgreSQL-Prozedursprache PL/pgSQL definieren, um eine Reihe von SQL-Abfragen und logischen Operationen durchzuführen. Die Prozedur wird in der Datenbank gespeichert und steht allen Benutzern mit ausreichenden Datenbankrechten zur Verfügung.

Im Gegensatz zu einer benutzerdefinierten Funktion (UDF) kann eine gespeicherte Prozedur neben SELECT-Abfragen auch Data Definition Language (DDL) und Data Manipulation Language (DML) beinhalten. Eine gespeicherte Prozedur muss keinen Wert zurückgeben. Sie können prozedurale Sprache, einschließlich Schleifen und bedingte Ausdrücke, verwenden, um den logischen Ablauf zu steuern.

Weitere Informationen zu SQL-Befehlen zum Erstellen und Verwalten von gespeicherten Prozeduren finden Sie in den folgenden Themen:

- CREATE PROCEDURE
- ALTER PROCEDURE
- DROP PROCEDURE
- SHOW PROCEDURE
- CALL
- GRANT
- REVOKE
- ALTER DEFAULT PRIVILEGES

Themen

- Übersicht über gespeicherte Prozeduren in Amazon Redshift
- PL/pgSQL-Sprachreferenz

Übersicht über gespeicherte Prozeduren in Amazon Redshift

Gespeicherte Prozeduren werden üblicherweise verwendet, um die Logik für Datentransformation, Datenvalidierung und die geschäftsspezifische Logik zu verkapseln. Durch das Zusammenfassen

mehrerer SQL-Schritte in eine gespeicherte Prozedur können Sie die Austauschläufe zwischen Ihren Anwendungen und der Datenbank reduzieren.

Für eine fein abgestimmte Zugriffskontrolle können Sie zur Ausführung von Funktionen gespeicherte Prozeduren erstellen, ohne einem Benutzer Zugriff auf die zugrunde liegenden Tabellen zu gewähren. So kann beispielsweise nur der Eigentümer oder ein Superuser eine Tabelle kürzen, während ein Benutzer Schreibrechte benötigt, um Daten in eine Tabelle einzufügen. Anstatt einem Benutzer Rechte für die zugrunde liegenden Tabellen zu erteilen, können Sie eine gespeicherte Prozedur erstellen, die die Aufgabe ausführt. Anschließend erteilen Sie dem Benutzer Rechte zum Ausführen der gespeicherten Prozedur.

Eine gespeicherte Prozedur mit dem Sicherheitsattribut DEFINER wird mit den Rechten des Eigentümers der gespeicherten Prozedur ausgeführt. Standardmäßig hat eine gespeicherte Prozedur die INVOKER-Sicherheit, d. h. die Prozedur verwendet die Rechte des Benutzers, der die Prozedur aufruft.

Verwenden Sie den CREATE PROCEDURE-Befehl zum Erstellen einer gespeicherten Prozedur. Verwenden Sie den CALL-Befehl zum Ausführen einer Prozedur. Beispiele dazu folgen später in diesem Abschnitt.

> **Note**
>
> Einige Clients zeigen beim Erstellen einer in Amazon Redshift gespeicherten Prozedur möglicherweise den folgenden Fehler an.
>
> ```
> ERROR: 42601: [Amazon](500310) unterminated dollar-quoted string at or near "$$
> ```
>
> Dieser Fehler tritt auf, wenn der Client die CREATE PROCEDURE-Anweisung mit Semikolons zur Trennung von Anweisungen und dem Dollarzeichen ($) für Zitate nicht korrekt parsen kann. Dies führt dazu, dass nur ein Teil der Anweisung an den Amazon-Redshift-Server gesendet wird. Dieser Fehler kann oftmals umgangen werden, indem die `Run as batch`- oder die `Execute selected`-Option des Clients verwendet wird.
> Verwenden Sie beispielsweise mit dem Aginity-Client die Option `Run entire script as batch`. Bei Verwendung von SQL Workbench/J empfehlen wir Version 124. Bei Verwendung von SQL Workbench/J Version 125 sollten Sie ein anderes Trennzeichen verwenden, um das Problem zu umgehen.
> CREATE PROCEDURE enthält SQL-Anweisungen mit einem Semikolon (;). Wenn ein anderes Trennzeichen, wie ein Schrägstrich (/), festgelegt und am Ende der CREATE

PROCEDURE-Anweisung platziert wird, wird die Anweisung zur Verarbeitung an den Amazon-Redshift-Server gesendet. Im Folgenden sehen Sie ein Beispiel.

```
CREATE OR REPLACE PROCEDURE test()
AS $$
BEGIN
  SELECT 1 a;
END;
$$
LANGUAGE plpgsql
;
/
```

Weitere Informationen finden Sie unter Alternative Trennzeichen in der SQL Workbench/J-Dokumentation. Oder verwenden Sie einen Client mit besserer Unterstützung für das Parsen von CREATE PROCEDURE-Anweisungen, z. B. den Abfrage-Editor in der Amazon-Redshift-Konsole oder TablePlus.

Themen

- Benennen von gespeicherten Prozeduren
- Sicherheit und Berechtigungen für gespeicherte Prozeduren
- Rückgabe einer Ergebnismenge
- Verwalten von Transaktionen
- Aufspüren von Fehlern
- Protokollieren von gespeicherten Prozeduren
- Überlegungen zur Unterstützung für gespeicherte Prozeduren

Das folgende Beispiel zeigt eine Prozedur ohne Ausgabeparameter. Standardmäßig handelt es sich bei Parametern um Eingabe-(IN)-Parameter.

```
CREATE OR REPLACE PROCEDURE test_sp1(f1 int, f2 varchar)
AS $$
BEGIN
  RAISE INFO 'f1 = %, f2 = %', f1, f2;
END;
$$ LANGUAGE plpgsql;
```

```
call test_sp1(5, 'abc');
INFO: f1 = 5, f2 = abc
CALL
```

> **Note**
>
> Wenn Sie gespeicherte Prozeduren schreiben, empfehlen wir eine bewährte Methode zum Sichern sensibler Werte:
> Nehmen Sie keine Hartkodierung für sensible Informationen in der Logik der gespeicherten Prozedur vor.hardco Weisen Sie beispielsweise kein Benutzerkennwort in einer CREATE USER-Anweisung im Text einer gespeicherten Prozedur zu. Dies stellt ein Sicherheitsrisiko dar, da hartkodierte Werte als Schema-Metadaten in Katalogtabellen aufgezeichnet werden können. Übergeben Sie stattdessen mithilfe von Parametern sensible Werte wie Passwörter als Argumente an die gespeicherte Prozedur.
> Weitere Informationen zu gespeicherten Prozeduren finden Sie unter CREATE PROCEDURE und Erstellen von gespeicherten Prozeduren in Amazon Redshift. Weitere Informationen zu Katalogtabellen finden Sie unter Systemkatalogtabellen.

Das folgende Beispiel zeigt eine Prozedur mit Ausgabeparametern. Es gibt Eingabe (IN)-, Eingabe und Ausgabe (INOUT)- sowie Ausgabe (OUT)-Parameter.

```
CREATE OR REPLACE PROCEDURE test_sp2(f1 IN int, f2 INOUT varchar(256), out_var OUT
 varchar(256))
AS $$
DECLARE
  loop_var int;
BEGIN
  IF f1 is null OR f2 is null THEN
    RAISE EXCEPTION 'input cannot be null';
  END IF;
  DROP TABLE if exists my_etl;
  CREATE TEMP TABLE my_etl(a int, b varchar);
    FOR loop_var IN 1..f1 LOOP
        insert into my_etl values (loop_var, f2);
        f2 := f2 || '+' || f2;
    END LOOP;
  SELECT INTO out_var count(*) from my_etl;
END;
```

```
$$ LANGUAGE plpgsql;


call test_sp2(2,'2019');

         f2           | column2
----------------------+---------
 2019+2019+2019+2019 | 2
(1 row)
```

Benennen von gespeicherten Prozeduren

Wenn Sie eine Prozedur mit dem gleichen Namen und verschiedenen Datentypen von Eingabeparametern oder einer Signatur definieren, erstellen Sie eine neue Prozedur. Infolgedessen ist der Prozedurname überladen. Weitere Informationen finden Sie unter Überladen von Prozedurnamen. Amazon Redshift lässt keine Prozedurenüberladung aufgrund von Ausgabeparametern zu. Sie können nicht zwei Prozeduren mit dem gleichen Namen und den gleichen Datentypen von Eingabeparametern, jedoch unterschiedlichen Typen von Ausgabeparametern haben.

Der Eigentümer oder ein Superuser kann den Inhalt einer gespeicherten Prozedur durch einen neuen mit derselben Signatur ersetzen. Zum Ändern der Signatur oder des Rückgabetyps einer gespeicherten Prozedur, lassen Sie die gespeicherte Prozedur fallen und erstellen sie neu. Weitere Informationen erhalten Sie unter DROP PROCEDURE und CREATE PROCEDURE.

Sie können mögliche Namenskonflikte und unerwartete Ergebnisse verhindern, wenn Sie sich bei der Implementierung an Ihre Namenskonventionen halten. Da Prozedurnamen überladen werden können, kann es Konflikte mit existierenden und zukünftigen Amazon-Redshift-Prozedurnamen geben.

Überladen von Prozedurnamen

Eine Prozedur wird anhand Ihres Namens und ihrer Signatur identifiziert, wobei die Signatur die Anzahl und die Datentypen der Eingabeparameter ist. Solange sich zwei Funktionen in demselben Schema bezüglich ihrer Signatur unterscheiden, können sie denselben Namen haben. Mit anderen Worten: Sie können Prozedurnamen überladen.

Wenn Sie eine Prozedur ausführen, bestimmt die Abfrage-Engine anhand der Anzahl der von Ihnen angegebenen Parameter und ihrer Datentypen, welches Verfahren aufgerufen werden soll. Sie können das Überladen nutzen, um Prozeduren mit einer variablen Anzahl von Parametern bis zu der

im Befehl CREATE PROCEDURE erlaubten Grenze stimulieren. Weitere Informationen finden Sie unter CREATE PROCEDURE.

Verhindern von Namenskonflikten

Es wird empfohlen, dass Sie alle Prozeduren mit dem Präfix `sp_` benennen. Amazon Redshift reserviert das Präfix `sp_` ausschließlich für gespeicherte Prozeduren. Indem Sie Ihren Prozedurnamen das Präfix `sp_` hinzufügen, stellen Sie sicher, dass Ihr Prozedurname keine Konflikte mit vorhandenen oder zukünftigen Amazon-Redshift-Prozedurnamen auslöst.

Sicherheit und Berechtigungen für gespeicherte Prozeduren

Standardmäßig sind alle Benutzer zum Erstellen einer Prozedur berechtigt. Zum Erstellen einer Prozedur müssen Sie über das USAGE-Recht für die Sprache PL/pgSQL verfügen, die standardmäßig PUBLIC gewährt wird. Nur Superuser und Eigentümer sind standardmäßig berechtigt, eine Prozedur aufzurufen. Superuser können REVOKE USAGE auf PL/pgSQL für einen Benutzer ausführen, wenn sie verhindern möchten, dass der Benutzer eine gespeicherte Prozedur erstellt.

Damit Sie eine Prozedur aufrufen können, muss Ihnen das EXECUTE-Recht für die Prozedur erteilt werden. Standardmäßig wird dem Eigentümer der Prozedur und Superusern das EXECUTE-Recht für neue Prozeduren erteilt. Weitere Informationen finden Sie unter GRANT.

Der Benutzer, der eine Prozedur erstellt hat, ist standardmäßig der Eigentümer. Standardmäßig hat der Eigentümer für die Prozedur folgende Berechtigungen: CREATE, DROP und EXECUTE. Superuser verfügen über alle Berechtigungen.

Das SECURITY-Attribut steuert die Berechtigungen einer Prozedur für den Zugriff auf Datenbankobjekte. Wenn Sie eine gespeicherte Prozedur erstellen, können Sie das SECURITY-Attribut entweder auf DEFINER oder auf INVOKER einstellen. Wenn Sie den SECURTIY INVOKER bestimmen, verwendet die Prozedur die Berechtigungen des Benutzers, der die Prozedur aufruft. Wenn Sie den SECURITY DEFINER bestimmen, verwendet die Prozedur die Berechtigungen des Eigentümers der Prozedur. INVOKER ist die Standardeinstellung.

Da eine SECURITY-DEFINER-Prozedur mit den Rechten des Eigentümers ausgeführt wird, müssen Sie sicherstellen, dass die Prozedur nicht missbraucht werden kann. Um sicherzustellen, dass SECURITY-DEFINER-Prozeduren nicht missbraucht werden können, sollten Sie Folgendes tun:

- Gewähren Sie die Ausführung von SECURITY DEFINER-Prozeduren nur bestimmten Benutzern und nicht der Öffentlichkeit.

- Qualifizieren Sie alle Datenbankobjekte, auf die die Prozedur zugreifen muss, mit Schema-Namen. Verwenden Sie z. B. `myschema.mytable` anstatt nur `mytable` zu verwenden.
- Wenn Sie einen Objektnamen nicht anhand seines Schemas qualifizieren können, legen Sie `search_path` beim Erstellen der Prozedur mit der Option „SET“ fest.- Legen Sie `search_path` fest, um alle Schemata auszuschließen, die von nicht vertrauenswürdigen Benutzern beschreibbar sind. Dieser Ansatz verhindert, dass Aufrufer dieser Prozedur Objekte (z. B. Tabellen oder Ansichten) anlegen, die Objekte maskieren, die für die Verwendung durch die Prozedur vorgesehen sind. Weitere Informationen zur Option „SET“ finden Sie unter CREATE PROCEDURE.

Das folgende Beispiel legt `search_path` auf `admin` fest, um zu gewährleisten, dass auf die `user_creds`-Tabelle über das `admin`-Schema zugegriffen werden kann und nicht aus der Öffentlichkeit oder über irgendein anderes Schema im `search_path` des Aufrufers.

```
CREATE OR REPLACE PROCEDURE sp_get_credentials(userid int, o_creds OUT varchar)
AS $$
BEGIN
  SELECT creds INTO o_creds
  FROM user_creds
  WHERE user_id = $1;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER
-- Set a secure search_path
SET search_path = admin;
```

Rückgabe einer Ergebnismenge

Sie können eine Ergebnismenge zurückgeben unter Verwendung eines Cursors oder einer temporären Tabelle.

Rückgabe eines Cursors

Erstellen Sie für die Rückgabe eines Cursors eine Prozedur mit einem INOUT-Argument, das mit einem `refcursor`-Datentyp definiert ist. Wenn Sie die Prozedur aufrufen, benennen Sie den Cursor. Dann können Sie die Ergebnisse anhand des Namens vom Cursor abrufen.

Das folgende Beispiel erstellt eine Prozedur mit Namen `get_result_set` mit einem INOUT-Argument mit Namen `rs_out`. Sie verwendet den `refcursor`-Datentyp. Die Prozedur öffnet den Cursor mit einer SELECT-Anweisung.

```
CREATE OR REPLACE PROCEDURE get_result_set (param IN integer, rs_out INOUT refcursor)
AS $$
BEGIN
  OPEN rs_out FOR SELECT * FROM fact_tbl where id >= param;
END;
$$ LANGUAGE plpgsql;
```

Der folgende AUFRUF-Befehl öffnet den Cursor mit dem Namen `mycursor`. Verwenden Sie Cursor nur während Transaktionen.

```
BEGIN;
CALL get_result_set(1, 'mycursor');
```

Wenn der Cursor geöffnet ist, können Sie aus dem Cursor abrufen, wie das folgende Beispiel zeigt.

```
FETCH ALL FROM mycursor;

    id | secondary_id | name
-------+--------------+---------
     1 |            1 | Joe
     1 |            2 | Ed
     2 |            1 | Mary
     1 |            3 | Mike
(4 rows)
```

Schließlich wird die Transaktion entweder bestätigt oder zurückgesetzt.

```
COMMIT;
```

Ein von einer gespeicherten Prozedur zurückgegebener Cursor unterliegt denselben Einschränkungen und Leistungsüberlegungen, wie in DECLARE CURSOR beschrieben. Weitere Informationen finden Sie unter Einschränkungen für Cursors.

Das folgende Beispiel zeigt das Aufrufen der `get_result_set` gespeicherten Prozedur unter Verwendung eines `refcursor`-Datentyps von JDBC. Der wortgetreue `'mycursor'` (der

Cursorname) wird an den `prepareStatement` übergeben. Anschließend werden die Ergebnisse von dem aufgerufen `ResultSet`.

```
static void refcursor_example(Connection conn) throws SQLException {
    conn.setAutoCommit(false);
    PreparedStatement proc = conn.prepareStatement("CALL get_result_set(1,
 'mycursor')");
    proc.execute();
    ResultSet rs = statement.executeQuery("fetch all from mycursor");
    while (rs.next()) {
      int n = rs.getInt(1);
      System.out.println("n " + n);
    }
```

Verwenden einer temporären Tabelle

Um Ergebnisse zu erhalten, können Sie ein Kürzel einer temporären Tabelle zurückgeben, die Ergebniszeilen enthält. Der Client kann der gespeicherten Prozedur als Parameter einen Namen zuweisen. Innerhalb der gespeicherten Prozedur kann Dynamic SQL verwendet werden, um die temporäre Tabelle zu bearbeiten. Es folgt ein Beispiel.

```
CREATE PROCEDURE get_result_set(param IN integer, tmp_name INOUT varchar(256)) as $$
DECLARE
  row record;
BEGIN
  EXECUTE 'drop table if exists ' || tmp_name;
  EXECUTE 'create temp table ' || tmp_name || ' as select * from fact_tbl where id <= '
 || param;
END;
$$ LANGUAGE plpgsql;

CALL get_result_set(2, 'myresult');
 tmp_name
-----------
 myresult
(1 row)

SELECT * from myresult;
 id | secondary_id | name
----+--------------+------
  1 |            1 | Joe
  2 |            1 | Mary
```

```
  1 |            2 | Ed
  1 |            3 | Mike
(4 rows)
```

Verwalten von Transaktionen

Sie können eine gespeicherte Prozedur mit standardmäßigem Transaktionsverwaltungsverhalten oder nichtatomarem Verhalten erstellen.

Standardmodus der Transaktionsverwaltung für gespeicherte Prozeduren

Die standardmäßige Verhalten der automatischen Commits im Transaktionsmodus bewirkt, das jeder SQL-Befehl, der separat ausgeführt wird, einzeln übernommen wird. Der Aufruf einer gespeicherten Prozedur wird wie ein einzelner SQL-Befehl behandelt. Die SQL-Anweisungen innerhalb einer Prozedur verhalten sich so, als ob sie sich in einem Transaktionsblock befinden, der implizit mit dem Start des Aufrufs beginnt und mit dem Ende des Aufrufs endet. Ein verschachtelter Aufruf einer anderen Prozedur wird wie jede andere SQL-Anweisung behandelt und arbeitet im Kontext derselben Transaktion wie der Aufrufer. Weitere Informationen über automatisches Commit-Verhalten finden Sie unter Serialisierbare Isolierung.

Nehmen wir jedoch an, Sie rufen eine gespeicherte Prozedur aus einem von einem Benutzer angegebenen Transaktionsblock auf (definiert durch BEGIN...COMMIT). In diesem Fall werden alle Anweisungen in der gespeicherten Prozedur im Kontext der vom Benutzer angegebenen Transaktion ausgeführt. Die Prozedur wird beim Beenden nicht implizit übergeben. Der Anrufer steuert die Commit- oder Rollback-Prozedur.

Tritt beim Ausführen einer gespeicherten Prozedur ein Fehler auf, werden alle in der aktuellen Transaktion vorgenommenen Änderungen zurückgesetzt.

Sie können die folgenden Transaktionskontroll-Anweisungen in einer gespeicherten Prozedur verwenden:

- COMMIT – Sendet alle erledigten Aufgaben in der aktuellen Transaktion und beginnt implizit eine neue Transaktion. Weitere Informationen finden Sie unter COMMIT.
- ROLLBACK – Setzt die erledigten Aufgaben in der aktuellen Transaktion zurück und beginnt implizit eine neue Transaktion. Weitere Informationen finden Sie unter ROLLBACK.

TRUNCATE ist eine weitere Anweisung, die Sie von innerhalb einer gespeicherten Prozedur aus zur Transaktionsverwaltung verwenden können. In Amazon Redshift gibt TRUNCATE implizit einen

Commit aus. Dieses Verhalten wird auch im Kontext von gespeicherten Prozeduren beibehalten. Wird eine TRUNCATE-Anweisung innerhalb einer gespeicherten Prozedur ausgegeben, überträgt sie die aktuelle Transaktion und beginnt eine neue. Weitere Informationen finden Sie unter TRUNCATE.

Alle Anweisungen, die einer COMMIT-, ROLLBACK- oder TRUNCATE-Anweisung folgen, werden im Kontext einer neuen Transaktion ausgeführt. Dies geschieht so lange, bis eine COMMIT-, ROLLBACK- oder TRUNCATE-Anweisung aufgerufen oder die gespeicherte Prozedur beendet wird.

Wenn Sie eine COMMIT-, ROLLBACK- oder TRUNCATE-Anweisung aus einer gespeicherten Prozedur heraus verwenden, gelten die folgenden Einschränkungen:

- Wenn die gespeicherte Prozedur innerhalb eines Transaktionsblocks aufgerufen wird, kann sie keine COMMIT-, ROLLBACK- oder TRUNCATE-Anweisung ausgeben. Diese Einschränkung gilt innerhalb des eigenen Hauptteils der gespeicherten Prozedur sowie innerhalb aller verschachtelten Prozeduraufrufe.
- Wenn die gespeicherte Prozedur mit `SET config`-Optionen erstellt wird, kann sie keine COMMIT-, ROLLBACK- oder TRUNCATE-Anweisung ausgeben. Diese Einschränkung gilt innerhalb des eigenen Hauptteils der gespeicherten Prozedur sowie innerhalb aller verschachtelten Prozeduraufrufe.
- Jeder offene Cursor (explizit oder implizit) wird automatisch geschlossen, wenn eine COMMIT-, ROLLBACK- oder TRUNCATE-Anweisung verarbeitet wird. Einschränkungen für explizite oder implizite Cursor finden Sie unter Überlegungen zur Unterstützung für gespeicherte Prozeduren.

Darüber hinaus können Sie mit dynamischen SQL keine COMMIT- oder ROLLBACK-Anweisungen ausführen. TRUNCATE-Anweisungen können jedoch mit dynamischem SQL ausgeführt werden. Weitere Informationen finden Sie unter Dynamisches SQL.

Wenn Sie mit gespeicherten Prozeduren arbeiten, beachten Sie, dass die BEGIN- und END-Anweisungen in PL/pgSQL nur der Gruppierung dienen. Sie beginnen weder eine Transaktion, noch beenden sie eine. Weitere Informationen finden Sie unter Block.

Das folgende Beispiel zeigt das Transaktionsverhalten beim Aufruf einer gespeicherten Prozedur aus einem explizitenTransaktionsblock heraus. Die beiden Insert-Anweisungen, die von außerhalb der gespeicherten Prozedur sowie aus ihr heraus ausgegeben werden, sind alle Teil derselben Transaktion (3382). Die Transaktion wird bestätigt, wenn der Benutzer den expliziten Commit durchführt.

```
CREATE OR REPLACE PROCEDURE sp_insert_table_a(a int) LANGUAGE plpgsql
```

```
AS $$
BEGIN
  INSERT INTO test_table_a values (a);
END;
$$;

Begin;
  insert into test_table_a values (1);
  Call sp_insert_table_a(2);
  insert into test_table_a values (3);
Commit;

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  | pid |  type   |               stmt_text
--------+------+-----+---------+----------------------------------------
    103 | 3382 | 599 | UTILITY | Begin;
    103 | 3382 | 599 | QUERY   | insert into test_table_a values (1);
    103 | 3382 | 599 | UTILITY | Call sp_insert_table_a(2);
    103 | 3382 | 599 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3382 | 599 | QUERY   | insert into test_table_a values (3);
    103 | 3382 | 599 | UTILITY | COMMIT
```

Sehen wir uns im Gegenzug folgendes Beispiel an: Wenn dieselben Anweisungen von außerhalb eines expliziten Transaktionsblocks ausgegeben werden, und Autocommit der Sitzung auf ON gesetzt ist, wird jede Anweisung in einer eigenen Transaktion ausgeführt.

```
insert into test_table_a values (1);
Call sp_insert_table_a(2);
insert into test_table_a values (3);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  | pid |  type   |
            stmt_text
--------+------+-----+---------
+-----------------------------------------------------------------------------------------------
    103 | 3388 | 599 | QUERY   | insert into test_table_a values (1);
    103 | 3388 | 599 | UTILITY | COMMIT
```

```
    103 | 3389 | 599 | UTILITY | Call sp_insert_table_a(2);
    103 | 3389 | 599 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3389 | 599 | UTILITY | COMMIT
    103 | 3390 | 599 | QUERY   | insert into test_table_a values (3);
    103 | 3390 | 599 | UTILITY | COMMIT
```

Das folgende Beispiel gibt eine TRUNCATE-Anweisung heraus, nach dem Einfügen in `test_table_a`. Die TRUNCATE-Anweisung gibt einen impliziten Commit heraus, der die derzeitige Transaktion (3335) bestätigt und eine neue startet (3336). Die neue Transaktion wird beim Beenden der Prozedur bestätigt.

```
CREATE OR REPLACE PROCEDURE sp_truncate_proc(a int, b int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table_a values (a);
  TRUNCATE test_table_b;
  INSERT INTO test_table_b values (b);
END;
$$;

Call sp_truncate_proc(1,2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |  pid  |  type   |
                                        stmt_text
--------+------+-------+---------
+-----------------------------------------------------------------------------------------------
    103 | 3335 | 23636 | UTILITY | Call sp_truncate_proc(1,2);
    103 | 3335 | 23636 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3335 | 23636 | UTILITY | TRUNCATE test_table_b
    103 | 3335 | 23636 | UTILITY | COMMIT
    103 | 3336 | 23636 | QUERY   | INSERT INTO test_table_b values ( $1 )
    103 | 3336 | 23636 | UTILITY | COMMIT
```

Das folgende Beispiel gibt ein TRUNCATE aus einem verschachtelten Aufruf heraus. TRUNCATE bestätigt die bisher geleistete Arbeit in den äußeren und inneren Prozeduren einer Transaktion (3344). Es startet eine neue Transaktion (3345). Die neue Transaktion wird beim Beenden der äußeren Prozedur bestätigt.

```
CREATE OR REPLACE PROCEDURE sp_inner(c int, d int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO inner_table values (c);
  TRUNCATE outer_table;
  INSERT INTO inner_table values (d);
END;
$$;

CREATE OR REPLACE PROCEDURE sp_outer(a int, b int, c int, d int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO outer_table values (a);
  Call sp_inner(c, d);
  INSERT INTO outer_table values (b);
END;
$$;

Call sp_outer(1, 2, 3, 4);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |  pid  |  type   |
                                          stmt_text
--------+------+-------+---------
+-----------------------------------------------------------------------------------------------
    103 | 3344 | 23636 | UTILITY | Call sp_outer(1, 2, 3, 4);
    103 | 3344 | 23636 | QUERY   | INSERT INTO outer_table values ( $1 )
    103 | 3344 | 23636 | UTILITY | CALL sp_inner( $1 , $2 )
    103 | 3344 | 23636 | QUERY   | INSERT INTO inner_table values ( $1 )
    103 | 3344 | 23636 | UTILITY | TRUNCATE outer_table
    103 | 3344 | 23636 | UTILITY | COMMIT
    103 | 3345 | 23636 | QUERY   | INSERT INTO inner_table values ( $1 )
    103 | 3345 | 23636 | QUERY   | INSERT INTO outer_table values ( $1 )
    103 | 3345 | 23636 | UTILITY | COMMIT
```

Das folgende Beispiel zeigt, dass Cursor `cur1` geschlossen wurde, als die TRUNCATE-Anweisung durchgeführt wurde.

```
CREATE OR REPLACE PROCEDURE sp_open_cursor_truncate()
LANGUAGE plpgsql
```

```
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  open cur1;
  TRUNCATE table test_table_b;
  Loop
    fetch cur1 into rec;
    raise info '%', rec.c1;
    exit when not found;
  End Loop;
END
$$;

call sp_open_cursor_truncate();
ERROR: cursor "cur1" does not exist
CONTEXT: PL/pgSQL function "sp_open_cursor_truncate" line 8 at fetch
```

Im folgenden Beispiel wird eine TRUNCATE-Anweisung herausgegeben und kann nicht aus einem expliziten Transaktionsblock heraus aufgerufen werden.

```
CREATE OR REPLACE PROCEDURE sp_truncate_atomic() LANGUAGE plpgsql
AS $$
BEGIN
  TRUNCATE test_table_b;
END;
$$;

Begin;
  Call sp_truncate_atomic();
ERROR: TRUNCATE cannot be invoked from a procedure that is executing in an atomic
 context.
HINT: Try calling the procedure as a top-level call i.e. not from within an explicit
 transaction block.
Or, if this procedure (or one of its ancestors in the call chain) was created with SET
 config options, recreate the procedure without them.
CONTEXT: SQL statement "TRUNCATE test_table_b"
PL/pgSQL function "sp_truncate_atomic" line 2 at SQL statement
```

Das folgende Beispiel zeigt, dass ein Benutzer, der kein Superuser oder Besitzer einer Tabelle ist, eine TRUNCATE-Anweisung für die Tabelle ausgeben kann. Der Benutzer verwendet dafür eine `Security Definer`-gespeicherte Prozedur. Das Beispiel zeigt die folgenden Aktionen:

- Der Benutzer1 erstellt eine Tabelle `test_tbl`.
- Der Benutzer1 erstellt die gespeicherte Prozedur `sp_truncate_test_tbl`.
- Der Benutzer1 erteilt Benutzer2 die Berechtigung `EXECUTE` für die gespeicherte Prozedur.
- Der Benutzer2 führt die gespeicherte Prozedur aus, um Tabelle verkürzen `test_tbl`. Das Beispiel zeigt die Zeilenanzahl vor und nach dem Befehl `TRUNCATE`.

```
set session_authorization to user1;
create table test_tbl(id int, name varchar(20));
insert into test_tbl values (1,'john'), (2, 'mary');
CREATE OR REPLACE PROCEDURE sp_truncate_test_tbl() LANGUAGE plpgsql
AS $$
DECLARE
  tbl_rows int;
BEGIN
  select count(*) into tbl_rows from test_tbl;
  RAISE INFO 'RowCount before Truncate: %', tbl_rows;
  TRUNCATE test_tbl;
  select count(*) into tbl_rows from test_tbl;
  RAISE INFO 'RowCount after Truncate: %', tbl_rows;
END;
$$ SECURITY DEFINER;
grant execute on procedure sp_truncate_test_tbl() to user2;
reset session_authorization;


set session_authorization to user2;
call sp_truncate_test_tbl();
INFO:  RowCount before Truncate: 2
INFO:  RowCount after Truncate: 0
CALL
reset session_authorization;
```

Im folgenden Beispiel wird COMMIT zweimal ausgegeben. Mit der ersten COMMIT-Anweisung werden alle in der Transaktion 10363 erledigten Aufgaben gesendet und implizit die Transaktion 10364 gestartet. Transaktion 10364 wird über die zweite COMMIT-Anweisung gesendet.

```
CREATE OR REPLACE PROCEDURE sp_commit(a int, b int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table values (a);
  COMMIT;
  INSERT INTO test_table values (b);
  COMMIT;
END;
$$;

call sp_commit(1,2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;
 userid |  xid  | pid  |  type   |
                               stmt_text
--------+-------+------+---------
+-----------------------------------------------------------------------------------------------------
    100 | 10363 | 3089 | UTILITY | call sp_commit(1,2);
    100 | 10363 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10363 | 3089 | UTILITY | COMMIT
    100 | 10364 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10364 | 3089 | UTILITY | COMMIT
```

Im folgenden Beispiel wird eine ROLLBACK-Anweisung ausgegeben, wenn `sum_vals` größer als 2 ist. Die erste ROLLBACK-Anweisung setzt alle erledigten Aufgaben in Transaktion 10377 zurück und startet eine neue Transaktion 10378. Die Transaktion 10378 wird beim Beenden der Prozedur bestätigt.

```
CREATE OR REPLACE PROCEDURE sp_rollback(a int, b int) LANGUAGE plpgsql
AS $$
DECLARE
  sum_vals int;
BEGIN
  INSERT INTO test_table values (a);
  SELECT sum(c1) into sum_vals from test_table;
  IF sum_vals > 2 THEN
    ROLLBACK;
  END IF;

  INSERT INTO test_table values (b);
```

```
END;
$$;

call sp_rollback(1, 2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

userid |  xid  | pid  |  type   |
                              stmt_text
--------+-------+------+---------
+-----------------------------------------------------------------------------------------------------
    100 | 10377 | 3089 | UTILITY | call sp_rollback(1, 2);
    100 | 10377 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10377 | 3089 | QUERY   | SELECT sum(c1) from test_table
    100 | 10377 | 3089 | QUERY   | Undoing 1 transactions on table 133646 with current
 xid 10377 : 10377
    100 | 10378 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10378 | 3089 | UTILITY | COMMIT
```

Nichtatomarer Modus der Transaktionsverwaltung für gespeicherte Prozeduren

Eine gespeicherte Prozedur, die im Modus NONATOMIC erstellt wurde, hat ein anderes Transaktionskontrollverhalten als eine Prozedur, die im Standardmodus erstellt wurde. Ähnlich wie beim automatischen Commit-Verhalten von SQL-Befehlen außerhalb von gespeicherten Prozeduren wird jede SQL-Anweisung innerhalb einer NONATOMIC-Prozedur in einer eigenen Transaktion ausgeführt und automatisch übernommen. Wenn ein Benutzer einen expliziten Transaktionsblock innerhalb einer gespeicherten NONATOMIC-Prozedur beginnt, werden die SQL-Anweisungen innerhalb des Blocks nicht automatisch übernommen. Der Transaktionsblock steuert den Commit oder das Rollback der darin enthaltenen Anweisungen.

In gespeicherten NONATOMIC-Prozeduren können Sie mithilfe der Anweisung START TRANSACTION einen expliziten Transaktionsblock innerhalb der Prozedur öffnen. Wenn es jedoch bereits einen offenen Transaktionsblock gibt, hat diese Anweisung keinerlei Auswirkungen, da Amazon Redshift Untertransaktionen nicht unterstützt. Die vorherige Transaktion wird fortgesetzt.

Wenn Sie mit FOR-Cursor-Schleifen innerhalb einer NONATOMIC-Prozedur arbeiten, stellen Sie sicher, dass Sie einen expliziten Transaktionsblock öffnen, bevor Sie die Ergebnisse einer Abfrage durchlaufen. Andernfalls wird der Cursor geschlossen, wenn die SQL-Anweisung innerhalb der Schleife automatisch übernommen wird.

Einige Überlegungen bei der Verwendung des Modus NONATOMIC lauten wie folgt:

- Jede SQL-Anweisung innerhalb der gespeicherten Prozedur wird automatisch übernommen, wenn kein Transaktionsblock geöffnet und Autocommit für die Sitzung auf AN gesetzt ist.
- Sie können die Anweisung COMMIT/ROLLBACK/TRUNCATE ausgeben, um die Transaktion zu beenden, wenn die gespeicherte Prozedur innerhalb eines Transaktionsblocks aufgerufen wird. Dies ist im Standardmodus nicht möglich.
- Sie können die Anweisung START TRANSACTION ausgeben, um einen Transaktionsblock innerhalb der gespeicherten Prozedur zu starten.

Die folgenden Beispiele veranschaulichen das Transaktionsverhalten bei der Arbeit mit gespeicherten NONATOMIC Prozeduren. In der Sitzung für alle folgenden Beispiele ist Autocommit auf AN gesetzt.

Im folgenden Beispiel umfasst eine gespeicherte NONATOMIC-Prozedur zwei INSERT-Anweisungen. Wenn die Prozedur außerhalb eines Transaktionsblocks aufgerufen wird, wird jede INSERT-Anweisung innerhalb der Prozedur automatisch übernommen.

```
CREATE TABLE test_table_a(v int);
CREATE TABLE test_table_b(v int);

CREATE OR REPLACE PROCEDURE sp_nonatomic_insert_table_a(a int, b int) NONATOMIC AS
$$
BEGIN
    INSERT INTO test_table_a values (a);
    INSERT INTO test_table_b values (b);
END;
$$
LANGUAGE plpgsql;

Call sp_nonatomic_insert_table_a(1,2);

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1792 | 1073807554 | UTILITY | Call sp_nonatomic_insert_table_a(1,2);
      1 | 1792 | 1073807554 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1792 | 1073807554 | UTILITY | COMMIT
      1 | 1793 | 1073807554 | QUERY   | INSERT INTO test_table_b values ( $1 )
```

```
      1 | 1793 | 1073807554 | UTILITY | COMMIT
(5 rows)
```

Wenn die Prozedur jedoch innerhalb eines BEGIN..COMMIT-Blocks aufgerufen wird, sind alle Anweisungen Teil derselben Transaktion (xid=1799).

```
Begin;
  INSERT INTO test_table_a values (10);
  Call sp_nonatomic_insert_table_a(20,30);
  INSERT INTO test_table_b values (40);
Commit;

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |                stmt_text
--------+------+------------+---------+-----------------------------------------
      1 | 1799 | 1073914035 | UTILITY | Begin;
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_a values (10);
      1 | 1799 | 1073914035 | UTILITY | Call sp_nonatomic_insert_table_a(20,30);
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_b values (40);
      1 | 1799 | 1073914035 | UTILITY | COMMIT
(7 rows)
```

In diesem Beispiel befinden sich zwei INSERT-Anweisungen zwischen START TRANSACTION...COMMIT. Wenn die Prozedur außerhalb eines Transaktionsblocks aufgerufen wird, befinden sich die beiden INSERT-Anweisungen in derselben Transaktion (xid=1866).

```
CREATE OR REPLACE PROCEDURE sp_nonatomic_txn_block(a int, b int) NONATOMIC AS
$$
BEGIN
    START TRANSACTION;
    INSERT INTO test_table_a values (a);
    INSERT INTO test_table_b values (b);
    COMMIT;
END;
$$
LANGUAGE plpgsql;
```

```
Call sp_nonatomic_txn_block(1,2);

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1865 | 1073823998 | UTILITY | Call sp_nonatomic_txn_block(1,2);
      1 | 1866 | 1073823998 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1866 | 1073823998 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1866 | 1073823998 | UTILITY | COMMIT
(4 rows)
```

Wenn die Prozedur innerhalb eines BEGIN...COMMIT-Blocks aufgerufen wird, bewirkt START TRANSACTION innerhalb der Prozedur nichts, da bereits eine offene Transaktion vorhanden ist. Durch den COMMIT innerhalb der Prozedur wird die aktuelle Transaktion übernommen (xid=1876) und eine neue gestartet.

```
Begin;
  INSERT INTO test_table_a values (10);
  Call sp_nonatomic_txn_block(20,30);
  INSERT INTO test_table_b values (40);
Commit;

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1876 | 1073832133 | UTILITY | Begin;
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_a values (10);
      1 | 1876 | 1073832133 | UTILITY | Call sp_nonatomic_txn_block(20,30);
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1876 | 1073832133 | UTILITY | COMMIT
      1 | 1878 | 1073832133 | QUERY   | INSERT INTO test_table_b values (40);
      1 | 1878 | 1073832133 | UTILITY | COMMIT
(8 rows)
```

Dieses Beispiel veranschaulicht die Arbeit mit Cursor-Schleifen. Die Tabelle test_table_a weist drei Werte auf. Ziel ist es, die drei Werte zu durchlaufen und sie in die Tabelle test_table_b einzufügen. Wenn eine gespeicherte NONATOMIC-Prozedur auf folgende Weise erstellt wird, wird nach der Ausführung der INSERT-Anweisung in der ersten Schleife die Fehlermeldung „cursor "cur1" does not exist“ (Cursor „cur1“ ist nicht vorhanden) ausgegeben. Dies liegt daran, dass das Autocommit von INSERT den geöffneten Cursor schließt.

```
insert into test_table_a values (1), (2), (3);

CREATE OR REPLACE PROCEDURE sp_nonatomic_cursor() NONATOMIC
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  open cur1;
  Loop
    fetch cur1 into rec;
    exit when not found;
    raise info '%', rec.v;
    insert into test_table_b values (rec.v);
  End Loop;
END
$$;

CALL sp_nonatomic_cursor();

INFO:  1
ERROR:  cursor "cur1" does not exist
CONTEXT:  PL/pgSQL function "sp_nonatomic_cursor" line 7 at fetch
```

Damit die Cursor-Schleife funktioniert, setzen Sie sie zwischen START TRANSACTION...COMMIT.

```
insert into test_table_a values (1), (2), (3);

CREATE OR REPLACE PROCEDURE sp_nonatomic_cursor() NONATOMIC
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
```

```
BEGIN
  START TRANSACTION;
  open cur1;
  Loop
    fetch cur1 into rec;
    exit when not found;
    raise info '%', rec.v;
    insert into test_table_b values (rec.v);
  End Loop;
  COMMIT;
END
$$;

CALL sp_nonatomic_cursor();

INFO:  1
INFO:  2
INFO:  3
CALL
```

Aufspüren von Fehlern

Wenn eine Abfrage oder ein Befehl in einer gespeicherten Prozedur einen Fehler verursacht, werden nachfolgende Abfragen nicht ausgeführt und die Transaktion wird rückgängig gemacht. Sie können Fehler jedoch unter Verwendung eines AUSNAHMEBLOCKS beheben.

> **Note**
>
> Standardmäßig führt ein Fehler dazu, dass nachfolgende Abfragen nicht ausgeführt werden, auch wenn die gespeicherte Prozedur keine zusätzlichen Bedingungen enthält, die zu Fehlern führen.

```
[ <<label>> ]
[ DECLARE
  declarations ]
BEGIN
  statements
EXCEPTION
  WHEN OTHERS THEN
    statements
```

```
END;
```

Wenn eine Ausnahme auftritt und Sie einen Ausnahmebehandlungsblock hinzufügen, können Sie RAISE-Anweisungen und die meisten anderen PL/pgSQL-Anweisungen schreiben. Sie können beispielsweise eine Ausnahme mit einer benutzerdefinierten Nachricht auslösen oder einen Datensatz in eine Protokollierungstabelle einfügen.

Bei der Eingabe des Ausnahmebehandlungsblocks wird ein Rollback für die aktuelle Transaktion ausgeführt und eine neue Transaktion erstellt, um die Anweisungen im Block auszuführen. Wenn die Anweisungen im Block fehlerfrei ausgeführt werden, wird ein Commit der Transaktion ausgeführt und die Ausnahme wird erneut ausgelöst. Zuletzt wird die gespeicherte Prozedur beendet.

Die einige unterstützte Bedingung in einem Ausnahmeblock ist OTHERS, die auf jeden Fehlertyp außer Abfrageabbruch zutrifft. Tritt ein Fehler in einem Ausnahmehandhabungsblock auf, kann er auch von einem äußeren Ausnahmebehandlungsblock erfasst werden.

Ein Fehler innerhalb der NONATOMIC-Prozedur wird nicht erneut ausgelöst, wenn er durch einen Ausnahmeblock behandelt wird. Informationen zum Auslösen einer Ausnahme, die vom Ausnahmebehandlungsblock abgefangen wurde, siehe PL/pgSQL-Anweisung `RAISE`. Diese Anweisung ist nur in Ausnahmebehandlungsblöcken gültig. Weitere Informationen finden Sie unter RAISE.

Steuern, was nach einem Fehler in einer gespeicherten Prozedur geschieht, mit dem CONTINUE-Handler

Der `CONTINUE`-Handler ist eine Art Ausnahme-Handler, der den Ausführungsablauf innerhalb einer gespeicherten NONATOMIC-Prozedur steuert. Damit können Sie Ausnahmen erfassen und behandeln, ohne den vorhandenen Anweisungsblock zu beenden. Wenn in einer gespeicherten Prozedur ein Fehler auftritt, wird der Ablauf normalerweise unterbrochen und der Fehler wird an den Aufrufer zurückgegeben. In manchen Anwendungsfällen ist der Fehler jedoch nicht schwerwiegend genug, um eine Unterbrechung des Datenflusses zu rechtfertigen. Möglicherweise möchten Sie den Fehler sorgfältig behandeln, indem Sie eine Fehlerbehandlungslogik Ihrer Wahl in einer separaten Transaktion verwenden und dann weitere Anweisungen ausführen, die auf den Fehler folgen. Nachfolgend wird die Syntax dargestellt.

```
[ DECLARE
  declarations ]
BEGIN
  statements
EXCEPTION
```

```
  [ CONTINUE_HANDLER | EXIT_HANDLER ] WHEN OTHERS THEN
    handler_statements
END;
```

Es sind mehrere Systemtabellen verfügbar, die Informationen über verschiedene Arten von Fehlern bereitstellen. Weitere Informationen finden Sie unter STL_LOAD_ERRORS, STL_ERROR und SYS_STREAM_SCAN_ERRORS. Es gibt auch zusätzliche Systemtabellen, die Sie zur Fehlerbehebung verwenden können. Weitere Informationen dazu finden Sie unter Referenz zu Systemtabellen und Ansichten.

Beispiel

Das folgende Beispiel zeigt, wie Anweisungen im Ausnahmebehandlungsblock geschrieben werden. Die gespeicherte Prozedur verwendet das standardmäßige Transaktionsverwaltungsverhalten.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

CREATE OR REPLACE PROCEDURE update_employee_sp() AS
$$
BEGIN
    UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
    EXECUTE 'select invalid';
EXCEPTION WHEN OTHERS THEN
    RAISE INFO 'An exception occurred.';
    INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp();

INFO:  An exception occurred.
ERROR:  column "invalid" does not exist
CONTEXT:  SQL statement "select invalid"
PL/pgSQL function "update_employee_sp" line 3 at execute statement
```

Wenn wir in diesem Beispiel `update_employee_sp` aufrufen, wird die Informationsmeldung An exception occurred. (Eine Ausnahme ist aufgetreten.) ausgelöst und die Fehlermeldung wird ins Protokoll `employee_error_log` der Protokollierungstabelle eingefügt. Die ursprüngliche Ausnahme

wird erneut ausgelöst, bevor die gespeicherte Prozedur beendet wird. Die folgenden Abfragen zeigen Datensätze, die sich aus der Ausführung des Beispiels ergeben.

```
SELECT * from employee;

firstname | lastname
-----------+----------
 Tomas     | Smith

SELECT * from employee_error_log;

          message
-----------------------------------------------
 Error message: column "invalid" does not exist
```

Weitere Informationen über RAISE, einschließlich Formatierungshilfe und einer Liste der zusätzlichen Ebenen, finden Sie unter Unterstützte PL/pgSQL-Anweisungen.

Das folgende Beispiel zeigt, wie Anweisungen im Ausnahmebehandlungsblock geschrieben werden. Die gespeicherte Prozedur verwendet das NONATOMIC-Transaktionsverwaltungsverhalten.
In diesem Beispiel wird nach Abschluss des Prozeduraufrufs kein Fehler an den Aufrufer zurückgegeben. Die UPDATE-Anweisung wird aufgrund des Fehlers in der nächsten Anweisung nicht rückgängig gemacht. Die Informationsmeldung wird ausgelöst und die Fehlermeldung wird in die Protokollierungstabelle eingefügt.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

-- Create the SP in NONATOMIC mode
CREATE OR REPLACE PROCEDURE update_employee_sp_2() NONATOMIC AS
$$
BEGIN
    UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
    EXECUTE 'select invalid';
EXCEPTION WHEN OTHERS THEN
    RAISE INFO 'An exception occurred.';
    INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
END;
$$
LANGUAGE plpgsql;
```

```
CALL update_employee_sp_2();
INFO:  An exception occurred.
CALL

SELECT * from employee;

 firstname | lastname
-----------+----------
 Adam      | Smith
(1 row)

SELECT * from employee_error_log;

                    message
-----------------------------------------------
 Error message: column "invalid" does not exist
(1 row)
```

Dieses Beispiel veranschaulicht das Erstellen einer Prozedur mit zwei Unterblöcken. Wenn die gespeicherte Prozedur aufgerufen wird, wird der Fehler aus dem ersten Unterblock von ihrem Ausnahmebehandlungsblock behandelt. Nachdem der erste Unterblock abgeschlossen ist, fährt die Prozedur mit der Ausführung des zweiten Unterblocks fort. Aus dem Ergebnis geht vorher, dass nach Abschluss des Prozeduraufrufs kein Fehler ausgelöst wird. Die Operationen UPDATE und INSERT für die Tabelle employee wurden übernommen. Fehlermeldungen aus beiden Ausnahmeblöcken wurden in die Protokollierungstabelle eingefügt.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

CREATE OR REPLACE PROCEDURE update_employee_sp_3() NONATOMIC AS
$$
BEGIN
    BEGIN
        UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
        EXECUTE 'select invalid1';
    EXCEPTION WHEN OTHERS THEN
        RAISE INFO 'An exception occurred in the first block.';
        INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
    END;
    BEGIN
        INSERT INTO employee VALUES ('Edie','Robertson');
```

```
        EXECUTE 'select invalid2';
    EXCEPTION WHEN OTHERS THEN
        RAISE INFO 'An exception occurred in the second block.';
        INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
    END;
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp_3();
INFO:  An exception occurred in the first block.
INFO:  An exception occurred in the second block.
CALL

SELECT * from employee;

 firstname | lastname
-----------+-----------
 Adam      | Smith
 Edie      | Robertson
(2 rows)

SELECT * from employee_error_log;

                     message
-------------------------------------------------
 Error message: column "invalid1" does not exist
 Error message: column "invalid2" does not exist
(2 rows)
```

Das folgende Beispiel demonstriert, wie Sie den CONTINUE-Ausnahme-Handler nutzen. In diesem Beispiel werden zwei Tabellen erstellt und in einer gespeicherten Prozedur verwendet. Der CONTINUE-Handler steuert den Ausführungsablauf in einer gespeicherten Prozedur mit dem Transaktionsverwaltungsverhalten NONATOMIC.

```
CREATE TABLE tbl_1 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_1() NONATOMIC AS
$$
BEGIN
    INSERT INTO tbl_1 VALUES (1);
    -- Expect an error for the insert statement following, because of the invalid value
```

```
    INSERT INTO tbl_1 VALUES ("val");
    INSERT INTO tbl_1 VALUES (2);
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Rufen Sie die gespeicherte Prozedur auf:

```
CALL sp_exc_handling_1();
```

Der Ablauf ist wie folgt:

1. Ein Fehler tritt auf, weil versucht wird, einen inkompatiblen Datentyp in eine Spalte einzufügen. Die Steuerung geht an den EXCEPTION-Block über. Bei der Eingabe des Ausnahmebehandlungsblocks wird ein Rollback für die aktuelle Transaktion ausgeführt und eine neue Transaktion erstellt, um die Anweisungen darin auszuführen.
2. Wenn die Anweisungen in CONTINUE_HANDLER ohne Fehler ausgeführt werden, geht die Steuerung an die Anweisung über, die unmittelbar auf die Anweisung folgt, die die Ausnahme verursacht hat. (Wenn eine Anweisung in CONTINUE_HANDLER eine neue Ausnahme auslöst, können Sie sie mit einer Ausnahmebehandlungsroutine innerhalb des EXCEPTION-Blocks behandeln.)

Nachdem Sie die gespeicherte Beispielprozedur aufgerufen haben, enthalten die Tabellen die folgenden Datensätze:

- Wenn Sie `SELECT * FROM tbl_1;` ausführen, werden zwei Datensätze zurückgegeben. Diese enthalten die Werte 1 und 2.
- Wenn Sie `SELECT * FROM tbl_error_logging;` ausführen, wird ein Datensatz mit den folgenden Werten zurückgegeben: Aufgetretener Fehler, 42703, und Spalte „val“ ist in tbl_1 nicht vorhanden.

Das folgende zusätzliche Beispiel zur Fehlerbehandlung verwendet sowohl einen EXIT-Handler als auch einen CONTINUE-Handler. Es erstellt zwei Tabellen: eine Datentabelle und eine Protokollierungstabelle. Außerdem wird eine gespeicherte Prozedur erstellt, die die Fehlerbehandlung demonstriert:

```
CREATE TABLE tbl_1 (a int);
```

```
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_2() NONATOMIC AS
$$
BEGIN
    INSERT INTO tbl_1 VALUES (1);
    BEGIN
        INSERT INTO tbl_1 VALUES (100);
        -- Expect an error for the insert statement following, because of the invalid
 value
        INSERT INTO tbl_1 VALUES ("val");
        INSERT INTO tbl_1 VALUES (101);
    EXCEPTION EXIT_HANDLER WHEN OTHERS THEN
        INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
    END;
    INSERT INTO tbl_1 VALUES (2);
    -- Expect an error for the insert statement following, because of the invalid value
    INSERT INTO tbl_1 VALUES ("val");
    INSERT INTO tbl_1 VALUES (3);
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Nachdem Sie die gespeicherte Prozedur erstellt haben, rufen Sie sie wie folgt auf:

```
CALL sp_exc_handling_2();
```

Wenn im inneren Ausnahmeblock, der vom inneren Satz von BEGIN und END in Klammern eingeschlossen ist, ein Fehler auftritt, wird er vom EXIT-Handler behandelt. Alle Fehler, die im äußeren Block auftreten, werden vom CONTINUE-Handler behandelt.

Nachdem Sie die gespeicherte Beispielprozedur aufgerufen haben, enthalten die Tabellen die folgenden Datensätze:

- Wenn Sie `SELECT * FROM tbl_1;` ausführen, werden vier Datensätze mit den Werten 1, 2, 3 und 100 zurückgegeben.
- Wenn Sie `SELECT * FROM tbl_error_logging;` ausführen, werden zwei Datensätze zurückgegeben. Sie haben die folgenden Werte: Aufgetretener Fehler, 42703, und Spalte „val“ ist in tbl_1 nicht vorhanden.

Wenn die Tabelle tbl_error_logging nicht existiert, wird eine Ausnahme ausgelöst.

Das folgende Beispiel demonstriert, wie Sie den CONTINUE-Ausnahme-Handler mit der FOR-Schleife nutzen. In diesem Beispiel werden zwei Tabellen erstellt und in einer FOR-Schleife in einer gespeicherten Prozedur verwendet. Die FOR-Schleife ist eine Ergebnismengenvariante, was bedeutet, dass sie über die Ergebnisse einer Abfrage iteriert:

```
CREATE TABLE tbl_1 (a int);
INSERT INTO tbl_1 VALUES (1), (2), (3);
CREATE TABLE tbl_2 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_loop() NONATOMIC AS
$$
DECLARE
 rec RECORD;
BEGIN
    FOR rec IN SELECT a FROM tbl_1
    LOOP
        IF rec.a = 2 THEN
            -- Expect an error for the insert statement following, because of the
 invalid value
            INSERT INTO tbl_2 VALUES("val");
        ELSE
            INSERT INTO tbl_2 VALUES (rec.a);
        END IF;
    END LOOP;
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Rufen Sie die gespeicherte Prozedur auf:

```
CALL sp_exc_handling_loop();
```

Nachdem Sie die gespeicherte Beispielprozedur aufgerufen haben, enthalten die Tabellen die folgenden Datensätze:

- Wenn Sie `SELECT * FROM tbl_2;` ausführen, werden zwei Datensätze zurückgegeben. Diese enthalten die Werte 1 und 3.

- Wenn Sie `SELECT * FROM tbl_error_logging;` ausführen, wird ein Datensatz mit den folgenden Werten zurückgegeben: Aufgetretener Fehler, 42703, und Spalte „val“ ist in tbl_2 nicht vorhanden.

Hinweise zur Verwendung des CONTINUE-Handlers:

- Die Schlüsselwörter CONTINUE_HANDLER und EXIT_HANDLER können nur in gespeicherten NONATOMIC-Prozeduren verwendet werden.
- Die Schlüsselwörter CONTINUE_HANDLER und EXIT_HANDLER sind optional. EXIT_HANDLER ist die Standardeinstellung.

Protokollieren von gespeicherten Prozeduren

Einzelheiten über gespeicherte Prozeduren sind in den folgenden Systemtabellen und -ansichern protokolliert:

- SVL_STORED_PROC_CALL – Einzelheiten über Start- und Endzeit der gespeicherten Prozeduraufrufe werden protokolliert, und ob der Aufruf vor dem Abschluss abgebrochen wurde. Weitere Informationen finden Sie unter SVL_STORED_PROC_CALL.
- SVL_STORED_PROC_MESSAGES – Nachrichten in gespeicherten Prozeduren, die von der RAISE-Abfrage ausgegeben werden, werden mit der entsprechenden Protokollierungsebene erfasst. Weitere Informationen finden Sie unter SVL_STORED_PROC_MESSAGES.
- SVL_QLOG – Für jede Abfrage einer gespeicherten Prozedur wird die Abfrage-ID des Prozeduraufrufs protokolliert. Weitere Informationen finden Sie unter SVL_QLOG.
- STL_UTILITYTEXT – Gespeicherte Prozeduraufrufe werden nach Abschluss protokolliert. Weitere Informationen finden Sie unter STL_UTILITYTEXT.
- PG_PROC_INFO – Diese Systemkatalogansicht stellt Informationen über gespeicherte Prozeduren dar. Weitere Informationen finden Sie unter PG_PROC_INFO.

Überlegungen zur Unterstützung für gespeicherte Prozeduren

Bei der Verwendung von in Amazon Redshift gespeicherten Prozeduren gelten folgende Überlegungen.

Unterschiede zwischen Amazon Redshift und PostgreSQL bezüglich des Supports für gespeicherte Prozeduren

Folgende Unterschiede gibt es zwischen dem Support für gespeicherte Prozeduren in Amazon Redshift und PostgreSQL:

- Amazon Redshift unterstützt keine Subtransaktionen und bietet deshalb begrenzten Support für Ausnahmehandhabungsblöcke.

Überlegungen und Limits

Folgende Überlegungen gelten für in Amazon Redshift gespeicherte Prozeduren:

- Die maximale Anzahl von gespeicherten Prozeduren für eine Datenbank beträgt 10 000.
- Der Quellcode einer Prozedur darf maximal 2 MB betragen.
- Die maximale Anzahl expliziter und impliziter Cursor, die Sie gleichzeitig in einer Benutzersitzung öffnen können, ist eins. FOR-Loops, die über die Ergebnismenge einer SQL-Anweisung iterieren, öffnen implizite Cursor. Verschachtelte Cursor werden nicht unterstützt.
- Explizite und implizite Cursors verfügen über dieselben Einschränkungen bezüglich der Ergebnismenge wie Standard-Amazon-Redshift-Cursors. Weitere Informationen finden Sie unter Einschränkungen für Cursors.
- Die maximale Anzahl an Ebenen für verschachtelte Aufrufe ist 16.
- Die maximale Anzahl an Prozedurparametern ist 32 für Eingabe- und 32 für Ausgabeparameter.
- Die maximale Anzahl an Variablen in einer gespeicherten Prozedur ist 1.024.
- Jeder SQL-Befehl, der einen eigenen Transaktionskontext erfordert, wird innerhalb einer gespeicherten Prozedur nicht unterstützt. Beispiele sind unter anderem:
  - PREPARE
  - CREATE/DROP DATABASE
  - CREATE EXTERNAL TABLE
  - VACUUM
  - SET LOCAL
  - ALTER TABLE APPEND

- Der Aufruf der `registerOutParameter`-Methoden durch den Java Database Connectivity (JDBC)-Treiber wird nicht für den Datentyp `refcursor` unterstützt. Ein Beispiel für die Verwendung des `refcursor`-Datentyps finden Sie unter Rückgabe einer Ergebnismenge.

PL/pgSQL-Sprachreferenz

Gespeicherte Prozeduren in Amazon Redshift basieren auf der prozeduralen PostgreSQL-PL/pgSQL-Sprache. Es gibt jedoch ein paar wichtige Unterschiede. In dieser Referenz finden Sie Details zur PL/pgSQL-Syntax, so wie sie von Amazon Redshift implementiert wurde. Weitere Informationen zu PL/pgSQL finden Sie unter PL/pgSQL – Prozedurale SQL-Sprache in der PostgreSQL-Dokumentation.

Themen

Konventionen für die PL/pgSQL-Referenz

In diesem Abschnitt finden Sie die Konventionen zum Schreiben der Syntax für die gespeicherte prozedurale PL/pgSQL-Sprache.

Zeichen	Beschreibung
GROSSBUCHSTABEN	Wörter in Großbuchstaben sind Schlüsselwörter.
[ ]	Eckige Klammern bezeichnen optionale Argumente. Mehrere Argumente in eckigen Klammern zeigen an, dass Sie eine beliebige Anzahl der Argumente verwenden können. Argumente in eckigen Klammern, die jeweils in einer eigenen Zeile stehen, zeigen außerdem an, dass der Amazon-Redshift-Parser die Argumente in der Reihenfolge erwartet, in der sie in der Syntax aufgelistet sind.
{ }	Geschweifte Klammern zeigen an, dass Sie nur eines der Argumente verwenden können, die innerhalb der Klammern stehen.

Zeichen	Beschreibung
|	Pipe-Zeichen zeigen an, dass Sie zwischen den Argumenten wählen können.
Rote Kursivschrift	Wörter in roter Kursivschrift zeigen Platzhalter an. Sie müssen das kursiv formatierte rote Wort durch den entsprechenden Wert ersetzen.
. . .	Auslassungspunkte zeigen an, dass Sie das Element davor wiederholen können.
'	Wörter in einfachen Anführungszeichen müssen zusammen mit den Anführungszeichen verwendet werden.

Struktur von PL/pgSQL

PL/pgSQL ist eine prozedurale Sprache mit vielen der gleichen Konstrukte wie andere prozedurale Sprachen.

Themen

- Block
- Variablendeklaration
- Aliasdeklaration
- Integrierte Variablen
- Datensatztypen

Block

PL/pgSQL ist eine Sprache in Blockstruktur. Der gesamte Text einer Prozedur ist in einem Block definiert, der variable Deklarationen und PL/pgSQL-Anweisungen enthält. Eine Anweisung kann auch ein verschachtelter Block oder Subblock sein.

Deklarationen und Anweisungen müssen mit einem Doppelpunkt enden. Dem END-Schlüsselwort in einem Block oder Subblock muss ein Doppelpunkt folgen. Verwenden Sie keine Doppelpunkte nach den Schlüsselwörtern DECLARE und BEGIN.

Sie können alle Schlüsselwörter und Kennungen sowohl in Groß- als auch Kleinbuchstaben schreiben. Kennungen werden implizit in Kleinbuchstaben konvertiert, außer sie werden in doppelte Anführungszeichen eingeschlossen.

Ein doppelter Bindestrich (--) beginnt einen Kommentar, der bis zum Ende der Zeile erweitert wird. Ein /* beginnt einen Blockkommentar, der bis zum nächsten Vorkommen von */ erweitert wird. Blockkommentare können nicht verschachtelt werden. Sie können jedoch Kommentare mit doppelten Bindestrichen in einem Blockkommentar einschließen und ein doppelter Bindestrich kann die Blockkommentar-Trennzeichen /* und */ ausblenden.

Jede Anweisung im Anweisungsbereich eines Blocks kann ein Subblock sein. Sie können Subblöcke für die logische Gruppierung oder zum Lokalisieren von Variablen in eine kleine Gruppe von Anweisungen verwenden.

```
[ <<label>> ]
[ DECLARE
  declarations ]
BEGIN
  statements
END [ label ];
```

Die im Deklarationsbereich vor einem Block deklarierten Variablen werden jedes Mal zu ihren Standardwerten initialisiert, wenn der Block eingegeben wird. Mit anderen Worten, sie werden nicht nur einmal pro Funktionsaufruf initialisiert.

Es folgt ein Beispiel.

```
CREATE PROCEDURE update_value() AS $$
DECLARE
  value integer := 20;
BEGIN
  RAISE NOTICE 'Value here is %', value;  -- Value here is 20
  value := 50;
  --
  -- Create a subblock
  --
  DECLARE
    value integer := 80;
  BEGIN
    RAISE NOTICE 'Value here is %', value;  -- Value here is 80
  END;
```

```
  RAISE NOTICE 'Value here is %', value;  -- Value here is 50
END;
$$ LANGUAGE plpgsql;
```

Verwenden Sie eine Bezeichnung zur Identifizierung des Blocks für eine EXIT-Anweisung oder zur Qualifizierung der Namen der Variablen, die im Block deklariert wurden.

Verwechseln Sie nicht die Verwendung von BEGIN/END für die Gruppierung von Anweisungen in PL/pgSQL mit den Datenbankbefehlen für die Transaktionskontrolle. BEGIN und END dienen in PL/pgSQL lediglich der Gruppierung. Sie beginnen weder eine Transaktion, noch beenden sie eine.

Variablendeklaration

Deklarieren Sie alle Variablen in einem Block, mit Ausnahme der Loop-Variablen, im DECLARE-Bereich des Blocks. Variablen verwenden jeden gültigen Amazon-Redshift-Datentyp. Informationen zu unterstützten Datentypen finden Sie unter Datentypen.

Bei PL/pgSQL-Variablen kann es sich um jeden von Amazon Redshift unterstützten Datentyp handeln sowie `RECORD` und `refcursor`. Mehr über `RECORD` erfahren Sie unter Datensatztypen. Mehr über `refcursor` erfahren Sie unter Cursor.

```
DECLARE
name [ CONSTANT ] type [ NOT NULL ] [ { DEFAULT | := } expression ];
```

Im Folgenden finden Sie Beispiele für Variablendeklarationen.

```
customerID integer;
numberofitems numeric(6);
link varchar;
onerow RECORD;
```

Die Loop-Variable eines FOR-Loops, die über einen Bereich von Ganzzahlen iteriert, wird automatisch als Ganzzahlvariable deklariert.

Die DEFAULT-Klausel gibt, wenn angegeben, den Anfangswert an, welcher der Variablen beim Eingeben des Blocks zugewiesen wird. Wenn die DEFAULT-Klausel nicht angegeben wird, dann wird die Variable zum SQL-NULL-Wert initialisiert. Die CONSTANT-Option verhindert, dass die Variable zugewiesen wird, sodass ihr Wert für die Dauer des Blocks konstant bleibt. Wenn NOT NULL festgelegt wird, führt eine Zuweisung eines Null-Werts zu einem Laufzeitfehler. Für alle Variablen, die als NOT NULL deklariert sind, muss ein standardmäßiger Nicht-Null-Wert angegeben sein.

Der Standardwert wird bei jeder Eingabe des Blocks ausgewertet. Wenn Sie beispielsweise `now()` einer Variablen des Typs `timestamp` zuweisen, weist die Variable den Zeitpunkt des aktuellen Funktionsaufrufs auf und nicht den Zeitpunkt, als die Funktion vorkompiliert wurde.

```
quantity INTEGER DEFAULT 32;
url VARCHAR := 'http://mysite.com';
user_id CONSTANT INTEGER := 10;
```

Der `refcursor`-Datentyp ist der Datentyp von Cursor-Variablen in gespeicherten Prozeduren. Ein `refcursor`-Wert kann von innerhalb einer gespeicherten Prozedur zurückgegeben werden. Weitere Informationen finden Sie unter Rückgabe einer Ergebnismenge.

Aliasdeklaration

Wenn die Signatur der gespeicherten Prozedur den Argumentnamen weg lässt, können Sie einen Alias für das Argument deklarieren.

```
name ALIAS FOR $n;
```

Integrierte Variablen

Die folgenden integrierten Variablen werden unterstützt:

- FOUND
- SQLSTATE
- SQLERRM
- GET DIAGNOSTICS integer_var := ROW_COUNT;

FOUND ist eine spezielle Variable vom Typ Boolesch. FOUND beginnt in jedem Prozeduraufruf mit „false“. FOUND wird von den folgenden Anweisungen festgelegt:

- SELECT INTO

  Legt FOUND auf „true“ fest, wenn eine Zeile zurückgegeben wird, und auf „false“, wenn keine Zeile zurückgegeben wird.
- UPDATE, INSERT und DELETE

  Legt FOUND auf „true“ fest, wenn mindestens eine Zeile betroffen ist, und auf „false“, wenn keine Zeile betroffen ist.

- FETCH

  Legt FOUND auf „true“ fest, wenn eine Zeile zurückgegeben wird, und auf „false“, wenn keine Zeile zurückgegeben wird.

- FOR-Anweisung

  Legt FOUND auf „true“ fest, wenn die FOR-Anweisung einmal oder mehrmals iteriert, und ansonsten auf „false“. Dies gilt für alle drei Varianten der FOR-Anweisung: Ganzzahl-FOR-Loops, Datensatz-FOR-Loops und dynamische Datensatz-FOR-Loops.

  FOUND wird beim Beenden des FOR-Loops festgelegt. Innerhalb der Laufzeit des Loops wird FOUND nicht von der FOR-Anweisung modifiziert. Es kann jedoch durch die Ausführung anderer Anweisungen im Loop-Text geändert werden.

Es folgt ein Beispiel.

```
CREATE TABLE employee(empname varchar);
CREATE OR REPLACE PROCEDURE show_found()
AS  $$
DECLARE
  myrec record;
BEGIN
  SELECT INTO myrec * FROM employee WHERE empname = 'John';
  IF NOT FOUND THEN
    RAISE EXCEPTION 'employee John not found';
  END IF;
END;
$$ LANGUAGE plpgsql;
```

Innerhalb eines Ausnahmehandlers enthält die spezielle Variable SQLSTATE den Fehlercode, welcher der Ausnahme entspricht, die ausgelöst wurde. Die spezielle Variable SQLERRM enthält die mit der Ausnahme verbundene Fehlermeldung. Diese Variablen sind außerhalb von Ausnahmehandlern undefiniert und zeigen bei Verwendung einen Fehler an.

Es folgt ein Beispiel.

```
CREATE OR REPLACE PROCEDURE sqlstate_sqlerrm() AS
$$
BEGIN
  UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
```

```
  EXECUTE 'select invalid';
  EXCEPTION WHEN OTHERS THEN
  RAISE INFO 'error message SQLERRM %', SQLERRM;
  RAISE INFO 'error message SQLSTATE %', SQLSTATE;
END;
$$ LANGUAGE plpgsql;
```

ROW_COUNT wird mit dem Befehl GET DIAGNOSTICS verwendet. Sie zeigt die Anzahl der Spalten an, die vom letzten SQL-Befehl verarbeitet wurden, der an die SQL-Engine gesendet wurde.

Es folgt ein Beispiel.

```
CREATE OR REPLACE PROCEDURE sp_row_count() AS
$$
DECLARE
  integer_var int;
BEGIN
  INSERT INTO tbl_row_count VALUES(1);
  GET DIAGNOSTICS integer_var := ROW_COUNT;
  RAISE INFO 'rows inserted = %', integer_var;
END;
$$ LANGUAGE plpgsql;
```

Datensatztypen

Ein RECORD-Typ ist kein echter Datensatztyp, sondern nur ein Platzhalter. Variablen vom Datensatztyp übernehmen die tatsächliche Zeilenstruktur der Zeile, der sie während des SELECT- oder FOR-Befehls zugeordnet werden. Die Unterstruktur einer Datensatzvariablen kann sich bei jeder Zuordnung zu einem Wert ändern. Eine Datensatzvariable verfügt erst dann über eine Unterstruktur, wenn sie zum ersten Mal zugewiesen wird. Bei jedem Versuch auf ein darin enthaltenes Feld zuzugreifen, wird ein Laufzeitfehler zurückgegeben.

```
name RECORD;
```

Es folgt ein Beispiel.

```
CREATE TABLE tbl_record(a int, b int);
INSERT INTO tbl_record VALUES(1, 2);
CREATE OR REPLACE PROCEDURE record_example()
LANGUAGE plpgsql
AS $$
```

```
DECLARE
  rec RECORD;
BEGIN
  FOR rec IN SELECT a FROM tbl_record
  LOOP
    RAISE INFO 'a = %', rec.a;
  END LOOP;
END;
$$;
```

Unterstützte PL/pgSQL-Anweisungen

PL/pgSQL-Anweisungen erweitern SQL-Befehle mit prozeduralen Konstrukten, einschließlich Schleifen- und bedingten Ausdrücken, um den logischen Fluss zu steuern. Die meisten SQL-Befehle können verwendet werden, einschließlich der Data Manipulation Language (DML) wie COPY, UNLOAD und INSERT, und der Data Definition Language (DDL) wie CREATE TABLE. Eine Liste umfassender SQL-Befehle finden Sie unter SQL-Befehle. Darüber hinaus werden die folgenden PL/pgSQL-Anweisungen von Amazon Redshift unterstützt.

Themen

- Zuweisung
- SELECT INTO
- No-op
- Dynamisches SQL
- Ergebnis
- Bedingungen: IF
- Bedingungen: CASE
- Loops
- Cursor
- RAISE
- Transaktionskontrolle

Zuweisung

Die Zuweisungsanweisung ordnet einer Variablen einen Wert zu. Der Ausdruck muss einen einzelnen Wert zurückgeben.

```
identifier := expression;
```

Die Verwendung des nicht standardmäßigen = für die Anweisung, anstelle von `:=`, wird auch akzeptiert.

Wenn der Datentyp des Ausdrucks nicht mit dem Datentyp der Variablen übereinstimmt oder die Variable eine Größe oder Genauigkeit aufweist, wird der Ergebniswert implizit konvertiert.

Es folgen Beispiele.

```
customer_number := 20;
tip := subtotal * 0.15;
```

SELECT INTO

Die SELECT INTO-Anweisung weist das Ergebnis mehrerer Spalten (jedoch nur eine Zeile) einer Datensatzvariablen oder Liste von skalaren Variablen zu.

```
SELECT INTO target select_expressions FROM ...;
```

In der vorhergehenden Syntax kann *Ziel* eine Datensatzvariable oder eine durch Kommata getrennte Liste einfacher Variablen und Datensatzfelder sein. Die *select_expressions*-Liste und der Rest des Befehls sind die gleichen wie bei regulärem SQL.

Wenn eine Variablenliste als *Ziel* verwendet wird, müssen die ausgewählten Werte genau der Struktur des Ziels entsprechen oder ein Laufzeitfehler tritt auf. Wenn eine Datensatzvariable das Ziel ist, konfiguriert sie sich automatisch für den Zeilentyp der Abfrageergebnisspalten.

Die INTO-Klausel kann fast überall in der SELECT-Anweisung erscheinen. Sie wird normalerweise direkt nach der SELECT-Klausel oder direkt vor der FROM-Klausel angezeigt. Das heißt, sie erscheint direkt vor oder nach der *select_expressions*-Liste.

Wenn die Abfrage null Zeilen ausgibt, werden dem *Ziel* NULL-Werte zugewiesen. Wenn die Abfrage mehrere Zeilen ausgibt, wird die erste Zeile dem *Ziel* zugewiesen und der Rest wird verworfen. Sofern die Anweisung kein ORDER BY enthält, ist die erste Zeile nicht deterministisch.

Um festzustellen, ob die Anweisung mindestens eine Zeile zurückgegeben hat, verwenden Sie die spezielle FOUND-Variable.

```
SELECT INTO customer_rec * FROM cust WHERE custname = lname;
```

```
IF NOT FOUND THEN
  RAISE EXCEPTION 'employee % not found', lname;
END IF;
```

Um herauszufinden, ob ein Datensatzergebnis null ist, können Sie die IS NULL-Bedingung verwenden. Es gibt keine Möglichkeit, zu bestimmen, ob zusätzliche Zeilen verworfen wurden. Das folgende Beispiel behandelt den Fall, bei dem keine Zeilen zurückgegeben wurden.

```
CREATE OR REPLACE PROCEDURE select_into_null(return_webpage OUT varchar(256))
AS $$
DECLARE
  customer_rec RECORD;
BEGIN
  SELECT INTO customer_rec * FROM users WHERE user_id=3;
  IF customer_rec.webpage IS NULL THEN
    -- user entered no webpage, return "http://"
    return_webpage = 'http://';
  END IF;
END;
$$ LANGUAGE plpgsql;
```

No-op

Die no-op-Anweisung (`NULL;`) ist eine Platzhalteranweisung, die nichts tut. Eine no-op-Anweisung kann anzeigen, dass eine Verzweigung einer IF-THEN-ELSE-Kette leer ist.

```
NULL;
```

Dynamisches SQL

Zum Generieren dynamischer Befehle, die jedes Mal verschiedene Tabellen oder unterschiedliche Datentypen umfassen können, wenn sie von einer gespeicherten PL/pgSQL-Prozedur ausgeführt werden, verwenden Sie die `EXECUTE`-Anweisung.

```
EXECUTE command-string [ INTO target ];
```

Im vorhergehenden Beispiel ist *`command-string`* ein Ausdruck, der eine Zeichenfolge (vom Typ Text) zum Ergebnis hat, die den auszuführenden Befehl enthält. Dieser *`command-string`*-Wert wird an die SQL-Engine gesendet. Es wird keine Ersetzung von PL/pgSQL-Variablen für die

Befehlszeichenfolge vorgenommen. Die Werte von Variablen müssen in die Befehlszeichenfolge eingefügt werden, während sie erstellt wird.

> **Note**
>
> Sie können COMMIT- und ROLLBACK-Anweisungen nicht innerhalb von dynamischem SQL verwenden. Informationen zur Verwendung von COMMIT- und ROLLBACK-Anweisungen innerhalb eines gespeicherten Verfahrens finden Sie unter Verwalten von Transaktionen.

Bei der Arbeit mit dynamischen Befehlen müssen Sie sich häufig um das Escaping von einfachen Anführungszeichen kümmern. Wir empfehlen, dass Sie festen Text im Funktionstext mit der Dollaranführung in Anführungszeichen einschließen. Dynamische Werte, die in eine erstellte Abfrage eingefügt werden sollen, erfordern eine spezielle Vorgehensweisen, da sie selber Anführungszeichen enthalten können. Das folgende Beispiel setzt die Dollaranführung für die gesamte Funktion voraus, daher müssen keine doppelten Anführungszeichen verwendet werden.

```
EXECUTE 'UPDATE tbl SET '
  || quote_ident(colname)
  || ' = '
  || quote_literal(newvalue)
  || ' WHERE key = '
  || quote_literal(keyvalue);
```

Das vorhergehende Beispiel zeigt die Funktionen `quote_ident(text)` und `quote_literal(text)`. Dieses Beispiel leitet Variablen, die Spalten- und Tabellenkennungen enthalten, an die `quote_ident`-Funktion weiter. Es übermittelt auch Variablen, die Literalzeichenfolgen im erstellten Befehl enthalten, an die `quote_literal`-Funktion. Beide Funktionen unternehmen alle erforderlichen Schritte, um den in doppelten oder einfachen Anführungszeichen eingeschlossenen Eingabetext entsprechend zurückzugeben. Dabei sind alle eingebetteten speziellen Zeichen ordnungsgemäß mit einem Escape-Zeichen versehen.

Die Dollaranführung ist nur nützlich, wenn Sie Anführungszeichen für festen Text setzen. Schreiben Sie das vorhergehende Beispiel nicht in folgendem Format.

```
EXECUTE 'UPDATE tbl SET '
  || quote_ident(colname)
  || ' = $$'
  || newvalue
```

```
  || '$$ WHERE key = '
  || quote_literal(keyvalue);
```

Sie tun dies nicht, weil im Beispiel Fehler auftreten, wenn der Inhalt von `newvalue` zufällig $$ enthält. Dasselbe Problem gilt für alle anderen Dollaranführungstrennzeichen, die Sie auswählen können. Verwenden Sie die `quote_literal`-Funktion, um vorher nicht bekannten Text sicher mit Anführungszeichen zu versehen.

Ergebnis

Die RETURN-Anweisung wird aus einer gespeicherten Prozedur zum Aufrufer zurückgegeben.

```
RETURN;
```

Es folgt ein Beispiel.

```
CREATE OR REPLACE PROCEDURE return_example(a int)
AS $$
BEGIN
  FOR b in 1..10 LOOP
    IF b < a THEN
      RAISE INFO 'b = %', b;
    ELSE
      RETURN;
    END IF;
  END LOOP;
END;
$$ LANGUAGE plpgsql;
```

Bedingungen: IF

Die IF-Bedingungsanweisung kann folgende Formen in der PL/pgSQL-Sprache annehmen, die Amazon Redshift verwendet:

- IF ... THEN

  ```
  IF boolean-expression THEN
    statements
  END IF;
  ```

  Es folgt ein Beispiel.

```
IF v_user_id <> 0 THEN
  UPDATE users SET email = v_email WHERE user_id = v_user_id;
END IF;
```

- IF ... THEN ... ELSE

```
IF boolean-expression THEN
  statements
ELSE
  statements
END IF;
```

Es folgt ein Beispiel.

```
IF parentid IS NULL OR parentid = ''
THEN
  return_name = fullname;
  RETURN;
ELSE
  return_name = hp_true_filename(parentid) || '/' || fullname;
  RETURN;
END IF;
```

- IF ... THEN ... ELSIF ... THEN ... ELSE

Das Schlüsselwort ELSIF kann auch als ELSEIF angegeben werden.

```
IF boolean-expression THEN
  statements
[ ELSIF boolean-expression THEN
  statements
[ ELSIF boolean-expression THEN
  statements
    ...] ]
[ ELSE
  statements ]
END IF;
```

Es folgt ein Beispiel.

```
IF number = 0 THEN
```

```
  result := 'zero';
ELSIF number > 0 THEN
  result := 'positive';
ELSIF number < 0 THEN
  result := 'negative';
ELSE
  -- the only other possibility is that number is null
  result := 'NULL';
END IF;
```

Bedingungen: CASE

Die CASE-Bedingungsanweisung kann folgende Formen in der PL/pgSQL-Sprache annehmen, die Amazon Redshift verwendet:

- Einfaches CASE

```
CASE search-expression
WHEN expression [, expression [ ... ]] THEN
  statements
[ WHEN expression [, expression [ ... ]] THEN
  statements
  ... ]
[ ELSE
  statements ]
END CASE;
```

Eine einfache CASE-Anweisung stellt eine Bedingungsausführung basierend auf der Gleichheit von Operanden bereit.

Der *search-expression*-Wert wir ein Mal ausgewertet und aufeinanderfolgend mit jedem *Ausdruck* in der WHEN-Klausel verglichen. Bei einer Übereinstimmung werden die entsprechenden *Anweisungen* ausgeführt und die Kontrolle wird an die nächste Anweisung nach END CASE übergeben. Nachfolgende WHEN-Ausdrücke werden nicht ausgewertet. Bei keiner Übereinstimmung werden die ELSE-*Anweisungen* ausgeführt. Wenn ELSE jedoch nicht vorhanden ist, wird eine CASE_NOT_FOUND-Ausnahme ausgelöst.

Es folgt ein Beispiel.

```
CASE x
```

```
WHEN 1, 2 THEN
  msg := 'one or two';
ELSE
  msg := 'other value than one or two';
END CASE;
```

- Gesuchtes CASE

```
CASE
WHEN boolean-expression THEN
  statements
[ WHEN boolean-expression THEN
  statements
  ... ]
[ ELSE
  statements ]
END CASE;
```

Die gesuchte CASE-Anweisung stellt Bedingungsausführungen basierend auf der Wahrheit von booleschen Ausdrücken bereit.

Der *boolean-expression* jeder WHEN-Klausel wird dann ausgewertet, bis einer gefunden wird, der als Ergebnis „true“ zurück gibt. Anschließend werden die entsprechenden Anweisungen ausgeführt und die Kontrolle wird an die nächste Anweisung nach END CASE übergeben. Nachfolgende WHEN-*Ausdrücke* werden nicht ausgewertet. Wenn kein „true“-Ergebnis gefunden wird, werden die ELSE-*Anweisungen* ausgeführt. Wenn ELSE jedoch nicht vorhanden ist, wird eine CASE_NOT_FOUND-Ausnahme ausgelöst.

Es folgt ein Beispiel.

```
CASE
WHEN x BETWEEN 0 AND 10 THEN
  msg := 'value is between zero and ten';
WHEN x BETWEEN 11 AND 20 THEN
  msg := 'value is between eleven and twenty';
END CASE;
```

Loops

Loop-Anweisungen können folgende Formen in der PL/pgSQL-Sprache annehmen, die Amazon Redshift verwendet:

- Einfacher Loop

```
[<<label>>]
LOOP
  statements
END LOOP [ label ];
```

Ein einfacher Loop definiert einen bedingungslosen Loop, der unbegrenzt wiederholt wird, bis er von einer EXIT- oder RETURN-Anweisung beendet wird. Die optionale Bezeichnung kann von EXIT- und CONTINUE-Anweisungen in verschachtelten Loops verwendet werden, um anzugeben, auf welchen Loop sich die EXIT- und CONTINUE-Anweisungen beziehen.

Es folgt ein Beispiel.

```
CREATE OR REPLACE PROCEDURE simple_loop()
LANGUAGE plpgsql
AS $$
BEGIN
  <<simple_while>>
  LOOP
    RAISE INFO 'I am raised once';
    EXIT simple_while;
    RAISE INFO 'I am not raised';
  END LOOP;
  RAISE INFO 'I am raised once as well';
END;
$$;
```

- Exit-Loop

```
EXIT [ label ] [ WHEN expression ];
```

Wenn *Bezeichnung* nicht vorhanden ist, wird der innerste Loop beendet und die Anweisung als nächstes ausgeführt, die dem END LOOP folgt. Wenn *Bezeichnung* vorhanden ist, muss es sich um die Bezeichnung des aktuellen oder eines verschachtelten Loops oder Blocks der äußeren

Ebene handeln. Anschließend wird der benannte Loop oder Block beendet und die Kontrolle fährt mit der Anweisung nach dem entsprechenden END des Loops oder Blocks fort.

Wenn WHEN angegeben ist, wird der Loop nur beendet, wenn *Ausdruck* „true“ lautet. Andernfalls wird die Kontrolle an die Anweisung nach EXIT weitergeleitet.

Sie können EXIT mit allen Arten von Loops verwenden. Es ist nicht auf die Nutzung mit bedingungslosen Loops beschränkt.

Bei Verwendung mit einem BEGIN-Block übergibt EXIT die Kontrolle an die nächste Anweisung nach dem Ende des Blocks. Zu diesem Zweck muss eine Bezeichnung verwendet werden. Ein nicht gekennzeichnetes EXIT gilt nie als übereinstimmend mit einem BEGIN-Block.

Es folgt ein Beispiel.

```
CREATE OR REPLACE PROCEDURE simple_loop_when(x int)
LANGUAGE plpgsql
AS $$
DECLARE i INTEGER := 0;
BEGIN
  <<simple_loop_when>>
  LOOP
    RAISE INFO 'i %', i;
    i := i + 1;
    EXIT simple_loop_when WHEN (i >= x);
  END LOOP;
END;
$$;
```

- Continue-Loop

```
CONTINUE [ label ] [ WHEN expression ];
```

Wenn *Bezeichnung* nicht angegeben ist, springt die Ausführung zur nächsten Iteration des innersten Loops. Das heißt, alle im Loop-Text verbleibenden Anweisungen werden übersprungen. Die Kontrolle geht dann an den Loop-Kontrollausdruck (falls vorhanden) zurück, um festzulegen, ob eine weitere Loop-Iteration erforderlich ist. Wenn *Bezeichnung* vorhanden ist, wird die Bezeichnung des Loops angegeben, dessen Ausführung fortgesetzt wird.

Wenn WHEN angegeben ist, wird die nächste Iteration des Loops nur dann begonnen, wenn *Ausdruck* „true“ lautet. Andernfalls wird die Kontrolle an die Anweisung nach CONTINUE weitergeleitet.

Sie können CONTINUE mit allen Arten von Loops verwenden. Es ist nicht auf die Nutzung mit bedingungslosen Loops beschränkt.

```
CONTINUE mylabel;
```

- WHILE-Loop

```
[<<label>>]
WHILE expression LOOP
  statements
END LOOP [ label ];
```

Die WHILE-Anweisung wiederholt eine Reihenfolge von Anweisungen solange der *boolean-expression* als „true“ ausgewertet wird. Der Ausdruck wird kurz vor jedem Eintritt in den Loop-Text geprüft.

Es folgt ein Beispiel.

```
WHILE amount_owed > 0 AND gift_certificate_balance > 0 LOOP
  -- some computations here
END LOOP;

WHILE NOT done LOOP
  -- some computations here
END LOOP;
```

- FOR-Loop (Ganzzahlvariante)

```
[<<label>>]
FOR name IN [ REVERSE ] expression .. expression LOOP
  statements
END LOOP [ label ];
```

Der FOR-Loop (Ganzzahlvariante) erstellt einen Loop, der über einen Bereich von Ganzzahlwerten iteriert. Der Variablenname wird automatisch als Typ Ganzzahl definiert und existiert nur innerhalb

des Loops. Jede vorhandene Definition des Variablennamen wird innerhalb des Loops ignoriert. Die zwei Ausdrücke, welche die Unter- und Obergrenze des Bereichs angeben, werden beim Eintritt in den Loop einmal ausgewertet. Wenn Sie REVERSE angeben, wird der Schrittwert nach jeder Iteration eher subtrahiert als addiert.

Wenn die Untergrenze größer ist als die Obergrenze (oder kleiner, im REVERSE-Fall), wird der Loop-Text nicht ausgeführt. Es wird kein Fehler ausgegeben.

Wenn eine Bezeichnung einem FOR-Loop angefügt ist, können Sie unter Verwendung dieser Bezeichnung mit einem qualifizierten Namen auf die Ganzzahl-Loop-Variable verweisen.

Es folgt ein Beispiel.

```
FOR i IN 1..10 LOOP
  -- i will take on the values 1,2,3,4,5,6,7,8,9,10 within the loop
END LOOP;

FOR i IN REVERSE 10..1 LOOP
  -- i will take on the values 10,9,8,7,6,5,4,3,2,1 within the loop
END LOOP;
```

- FOR-Loop (Ergebnissatzvariante)

```
[<<label>>]
FOR target IN query LOOP
  statements
END LOOP [ label ];
```

Das *Ziel* ist eine Datensatzvariable oder eine durch Kommata getrennte Liste skalarer Variablen. Das Ziel wird jeder Zeile aufeinanderfolgend zugewiesen, die aus der Abfrage resultiert, und der Loop-Text wird für jede Abfrage ausgeführt.

Der FOR-Loop (Ergebnissatzvariante) ermöglicht einer gespeicherten Prozedur, die Ergebnisse einer Abfrage zu durchlaufen und diese Daten entsprechend zu bearbeiten.

Es folgt ein Beispiel.

```
CREATE PROCEDURE cs_refresh_reports() AS $$
DECLARE
  reports RECORD;
BEGIN
```

```
  FOR reports IN SELECT * FROM cs_reports ORDER BY sort_key LOOP
    -- Now "reports" has one record from cs_reports
    EXECUTE 'INSERT INTO ' || quote_ident(reports.report_name) || ' ' ||
 reports.report_query;
  END LOOP;
  RETURN;
END;
$$ LANGUAGE plpgsql;
```

- FOR-Loop mit dynamischem SQL

```
[<<label>>]
FOR record_or_row IN EXECUTE text_expression LOOP
  statements
END LOOP;
```

Der FOR-Loop mit dynamischem SQL ermöglicht einer gespeicherten Prozedur, die Ergebnisse einer dynamischen Abfrage zu durchlaufen und diese Daten entsprechend zu bearbeiten.

Es folgt ein Beispiel.

```
CREATE OR REPLACE PROCEDURE for_loop_dynamic_sql(x int)
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  query text;
BEGIN
  query := 'SELECT * FROM tbl_dynamic_sql LIMIT ' || x;
  FOR rec IN EXECUTE query
  LOOP
    RAISE INFO 'a %', rec.a;
  END LOOP;
END;
$$;
```

Cursor

Statt eine ganze Abfrage auf einmal auszuführen, können Sie einen Cursor einrichten. Ein Cursor kapselt eine Abfrage und liest jeweils wenige Zeilen des Abfrageergebnisses. Ein Grund hierfür ist das Vermeiden von Speicherüberlauf, wenn das Ergebnis eine große Anzahl von Zeilen enthält. Ein

weiterer Grund ist die Rückgabe einer Referenz an einen Cursor, den eine gespeicherte Prozedur erstellt hat. Dies erlaubt dem Aufrufer, die Zeilen zu lesen. Dieser Ansatz bietet eine effiziente Methode, große Zeilensätze von gespeicherten Prozeduren zurückzugeben.

Um Cursor in einer gespeicherten NONATOMIC-Prozedur zu verwenden, platzieren Sie die Cursor-Schleife zwischen START TRANSACTION...COMMIT.

Zum Einrichten eines Cursors deklarieren Sie zuerst eine Cursor-Variable. Jeder Zugriff auf Cursor in PL/pgSQL durchläuft Cursor-Variablen, die immer vom speziellen Datentyp sind `refcursor`. Ein `refcursor`-Datentyp hält einfach eine Referenz auf einen Cursor.

Sie können eine Cursor-Variable erstellen, indem Sie sie als Variable des Typs deklarieren `refcursor`. Alternativ können Sie die folgende Cursor-Deklarationssyntax verwenden.

```
name CURSOR [ ( arguments ) ] FOR query ;
```

Im vorhergehenden Beispiel ist *Argumente* (wenn angegeben) eine durch Kommata getrennte Liste von *Name-Datentyp*-Paaren, die jeweils Namen definieren, die in *Abfrage* durch Parameterwerte ersetzt werden sollen. Die tatsächlichen Werte, durch die diese Namen ersetzt werden sollen, werden später beim Öffnen des Cursors angegeben.

Es folgen Beispiele.

```
DECLARE
  curs1 refcursor;
  curs2 CURSOR FOR SELECT * FROM tenk1;
  curs3 CURSOR (key integer) IS SELECT * FROM tenk1 WHERE unique1 = key;
```

Alle drei dieser Variablen haben den Datentyp `refcursor`, aber die erste kann mit jeder Abfrage verwendet werden. Im Gegensatz dazu verfügt die zweite über eine vollständig angegebene Abfrage, die bereits an sie gebunden ist, und die letzte über eine an sie gebundene parametrisierte Abfrage. Der `key`-Wert wird durch einen Ganzzahl-Parameterwert ersetzt, wenn der Cursor geöffnet wird. Die Variable `curs1` gilt als ungebunden, weil sie nicht an eine bestimmte Abfrage gebunden ist.

Bevor Sie einen Cursor zum Abrufen von Zeilen verwenden können, muss er geöffnet werden. PL/pgSQL verfügt über drei Formen der OPEN-Anweisung. Zwei davon verwenden ungebundene Cursor-Variablen und die dritte eine gebundene Cursor-Variable:

- Öffnen zum Auswählen: Die Cursor-Variable wird geöffnet und die angegebene Abfrage zur Ausführung zugeteilt. Der Cursor darf nicht bereits geöffnet sein. Außerdem muss er als ungebundener Cursor deklariert worden sein (das heißt, als eine einfache `refcursor`-Variable). Die SELECT-Abfrage wird genauso wie andere SELECT-Anweisungen in PL/pgSQL behandelt.

  ```
  OPEN cursor_name FOR SELECT ...;
  ```

  Es folgt ein Beispiel.

  ```
  OPEN curs1 FOR SELECT * FROM foo WHERE key = mykey;
  ```

- Öffnen zum Ausführen: Die Cursor-Variable wird geöffnet und die angegebene Abfrage zur Ausführung zugeteilt. Der Cursor darf nicht bereits geöffnet sein. Außerdem muss er als ungebundener Cursor deklariert worden sein (das heißt, als eine einfache `refcursor`-Variable). Die Abfrage wird auf die gleiche Art und Weise als Zeichenfolgeausdruck angegeben wie im EXECUTE-Befehl. Dieser Ansatz bietet Flexibilität, sodass die Abfrage von einer Ausführung zur nächsten variieren kann.

  ```
  OPEN cursor_name FOR EXECUTE query_string;
  ```

  Es folgt ein Beispiel.

  ```
  OPEN curs1 FOR EXECUTE 'SELECT * FROM ' || quote_ident($1);
  ```

- Öffnen eines gebundenen Cursors: Diese Form von OPEN wird zum Öffnen einer Cursor-Variablen verwendet, deren Abfrage beim Deklarieren an sie gebunden wurde. Der Cursor darf nicht bereits geöffnet sein. Eine Liste der tatsächlichen Argumentwertausdrücke muss angezeigt werden, wenn und nur wenn der Cursor zur Verwendung von Argumenten deklariert wurde. Diese Werte werden in der Abfrage ersetzt.

  ```
  OPEN bound_cursor_name [ ( argument_values ) ];
  ```

  Es folgt ein Beispiel.

  ```
  OPEN curs2;
  OPEN curs3(42);
  ```

Nachdem ein Cursor geöffnet wurde, können Sie mit ihm arbeiten, indem Sie die Anweisungen verwenden, die im Folgenden beschrieben sind. Diese Anweisungen müssen nicht in derselben gespeicherten Prozedur ausgeführt werden, die den Cursor geöffnet hat. Sie können einen `refcursor`-Wert aus einer gespeicherten Prozedur zurück geben und den Aufrufer mit dem Cursor arbeiten lassen. Alle Portale werden am Transaktionsende implizit geschlossen. Daher können Sie einen `refcursor`-Wert für das Referenzieren eines offenen Cursors nur bis zum Ende der Transaktion verwenden.

- FETCH überträgt die nächste Zeile aus dem Cursor in ein Ziel. Dieses Ziel kann eine Zeilenvariable, eine Datensatzvariable oder eine durch Kommata getrennte Liste von einfachen Variablen wie bei SELECT INTO sein. Wie bei SELECT INTO können Sie in der speziellen FOUND-Variable überprüfen, ob eine Zeile bezogen wurde.

  ```
  FETCH cursor INTO target;
  ```

  Es folgt ein Beispiel.

  ```
  FETCH curs1 INTO rowvar;
  ```

- CLOSE schließt das Portal, das einem offenen Cursor zugrunde liegt. Sie können diese Anweisung verwenden, um Ressourcen noch vor dem Ende der Transaktion freizugeben. Sie können diese Anweisung auch verwenden, um die Cursor-Variable zu entblocken, damit sie wieder geöffnet werden kann.

  ```
  CLOSE cursor;
  ```

  Es folgt ein Beispiel.

  ```
  CLOSE curs1;
  ```

RAISE

Verwenden Sie die `RAISE level`-Anweisung zum Melden von Nachrichten und zum Auslösen von Fehlern.

```
RAISE level 'format' [, variable [, ...]];
```

Mögliche Ebenen sind NOTICE, INFO, LOG, WARNING und EXCEPTION. EXCEPTION meldet einen Fehler, der die aktuelle Transaktion in der Regel abbricht. Die anderen Ebenen generieren nur Nachrichten mit unterschiedlichen Prioritätsstufen.

Innerhalb der Formatzeichenfolge wird % durch die Zeichenfolgedarstellung des nächsten optionalen Arguments ersetzt. Schreiben Sie %% zum Übermitteln des Literals %. Derzeit müssen optionale Argumente einfache Variablen sein und nicht Ausdrücke, und das Format muss ein einfaches Zeichenfolgeliteral sein.

Im folgenden Beispiel ersetzt der Wert von `v_job_id` das % in der Zeichenfolge.

```
RAISE NOTICE 'Calling cs_create_job(%)', v_job_id;
```

Verwenden Sie die `RAISE`-Anweisung zum erneuten Auslösen der Ausnahme, die vom Ausnahmebehandlungsblock abgefangen wurde. Diese Anweisung ist nur in Ausnahmebehandlungsblöcken von gespeicherten Prozeduren im NONATOMIC-Modus gültig.

```
RAISE;
```

Transaktionskontrolle

Sie können mit Transaktionskontroll-Anweisungen in der PL/pgSQL-Sprache arbeiten, die Amazon Redshift verwendet. Informationen zur Verwendung von COMMIT-, ROLLBACK- und TRUNCATE-Anweisungen innerhalb eines gespeicherten Verfahrens finden Sie unter Verwalten von Transaktionen.

Verwenden Sie für gespeicherte Prozeduren im NONATOMIC-Modus `START TRANSACTION`, um einen Transaktionsblock zu starten.

```
START TRANSACTION;
```

> **Note**
>
> Die PL/pgSQL-Anweisung START TRANSACTION unterscheidet sich folgendermaßen vom SQL-Befehl START TRANSACTION:
>
> - In gespeicherten Prozeduren ist START TRANSACTION nicht gleichbedeutend mit BEGIN.

- Die PL/pgSQL-Anweisung unterstützt keine optionalen Schlüsselwörter für Isolationsstufe und Zugriffsberechtigungen.

Erstellen von materialisierten Ansichten in Amazon Redshift

In einer Data-Warehouse-Umgebung müssen Anwendungen häufig komplexe Abfragen für große Tabellen ausführen. Ein Beispiel sind SELECT-Anweisungen, die Aggregationen und Joins mit mehreren Tabellen für Tabellen ausführen, die Milliarden von Zeilen enthalten. Die Verarbeitung dieser Abfragen kann teuer sein, was die Systemressourcen und die Zeit bis zur Rückgabe der Ergebnisse betrifft.

Materialisierte Ansichten in Amazon Redshift bieten eine Möglichkeit, diese Probleme zu lösen. Eine materialisierte Ansicht enthält eine vorberechnete Ergebnismenge, die auf einer SQL-Abfrage über eine oder mehrere Basistabellen basiert. Sie können SELECT-Anweisungen verwenden, um eine materialisierte Ansicht abzufragen, so wie Sie auch andere Tabellen oder Ansichten für die Datenbank abfragen können. Amazon Redshift gibt die vorberechneten Ergebnisse aus der materialisierten Ansicht zurück, ohne überhaupt auf die Basistabellen zugreifen zu müssen. Aus Benutzersicht werden die Abfrageergebnisse viel schneller zurückgegeben als beim Abrufen derselben Daten aus den Basistabellen.

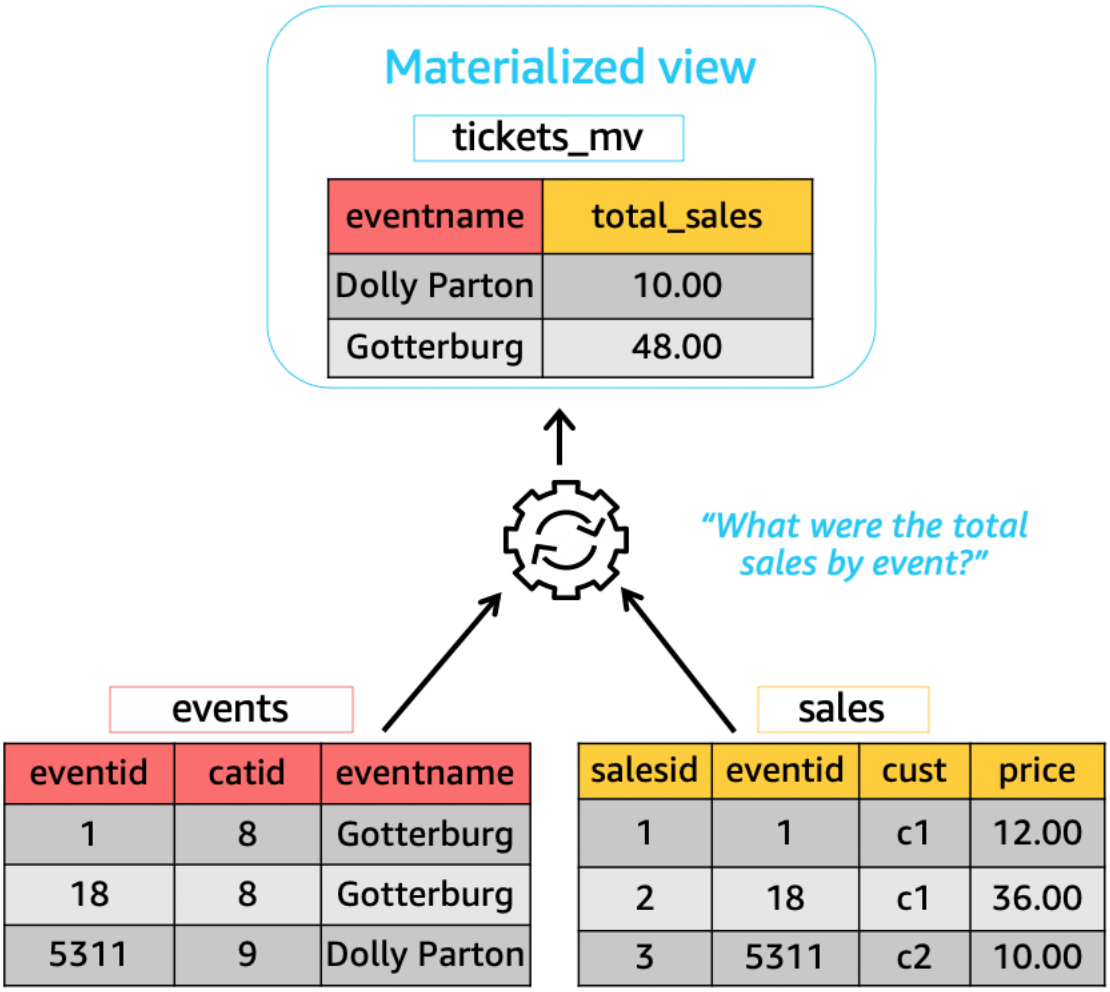
Materialisierte Ansichten sind besonders nützlich, um Abfragen zu beschleunigen, die vorhersehbar und wiederholbar sind. Anstatt ressourcenintensive Abfragen für großen Tabellen (z. B. Aggregates oder Mehrfach-Joins) durchzuführen, können Anwendungen eine materialisierte Ansicht abfragen und einen vorberechneten Ergebnissatz abrufen. Stellen Sie sich zum Beispiel das Szenario vor, in dem eine Reihe von Abfragen verwendet wird, um Dashboards zu füllen, z. B. Amazon. QuickSight Dieser Anwendungsfall ist ideal für eine materialisierte Ansicht, da die Abfragen vorhersehbar sind und sich immer wieder wiederholen.

Sie können eine materialisierte Ansicht in Bezug auf andere materialisierte Ansichten definieren. Verwenden Sie materialisierte Ansichten auf materialisierten Ansichten, um die Funktionen der materialisierten Ansichten zu erweitern. Bei diesem Ansatz spielt eine vorhandene materialisierte Ansicht die gleiche Rolle wie eine Basistabelle für die Abfrage zum Abrufen von Daten.

Dieser Ansatz ist besonders nützlich, wenn Sie vorberechnete Joins für unterschiedliche Aggregate- oder GROUP-BY-Optionen wiederverwenden möchten. Nehmen Sie beispielsweise eine materialisierte Ansicht, die Kundeninformationen (Millionen von Zeilen) mit Detailinformationen zur Artikelbestellung (Milliarden von Zeilen) verknüpft. Eine wiederholte On-Demand-Berechnung einer solchen Abfrage ist teuer. Sie können verschiedene GROUP-BY-Optionen für die materialisierten Ansichten verwenden, die über dieser materialisierten Ansicht erstellt wurden, und einen Join mit anderen Tabellen durchführen. Dies verkürzt die Datenverarbeitungszeit, die sonst für die wiederholte

Ausführung des teuren zugrundliegenden Joins gebraucht würde. Die STV_MV_DEPS-Tabelle zeigt die Abhängigkeiten einer materialisierten Ansicht auf anderen materialisierten Ansichten.

Wenn Sie eine materialisierte Ansicht erstellen, führt Amazon Redshift die vom Benutzer angegebene SQL-Anweisung aus, um die Daten aus der oder den Basistabelle(n) zu sammeln und den Ergebnissatz zu speichern. Die folgende Abbildung bietet einen Überblick über die materialisierte Ansicht `tickets_mv`, die eine SQL-Abfrage mithilfe von zwei Basistabellen (`events` und `sales`) definiert.



Sie können diese materialisierten Ansichten dann in Abfragen verwenden, um diese zu beschleunigen. Darüber hinaus kann Amazon Redshift diese Abfragen automatisch umschreiben, um materialisierte Ansichten zu verwenden, selbst wenn die Abfrage nicht explizit auf eine materialisierte Ansicht verweist. Das automatische Umschreiben von Abfragen ist besonders geeignet, um die Leistung zu verbessern, wenn Sie Ihre Abfragen nicht ändern können, um materialisierte Ansichten zu verwenden.

Um die Daten in der materialisierten Ansicht manuell zu aktualisieren, können Sie jederzeit die Anweisung REFRESH MATERIALIZED VIEW verwenden. Amazon Redshift identifiziert Änderungen, die in der oder den Basistabelle(n) stattgefunden haben, und wendet diese Änderungen dann auf die materialisierte Ansicht an. Da das automatische Umschreiben von Abfragen erfordert, dass materialisierte Ansichten auf dem neuesten Stand sind, stellen Sie als Eigentümer der materialisierten Ansicht sicher, dass materialisierte Ansichten aktualisiert werden, wenn sich eine Basistabelle ändert.

Amazon Redshift bietet einige Möglichkeiten, um materialisierte Ansichten für das automatische Umschreiben auf dem neuesten Stand zu halten. Sie können beispielsweise eine automatische Aktualisierung für die materialisierten Ansichten konfigurieren, damit diese aktualisiert werden, wenn ihre Basistabellen aktualisiert werden. Dieser automatische Aktualisierungsvorgang wird zu einem Zeitpunkt ausgeführt, zu dem Clusterressourcen verfügbar sind, um Unterbrechungen anderer Workloads zu minimieren. Da die Planung der automatischen Aktualisierung vom Workload abhängig ist, können Sie selbst entscheiden, wann Amazon Redshift Ihre materialisierten Ansichten aktualisieren soll. Für die Planung der Aktualisierung einer materialisierten Ansicht können Sie die Amazon-Redshift-Scheduler-API und die Konsolenintegration verwenden. Weitere Informationen zur Abfrageplanung finden Sie unter Planen einer Abfrage auf der Amazon-Redshift-Konsole.

Dies ist besonders nützlich, wenn ein Service Level Agreement (SLA) für up-to-date Daten aus einer materialisierten Ansicht erforderlich ist. Materialisierte Ansichten, die automatisch aktualisiert werden können, können auch manuell aktualisiert werden. Informationen zum Erstellen materialisierter Ansichten finden Sie unter CREATE MATERIALIZED VIEW.

Sie können SELECT-Anweisungen ausgeben, um eine materialisierte Ansicht abzufragen. Hinweise zum Abfragen materialisierter Ansichten finden Sie unter Abfragen einer materialisierten Ansicht. Der Ergebnissatz veraltet irgendwann, wenn Daten in den Basistabellen eingefügt, aktualisiert und gelöscht werden. Sie können die materialisierte Ansicht jederzeit aktualisieren, um sie mit den neuesten Änderungen aus den Basistabellen auf den neuesten Stand zu bringen. Informationen zum Aktualisieren materialisierter Ansichten finden Sie unter REFRESH MATERIALIZED VIEW.

Weitere Informationen zu SQL-Befehlen zum Erstellen und Verwalten von materialisierten Ansichten finden Sie in den folgenden Themen:

- CREATE MATERIALIZED VIEW
- ALTER MATERIALIZED VIEW
- REFRESH MATERIALIZED VIEW
- DROP MATERIALIZED VIEW

Informationen zu Systemtabellen und Ansichten zum Überwachen materialisierter Ansichten finden Sie in den folgenden Themen:

- STV_MV_INFO
- STL_MV_STATE
- SVL_MV_REFRESH_STATUS
- STV_MV_DEPS

Themen

Abfragen einer materialisierten Ansicht

Sie können eine materialisierte Ansicht in jeder beliebigen SQL-Abfrage verwenden, indem Sie den Namen der materialisierten Ansicht als Datenquelle (so wie beispielsweise eine Tabelle oder eine Standardansicht) referenzieren.

Wenn eine Abfrage auf eine materialisierte Ansicht zugreift, sieht sie nur die Daten, die in der materialisierten Ansicht gespeichert sind (bis zur letzten Aktualisierung). Daher erfasst die Abfrage möglicherweise nicht alle aktuellen Änderungen aus den entsprechenden Basistabellen der materialisierten Ansicht.

Wenn andere Benutzer die materialisierte Ansicht abfragen möchten, gewährt der Besitzer der materialisierten Ansicht diesen Benutzern die SELECT-Berechtigung. Die anderen Benutzer müssen nicht über die SELECT-Berechtigung für die zugrunde liegenden Basistabellen verfügen. Ebenso kann der Besitzer der materialisierten Ansicht die Berechtigung SELECT für andere Benutzer widerrufen, um sie am Abfragen der materialisierten Ansicht zu hindern.

Wenn der Besitzer der materialisierten Ansicht die SELECT-Berechtigung für die zugrundeliegenden Basistabellen nicht mehr besitzt:

- Der Besitzer kann die materialisierte Ansicht nicht mehr abfragen.
- Andere Benutzer, die über die SELECT-Berechtigung für die materialisierte Ansicht verfügen, können die materialisierte Ansicht nicht mehr abfragen.

Das folgende Beispiel fragt die materialisierte Ansicht `tickets_mv` ab. Weitere Informationen über den SQL-Befehl zum Erstellen einer materialisierten Ansicht finden Sie unter CREATE MATERIALIZED VIEW.

```
SELECT sold
FROM tickets_mv
WHERE catgroup = 'Concerts';
```

Da die Abfrageergebnisse vorberechnet sind, ist es nicht erforderlich, auf die zugrundeliegenden Tabellen (`category`, `event` und `sales`) zuzugreifen. Amazon Redshift kann die Ergebnisse direkt aus zurückgeben `tickets_mv`.

Automatisches Umschreiben von Abfragen zur Verwendung materialisierter Ansichten

Sie können das automatische Umschreiben von Abfragen materialisierter Ansichten in Amazon Redshift verwenden, damit Amazon Redshift Abfragen für die Verwendung von materialisierten Ansichten umschreibt. Dadurch werden Abfrage-Workloads sogar für solche Abfragen beschleunigt, die nicht explizit auf eine materialisierte Ansicht verweisen. Wenn Amazon Redshift Abfragen umschreibt, verwendet es ausschließlich aktuelle materialisierte Ansichten.

Nutzungshinweise

Um festzustellen, ob das automatische Umschreiben von Abfragen für eine Abfrage verwendet wird, können Sie den Abfrageplan oder STL_EXPLAIN überprüfen. Im Folgenden sehen Sie eine SELECT-Anweisung und die EXPLAIN-Ausgabe des ursprünglichen Abfrageplans.

```
SELECT catgroup, SUM(qtysold) AS sold
FROM category c, event e, sales s
WHERE c.catid = e.catid AND e.eventid = s.eventid
GROUP BY 1;

EXPLAIN
 XN HashAggregate  (cost=920021.24..920021.24 rows=1 width=35)
```

```
   ->  XN Hash Join DS_BCAST_INNER  (cost=440004.53..920021.22 rows=4 width=35)
         Hash Cond: ("outer".eventid = "inner".eventid)
         ->  XN Seq Scan on sales s  (cost=0.00..7.40 rows=740 width=6)
         ->  XN Hash  (cost=440004.52..440004.52 rows=1 width=37)
               ->  XN Hash Join DS_BCAST_INNER  (cost=0.01..440004.52 rows=1 width=37)
                     Hash Cond: ("outer".catid = "inner".catid)
                     ->  XN Seq Scan on event e  (cost=0.00..2.00 rows=200 width=6)
                     ->  XN Hash  (cost=0.01..0.01 rows=1 width=35)
                           ->  XN Seq Scan on category c  (cost=0.00..0.01 rows=1
 width=35)
```

Im Folgenden sehen Sie die EXPLAIN-Ausgabe nach erfolgreichem automatischen Umschreiben. Diese Ausgabe enthält einen Scan der materialisierten Ansicht im Abfrageplan, der Teile des ursprünglichen Abfrageplans ersetzt.

```
* EXPLAIN
     XN HashAggregate  (cost=11.85..12.35 rows=200 width=41)
       ->  XN Seq Scan on mv_tbl__tickets_mv__0 derived_table1  (cost=0.00..7.90
 rows=790 width=41)
```

Nur up-to-date (neue) materialisierte Ansichten werden für das automatische Umschreiben von Abfragen berücksichtigt, unabhängig von der Aktualisierungsstrategie, z. B. auto, geplant oder manuell. Daher gibt die ursprüngliche Abfrage Ergebnisse zurück up-to-date . Wenn in Abfragen explizit auf eine materialisierte Ansicht verwiesen wird, greift Amazon Redshift auf aktuell gespeicherte Daten in der materialisierten Ansicht zu. Diese Daten spiegeln möglicherweise nicht die aktuellen Änderungen aus den Basistabellen der materialisierten Ansicht wider.

Sie können das automatische Umschreiben von Abfragen für materialisierte Ansichten verwenden, die mit der Cluster-Version 1.0.20949 oder höher erstellt wurden.

Sie können das automatische Umschreiben von Abfragen auf Sitzungsebene stoppen, indem Sie „SET mv_enable_aqmv_for_session to FALSE“ verwenden.

Einschränkungen

Bei der Verwendung des automatischen Umschreibens von Abfragen materialisierter Ansichten gelten folgende Einschränkungen:

- Das automatische Umschreiben von Abfragen funktioniert mit materialisierten Ansichten, die keine der folgenden Elemente enthalten und nicht auf diese verweisen:
  - Unterabfragen

  - Left, Right oder Full Outer Joins
  - Set-Operationen
  - Alle Aggregatfunktionen, außer SUM, COUNT, MIN. MAX und AVG. (Dies sind die einzigen Aggregationsfunktionen, die mit dem automatischen Umschreiben von Abfragen arbeiten.)
  - Alle Aggregationsfunktionen mit DISTINCT
  - Alle Fensterfunktionen
  - SELECT-DISTINCT- oder HAVING-Klauseln
  - Externe Tabellen
  - Andere materialisierte Ansichten
- Beim automatischen Umschreiben von Abfragen werden SELECT-Abfragen umgeschrieben, die auf benutzerdefinierte Amazon-Redshift-Tabellen verweisen. Folgende Abfragen werden in Amazon Redshift nicht umgeschrieben:
  - CREATE-TABLE-AS-Anweisungen
  - SELECT INTO-Anweisungen
  - Abfragen zu Katalogen oder Systemtabellen
  - Abfragen mit Outer Joins oder einer SELECT-DISTINCT-Klausel
- Wenn eine Abfrage nicht automatisch umgeschrieben wird, überprüfen Sie, ob Sie über die SELECT-Berechtigung für die angegebene materialisierte Ansicht verfügen und ob die Option mv_enable_aqmv_for_session den Wert TRUE hat.

  Sie können auch feststellen, ob Ihre materialisierten Ansichten für das automatische Umschreiben von Abfragen berechtigt sind, indem Sie STV_MV_INFO überprüfen. Weitere Informationen finden Sie unter STV_MV_INFO.

Aktualisieren einer materialisierten Ansicht

Wenn Sie eine materialisierte Ansicht erstellen, spiegelt ihr Inhalt den Zustand der zugrundeliegenden Datenbanktabelle(n) zu diesem Zeitpunkt wider. Die Daten in der materialisierten Ansicht bleiben unverändert, auch wenn Anwendungen die Daten in den zugrundeliegenden Tabellen ändern. Um die Daten in der materialisierten Ansicht manuell zu aktualisieren, können Sie jederzeit die Anweisung `REFRESH MATERIALIZED VIEW` verwenden. Wenn Sie diese Anweisung verwenden, identifiziert Amazon Redshift Änderungen, die in der oder den Basistabellen stattgefunden haben, und wendet diese Änderungen auf die materialisierte Ansicht an.

Amazon Redshift besitzt zwei Strategien zum Aktualisieren einer materialisierten Ansicht:

- In vielen Fällen kann Amazon Redshift eine inkrementelle Aktualisierung durchführen. Bei einer inkrementellen Aktualisierung kann Amazon Redshift Änderungen an den Daten in den Basistabellen seit der letzten Aktualisierung schnell identifizieren und die Daten in der materialisierten Ansicht aktualisieren. Die inkrementelle Aktualisierung wird für die folgenden SQL-Konstrukte unterstützt, die in der Abfrage beim Definieren der materialisierten Ansicht verwendet werden:
  - Konstrukte mit den Klauseln SELECT, FROM, [INNER] JOIN, WHERE, GROUP BY oder HAVING.
  - Konstrukte mit Aggregationen, wie SUM, MIN, MAX, AVG und COUNT.
  - Die meisten integrierten SQL-Funktionen, insbesondere diejenigen, die unveränderlich sind, erzeugen bei den gleichen Eingabeargumenten immer die gleiche Ausgabe.

  Die inkrementelle Aktualisierung wird auch für eine materialisierte Ansicht unterstützt, die auf einer Datashare-Tabelle basiert.
- Wenn eine inkrementelle Aktualisierung nicht möglich ist, führt Amazon Redshift eine vollständige Aktualisierung durch. Eine Vollständige Aktualisierung führt die zugrundeliegende SQL-Anweisung erneut aus und ersetzt alle Daten in der materialisierten Ansicht.
- Amazon Redshift wählt automatisch die Aktualisierungsmethode für eine materialisierte Ansicht, abhängig von der SELECT-Abfrage, die zur Definition der materialisierten Ansicht verwendet wurde.

Das Aktualisieren einer materialisierten Ansicht auf einer materialisierten Ansicht ist kein kaskadierender Prozess. Mit anderen Worten, nehmen Sie an, Sie haben eine materialisierte Ansicht A, die von der materialisierten Ansicht B abhängt. In diesem Fall wird beim Aufrufen von REFRESH MATERIALIZED VIEW A A A mit der aktuellen Version von B aktualisiert, auch wenn B dies ist. out-of-date Um A auf den neuesten Stand zu bringen, müssen Sie vor der Aktualisierung von A zunächst B aktualisieren.

Das folgende Beispiel zeigt, wie Sie programmatisch einen Plan für eine vollständige Aktualisierung einer materialisierten Ansicht erstellen können. Um die materialisierte Ansicht v zu aktualisieren, aktualisieren Sie zuerst die materialisierte Ansicht u. Um die materialisierte Ansicht w zu aktualisieren, aktualisieren Sie zuerst die materialisierte Ansicht u und dann die materialisierte Ansicht v.

```
CREATE TABLE t(a INT);
```

```
CREATE MATERIALIZED VIEW u AS SELECT * FROM t;
CREATE MATERIALIZED VIEW v AS SELECT * FROM u;
CREATE MATERIALIZED VIEW w AS SELECT * FROM v;

WITH RECURSIVE recursive_deps (mv_tgt, lvl, mv_dep) AS
( SELECT trim(name) as mv_tgt, 0 as lvl, trim(ref_name) as mv_dep
  FROM stv_mv_deps
  UNION ALL
  SELECT R.mv_tgt, R.lvl+1 as lvl, trim(S.ref_name) as mv_dep
  FROM stv_mv_deps S, recursive_deps R
  WHERE R.mv_dep = S.name
)

SELECT mv_tgt, mv_dep from recursive_deps
ORDER BY mv_tgt, lvl DESC;

 mv_tgt | mv_dep
--------+--------
 v      | u
 w      | u
 w      | v
(3 rows)
```

Das folgende Beispiel zeigt eine informative Meldung, wenn Sie REFRESH MATERIALIZED VIEW für eine materialisierte Ansicht ausführen, die von einer materialisierten Ansicht abhängt. out-of-date

```
create table a(a int);
```

```
create materialized view b as select * from a;
```

```
create materialized view c as select * from b;
```

```
insert into a values (1);
```

```
refresh materialized view c;

INFO:  Materialized view c is already up to date.  However, it depends on another
 materialized view that is not up to date.
```

```
REFRESH MATERIALIZED VIEW b;
```

```
INFO:  Materialized view b was incrementally updated successfully.
```

```
REFRESH MATERIALIZED VIEW c;
INFO:  Materialized view c was incrementally updated successfully.
```

Beim inkrementellen Aktualisieren von materialisierten Ansichten in Amazon Redshift gelten derzeit folgende Einschränkungen.

Amazon Redshift unterstützt derzeit keine inkrementelle Aktualisierung für materialisierte Ansichten, die mit einer Abfrage mit den folgenden SQL-Elementen definiert sind:

- OUTER JOIN (RIGHT, LEFT oder FULL).
- Die Set-Operationen UNION, INTERSECT, EXCEPT und MINUS.
- Die Aggregatfunktionen MEDIAN, PERCENTILE_CONT, LISTAGG, STDDEV_SAMP, STDDEV_POP, APPROXIMATE COUNT, APPROXIMATE PERCENTILE sowie bitweise Aggregatfunktionen.

  > **Note**
  >
  > Die Aggregatfunktionen COUNT, SUM und AVG werden unterstützt.

- DISTINCT-Aggregatfunktionen, wie DISTINCT COUNT, DISTINCT SUM usw.
- Fensterfunktionen.
- Eine Abfrage, die temporäre Tabellen für die Abfrageoptimierung verwendet, z. B. das Optimieren allgemeiner Unterausdrücke.
- Unterabfragen.
- Externe Tabellen, die in der Abfrage, die die materialisierte Ansicht definiert, auf die folgenden Formate verweisen.
  - Delta Lake
  - Hudi

  Die inkrementelle Aktualisierung wird im Vorschau-Track für materialisierte Ansichten unterstützt, die mit anderen als den oben aufgeführten Formaten definiert wurden. Weitere Informationen zum Einrichten von Vorschau-Clustern finden Sie unter Erstellen eines Vorschau-Clusters im Amazon-Redshift-Verwaltungshandbuch. Informationen zum Einrichten von Vorschau-Arbeitsgruppen finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe im Amazon-Redshift-Verwaltungshandbuch.

Automatisches Aktualisieren einer materialisierten Ansicht

Amazon Redshift kann materialisierte Ansichten automatisch mit up-to-date Daten aus seinen Basistabellen aktualisieren, wenn materialisierte Ansichten mit der Option Autorefresh erstellt oder geändert werden. Die automatische Aktualisierung durch Amazon Redshift erfolgt nach Änderungen an den Basistabellen so schnell wie möglich.

Um die Aktualisierung der wichtigsten materialisierten Ansichten mit minimalen Auswirkungen auf aktive Workloads in Ihrem Cluster abzuschließen, berücksichtigt Amazon Redshift mehrere Faktoren. Zu diesen Faktoren zählen die aktuelle Systemlast, die für die Aktualisierung erforderlichen Ressourcen, die verfügbaren Clusterressourcen und die Häufigkeit der Verwendung der materialisierten Ansichten.

Amazon Redshift priorisiert Ihre Workloads gegenüber der automatischen Aktualisierung und stoppt möglicherweise die automatische Aktualisierung, um die Leistung des Benutzer-Workloads aufrechtzuerhalten. Dieser Ansatz kann die Aktualisierung einiger materialisierter Ansichten verzögern. In einigen Fällen ist möglicherweise ein deterministisches Aktualisierungsverhalten für Ihre materialisierten Ansichten erforderlich. Wenn dies der Fall ist, sollten Sie die manuelle Aktualisierung verwenden, wie unter REFRESH MATERIALIZED VIEW beschrieben, oder die geplante Aktualisierung mithilfe der Amazon-Redshift Scheduler-API-Vorgänge oder der Konsole.

Sie können die automatische Aktualisierung für materialisierte Ansichten mit CREATE MATERIALIZED VIEW festlegen. Sie können außerdem die AUTO-REFRESH-Klausel verwenden, um materialisierte Ansichten automatisch zu aktualisieren. Weitere Hinweise zur Erstellung von materialisierten Ansichten finden Sie unter CREATE MATERIALIZED VIEW. Sie können die automatische Aktualisierung für eine aktuelle materialisierte Ansicht aktivieren, indem Sie verwenden ALTER MATERIALIZED VIEW.

Beachten Sie Folgendes, wenn Sie materialisierte Ansichten aktualisieren:

- Sie können eine materialisierte Ansicht weiterhin explizit mit dem Befehl REFRESH MATERIALIZED VIEW aktualisieren, auch wenn Sie die automatische Aktualisierung für die materialisierte Ansicht nicht aktiviert haben.
- Amazon Redshift aktualisiert materialisierte Ansichten, die für externe Tabellen definiert sind, nicht automatisch.
- Für den Aktualisierungsstatus können Sie SVL_MV_REFRESH_STATUS überprüfen, was Abfragen aufzeichnet, die vom Benutzer initiiert oder automatisch aktualisiert wurden.

- Um REFRESH für ausschließlich neu berechnete materialisierte Ansichten auszuführen, benötigen Sie die CREATE-Berechtigung für Schemata. Weitere Informationen finden Sie unter GRANT.

Automatisierte materialisierte Ansichten

Materialisierte Ansichten sind ein leistungsfähiges Tool zur Verbesserung der Abfrageleistung in Amazon Redshift. Dazu wird eine im Voraus berechnete Ergebnismenge gespeichert. Ähnliche Abfragen müssen nicht jedes Mal dieselbe Logik erneut ausführen, da sie Datensätze aus der vorhandenen Ergebnismenge abrufen können. Entwickler und Analysten erstellen materialisierte Ansichten, nachdem sie ihre Workloads analysiert haben, um zu bestimmen, welche Abfragen davon profitieren würden und ob sich die Wartungskosten jeder materialisierten Ansicht lohnen. Wenn Workloads zunehmen oder sich ändern, müssen diese materialisierten Ansichten dahingehend überprüft werden, dass sie weiterhin greifbare Leistungsvorteile bieten.

Die Funktion Automated Materialized Views (AutoMV) in Redshift bietet die gleichen Leistungsvorteile wie vom Benutzer erstellte materialisierte Ansichten. Amazon Redshift überwacht den Workload kontinuierlich mithilfe von Machine Learning und erstellt neue materialisierte Ansichten, wenn diese nützlich sind. AutoMV gleicht die Kosten für das Erstellen und Aktualisieren materialisierter Ansichten auf den neuesten Stand gegenüber den erwarteten Vorteilen bei der Abfragelatenz aus. Das System überwacht auch zuvor erstellte AutoMVs und streicht sie, wenn sie nicht mehr vorteilhaft sind.

AutoVMs sind vom Verhalten und den Fähigkeiten her identisch zu von Benutzern erstellten materialisierten Ansichten. Sie werden automatisch und schrittweise unter Berücksichtigung der gleichen Kriterien und Einschränkungen aktualisiert. Genau wie bei materialisierten Ansichten, die von Benutzern erstellt wurden, identifiziert Automatisches Umschreiben von Abfragen zur Verwendung materialisierter Ansichten Abfragen, die von AutoMVs profitieren können, die vom System erstellt wurden. Diese Abfragen werden automatisch umgeschrieben, um die AutoMVS und so die Abfrageleistung zu verbessern. Entwickler müssen dabei die Abfragen nicht überarbeiten, um die Vorteile von AutoMV zu nutzen.

> **Note**
>
> Automatisierte materialisierte Ansichten werden sporadisch aktualisiert. Abfragen, die zur Verwendung von AutoMV umgeschrieben wurden, geben immer die neuesten Ergebnisse zurück. Wenn Redshift feststellt, dass Daten nicht aktuell sind, werden Abfragen nicht

umgeschrieben, um die Daten aus automatisierten materialisierten Ansichten zu lesen. Die Abfragen wählen vielmehr die neuesten Daten aus den Basistabellen aus.

Jede Workload mit wiederholt verwendeten Abfragen kann von AutoMV profitieren. Häufige Anwendungsfälle umfassen:

- Dashboards – Dashboards dienen häufig zur Bereitstellung schneller Ansichten von Key Business Indicators (KPIs), Ereignissen, Trends und anderen Kennzahlen. Sie haben oft ein gemeinsames Layout mit Diagrammen und Tabellen, zeigen jedoch unterschiedliche Ansichten zum Filtern oder für Vorgänge zur Dimensionsauswahl wie Drilldown. Dashboards haben oft eine gemeinsame Abfragereihe, die wiederholt mit unterschiedlichen Parametern verwendet wird. Dashboard-Abfragen können erheblich von automatisierten materialisierten Ansichten profitieren.
- Berichte – Berichtsabfragen können basierend auf den Geschäftsanforderungen und der Art des Berichts in verschiedenen zeitlichen Abständen geplant werden. Darüber lassen sie sich automatisiert oder bedarfsabhängig erstellen. Ein gemeinsames Merkmal von Berichtsabfragen ist, dass sie lange laufen und ressourcenintensiv sein können. Bei Verwendung von AutoMV müssen diese Abfragen nicht bei jeder Ausführung neu berechnet werden, wodurch die Ausführungszeit für jede Abfrage und die Ressourcenauslastung in Redshift verringert werden können.

Um automatisierte materialisierte Ansichten zu deaktivieren, aktualisieren Sie die Parametergruppe `auto_mv` auf `false`. Weitere Informationen finden Sie unter Amazon Redshift Parameter Groups (Amazon-Redshift-Parametergruppen) im Amazon-Redshift-Clusterverwaltungshandbuch.

SQL-Umfang und Überlegungen für automatisierte materialisierte Ansichten

- Eine automatisierte materialisierte Ansicht kann durch eine Abfrage oder Unterabfrage initiiert und erstellt werden, sofern sie eine `GROUP BY`-Klausel oder eine der folgenden Aggregatfunktionen enthält: SUM, COUNT, MIN, MAX oder AVG. Sie darf jedoch keines der folgenden Elemente enthalten:
  - Left, Right oder Full Outer Joins
  - Aggregierungsfunktionen außer SUM, COUNT, MIN, MAX und AVG. (Diese speziellen Funktionen arbeiten mit dem automatischen Umschreiben von Abfragen.)
  - Alle Aggregatfunktionen, die DISTINCT enthalten
  - Alle Fensterfunktionen
  - SELECT-DISTINCT- oder HAVING-Klauseln

- Andere materialisierte Ansichten

  Es ist nicht garantiert, dass eine Abfrage, die die Kriterien erfüllt, die Erstellung einer automatisierten materialisierten Ansicht initiiert. Das System bestimmt, von welchen Kandidaten eine Ansicht erstellt werden sollen, basierend auf deren erwartetem Nutzen für die Workloads und den Kosten der zu unterhaltenden Ressourcen, einschließlich der Kosten für das zu aktualisierende System. Jede resultierende materialisierte Ansicht ist durch automatisches Umschreiben von Abfragen nutzbar.

- Obwohl AutoMV durch eine Unterabfrage oder einzelne Abschnitte von Set-Operatoren initiiert werden könnte, enthält die resultierende materialisierte Ansicht keine Unterabfragen oder Set-Operatoren.
- Um festzustellen, ob AutoMV für Abfragen verwendet wurde, sehen Sie sich den EXPLAIN-Plan an und suchen Sie nach `%_auto_mv_%` in der Ausgabe. Weitere Informationen finden Sie unter EXPLAIN.
- Automatisierte materialisierte Ansichten werden in externen Tabellen wie Datashares und Verbundtabellen nicht unterstützt.

Einschränkungen für automatisierte materialisierte Ansichten

Im Folgenden finden Sie Einschränkungen für die Arbeit mit automatisierten materialisierten Ansichten:

- Maximale Anzahl der AutoVs – Die Grenze für automatisierte materialisierte Ansichten liegt bei 200 pro Datenbank im Cluster.
- Speicherplatz und Kapazität – Ein wichtiges Merkmal von AutoMV ist, dass es mit zusätzlichen Hintergrundzyklen ausgeführt wird, damit die Workloads der Benutzer nicht beeinträchtigt werden. Wenn der Cluster ausgelastet ist oder keinen Speicherplatz mehr hat, stellt AutoMV seine Aktivität ein. Insbesondere werden bei Erreichen von 80 % der gesamten Cluster-Kapazität keine neuen automatisierten materialisierten Ansichten erstellt. Sie können bei Erreichen von 90 % der gesamten Cluster-Kapazität entfernt werden, damit die Workloads der Benutzer ohne Leistungseinbußen fortgesetzt werden können. Weitere Informationen zur Ermittlung der Cluster-Kapazität finden Sie unter STV_NODE_STORAGE_CAPACITY.

Abrechnung für automatisierte materialisierte Ansichten

Die Amazon-Redshift-Funktion zur automatischen Optimierung erstellt und aktualisiert automatisierte materialisierte Ansichten. Für diesen Prozess fallen keine Gebühren für Rechenressourcen an. Die Speicherung automatisierter materialisierter Ansichten wird zum regulären Speichertarif in Rechnung gestellt. Weitere Informationen finden Sie unter Amazon Redshift – Preise.

Weitere Ressourcen

Der folgende Blogbeitrag enthält weitere Erläuterungen zu automatisierten materialisierten Ansichten. In dem Beitrag wird erörtert, wie diese erstellt, verwaltet und gelöscht werden. Darüber hinaus werden die zugrundeliegenden Algorithmen erläutert, die diese Entscheidungen steuern: Optimize your Amazon Redshift query performance with automated materialized views (Optimieren Sie Ihre Amazon-Redshift-Abfrageleistung mit automatisierten materialisierten Ansichten).

Dieses Video beginnt mit einer Erläuterung materialisierter Ansichten und zeigt, wie diese die Leistung verbessern und Ressourcen sparen können. Im weiteren Verlauf werden automatisierte materialisierte Ansichten in einer Prozessablaufanimation und einer Live-Demonstration eingehend erörtert.

Verwenden einer benutzerdefinierten Funktion (UDF) in einer materialisierten Ansicht

Sie können eine skalare UDF in einer materialisierten Ansicht in Amazon Redshift verwenden. Definieren Sie diese entweder in Python oder SQL und verweisen Sie in der Definition der materialisierten Ansicht darauf.

Verweisen auf eine UDF in einer materialisierten Ansicht

Das folgende Verfahren erläutert, wie Sie UDFs, die einfache arithmetische Vergleiche ausführen, in einer Definition einer materialisierten Ansicht verwenden.

1. Erstellen Sie eine Tabelle zur Verwendung in der Definition der materialisierten Ansicht.

   ```
   CREATE TABLE base_table (a int, b int);
   ```

2. Erstellen Sie eine skalare benutzerdefinierte Funktion in Python, die einen booleschen Wert zurückgibt, der angibt, ob eine Ganzzahl größer als eine Vergleichsganzzahl ist.

```
CREATE OR REPLACE FUNCTION udf_python_bool(x1 int, x2 int) RETURNS bool IMMUTABLE
AS $$
  return x1 > x2
$$ LANGUAGE plpythonu;
```

Erstellen Sie optional eine funktionell ähnliche UDF mit SQL, die Sie zum Vergleichen der Ergebnisse mit den ersten Ergebnissen verwenden können.

```
CREATE OR REPLACE FUNCTION udf_sql_bool(int, int) RETURNS bool IMMUTABLE
AS $$
  select $1 > $2;
$$ LANGUAGE SQL;
```

3. Erstellen Sie eine materialisierte Ansicht, die aus der von Ihnen erstellten Tabelle auswählt und auf die UDF verweist.

```
CREATE MATERIALIZED VIEW mv_python_udf AS SELECT udf_python_bool(a, b) AS a FROM
 base_table;
```

Optional können Sie eine materialisierte Ansicht erstellen, die auf die SQL-UDF verweist.

```
CREATE MATERIALIZED VIEW mv_sql_udf AS SELECT udf_sql_bool(a, b) AS a FROM
 base_table;
```

4. Fügen Sie der Tabelle Daten hinzu und aktualisieren Sie die materialisierte Ansicht.

```
INSERT INTO base_table VALUES (1,2), (1,3), (4,2);
```

```
REFRESH MATERIALIZED VIEW mv_python_udf;
```

Optional können Sie die materialisierte Ansicht aktualisieren, die auf die SQL-UDF verweist.

```
REFRESH MATERIALIZED VIEW mv_sql_udf;
```

5. Fragen Sie Daten aus Ihrer materialisierten Ansicht ab.

```
SELECT * FROM mv_python_udf ORDER BY a;
```

Die Ergebnisse der Abfrage sehen wie folgt aus:

```
a
-----
false
false
true
```

Dies gibt `true` für den letzten Satz von Werten zurück, da der Wert für Spalte a (4) größer als der Wert für Spalte b (2) ist.

6. Optional können Sie die materialisierte Ansicht abfragen, die auf die SQL-UDF verweist. Die Ergebnisse für die SQL-Funktion stimmen mit den Ergebnissen der Python-Version überein.

```
SELECT * FROM mv_sql_udf ORDER BY a;
```

Die Ergebnisse der Abfrage sehen wie folgt aus:

```
a
-----
false
false
true
```

Dies gibt `true` für den letzten Satz von Werten zurück, die verglichen werden sollen.

7. Verwenden Sie eine DROP-Anweisung mit CASCADE, um die benutzerdefinierte Funktion und die materialisierte Ansicht, die darauf verweist, zu löschen.

```
DROP FUNCTION udf_python_bool(int, int) CASCADE;
```

```
DROP FUNCTION udf_sql_bool(int, int) CASCADE;
```

Streaming-Erfassung

Die Streaming-Erfassung ermöglicht das Erfassen von Stream-Daten mit geringer Latenz und hoher Geschwindigkeit aus Amazon Kinesis Data Streams und Amazon Managed Streaming for Apache Kafka in einer von Amazon Redshift bereitgestellten Ansicht oder einer materialisierten

Amazon-Redshift-Serverless-Ansicht. Es wird weniger Zeit für den Datenzugriff benötigt und die Speicherkosten werden gesenkt. Sie können die Streaming-Erfassung für Ihren Amazon-Redshift-Cluster oder für Amazon Redshift Serverless konfigurieren und mithilfe von SQL-Anweisungen eine materialisierte Ansicht erstellen, wie unter Erstellen von materialisierten Ansichten in Amazon Redshift beschrieben. Danach können Sie mithilfe einer Aktualisierung der materialisierten Ansicht Hunderte von Megabyte Daten pro Sekunde erfassen. Das ermöglicht eine schnellen Zugriff auf externe Daten, die schnell aktualisiert werden.

Datenfluss

Ein von Amazon Redshift bereitgestellter Cluster oder eine Amazon-Redshift-Serverless-Arbeitsgruppe ist der Stream-Verbraucher. Eine materialisierte Ansicht ist der Eingangsbereich für Daten, die aus dem Stream gelesen und bei ihrer Ankunft verarbeitet werden. Beispielsweise können JSON-Werte mit bekanntem SQL konsumiert und den Datenspalten der materialisierten Ansicht zugeordnet werden. Wenn die materialisierte Ansicht aktualisiert wird, verwendet Redshift Daten von zugewiesenen Kinesis-Daten-Shards oder Kafka-Partitionen, bis die Ansicht die Parität mit dem für den Kinesis-Stream oder zuletzt `SEQUENCE_NUMBER` für das Kafka-Thema erreicht. `Offset` Die nachfolgende materialisierte Ansicht aktualisiert die gelesenen Daten aus der letzten `SEQUENCE_NUMBER` der vorherigen Aktualisierung, bis Parität mit den Stream- oder Themendaten erreicht wird.

Anwendungsfälle für Streaming-Erfassung

Die Anwendungsfälle für die Streaming-Erfassung von Amazon Redshift beinhalten die Arbeit mit Daten, die kontinuierlich generiert (gestreamt) werden und innerhalb eines kurzen Zeitraums (Latenz) nach ihrer Generierung verarbeitet werden müssen. Dies wird als Analytik in nahezu Echtzeit bezeichnet. Die Datenquellen können variieren und beinhalten IOT-Geräte, Systemtelemetriedaten oder Clickstream-Daten von einer ausgelasteten Website oder Anwendung.

Überlegungen zur Streaming-Erfassung

Im Folgenden finden Sie wichtige Überlegungen und bewährte Methoden in Bezug auf die Leistung und Abrechnung bei der Einrichtung Ihrer Umgebung für die Streaming-Erfassung.

- Verwendung und Aktivierung der automatischen Aktualisierung – Abfragen zur automatischen Aktualisierung für eine oder mehrere materialisierte Ansichten werden wie jeder andere Benutzer-Workload behandelt. Bei der automatischen Aktualisierung werden Daten bei ihrer Ankunft aus dem Stream geladen.

Die automatische Aktualisierung kann für eine materialisierte Ansicht, die für die Streaming-Erfassung erstellt wurde, explizit aktiviert werden. Geben Sie hierfür `AUTO REFRESH` in der Definition der materialisierten Ansicht an. Standardmäßig wird die manuelle Aktualisierung verwendet. Um die automatische Aktualisierung für eine vorhandene materialisierte Ansicht für die Streaming-Erfassung festzulegen, können Sie die automatische Aktualisierung durch Ausführung von `ALTER MATERIALIZED VIEW` aktivieren. Weitere Informationen finden Sie unter CREATE MATERIALIZED VIEW oder ALTER MATERIALIZED VIEW.

- Streaming-Erfassung und Amazon Redshift Serverless – Die für die Streaming-Erfassung von Amazon Redshift in einem bereitgestellten Cluster geltenden Einrichtungs- und Konfigurationsanweisungen gelten auch für die Streaming-Erfassung in Amazon Redshift Serverless. Es ist wichtig, Amazon Redshift Serverless mit der erforderlichen Anzahl an RPUs auszustatten, um eine Streaming-Erfassung mit automatischer Aktualisierung und andere Workloads zu unterstützen. Weitere Informationen finden Sie unter Abrechnung für Amazon Redshift Serverless.
- Amazon-Redshift-Knoten in einer anderen Availability Zone als der Amazon-MSK-Cluster – Wenn Sie die Streaming-Erfassung konfigurieren, versucht Amazon Redshift, eine Verbindung zu einem Amazon-MSK-Cluster in derselben Availability Zone herzustellen, sofern Rackinformationen für Amazon MSK aktiviert sind. Wenn sich all Ihre Knoten in anderen Availability Zones als Ihr Amazon-Redshift-Cluster befinden, können Kosten für die Datenübertragung zwischen Availability Zones anfallen. Um dies zu vermeiden, sollten Sie mindestens einen Amazon MSK-Broker-Clusterknoten in derselben AZ wie Ihr von Redshift bereitgestellter Cluster oder Ihre von Redshift bereitgestellte Arbeitsgruppe beibehalten.
- Startposition aktualisieren – Nachdem Sie eine materialisierte Ansicht erstellt haben, beginnt die erste Aktualisierung beim Wert `TRIM_HORIZON` eines Kinesis-Streams oder bei Offset 0 eines Amazon-MSK-Themas.
- Datenformate – Unterstützte Datenformate sind auf diejenigen beschränkt, die aus `VARBYTE` konvertiert werden können. Weitere Informationen finden Sie unter Typ VARBYTE und VARBYTE-Operatoren.
- Datensätze an eine Tabelle anfügen – Sie können `ALTER TABLE APPEND` ausführen, um Zeilen an eine Zieltabelle aus einer vorhandenen materialisierten Quellansicht anzufügen. Dies funktioniert nur, wenn die materialisierte Ansicht für die Streaming-Aufnahme konfiguriert ist. Weitere Informationen finden Sie unter ALTER TABLE APPEND.
- Ausführen von TRUNCATE oder DELETE – Sie können Datensätze aus einer materialisierten Ansicht entfernen, die für die Streaming-Aufnahme benutzt wird, indem Sie mehrere Methoden anwenden:

- `TRUNCATE` – Dieser Befehl löscht alle Zeilen aus einer materialisierten Ansicht, die für die Streaming-Aufnahme konfiguriert ist. Es wird kein Tabellenscan durchgeführt. Weitere Informationen finden Sie unter TRUNCATE.
- `DELETE` – Dieser Befehl löscht alle Zeilen aus einer materialisierten Ansicht, die für die Streaming-Aufnahme konfiguriert ist. Weitere Informationen finden Sie unter DELETE.

Bewährte Methoden und Empfehlungen für die Streaming-Aufnahme

Es gibt Fälle, in denen Ihnen Optionen zur Konfiguration der Streaming-Aufnahme angezeigt werden. Wir empfehlen die folgenden bewährten Methoden. Diese basieren auf unseren eigenen Tests und helfen unseren Kunden dabei, Probleme zu vermeiden, die zu Datenverlust führen.

- Extrahieren von Werten aus gestreamten Daten — Wenn Sie die Funktion JSON_EXTRACT_PATH_TEXT in Ihrer materialisierten View-Definition verwenden, um eingehendes Streaming-JSON zu vernichten, kann dies die Leistung und Latenz erheblich beeinträchtigen. Zur Erläuterung: Für jede mit JSON_EXTRACT_PATH_TEXT extrahierte Spalte wird das eingehende JSON erneut analysiert. Danach erfolgt jede Konvertierung, Filterung und Geschäftslogik von Datentypen. Das bedeutet, dass, wenn Sie beispielsweise 10 Spalten aus Ihren JSON-Daten extrahieren, jeder JSON-Datensatz zehnmal analysiert wird, was Typkonvertierungen und zusätzliche Logik beinhaltet. Dies führt zu einer höheren Aufnahmelatenz. Ein alternativer Ansatz, den wir empfehlen, besteht darin, die Funktion JSON_PARSE zu verwenden, um JSON-Datensätze in den SUPER-Datentyp von Redshift zu konvertieren. Nachdem die gestreamten Daten in der materialisierten Ansicht gelandet sind, verwenden Sie PartiQL, um einzelne Zeichenketten aus der Darstellung der JSON-Daten durch SUPER zu extrahieren. Weitere Informationen finden Sie unter Semistrukturierte Daten abfragen.

  Es ist auch wichtig zu beachten, dass JSON_EXTRACT_PATH_TEXT eine maximale Datengröße von 64 KB hat. Wenn also ein JSON-Datensatz größer als 64 KB ist, führt die Verarbeitung mit JSON_EXTRACT_PATH_TEXT zu einem Fehler.
- Zuordnen eines Amazon Kinesis Data Streams Stream- oder Amazon MSK-Themas zu einer materialisierten Amazon Redshift Redshift-Streaming-Ingestion-Ansicht — Es wird nicht empfohlen, mehrere materialisierte Streaming-Ingestion-Ansichten zu erstellen, um Daten aus einem einzelnen Stream oder Amazon MSK-Thema aufzunehmen. Amazon Kinesis Data Streams Dies liegt daran, dass jede materialisierte Ansicht einen Consumer für jeden Shard im Kinesis Data Streams Streams-Stream oder in der Partition im Kafka-Thema erstellt. Dies kann dazu führen, dass der Durchsatz des Streams oder Themas gedrosselt oder überschritten wird. Dies kann auch zu

höheren Kosten führen, da Sie dieselben Daten mehrfach aufnehmen. Wir empfehlen, dass Sie für jeden Stream oder jedes Thema eine materialisierte Streaming-Ansicht erstellen.

Wenn Ihr Anwendungsfall erfordert, dass Sie die Daten aus einem KDS-Stream oder einem MSK-Thema in mehreren materialisierten Ansichten zusammenführen, lesen Sie vorher den AWS Big Data-Blog, insbesondere Best Practices zur Implementierung von near-real-time Analysen mit Amazon Redshift Streaming Ingestion with Amazon MSK.

Verwendung der Streaming-Erfassung im Vergleich zum Bereitstellen von Daten in Amazon S3

Es gibt verschiedene Optionen für das Streamen von Daten zu Amazon Redshift oder zu Amazon Redshift Serverless. Zwei bekannte Optionen sind Streaming-Ingestion, wie in diesem Thema beschrieben, oder die Einrichtung eines Lieferstreams für Amazon S3 mit Firehose. In der folgenden Liste werden die einzelnen Methoden beschrieben:

1. Die Streaming-Erfassung von Kinesis Data Streams oder Amazon Managed Streaming für Apache Kafka zu Amazon Redshift oder Amazon Redshift Serverless umfasst das Konfigurieren einer materialisierten Ansicht für den Empfang der Daten.
2. Die Bereitstellung von Daten in Amazon Redshift mithilfe von Kinesis Data Streams und Streaming über Firehose beinhaltet die Verbindung des Quell-Streams mit Amazon Data Firehose und das Warten darauf, dass Firehose die Daten in Amazon S3 bereitstellt. Bei diesem Verfahren werden Batches verschiedener Größe in Pufferintervallen unterschiedlicher Länge verwendet. Nach dem Streaming zu Amazon S3 initiiert Firehose einen COPY-Befehl, um die Daten zu laden.

Bei der Streaming-Erfassung umgehen Sie mehrere Schritte, die für den zweiten Vorgang erforderlich sind:

- Sie müssen keine Daten an einen Amazon Data Firehose-Lieferstream senden, da mit Streaming-Ingestion Daten direkt von Kinesis Data Streams an eine materialisierte Ansicht in einer Redshift-Datenbank gesendet werden können.
- Sie müssen gestreamte Daten nicht in Amazon S3 speichern, da Streaming-Erfassungsdaten direkt in die materialisierte Ansicht von Redshift übertragen werden.
- Sie müssen keine COPY-Befehle schreiben und ausführen, da die Daten in der materialisierten Ansicht direkt aus dem Stream aktualisiert werden. Das Laden von Daten von Amazon S3 in Redshift ist nicht Teil des Prozesses.

Beachten Sie, dass die Streaming-Erfassung auf Streams von Amazon Kinesis Data Streams und Themen von Amazon MSK beschränkt ist. Für das Streaming von Kinesis Data Streams zu anderen Zielen als Amazon Redshift benötigen Sie wahrscheinlich einen Firehose-Lieferstream. Weitere Informationen finden Sie unter Daten an einen Amazon Data Firehose Delivery Stream senden.

Überlegungen

Im Folgenden finden Sie Überlegungen zur Streaming-Aufnahme in Amazon Redshift.

Funktion oder Verhalten	Beschreibung
Längenbegrenzung für Kafka-Themen	Die Verwendung von Kafka-Themen, deren Name mehr als 128 Zeichen umfasst (ohne Anführungszeichen) ist nicht möglich. Weitere Informationen finden Sie unter Namen und Kennungen.
Inkrementelle Aktualisierungen und JOINS in einer materialisierten Ansicht	Die materialisierte Ansicht muss inkrementell wartbar sein. Eine vollständige Neuberechnung ist für Kinesis oder Amazon MSK nicht möglich, da der Stream- oder Themenverlauf standardmäßig nicht über 24 Stunden bzw. 7 Tage hinaus beibehalten wird. Sie können längere Datenaufbewahrungsfristen in Kinesis oder Amazon MSK festlegen. Dies kann jedoch zu einem größeren Verwaltungsaufwand und höheren Kosten führen. Darüber hinaus werden JOINs derzeit in materialisierten Ansichten, die in einem Kinesis-Stream oder einem Amazon-MSK-Thema erstellt wurden, nicht unterstützt. Nachdem Sie eine materialisierte Ansicht in Ihrem Stream oder Thema erstellt haben, können Sie eine andere materialisierte Ansicht erstellen, um Ihre materialisierte Streaming-Ansicht mit anderen materialisierten Ansichten, Tabellen oder Ansichten zu verbinden. Weitere Informationen finden Sie unter REFRESH MATERIALIZED VIEW.
Analyse von Datensätzen	Die Streaming-Erfassung von Amazon Redshift unterstützt die Analyse von Datensätzen, die von der Kinesis Producer Library (KPL-Schlüsselkonzepte – Aggregation) aggregiert wurden. Die aggregierten Datensätze werden erfasst, werden jedoch

Funktion oder Verhalten	Beschreibung
	als Pufferdaten für das Binärprotokoll verwendet. (Weitere Informationen finden Sie unter Protokollpuffer.) Je nachdem, wie Sie Daten an Kinesis übertragen, müssen Sie diese Funktion möglicherweise. deaktivieren.
Dekomprimierung	`VARBYTE` unterstützt zurzeit keine Dekomprimierungsmethoden. Aus diesem Grund können Datensätze, die komprimierte Daten enthalten, nicht in Redshift abgefragt werden. Dekomprimieren Sie Ihre Daten, bevor Sie sie an den Kinesis-Stream oder das Amazon-MSK-Thema übertragen.

Funktion oder Verhalten	Beschreibung
Maximale Datensatzgröße	Die maximale Größe jedes Datensatzfeldes, das Amazon Redshift von Kinesis oder Amazon MSK erfassen kann, beträgt etwas weniger als 1 MB. Einige Erläuterungen zum Verhalten: • Maximale VARBYTE-Länge — Für die Streaming-Aufnahme unterstützt der VARBYTE Typ Daten bis zu einer maximalen Länge von 1.024.000 Byte. Kinesis begrenzt die Nutzlasten auf 1 MB. • Nachrichtenlimits — Die Standardkonfiguration von Amazon MSK begrenzt Nachrichten auf 1 MB. Wenn eine Nachricht Header enthält, ist die Datenmenge außerdem auf 1 048 470 Byte begrenzt. Bei Verwendung der Standarde instellungen gibt es keine Probleme bei der Erfassung. Sie können jedoch die maximale Nachrichtengröße für Kafka und damit auch für Amazon MSK ändern und einen größeren Wert festlegen. In diesem Fall kann es möglich sein, dass das Schlüssel/Wert-Feld eines Kafka-Datensatzes oder der Header die Größenbeschränkung überschreitet. Solche Datensätze können zu einem Fehler führen und werden nicht erfasst. **Note** Amazon Redshift unterstützt eine maximale Größe von 1.024.000 Byte für die Streaming-Aufnahme von Kinesis oder Amazon MSK, obwohl Amazon Redshift eine maximale Größe von 16 MB für den Datentyp unterstützt. VARBYTE

Funktion oder Verhalten	Beschreibung
Fehlerdatensätze	Immer wenn ein Datensatz nicht in Redshift erfasst werden kann, da die Daten die maximale Größe überschreiten, wird der betreffende Datensatz übersprungen. Die materialisierte Ansicht kann auch in diesem Fall erfolgreich aktualisiert werden und ein Segment jedes Fehlerdatensatzes wird in die Systemtab elle SYS_STREAM_SCAN_ERRORS geschrieben. Fehler, die sich aus der Geschäftslogik ergeben, wie beispielsweise Fehler in einer Berechnung oder Fehler infolge einer Typkonver tierung, werden nicht übersprungen. Bevor Sie der Definitio n Ihrer materialisierten Ansicht Logik hinzufügen, sollten Sie diese sorgfältig testen, damit es nicht zu einer solchen Situation kommt.
Private Konnektivität mit mehreren VPC von Amazon MSK	Private Amazon MSK Multi-VPC-Konnektivität wird derzeit nicht für die Redshift-Streaming-Aufnahme unterstützt. Alternativ können Sie VPC-Peering verwenden, um VPCs zu verbinden oder AWS Transit Gatewayum VPCs und lokale Netzwerke über einen zentralen Hub zu verbinden. Mit beiden kann Redshift mit einem Amazon MSK-Cluster oder mit Amazon MSK Serverless in einer anderen VPC kommunizieren.

Erste Schritte mit der Streaming-Erfassung aus Amazon Kinesis Data Streams

Das Einrichten der Streaming-Erfassung von Amazon Redshift beinhaltet das Erstellen eines externen Schemas, das der Streaming-Datenquelle zugeordnet ist, und das Erstellen einer materialisierten Ansicht, die auf das externe Schema verweist. Die Streaming-Erfassung von Amazon Redshift unterstützt Kinesis Data Streams als Quelle. Daher benötigen Sie eine Kinesis-Data-Streams-Quelle, bevor Sie die Streaming-Erfassung konfigurieren. Wenn Sie keine Quelle haben, folgen Sie den Anweisungen in der Kinesis-Dokumentation unter Erste Schritte mit Amazon Kinesis Data Streams oder erstellen Sie eine auf der Konsole, indem Sie die Anweisungen unter Stream über die AWS Management Console erstellen.

Die Streaming-Erfassung von Amazon Redshift verwendet eine materialisierte Ansicht, die direkt aus dem Stream aktualisiert wird, wenn `REFRESH` ausgeführt wird. Die materialisierte Ansicht wird der Stream-Datenquelle zugeordnet. Sie können für die Stream-Daten als Teil der materialisierten View-Definition Filter anwenden und Aggregationen durchführen. Ihre materialisierte Ansicht der Streaming-Erfassung (die materialisierte Basisansicht) kann nur auf einen Stream verweisen, aber Sie können zusätzliche materialisierte Ansichten erstellen, die mit der materialisierten Basisansicht und anderen materialisierten Ansichten oder Tabellen verknüpft sind.

> **Note**
>
> Streaming-Erfassung und Amazon Redshift Serverless – Die Konfigurationsschritte in diesem Thema gelten für bereitgestellte Amazon-Redshift-Cluster und Amazon Redshift Serverless. Weitere Informationen finden Sie unter Überlegungen zur Streaming-Erfassung.

Angenommen, Sie haben einen Kinesis Data Streams-Stream. Dann besteht der erste Schritt darin, ein Schema in Amazon Redshift mit `CREATE EXTERNAL SCHEMA` zu definieren und auf eine Kinesis Data Streams-Ressource zu verweisen. Um auf Daten im Stream zuzugreifen, definieren Sie anschließend den `STREAM` in einer materialisierten Ansicht. Sie können Stream-Datensätze im halbstrukturierten `SUPER`-Format speichern oder ein Schema definieren, das dazu führt, dass Daten in Amazon Redshift-Datentypen konvertiert werden. Wenn Sie die materialisierte Ansicht abfragen, sind die zurückgegebenen Datensätze eine point-in-time Ansicht des Streams.

1. Erstellen Sie eine IAM-Rolle mit einer Vertrauensrichtlinie, die es Ihrem Amazon Redshift-Cluster oder Ihrer Amazon-Redshift-Serverless-Arbeitsgruppe ermöglicht, die Rolle anzunehmen. Informationen zur Konfiguration der Vertrauensrichtlinie für die IAM-Rolle finden Sie unter Autorisieren von Amazon Redshift für den Zugriff auf andere AWS Dienste in Ihrem Namen. Nach der Erstellung sollte die Rolle über die folgende IAM-Richtlinie verfügen, die sie zur Kommunikation mit dem Amazon Kinesis-Datenstrom berechtigt.

   IAM-Richtlinie für einen nicht verschlüsselten Stream von Kinesis-Daten-Streams

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "ReadStream",
               "Effect": "Allow",
               "Action": [
   ```

```
                "kinesis:DescribeStreamSummary",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:DescribeStream"
            ],
            "Resource": "arn:aws:kinesis:*:0123456789:stream/*"
        },
        {
            "Sid": "ListStream",
            "Effect": "Allow",
            "Action": [
                "kinesis:ListStreams",
                "kinesis:ListShards"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM-Richtlinie für einen verschlüsselten Stream von Kinesis-Daten-Streams

```
{
 "Version": "2012-10-17",
 "Statement": [{
   "Sid": "ReadStream",
   "Effect": "Allow",
   "Action": [
    "kinesis:DescribeStreamSummary",
    "kinesis:GetShardIterator",
    "kinesis:GetRecords",
    "kinesis:DescribeStream"
   ],
   "Resource": "arn:aws:kinesis:*:0123456789:stream/*"
  },
  {
   "Sid": "DecryptStream",
   "Effect": "Allow",
   "Action": [
    "kms:Decrypt"
   ],
   "Resource": "arn:aws:kms:us-
east-1:0123456789:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
```

```
        {
   "Sid": "ListStream",
   "Effect": "Allow",
   "Action": [
    "kinesis:ListStreams",
    "kinesis:ListShards"
   ],
   "Resource": "*"
  }
 ]
}
```

2. Überprüfen Sie Ihre VPC und stellen Sie sicher, dass Ihr Amazon-Redshift-Cluster oder Amazon Redshift Serverless über eine Route verfügt, um über das Internet unter Verwendung eines NAT-Gateways oder Internet-Gateways zu den Kinesis-Data-Streams-Endpunkten zu gelangen. Wenn Sie möchten, dass der Datenverkehr zwischen Redshift und Kinesis Data Streams im AWS Netzwerk verbleibt, sollten Sie die Verwendung eines Kinesis Interface VPC-Endpoints in Betracht ziehen. Weitere Informationen finden Sie unter Verwenden von Amazon Kinesis Data Streams mit Schnittstellen-VPC-Endpunkten.
3. Erstellen Sie in Amazon Redshift ein externes Schema, um die Daten aus Kinesis einem Schema zuzuordnen.

```
CREATE EXTERNAL SCHEMA kds
FROM KINESIS
IAM_ROLE { default | 'iam-role-arn' };
```

Die Streaming-Erfassung für Kinesis Data Streams erfordert keinen Authentifizierungstyp. Für Kinesis-Data-Streams-Anforderungen wird die in der Anweisung `CREATE EXTERNAL SCHEMA` definierte IAM-Rolle verwendet.

Optional: Verwenden Sie das Schlüsselwort REGION, um die Region anzugeben, in der sich der Amazon Kinesis Data Streams oder der Amazon MSK-Stream befindet.

```
CREATE EXTERNAL SCHEMA kds
FROM KINESIS
REGION 'us-west-2'
IAM_ROLE { default | 'iam-role-arn' };
```

In diesem Beispiel gibt die Region den Speicherort des Quellstreams an. IAM_ROLE ist ein Beispiel.

4. Erstellen Sie eine materialisierte Ansicht, um die Stream-Daten zu konsumieren. Wenn bei einer Anweisung wie der folgenden ein Datensatz nicht analysiert werden kann, führt dies zu einem Fehler. Verwenden Sie einen Befehl wie diesen, wenn Sie nicht möchten, dass Fehlerdatensätze übersprungen werden.

```
CREATE MATERIALIZED VIEW my_view AUTO REFRESH YES AS
SELECT *
FROM kds.my_stream_name;
```

Das folgende Beispiel definiert eine materialisierte Ansicht für Quelldaten im JSON-Format. Die Ansicht überprüft, ob eingehende Daten ordnungsgemäß im JSON-Format formatiert sind. Bei den Namen von Kinesis-Streams wird die Groß- und Kleinschreibung beachtet. Die Namen können sowohl Groß- als auch Kleinbuchstaben enthalten. Um Daten aus Streams mit Namen in Großbuchstaben aufzunehmen, können Sie die Konfiguration `enable_case_sensitive_identifier` `true` auf Datenbankebene so einstellen. Weitere Informationen finden Sie unter Namen und Kennungen und enable_case_sensitive_identifier.

```
CREATE MATERIALIZED VIEW my_view AUTO REFRESH YES AS
SELECT approximate_arrival_timestamp,
partition_key,
shard_id,
sequence_number,
refresh_time,
JSON_PARSE(kinesis_data) as kinesis_data
FROM kds.my_stream_name
WHERE CAN_JSON_PARSE(kinesis_data);
```

Um die automatische Aktualisierung zu aktivieren, verwenden Sie `AUTO REFRESH YES`. Standardmäßig wird die manuelle Aktualisierung verwendet. Beachten Sie, dass bei der Verwendung von CAN_JSON_PARSE Datensätze, die nicht analysiert werden können, möglicherweise übersprungen werden.

Die Metadatenspalten umfassen folgende Spalten:

Metadatenspalte	Datentyp	Beschreibung
approximate_arrival_timestamp	Timestamp ohne Zeitzone	Die ungefähre Zeit, zu der der Datensatz in den Kinesis-Stream eingefügt wurde
partition_key	varchar(256)	Der von Kinesis verwendete Schlüssel, um den Datensatz einem Shard zuzuweisen
shard_id	char(20)	Die eindeutige Kennung des Shards innerhalb des Streams, aus dem der Datensatz abgerufen wurde
sequence_number	varchar(128)	Die eindeutige Kennung des Datensatzes aus dem Kinesis-Shard
refresh_time	Timestamp ohne Zeitzone	Der Zeitpunkt, zu dem die Aktualisierung gestartet wurde
kinesis_data	varbyte	Der Datensatz aus dem Kinesis-Stream

Wenn Ihre Materialized View Definition Geschäftslogik enthält, sollten Sie unbedingt beachten, dass Fehler in der Geschäftslogik in einigen Fällen dazu führen können, dass die Streaming-Aufnahme blockiert wird. Dies kann dazu führen, dass Sie die materialisierte Ansicht löschen und neu erstellen müssen. Um dies zu vermeiden, empfehlen wir Ihnen, Ihre Logik so einfach wie möglich zu halten und die meisten Ihrer Geschäftslogikprüfungen an den Daten durchzuführen, nachdem sie aufgenommen wurden.

5. Aktualisieren Sie die Ansicht. Dadurch wird Redshift aufgerufen, um Daten aus dem Stream zu lesen und Daten in die materialisierte Ansicht zu laden.

```
REFRESH MATERIALIZED VIEW my_view;
```

6. Fragen Sie Daten in der materialisierten Ansicht ab.

```
select * from my_view;
```

Erste Schritte mit der Streaming-Erfassung aus Amazon Managed Streaming for Apache Kafka

Die Streaming-Erfassung von Amazon Redshift soll den Prozess der direkten Erfassung von Stream-Daten von einem Streaming-Dienst in Amazon Redshift oder Amazon Redshift Serverless vereinfachen. Dies funktioniert mit Amazon MSK und Amazon MSK Serverless sowie mit Kinesis. Dank der Streaming-Erfassung von Amazon Redshift ist es nicht mehr notwendig, einen Kinesis-Data-Streams-Stream oder ein Amazon-MSK-Thema in Amazon S3 bereitzustellen, bevor die Stream-Daten in Redshift erfasst werden.

Technisch gesehen bietet die Streaming-Erfassung von Amazon Kinesis Data Streams und Amazon Managed Streaming for Apache Kafka eine Hochgeschwindigkeits-Erfassung mit geringer Latenz von Stream- oder Themendaten in einer materialisierten Ansicht von Amazon Redshift. Nach der Einrichtung können Sie mithilfe einer Aktualisierung der materialisierten Ansicht große Datenmengen erfassen.

Führen Sie die folgenden Schritte aus, um die Streaming-Erfassung von Amazon Redshift für Amazon MSK einzurichten:

1. Erstellen Sie ein externes Schema, das der Streaming-Datenquelle zugeordnet ist.
2. Erstellen Sie eine materialisierte Ansicht, die auf das externe Schema verweist.

Sie müssen über eine Amazon-MSK-Quelle verfügen, bevor Sie die Streaming-Erfassung von Amazon Redshift konfigurieren können. Wenn Sie keine Quelle haben, folgen Sie den Anweisungen unter Erste Schritte mit Amazon MSK.

> **Note**
>
> Streaming-Erfassung und Amazon Redshift Serverless – Die Konfigurationsschritte in diesem Thema gelten für bereitgestellte Amazon-Redshift-Cluster und Amazon Redshift Serverless. Weitere Informationen finden Sie unter Überlegungen zur Streaming-Erfassung.

Einrichten von IAM und Durchführen der Streaming-Erfassung von Kafka aus

Wenn Sie einen Amazon-MSK-Cluster zur Verfügung haben, besteht der erste Schritt darin, mit `CREATE EXTERNAL SCHEMA` ein Schema in Redshift zu definieren und auf das Kafka-Thema als Datenquelle zu verweisen. Um auf Daten in dem Thema zuzugreifen, definieren Sie anschließend den `STREAM` in einer materialisierten Ansicht. Sie können Datensätze aus Ihrem Thema im halbstrukturierten `SUPER`-Format speichern oder ein Schema definieren, das dazu führt, dass Daten in Amazon-Redshift-Datentypen konvertiert werden. Wenn Sie die materialisierte Ansicht abfragen, stellen die zurückgegebenen Datensätze eine point-in-time Ansicht des Themas dar.

1. Erstellen Sie eine IAM-Rolle mit einer Vertrauensrichtlinie, die es Ihrem Amazon Redshift-Cluster oder Amazon Redshift Serverless ermöglicht, die Rolle anzunehmen. Informationen zur Konfiguration der Vertrauensrichtlinie für die IAM-Rolle finden Sie unter Autorisieren von Amazon Redshift für den Zugriff auf andere AWS Dienste in Ihrem Namen. Nach der Erstellung sollte die Rolle über die folgende IAM-Richtlinie verfügen, die sie zur Kommunikation mit dem Amazon-MSK-Cluster berechtigt. Welche Richtlinie Sie benötigen, hängt von der Authentifizierungsmethode ab, die in Ihrem Cluster verwendet wird, wenn Sie Amazon MSK verwenden. Informationen zu den in Amazon MSK verfügbaren Authentifizierungsmethoden finden Sie unter Authentifizierung und Autorisierung für Apache Kafka-APIs.

   Eine IAM-Richtlinie für Amazon MSK mit nicht authentifiziertem Zugriff:

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "VisualEditor0",
               "Effect": "Allow",
               "Action": [
                   "kafka:GetBootstrapBrokers"
               ],
               "Resource": "*"
   ```

```
        }
    ]
}
```

Eine IAM-Richtlinie für Amazon MSK bei Verwendung der IAM-Authentifizierung:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MSKIAMpolicy",
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeTopic",
                "kafka-cluster:Connect"
            ],
            "Resource": [
                "arn:aws:kafka:*:0123456789:cluster/MyTestCluster/*",
                "arn:aws:kafka:*:0123456789:topic/MyTestCluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:*:0123456789:group/MyTestCluster/*"
            ]
        },
        {
            "Sid": "MSKPolicy",
            "Effect": "Allow",
            "Action": [
                "kafka:GetBootstrapBrokers"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Überprüfen Sie Ihre VPC und stellen Sie sicher, dass Ihr Amazon-Redshift-Cluster oder Amazon Redshift Serverless über eine Route zu Ihrem Amazon-MSK-Cluster verfügt. Die eingehenden Sicherheitsgruppenregeln für Ihren Amazon-MSK-Cluster sollten die Sicherheitsgruppe Ihres Amazon-Redshift-Clusters oder Ihrer Amazon-Redshift-Serverless-Arbeitsgruppe zulassen. Welche Ports Sie angeben, hängt von der Authentifizierungsmethode ab, die in Ihrem Cluster verwendet wird, wenn Sie Amazon MSK verwenden. Weitere Informationen finden Sie unter Portinformationen und Zugriff von innerhalb und AWS außerhalb der VPC.

   Beachten Sie, dass die Client-Authentifizierung mit mTLS für die Streaming-Erfassung nicht unterstützt wird. Weitere Informationen finden Sie unter Limitations.

   Die folgende Tabelle enthält kostenlose Konfigurationsoptionen, die für die Streaming-Erfassung von Amazon MSK festgelegt werden können:

Amazon-Redshift-Konfiguration	Amazon-MSK-Konfiguration	Port zum Öffnen zwischen Redshift und Amazon MSK
AUTHENTICATION NONE	TLS-Transport deaktiviert	9092
AUTHENTICATION NONE	TLS-Transport aktiviert	9094
AUTHENTICATION IAM	IAM	9098/9198

   Die Amazon-Redshift-Authentifizierung wird in der Anweisung CREATE EXTERNAL SCHEMA festgelegt.

   Im Fall, dass der Amazon-MSK-Cluster über aktivierte Mutual Transport Layer Security (mTLS)-Authentifizierung verfügt, wird Amazon Redshift durch die Konfiguration von AUTHENTICATION NONE angewiesen, Port 9094 für nicht authentifizierten Zugriff zu verwenden. Da der Port jedoch für die mTLS-Authentifizierung verwendet wird, schlägt dies fehl. Aus diesem Grund empfehlen wir, bei Verwendung von mTLS zu AUTHENTICATION IAM zu wechseln.

3. Aktivieren Sie Enhanced VPC Routing in Ihrem Amazon-Redshift-Cluster oder Ihrer Amazon-Redshift-Serverless-Arbeitsgruppe. Weitere Informationen finden Sie unter Aktivieren von Enhanced VPC Routing.

> **Note**
>
> Um die Amazon MSK-Bootstrap-Broker-URL abzurufen, führt Amazon Redshift einen GetBootstrapBroker-API-Aufruf durch und verwendet dabei die von der angehängten IAM-Rolle bereitgestellten Berechtigungen. Beachten Sie, dass das Subnetz für Ihren von Amazon Redshift bereitgestellten Cluster oder Ihre Amazon Redshift Serverless-Arbeitsgruppe über ein NAT-Gateway oder ein Internet-Gateway verfügen muss, damit diese Anfrage erfolgreich ist, wenn erweitertes VPC-Routing aktiviert ist. Ihre Netzwerk-ACLs und ausgehenden Sicherheitsgruppenregeln für das oben genannte Subnetz müssen auch den Zugriff auf die Amazon MSK API-Serviceendpunkte ermöglichen. Weitere Informationen finden Sie unter Amazon Managed Streaming for Apache Kafka Kafka-Endpunkte und Kontingente.

4. Erstellen Sie ein externes Schema in Amazon Redshift, das dem Amazon-MSK-Cluster zugeordnet werden soll.

```
CREATE EXTERNAL SCHEMA MySchema
FROM MSK
IAM_ROLE { default | 'iam-role-arn' }
AUTHENTICATION { none | iam }
CLUSTER_ARN 'msk-cluster-arn';
```

In der FROM-Klausel gibt Amazon MSK an, dass das Schema Daten von Managed Kafka Services zuordnet.

Die Streaming-Erfassung für Amazon MSK bietet die folgenden Authentifizierungstypen, wenn Sie das externe Schema erstellen:

- none: Gibt an, dass es keinen Authentifizierungsschritt gibt.
- iam: Gibt an, dass eine IAM-Authentifizierung erfolgt. Stellen Sie bei Auswahl dieser Option sicher, dass die IAM-Rolle über Berechtigungen für die IAM-Authentifizierung verfügt.

Weitere Amazon-MSK-Authentifizierungsmethoden, wie die TLS-Authentifizierung oder die Authentifizierung mit Benutzername und Passwort, werden für die Streaming-Erfassung nicht unterstützt.

CLUSTER_ARN gibt den Amazon-MSK-Cluster an, von dem Sie streamen.

5. Erstellen Sie eine materialisierte Ansicht, um die Daten aus dem Thema zu konsumieren. Verwenden Sie einen SQL-Befehl wie dieses Beispiel, wenn Sie nicht möchten, dass Fehlerdatensätze übersprungen werden.

```
CREATE MATERIALIZED VIEW MyView AUTO REFRESH YES AS
SELECT *
FROM MySchema."mytopic";
```

Im folgenden Beispiel wird eine materialisierte Ansicht mit JSON-Quelldaten definiert. Beachten Sie, dass mit der folgenden Ansicht validiert wird, ob die Daten ein gültiges JSON-Format aufweisen und UTF8-kodiert sind. Bei den Namen von Kafka-Themen wird die Groß- und Kleinschreibung beachtet. Die Namen können sowohl Groß- als auch Kleinbuchstaben enthalten. Für die Aufnahme von Themen mit Großbuchstaben können Sie die Konfiguration `enable_case_sensitive_identifier true` auf Datenbankebene festlegen. Weitere Informationen finden Sie unter Namen und Kennungen und enable_case_sensitive_identifier.

```
CREATE MATERIALIZED VIEW MyView AUTO REFRESH YES AS
SELECT kafka_partition,
 kafka_offset,
 kafka_timestamp_type,
 kafka_timestamp,
 kafka_key,
 JSON_PARSE(kafka_value) as kafka_data,
 kafka_headers,
 refresh_time
FROM MySchema."mytopic"
WHERE CAN_JSON_PARSE(kafka_value);
```

Um die automatische Aktualisierung zu aktivieren, verwenden Sie `AUTO REFRESH YES`. Standardmäßig wird die manuelle Aktualisierung verwendet.

Die Metadatenspalten umfassen folgende Spalten:

Metadatenspalte	Datentyp	Beschreibung
kafka_partition	bigint	Partitions-ID des Datensatzes aus dem Kafka-Thema

Metadatenspalte	Datentyp	Beschreibung
kafka_offset	bigint	Offset des Datensatzes im Kafka-Thema für eine bestimmte Partition
kafka_timestamp_type	char(1)	Art des im Kafka-Datensatz verwendeten Zeitstempels: • C – Erstellungszeit des Datensatzes (CREATE_TIME) auf der Client-Seite • L – Anfügezeit des Datensatzes (LOG_APPEND_TIME) auf der Kafka-Serverseite • U – Die Erstellungszeit des Datensatzes ist nicht verfügbar (NO_TIMESTAMP_TYPE)
kafka_timestamp	Timestamp ohne Zeitzone	Der timestamp-Wert für den Datensatz
kafka_key	varbyte	Der Schlüssel des Kafka-Datensatzes
kafka_value	varbyte	Der von Kafka erhaltene Datensatz
kafka_headers	super	Der Header des von Kafka erhaltenen Datensatzes
refresh_time	Timestamp ohne Zeitzone	Der Zeitpunkt, zu dem die Aktualisierung gestartet wurde

Wenn Ihre Materialized View-Definition Geschäftslogik enthält, sollten Sie unbedingt beachten, dass Fehler in der Geschäftslogik in einigen Fällen dazu führen können, dass die Streaming-Aufnahme blockiert wird. Dies kann dazu führen, dass Sie die materialisierte Ansicht löschen und neu erstellen müssen. Um dies zu vermeiden, empfehlen wir Ihnen, Ihre Geschäftslogik einfach zu halten und zusätzliche Logik auf die Daten anzuwenden, nachdem Sie sie aufgenommen haben.

6. Aktualisieren Sie die Ansicht. Dadurch wird Amazon Redshift aufgerufen, um Daten aus dem Thema zu lesen und Daten in die materialisierte Ansicht zu laden.

```
REFRESH MATERIALIZED VIEW MyView;
```

7. Fragen Sie Daten in der materialisierten Ansicht ab.

```
select * from MyView;
```

Die materialisierte Ansicht wird direkt vom Thema aus aktualisiert, wenn `REFRESH` ausgeführt wird. Sie erstellen eine materialisierte Ansicht, die der Kafka-Themendatenquelle zugeordnet ist. Sie können für die Daten als Teil der Definition der materialisierten Ansicht Filter anwenden und Aggregationen durchführen. Ihre materialisierte Ansicht der Streaming-Erfassung (die materialisierte Basisansicht) kann nur auf ein Kafka-Thema verweisen, Sie können jedoch zusätzliche materialisierte Ansichten erstellen, die mit der materialisierten Basisansicht und anderen materialisierten Ansichten oder Tabellen verknüpft werden.

Weitere Informationen zu Einschränkungen bei der Streaming-Erfassung finden Sie unter Überlegungen.

Tutorial zur Streaming-Erfassung von Ladestationsdaten für Elektrofahrzeuge mit Kinesis

Dieses Verfahren zeigt, wie Daten aus einem Kinesis-Stream mit dem Namen ev_station_data erfasst werden, der Verbrauchsdaten von verschiedenen Ladestationen für Elektrofahrzeuge im JSON-Format enthält. Das Schema ist gut definiert. Das Beispiel zeigt, wie die Daten als unformatierte JSON-Daten gespeichert werden und wie die JSON-Daten bei der Erfassung in Amazon Redshift-Datentypen konvertiert werden.

Einrichten von Produzenten

1. Führen Sie mithilfe von Amazon Kinesis Data Streams die Schritte zum Erstellen eines Streams mit dem Namen `ev_station_data` aus. Wählen Sie On-Demand für Kapazitätsmodus aus. Weitere Informationen finden Sie unter Einen Stream über die AWS Management Console erstellen.

2. Der Amazon Kinesis Data Generator kann Ihnen dabei helfen, Testdaten zur Verwendung mit Ihrem Stream zu generieren. Befolgen Sie die im Tool beschriebenen Schritte, um loszulegen, und verwenden Sie die folgende Datenvorlage zum Generieren Ihrer Daten:

```
{

   "_id" : "{{random.uuid}}",
   "clusterID": "{{random.number(
        {   "min":1,
            "max":50
        }
    )}}",
    "connectionTime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "kWhDelivered": "{{commerce.price}}",
    "stationID": "{{random.number(
        {   "min":1,
            "max":467
        }
    )}}",
      "spaceID": "{{random.word}}-{{random.number(
        {   "min":1,
            "max":20
        }
    )}}",

   "timezone": "America/Los_Angeles",
   "userID": "{{random.number(
        {   "min":1000,
            "max":500000
        }
    )}}"
}
```

Jedes JSON-Objekt in den Stream-Daten hat die folgenden Eigenschaften:

```
{
    "_id": "12084f2f-fc41-41fb-a218-8cc1ac6146eb",
```

```
    "clusterID": "49",
    "connectionTime": "2022-01-31 13:17:15",
    "kWhDelivered": "74.00",
    "stationID": "421",
    "spaceID": "technologies-2",
    "timezone": "America/Los_Angeles",
    "userID": "482329"
}
```

Amazon Redshift-Einrichtung

Diese Schritte zeigen Ihnen, wie Sie die materialisierte Ansicht so konfigurieren, dass Daten erfasst werden.

1. Erstellen Sie ein externes Schema, um die Daten aus Kinesis einem Amazon Redshift-Objekt zuzuordnen.

   ```
   CREATE EXTERNAL SCHEMA evdata FROM KINESIS
   IAM_ROLE 'arn:aws:iam::0123456789:role/redshift-streaming-role';
   ```

   Weitere Informationen zum Konfigurieren der IAM-Rolle finden Sie unter Erste Schritte mit der Streaming-Erfassung aus Amazon Kinesis Data Streams.

2. Erstellen Sie eine materialisierte Ansicht, um die Stream-Daten zu konsumieren. Die folgenden Beispiele zeigen beide Methoden zum Definieren materialisierter Ansichten für die Erfassung der JSON-Quelldaten.

   Speichern Sie zunächst Stream-Datensätze im halbstrukturierten SUPER-Format. In diesem Beispiel wird die JSON-Quelle in Amazon Redshift gespeichert, ohne eine Konvertierung in Amazon Redshift-Typen vorzunehmen.

   ```
   CREATE MATERIALIZED VIEW ev_station_data AS
       SELECT approximate_arrival_timestamp,
       partition_key,
       shard_id,
       sequence_number,
       json_parse(kinesis_data) as payload
       FROM evdata."ev_station_data" WHERE can_json_parse(kinesis_data);
   ```

Im Gegensatz dazu verfügt die materialisierte Ansicht in der folgenden Definition der materialisierten Ansicht über ein definiertes Schema in Amazon Redshift. Die materialisierte Ansicht wird basierend auf dem UUID-Wert aus dem Stream verteilt und nach dem Wert `approximatearrivaltimestamp` sortiert.

```
CREATE MATERIALIZED VIEW ev_station_data_extract DISTKEY(6) sortkey(1) AUTO REFRESH
 YES AS
    SELECT refresh_time,
    approximate_arrival_timestamp,
    partition_key,
    shard_id,
    sequence_number,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'_id',true)::character(36)
 as ID,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'clusterID',true)::varchar(30)
 as clusterID,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'connectionTime',true)::varchar(
 as connectionTime,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'kWhDelivered',true)::DECIMAL(10
 as kWhDelivered,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'stationID',true)::DECIMAL(10,2)
 as stationID,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'spaceID',true)::varchar(100)
 as spaceID,
    json_extract_path_text(from_varbyte(kinesis_data,
'utf-8'),'timezone',true)::varchar(30)as timezone,

 json_extract_path_text(from_varbyte(kinesis_data,'utf-8'),'userID',true)::varchar(30)
 as userID
    FROM evdata."ev_station_data"
    WHERE LENGTH(kinesis_data) < 65355;
```

Stream-Abfrage

1. Fragen Sie die aktualisierte materialisierte Ansicht ab, um Nutzungsstatistiken abzurufen.

```
SELECT to_timestamp(connectionTime, 'YYYY-MM-DD HH24:MI:SS') as connectiontime
,SUM(kWhDelivered) AS Energy_Consumed
,count(distinct userID) AS #Users
from ev_station_data_extract
group by  to_timestamp(connectionTime, 'YYYY-MM-DD HH24:MI:SS')
order by 1 desc;
```

2. Zeigen Sie die Ergebnisse an.

```
connectiontime         energy_consumed #users
2022-02-08 16:07:21+00   4139         10
2022-02-08 16:07:20+00   5571         10
2022-02-08 16:07:19+00   8697         20
2022-02-08 16:07:18+00   4408         10
2022-02-08 16:07:17+00   4257         10
2022-02-08 16:07:16+00   6861         10
2022-02-08 16:07:15+00   5643         10
2022-02-08 16:07:14+00   3677         10
2022-02-08 16:07:13+00   4673         10
2022-02-08 16:07:11+00   9689         20
```

Erstellen von Ansichten im AWS Glue Data Catalog (Vorschau)

Hierbei handelt es sich um die vorab veröffentlichte Dokumentation im Datenkatalog für Amazon Redshift, derzeit in der Vorschauversion. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter Beta- und Vorschauversionen in den AWS -Servicebedingungen.

Sie können einen Amazon-Redshift-Cluster in der Vorschau erstellen, um neue Funktionen von Amazon Redshift zu testen. Sie haben nicht die Möglichkeit, diese Funktionen in der Produktion zu verwenden oder Ihren Vorschau-Cluster in einen Produktionscluster oder einen Cluster auf einem anderen Pfad zu verschieben. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Erstellen eines Clusters in der Vorschau

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Provisioned clusters dashboard (Dashboard für bereitgestellte Cluster) und dann Clusters (Cluster) aus. Die aktuellen Cluster für Ihr Konto AWS-Region sind aufgeführt. Eine Teilmenge der Eigenschaften jedes Clusters wird in den Spalten der Liste angezeigt.
3. Auf der Seite mit der Clusterliste (Clusters) wird ein Banner angezeigt, das die Vorschau vorstellt. Wählen Sie die Schaltfläche Create preview cluster (Vorschau-Cluster erstellen) aus, um die Seite zum Erstellen von Clustern zu öffnen.
4. Geben Sie Eigenschaften für Ihren Cluster ein. Wählen Sie den Vorschaupfad (Preview track) aus, der die zu testenden Funktionen enthält. Wir empfehlen, einen Namen für den Cluster zu verwenden, der darauf hinweist, dass sich dieser auf einem Vorschaupfad befindet. Wählen Sie Optionen für Ihren Cluster, einschließlich Optionen mit der Bezeichnung -preview (Vorschau), für die zu testenden Funktionen. Allgemeine Informationen zum Erstellen von Clustern finden Sie unter Erstellen eines Clusters im Amazon-Redshift-Verwaltungshandbuch.
5. Wählen Sie Vorschau-Cluster erstellen aus, um einen Cluster in der Vorschau zu erstellen.

> **Note**
>
> Der `preview_2023`-Track ist der neueste verfügbare Vorschau-Track. Dieser Track unterstützt nur die Erstellung von Clustern mit RA3-Knotentypen. Der Knotentyp DC2 und alle älteren Knotentypen werden nicht unterstützt.

6. Wenn Ihr Vorschau-Cluster verfügbar ist, verwenden Sie Ihren SQL-Client, um Daten zu laden und abzufragen.

Das Vorschau-Feature für Datenkatalog-Ansichten ist nur in den folgenden Regionen verfügbar.

- USA Ost (Ohio): (us-east-2)
- USA Ost (Nord-Virginia): (us-east-1)
- USA West (Nordkalifornien) (us-west-1)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Sie können außerdem eine Vorschau-Arbeitsgruppe erstellen, um Datenkatalog-Ansichten zu testen. Sie können diese Features nicht in der Produktion verwenden und Ihre Vorschau-Arbeitsgruppe auch nicht in eine andere Arbeitsgruppe verschieben. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter „Beta- und Vorschauversionen“ in den AWS -Servicebedingungen. Eine Anleitung zur Erstellung einer Vorschau-Arbeitsgruppe finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe.

Durch das Erstellen von Ansichten in der AWS Glue Data Catalog können Sie ein einziges gemeinsames Ansichtsschema und ein Metadatenobjekt erstellen, das von Engines wie Amazon Athena und Amazon EMR Spark verwendet werden kann. Auf diese Weise können Sie in Ihren Data Lakes und Data Warehouses dieselben Ansichten verwenden, um Ihren Anwendungsfällen gerecht zu werden. Das Besondere an Ansichten im Datenkatalog ist die Tatsache, dass die Zugriffsberechtigungen durch den Benutzer definiert werden, der die Ansicht erstellt hat, und nicht durch den Benutzer, der die Ansicht abfragt. Im Folgenden sind einige Anwendungsfälle aufgeführt, für die das Erstellen von Ansichten im Datenkatalog von Vorteil sein kann:

- Erstellen einer Ansicht, die den Datenzugriff auf der Grundlage der vom Benutzer benötigten Berechtigungen einschränkt. Mithilfe von Ansichten im Datenkatalog können Sie beispielsweise

verhindern, dass Mitarbeiter, die nicht in der Personalabteilung arbeiten, persönlich identifizierbare Informationen (PII) sehen.

- Sicherstellen, dass Benutzer nicht auf unvollständige Datensätze zugreifen können. Durch die Anwendung bestimmter Filter auf Ihre Ansicht im Datenkatalog stellen Sie sicher, dass die Datensätze darin immer vollständig sind.
- Datenkatalog-Ansichten bieten auch einen Sicherheitsvorteil, da sie sicherstellen, dass die zur Erstellung der Ansicht verwendete Abfragedefinition vollständig sein muss, um die Ansicht zu erstellen. Dieser Sicherheitsvorteil bedeutet, dass Ansichten im Datenkatalog nicht anfällig für SQL-Befehle von böswilligen Angreifern sind.
- Ansichten im Datenkatalog bieten dieselben Vorteile wie normale Ansichten, so ermöglichen sie beispielsweise Benutzern den Zugriff auf eine Ansicht, ohne den Benutzern die zugrunde liegende Tabelle zur Verfügung zu stellen.

Um eine Ansicht im Datenkatalog zu erstellen, benötigen Sie eine externe Spectrum-Tabelle, ein Objekt, das in einem von Lake Formation verwalteten Datashare enthalten ist, oder eine Apache-Iceberg-Tabelle.

Definitionen von Datenkatalog-Ansichten werden im AWS Glue Data Catalog gespeichert. Wird verwendet AWS Lake Formation , um Zugriff über Ressourcenzuweisungen, Spaltenzuweisungen oder Tag-basierte Zugriffskontrollen zu gewähren. Weitere Informationen zum Gewähren und Widerrufen des Zugriffs in Lake Formation finden Sie unter Gewähren und Widerrufen von Berechtigungen für Datenkatalog-Ressourcen.

Voraussetzungen

Bevor Sie eine Ansicht im Datenkatalog erstellen können, müssen Sie sicherstellen, dass die folgenden Voraussetzungen erfüllt sind:

- Stellen Sie sicher, dass für Ihre IAM-Rolle die folgende Vertrauensrichtlinie definiert ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
               "Service": [
```

```
                    "glue.amazonaws.com",
                    "lakeformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

- Sie benötigen außerdem die folgende Richtlinie für die Rollenweitergabe.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1",
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com",
                        "lakeformation.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

- Und schließlich benötigen Sie die folgenden Berechtigungen:
  - `Glue:GetDatabase`
  - `Glue:GetDatabases`
  - `Glue:CreateTable`
  - `Glue:GetTable`
  - `Glue:UpdateTable`
  - `Glue:DeleteTable`
  - `Glue:GetTables`

- `Glue:SearchTables`
- `Glue:BatchGetPartition`
- `Glue:GetPartitions`
- `Glue:GetPartition`
- `Glue:GetTableVersion`
- `Glue:GetTableVersions`

Ein nd-to-end Beispiel

Erstellen Sie zunächst ein externes Schema, das auf Ihrer Datenkatalog-Datenbank basiert.

```
CREATE EXTERNAL SCHEMA IF NOT EXISTS external_schema FROM DATA CATALOG DATABASE
 'external_data_catalog_db'
IAM_ROLE 'arn:aws:iam::123456789012:role/sample-role';
```

Sie können jetzt eine Datenkatalog-Ansicht erstellen.

```
CREATE EXTERNAL PROTECTED VIEW external_schema.remote_view
AS SELECT * FROM external_schema.remote_table;
```

Dann können Sie mit der Abfrage Ihrer Ansicht beginnen.

```
SELECT * FROM external_schema.remote_view;
```

Weitere Informationen zu den SQL-Befehlen für Ansichten im Datenkatalog finden Sie unter CREATE EXTERNAL VIEW, ALTER EXTERNAL VIEW und DROP EXTERNAL VIEW.

Überlegungen und Einschränkungen

Im Folgenden sind einige Überlegungen und Einschränkungen zu im Datenkatalog erstellten Ansichten aufgeführt.

- Sie können keine Datenkatalog-Ansicht erstellen, die auf einer anderen Ansicht basiert.
- In einer Datenkatalog-Ansicht können Sie nur 10 Basistabellen haben.
- Der Definierer der Ansicht muss über volle `SELECT GRANTABLE`-Berechtigungen für die Basistabellen verfügen.

- Ansichten können nur Objekte und integrierte Elemente von Lake Formation enthalten. Die folgenden Objekte sind innerhalb einer Ansicht nicht zulässig.
  - Systemtabellen
  - Benutzerdefinierte Funktionen (User-defined functions, UDFs)
  - Redshift-Tabellen, Ansichten, materialisierte Ansichten und Late-Binding-Ansichten, die sich nicht in einem von Lake Formation verwalteten Datashare befinden.
- Ansichten können keine verschachtelten Redshift-Spectrum-Tabellen enthalten.
- Sie können Ansichten nur mithilfe der Zweipunktnotation abfragen. Das Abfragen von Lake-Formation-Ansichten aus einer extern bereitgestellten Datenbank wird nicht unterstützt.
- Der ARN einer Lake-Formation-Tabelle, auf die in einer Redshift-Ansicht verwiesen wird, muss weniger als 127 Zeichen lang sein.
- AWS Glue Repräsentationen der Basisobjekte einer Ansicht müssen sich in derselben AWS-Konto Region wie die Ansicht befinden.

Abfrage von Geodaten in Amazon Redshift

Spatial data (Geodaten) beschreibt die Position und Form einer Geometrie in einem definierten Raum (einem räumlichen Bezugssystem). Amazon Redshift unterstützt Geodaten mit den Datentypen `GEOMETRY` und `GEOGRAPHY`, die Geodaten und optional den SRID (Spatial Reference System Identifier) der Daten enthalten.

Geodaten enthalten geometrische Daten, die zur Darstellung geografischer Merkmale verwendet werden können. Zu den Beispielen für diese Art von Daten zählen Wetterberichte, Wegbeschreibungen, Tweets mit geografischen Positionen, Filialstandorte und Flugrouten. Geodaten spielen eine wichtige Rolle in der Geschäftsanalyse, im Berichtswesen und in der Prognose.

Sie können Geodaten mit Amazon-Redshift-SQL-Funktionen abfragen. Geodaten enthalten geometrische Werte für ein Objekt.

Die `GEOMETRY`-Datentyp-Operationen arbeiten auf der kartesischen Ebene. Obwohl der SRID (Spatial Reference System Identifier) innerhalb des Objekts gespeichert ist, ist diese SRID lediglich eine Kennung des Koordinatensystems und spielt keine Rolle in den Algorithmen, die zur Verarbeitung des `GEOMETRY`-Objekte verwendet werden. Umgekehrt behandeln die Operationen für den `GEOGRAPHY`-Datentyp die Koordinaten in Objekten als sphärische Koordinaten auf einem Sphäroid. Dieses Sphäroid wird durch die SRID definiert, die auf ein geografisches räumliches Bezugssystem verweist. Standardmäßig werden `GEOGRAPHY`-Datentypen mit Raumbezug (SRID) 4326 erstellt, wobei sie auf das World Geodetic System (WGS) 84 verweisen. Weitere Informationen zu SRIDs finden Sie unter Räumliches Bezugssystem in Wikipedia.

Sie können die Funktion „ST_Transform“ verwenden, um die Koordinaten aus verschiedenen räumlichen Bezugssystemen zu transformieren. Nachdem Abschluss der Koordinatentransformation können Sie auch eine einfache Verteilung zwischen den beiden verwenden, solange die Eingabe `GEOMETRY` mit der geografischen SRID codiert wird. Diese Verteilung kopiert einfach die Koordinaten ohne weitere Transformation. Beispiel:

```
SELECT ST_AsEWKT(ST_GeomFromEWKT('SRID=4326;POINT(10 20)')::geography);
```

```
st_asewkt
------------------------
 SRID=4326;POINT(10 20)
```

Um den Unterschied zwischen den Datentypen GEOMETRY und GEOGRAPHY besser zu verstehen, sollten Sie die Entfernung zwischen dem Flughafen Berlin (BER) und dem Flughafen San Francisco (SFO) unter Verwendung des World Geodetic System (WGS) 84 berechnen. Verwendung des Datentyps GEOGRAPHY, das Ergebnis wird in Metern angegeben. Bei Verwendung des Datentyps GEOMETRY mit SRID 4326 wird das Ergebnis in Grad angegeben, die nicht in Meter umgewandelt werden können, da die Entfernung von einem Grad davon abhängt, wo sich die Geometrien auf der Erde befinden.

Berechnungen zum Datentyp GEOGRAPHY werden hauptsächlich für realistische Runderdberechnungen wie die genaue Fläche eines Landes ohne Verzerrung verwendet. Aber sie sind weitaus teurer zu berechnen. Daher kann ST_Transform die Koordinaten in ein geeignetes lokal projiziertes Koordinatensystem umwandeln und die Berechnung für den Datentyp GEOMETRY schneller ausführen.

Mit Hilfe von Geodaten können Sie Abfragen ausführen, um Folgendes zu erreichen:

- Ermitteln Sie den Abstand zwischen zwei Punkten.
- Überprüfen Sie, ob ein Bereich (Polygon) ein weiteres enthält.
- Überprüfen Sie, ob ein Linestring einen anderen Linestring oder ein Polygon schneidet.

Sie können den GEOMETRY-Datentyp verwenden, um die Werte von Geodaten zu speichern. Ein GEOMETRY-Wert in Amazon Redshift kann zweidimensionale (2D)-, dreidimensionale (3D)-, zweidimensionale Datentypen mit Maß (3DM) und vierdimensionale (4D) geometrisch primitive Datentypen definieren:

- Eine zweidimensionale (2D)-Geometrie wird durch zwei kartesische Koordinaten (x, y) in einer Ebene angegeben.
- Eine dreidimensionale (3DZ)-Geometrie wird durch zwei kartesische Koordinaten (x, y, z) in einer Ebene angegeben.
- Eine zweidimensionale Geometrie mit Maß (3DM) wird durch drei Koordinaten (x, y, m) angegeben, wobei die ersten beiden kartesischen Koordinaten in einer Ebene und die dritte eine Messung ist.
- Eine vierdimensionale Geometrie (4D) wird durch vier Koordinaten (x, y, z, m) angegeben, wobei die ersten drei kartesischen Koordinaten in einer Ebene und die vierte ein Maß ist.

Weitere Informationen zu geo-primitiven Datentypen finden Sie unter Well-known text representation of geometry in Wikipedia.

Sie können den GEOGRAPHY-Datentyp verwenden, um die Werte von Geodaten zu speichern. Ein GEOGRAPHY-Wert in Amazon Redshift kann zweidimensionale (2D)-, dreidimensionale (3D)-, zweidimensionale Datentypen mit Maß (3DM) und vierdimensionale (4D) geometrisch primitive Datentypen definieren:

- Eine zweidimensionale Geometrie (2D) wird durch Längen- und Breitengradkoordinaten auf einem Sphäroid angegeben.
- Eine dreidimensionale Geometrie (3D) wird durch Längen- und Breitengrad- sowie Höhenkoordinaten auf einem Sphäroid angegeben.
- Eine zweidimensionale Geometrie mit Maß (3DM) wird durch drei Koordinaten (Längengrad, Breitengrad, Maß) angegeben, wobei die ersten beiden Winkelkoordinaten in einer Ebene liegen und die dritte eine Messung ist.
- Eine vierdimensionale Geometrie (4D) wird durch vier Koordinaten (Längengrad, Breitengrad, Höhe, Maß) angegeben, wobei die ersten drei Längengrad, Breitengrad und Höhe sind und die vierte eine Messung ist.

Weitere Informationen über geografische Koordinatensysteme finden Sie unter Geographisches Koordinatensystem und Sphärisches Koordinatensystem in Wikipedia.

Die Datentypen GEOMETRY und GEOGRAPHY haben die folgenden Subtypen:

- POINT
- LINESTRING
- POLYGON
- MULTIPOINT
- MULTILINESTRING
- MULTIPOLYGON
- GEOMETRYCOLLECTION

Es gibt Amazon-Redshift-SQL-Funktionen, die die folgenden Darstellungen von Geometriedaten unterstützen:

- GeoJSON
- WKT (Well-known text)
- EWKT (Extended well-known text)

- WKB-Darstellung (Well-known binary)
- EWKB (Extended well-known binary)

Sie können zwischen den Datentypen GEOMETRY und GEOGRAPHY umwandeln.

Die folgende SQL-Anweisung verwandelt einen Linestring vom Format GEOMETRY zu GEOGRAPHY.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)')::geography);
```

```
 st_asewkt
---------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

Die folgende SQL-Anweisung verwandelt einen Linestring vom Format GEOGRAPHY zu GEOMETRY.

```
SELECT ST_AsEWKT(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)')::geometry);
```

```
 st_asewkt
---------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

Amazon Redshift bietet viele SQL-Funktionen zum Abfragen von Geodaten. Mit Ausnahme der ST_IsValid-Funktion, erwarten Geofunktionen, die ein GEOMETRY-Objekt als Argument annehmen, dass das GEOMETRY-Objekt ein gültiges Geometrieobjekt ist. Wenn das GEOMETRY- oder GEOGRAPHY-Objekt nicht gültig ist, ist das Verhalten der Geofunktion nicht definiert. Weitere Informationen zur Validierung finden Sie unter Geometrische Gültigkeit.

Weitere Informationen zu SQL-Funktionen zum Abfragen von Geodaten finden Sie unter Geofunktionen.

Weitere Informationen zum Laden von Geodaten finden Sie unter Laden einer Spalte des Datentyps GEOMETRY oder GEOGRAPHY.

Themen

- Tutorial: Verwenden von Geo-SQL-Funktionen mit Amazon Redshift
- Laden eines Shapefile in Amazon Redshift

- Terminologie für Geodaten von Amazon Redshift
- Überlegungen bei Verwendung von Geodaten in Amazon Redshift

Tutorial: Verwenden von Geo-SQL-Funktionen mit Amazon Redshift

In diesem Tutorial wird gezeigt, wie Sie einige der Geo-SQL-Funktionen mit Amazon Redshift verwenden.

Dazu fragen Sie zwei Tabellen mit Geo-SQL-Funktionen ab. Das Tutorial verwendet Daten aus öffentlichen Datensätzen, die Standortdaten von Mietunterkünften mit Postleitzahlen in Berlin, Deutschland, korrelieren.

Themen

- Voraussetzungen
- Schritt 1: Erstellen von Tabellen und Laden von Testdaten
- Schritt 2: Abfrage von Geodaten
- Schritt 3: Bereinigen Ihrer Ressourcen

Voraussetzungen

Für dieses Tutorial benötigen Sie folgende Ressourcen:

- Ein vorhandener Amazon-Redshift-Cluster und -Datenbank, auf die Sie zugreifen und die Sie aktualisieren können. Im vorhandenen Cluster erstellen Sie Tabellen, laden Beispieldaten und führen SQL-Abfragen aus, um Geofunktionen zu demonstrieren. Ihr Cluster sollte mindestens zwei Knoten enthalten. Um zu erfahren, wie Sie einen Cluster erstellen, befolgen Sie die Schritte im Handbuch Erste Schritte in Amazon Redshift.
- Um den Abfrage-Editor von Amazon Redshift zu verwenden, stellen Sie sicher, dass sich Ihr Cluster in einer AWS -Region befindet, die den Abfrageeditor unterstützt. Weitere Informationen finden Sie unter Abfragen einer Datenbank mit dem Abfrage-Editor im Amazon-Redshift-Verwaltungshandbuch.
- AWS Anmeldeinformationen für Ihren Amazon Redshift Redshift-Cluster, die es ihm ermöglichen, Testdaten aus Amazon S3 zu laden. Informationen zum Zugriff auf andere AWS Dienste wie Amazon S3 finden Sie unter Autorisieren von Amazon Redshift für den Zugriff auf AWS Dienste.

- Die benannte AWS Identity and Access Management (IAM-) Rolle`mySpatialDemoRole`, der die verwaltete Richtlinie zum Lesen von Amazon S3 S3-Daten `AmazonS3ReadOnlyAccess` zugeordnet ist. Informationen zum Erstellen einer Rolle mit der Berechtigung zum Laden von Daten aus einem Amazon-S3-Bucket finden Sie unter Authorizing COPY, UNLOAD, and CREATE EXTERNAL SCHEMA operations using IAM roles (Autorisieren der Befehle COPY, UNLOAD und CREATE EXTERNAL SCHEMA mithilfe von IAM-Rollen) im Amazon-Redshift-Verwaltungshandbuch.
- Nachdem Sie die IAM-Rolle `mySpatialDemoRole` erstellt haben, benötigt diese Rolle eine Zuordnung zu Ihrem Amazon-Redshift-Cluster. Weitere Informationen finden Sie unter Autorisierung der Operationen COPY, UNLOAD und CREATE EXTERNAL SCHEMA mit IAM-Rollen im Amazon-Redshift-Verwaltungshandbuch.

Schritt 1: Erstellen von Tabellen und Laden von Testdaten

Die in diesem Tutorial verwendeten Quelldaten befinden sich in Dateien mit den Namen `accommodations.csv` und `zipcodes.csv`.

Die `accommodations.csv`-Datei ist Open-Source-Daten von insideairbnb.com. Die `zipcodes.csv`-Datei enthält Postleitzahlen, die Open-Source-Daten des nationalen Statistik-Instituts Berlin-Brandenburg in Deutschland (Amt für Statistik Berlin-Brandenburg) sind. Beide Datenquellen werden unter einer Creative-Commons-Lizenz bereitgestellt. Die Daten sind auf die Region Berlin, Deutschland, beschränkt. Diese Dateien befinden sich in einem öffentlichen Amazon-S3-Bucket, der mit diesem Tutorial verwendet werden kann.

Optional können Sie die Quelldaten von den folgenden Amazon-S3-Links herunterladen:

- Quelldaten für die `accommodations`-Tabelle.
- Quelldaten für die `zipcode`-Tabelle.

Gehen Sie wie folgt vor, um Tabellen zu erstellen und Testdaten zu laden.

So erstellen Sie Tabellen und laden Testdaten:

1. Öffnen Sie den Abfrage-Editor von Amazon Redshift. Weitere Informationen zur Arbeit mit dem Abfrage-Editor finden Sie unter Querying a database using the query editor (Abfragen für Datenbanken mit dem Abfrage-Editor) im Amazon-Redshift-Verwaltungshandbuch.

2. Löschen Sie alle Tabellen, die in diesem Tutorial verwendet werden, wenn sie bereits in Ihrer Datenbank vorhanden sind. Weitere Informationen finden Sie unter Schritt 3: Bereinigen Ihrer Ressourcen.

3. Erstellen der `accommodations`-Tabelle, um die geografische Lage der einzelnen Unterkünfte (Längen- und Breitengrad), den Namen der Auflistung und andere Geschäftsdaten zu speichern.

   Dieses Tutorial erforscht Zimmervermietungen in Berlin, Deutschland. Die Spalte `shape` speichert geografische Punkte des Standortes von Unterkünften. Die anderen Spalten enthalten Informationen über die Vermietung.

   Führen Sie zum Erstellen der Tabelle `accommodations` die folgende SQL-Anweisung im Abfrage-Editor von Amazon Redshift aus.

```
CREATE TABLE public.accommodations (
  id INTEGER PRIMARY KEY,
  shape GEOMETRY,
  name VARCHAR(100),
  host_name VARCHAR(100),
  neighbourhood_group VARCHAR(100),
  neighbourhood VARCHAR(100),
  room_type VARCHAR(100),
  price SMALLINT,
  minimum_nights SMALLINT,
  number_of_reviews SMALLINT,
  last_review DATE,
  reviews_per_month NUMERIC(8,2),
  calculated_host_listings_count SMALLINT,
  availability_365 SMALLINT
);
```

4. Erstellen Sie die Tabelle `zipcode` im Abfrage-Editor, um Postleitzahlen von Berlin zu speichern.

   Eine Postleitzahl wird in der Spalte `wkb_geometry` als Polygon definiert. In den restlichen Spalten werden zusätzliche Geo-Metadaten zur Postleitzahl beschrieben.

   Führen Sie zum Erstellen der Tabelle `zipcode` die folgende SQL-Anweisung im Abfrage-Editor von Amazon Redshift aus.

```
CREATE TABLE public.zipcode (
  ogc_field INTEGER PRIMARY KEY NOT NULL,
  wkb_geometry GEOMETRY,
```

```
  gml_id VARCHAR(256),
  spatial_name VARCHAR(256),
  spatial_alias VARCHAR(256),
  spatial_type VARCHAR(256)
 );
```

5. Laden Sie die Tabellen mit Beispieldaten.

   Die Beispieldaten für dieses Tutorial werden in einem Amazon S3 S3-Bucket bereitgestellt, der allen authentifizierten AWS Benutzern Lesezugriff ermöglicht. Stellen Sie sicher, dass Sie gültige AWS -Anmeldeinformationen angeben, die den Zugriff auf Amazon S3 gewähren.

   Führen Sie die folgenden COPY-Befehle aus, um Testdaten in Ihre Tabellen zu laden. Ersetzen Sie *account-number* mit Ihrer AWS -Kontonummer. Das Segment der Anmeldedatenzeichenfolge, das in einfachen Anführungszeichen eingeschlossen ist, darf keine Leerzeichen oder Zeilenumbrüche enthalten.

```
COPY public.accommodations
FROM 's3://redshift-downloads/spatial-data/accommodations.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
CREDENTIALS 'aws_iam_role=arn:aws:iam::account-number:role/mySpatialDemoRole';
```

```
COPY public.zipcode
FROM 's3://redshift-downloads/spatial-data/zipcode.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
CREDENTIALS 'aws_iam_role=arn:aws:iam::account-number:role/mySpatialDemoRole';
```

6. Um zu prüfen, ob jede Tabelle korrekt geladen wurde, führen Sie die folgenden Befehle aus.

```
select count(*) from accommodations;
```

```
select count(*) from zipcode;
```

   Die folgende Ergebnistabelle zeigt die Anzahl der Zeilen in jeder Tabelle mit Testdaten.

Tabellenname	Rows
Unterkünfte	22.248
Postleitzahl	190

Schritt 2: Abfrage von Geodaten

Nachdem Ihre Tabellen erstellt und geladen wurden, können Sie sie mithilfe von SQL-SELECT-Anweisungen abfragen. Die folgenden Abfragen veranschaulichen einige der Informationen, die Sie abrufen können. Sie können viele andere Abfragen schreiben, die räumliche Geofunktionen verwenden, um Ihre Anforderungen zu erfüllen.

So fragen Sie Geodaten ab:

1. Abfrage, um die Gesamtzahl der Angebote zu erhalten, die in der Tabelle `accommodations` (siehe unten) gespeichert werden. Das Georeferenzsystem ist das World Geodetic System (WGS) 84, das den eindeutigen Georeferenzbezeichner 4326 aufweist.

```
SELECT count(*) FROM public.accommodations WHERE ST_SRID(shape) = 4326;
```

```
 count
-------
 22248
```

2. Rufen Sie die Geometrieobjekte im WKT-Format (bekannter Text) mit einigen zusätzlichen Attributen ab. Darüber hinaus können Sie überprüfen, ob diese Postleitzahlendaten auch in World Geodetic System (WGS) 84 gespeichert sind, das die Georeferenz-ID (SRID) 4326 verwendet. Geodaten müssen im selben Georeferenzsystem gespeichert werden, um interoperabel zu sein.

```
SELECT ogc_field, spatial_name, spatial_type, ST_SRID(wkb_geometry),
 ST_AsText(wkb_geometry)
FROM public.zipcode
ORDER BY spatial_name;
```

```
ogc_field  spatial_name  spatial_type  st_srid  st_astext
-------------------------------------------------------------
0          10115        Polygon        4326     POLYGON((...))
4          10117        Polygon        4326     POLYGON((...))
8          10119        Polygon        4326     POLYGON((...))
...
(190 rows returned)
```

3. Wählen Sie das Polygon von Berlin Mitte (10117), einem Berliner Bezirk, im GeoJSON-Format, seine Dimension und die Anzahl der Punkte in diesem Polygon aus.

```
SELECT ogc_field, spatial_name, ST_AsGeoJSON(wkb_geometry),
 ST_Dimension(wkb_geometry), ST_NPoints(wkb_geometry)
FROM public.zipcode
WHERE spatial_name='10117';
```

```
ogc_field  spatial_name  spatial_type
 st_dimension  st_npoint
-------------------------------------------------------------------------------------------
4          10117         {"type":"Polygon", "coordinates":[[[...]]]}    2
    331
```

4. Führen Sie den folgenden SQL-Befehl aus, um anzuzeigen, wie viele Unterkünfte sich im Umkreis von 500 Metern vom Brandenburger Tor befinden.

```
SELECT count(*)
FROM public.accommodations
WHERE ST_DistanceSphere(shape, ST_GeomFromText('POINT(13.377704 52.516431)', 4326))
 < 500;
```

```
count
------
  29
```

5. Holen Sie sich den ungefähren Standort des Brandenburger Tores aus Daten, die mit den Unterkünften gespeichert sind, die als in der Nähe aufgeführt sind, indem Sie die folgende Abfrage ausführen.

Diese Abfrage erfordert eine untergeordnete Auswahl. Es führt zu einer anderen Zählung, da der angeforderte Standort nicht mit der vorherigen Abfrage identisch ist, da er näher an den Unterkünften liegt.

```
WITH poi(loc) as (
  SELECT st_astext(shape) FROM accommodations WHERE name LIKE '%brandenburg gate%'
)
SELECT count(*)
FROM accommodations a, poi p
WHERE ST_DistanceSphere(a.shape, ST_GeomFromText(p.loc, 4326)) < 500;
```

```
count
------
  60
```

6. Führen Sie die folgende Abfrage aus, um die Details aller Unterkünfte rund um das Brandenburger Tor anzuzeigen, sortiert nach Preis in absteigender Reihenfolge.

```
SELECT name, price, ST_AsText(shape)
FROM public.accommodations
WHERE ST_DistanceSphere(shape, ST_GeomFromText('POINT(13.377704 52.516431)', 4326))
 < 500
ORDER BY price DESC;
```

```
name                                                    price  st_astext
-------------------------------------------------------------------------------------------
DUPLEX APARTMENT/PENTHOUSE in 5* LOCATION! 7583         300
 POINT(13.3826510209548 52.5159819722552)
DUPLEX-PENTHOUSE IN FIRST LOCATION! 7582                300
 POINT(13.3799997083855 52.5135918444834)
...
(29 rows returned)
```

7. Führen Sie die folgende Abfrage aus, um die teuerste Unterkunft mit ihrer Postleitzahl abzurufen.

```
SELECT
  a.price, a.name, ST_AsText(a.shape),
  z.spatial_name, ST_AsText(z.wkb_geometry)
```

```
FROM accommodations a, zipcode z
WHERE price = 9000 AND ST_Within(a.shape, z.wkb_geometry);
```

```
price   name                                    st_astext
     spatial_name      st_astext
-------------------------------------------------------------------------------------------
9000    Ueber den Dächern Berlins Zentrum    POINT(13.334436985013
 52.4979779501538)    10777             POLYGON((13.3318284987227
 52.4956021172799,...
```

8. Berechnen Sie den maximalen, minimalen oder mittleren Preis von Unterkünften mithilfe einer untergeordneten Abfrage.

   Die folgende Abfrage führt den mittleren Preis der Unterkünfte nach Postleitzahl auf.

```
SELECT
  a.price, a.name, ST_AsText(a.shape),
  z.spatial_name, ST_AsText(z.wkb_geometry)
FROM accommodations a, zipcode z
WHERE
  ST_Within(a.shape, z.wkb_geometry) AND
  price = (SELECT median(price) FROM accommodations)
ORDER BY a.price;
```

```
price name                              st_astext
 spatial_name   st_astext
-------------------------------------------------------------------------------------------

45    "Cozy room Berlin-Mitte"        POINT(13.3864349535358 52.5292016386514)
 10115          POLYGON((13.3658598465795 52.535659581048,...
...
(723 rows returned)
```

9. Führen Sie die folgende Abfrage aus, um die Anzahl der in Berlin aufgeführten Unterkünfte abzurufen. Um die Hotspots zu finden, werden diese nach Postleitzahl gruppiert und nach der Menge des Angebots sortiert.

```
SELECT z.spatial_name as zip, count(*) as numAccommodations
FROM public.accommodations a, public.zipcode z
```

```
WHERE ST_Within(a.shape, z.wkb_geometry)
GROUP BY zip
ORDER BY numAccommodations DESC;
```

```
zip   numaccommodations
---------------------------
10245 872
10247 832
10437 733
10115 664
...
(187 rows returned)
```

Schritt 3: Bereinigen Ihrer Ressourcen

Solange Ihr Cluster ausgeführt wird, fallen Gebühren dafür an. Nachdem Sie dieses Tutorial abgeschlossen haben, können Sie Ihren Beispiel-Cluster löschen.

Wenn Sie den Cluster behalten möchten, den von den Testdatentabellen beanspruchten Speicherplatz jedoch zurückgewinnen möchten, führen Sie die folgenden Befehle zum Löschen der Tabellen aus.

```
drop table public.accommodations cascade;
```

```
drop table public.zipcode cascade;
```

Laden eines Shapefile in Amazon Redshift

Sie können den Befehl COPY verwenden, um Esri Shapefiles, die in Amazon S3 gespeichert sind, in Amazon-Redshift-Tabellen aufzunehmen. Ein Shapefile speichert die geometrische Position und Attributinformationen von geografischen Features in einem Vektorformat. Das Shapefile-Format kann Geo-Objekte wie Punkte, Linien und Polygone beschreiben. Weitere Informationen zu Shapefiles finden Sie unter Shapefile in Wikipedia.

Der Befehl COPY unterstützt den Datenformatparameter `SHAPEFILE`. Standardmäßig ist die erste Spalte des Shapefile entweder eine `GEOMETRY`- oder `IDENTITY`-Spalte. Alle nachfolgenden

Spalten folgen der im Shapefile angegebenen Reihenfolge. Die Zieltabelle muss sich jedoch nicht in diesem exakten Layout befinden, da Sie das COPY-Spalten-Mapping verwenden können, um die Reihenfolge zu definieren. Weitere Informationen zur Unterstützung von Shapefile-Befehlen finden Sie unter SHAPEFILE.

In einigen Fällen kann die resultierende Geometriegröße größer sein als das Maximum zum Speichern einer Geometrie in Amazon Redshift. Wenn ja, können Sie den COPY-Befehl `SIMPLIFY` oder `SIMPLIFY AUTO` verwenden, um die Geometrien während der Erfassung wie folgt zu vereinfachen:

- Geben Sie `SIMPLIFY` *`tolerance`* an, um alle Geometrien während der Erfassung mithilfe des Ramer-Douglas-Peucker-Algorithmus und der angegebenen Toleranz zu vereinfachen.
- Geben Sie `SIMPLIFY AUTO` ohne Toleranz an, um mit dem Ramer-Douglas-Peucker-Algorithmus nur Geometrien zu vereinfachen, die größer als die maximale Größe sind. Dieser Ansatz berechnet die minimale Toleranz, die groß genug ist, um das Objekt innerhalb der maximalen Größenbeschränkung zu speichern.
- Geben Sie `SIMPLIFY AUTO` *`max_tolerance`* an, um mit dem Ramer-Douglas-Peucker-Algorithmus nur Geometrien zu vereinfachen, die größer als die maximale Größe sind. Dieser Ansatz stellt sicher, dass die Toleranz die maximale Toleranz nicht überschreitet.

Weitere Informationen zur maximalen Größe eines `GEOMETRY`-Datenwerts finden Sie unter Überlegungen bei Verwendung von Geodaten in Amazon Redshift.

In einigen Fällen ist die Toleranz niedrig genug, dass der Datensatz nicht unter die maximale Größe eines `GEOMETRY`-Datenwerts sinkt. In diesen Fällen können Sie die Option `MAXERROR` des COPY-Befehls verwenden, um alle oder eine bestimmte Anzahl von Erfassungsfehlern zu ignorieren.

Der Befehl COPY unterstützt auch das Laden von GZIP-Shapefiles. Geben Sie hierfür den Parameter COPY GZIP an. Mit dieser Option müssen alle Shapefile-Komponenten unabhängig voneinander komprimiert werden und dasselbe Komprimierungssuffix verwenden.

Wenn eine Projektionsbeschreibungsdatei (.prj) mit dem Shapefile vorhanden ist, bestimmt Redshift damit die ID des räumlichen Bezugssystems (SRID). Wenn die SRID gültig ist, wird der resultierenden Geometrie diese SRID zugewiesen. Wenn der mit der Eingabegeometrie verbundene SRID-Wert nicht existiert, hat die resultierende Geometrie den SRID-Wert null. Sie können die automatische Erkennung der ID des räumlichen Bezugssystems auf Vortragsbene deaktivieren, indem Sie `SET read_srid_on_shapefile_ingestion` zu `OFF` verwenden.

Fragen Sie die Systemansicht SYS_SPATIAL_SIMPLIFY oder SVL_SPATIAL_SIMPLIFY ab, um zusammen mit der berechneten Toleranz anzuzeigen, welche Datensätze vereinfacht wurden. Wenn Sie SIMPLIFY *tolerance* angeben, enthält diese Ansicht einen Datensatz für jeden COPY-Vorgang. Andernfalls enthält sie einen Datensatz für jede vereinfachte Geometrie. Weitere Informationen finden Sie unter SYS_SPATIAL_SIMPLIFY oder SVL_SPATIAL_SIMPLIFY.

Beispiele zum Laden eines Shapefiles finden Sie unter Laden eines Shapefile in Amazon Redshift.

Terminologie für Geodaten von Amazon Redshift

Die folgenden Beschreibungen werden verwendet, um einige Geofunktionen von Amazon Redshift zu beschreiben.

Begrenzungsrahmen

Eine Bounding Box einer Geometrie oder Geographie ist definiert als Vektorprodukt (über Dimensionen hinweg) der Grenzen der Koordinaten aller Punkte in der Geometrie oder Geographie. Bei zweidimensionalen Geometrien ist die Bounding Box ein Rechteck, das alle Punkte in der Geometrie vollständig enthält. Beispielsweise ist eine Bounding Box des POLYGON((0 0,1 0,0 2,0 0))-Polygons das Rechteck, das durch die Punkte (0, 0) und (1, 2) als seine unteren linken und oberen rechten Ecken definiert wird. Amazon Redshift berechnet und speichert eine Bounding Box innerhalb einer Geometrie, um geometrische Prädikate und Verbindungen zu beschleunigen. Wenn sich beispielsweise die Bounding Boxes zweier Geometrien nicht schneiden, können sich diese beiden Geometrien nicht überschneiden. Sie können sich nicht mit dem Prädikat „ST_Intersects“ in der Ergebnismenge einer räumlichen Verbindung befinden.

Sie können Geofunktionen verwenden, um Support für eine Bounding Box hinzuzufügen (AddBBox), zu löschen (DropBBox), und zu bestimmen (SupportsBBox). Amazon Redshift unterstützt die Vorberechnung von Bounding Boxes für alle Geometrie-Untertypen.

Das folgende Beispiel zeigt, wie Sie vorhandene Geometrien in einer Tabelle aktualisieren, um sie mit einer Bounding Box zu speichern. Wenn sich Ihr Cluster auf der Clusterversion 1.0.26809 oder höher befindet, werden alle neuen Geometrien standardmäßig mit einer vorberechneten Bounding Box erstellt.

```
UPDATE my_table SET geom = AddBBox(geom) WHERE SupportsBBox(geom) = false;
```

Nachdem Sie vorhandene Geometrien aktualisiert haben, wird empfohlen, den Befehl VACUUM für die aktualisierte Tabelle auszuführen. Weitere Informationen finden Sie unter VACUUM.

Weitere Informationen zum Festlegen der Verschlüsselung von Geometrien mit einer Bounding Box während einer Sitzung finden Sie unter default_geometry_encoding.

Geometrische Gültigkeit

Geometrische Algorithmen, die von Amazon Redshift verwendet werden, gehen davon aus, dass die Eingabegeometrie eine gültige Geometrie ist. Wenn eine Eingabe in einen Algorithmus ungültig ist, ist das Ergebnis nicht definiert. Im folgenden Abschnitt werden die geometrischen Gültigkeitsdefinitionen beschrieben, die Amazon Redshift für jeden Geometrie-Untertyp verwendet.

Point

Ein Punkt wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Der Punkt kann der leere Punkt sein.
- Alle Punktkoordinaten sind endliche Gleitkommazahlen.

Der Punkt kann der leere Punkt sein.

Linestring

Ein Linestring wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Der Linestring ist leer, d. h. er enthält keine Punkte.
- Alle Punkte in einem nicht leeren Linestring haben Koordinaten, die endliche Gleitkommazahlen sind.
- Der Linestring, wenn er nicht leer ist, muss eindimensional sein; das heißt, er kann nicht auf einen Punkt reduziert werden.

Ein Linestring darf keine leeren Punkte enthalten.

Ein Linestring kann doppelte aufeinanderfolgende Punkte aufweisen.

Ein Linestring kann sich selbst überschneiden.

Polygon

Ein Polygon wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Das Polygon ist leer, d. h. es enthält keine Ringe.
- Ist es nicht leer, ist ein Polygon gültig, wenn alle der folgenden Bedingungen erfüllt sind:

- Alle Ringe des Polygons sind gültig. Ein Ring wird als gültig angesehen, wenn alle der folgenden Bedingungen erfüllt sind:
  - Alle Punkte des Rings haben Koordinaten, bei denen es sich um endliche Gleitkommazahlen handelt.
  - Der Ring ist geschlossen, das heißt, sein erster Punkt und sein letzter Punkt stimmen überein.
  - Der Ring hat keine internen Überschneidungen.
  - Der Ring ist zweidimensional.
- Die Ringe des Polygons haben konsistente Ausrichtungen. Das heißt, wenn Sie einen Ring durchqueren, befindet sich das Innere des Polygons entweder rechts oder links. Wenn der Außenring eines Polygons im Uhrzeigersinn oder gegen den Uhrzeigersinn ausgerichtet ist, müssen alle inneren Ringe des Polygons die gleiche Ausrichtung gegen den Uhrzeigersinn oder im Uhrzeigersinn haben.
- Alle inneren Ringe müssen innerhalb des äußeren Rings des Polygons liegen.
- Innenringe können nicht verschachtelt werden, d. h. ein Innenring darf sich nicht in einem anderen Innenring befinden.
- Innen- und Außenringe können sich nur an einer begrenzten Anzahl von Punkten schneiden.
- Das Innere des Polygons muss einfach verbunden werden.

Ein Polygon darf keine leeren Punkte enthalten.

Multipoint

Ein Multipoint wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Der Multipoint ist leer, d. h. er enthält keine Punkte.
- Ein Multipoint ist nicht leer, und alle Punkte sind gemäß der Punktgültigkeitsdefinition gültig.

Ein Multipoint kann einen oder mehrere leere Punkte enthalten.

Ein Multipoint kann doppelte Punkte haben.

Multilinestring

Ein Multilinestring wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Der Multilinestring ist leer, d. h. er enthält keine Linestrings.
- Alle Linestrings in einem nicht leeren Multilinestring sind gemäß der Linestring-Gültigkeitsdefinition gültig.

Ein nicht leerer Multilinestring, der nur aus leeren Linestrings besteht, wird als gültig angesehen.

Ein leerer Linestring in einem Multilinestring wirkt sich nicht auf ihre Gültigkeit aus.

Ein Multilinestring kann doppelte aufeinanderfolgende Punkte aufweisen.

Ein Multilinestring kann sich intern überschneiden.

Ein Multilinestring darf keine leeren Punkte enthalten.

Multipolygon

Ein Multipolygon wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Das Multipolygon enthält keine Polygone (es ist leer).
- Das Multipolygon ist nicht leer, und alle folgenden Bedingungen treffen zu:
  - Alle Polygone im Multipolygon sind gültig.
  - Zwei Polygone im Multipolygon können sich nicht an einer unendlichen Anzahl von Punkten schneiden. Dies bedeutet insbesondere, dass sich das Innere von zwei beliebigen Polygonen nicht überschneiden kann und dass sie sich nur an einer begrenzten Anzahl von Punkten berühren können.

Ein leeres Polygon in einem Multipolygon macht ein Multipolygon nicht ungültig.

Ein Multipolygon darf keine leeren Punkte enthalten.

Geometriesammlung

Eine Geometriesammlung wird als gültig angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Die Geometriesammlung ist leer; d. h. sie enthält keine Geometrien.
- Alle Geometrien in einer nicht leeren Geometriesammlung sind gültig.

Diese Definition gilt zwar rekursiv für verschachtelte Geometriesammlungen.

Eine Geometriesammlung kann leere Punkte und Multipoints mit leeren Punkten enthalten.

Geometrische Einfachheit

Geometrische Algorithmen, die von Amazon Redshift verwendet werden, gehen davon aus, dass die Eingabegeometrie eine gültige Geometrie ist. Wenn eine Eingabe in einen Algorithmus ungültig

ist, ist die Einfachheitsprüfung nicht definiert. Im folgenden Abschnitt werden die geometrischen Einfachheitsdefinitionen beschrieben, die Amazon Redshift für jeden Geometrie-Untertyp verwendet.

Point

Ein gültiger Punkt wird als einfach angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Ein gültiger Punkt wird immer als einfach angesehen.
- Ein leerer Punkt gilt als einfach.

Linestring

Ein gültiger Linestring wird als einfach angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Der Linestring ist leer.
- Der Linestring ist nicht leer, und alle der folgenden Bedingungen sind erfüllt:
  - Er hat keine doppelten aufeinanderfolgenden Punkte.
  - Er hat keine Selbstüberschneidungen, außer möglicherweise für seinen ersten und letzten Punkt, der übereinstimmen kann. Mit anderen Worten, der Linestring kann nur an Grenzpunkten interne Überschneidungen aufweisen.

Polygon

Ein gültiges Polygon gilt als einfach, wenn es keine doppelten aufeinander folgenden Punkte enthält.

Multipoint

Ein gültiger Multipoint wird als einfach angesehen, wenn eine der folgenden Bedingungen zutrifft:

- Der Multipoint ist leer, d. h. er enthält keine Punkte.
- Es stimmen keine zwei nicht leere Punkte des Multipoints überein.

Multilinestring

Ein gültiger Multilinestring wird als einfach angesehen, wenn eine der folgenden Bedingungen erfüllt sind:

- Der Multilinestring ist leer.
- Der Multilinestring ist nicht leer, und alle der folgenden Bedingungen sind erfüllt:
  - Alle Linestrings sind einfach.
  - Zwei beliebige Linestrings des Multilinestring schneiden sich nicht, außer an Punkten, die Begrenzungspunkte der beiden Linestrings sind.

Ein nicht leerer Multilinestring, der nur aus leeren Linestrings besteht, wird nur als leer angesehen.

Ein leerer Linestring in einem Multilinestring wirkt sich nicht auf die Einfachheit aus.

Ein geschlossener Linestring in einem Multilinestring kann sich nicht mit einem anderen Linestring im Multilinestring überschneiden.

Ein Multilinestring kann keine Linestrings mit doppelten aufeinanderfolgende Punkte aufweisen.

Multipolygon

Ein gültiges Multipolygon gilt als einfach, wenn es keine doppelten aufeinander folgenden Punkte enthält.

Geometriesammlung

Eine gültige Geometriesammlung wird als einfach angesehen, wenn eine der folgenden Bedingungen erfüllt sind:

- Die Geometriesammlung ist leer; d. h. sie enthält keine Geometrien.
- Alle Geometrien in einer nicht leeren Geometriesammlung sind einfach.

Diese Definition gilt zwar rekursiv für verschachtelte Geometriesammlungen.

H3

H3 ist ein hierarchisches Indizierungsrastersystem für Geodaten, das die Möglichkeit bietet, räumliche Koordinaten bis zu einer Auflösung auf Quadratmeterebene zu indizieren. Indizierte Daten können über unterschiedliche Datensätze hinweg verknüpft und mit unterschiedlichen Genauigkeitsstufen aggregiert werden. H3 ermöglicht eine Reihe von Algorithmen und Optimierungen, die auf dem Raster basieren, darunter „Nächste Nachbarn“, „Kürzester Pfad“, Gradientenglättung und mehr. H3-Indizes beziehen sich auf Zellen, die entweder Sechsecke oder Fünfecke sein können. Der Raum ist bei gegebener Auflösung hierarchisch unterteilt. H3 unterstützt 16 Auflösungen von 0 bis einschließlich 15. Dabei ist `0` die größte und `15` die feinste Auflösung.

Amazon Redshift bietet die folgenden H3-Geofunktionen:

- H3_ FromLongLat
- H3_ FromPoint
- H3_Polyfill

Überlegungen bei Verwendung von Geodaten in Amazon Redshift

Nachfolgend sind die Überlegungen bei der Verwendung von Geodaten in Amazon Redshift aufgeführt:

- Die maximale Größe eines GEOMETRY- oder GEOGRAPHY-Objekts beträgt 1 048 447 Byte.
- Amazon Redshift Spectrum unterstützt Geodaten nicht standardmäßig. Daher können Sie keine externe Tabelle mit einer GEOMETRY- oder GEOGRAPHY-Spalte erstellen oder ändern.
- Datentypen für benutzerdefinierte Python-Funktionen (UDFs) unterstützen den Datentyp GEOMETRY oder GEOGRAPHY nicht.
- Sie können eine GEOMETRY- oder GEOGRAPHY-Spalte nicht als Sortierschlüssel oder Verteilungsschlüssel einer Amazon-Redshift-Tabelle verwenden.
- Sie können keine GEOMETRY- oder GEOGRAPHY-Spalten in ORDER BY-, GROUP BY- oder DISTINCT-SQL-Klauseln verwenden.
- Sie können GEOMETRY- oder GEOGRAPHY-Spalten in vielen SQL-Funktionen nicht verwenden.
- Sie können nicht in jedem Format eine UNLOAD-Operation für die Spalten GEOMETRY oder GEOGRAPHY durchführen. Sie können für die die GEOMETRY- oder GEOGRAPHY-Spalten den Befehl UNLOAD in Text- oder CSV-Dateien (durch Kommas getrennte Werte) ausführen. In diesem Fall werden GEOMETRY- oder GEOGRAPHY-Daten im hexadezimalen EWKB-Format geschrieben. Wenn die Größe der EWKB-Daten mehr als 4 MB beträgt, erfolgt eine Warnung, da die Daten später nicht mehr in eine Tabelle geladen werden können.
- Die unterstützte Kompressionskodierung von GEOMETRY- oder GEOGRAPHY-Daten ist RAW.
- Wenn Sie JDBC- oder ODBC-Treiber verwenden, verwenden Sie das benutzerdefinierte Typ-Mapping. In diesem Fall muss die Client-Anwendung Informationen darüber haben, welche Parameter eines ResultSet-Objekts GEOMETRY- oder GEOGRAPHY-Objekte sind. Die Operation ResultSetMetadata gibt den VARCHAR-Typ zurück.
- Um das geografische Datum aus einer SHAPEFILE zu kopieren, erfassen Sie zuerst in eine GEOMETRY-Spalte und wandeln Sie dann die Objekte in GEOGRAPHY-Objekte um.

Die folgenden Nicht-Geo-Funktionen können eine Eingabe vom Typ GEOMETRY oder GEOGRAPHY oder Spalten vom Typ GEOMETRY oder GEOGRAPHY akzeptieren:

- Die Aggregatfunktion COUNT
- Die bedingten Ausdrücke COALESCE und NVL

- CASE-Ausdrücke
- Die Standardkodierung für GEOMETRY und GEOGRAPHY ist RAW. Weitere Informationen finden Sie unter Kompressionskodierungen.

Abfragen von Daten mit Verbundabfragen in Amazon Redshift

Mit Verbundabfragen in Amazon Redshift können Sie Daten über Betriebsdatenbanken, Data Warehouses und Data Lakes hinweg abfragen und analysieren. Mit der Funktion „Federated Query“ (Verbundabfrage) können Sie Abfragen aus Amazon Redshift von Live-Daten in externe Datenbanken mit Abfragen über Ihre Amazon-Redshift- und Amazon-S3-Umgebungen hinweg integrieren. Verbundabfragen können mit externen Datenbanken in Amazon RDS for PostgreSQL, Amazon Aurora PostgreSQL-kompatible Edition, Amazon RDS for MySQL und Amazon Aurora MySQL-kompatible Edition verwendet werden.

Mit Verbundabfragen können Sie Live-Daten als Teil Ihrer Business Intelligence (BI)- und Reporting-Anwendungen verwenden. Um beispielsweise die Datenaufnahme in Amazon Redshift zu vereinfachen, können Sie mit Verbundabfragen Folgendes ausführen:

- Direkte Abfrage von Betriebsdatenbanken.
- Schnelles Anwenden von Transformationen.
- Laden von Daten in die Zieltabellen ohne komplexe Extrahieren, Transformieren oder Laden von (ETL) Pipelines.

Um die Datenbewegungen über das Netzwerk zu reduzieren und die Leistung zu verbessern, verteilt Amazon Redshift einen Teil der Berechnung für Verbundabfragen direkt in die Remote-Betriebsdatenbanken. Amazon Redshift verwendet auch seine Parallelverarbeitungskapazität, um die Ausführung dieser Abfragen bei Bedarf zu unterstützen.

Bei der Ausführung von Verbundabfragen stellt Amazon Redshift zunächst eine Client-Verbindung zur RDS- oder Aurora-DB-Cluster-DB-Instance vom Leader-Knoten her, um Tabellenmetadaten abzurufen. Aus einem Rechenknoten gibt Amazon Redshift Unterabfragen mit einem nach unten verschobenen Prädikat aus und ruft die Ergebniszeilen ab. Amazon Redshift verteilt die Ergebniszeilen dann zur weiteren Verarbeitung an die Rechenknoten.

Details zu Abfragen, die an die Amazon-Aurora-PostgreSQL-Datenbank oder Amazon-RDS-for-PostgreSQL-Datenbank gesendet werden, werden in der Systemansicht SVL_FEDERATED_QUERY protokolliert.

Themen

- Erste Schritte mit der Verwendung von Verbundabfragen an PostgreSQL
- Erste Schritte mit der Verwendung von Verbundabfragen an PostgreSQL mit AWS CloudFormation
- Erste Schritte bei der Verwendung von Verbundabfragen für MySQL
- Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen
- Beispiele für die Verwendung einer Verbundabfrage
- Datentypunterschiede zwischen Amazon Redshift und unterstützten PostgreSQL- oder MySQL-Datenbanken
- Einschränkungen beim Zugriff auf Verbunddaten mit Amazon Redshift

Erste Schritte mit der Verwendung von Verbundabfragen an PostgreSQL

Verfolgen Sie diesen allgemeinen Ansatz, um eine Verbundabfrage zu erstellen:

1. Richten Sie die Verbindung zwischen Ihrem Amazon-Redshift-Cluster und Ihrer Amazon-RDS- oder Aurora-PostgreSQL-DB-Instance ein.

   Stellen Sie dazu sicher, dass Ihre RDS-PostgreSQL- oder Aurora-PostgreSQL-DB-Instance Verbindungen von Ihrem Amazon-Redshift-Cluster-akzeptieren kann. Wir empfehlen, dass sich Ihr Amazon-Redshift-Cluster und die Amazon RDS- oder Aurora-PostgreSQL-Instance in derselben Virtual Private Cloud (VPC) und Subnetzgruppe befinden. Auf diese Weise können Sie die Sicherheitsgruppe für den Amazon-Redshift-Cluster zu den eingehenden Regeln der Sicherheitsgruppe für Ihre RDS- oder Aurora-PostgreSQL-DB-Instance hinzufügen.

   Sie können auch VPC-Peering oder andere Netzwerke einrichten, mit denen Amazon Redshift Verbindungen zu Ihrer RDS- oder Aurora-Postgre-SQL-Instance herstellen kann. Weitere Informationen zu VPC-Networking finden Sie in folgenden Quellen.

   - Was ist VPC Peering? im Amazon VPC Peering Guide
   - Arbeiten mit einer DB-Instance in einer VPC im Amazon-RDS-Benutzerhandbuch

   > **Note**
   >
   > Es gibt Fälle, in denen Sie das erweiterte VPC-Routing aktivieren müssen. Beispiel: Wenn sich Ihr Amazon-Redshift-Cluster in einer anderen VPC befindet als Ihre RDS- oder Aurora-PostgreSQL-Instance oder wenn sich diese in derselben VPC befinden und Ihre

Routen dies erfordern. Andernfalls kann es beim Ausführen einer Verbundabfrage zu Timeout-Fehlern kommen.

2. Richten Sie Geheimnisse AWS Secrets Manager für Ihre RDS PostgreSQL- und Aurora PostgreSQL-Datenbanken ein. Verweisen Sie dann in den AWS Identity and Access Management (IAM-) Zugriffsrichtlinien und -rollen auf die Geheimnisse. Weitere Informationen finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.

   > **Note**
   >
   > Wenn Ihr Cluster das erweiterte VPC-Routing verwendet, müssen Sie möglicherweise einen Schnittstellen-VPC-Endpunkt für konfigurieren AWS Secrets Manager. Dies ist erforderlich, wenn die VPC und das Subnetz Ihres Amazon Redshift Redshift-Clusters keinen Zugriff auf den öffentlichen Endpunkt haben. AWS Secrets Manager Wenn Sie einen VPC-Schnittstellenendpunkt verwenden, AWS Secrets Manager wird die Kommunikation zwischen dem Amazon Redshift Redshift-Cluster in Ihrer VPC und privat von Ihrer VPC zur Endpunktschnittstelle weitergeleitet. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Amazon VPC Benutzerhandbuch.

3. Wenden Sie die zuvor erstellte IAM-Rolle auf den Amazon-Redshift-Cluster an. Weitere Informationen finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.
4. Verbinden Sie sich mit Ihren RDS-PostgreSQL- und Auora-Postgre-SQL-Datenbanken mit einem externen Schema. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA. Beispiele für die Verwendung von Verbundabfragen finden Sie unter Beispiele für die Verwendung einer Verbundabfrage.
5. Führen Sie die SQL-Abfragen aus, die auf das externe Schema verweisen, das auf Ihre RDS-PostgreSQL- und Aurora-PostgreSQL-Datenbanken verweist.

Erste Schritte mit der Verwendung von Verbundabfragen an PostgreSQL mit AWS CloudFormation

Sie können Verbundabfragen verwenden, um über im Betrieb befindliche Datenbanken hinweg abzufragen. In diesem Leitfaden für die ersten Schritte können Sie die Einrichtung automatisieren, indem Sie einen AWS CloudFormation Beispielstapel verwenden, um eine föderierte Abfrage von einem Amazon Redshift Redshift-Cluster an eine serverlose Aurora PostgreSQL-Datenbank zu

aktivieren. Sie können schnell loslegen, ohne zur Ressourcenbereitstellung SQL-Anweisungen ausführen zu müssen.

Der Stack erstellt ein externes Schema, das auf Ihre Aurora-PostgreSQL-Instance verweist, die Tabellen mit Beispieldaten enthält. Sie können Tabellen im externen Schema von Ihrem Redshift-Cluster abfragen.

Wenn Sie stattdessen mit Verbundabfragen beginnen möchten, indem Sie SQL-Anweisungen ausführen, um ein externes Schema einzurichten, ohne es zu verwenden, finden Sie unter. CloudFormation Erste Schritte mit der Verwendung von Verbundabfragen an PostgreSQL

Bevor Sie den CloudFormation Stack für Verbundabfragen ausführen, stellen Sie sicher, dass Sie über eine serverlose Amazon Aurora PostgreSQL-Compatible Edition-Datenbank mit aktivierter Daten-API verfügen. Sie können die Daten-API in den Datenbankeigenschaften aktivieren. Wenn Sie die Einstellung nicht finden können, überprüfen Sie, ob Sie eine Serverless-Instance von Aurora PostgreSQL ausführen. Stellen Sie außerdem sicher, dass Sie über einen Amazon-Redshift-Cluster verfügen, der RA3-Knoten verwendet. Wir empfehlen, dass sich Ihr Redshift-Cluster und die Serverless-Aurora-PostgreSQL-Instance in derselben Virtual Private Cloud (VPC) und Subnetzgruppe befinden. Auf diese Weise können Sie die Sicherheitsgruppe für den Amazon-Redshift-Cluster zu den eingehenden Regeln der Sicherheitsgruppe für Ihre Aurora-PostgreSQL-DB-Instance hinzufügen.

Weitere Informationen zu den ersten Schritten beim Einrichten eines Amazon Redshift Redshift-Clusters finden Sie unter Bereitgestellte Amazon Redshift Redshift-Cluster. Weitere Informationen zum Einrichten von Ressourcen mit finden Sie unter Was CloudFormation ist? AWS CloudFormation . Weitere Informationen zum Einrichten einer Aurora-DB-Cluster-Datenbank finden Sie unter Erstellen eines Aurora-DB-Clusters Serverless v1-DB-Cluster.

Starten eines CloudFormation Stacks für Redshift-Verbundabfragen

Gehen Sie wie folgt vor, um Ihren CloudFormation Stack für Amazon Redshift zu starten, um föderierte Abfragen zu aktivieren. Stellen Sie zuvor sicher, dass Sie Ihren Amazon-Redshift-Cluster und Ihre Serverless-Aurora-PostgreSQL-Instance eingerichtet haben.

Um Ihren CloudFormation Stack für Verbundabfragen zu starten

1. Klicken Sie hier auf CFN-Stack starten, um den CloudFormation Dienst im zu starten. AWS Management Console

   Melden Sie sich an, wenn Sie dazu aufgefordert werden.

Der Stack-Erstellungsprozess beginnt und verweist auf eine CloudFormation Vorlagendatei, die in Amazon S3 gespeichert ist. Eine CloudFormation Vorlage ist eine Textdatei im JSON-Format, die AWS Ressourcen deklariert, aus denen ein Stack besteht.

2. Klicken Sie auf Next (Weiter) und geben Sie die Stack-Details ein.
3. Geben Sie unter Parameters (Parameter) für den Cluster Folgendes ein:

   - Den Namen des Amazon-Redshift-Clusters, zum Beispiel **`ra3-consumer-cluster`**
   - Einen bestimmten Datenbanknamen, zum Beispiel **`dev`**
   - Den Namen eines Datenbankbenutzers, zum Beispiel **`consumeruser`**

   Geben Sie auch die Parameter für die Aurora-DB-Cluster-Datenbank ein, einschließlich Benutzer, Datenbankname, Port und Endpunkt. Wir empfehlen, einen Testcluster und eine Serverless-Testdatenbank zu verwenden, da der Stack mehrere Datenbankobjekte erstellt.

   Wählen Sie Next (Weiter).

   Die Stack-Optionen werden angezeigt.
4. Klicken Sie auf Next (Weiter), um die Standardeinstellungen zu übernehmen.
5. Wählen Sie unter Funktionen die Option Ich bestätige, dass AWS CloudFormation möglicherweise IAM-Ressourcen erstellt werden.
6. Wählen Sie Stack erstellen aus.

Wählen Sie Stack erstellen aus. CloudFormation stellt die Vorlagenressourcen bereit, was etwa 10 Minuten dauert, und erstellt ein externes Schema.

Falls während der Erstellung des Stacks ein Fehler auftritt, unternehmen Sie folgende Schritte:

- Auf der Registerkarte CloudFormation Ereignisse finden Sie Informationen, die Ihnen bei der Behebung des Fehlers helfen können.
- Stellen Sie sicher, dass Sie den richtigen Namen, Datenbanknamen und den Datenbankbenutzernamen für den Redshift-Cluster eingegeben haben. Überprüfen Sie auch die Parameter für die Aurora-PostgreSQL-Instance.
- Stellen Sie sicher, dass Ihr Cluster RA3-Knoten hat.
- Stellen Sie sicher, dass sich Ihre Datenbank und der Redshift-Cluster im selben Subnetz und in derselben Sicherheitsgruppe befinden.

Abfragen von Daten aus dem externen Schema

Stellen Sie für den folgenden Vorgang sicher, dass Sie die erforderlichen Berechtigungen zum Ausführen von Abfragen im Cluster und der beschriebenen Datenbank besitzen.

So fragen Sie eine externe Datenbank mit Verbundabfrage ab

1. Stellen Sie mit einem Client-Tool wie dem Redshift-Abfrage-Editor eine Verbindung zur Redshift-Datenbank her, die Sie beim Erstellen des Stacks eingegeben haben.
2. Fragen Sie nach dem vom Stack erstellten externen Schema ab.

   ```
   select * from svv_external_schemas;
   ```

   In der Ansicht SVV_EXTERNAL_SCHEMAS werden Informationen über verfügbare externe Schemas zurückgegeben. In diesem Fall wird das vom Stack erstellte externe Schema zurückgegeben, `myfederated_schema`. Möglicherweise werden auch andere externe Schemas zurückgegeben, wenn Sie diese eingerichtet haben. In der Ansicht wird auch die zugeordnete Datenbank des Schemas zurückgegeben. Die Datenbank ist die Aurora-DB-Cluster-Datenbank, die Sie bei der Erstellung des Stacks eingegeben haben. Der Stack fügt der Aurora-DB-Cluster-Datenbank eine Tabelle hinzu, die aufgerufen wird`category`, und eine weitere Tabelle namens`sales`.
3. Führen Sie SQL-Abfragen für Tabellen im externen Schema aus, das auf Ihre Aurora PostgreSQL-Datenbank verweist. Das folgende Beispiel zeigt eine Abfrage.

   ```
   SELECT count(*) FROM myfederated_schema.category;
   ```

   Die Tabelle `category` gibt mehrere Datensätze zurück. Sie können auch Datensätze aus der Tabelle `sales` zurückgeben.

   ```
   SELECT count(*) FROM myfederated_schema.sales;
   ```

   Weitere Beispiele finden Sie unter Beispiele für die Verwendung einer Verbundabfrage.

Erste Schritte bei der Verwendung von Verbundabfragen für MySQL

Verfolgen Sie diesen allgemeinen Ansatz, um eine Verbundabfrage an MySQL-Datenbanken zu erstellen:

1. Richten Sie die Verbindung zwischen Ihrem Amazon-Redshift-Cluster und Ihrer Amazon-RDS- oder Aurora-MySQL-DB-Instance ein.

   Stellen Sie dazu sicher, dass Ihre RDS-MySQL- oder Aurora-MySQL-DB-Instance Verbindungen von Ihrem Amazon-Redshift-Cluster-akzeptieren kann. Wir empfehlen, dass sich Ihr Amazon-Redshift-Cluster und die Amazon RDS- oder Aurora-MySQL-Instance in derselben Virtual Private Cloud (VPC) und Subnetzgruppe befinden. Auf diese Weise können Sie die Sicherheitsgruppe für den Amazon-Redshift-Cluster zu den eingehenden Regeln der Sicherheitsgruppe für Ihre RDS- oder Aurora-MySQL-DB-Instance hinzufügen.

   Sie können auch VPC-Peering oder andere Netzwerke einrichten, mit denen Amazon Redshift Verbindungen zu Ihrer RDS- oder Aurora-My-SQL-Instance herstellen kann. Weitere Informationen zu VPC-Networking finden Sie in folgenden Quellen.

   - Was ist VPC Peering? im Amazon VPC Peering Guide
   - Arbeiten mit einer DB-Instance in einer VPC im Amazon-RDS-Benutzerhandbuch

   > **Note**
   >
   > Wenn sich Ihr Amazon-Redshift-Cluster in einer anderen VPC befindet als Ihre RDS- oder Aurora-MySQL-Instance, aktivieren Sie das erweiterte VPC-Routing. Andernfalls kann es beim Ausführen einer Verbundabfrage zu Timeout-Fehlern kommen.

2. Richten Sie Geheimnisse AWS Secrets Manager für Ihre RDS-MySQL- und Aurora MySQL-Datenbanken ein. Verweisen Sie dann auf die Geheimnisse in AWS Identity and Access Management (IAM-) Zugriffsrichtlinien und Rollen. Weitere Informationen finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.

   > **Note**
   >
   > Wenn Ihr Cluster das erweiterte VPC-Routing verwendet, müssen Sie möglicherweise einen Schnittstellen-VPC-Endpunkt für konfigurieren AWS Secrets Manager. Dies ist

erforderlich, wenn die VPC und das Subnetz Ihres Amazon Redshift Redshift-Clusters keinen Zugriff auf den öffentlichen Endpunkt haben. AWS Secrets Manager Wenn Sie einen VPC-Schnittstellenendpunkt verwenden, wird die Kommunikation zwischen dem Amazon-Redshift-Cluster in der VPC und AWS Secrets Manager von der VPC privat an die Endpunktschnittstelle weitergeleitet. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Amazon VPC Benutzerhandbuch.

3. Wenden Sie die zuvor erstellte IAM-Rolle auf den Amazon-Redshift-Cluster an. Weitere Informationen finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.
4. Verbinden Sie sich mit Ihren RDS-MySQL- und Auora-MySQL-Datenbanken mit einem externen Schema. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA. Beispiele für die Verwendung von Verbundabfragen finden Sie unter Beispiel für die Verwendung einer Verbundabfrage mit MySQL.
5. Führen Sie die SQL-Abfragen aus, die auf das externe Schema verweisen, das auf Ihre RDS-MySQL- und Aurora-MySQL-Datenbanken verweist.

Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen

Die folgenden Schritte zeigen, wie Sie ein Secret und eine IAM-Rolle erstellen, die mit einer Verbundabfrage verwendet werden sollen.

Voraussetzungen

Sie müssen die folgenden Voraussetzungen erfüllen, um ein Secret und eine IAM-Rolle für die Verwendung mit Verbundabfragen zu erstellen:

- Eine RDS-PostgreSQL-, Aurora-PostgreSQL-DB-Instance, RDS-MySQL- oder Aurora MySQL-DB-Instance mit Benutzernamen- und Passwortauthentifizierung.
- Ein Amazon-Redshift-Cluster mit einer Cluster-Wartungsversion, die Verbundabfragen unterstützt.

Um ein Geheimnis (Benutzername und Passwort) zu erstellen mit AWS Secrets Manager

1. Melden Sie sich bei der Secrets Manager Manager-Konsole mit dem Konto an, dem Ihre RDS- oder Aurora-DB-Cluster-Instance gehört.

2. Wählen Sie Store a new secret (Ein neues Secret speichern).
3. Wählen Sie die Kachel Credentials for RDS database (Anmeldeinformationen für die RDS-Datenbank) aus. Geben Sie unter User name (Benutzername) und Password (Passwort) Entsprechendes für Ihre Instance ein. Bestätigen oder wählen Sie einen Wert für den Verschlüsselungsschlüssel aus. Wählen Sie dann die RDS-Datenbank aus, auf die Ihr Secret zugreifen soll.

   > **Note**
   >
   > Wir raten zur Verwendung des Standard-Verschlüsselungsschlüssels (`DefaultEncryptionKey`). Wenn Sie einen benutzerdefinierten Verschlüsselungsschlüssel verwenden, muss die IAM-Rolle, die für den Zugriff auf den geheimen Schlüssel verwendet wird, als Schlüsselbenutzer hinzugefügt werden.

4. Geben Sie einen Namen für das Secret ein, fahren Sie unter Verwendung der Standardoptionen mit den Erstellungsschritten fort ,und klicken Sie auf Store (Speichern).
5. Lassen Sie sich Ihr Secret anzeigen und notieren Sie sich den Secret-ARN-Wert den Sie erstellt haben, um das Secret zu identifizieren.

So erstellen Sie mit dem Secret eine Sicherheitsrichtlinie

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Erstellen Sie eine Richtlinie mit JSON ähnlich der folgenden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
```

```
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Zum Abrufen des Secrets benötigen Sie Listen- und Lese- Aktionen. Es wird empfohlen, die Ressource auf das bestimmte Secret zu beschränken, das Sie erstellt haben. Verwenden Sie dazu den Amazon-Ressourcennamen (ARN) des Secrets, um die Ressource zu beschränken. Sie können die Berechtigungen und Ressourcen auch mithilfe des visuellen Editors auf der IAM-Konsole angeben.

3. Geben Sie der Richtlinie einen Namen und beenden Sie die Erstellung.
4. Navigieren Sie zu IAM roles (IAM-Rollen).
5. Erstellen Sie eine IAM-Rolle für Redshift – Customizable (Redshift – Anpassbar).
6. Fügen Sie entweder die soeben erstellte IAM-Richtlinie an eine vorhandene IAM-Rolle an oder erstellen Sie eine neue IAM-Rolle und fügen Sie die Richtlinie an.
7. Bestätigen Sie auf der Registerkarte Trust relationships (Vertrauensstellungen) Ihrer IAM-Rolle, dass sie die Vertrauensentität `redshift.amazonaws.com` enthält.
8. Notieren Sie den Rollen-ARN den Sie erstellt haben. Dieser ARN hat Zugriff auf das Secret.

So fügen Sie die IAM-Rolle an Ihren Amazon-Redshift-Cluster an

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) aus. Die Cluster für Ihr Konto in der aktuellen AWS Region werden aufgelistet.
3. Wählen Sie den Cluster-Namen in der Liste aus, um weitere Details zu einem Cluster anzuzeigen.
4. Wählen Sie für Actions (Aktionen) Manage IAM roles (IAM-Rollen verwalten) aus. Es wird die Seite Manage IAM roles (IAM-Rollen verwalten) angezeigt.

5. Fügen Sie dem Cluster Ihre IAM-Rolle hinzu.

Beispiele für die Verwendung einer Verbundabfrage

In den folgenden Beispielen wird gezeigt, wie Sie eine Verbundabfrage ausführen. Führen Sie SQL mit Ihrem SQL-Client aus, der mit der Amazon-Redshift-Datenbank verbunden ist.

Beispiel für die Verwendung einer Verbundabfrage mit PostgreSQL

Das folgende Beispiel zeigt, wie Sie eine Verbundabfrage einrichten, die auf eine Amazon-Redshift-Datenbank, eine Aurora-PostgreSQL-Datenbank und auf Amazon S3 verweist. In diesem Beispiel wird gezeigt, wie Verbundabfragen funktionieren. Um sie in Ihrer eigenen Umgebung auszuführen, ändern Sie diese so, dass sie Ihrer Umgebung entsprechen. Die Voraussetzungen hierfür finden Sie unter Erste Schritte mit der Verwendung von Verbundabfragen an PostgreSQL.

Erstellen Sie ein externes Schema, das auf eine Aurora-PostgreSQL-Datenbank verweist.

```
CREATE EXTERNAL SCHEMA apg
FROM POSTGRES
DATABASE 'database-1' SCHEMA 'myschema'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Erstellen Sie ein weiteres externes Schema, das auf Amazon S3 verweist, was Amazon Redshift Spectrum verwendet. Erteilen Sie außerdem die Berechtigung, das Schema zu verwenden `public`.

```
CREATE EXTERNAL SCHEMA s3
FROM DATA CATALOG
DATABASE 'default' REGION 'us-west-2'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-S3';

GRANT USAGE ON SCHEMA s3 TO public;
```

Zeigen Sie die Zeilenanzahl in der Amazon-Redshift-Tabelle an.

```
SELECT count(*) FROM public.lineitem;

  count
```

```
----------
25075099
```

Zeigen Sie die Zeilenanzahl in der Aurora-PostgreSQL-Tabelle an.

```
SELECT count(*) FROM apg.lineitem;

count
-------
11760
```

Zeigen Sie die Zeilenanzahl in Amazon S3 an.

```
SELECT count(*) FROM s3.lineitem_1t_part;

   count
------------
6144008876
```

Erstellen Sie eine Ansicht der Tabellen aus Amazon Redshift, Aurora PostgreSQL und Amazon S3. Diese Ansicht wird verwendet, um Ihre Verbundabfragen auszuführen.

```
CREATE VIEW lineitem_all AS
  SELECT
 l_orderkey,l_partkey,l_suppkey,l_linenumber,l_quantity,l_extendedprice,l_discount,l_tax,l_retu
        l_shipdate::date,l_commitdate::date,l_receiptdate::date,
 l_shipinstruct ,l_shipmode,l_comment
  FROM s3.lineitem_1t_part
  UNION ALL SELECT * FROM public.lineitem
  UNION ALL SELECT * FROM apg.lineitem
     with no schema binding;
```

Zeigen Sie zur Begrenzung der Ergebnisse die Anzahl der Zeilen in der Ansicht `lineitem_all` mit einem Prädikat an.

```
SELECT count(*) from lineitem_all WHERE l_quantity = 10;

   count
-----------
123373836
```

Finden Sie heraus, wie viele Verkäufe eines Artikels es im Januar eines jeden Jahres gab.

```
SELECT extract(year from l_shipdate) as year,
       extract(month from l_shipdate) as month,
       count(*) as orders
FROM lineitem_all
WHERE extract(month from l_shipdate) = 1
AND l_quantity < 2
GROUP BY 1,2
ORDER BY 1,2;

 year | month | orders
------+-------+---------
 1992 |     1 |  196019
 1993 |     1 | 1582034
 1994 |     1 | 1583181
 1995 |     1 | 1583919
 1996 |     1 | 1583622
 1997 |     1 | 1586541
 1998 |     1 | 1583198
 2016 |     1 |   15542
 2017 |     1 |   15414
 2018 |     1 |   15527
 2019 |     1 |     151
```

Beispiel für die Verwendung eines Namens mit sowohl Groß- als auch Kleinbuchstaben

Zur Abfrage einer unterstützten PostgreSQL-Remote-Datenbank, die einen Datenbank-. Schema-, Tabellen- oder Spaltennamen mit sowohl Groß- auch Kleinbuchstaben hat, legen Sie `enable_case_sensitive_identifier` auf `true` fest. Weitere Informationen zu diesem Sitzungsparameter finden Sie unter enable_case_sensitive_identifier.

```
SET enable_case_sensitive_identifier TO TRUE;
```

In der Regel werden die Datenbank- und Schemanamen in Kleinbuchstaben angegeben. Das folgende Beispiel zeigt, wie Sie eine Verbindung zu einer unterstützten Remote-PostgreSQL-Datenbank herstellen können, die Datenbank- und Schemanamen mit Kleinbuchstaben sowie Tabellen- und Spaltennamen mit sowohl Groß- als auch Kleinbuchstaben enthält.

Erstellen Sie ein externes Schema, das auf eine Aurora-PostgreSQL-Datenbank verweist, die einen Datenbanknamen mit Kleinbuchstaben (`dblower`) und einen Schemanamen mit Kleinbuchstaben (`schemalower`) hat.

```
CREATE EXTERNAL SCHEMA apg_lower
FROM POSTGRES
DATABASE 'dblower' SCHEMA 'schemalower'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Legen Sie in der Sitzung, in der die Abfrage ausgeführt wird, `enable_case_sensitive_identifier` auf `true` fest.

```
SET enable_case_sensitive_identifier TO TRUE;
```

Führen Sie eine Verbundabfrage aus, um alle Daten aus der PostgreSQL-Datenbank auszuwählen. Die Tabellen- (`MixedCaseTab`) und Spaltennamen (`MixedCaseName`) enthalten sowohl Groß- als auch Kleinbuchstaben. Der Ergebnis ist eine Zeile (`Harry`).

```
select * from apg_lower."MixedCaseTab";
```

```
 MixedCaseName
-------
 Harry
```

Das folgende Beispiel zeigt, wie Sie eine Verbindung zu einer unterstützten Remote-PostgreSQL-Datenbank herstellen können, die einen Datenbank-, Schema-, Tabellen- und Spaltennamen mit sowohl Groß- als auch Kleinbuchstaben enthält.

Legen Sie `enable_case_sensitive_identifier` auf `true` fest, bevor Sie das externe Schema erstellen. Wenn Sie `enable_case_sensitive_identifier` nicht auf `true` festlegen, bevor Sie das externe Schema erstellen, erhalten Sie einen Fehler, dass die Datenbank nicht existiert.

Erstellen Sie ein externes Schema, das auf eine Aurora-PostgreSQL-Datenbank verweist, die einen Datenbanknamen (`UpperDB`) und einen Schemanamen (`UpperSchema`) mit sowohl Groß- als auch Kleinbuchstaben hat.

```
CREATE EXTERNAL SCHEMA apg_upper
FROM POSTGRES
DATABASE 'UpperDB' SCHEMA 'UpperSchema'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Führen Sie eine Verbundabfrage aus, um alle Daten aus der PostgreSQL-Datenbank auszuwählen. Die Tabellen- (`MixedCaseTab`) und Spaltennamen (`MixedCaseName`) enthalten sowohl Groß- als auch Kleinbuchstaben. Der Ergebnis ist eine Zeile (`Harry`).

```
select * from apg_upper."MixedCaseTab";
```

```
 MixedCaseName
-------
 Harry
```

Beispiel für die Verwendung einer Verbundabfrage mit MySQL

Das folgende Beispiel zeigt, wie Sie eine Verbundabfrage einrichten, die auf eine Aurora-MySQL-Datenbank verweist. In diesem Beispiel wird gezeigt, wie Verbundabfragen funktionieren. Um sie in Ihrer eigenen Umgebung auszuführen, ändern Sie diese so, dass sie Ihrer Umgebung entsprechen. Die Voraussetzungen hierfür finden Sie unter Erste Schritte bei der Verwendung von Verbundabfragen für MySQL.

Dieses Beispiel hängt von den folgenden Voraussetzungen ab:

- Ein Secret, das im Secrets Manager für die Aurora-MySQL-Datenbank eingerichtet wurde. Auf dieses Secret wird in den IAM-Zugriffsrichtlinien und -rollen verwiesen. Weitere Informationen finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.
- Eine Sicherheitsgruppe, die eingerichtet wurde, um Amazon Redshift und Aurora MySQL zu verknüpfen.

Erstellen Sie ein externes Schema, das auf eine Aurora-MySQL-Datenbank verweist.

```
CREATE EXTERNAL SCHEMA amysql
FROM MYSQL
DATABASE 'functional'
URI 'endpoint to remote hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Führen Sie eine Beispiel-SQL-Auswahl für die Aurora-MySQL-Tabelle aus, um eine Zeile aus der Mitarbeitertabelle in Aurora MySQL aufzurufen.

```
SELECT level FROM amysql.employees LIMIT 1;

 level
-------
     8
```

Datentypunterschiede zwischen Amazon Redshift und unterstützten PostgreSQL- oder MySQL-Datenbanken

Die folgende Tabelle zeigt die Zuordnung eines Amazon-Redshift-Datentyps zu einem entsprechenden Amazon-RDS-PostgreSQL- oder Aurora-PostgreSQL-Datentyp.

Amazon-Redshift-Datentyp	RDS-PostgreSQL- oder Aurora-PostgreSQL-Datentyp	Beschreibung
SMALLINT	SMALLINT	2-Byte-Ganzzahl mit Vorzeichen
INTEGER	INTEGER	4-Byte-Ganzzahl mit Vorzeichen
BIGINT	BIGINT	8-Byte-Ganzzahl mit Vorzeichen
DECIMAL	DECIMAL	Genauer Zahlenwert mit wählbarer Genauigkeit

Amazon-Redshift-Datentyp	RDS-PostgreSQL- oder Aurora-PostgreSQL-Datentyp	Beschreibung
REAL	REAL	Gleitkommazahl mit einfacher Genauigkeit
DOUBLE PRECISION	DOUBLE PRECISION	Gleitkommazahl mit doppelter Genauigkeit
BOOLEAN	BOOLEAN	Logischer/Boolescher Wert (wahr/falsch)
CHAR	CHAR	Zeichenfolge mit fester Länge
VARCHAR	VARCHAR	Zeichenfolge mit variabler Länge und benutzerdefiniertem Grenzwert
DATUM	DATUM	Kalenderdatum (Jahr, Monat, Tag)
TIMESTAMP	TIMESTAMP	Datum und Uhrzeit (ohne Zeitzone)
TIMESTAMPTZ	TIMESTAMPTZ	Datum und Uhrzeit (mit Zeitzone)
GEOMETRY	PostGIS GEOMETRY	Geodaten

Die folgenden RDS-PostgreSQL- und Aurora-PostgreSQL-Datentypen werden in Amazon Redshift zu VARCHAR(64K) umgewandelt:

- JSON, JSONB
- Arrays
- BIT, BIT VARYING
- BYTEA

- Zusammengesetzte Typen
- Datums- und Zeittypen INTERVAL, TIME, TIME WITH TIMEZONE
- Typen mit Aufzählungswerten
- Monetäre Typen
- Netzwerkadress-Typen
- Numerische Typen SERIAL, BIGSERIAL, SMALLSERIAL und MONEY
- Objekt-ID-Typen
- pg_lsn-Typ
- Pseudotypen
- Intervall-Typen
- Textsuche-Typen
- TXID_SNAPSHOT
- UUID
- XML-Typ

Die folgende Tabelle zeigt die Zuordnung eines Amazon-Redshift-Datentyps zu einem entsprechenden Amazon-RDS-MySQL- oder Aurora-MySQL-Datentyp.

Amazon-Redshift-Datentyp	RDS-MySQL- oder Aurora-MySQL-Datentyp	Beschreibung
BOOLEAN	TINYINT(1)	Logischer/Boolescher Wert (true oder false)
SMALLINT	TINYINT(UNSIGNED)	2-Byte-Ganzzahl mit Vorzeichen
SMALLINT	SMALLINT	2-Byte-Ganzzahl mit Vorzeichen
INTEGER	SMALLINT UNSIGNED	4-Byte-Ganzzahl mit Vorzeichen
INTEGER	MEDIUMINT (UNSIGNED)	4-Byte-Ganzzahl mit Vorzeichen

Amazon-Redshift-Datentyp	RDS-MySQL- oder Aurora-MySQL-Datentyp	Beschreibung
INTEGER	INT	4-Byte-Ganzzahl mit Vorzeichen
BIGINT	INT UNSIGNED	8-Byte-Ganzzahl mit Vorzeichen
BIGINT	BIGINT	8-Byte-Ganzzahl mit Vorzeichen
DECIMAL	BIGINT UNSIGNED	Genauer Zahlenwert mit wählbarer Genauigkeit
DECIMAL	DECIMAL(M,D)	Genauer Zahlenwert mit wählbarer Genauigkeit
REAL	FLOAT	Gleitkommazahl mit einfacher Genauigkeit
DOUBLE PRECISION	DOUBLE	Gleitkommazahl mit doppelter Genauigkeit
CHAR	CHAR	Zeichenfolge mit fester Länge
VARCHAR	VARCHAR	Zeichenfolge mit variabler Länge und benutzerdefiniertem Grenzwert
DATUM	DATUM	Kalenderdatum (Jahr, Monat, Tag)
TIME	TIME	Uhrzeit (ohne Zeitzone)

Amazon-Redshift-Datentyp	RDS-MySQL- oder Aurora-MySQL-Datentyp	Beschreibung
TIMESTAMP	TIMESTAMP	Datum und Uhrzeit (ohne Zeitzone)
TIMESTAMP	DATETIME	Uhrzeit (ohne Zeitzone)
VARCHAR(4)	YEAR	Zeichen mit variabler Länge, die das Jahr darstellen

Ein Fehler tritt auf, wenn TIME-Daten außerhalb des Bereichs liegen (00:00:00 –24:00:00).

Die folgenden RDS-MySQL- und Aurora-MySQL-Datentypen werden in Amazon Redshift in VARCHAR(64K) umgewandelt:

- BIT
- BINARY
- VARBINARY
- TINYBLOB, BLOB, MEDIUMBLOB, LONGBLOB
- TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT
- ENUM
- SET
- SPATIAL

Einschränkungen beim Zugriff auf Verbunddaten mit Amazon Redshift

Einige Amazon-Redshift-Funktionen unterstützen den Zugriff auf Verbunddaten nicht. Im Folgenden finden Sie zugehörige Einschränkungen und Überlegungen.

Bei der Verwendung von Verbundabfragen in Amazon Redshift gelten folgende Einschränkungen und Überlegungen:

- Verbundabfragen unterstützen den Lesezugriff auf externe Datenquellen. In der externen Datenquelle können Sie keine Datenbankobjekte schreiben oder erstellen.
- In einigen Fällen greifen Sie möglicherweise auf eine Amazon RDS- oder Aurora DB-Cluster-Datenbank in einer anderen AWS Region als Amazon Redshift zu. In diesen Fällen fallen in der Regel Netzwerklatenz und Abrechnungsgebühren für die Übertragung von Daten zwischen Regionen AWS an. Wir empfehlen die Verwendung einer globalen Aurora-Datenbank mit einem lokalen Endpunkt in derselben AWS Region wie Ihr Amazon Redshift Redshift-Cluster. Die globalen Datenbanken von Aurora nutzen eine dedizierte Infrastruktur für die speicherbasierte Replikation in zwei beliebigen AWS -Regionen mit einer typischen Latenzzeit von unter 1 Sekunde.
- Bedenken Sie die Kosten für den Zugriff auf Amazon RDS oder den Aurora-DB-Cluster. Wenn Sie diese Funktion beispielsweise für den Zugriff auf den Aurora-DB-Cluster verwenden, basieren die Gebühren für den Aurora-DB-Cluster auf IOPS.
- Verbundabfragen ermöglichen keinen Zugriff auf Amazon Redshift von einem RDS- oder Aurora-DB-Cluster aus.
- Verbundabfragen sind nur in AWS Regionen verfügbar, in denen sowohl Amazon Redshift als auch Amazon RDS oder Aurora DB-Cluster verfügbar sind.
- Verbundabfragen unterstützen derzeit nicht `ALTER SCHEMA`. Um ein Schema zu ändern, verwenden Sie `DROP` und dann `CREATE EXTERNAL SCHEMA`.
- Verbundabfragen funktionieren nicht mit der Parallelitätsskalierung.
- Verbundabfragen unterstützen derzeit keinen Zugriff über einen PostgreSQL-Fremddaten-Wrapper.
- Verbundabfragen an RDS MySQL oder Aurora MySQL unterstützen die Transaktionsisolierung auf der Ebene READ COMMITTED.
- Sofern nicht anders angegeben, stellt Amazon Redshift eine Verbindung mit RDS for MySQL oder Aurora MySQL auf Port 3306 her. Bestätigen Sie die MySQL-Portnummer, bevor Sie ein externes Schema für MySQL erstellen.
- Sofern nicht anders angegeben, stellt Amazon Redshift eine Verbindung mit RDS PostgreSQL oder Aurora PostgreSQL auf Port 5432 her. Bestätigen Sie die PostgreSQL-Portnummer, bevor Sie ein externes Schema für PostgreSQL erstellen.
- Beim Abrufen der Datentypen TIMESTAMP und DATE aus MySQL werden Nullwerte als NULL behandelt.
- Wenn ein Aurora-DB-Cluster-Datenbank-Reader-Endpunkt verwendet wird, kann der Fehler „Ungültiger Snapshot“ auftreten. Dies lässt sich über eine der folgenden Methoden vermeiden:

- Verwenden Sie einen bestimmten Aurora-DB-Cluster-Instance-Endpunkt (anstatt den Aurora-DB-Cluster-Cluster-Endpunkt zu verwenden). Bei dieser Methode wird die Transaktionsisolation REPEATABLE READ für die Ergebnisse aus der PostgreSQL-Datenbank verwendet.
- Verwenden Sie einen Aurora-DB-Cluster-Reader-Endpunkt und `pg_federation_repeatable_read` setzen Sie ihn für die Sitzung auf false. Bei dieser Methode wird die Transaktionsisolation READ COMMITTED für die Ergebnisse aus der PostgreSQL-Datenbank verwendet. Weitere Informationen zu Aurora-DB-Cluster-Reader-Endpunkten finden Sie unter Typen von Aurora-DB-Cluster-Endpunkten im Amazon Aurora Aurora-Benutzerhandbuch. Weitere Informationen zu `pg_federation_repeatable_read` finden Sie unter pg_federation_repeatable_read.

Für Transaktionen beim Arbeiten mit Verbundabfragen an PostgreSQL-Datenbanken gelten folgende Überlegungen:

- Wenn eine Abfrage aus Verbundtabellen besteht, startet der Leader-Knoten eine READ ONLY REPEATABLE READ-Transaktion in der fernen Datenbank. Diese Transaktion bleibt für die Dauer der Amazon-Redshift-Transaktion bestehen.
- Der Leader-Knoten erstellt einen Snapshot der fernen Datenbank durch Aufruf von `pg_export_snapshot` und setzt eine Lesesperre auf die betroffenen Tabellen.
- Ein Datenverarbeitungsknoten startet eine Transaktion und verwendet den Snapshot, der am Leader-Knoten erstellt wurde, um Abfragen an die ferne Datenbank auszugeben.

Unterstützte Versionen von Verbunddatenbanken

Ein externes Amazon-Redshift-Schema kann auf eine Datenbank in einem externen RDS PostgreSQL oder Aurora PostgreSQL verweisen. Dabei gelten die folgenden Einschränkungen:

- Wenn Sie ein externes Schema erstellen, das auf den Aurora-DB-Cluster verweist, muss die Aurora PostgreSQL-Datenbank Version 9.6 oder höher haben.
- Wenn Sie ein externes Schema mit Verweis auf Amazon RDS erstellen, muss die Amazon-RDS-PostgreSQL-Datenbank in der Version 9.6 oder später vorliegen.

Ein externes Amazon-Redshift-Schema kann auf eine Datenbank in einem externen RDS MySQL oder Aurora MySQL verweisen. Dabei gelten die folgenden Einschränkungen:

- Wenn Sie ein externes Schema erstellen, das auf den Aurora-DB-Cluster verweist, muss die Aurora MySQL-Datenbank Version 5.6 oder höher haben.
- Wenn Sie ein externes Schema mit Verweis auf Amazon RDS erstellen, muss die RDS-MySQL-Datenbank in der Version 5.6 oder später vorliegen.

Abfrage externer Daten mit Amazon Redshift Spectrum

Mit Amazon Redshift Spectrum können Sie effektiv strukturierte und halbstrukturierte Daten aus Dateien in Amazon S3 abfragen und abrufen, ohne die Daten in Amazon-Redshift-Tabellen laden zu müssen. Redshift-Spectrum-Abfragen nutzen massive Parallelität zur sehr schnellen Ausführung bei sehr großen Datensätzen. Ein großer Teil der Verarbeitung findet auf der Redshift-Spectrum-Ebene statt, und die meisten Daten bleiben in Amazon S3. Mehrere Cluster können denselben Datensatz in Amazon S3 gleichzeitig abfragen, ohne dass Kopien der Daten für jeden Cluster erstellt werden müssen.

Themen

Übersicht zu Amazon Redshift Spectrum

Amazon Redshift Spectrum befindet sich auf dedizierten Amazon-Redshift-Servern, die von Ihrem Cluster unabhängig sind. Amazon Redshift verschiebt viele datenverarbeitungsintensive Aufgaben, wie etwa die Prädikatfilterung und -aggregierung, auf die Redshift-Spectrum-Ebene. So verwenden

Redshift Spectrum-Abfragen viel weniger der Verarbeitungskapazität Ihres Clusters als andere Abfragen. Dazu kann Redshift Spectrum in intelligenter Weise skaliert werden. Auf der Grundlage der Anforderungen Ihrer Abfragen kann Redshift Spectrum potenziell Tausende von Instances nutzen und so sehr umfangreiche parallele Verarbeitungsmöglichkeiten bieten.

Sie erstellen Redshift Spectrum-Tabellen, indem Sie die Struktur für Ihre Dateien definieren und diese als Tabellen in einem externen Datenkatalog registrieren. Der externe Datenkatalog kann AWS Glue der im Lieferumfang von Amazon Athena enthaltene Datenkatalog oder Ihr eigener Apache Hive-Metastore sein. Sie können externe Tabellen von Amazon Redshift aus erstellen und verwalten, DDL-Befehle (Data Definition Language) nutzen oder jedes andere Tool verwenden, das sich mit dem externen Datenkatalog verbinden kann. Änderungen an dem externen Datenkatalog sind sofort für jeden Ihrer Amazon-Redshift-Cluster verfügbar.

Sie können optional die externen Tabellen auf einer oder mehreren Spalten partitionieren. Die Definition von Partitionen als Teil der externen Tabelle kann die Leistung verbessern. Diese Verbesserung beruht darauf, dass der Amazon-Redshift-Abfrageoptimierer Partitionen, die keine Daten für die Abfrage enthalten, entfernt.

Nachdem Ihre Redshift-Spectrum-Tabellen definiert wurden, können Sie die Tabellen wie jede andere Amazon-Redshift-Tabelle abfragen oder verbinden. Redshift Spectrum unterstützt keine Aktualisierungsvorgänge auf externen Tabellen. Sie können Redshift Spectrum-Tabellen zu mehreren Amazon Redshift Redshift-Clustern hinzufügen und dieselben Daten auf Amazon S3 von jedem Cluster in derselben AWS Region abfragen. Wenn Sie Amazon-S3-Datendateien aktualisieren, stehen diese Daten sofort zur Abfrage von allen Ihren Amazon-Redshift-Clustern aus zur Verfügung.

Der AWS Glue Datenkatalog, auf den Sie zugreifen, ist möglicherweise verschlüsselt, um die Sicherheit zu erhöhen. Wenn der AWS Glue Katalog verschlüsselt ist, benötigen Sie den Schlüssel AWS Key Management Service (AWS KMS) für AWS Glue , um auf den AWS Glue Katalog zuzugreifen. AWS Glue Die Katalogverschlüsselung ist nicht in allen AWS Regionen verfügbar. Eine Liste der unterstützten AWS Regionen finden Sie unter Verschlüsselung und sicheren Zugriff für AWS Glue im AWS Glue Entwicklerhandbuch.Weitere Informationen zur AWS Glue Datenkatalogverschlüsselung finden Sie unter Verschlüsseln Ihres AWS Glue Datenkatalogs im AWS Glue Entwicklerhandbuch.

> **Note**
>
> Sie können die Details für Redshift-Spectrum-Tabellen nicht mit den gleichen Ressourcen anzeigen, die Sie für Amazon-Redshift-Standardtabellen verwenden, wie PG_TABLE_DEF, STV_TBL_PERM, PG_CLASS oder information_schema. Wenn Ihr Business Intelligence-

oder Analyse-Tool externe Redshift Spectrum-Tabellen nicht erkennt, konfigurieren Sie Ihre Anwendung für die Ausführung von Abfragen für SVV_EXTERNAL_TABLES und SVV_EXTERNAL_COLUMNS.

Amazon-Redshift-Spectrum-Regionen

Redshift Spectrum ist dort verfügbar AWS-Regionen , wo Amazon Redshift verfügbar ist, sofern in der regionsspezifischen Dokumentation nichts anderes angegeben ist. Informationen zur AWS-Region Verfügbarkeit in kommerziellen Regionen finden Sie unter Service-Endpunkte für die Redshift-API in der. Allgemeine Amazon Web Services-Referenz

Überlegungen zu Amazon Redshift Spectrum

Beachten Sie die folgenden Überlegungen bei der Verwendung von Amazon Redshift Spectrum:

- Der Amazon Redshift Redshift-Cluster und der Amazon S3 S3-Bucket müssen sich in derselben AWS Region befinden.
- Redshift Spectrum bietet keine Unterstützung für Enhanced VPC Routing mit bereitgestellten Clustern. Möglicherweise müssen Sie weitere Konfigurationsschritte ausführen, um auf Ihre Amazon-S3-Daten zuzugreifen. Weitere Informationen finden Sie unter Redshift Spectrum und Enhanced VPC Routing im Amazon-Redshift-Verwaltungshandbuch.
- Redshift Spectrum unterstützt Amazon-S3-Zugriffspunkt-Aliase. Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren Zugriffspunkt im Amazon-Simple-Storage-Service-Benutzerhandbuch. Redshift Spectrum unterstützt jedoch keine VPC mit Amazon-S3-Zugriffspunkt-Aliase. Weitere Informationen finden Sie unter Redshift Spectrum und Enhanced VPC Routing im Amazon-Redshift-Verwaltungshandbuch.
- Sie können keine Aktualisierungs- oder Löschoperationen für externe Tabellen ausführen. Um eine neue externe Tabelle im angegebenen Schema zu erstellen, können Sie CREATE EXTERNAL TABLE verwenden. Weitere Hinweise zu CREATE EXTERNAL TABLE finden Sie unter CREATE EXTERNAL TABLE. Um die Ergebnisse einer SELECT-Abfrage in vorhandene externe Tabellen in externen Katalogen einzufügen, können Sie INSERT (externe Tabelle) verwenden. Weitere Informationen zu INSERT (externe Tabelle) finden Sie unter INSERT (externe Tabelle).
- Sofern Sie keine verwenden AWS Glue Data Catalog , die für AWS Lake Formation aktiviert ist, können Sie Benutzerberechtigungen für eine externe Tabelle nicht steuern. Sie gewähren oder widerrufen stattdessen die Berechtigungen für das externe Schema. Weitere Informationen zur

Arbeit mit AWS Lake Formation finden Sie unterVerwenden von Redshift Spectrum mit AWS Lake Formation.

- Um Redshift Spectrum-Abfragen auszuführen, benötigt der Datenbankbenutzer die Berechtigung, temporäre Tabellen in der Datenbank zu erstellen. Das folgende Beispiel erteilt der Benutzergruppe `spectrumdb` temporäre Berechtigungen für die Datenbank `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

  Weitere Informationen finden Sie unter GRANT.
- Wenn Sie den Athena-Datenkatalog oder den AWS Glue Datenkatalog als Metadatenspeicher verwenden, finden Sie weitere Informationen unter Kontingente und Grenzwerte im Amazon Redshift Management Guide.
- Redshift Spectrum unterstützt Amazon EMR nicht mit Kerberos.

Erste Schritte mit Amazon Redshift Spectrum

In diesem Tutorial erfahren Sie, wie Sie mit Amazon Redshift Spectrum Daten direkt aus Dateien auf Amazon S3 abfragen. Wenn Sie bereits einen Cluster und einen SQL-Client haben, können Sie dieses Tutorial mit minimaler Einrichtung abschließen.

> **Note**
>
> Redshift Spectrum-Abfragen sind mit zusätzlichen Gebühren verbunden. Die Kosten für die Beispielabfragen in diesem Tutorial sind äußerst gering. Weitere Informationen zu Preisen finden Sie unter Amazon Redshift Spectrum – Preise.

Voraussetzungen

Zu Verwendung von Redshift Spectrum benötigen Sie einen Amazon-Redshift-Cluster und einen SQL-Client, der mit Ihrem Cluster verbunden ist, damit Sie SQL-Befehle ausführen können. Der Cluster und die Datendateien in Amazon S3 müssen sich in derselben AWS-Region befinden.

Informationen zum Erstellen eines Amazon Redshift-Clusters finden Sie unter Bereitgestellte Amazon Redshift Redshift-Cluster im Amazon Redshift Getting Started Guide. Informationen zu Verbindungsmöglichkeiten mit einem Cluster finden Sie unter Verbindung zu Amazon Redshift Data Warehouses im Amazon Redshift Getting Started Guide.

In einigen der folgenden Beispiele befinden sich die Beispieldaten in der Region USA Ost (Nord-Virginia) (`us-east-1`), sodass Sie einen Cluster benötigen, der sich auch in `us-east-1` befindet. Oder Sie können Amazon S3 verwenden, um Datenobjekte aus den folgenden Buckets und Ordnern in Ihren Bucket zu kopieren, in AWS-Region dem sich Ihr Cluster befindet:

- `s3://redshift-downloads/tickit/spectrum/customers/*`
- `s3://redshift-downloads/tickit/spectrum/sales_partition/*`
- `s3://redshift-downloads/tickit/spectrum/sales/*`
- `s3://redshift-downloads/tickit/spectrum/salesevent/*`

Führen Sie einen Amazon-S3-Befehl ähnlich dem folgenden aus, um Beispieldaten, die sich in der Region USA Ost (Nord-Virginia) befinden, in Ihre AWS-Region zu kopieren. Erstellen Sie vor dem Ausführen des Befehls Ihren Bucket und die Ordner in Ihrem Bucket so, dass sie Ihrem Amazon-S3-Befehl entsprechen. Die Ausgabe des Amazon-S3-Kopierbefehls bestätigt, dass die Dateien in den *`bucket-name`* Ihrer gewünschten AWS-Region kopiert wurden.

```
aws s3 cp s3://redshift-downloads/tickit/spectrum/ s3://bucket-name/tickit/spectrum/ --
copy-props none --recursive
```

Erste Schritte mit Redshift Spectrum unter AWS CloudFormation

Als Alternative zu den folgenden Schritten können Sie auf die Redshift DataLake AWS CloudFormation Spectrum-Vorlage zugreifen, um einen Stack mit einem Amazon S3 S3-Bucket zu erstellen, den Sie abfragen können. Weitere Informationen finden Sie unter Starten Sie Ihren AWS CloudFormation Stack und fragen Sie dann Ihre Daten in Amazon S3 ab.

Erste Schritte mit Amazon Redshift Spectrum – Schritt für Schritt

Gehen Sie wie folgt vor, um mit der Verwendung von Amazon Redshift Spectrum zu beginnen:

- Schritt 1. Erstellen Sie eine IAM-Rolle für Amazon Redshift
- Schritt 2: Zuordnen der IAM-Rolle zu Ihrem Cluster
- Schritt 3: Erstellen eines externen Schemas und einer externen Tabelle
- Schritt 4: Abfragen Ihrer Daten in Amazon S3

Schritt 1. Erstellen Sie eine IAM-Rolle für Amazon Redshift

Ihr Cluster benötigt eine Autorisierung für den Zugriff auf Ihren externen Datenkatalog in AWS Glue oder Amazon Athena und Ihre Datendateien in Amazon S3. Diese Berechtigung wird durch die Referenzierung einer AWS Identity and Access Management -Rolle (IAM) gewährt, die mit Ihrem Cluster verbunden ist. Weitere Informationen zur Verwendung von Rollen mit Amazon Redshift finden Sie unter Autorisierung von COPY- und UNLOAD-Vorgängen mit IAM-Rollen.

> **Note**
>
> In bestimmten Fällen können Sie Ihren Athena-Datenkatalog zu einem AWS Glue Datenkatalog migrieren. Sie können dies tun, wenn sich Ihr Cluster in einer AWS Region befindet, die unterstützt AWS Glue wird, und Sie externe Redshift Spectrum-Tabellen im Athena-Datenkatalog haben. Um den AWS Glue Datenkatalog mit Redshift Spectrum zu verwenden, müssen Sie möglicherweise Ihre IAM-Richtlinien ändern. Weitere Informationen finden Sie unter Upgrade auf den AWS Glue -Datenkatalog im Athena-Benutzerhandbuch.

Wenn Sie eine Rolle für Amazon Redshift erstellen, wählen Sie einen der folgenden Ansätze aus:

- Wenn Sie Redshift Spectrum entweder mit einem Athena-Datenkatalog oder AWS Glue einem Datenkatalog verwenden, folgen Sie den unter beschriebenen Schritten. So erstellen Sie eine IAM-Rolle für Amazon Redshift
- Wenn Sie Redshift Spectrum mit einem verwenden AWS Glue Data Catalog , für aktiviert ist AWS Lake Formation, gehen Sie wie folgt vor:
  - So erstellen Sie eine IAM-Rolle für Amazon Redshift mit einem aktivierten AWS Glue Data CatalogAWS Lake Formation
  - So gewähren Sie SELECT-Berechtigungen für eine Tabelle, um diese in der Lake-Formation-Datenbank abzufragen

So erstellen Sie eine IAM-Rolle für Amazon Redshift

1. Öffnen Sie die IAM-Konsole.
2. Wählen Sie im Navigationsbereich Roles aus.
3. Wählen Sie Rolle erstellen aus.

4. Wählen Sie AWS -Service als vertrauenswürdige Entität und dann Redshift als Anwendungsfall aus.
5. Wählen Sie unter Anwendungsfall für andere AWS-Services die Option Redshift — Anpassbar und wählen Sie dann Weiter aus.
6. Die Seite Add permissions policy (Berechtigungsrichtlinie hinzufügen) wird angezeigt. Wählen Sie `AmazonS3ReadOnlyAccess` und`AWSGlueConsoleFullAccess`, wenn Sie den AWS Glue Datenkatalog verwenden. Oder wählen Sie `AmazonAthenaFullAccess` aus, wenn Sie den Athena-Datenkatalog verwenden. Wählen Sie Weiter aus.

   > **Note**
   >
   > Über die Richtlinie `AmazonS3ReadOnlyAccess` gewähren Sie dem Cluster Leseberechtigungen für alle Amazon-S3-Buckets. Um nur Zugriff auf den AWS Beispieldaten-Bucket zu gewähren, erstellen Sie eine neue Richtlinie und fügen Sie die folgenden Berechtigungen hinzu.
   >
   > ```
   > {
   >     "Version": "2012-10-17",
   >     "Statement": [
   >         {
   >             "Effect": "Allow",
   >             "Action": [
   >                 "s3:Get*",
   >                 "s3:List*"
   >             ],
   >             "Resource": "arn:aws:s3:::redshift-downloads/*"
   >         }
   >     ]
   > }
   > ```

7. Geben Sie in Role name (Rollenname) einen Namen für die Rolle ein, z. B. **`myspectrum_role`**.
8. Prüfen Sie die Rolleninformationen, und klicken Sie dann auf Create role.
9. Wählen Sie im Navigationsbereich Roles. Wählen Sie den Namen der neuen Rolle aus, um die Zusammenfassung anzuzeigen, und kopieren Sie den Role ARN in die Zwischenablage. Dieser Wert ist der Amazon-Ressourcenname (ARN) für die Rolle, die Sie soeben erstellt haben. Sie verwenden diesen Wert, wenn Sie externe Daten zur Referenzierung Ihrer Datendateien auf Amazon S3 erstellen.

So erstellen Sie eine IAM-Rolle für Amazon Redshift mit einem aktivierten AWS Glue Data CatalogAWS Lake Formation

1. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Policies aus.

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie Create Policy (Richtlinie erstellen) aus.
4. Wählen Sie die Option zum Erstellen der Richtlinie auf der Registerkarte JSON.
5. Fügen Sie das folgende JSON-Richtliniendokument ein, das Zugriff auf den Datenkatalog gewährt, jedoch Administratorberechtigungen für Lake Formation verweigert.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RedshiftPolicyForLF",
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "lakeformation:GetDataAccess"
            ],
            "Resource": "*"
        }
    ]
}
```

6. Wählen Sie, wenn Sie fertig sind, Review (Überprüfen) aus. Die Richtlinienvalidierung meldet mögliche Syntaxfehler.
7. Geben Sie auf der Seite Review Policy (Richtlinie überprüfen) im Feld Name (Name) **`myspectrum_policy`** für die zu erstellende Richtlinie ein. Geben Sie eine Beschreibung ein (Optional) . Überprüfen Sie unter Summary die Richtlinienzusammenfassung, um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden. Wählen Sie dann Create policy aus, um Ihre Eingaben zu speichern.

   Nachdem Sie eine Richtlinie erstellt haben, können Sie Ihren Benutzern Zugriff erteilen.

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in: AWS IAM Identity Center

  Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter Erstellen eines Berechtigungssatzes im AWS IAM Identity Center -Benutzerhandbuch.

- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

  Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter Erstellen einer Rolle für einen externen Identitätsanbieter (Verbund) im IAM-Benutzerhandbuch.

- IAM-Benutzer:
  - Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter Erstellen einer Rolle für einen IAM-Benutzer im IAM-Benutzerhandbuch.
  - (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter Hinzufügen von Berechtigungen zu einem Benutzer (Konsole) im IAM-Benutzerhandbuch.

So gewähren Sie SELECT-Berechtigungen für eine Tabelle, um diese in der Lake-Formation-Datenbank abzufragen

1. Öffnen Sie die Lake-Formation-Konsole unter https://console.aws.amazon.com/lakeformation/.
2. Wählen Sie im Navigationsbereich Data-Lake-Berechtigungen und dann Gewähren.
3. Folgen Sie den Anweisungen unter Gewähren von Tabellenberechtigungen mithilfe der benannten Ressourcenmethode im AWS Lake Formation -Entwicklerhandbuch. Geben Sie die folgenden Informationen ein:

   - Wählen Sie für IAM role (IAM-Rolle) die IAM-Rolle aus, die Sie erstellt haben (`myspectrum_role`). Wenn Sie den Amazon Redshift Query Editor ausführen, verwendet dieser die IAM-Rolle, um die erforderlichen Berechtigungen für die Daten zu erhalten.

   > **Note**
   >
   > Um die SELECT-Berechtigung für die Tabelle in einem abzufragenden Lake-Formation-aktivierten Datenkatalog zu erteilen, gehen Sie wie folgt vor:
   >
   > - Registrieren Sie den Datenpfad in Lake Formation.
   > - Gewähren Sie Benutzern die Berechtigung für diesen Pfad in Lake Formation.
   > - Erstellte Tabellen finden Sie in dem in Lake Formation registrierten Pfad.

4. Wählen Sie Gewähren.

> **Important**
>
> Es hat sich bewährt, über Lake-Formation-Berechtigungen nur Zugriff auf die zugrundeliegenden Amazon-S3-Objekte zu gewähren. Um unbefugten Zugriff zu verhindern, entfernen Sie alle Berechtigungen, die für Amazon-S3-Objekte außerhalb von Lake Formation gewährt wurden. Wenn Sie zuvor auf Amazon-S3-Objekte zugegriffen haben, bevor Sie Lake Formation eingerichtet haben, entfernen Sie alle IAM-Richtlinien oder Bucket-Berechtigungen, die zuvor eingerichtet waren. Weitere Informationen finden Sie unter Upgrading AWS Glue Data Permissions to the AWS Lake Formation Model und Lake Formation Permissions.

Schritt 2: Zuordnen der IAM-Rolle zu Ihrem Cluster

Jetzt besitzen Sie eine IAM-Rolle, die Amazon Redshift zum Zugriff auf den externen Datenkatalog und auf Amazon S3 in Ihrem Namen berechtigt. An dieser Stelle müssen Sie diese Rolle Ihrem Amazon-Redshift-Cluster zuordnen.

So ordnen Sie eine IAM-Rolle einem Cluster zu:

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann den Namen des Clusters aus, den Sie aktualisieren möchten.
3. Wählen Sie für Actions (Aktionen) Manage IAM roles (IAM-Rollen verwalten) aus. Die Seite IAM roles (IAM-Rollen) wird angezeigt.
4. Wählen Sie Enter ARN (ARN eingeben) aus und geben Sie dann einen ARN oder eine IAM-Rolle ein. Oder wählen Sie eine IAM-Rolle aus der Liste aus. Wählen Sie dann Add IAM role (IAM-Rolle hinzufügen) aus, um sie zur Liste der Attached IAM-Rollen (Zugewiesenen IAM-Rollen) hinzuzufügen.
5. Wählen Sie Done (Beenden) aus, um die IAM-Rolle dem Cluster zuzuordnen. Der Cluster wird modifiziert, um die Änderung abzuschließen.

Schritt 3: Erstellen eines externen Schemas und einer externen Tabelle

Erstellen Sie externe Tabellen in einem externen Schema. Das externe Schema verweist auf eine Datenbank im externen Datenkatalog und stellt den IAM-Rollen-ARN bereit, der Ihrem Cluster erlaubt, in Ihrem Namen auf Amazon S3 zuzugreifen. Sie können eine externe Datenbank in einem Amazon Athena Athena-Datenkatalog oder einem Apache Hive-Metastore wie Amazon EMR erstellen. AWS Glue Data Catalog Für dieses Beispiel erstellen Sie die externe Datenbank in einem Amazon-Athena-Datenkatalog, wenn Sie das externe Schema Amazon Redshift erstellen. Weitere Informationen finden Sie unter Erstellen externer Schemata für Amazon Redshift Spectrum.

So erstellen Sie ein externes Schema und eine externe Tabelle:

1. Um ein externes Schema zu erstellen, ersetzen Sie den ARN der IAM-Rolle im folgenden Befehl durch den Rollen-ARN, den Sie in Schritt 1 erstellt haben. Führen Sie dann den Befehl in Ihrem SQL-Client aus.

```
create external schema myspectrum_schema
from data catalog
database 'myspectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
create external database if not exists;
```

2. Führen Sie zur Erstellung einer externen Tabelle den folgenden CREATE EXTERNAL TABLE-Befehl aus.

> **Note**
>
> Ihr Cluster und der Amazon-S3-Bucket müssen sich in der gleichen AWS-Region befinden. Für diesen Beispielbefehl CREATE EXTERNAL TABLE befindet sich der Amazon S3 S3-Bucket mit den Beispieldaten im Osten der USA (Nord-Virginia) AWS-Region. Laden Sie die Datei `sales_ts.000` herunter, um die Quelldaten zu sehen. . Sie können dieses Beispiel so ändern, dass es in einem anderen ausgeführt wird AWS-Region. Erstellen Sie einen Amazon S3 S3-Bucket in Ihrem gewünschten Format AWS-Region. Kopieren Sie die Verkaufsdaten mit einem Amazon-S3-Kopierbefehl. Aktualisieren Sie dann die Option für den Speicherort im `CREATE EXTERNAL TABLE`-Beispielbefehl auf Ihren Bucket.
>
> ```
> aws s3 cp s3://redshift-downloads/tickit/spectrum/sales/ s3://bucket-name/
> tickit/spectrum/sales/ --copy-props none --recursive
> ```

Die Ausgabe des Amazon-S3-Kopierbefehls bestätigt, dass die Dateien in den *bucket-name* Ihrer gewünschten AWS-Region kopiert wurden.

```
copy: s3://redshift-downloads/tickit/spectrum/sales/sales_ts.000 to
 s3://bucket-name/tickit/spectrum/sales/sales_ts.000
```

```
create external table myspectrum_schema.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='172000');
```

Schritt 4: Abfragen Ihrer Daten in Amazon S3

Nach der Erstellung Ihrer externen Tabellen können Sie sie mit den gleichen SELECT-Anweisungen abfragen, mit denen Sie auch andere Amazon-Redshift-Tabellen abfragen. Diese SELECT-Anweisungsabfragen beinhalten die Verbindung von Tabellen, die Aggregierung von Daten und die Filterung auf Prädikaten.

So fragen Sie Ihre Daten in Amazon S3 ab

1. Abruf der Anzahl der Zeilen in der Tabelle MYSPECTRUM_SCHEMA.SALES.

```
select count(*) from myspectrum_schema.sales;
```

```
count
```

```
------
172462
```

2. Es wird empfohlen, dass Sie Ihre größeren Faktentabellen in Amazon S3 und Ihre kleineren Dimensionstabellen in Amazon Redshift speichern. Wenn Sie die Beispieldaten in Load data geladen haben, haben Sie eine Tabelle mit dem Namen EVENT in Ihrer Datenbank. Wenn dies nicht der Fall ist, erstellen Sie mit dem folgenden Befehl die Tabelle EVENT.

```
create table event(
eventid integer not null distkey,
venueid smallint not null,
catid smallint not null,
dateid smallint not null sortkey,
eventname varchar(200),
starttime timestamp);
```

3. Laden Sie die Tabelle „EVENT“, indem Sie den IAM-Rollen-ARN in dem folgenden COPY-Befehl durch den Rollen-ARN ersetzen, den Sie in erstellt haben Schritt 1. Erstellen Sie eine IAM-Rolle für Amazon Redshift. Sie können optional die Quelldaten für `allevents_pipe.txt` aus einem Amazon S3 S3-Bucket in herunterladen und anzeigen AWS-Region `us-east-1`.

```
copy event from 's3://redshift-downloads/tickit/allevents_pipe.txt'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
delimiter '|' timeformat 'YYYY-MM-DD HH:MI:SS' region 'us-east-1';
```

Das folgende Beispiel verbindet die externe Amazon-S3-Tabelle MYSPECTRUM_SCHEMA.SALES mit der lokalen Amazon-Redshift-Tabelle EVENT, um den Gesamtumsatz für die 10 führenden Veranstaltungen zu ermitteln.

```
select top 10 myspectrum_schema.sales.eventid,
 sum(myspectrum_schema.sales.pricepaid) from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    289 | 51846.00
   7895 | 51049.00
```

```
  1602 | 50301.00
   851 | 49956.00
  7315 | 49823.00
  6471 | 47997.00
  2118 | 47863.00
   984 | 46780.00
  7851 | 46661.00
  5638 | 46280.00
```

4. Zeigen Sie den Abfrageplan für die vorherige Abfrage an. Beachten Sie, dass die Schritte `S3 Seq Scan`, `S3 HashAggregate` und `S3 Query Scan` für die Daten auf Amazon S3 ausgeführt wurden.

```
explain
select top 10 myspectrum_schema.sales.eventid,
 sum(myspectrum_schema.sales.pricepaid)
from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader
```

```
              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200
width=31)

                    Sort Key: sum(sales.derived_col2)


                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99
rows=200 width=31)

                          ->  XN Hash Join DS_BCAST_INNER
(cost=3119.97..1055769620.49 rows=200000 width=31)

                                Hash Cond: ("outer".derived_col1 = "inner".eventid)


                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
rows=200000 width=31)

                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
rows=200000 width=16)

                                            ->  S3 Seq Scan myspectrum_schema.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                  Filter: (pricepaid > 30.00)


                                ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)


                                      ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Starten Sie Ihren AWS CloudFormation Stack und fragen Sie dann Ihre Daten in Amazon S3 ab

Nachdem Sie einen Amazon Redshift Redshift-Cluster erstellt und eine Verbindung zum Cluster hergestellt haben, können Sie Ihre Redshift DataLake AWS CloudFormation Spectrum-Vorlage installieren und dann Ihre Daten abfragen.

CloudFormation installiert die Redshift Spectrum Getting DataLake Started-Vorlage und erstellt einen Stack, der Folgendes umfasst:

- Eine Rolle namens `myspectrum_role`, die mit Ihrem Redshift-Cluster verknüpft ist
- Ein externes Schema mit dem Namen `myspectrum_schema`
- Eine externe Tabelle mit dem Namen `sales` in einem Amazon S3 Bucket
- Eine Redshift-Tabelle mit dem Namen `event` und geladenen Daten

So starten Sie Ihren Redshift Spectrum Getting Started Stack DataLake CloudFormation

1. Klicken Sie auf Launch CFN stack (CFN-Stack starten). Die CloudFormation Konsole wird mit der ausgewählten Vorlage DataLake .yml geöffnet.

   Sie können auch die Redshift Spectrum Getting Started DataLake CloudFormation CFN-Vorlage herunterladen und anpassen, dann die CloudFormation Konsole (https://console.aws.amazon.com/cloudformation) öffnen und einen Stack mit der benutzerdefinierten Vorlage erstellen.
2. Wählen Sie Next (Weiter).
3. Geben Sie unter Parameters (Parameter) den Namen des Amazon-Redshift-Clusters, den Datenbanknamen und Ihren Namen als Datenbankbenutzer ein.
4. Wählen Sie Next (Weiter).

   Die Stack-Optionen werden angezeigt.
5. Klicken Sie auf Next (Weiter), um die Standardeinstellungen zu übernehmen.
6. Lesen Sie die Informationen und wählen Sie unter Funktionen die Option Ich bestätige, dass AWS CloudFormation möglicherweise IAM-Ressourcen erstellt werden.
7. Wählen Sie Create stack (Stack erstellen) aus.

Falls während der Erstellung des Stacks ein Fehler auftritt, lesen Sie die folgenden Informationen:

- Auf der Registerkarte CloudFormation Ereignisse finden Sie Informationen, die Ihnen bei der Behebung des Fehlers helfen können.
- Löschen Sie den DataLake CloudFormation Stapel, bevor Sie den Vorgang erneut versuchen.
- Vergewissern Sie sich, dass Sie mit Ihrer Amazon-Redshift-Datenbank verbunden sind.

- Stellen Sie sicher, dass Sie die richtigen Informationen für den Namen des Amazon-Redshift-Clusters, den Datenbanknamen und den Namen des Datenbankbenutzers eingegeben haben.

Abfragen Ihrer Daten in Amazon S3

Sie fragen externe Tabellen mit den gleichen SELECT-Anweisungen ab, mit denen Sie auch andere Amazon-Redshift-Tabellen abfragen. Diese SELECT-Anweisungsabfragen beinhalten die Verbindung von Tabellen, die Aggregierung von Daten und die Filterung auf Prädikaten.

Die folgende Abfrage gibt die Anzahl von Zeilen in der externen Tabelle `myspectrum_schema.sales` aus.

```
select count(*) from myspectrum_schema.sales;
```

```
count
------
172462
```

Verbinden einer externen mit einer lokalen Tabelle

Das folgende Beispiel verbindet die externe Tabelle `myspectrum_schema.sales` mit der lokalen Tabelle `event`, um den Gesamtumsatz für die 10 führenden Veranstaltungen zu ermitteln.

```
select top 10 myspectrum_schema.sales.eventid, sum(myspectrum_schema.sales.pricepaid)
 from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
   6471 | 47997.00
   2118 | 47863.00
    984 | 46780.00
```

```
   7851 | 46661.00
   5638 | 46280.00
```

Aufrufen des Abfrageplans

Zeigen Sie den Abfrageplan für die vorherige Abfrage an. Beachten Sie, dass die Schritte `S3 Seq Scan`, `S3 HashAggregate` und `S3 Query Scan` für die Daten auf Amazon S3 ausgeführt wurden.

```
explain
select top 10 myspectrum_schema.sales.eventid, sum(myspectrum_schema.sales.pricepaid)
from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader


              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


                    Sort Key: sum(sales.derived_col2)
```

```
                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99 rows=200
width=31)

                          ->  XN Hash Join DS_BCAST_INNER  (cost=3119.97..1055769620.49
rows=200000 width=31)

                                Hash Cond: ("outer".derived_col1 = "inner".eventid)


                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
rows=200000 width=31)

                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
rows=200000 width=16)

                                            ->  S3 Seq Scan spectrum.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                  Filter: (pricepaid > 30.00)


                                ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)


                                      ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

IAM-Richtlinien für Amazon Redshift Spectrum

Standardmäßig verwendet Amazon Redshift Spectrum die AWS Regionen, die AWS Glue Data Catalog dies unterstützen AWS Glue. In anderen AWS Regionen verwendet Redshift Spectrum den Athena-Datenkatalog. Ihr Cluster benötigt eine Autorisierung, um auf Ihren externen Datenkatalog in AWS Glue oder Athena und Ihre Datendateien in Amazon S3 zuzugreifen. Sie erteilen diese Autorisierung, indem Sie auf eine AWS Identity and Access Management (IAM-) Rolle verweisen, die Ihrem Cluster zugeordnet ist. Wenn Sie einen Apache-Hive-Metastore zur Verwaltung Ihres Datenkatalogs verwenden, müssen Sie keinen Zugriff auf Athena ermöglichen.

Sie können Rollen miteinander verketten. Auf diese Weise kann der Cluster andere Rolle annehmen, die nicht dem Cluster angefügt sind. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.

Der AWS Glue Katalog, auf den Sie zugreifen, ist möglicherweise verschlüsselt, um die Sicherheit zu erhöhen. Wenn der AWS Glue Katalog verschlüsselt ist, benötigen Sie den AWS KMS Schlüssel für, AWS Glue um auf den AWS Glue Datenkatalog zuzugreifen. Weitere Informationen finden Sie unter Verschlüsseln Ihres AWS Glue Datenkatalogs im AWS Glue Entwicklerhandbuch.

Themen

Amazon-S3-Berechtigungen

Ihr Cluster benötigt mindestens GET- und LIST-Zugriff zu Ihrem Amazon-S3-Bucket. Wenn sich Ihr Bucket nicht in demselben AWS Konto wie Ihr Cluster befindet, muss Ihr Bucket auch Ihren Cluster autorisieren, auf die Daten zuzugreifen. Weitere Informationen finden Sie unter Autorisieren von Amazon Redshift, in Ihrem Namen auf andere AWS Dienste zuzugreifen.

> **Note**
>
> Der Amazon-S3-Bucket kann keine Bucket-Richtlinie verwenden, die den Zugriff nur auf bestimmte VPC-Endpunkte beschränkt.

Die folgende Richtlinie gewährt GET- und LIST-Zugriff auf alle Amazon-S3-Buckets. Die Richtlinie erlaubt den Zugriff auf Amazon-S3-Buckets für Redshift Spectrum sowie COPY-Vorgänge.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["s3:Get*", "s3:List*"],
  "Resource": "*"
 }]
}
```

Die folgende Richtlinie gewährt GET- und LIST-Zugriff auf Ihren Amazon-S3-Bucket mit der Bezeichnung `myBucket`.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["s3:Get*", "s3:List*"],
  "Resource": "arn:aws:s3:::myBucket/*"
 }]
}
```

Kontoübergreifende Amazon-S3-Berechtigungen

Um Redshift Spectrum die Erlaubnis zu erteilen, auf Daten in einem Amazon S3 S3-Bucket zuzugreifen, der zu einem anderen AWS Konto gehört, fügen Sie dem Amazon S3 S3-Bucket die folgende Richtlinie hinzu. Weitere Informationen finden Sie unter Beispiel 2: Bucket-Eigentümer erteilt kontoübergreifende Bucket-Berechtigungen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Example permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::redshift-account:role/spectrumrole"
            },
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListMultipartUploadParts",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::bucketname",
                "arn:aws:s3:::bucketname/*"
            ]
        }
    ]
}
```

Richtlinien zum Gewähren oder Beschränken des Zugriffs mit Redshift Spectrum

Um den Zugriff auf einen Amazon-S3-Bucket nur mit der Verwendung von Redshift Spectrum zu gewähren, fügen Sie eine Bedingung ein, die den Zugriff für den Benutzeragenten `AWS Redshift/Spectrum` gewährt. Die folgende Richtlinie erlaubt den Zugriff auf Amazon-S3-Buckets nur für Redshift Spectrum. Andere Zugriffsmöglichkeiten, etwa COPY-Operationen, sind ausgeschlossen.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": ["s3:Get*", "s3:List*"],
        "Resource": "arn:aws:s3:::myBucket/*",
                "Condition": {"StringEquals": {"aws:UserAgent": "AWS Redshift/
Spectrum"}}
    }]
}
```

Auf ähnliche Weise können Sie eine IAM-Rolle erstellen, die den Zugriff für COPY-Operationen erlaubt, den Zugriff für Redshift Spectrum jedoch ausschließt. Fügen Sie dazu eine Bedingung ein, die dem Benutzeragenten **`AWS Redshift/Spectrum`** den Zugriff verweigert. Die folgende Richtlinie erlaubt den Zugriff auf einen Amazon-S3-Bucket, ausgenommen für Redshift Spectrum.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": ["s3:Get*", "s3:List*"],
        "Resource": "arn:aws:s3:::myBucket/*",
                "Condition": {"StringNotEquals": {"aws:UserAgent": "AWS Redshift/
Spectrum"}}
    }]
}
```

Richtlinien zum Gewähren von Mindestberechtigungen

Die folgende Richtlinie gewährt die Mindestberechtigungen, die für die Verwendung von Redshift Spectrum mit Amazon S3 AWS Glue, und Athena erforderlich sind.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListMultipartUploadParts",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::bucketname",
                "arn:aws:s3:::bucketname/folder1/folder2/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:DeleteDatabase",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:UpdateDatabase",
                "glue:CreateTable",
                "glue:DeleteTable",
                "glue:BatchDeleteTable",
                "glue:UpdateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:BatchCreatePartition",
                "glue:CreatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition",
                "glue:UpdatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition"
            ],
            "Resource": [
                "*"
            ]
```

```
        }
    ]
}
```

Wenn Sie stattdessen Athena für Ihren Datenkatalog verwenden AWS Glue, erfordert die Richtlinie vollen Athena-Zugriff. Die folgende Richtlinie gewährt den Zugriff auf Athena-Ressourcen. Wenn sich Ihre externe Datenbank in einem Hive-Metastore befindet, benötigen Sie keinen Athena-Zugriff.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["athena:*"],
  "Resource": ["*"]
 }]
}
```

Verketten von IAM-Rollen in Amazon Redshift Spectrum

Wenn Sie Ihrem Cluster eine Rolle zuordnen, kann Ihr Cluster diese Rolle übernehmen, um auf Amazon S3, Athena und in AWS Glue Ihrem Namen zuzugreifen. Wenn eine an den Cluster angefügte Rolle keinen Zugriff auf die erforderlichen Ressourcen hat, können Sie mit ihr eine andere Rolle verketten, die möglicherweise zu einem anderen Konto gehört. Ihr Cluster nimmt dann vorübergehend die verkettete Rolle an, um auf die Daten zuzugreifen. Sie können über verkettete Rollen auch kontoübergreifenden Zugriff gewähren. Sie können maximal 10 Rollen miteinander verketten. Jede Rolle in der Kette nimmt die nächste Rolle in der Kette an, bis hin zum Cluster, der die Rolle am Ende der Kette annimmt.

Zum Verketten von Rollen richten Sie eine Vertrauensstellung zwischen den Rollen ein. Eine Rolle, die eine andere Rolle annimmt, muss über eine Berechtigungsrichtlinie verfügen, die es ihr erlaubt, die angegebene Rolle anzunehmen. Die Rolle, die Berechtigungen übergibt, muss wiederum über eine Vertrauensstellung verfügen, die es ihr erlaubt, ihre Berechtigungen an eine andere Rolle zu übergeben. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift.

Wenn Sie den Befehl CREATE EXTERNAL SCHEMA ausführen, können Sie Rollen durch Einschluss einer durch Komma getrennten Liste von Rollen-ARNs verketten.

> **Note**
>
> Die Liste der verketteten Rollen darf keine Leerstellen enthalten.

Im folgenden Beispiel wird `MyRedshiftRole` an den Cluster angehängt. `MyRedshiftRole` nimmt die Rolle `AcmeData` an, die zum Konto `111122223333` gehört.

```
create external schema acme from data catalog
database 'acmedb' region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole,arn:aws:iam::111122223333:role/
AcmeData';
```

Steuern des Zugriffs auf den AWS Glue Datenkatalog

Wenn Sie dies AWS Glue für Ihren Datenkatalog verwenden, können Sie mit Ihrer IAM-Richtlinie eine differenzierte Zugriffskontrolle auf den AWS Glue Datenkatalog anwenden. Beispielsweise können Sie festlegen, dass nur bestimmte Datenbanken und Tabellen für eine spezifische IAM-Rolle bereitgestellt werden.

In den folgenden Abschnitten werden die IAM-Richtlinien für verschiedene Zugriffsebenen auf im Datenkatalog gespeicherte Daten beschrieben. AWS Glue

Themen

- Richtlinie für Datenbankoperationen
- Richtlinie für Tabellenoperationen
- Richtlinie für Partitionsoperationen

Richtlinie für Datenbankoperationen

Wenn Sie Benutzern Berechtigungen zum Anzeigen und Erstellen einer Datenbank erteilen möchten, benötigen sie Zugriffsrechte sowohl für die Datenbank als auch für den AWS Glue Datenkatalog.

Mit der folgenden Beispielabfrage wird eine Datenbank erstellt.

```
CREATE EXTERNAL SCHEMA example_db
FROM DATA CATALOG DATABASE 'example_db' region 'us-west-2'
```

```
IAM_ROLE 'arn:aws:iam::redshift-account:role/spectrumrole'
CREATE EXTERNAL DATABASE IF NOT EXISTS
```

Mit der folgenden IAM-Richtlinie werden die zum Erstellen einer Datenbank benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:CreateDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:catalog"
            ]
        }
    ]
}
```

Mit der folgenden Beispielabfrage werden die aktuellen Datenbanken aufgelistet.

```
SELECT * FROM SVV_EXTERNAL_DATABASES WHERE
databasename = 'example_db1' or databasename = 'example_db2';
```

Mit der folgenden IAM-Richtlinie werden die zum Auflisten der aktuellen Datenbanken benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "glue:GetDatabases"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:database/example_db1",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db2",
                "arn:aws:glue:us-west-2:redshift-account:catalog"

            ]
        }
    ]
}
```

Richtlinie für Tabellenoperationen

Wenn Sie Benutzern Berechtigungen für View, Create, Drop, Alter oder andere Aktionen für Tabellen erteilen möchten, benötigen sie mehrere Zugriffsarten. Sie benötigen Zugriff auf die Tabellen selbst, die Datenbanken, zu denen sie gehören, und den Katalog.

Mit der folgenden Beispielabfrage wird eine externe Tabelle erstellt.

```
CREATE EXTERNAL TABLE example_db.example_tbl0(
    col0 INT,
    col1 VARCHAR(255)
) PARTITIONED BY (part INT) STORED AS TEXTFILE
LOCATION 's3://test/s3/location/';
```

Mit der folgenden IAM-Richtlinie werden die zum Erstellen einer externen Tabelle benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateTable"
            ],
```

```
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Mit den folgenden Beispielabfragen werden die aktuellen externen Tabellen aufgelistet.

```
SELECT * FROM svv_external_tables
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

```
SELECT * FROM svv_external_columns
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

```
SELECT parameters FROM svv_external_tables
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

Mit der folgenden IAM-Richtlinie werden die zum Auflisten der aktuellen externen Tabellen benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
```

```
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/
example_tbl0",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl1"

            ]
        }
    ]
}
```

Mit der folgenden Beispielabfrage wird eine vorhandene Tabelle geändert.

```
ALTER TABLE example_db.example_tbl0
SET TABLE PROPERTIES ('numRows' = '100');
```

Mit der folgenden IAM-Richtlinie werden die zum Ändern einer vorhandenen Tabelle benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:UpdateTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"

            ]
        }
    ]
```

```
}
```

Mit der folgenden Beispielabfrage wird eine vorhandene Tabelle entfernt.

```
DROP TABLE example_db.example_tbl0;
```

Mit der folgenden IAM-Richtlinie werden die zum Entfernen einer vorhandenen Tabelle benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:DeleteTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Richtlinie für Partitionsoperationen

Wenn Sie Benutzern die Berechtigung zum Ausführen von Vorgängen auf Partitionsebene (Anzeigen, Erstellen, Entfernen, Ändern usw.) erteilen möchten, benötigen sie Berechtigungen für die Tabellen, zu denen die Partitionen gehören. Sie benötigen darüber hinaus Berechtigungen für die zugehörigen Datenbanken und den AWS Glue -Datenkatalog.

Mit der folgenden Beispielabfrage wird eine Partition erstellt.

```
ALTER TABLE example_db.example_tbl0
```

```
ADD PARTITION (part=0) LOCATION 's3://test/s3/location/part=0/';
ALTER TABLE example_db.example_t
ADD PARTITION (part=1) LOCATION 's3://test/s3/location/part=1/';
```

Mit der folgenden IAM-Richtlinie werden die zum Erstellen einer Partition benötigten Mindestberechtigungen erteilt.

```
 {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:BatchCreatePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Mit der folgenden Beispielabfrage werden die aktuellen Partitionen aufgelistet.

```
SELECT * FROM svv_external_partitions
WHERE schemname = 'example_db' AND
tablename = 'example_tbl0'
```

Mit der folgenden IAM-Richtlinie werden die zum Auflisten der aktuellen Partitionen benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetPartitions",
                "glue:GetTables",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Mit der folgenden Beispielabfrage wird eine vorhandene Partition geändert.

```
ALTER TABLE example_db.example_tbl0 PARTITION(part='0')
SET LOCATION 's3://test/s3/new/location/part=0/';
```

Mit der folgenden IAM-Richtlinie werden die zum Ändern einer vorhandenen Partition benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetPartition",
                "glue:UpdatePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
```

```
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Mit der folgenden Beispielabfrage wird eine vorhandene Partition entfernt.

```
ALTER TABLE example_db.example_tbl0 DROP PARTITION(part='0');
```

Mit der folgenden IAM-Richtlinie werden die zum Entfernen einer vorhandenen Partition benötigten Mindestberechtigungen erteilt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:DeletePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Verwenden von Redshift Spectrum mit AWS Lake Formation

Sie können AWS Lake Formation damit zentral Zugriffsrichtlinien auf Datenbank-, Tabellen- und Spaltenebene für in Amazon S3 gespeicherte Daten definieren und durchsetzen. Nachdem Ihre

Daten bei einem mit Lake Formation aktivierten AWS Glue Data Catalog registriert wurden, können Sie Abfragen über verschiedene Services senden, darunter auch Redshift Spectrum.

Lake Formation sorgt für die Sicherheit und Governance des Datenkatalogs. Sie können innerhalb von Lake Formation Berechtigungen für die Datenkatalogobjekte wie Datenbanken, Tabellen, Spalten und zugrundeliegenden Amazon-S3-Speicher gewähren und widerrufen.

> **Important**
>
> Sie können Redshift Spectrum mit einem Datenkatalog, für den Lake Formation aktiviert ist, nur in AWS Regionen verwenden, in denen Lake Formation verfügbar ist. Eine Liste der verfügbaren Regionen finden Sie unter AWS Lake Formation -Endpunkte und -Kontingente im Allgemeine AWS-Referenz.

Wenn Sie Redshift Spectrum mit Lake Formation verwenden, haben Sie folgende Möglichkeiten:

- Verwenden Sie Lake Formation für die zentrale Steuerung von Berechtigungen und Zugriffskontrollrichtlinien für alle Ihre Daten im Data Lake. Lake Formation stellt eine Berechtigungshierarchie bereit, mit der Sie den Zugriff auf Datenbanken und Tabellen in einem Datenkatalog steuern können. Weitere Informationen finden Sie unter Übersicht über Lake-Formation-Berechtigungen im AWS Lake Formation -Entwicklerhandbuch.
- Erstellen Sie externe Tabellen und führen Sie Abfragen zu Daten im Data Lake aus. Bevor Benutzer in Ihrem Konto Abfragen ausführen können, registriert ein Data Lake-Kontoadministrator Ihre vorhandenen Amazon-S3-Pfade, die Quelldaten enthalten, mit Lake Formation. Der Administrator erstellt auch Tabellen und gewährt den Benutzern Berechtigungen. Zugriff kann auf Datenbanken, Tabellen oder Spalten gewährt werden. Der Administrator kann Datenfilter in Lake Formation verwenden, um eine detaillierte Zugriffskontrolle für Ihre sensiblen Daten zu gewähren, die in Amazon S3 gespeichert sind. Weitere Informationen finden Sie unter Verwenden von Datenfiltern für die Sicherheit auf Zeilen- und Zellenebene.

  Nachdem die Daten im Datenkatalog registriert wurden, verifiziert Lake Formation jedes Mal, wenn Benutzer versuchen, Abfragen auszuführen, ob sie Zugriff auf die Tabelle für diesen spezifischen Prinzipal haben. Lake Formation vergibt temporäre Anmeldeinformationen an Redshift Spectrum und die Abfrage wird ausgeführt.
- Führen Sie Redshift Spectrum-Abfragen für eine automatisch gemountete Datei mit IAM-Anmeldeinformationen aus, die AWS Glue Data Catalog Sie mit `GetCredentials` oder erhalten

habenGetClusterCredentials, und verwalten Sie Lake Formation Formation-Berechtigungen nach Datenbankbenutzern (iamr:Username oder iam:Username).

Wenn Sie Redshift Spectrum mit einem für Lake Formation aktivierten Datenkatalog verwenden, muss eines der im Folgenden Genannten vorhanden sein:

- Eine IAM-Rolle, die dem Cluster zugeordnet ist und über die Berechtigung für den Datenkatalog verfügt.
- Eine IAM-Verbundidentität, die für die Verwaltung des Zugriffs auf externe Ressourcen konfiguriert ist. Weitere Informationen finden Sie unter Verwenden einer Verbundidentität zur Verwaltung des Amazon-Redshift-Zugriffs auf lokale Ressourcen und externe Amazon-Redshift-Tabellen.

> **Important**
>
> Wenn Sie Redshift Spectrum mit einem für Lake Formation aktivierten Datenkatalog verwenden, können Sie IAM-Rollen nicht verketten.

Weitere Informationen zu den Schritten, die AWS Lake Formation zur Einrichtung für die Verwendung mit Redshift Spectrum erforderlich sind, finden Sie unter Tutorial: Erstellen eines Data Lake aus einer JDBC-Quelle in Lake Formation im AWS Lake Formation Developer Guide. Weitere Informationen zur Integration in Redshift Spectrum finden Sie insbesondere unter Abfrage der Daten im Data Lake mit Amazon Redshift Spectrum. Die in diesem Thema verwendeten Daten und AWS Ressourcen hängen von den vorherigen Schritten im Tutorial ab.

Verwenden von Datenfiltern für die Sicherheit auf Zeilen- und Zellenebene

Sie können Datenfilter definieren, AWS Lake Formation um den Zugriff Ihrer Redshift Spectrum-Abfragen auf Zeilen- und Zellenebene auf Daten zu steuern, die in Ihrem Datenkatalog definiert sind. Dazu führen Sie die folgenden Aufgaben aus:

- Erstellen Sie in Lake Formation einen Datenfilter mit den folgenden Informationen:
  - Eine Spaltenspezifikation mit einer Liste von Spalten, die in Abfrageergebnissen ein- oder ausgeschlossen werden sollen.
  - Ein Zeilenfilterausdruck, der die Zeilen angibt, die in die Abfrageergebnisse aufgenommen werden sollen.

Weitere Informationen zum Erstellen eines Datenfilters finden Sie unter Datenfilter in Lake Formation im AWS Lake Formation -Entwicklerhandbuch.

- Erstellen Sie eine externe Tabelle in Amazon Redshift, die auf eine Tabelle in Ihrem Datenkatalog verweist, der für Lake Formation aktiviert ist. Weitere Informationen zum Abfragen einer Lake-Formation-Tabelle mithilfe von Redshift Spectrum finden Sie unter Abfrage der Daten im Data Lake mit Amazon Redshift Spectrum im AWS Lake Formation -Entwicklerhandbuch.

Nachdem die Tabelle in Amazon Redshift definiert wurde, können Sie die Lake-Formation-Tabelle abfragen und nur auf die Zeilen und Spalten zugreifen, die vom Datenfilter zugelassen sind.

Eine ausführliche Anleitung zum Einrichten der Sicherheit auf Zeilen- und Zellenebene in Lake Formation und zum anschließenden Abfragen mithilfe von Redshift Spectrum finden Sie unter Verwenden von Amazon Redshift Spectrum mit Sicherheitsrichtlinien auf Zeilen- und Zellebene, die in AWS Lake Formation definiert sind.

Erstellen von Datendateien für Abfragen in Amazon Redshift Spectrum

Die Datendateien, die Sie für Abfragen in Amazon Redshift Spectrum verwenden, sind im Allgemeinen dieselben Dateitypen, die Sie für andere Anwendungen verwenden. Beispielsweise werden dieselben Dateitypen mit Amazon Athena, Amazon EMR und Amazon verwendet. QuickSight Sie können die Daten im Originalformat direkt aus Amazon S3 abfragen. Dazu müssen die Datendateien in einem Format vorliegen, das Redshift Spectrum unterstützt und sich in einem Amazon-S3-Bucket befinden, auf den Ihr Cluster zugreifen kann.

Der Amazon S3 S3-Bucket mit den Datendateien und der Amazon Redshift Redshift-Cluster müssen sich in derselben AWS Region befinden. Informationen zu den unterstützten AWS Regionen finden Sie unterAmazon-Redshift-Spectrum-Regionen.

Datenformate für Redshift Spectrum

Redshift Spectrum unterstützt die folgenden strukturierten und halbstrukturierten Datenformate.

Dateiformat	Säulenförmig	Unterstützt parallele Lesevorgänge	Geteilte Einheit
Parquet	Ja	Ja	Zeilengruppe
ORC	Ja	Ja	Stripe
RCFile	Ja	Ja	Zeilengruppe
TextFile	Nein	Ja	Zeile
SequenceFile	Nein	Ja	Zeile oder Block
RegexSerde	Nein	Ja	Zeile
OpenCSV	Nein	Ja	Zeile
AVRO	Nein	Ja	Block
Ion	Nein	Nein	N/A
JSON	Nein	Nein	N/A

In der vorhergehenden Tabelle geben die Überschriften Folgendes an:

- Columnar (Spaltenorientiert) – Gibt an, ob das Dateiformat Daten physisch in einer spaltenorientierten Struktur im Gegensatz zu einer zeilenorientierten Struktur speichert.
- Supports Parallel reads (Unterstützt parallele Lesevorgänge) – Gibt an, ob das Dateiformat das Lesen einzelner Blöcke innerhalb der Datei unterstützt. Das Lesen einzelner Blöcke ermöglicht die verteilte Verarbeitung einer Datei über mehrere unabhängige Redshift-Spectrum-Anfragen, anstatt die vollständige Datei in einer einzigen Anforderung lesen zu müssen.
- Split Unit (Geteilte Einheit) – Bei Dateiformaten, die parallel gelesen werden können, ist die geteilte Einheit der kleinste Datenblock, den eine einzelne Redshift-Spectrum-Anforderung verarbeiten kann.

> **Note**
>
> Die Zeitstempelwerte in Textdateien müssen das Format `yyyy-MM-dd HH:mm:ss.SSSSSS` haben, wie der folgende Zeitstempelwert zeigt: `2017-05-01 11:30:59.000000`.

Wir empfehlen die Verwendung eines nach Spalten organisierten Speicherdateiformats, wie etwa Apache Parquet. Mit einem solchen Format minimieren Sie die Datenübertragung aus Amazon S3, indem Sie nur die benötigten Spalten auswählen.

Komprimierungstypen für Redshift Spectrum

Um den Speicherplatz zu reduzieren, die Leistung zu verbessern und die Kosten zu minimieren, empfehlen wir dringend, Ihre Datendateien zu komprimieren. Redshift Spectrum erkennt Dateikomprimierungstypen anhand der Dateierweiterung.

Redshift Spectrum unterstützt die folgenden Komprimierungstypen und Erweiterungen.

Komprimierungsalgorithmus	Dateierweiterung	Unterstützt parallele Lesevorgänge
Gzip	.gz	Nein
Bzip2	.bz2	Ja
Snappy	.snappy	Nein

Sie können die Komprimierung auf verschiedenen Ebenen anwenden. Zumeist komprimieren Sie eine ganze Datei oder einzelne Blöcke innerhalb einer Datei. Das Komprimieren von Spaltenformaten auf Dateiebene bringt keine Leistungsvorteile.

Damit Redshift Spectrum eine Datei parallel lesen kann, muss Folgendes zutreffen:

- Das Dateiformat unterstützt parallele Lesevorgänge.
- Die Komprimierung auf Dateiebene (falls vorhanden) unterstützt parallele Lesevorgänge.

Es spielt keine Rolle, ob die einzelnen geteilten Einheiten innerhalb einer Datei mit einem Komprimierungsalgorithmus komprimiert werden, der parallel gelesen werden kann, da jede geteilte Einheit von einer einzigen Redshift-Spectrum-Anforderung verarbeitet wird. Ein Beispiel dafür sind Snappy-komprimierte Parquet-Dateien. Einzelne Zeilengruppen innerhalb der Parquet-Datei werden mit Snappy komprimiert, aber die oberste Struktur der Datei bleibt unkomprimiert. In diesem Fall kann die Datei parallel gelesen werden, da jede Redshift-Spectrum-Anforderung einzelne Zeilengruppen von Amazon S3 lesen und verarbeiten kann.

Verschlüsselung für Redshift Spectrum

Redshift Spectrum entschlüsselt in transparenter Weise Datendateien, die mit den folgenden Verschlüsselungsoptionen verschlüsselt wurden:

- Serverseitige Verschlüsselung (SSE-S3) mit einem von Amazon S3 verwalteten AES-256-Verschlüsselungsschlüssel.
- Serverseitige Verschlüsselung mit Schlüsseln, die von AWS Key Management Service (SSE-KMS) verwaltet werden.

Redshift Spectrum unterstützt die clientseitige Amazon-S3-Verschlüsselung nicht. Weitere Informationen zur serverseitigen Verschlüsselung finden Sie unter Schutz von Daten mittels serverseitiger Verschlüsselung im Amazon-Simple-Storage-Service-Benutzerhandbuch.

Amazon Redshift unterstützt die massiv parallele Verarbeitung (Massively Parallel Processing, MPP) für die schnelle Ausführung komplexer Abfragen für sehr große Datenmengen. Redshift Spectrum verwendet das gleiche Prinzip auch für die Abfrage externer Daten und nutzt nach Bedarf mehrere Redshift Spectrum-Instances zum Scanning von Dateien. Setzen Sie die Dateien in separate Ordner für jede Tabelle.

Sie können mithilfe der folgenden Verfahren Ihre Daten für die parallele Verarbeitung optimieren:

- Wenn Ihr Dateiformat oder Ihre Komprimierung das parallele Lesen nicht unterstützt, teilen Sie große Dateien in viele kleinere Dateien auf. Wir empfehlen Dateigrößen zwischen 64 MB und 1 GB.
- Sorgen Sie dafür, dass alle Dateien etwa die gleiche Größe haben. Wenn einige Dateien viel größer als andere sind, kann Redshift Spectrum den Workload nicht gleichmäßig verteilen.

Erstellen externer Schemata für Amazon Redshift Spectrum

Alle externen Tabellen müssen in einem externen Schema erstellt werden, das Sie mit einer CREATE EXTERNAL SCHEMA-Anweisung erstellen.

> **Note**
>
> Einige Anwendungen verwenden die Begriffe Datenbank und Schema mit gleicher Bedeutung. In Amazon Redshift verwenden wir den Begriff Schema.

Ein externes Amazon-Redshift-Schema verweist auf eine externe Datenbank in einem externen Datenkatalog. Sie können die externe Datenbank in Amazon Redshift, in Amazon Athena, in AWS Glue Data Catalog oder in einem Apache-Hive-Metastore, wie etwa Amazon EMR, erstellen. Wenn Sie eine externe Datenbank in Amazon Redshift erstellen, befindet sich diese im Athena-Datenkatalog. Zur Erstellung einer Datenbank in einem Hive-Metastore müssen Sie sie in Ihrer Hive-Anwendung erstellen.

Amazon Redshift benötigt in Ihrem Namen die Autorisierung zum Zugriff auf den Datenkatalog in Athena und die Datendateien in Amazon S3. Um diese Autorisierung bereitzustellen, erstellen Sie zunächst eine AWS Identity and Access Management (IAM-) Rolle. Dann fügen Sie die Rolle Ihrem Cluster hinzu und stellen den Amazon-Ressourcennamen (ARN) für die Rolle in der `CREATE EXTERNAL SCHEMA`-Anweisung von Amazon Redshift bereit. Weitere Informationen zur -Autorisierung finden Sie unter IAM-Richtlinien für Amazon Redshift Spectrum.

> **Note**
>
> Wenn Sie derzeit externe Redshift Spectrum-Tabellen im Athena-Datenkatalog haben, können Sie Ihren Athena-Datenkatalog in einen AWS Glue Datenkatalog migrieren. Um einen AWS Glue Datenkatalog mit Redshift Spectrum zu verwenden, müssen Sie möglicherweise Ihre IAM-Richtlinien ändern. Weitere Informationen finden Sie unter Upgrade auf den AWS Glue Datenkatalog im Amazon Athena Athena-Benutzerhandbuch.

Um eine externe Datenbank gleichzeitig mit der Erstellung eines externen Schemas zu erstellen, geben Sie `FROM DATA CATALOG` an, und schließen Sie die `CREATE EXTERNAL DATABASE`-Klausel in Ihre `CREATE EXTERNAL SCHEMA`-Anweisung ein.

Das folgende Beispiel erstellt ein externes Schema mit der Bezeichnung `spectrum_schema` unter Verwendung der externen Datenbank `spectrum_db`.

```
create external schema spectrum_schema from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
create external database if not exists;
```

Wenn Sie Ihren Datenkatalog mit Athena verwalten, geben Sie den Athena-Datenbanknamen und die AWS -Region an, in der sich der Athena-Datenkatalog befindet.

Das folgende Beispiel erstellt ein externes Schema mit der Standarddatenbank `sampledb` im Athena-Datenkatalog.

```
create external schema athena_schema from data catalog
database 'sampledb'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
region 'us-east-2';
```

> **Note**
>
> Der `region` Parameter bezieht sich auf die AWS Region, in der sich der Athena-Datenkatalog befindet, nicht auf den Speicherort der Datendateien in Amazon S3.

Wenn Sie Ihren Datenkatalog mit einem Hive-Metastore, etwa mit Amazon EMR, verwalten, müssen Ihre Sicherheitsgruppen so konfiguriert sein, dass der Datenverkehr zwischen den Clustern möglich ist.

Geben Sie in der CREATE EXTERNAL SCHEMA-Anweisung `FROM HIVE METASTORE` an, und schließen Sie die URI und die Portnummer des Metastores ein. Im folgenden Beispiel wird ein externes Schema mittels einer Hive-Metastore-Datenbank namens erstellt `hive_db`.

```
create external schema hive_schema
from hive metastore
database 'hive_db'
uri '172.10.10.10' port 99
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
```

Um die externen Schemata für Ihren Cluster anzuzeigen, fragen Sie die Katalogtabelle PG_EXTERNAL_SCHEMA oder die Ansicht SVV_EXTERNAL_SCHEMAS ab. Das folgende Beispiel fragt SVV_EXTERNAL_SCHEMAS ab, wobei PG_EXTERNAL_SCHEMA und PG_NAMESPACE verbunden werden.

```
select * from svv_external_schemas
```

Für die vollständige Befehlssyntax und Beispiele vgl. CREATE EXTERNAL SCHEMA.

Arbeiten mit externen Katalogen in Amazon Redshift Spectrum

Die Metadaten für externe Amazon-Redshift-Spectrum-Datenbanken und externe Tabellen werden in einem externen Datenkatalog gespeichert. Standardmäßig werden Redshift-Spectrum-Metadaten in einem Athena-Datenkatalog gespeichert. Sie können Redshift-Spectrum-Datenbanken und -Tabellen in Ihrer Athena-Konsole anzeigen und verwalten.

Sie können externe Datenbanken und externe Tabellen auch mit der Hive-Data-Definition-Language (DDL) unter Verwendung von Athena oder eines Hive-Metastores wie Amazon EMR erstellen und verwalten.

> **Note**
>
> Wir empfehlen die Verwendung von Amazon Redshift zur Erstellung und Verwaltung Ihrer externen Redshift-Spectrum-Datenbanken und externer Tabellen.

Redshift Spectrum-Datenbanken in Athena anzeigen und AWS Glue

Sie können eine externe Datenbank erstellen, indem Sie die Klausel CREATE EXTERNAL DATABASE IF NOT EXISTS in Ihre CREATE EXTERNAL SCHEMA-Anweisung aufnehmen. In solchen Fällen werden die Metadaten der externen Datenbank in Ihrem Datenkatalog gespeichert. Die Metadaten für externe Tabellen, die Sie mit Qualifizierung durch das externe Schema erstellen, werden ebenfalls in Ihrem -Datenkatalog gespeichert.

Athena und AWS Glue pflegen einen Datenkatalog für jeden unterstützten AWS-Region. Um Tabellenmetadaten anzuzeigen, melden Sie sich bei Athena oder der AWS Glue Konsole an. Wählen Sie in Athena Data sources, your AWS Glue, und zeigen Sie dann die Details Ihrer Datenbank an. Wählen Sie in AWS GlueDatenbanken, Ihre externe Datenbank, und zeigen Sie dann die Details Ihrer Datenbank an.

Wenn Sie Ihre externen Tabellen mit Athena erstellen und verwalten, registrieren Sie die Datenbank mit CREATE EXTERNAL SCHEMA. Beispielsweise registriert der folgende Befehl die Athena-Datenbank mit der Bezeichnung `sampledb`.

```
create external schema athena_sample
from data catalog
database 'sampledb'
iam_role 'arn:aws:iam::123456789012:role/mySpectrumRole'
region 'us-east-1';
```

Wenn Sie die Systemansicht SVV_EXTERNAL_TABLES abfragen, sehen Sie Tabellen in der Athena-`sampledb`-Datenbank und die Tabellen, die Sie in Amazon Redshift erstellt haben.

```
select * from svv_external_tables;
```

```
schemaname    | tablename        | location

--------------+------------------
+-----------------------------------------------------
athena_sample | elb_logs         | s3://athena-examples/elb/plaintext
athena_sample | lineitem_1t_csv  | s3://myspectrum/tpch/1000/lineitem_csv

athena_sample | lineitem_1t_part | s3://myspectrum/tpch/1000/lineitem_partition

spectrum      | sales            | s3://redshift-downloads/tickit/spectrum/sales

spectrum      | sales_part       | s3://redshift-downloads/tickit/spectrum/sales_part
```

Registrieren einer Apache Hive-Metastore-Datenbank

Wenn Sie externe Tabellen in einem Apache Hive-Metastore erstellen, können Sie diese Tabellen mit CREATE EXTERNAL SCHEMA in Redshift Spectrum registrieren.

Geben Sie in der CREATE EXTERNAL SCHEMA-Anweisung die Klausel FROM HIVE METASTORE sowie die URI und die Portnummer des Hive-Metastores an. Die IAM-Rolle muss die Berechtigung zum Zugriff auf Amazon S3 enthalten, Athena-Berechtigungen sind jedoch nicht erforderlich. Das folgende Beispiel registriert einen Hive-Metastore.

```
create external schema if not exists hive_schema
```

```
from hive metastore
database 'hive_database'
uri 'ip-10-0-111-111.us-west-2.compute.internal' port 9083
iam_role 'arn:aws:iam::123456789012:role/mySpectrumRole';
```

Zugriff Ihres Amazon-Redshift-Clusters auf Ihren Amazon-EMR-Cluster

Wenn sich Ihr Hive-Metastore in Amazon EMR befindet, müssen Sie Ihrem Amazon-Redshift-Cluster den Zugriff auf Ihren Amazon-Redshift-Cluster gewähren. Dazu erstellen Sie eine Amazon-EC2-Sicherheitsgruppe. Sie lassen dann den gesamten eingehenden Datenverkehr an die EC2-Sicherheitsgruppe aus der Sicherheitsgruppe Ihres Amazon-Redshift-Clusters und der Sicherheitsgruppe Ihres Amazon-EMR-Clusters zu. Dann fügen Sie die EC2-Sicherheitsgruppe Ihrem Amazon-Redshift-Cluster und Ihrem Amazon-EMR-Cluster hinzu.

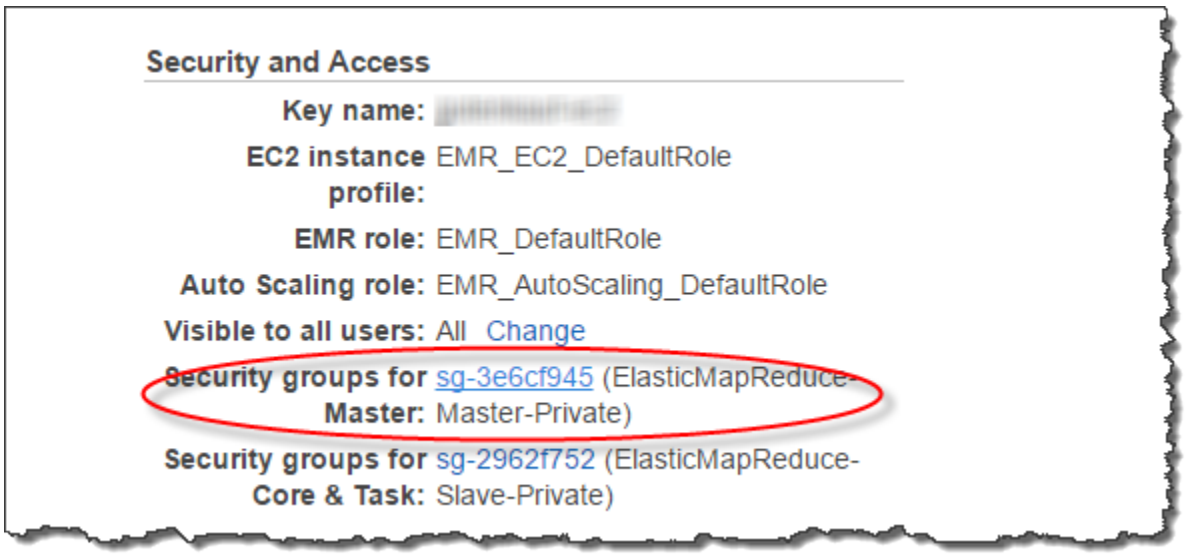
Aufrufen des Namens der Sicherheitsgruppe Ihres Amazon-Redshift-Clusters

Um die Sicherheitsgruppe anzuzeigen, gehen Sie wie folgt vor:

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü die Option Clusters (Cluster) und anschließend in der Liste den gewünschten Cluster aus, um die Details zu dem Cluster zu öffnen.
3. Wählen Sie Properties (Eigenschaften) aus und sehen Sie sich den Abschnitt Network and security (Netzwerk und Sicherheit) an.
4. Machen Sie Ihre Sicherheitsgruppe in VPC Security Group (VPC-Sicherheitsgruppe) ausfindig und notieren Sie sie.

Aufrufen des Namens der Amazon-EMR-Hauptknoten-Sicherheitsgruppe

1. Öffnen Sie Ihren Amazon-EMR-Cluster. Weitere Informationen finden Sie unter Verwenden von Sicherheitskonfigurationen zur Einrichtung der Clustersicherheit im Amazon-EMR-Verwaltungshandbuch
2. Notieren Sie sich unter Security and access (Sicherheit und Zugriff) den Namen der Amazon-EMR-Hauptknoten-Sicherheitsgruppe.



So erstellen oder modifizieren Sie eine Amazon-EC2-Sicherheitsgruppe, um eine Verbindung zwischen Amazon Redshift und Amazon EMR zuzulassen

1. Wählen Sie im Amazon-EC2-Dashboard Security Groups (Sicherheitsgruppen) aus. Weitere Informationen finden Sie unter Sicherheitsgruppenregeln im Amazon EC2 EC2-Benutzerhandbuch
2. Wählen Sie Create security group (Sicherheitsgruppe erstellen) aus.
3. Wenn Sie VPC verwenden, wählen Sie die VPC aus, in der sich sowohl Ihr Amazon-Redshift- als auch Ihr Amazon-EMR-Cluster befinden.
4. Fügen Sie eine eingehende Regel hinzu.
    1. Wählen Sie für Type (Typ) die Option Custom TCP (Benutzerdefiniertes TCP) aus.
    2. Wählen Sie für Source (Quelle) die Option Custom (Benutzerdefiniert) aus.
    3. Geben Sie den Namen Ihrer Amazon-Redshift-Sicherheitsgruppe ein.
5. Fügen Sie eine weitere eingehende Regel hinzu.
    1. Wählen Sie unter Type die Option TCP aus.
    2. Geben Sie für Port Range (Portbereich) 9083 ein.

> **Note**
>
> Der Standardport für einen EMR-HMS ist 9083. Wenn Ihr HMS einen anderen Port verwendet, geben Sie diesen in der eingehenden Regel und der Definition des externen Schemas an.

3. Wählen Sie für Source (Quelle) die Option Custom (Benutzerdefiniert) aus.

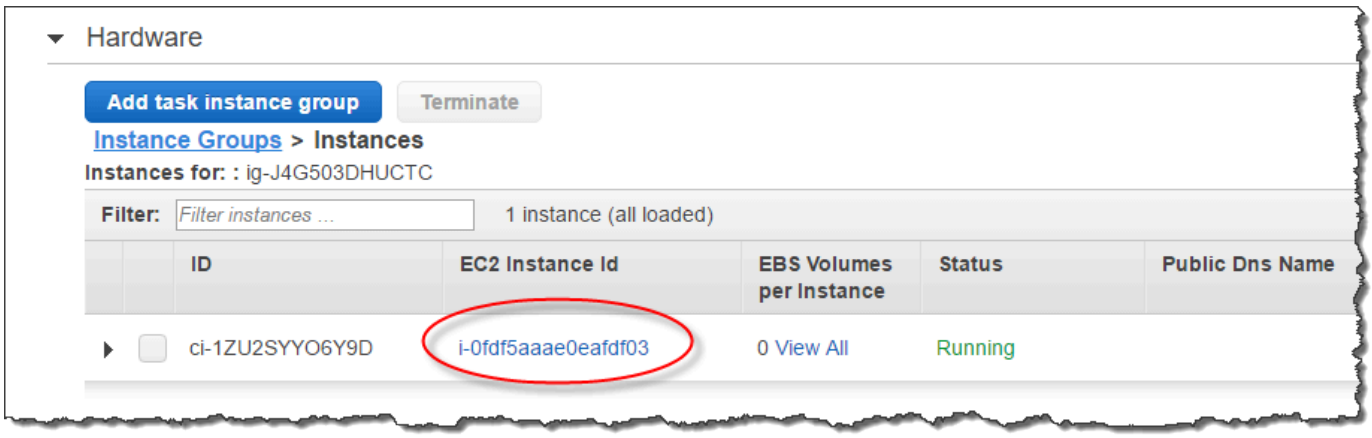
6. Geben Sie einen Namen und eine Beschreibung für die Sicherheitsgruppe ein.
7. Wählen Sie Create security group (Sicherheitsgruppe erstellen) aus.

So fügen Sie die Amazon-EC2-Sicherheitsgruppe, die Sie im vorherigen Verfahren erstellt haben, Ihrem Amazon-Redshift-Cluster hinzu

1. Wählen Sie Ihren Cluster in Amazon Redshift aus.
2. Wählen Sie Properties (Eigenschaften).
3. Rufen Sie die Network and security settings (Netzwerk- und Sicherheitseinstellungen) auf und wählen Sie Edit (Bearbeiten) aus.
4. Wählen Sie in VPC Security Group (VPC-Sicherheitsgruppe) den Namen der neuen Sicherheitsgruppe aus.
5. Wählen Sie Save Changes (Änderungen speichern).

So fügen Sie die Amazon-EC2-Sicherheitsgruppe zu Ihrem Amazon-EMR-Cluster hinzu

1. Wählen Sie Ihren Cluster in Amazon EMR aus. Weitere Informationen finden Sie unter Verwenden von Sicherheitskonfigurationen zur Einrichtung der Clustersicherheit im Amazon-EMR-Verwaltungshandbuch
2. Wählen Sie unter Hardware den Link für den Master-Knoten.
3. Wählen Sie den Link in der Spalte EC2 instance ID (EC2-Instance-ID).



4. Wählen Sie für Actions (Aktionen) die Option Security (Sicherheit), Change security groups (Sicherheitsgruppen ändern) aus.
5. Wählen Sie unter Associated sercurity groups (Zugehörige Sicherheitsgruppen) die neue Sicherheitsgruppe und dann Add security group (Sicherheitsgruppe hinzufügen) aus.
6. Wählen Sie Save (Speichern) aus.

Erstellen externer Tabellen für Redshift Spectrum

Sie erstellen eine externe Tabelle in einem externen Schema. Zum Erstellen externer Tabellen müssen Sie der Eigentümer des externen Schemas oder ein Superuser sein. Mit dem Befehl ALTER SCHEMA können Sie den Besitzer eines externen Schemas ändern. Das folgende Beispiel ändert den Eigentümer des Schemas `spectrum_schema` in `newowner`.

```
alter schema spectrum_schema owner to newowner;
```

Um eine Redshift Spectrum-Abfrage auszuführen, benötigen Sie die folgenden Berechtigungen:

- Nutzungsberechtigung für das Schema
- Berechtigung, temporäre Tabellen in der aktuellen Datenbank zu erstellen

Das folgende Beispiel erteilt der Benutzergruppe `spectrum_schema` Nutzungsberechtigungen für das Schema `spectrumusers`.

```
grant usage on schema spectrum_schema to group spectrumusers;
```

Das folgende Beispiel erteilt der Benutzergruppe `spectrumdb` temporäre Berechtigungen für die Datenbank `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Sie können eine externe Tabelle in Amazon Redshift AWS Glue, Amazon Athena oder einem Apache Hive-Metastore erstellen. Weitere Informationen finden Sie unter Erste Schritte mit AWS Glue im AWS Glue -Entwicklerhandbuch, Erste Schritte im Amazon-Athena-Benutzerhandbuch oder Apache Hive im Amazon-EMR-Entwicklerhandbuch.

Wenn Ihre externe Tabelle in AWS Glue Athena oder einem Hive-Metastore definiert ist, erstellen Sie zunächst ein externes Schema, das auf die externe Datenbank verweist. Anschließend können Sie in Ihrer SELECT-Anweisung auf die externe Tabelle verweisen, indem Sie dem Tabellennamen den Schemanamen voranstellen, ohne dass Sie die Tabelle in Amazon Redshift erstellen müssen. Weitere Informationen finden Sie unter Erstellen externer Schemata für Amazon Redshift Spectrum.

Damit Amazon Redshift Tabellen in anzeigen kann AWS Glue Data Catalog, fügen Sie `glue:GetTable` der Amazon Redshift IAM-Rolle hinzu. Andernfalls wird möglicherweise eine Fehlermeldung wie die folgende angezeigt:

```
RedshiftIamRoleSession is not authorized to perform: glue:GetTable on resource: *;
```

Nehmen Sie beispielsweise an, Sie haben eine externe Tabelle mit der Bezeichnung `lineitem_athena` in einem externen Athena-Katalog definiert. In diesem Fall können Sie ein externes Schema mit der Bezeichnung `athena_schema` definieren und die Tabelle dann mit der folgenden SELECT-Anweisung abfragen.

```
select count(*) from athena_schema.lineitem_athena;
```

Verwenden Sie zur Definition einer externen Tabelle in Amazon Redshift den Befehl CREATE EXTERNAL TABLE. Die Anweisung für die externe Tabelle definiert die Tabellenspalten, das Format Ihrer Datendateien sowie den Speicherort Ihrer Daten in Amazon S3. Redshift Spectrum scannt die Dateien in dem angegebenen Ordner und in allen Unterordnern. Redshift Spectrum ignoriert verborgene Dateien und Dateien, die mit einem Punkt, einem Unterstrich oder einem Hash-Zeichen (. , _ oder #) oder mit einer Tilde (~) enden.

Im folgenden Beispiel wird eine Tabelle namens SALES im externen Amazon-Redshift-Schema namens erstellt `spectrum`. Die Daten befinden sich in Textdateien, die Tabulatoren als Trennzeichen verwenden.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='172000');
```

Um externe Tabellen anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_EXTERNAL_TABLES aus.

Pseudospalten

Standardmäßig erstellt Amazon Redshift externe Tabellen mit den Pseudospalten `$path`, `$size` und `$spectrum_oid`. Wählen Sie die Spalte `$path` aus, um den Pfad zu den Datendateien auf Amazon S3 anzuzeigen, und die Spalte `$size`, um die Größe der Datendateien für jede Zeile anzuzeigen, die von einer Abfrage zurückgegeben wird. Die Spalte `$spectrum_oid` bietet die Möglichkeit, korrelierte Abfragen mit Redshift Spectrum durchzuführen. Ein Beispiel finden Sie unter Beispiel: Durchführen korrelierter Unterabfragen in Redshift Spectrum. Die Spaltennamen `$path`, `$size` und `$spectrum_oid` müssen in doppelte Anführungszeichen eingeschlossen werden. Eine SELECT *-Klausel gibt die Pseudospalten nicht zurück. Sie müssen die Spaltennamen `$path`, `$size` und `$spectrum_oid` explizit in Ihre Abfrage einfügen, wie im folgenden Beispiel gezeigt.

```
select "$path", "$size", "$spectrum_oid"
from spectrum.sales_part where saledate = '2008-12-01';
```

Sie können die Erstellung von Pseudospalten für eine Sitzung deaktivieren, indem Sie den Konfigurationsparameter `spectrum_enable_pseudo_columns` auf `false` festlegen. Weitere Informationen finden Sie unter spectrum_enable_pseudo_columns. Sie können auch nur die Pseudospalte `$spectrum_oid` deaktivieren, indem Sie `enable_spectrum_oid` auf `false`

festlegen. Weitere Informationen finden Sie unter enable_spectrum_oid. Wenn Sie die Pseudospalte `$spectrum_oid` deaktivieren, wird allerdings auch die Unterstützung für korrelierte Abfragen mit Redshift Spectrum deaktiviert.

> **Important**
>
> Bei Auswahl von `$size`, `$path` oder `$spectrum_oid` fallen Gebühren an, da Redshift Spectrum die Datendateien auf Amazon S3 scannt, um die Größe des Ergebnissatzes zu bestimmen. Weitere Informationen finden Sie unter Amazon-Redshift-Preise.

Beispiel für Pseudospalten

Das folgende Beispiel gibt die Gesamtgröße der entsprechenden Datendateien für eine externe Tabelle zurück.

```
select distinct "$path", "$size"
from spectrum.sales_part;

 $path                                                                      | $size
----------------------------------------------------------------------------+-------
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/ |  1616
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/ |  1644
```

Partitionierung externer Redshift-Spectrum-Tabellen

Wenn Sie Ihre Daten partitionieren, können Sie die Menge der von Redshift Spectrum gescannten Daten durch Filterung auf dem Partitionierungsschlüssel begrenzen. Sie können Ihre Daten nach jedem beliebigen Schlüssel partitionieren.

Häufig werden die Daten nach der Zeit partitioniert. So können Sie etwa die Partitionierung nach Jahr, Monat, Datum oder Stunde durchführen. Wenn Sie aus zahlreichen Quellen kommende Daten haben, können Sie auch nach Datenquellen-ID und Datum partitionieren.

Die folgende Vorgehensweise beschreibt die Partitionierung Ihrer Daten.

So partitionieren Sie Ihre Daten:

1. Speichern Sie Ihre Daten gemäß Ihrem Partitionierungsschlüssel in Ordnern in Amazon S3.

Erstellen Sie einen Ordner für jeden Partitionierungswert, und benennen Sie den Ordner mit Partitionsschlüssel und -wert. Zum Beispiel: Wenn Sie nach Datum partitionieren, haben Sie möglicherweise Ordner mit den Bezeichnungen `saledate=2017-04-01`, `saledate=2017-04-02` und so weiter. Redshift Spectrum scannt die Dateien im Partitionierungsordner und in allen Unterordnern. Redshift Spectrum ignoriert verborgene Dateien und Dateien, die mit einem Punkt, einem Unterstrich oder einem Hash-Zeichen (. , _ oder #) oder mit einer Tilde (~) enden.

2. Erstellen Sie eine externe Tabelle, und geben Sie den Partitionierungsschlüssel in der PARTITIONED BY-Klausel an.

   Der Partitionierungsschlüssel darf nicht der Name einer Spalte sein. Als Datentyp kommt SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE oder TIMESTAMP infrage.

3. Fügen Sie die Partitionen hinzu.

   Fügen Sie mit ALTER TABLE … ADD PARTITION jede Partition hinzu, und geben Sie dabei die Partitionierungsspalte und den Schlüsselwert sowie den Speicherort des Partitionierungsordners in Amazon S3 an. Sie können mehrere Partitionen mit einer einzelnen ALTER TABLE … ADD-Anweisung hinzufügen. Im folgenden Beispiel werden Partitionen für `'2008-01'` und `'2008-03'` hinzugefügt.

```
alter table spectrum.sales_part add
partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-01/'
partition(saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-03/';
```

> **Note**
>
> Wenn Sie den AWS Glue Katalog verwenden, können Sie mit einer einzigen ALTER TABLE-Anweisung bis zu 100 Partitionen hinzufügen.

Beispiele für die Datenpartitionierung

In diesem Beispiel erstellen Sie eine externe Tabelle, die von einem einzelnen Partitionsschlüssel partitioniert wird, und eine externe Tabelle, die von zwei Partitionsschlüsseln partitioniert wird.

Die Beispieldaten für dieses Beispiel befinden sich in einem Amazon S3 S3-Bucket, der allen authentifizierten AWS Benutzern Lesezugriff gewährt. Ihr Cluster und Ihre externen Datendateien müssen sich in derselben AWS-Region befinden. Der Beispieldaten-Bucket befindet sich in der Region USA Ost (Nord-Virginia) (us-east-1). Zum Zugriff auf die Daten mit Redshift Spectrum muss sich Ihr Cluster ebenfalls in us-east-1 befinden. Führen Sie zur Auflistung der Ordner in Amazon S3 den folgenden Befehl aus.

```
aws s3 ls s3://redshift-downloads/tickit/spectrum/sales_partition/
```

```
PRE saledate=2008-01/
PRE saledate=2008-03/
PRE saledate=2008-04/
PRE saledate=2008-05/
PRE saledate=2008-06/
PRE saledate=2008-12/
```

Wenn Sie noch kein externes Schema haben, führen Sie den folgenden Befehl aus. Ersetzen Sie Ihre (IAM) -Rolle durch den Amazon Resource Name AWS Identity and Access Management (ARN).

```
create external schema spectrum
from data catalog
database 'spectrumdb'
iam_role 'arn:aws:iam::123456789012:role/myspectrumrole'
create external database if not exists;
```

Beispiel 1: Partitionierung mit einem einzelnen Partitionsschlüssel

Im folgenden Beispiel erstellen Sie eine externe Tabelle, die nach Monaten partitioniert ist.

Führen Sie zur Erstellung einer nach Monat partitionierten Tabelle den folgenden Befehl aus.

```
create external table spectrum.sales_part(
salesid integer,
listid integer,
sellerid integer,
```

```
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (saledate char(10))
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales_partition/'
table properties ('numRows'='172000');
```

Führen Sie zum Hinzufügen der Partitionen den folgenden ALTER TABLE-Befehl aus.

```
alter table spectrum.sales_part add
partition(saledate='2008-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/'

partition(saledate='2008-03')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/'

partition(saledate='2008-04')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-04/';
```

Führen Sie die folgende Abfrage aus, um Daten aus der partitionierten Tabelle auszuwählen.

```
select top 5 spectrum.sales_part.eventid, sum(spectrum.sales_part.pricepaid)
from spectrum.sales_part, event
where spectrum.sales_part.eventid = event.eventid
  and spectrum.sales_part.pricepaid > 30
  and saledate = '2008-01'
group by spectrum.sales_part.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
   4124 | 21179.00
   1924 | 20569.00
   2294 | 18830.00
   2260 | 17669.00
```

```
   6032 | 17265.00
```

Um externe Tabellenpartitionen anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_EXTERNAL_PARTITIONS aus.

```
select schemaname, tablename, values, location from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values      | location

-----------+------------+-------------
+-------------------------------------------------------------------------
spectrum   | sales_part | ["2008-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-03"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
```

Beispiel 2: Partitionierung mit mehreren Partitionsschlüsseln

Führen Sie zur Erstellung einer nach `date` und `eventid` partitionierten externen Tabelle den folgenden Befehl aus.

```
create external table spectrum.sales_event(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (salesmonth char(10), event integer)
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/salesevent/'
table properties ('numRows'='172000');
```

Führen Sie zum Hinzufügen der Partitionen den folgenden ALTER TABLE-Befehl aus.

```
alter table spectrum.sales_event add
partition(salesmonth='2008-01', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=101/'

partition(salesmonth='2008-01', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=102/'

partition(salesmonth='2008-01', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=103/'

partition(salesmonth='2008-02', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=101/'

partition(salesmonth='2008-02', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=102/'

partition(salesmonth='2008-02', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=103/'

partition(salesmonth='2008-03', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=101/'

partition(salesmonth='2008-03', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=102/'

partition(salesmonth='2008-03', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=103/';
```

Führen Sie die folgende Abfrage aus, um Daten aus der partitionierten Tabelle auszuwählen.

```
select spectrum.sales_event.salesmonth, event.eventname,
 sum(spectrum.sales_event.pricepaid)
```

```
from spectrum.sales_event, event
where spectrum.sales_event.eventid = event.eventid
  and salesmonth = '2008-02'
 and (event = '101'
 or event = '102'
 or event = '103')
group by event.eventname, spectrum.sales_event.salesmonth
order by 3 desc;
```

```
salesmonth | eventname       | sum
-----------+-----------------+--------
2008-02    | The Magic Flute | 5062.00
2008-02    | La Sonnambula   | 3498.00
2008-02    | Die Walkure     |  534.00
```

Mapping externer Tabellenspalten zu ORC-Spalten

Sie verwenden externe Amazon-Redshift-Spectrum-Tabellen zur Abfrage von Dateien im ORC-Format. Das ORC (Optimized Row Columnar) Format ist ein in Spalten organisiertes Speicherdateiformat, dass genestete Datenstrukturen unterstützt. Weitere Informationen zur Abfrage verschachtelter Daten finden Sie unter Abfragen verschachtelter Daten mit Amazon Redshift Spectrum.

Wenn Sie eine externe Tabelle erstellen, die auf Daten in einer ORC-Datei verweist, weisen Sie jede Spalte in der externen Tabelle einer Spalte in den ORC-Daten zu. Verwenden Sie dazu eines der folgenden verfahren.

- Mapping nach Position
- Mapping nach Spaltenname

Das Mapping nach Spaltenname ist das Standardverfahren.

Mapping nach Position

Beim Positions-Mapping wird die erste in der externen Tabelle definierte Spalte der ersten Spalte in der ORC-Datendatei zugewiesen, die zweite der zweiten usw. Das Mapping nach Position erfordert, dass die Reihenfolge der Spalten in der externen Tabelle und in der ORC-Datei einander entsprechen. Wenn die Reihenfolge der Spalten nicht übereinstimmt, können Sie die Spalten nach Name zuweisen.

Important

In früheren Versionen verwendete Redshift Spectrum standardmäßig das Positions-Mapping. Wenn Sie für vorhandene Tabellen mit dem Positions-Mapping fortfahren müssen, stellen Sie die Tabelleneigenschaft `orc.schema.resolution` auf `position`, wie das folgende Beispiel zeigt.

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='position');
```

Beispielsweise ist die Tabelle `SPECTRUM.ORC_EXAMPLE` wie folgt definiert.

```
create external table spectrum.orc_example(
int_col int,
float_col float,
nested_col struct<
  "int_col" : int,
  "map_col" : map<int, array<float >>
   >
) stored as orc
location 's3://example/orc/files/';
```

Die Tabellenstruktur kann wie folgt abstrahiert werden.

```
• 'int_col' : int
• 'float_col' : float
• 'nested_col' : struct
   o 'int_col' : int
   o 'map_col' : map
      - key : int
      - value : array
         - value : float
```

Die zugrunde liegende ORC-Datei hat die folgende Dateistruktur.

```
• ORC file root(id = 0)
   o 'int_col' : int (id = 1)
   o 'float_col' : float (id = 2)
   o 'nested_col' : struct (id = 3)
```

```
      - 'int_col' : int (id = 4)
      - 'map_col' : map (id = 5)
         - key : int (id = 6)
         - value : array (id = 7)
            - value : float (id = 8)
```

In diesem Beispiel können Sie jede Spalte der externen Tabelle einer Spalte in der ORC-Datei streng nach Position zuweisen. Nachfolgend sehen Sie das Mapping.

Spaltenname der externen Tabelle	ORC-Spalten-ID	Name der ORC-Spalte
int_col	1	int_col
float_col	2	float_col
nested_col	3	nested_col
nested_col.int_col	4	int_col
nested_col.map_col	5	map_col
nested_col.map_col.key	6	N/A
nested_col.map_col.value	7	N/A
nested_col.map_col.value.item	8	N/A

Mapping nach Spaltenname

Beim Namens-Mapping weisen Sie Spalten in einer externen Tabelle benannten Spalten in ORC-Dateien auf derselben Ebene und demselben Namen zu.

Zum Beispiel: Angenommen, Sie möchten die Tabelle aus dem vorherigen Beispiel, `SPECTRUM.ORC_EXAMPLE`, einer ORC-Datei mit der folgenden Dateistruktur zuweisen.

```
• ORC file root(id = 0)
   o 'nested_col' : struct (id = 1)
      - 'map_col' : map (id = 2)
         - key : int (id = 3)
         - value : array (id = 4)
```

```
            - value : float (id = 5)
      - 'int_col' : int (id = 6)
   o 'int_col' : int (id = 7)
   o 'float_col' : float (id = 8)
```

Mit Positionierungszuweisung versucht Redshift Spectrum das folgende Mapping.

Spaltenname der externen Tabelle	ORC-Spalten-ID	Name der ORC-Spalte
int_col	1	struct
float_col	7	int_col
nested_col	8	float_col

Wenn Sie eine Tabelle mit dem vorherigen Positions-Mapping abfragen, schlägt der Befehl SELECT bei der Typenvalidierung fehl, da die Strukturen unterschiedlich sind.

Sie weisen dieselbe externe Tabelle beiden in den vorherigen Beispielen gezeigten Dateistrukturen mithilfe des Spaltennamen-Mappings zu. Die Tabellenspalten `int_col`, `float_col` und `nested_col` werden nach Spaltenname zu Spalten mit denselben Namen in der ORC-Datei zugewiesen. Die Spalte mit der Bezeichnung `nested_col` in der externen Tabelle ist eine `struct`-Spalte mit Unterspalten mit den Bezeichnungen `map_col` und `int_col`. Die Unterspalten werden ebenfalls korrekt den entsprechenden Spalten in der ORC-Datei nach Spaltenname zugewiesen.

Erstellen externer Tabellen für in Apache Hudi verwaltete Daten

Um Daten im Apache-Hudi-Format CoW (Copy on Write) abzufragen, können Sie externe Amazon-Redshift-Spectrum-Tabellen verwenden. Eine Hudi-Copy-On-Write-Tabelle ist eine Sammlung von Apache-Parquet-Dateien, die in Amazon S3 gespeichert sind. Sie können Copy-on-Write-Tabellen (CoW) in den Apache-Hudi-Versionen 0.5.2, 0.6.0, 0.7.0, 0.8.0, 0.9.0, 0.10.0, 0.10.1, 0.11.0 und 0.11.1 lesen, die mit Einfüge-, Lösch- und Upsert-Schreiboperationen erstellt und geändert werden. Bootstrap-Tabellen werden beispielsweise nicht unterstützt. Weitere Informationen finden Sie unter Copy On Write Table in der Open-Source-Dokumentation von Apache Hudi.

Wenn Sie eine externe Tabelle erstellen, die auf Daten im Hudi-CoW-Format verweist, weisen Sie jede Spalte in der externen Tabelle einer Spalte in den Hudi-Daten zu. Das Mapping erfolgt nach Spalte.

Die DDL-Anweisungen (Data Definition Language) für die partitionierten und nicht partitionierten Hudi-Tabellen ähneln den Anweisungen für andere Apache-Parquet-Dateiformate. Für Hudi-Tabellen definieren Sie `INPUTFORMAT` als `org.apache.hudi.hadoop.HoodieParquetInputFormat`. Der `LOCATION`-Parameter muss auf den Basisordner der Hudi-Tabelle verweisen, der den `.hoodie`-Ordner enthält, der erforderlich ist, um die Hudi-Commit-Timeline zu erstellen. In manchen Fällen kann der SELECT-Vorgang auf einer Hudi-Tabelle fehlschlagen. Sie erhalten dann die Nachricht No valid Hudi commit timeline found (Keine gültige Hudi-Commit-Timeline gefunden). Wenn dies der Fall ist, sollten Sie überprüfen, ob sich der `.hoodie`-Ordner am richtigen Ort befindet und die gültige Hudi-Commit-Timeline enthält.

> **Note**
>
> Das Apache-Hudi-Format wird nur unterstützt, wenn Sie einen verwenden AWS Glue Data Catalog. Es wird nicht unterstützt, wenn Sie einen Apache-Hive-Metastore als externen Katalog verwenden.

Die DDL für die Definition einer nicht partitionierten Tabelle hat das folgende Format.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://s3-bucket/prefix'
```

Die DDL für die Definition einer partitionierten Tabelle hat das folgende Format.

```
CREATE EXTERNAL TABLE tbl_name (columns)
PARTITIONED BY(pcolumn1 pcolumn1-type[,...])
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://s3-bucket/prefix'
```

Um Partitionen zu einer partitionierten Hudi-Tabelle hinzuzufügen, führen Sie den Befehl ALTER TABLE ADD PARTITION aus, wobei der `LOCATION`-Parameter auf den Amazon-S3-Unterordner mit den Dateien verweist, die zur Partition gehören.

Die DDL zum Hinzufügen von Partitionen hat das folgende Format.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION 's3://s3-bucket/prefix/partition-path'
```

Erstellen externer Tabellen für in Delta Lake verwaltete Daten

Um Daten in Delta-Lake-Tabellen abzufragen, können Sie externe Tabellen von Amazon Redshift Spectrum verwenden.

Um aus Redshift Spectrum auf eine Delta-Lake-Tabelle zuzugreifen, generieren Sie vor der Abfrage ein Manifest. Ein Delta-Lake-Manifest enthält eine Liste von Dateien, aus denen ein konsistenter Snapshot in der Delta-Lake-Tabelle besteht. In einer partitionierten Tabelle gibt es ein Manifest pro Partition. Eine Delta-Lake-Tabelle ist eine Sammlung von Apache-Parquet-Dateien, die in Amazon S3 gespeichert sind. Weitere Informationen finden Sie unter Delta Lake in der Open-Source-Dokumentation von Delta Lake.

Wenn Sie eine externe Tabelle erstellen, die auf Daten in Delta-Lake-Tabellen verweist, weisen Sie jede Spalte in der externen Tabelle einer Spalte in der Delta-Lake-Tabelle zu. Das Mapping erfolgt nach Spaltenname.

Die DDL für partitionierte und nicht partitionierte Delta-Lake-Tabellen ähnelt der für andere Apache-Parquet-Dateiformate. Für Delta-Lake-Tabellen definieren Sie `INPUTFORMAT` als `org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat` und `OUTPUTFORMAT` als `org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat`. Der `LOCATION`-Parameter muss auf den Manifestordner im Basisordner der Tabelle verweisen. Gründe für ein mögliches Fehlschlagen des SELECT-Vorgangs bei Delta-Lake-Tabellen finden Sie unter Einschränkungen und Fehlerbehebung bei Delta-Lake-Tabellen.

Die DDL für die Definition einer nicht partitionierten Tabelle hat das folgende Format.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://s3-bucket/prefix/_symlink_format_manifest'
```

Die DDL für die Definition einer partitionierten Tabelle hat das folgende Format.

```
CREATE EXTERNAL TABLE tbl_name (columns)
PARTITIONED BY(pcolumn1 pcolumn1-type[,...])
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://s3-bucket>/prefix/_symlink_format_manifest'
```

Um Partitionen zu einer partitionierten Delta-Lake-Tabelle hinzuzufügen, führen Sie den Befehl ALTER TABLE ADD PARTITION aus, wobei der LOCATION-Parameter auf den Amazon-S3-Unterordner verweist, der das Manifest für die Partition enthält.

Die DDL zum Hinzufügen von Partitionen hat das folgende Format.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION
's3://s3-bucket/prefix/_symlink_format_manifest/partition-path'
```

Sie können auch eine DDL ausführen, die direkt auf die Data-Lake-Manifestdatei verweist.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION
's3://s3-bucket/prefix/_symlink_format_manifest/partition-path/manifest'
```

Einschränkungen und Fehlerbehebung bei Delta-Lake-Tabellen

Beachten Sie Folgendes, wenn Sie Delta-Lake-Tabellen aus Redshift Spectrum abfragen:

- Wenn ein Manifest auf einen Snapshot oder eine Partition verweist, der/die nicht mehr vorhanden ist, schlagen Abfragen fehl, bis ein neues gültiges Manifest generiert wurde. Der Grund hierfür kann beispielsweise ein VACUUM-Vorgang in einer zugrundeliegenden Tabelle sein.
- Delta-Lake-Manifeste bieten nur Konsistenz auf Partitionsebene.

Die folgende Tabelle zeigt einige Gründe für Fehler bei der Abfrage einer Data-Lake-Tabelle.

Fehlermeldung	Möglicher Grund
Delta Lake manifest in bucket s3-bucket-1 cannot contain entries in bucket s3-bucket-2.	Die Manifesteinträge verweisen auf Dateien in einem anderen Amazon-S3-Bucket als dem angegebenen.
Delta Lake files are expected to be in the same folder.	Die Manifesteinträge verweisen auf Dateien mit einem anderen Amazon-S3-Präfix als dem angegebenen.
File filename listed in Delta Lake manifest manifest-path was not found.	Eine im Manifest aufgelistete Datei wurde nicht in Amazon S3 gefunden.
Error fetching Delta Lake manifest.	Das Manifest wurde nicht in Amazon S3 gefunden.
Invalid S3 Path.	Ein Eintrag in der Manifestdatei ist kein gültiger Amazon-S3-Pfad oder die Manifestdatei ist beschädigt.

Verwenden von Apache-Iceberg-Tabellen mit Amazon Redshift

Sie können Redshift Spectrum oder Redshift Serverless verwenden, um Apache-Iceberg-Tabellen abzufragen, die im AWS Glue Data Catalog katalogisiert sind. Apache Iceberg ist ein Open-Source-Tabellenformat für Data Lakes. Weitere Informationen finden Sie unter Apache Iceberg in der Apache-Iceberg-Dokumentation.

Amazon Redshift bietet Transaktionskonsistenz für Abfragen von Apache-Iceberg-Tabellen. Sie können die Daten in Ihren Tabellen mithilfe von ACID-konformen Services (Atomizität, Konsistenz, Isolation, Dauerhaftigkeit) wie Amazon Athena und Amazon EMR bearbeiten, während Sie Abfragen mit Amazon Redshift ausführen. Amazon Redshift kann die in den Apache-Iceberg-Metadaten gespeicherten Tabellenstatistiken verwenden, um Abfragepläne zu optimieren und Dateiscans während der Abfrageverarbeitung zu reduzieren. Mit Amazon Redshift SQL können Sie Redshift-Tabellen mit Data-Lake-Tabellen verbinden.

So führen Sie die ersten Schritte für Iceberg-Tabellen mit Amazon Redshift aus:

1. Erstellen Sie mit einem kompatiblen Service wie Amazon Athena oder Amazon EMR eine Apache Iceberg-Tabelle in einer AWS Glue Data Catalog Datenbank. Informationen zum Erstellen einer Iceberg-Tabelle mit Athena finden Sie unter Verwenden von Apache-Iceberg-Tabellen im Amazon-Athena-Benutzerhandbuch.
2. Erstellen Sie einen Amazon-Redshift-Cluster oder eine Redshift-Serverless-Arbeitsgruppe mit einer zugehörigen IAM-Rolle, die den Zugriff auf Ihren Data Lake ermöglicht. Weitere Informationen zum Erstellen von Clustern oder Arbeitsgruppen finden Sie unter Bereitgestellte Amazon-Redshift-Cluster und Redshift Serverless im Handbuch Erste Schritte mit Amazon Redshift.
3. Stellen Sie mithilfe des Abfrage-Editors v2 oder eines SQL-Clients eines Drittanbieters eine Verbindung mit Ihrem Cluster oder Ihrer Arbeitsgruppe her. Informationen zum Herstellen einer Verbindung mit dem Abfrage-Editor v2 finden Sie unter Herstellen einer Verbindung zu einem Amazon Redshift Data Warehouse mithilfe von SQL-Client-Tools im Amazon Redshift Management Guide.
4. Erstellen Sie in Ihrer Amazon-Redshift-Datenbank ein externes Schema für eine bestimmte Datenkatalogdatenbank, die Ihre Iceberg-Tabellen enthält. Informationen zum Erstellen eines externen Schemas finden Sie unter Erstellen externer Schemata für Amazon Redshift Spectrum.
5. Führen Sie SQL-Abfragen aus, um auf die Iceberg-Tabellen in dem von Ihnen erstellten externen Schema zuzugreifen.

Überlegungen zur Verwendung von Apache-Iceberg-Tabellen mit Amazon Redshift

Bei der Verwendung von Amazon Redshift mit Iceberg-Tabellen sollten Sie Folgendes berücksichtigen:

- Unterstützung der Iceberg-Version – Amazon Redshift unterstützt die Ausführung von Abfragen für die folgenden Versionen von Iceberg-Tabellen:
  - Version 1 definiert, wie große Analysetabellen mithilfe unveränderlicher Datendateien verwaltet werden.
  - Version 2 fügt die Möglichkeit hinzu, Update- und Löschvorgänge auf Zeilenebene zu unterstützen, während die vorhandenen Datendateien unverändert bleiben und die Änderungen der Tabellendaten mithilfe von Löschdateien verarbeitet werden.

Den Unterschied zwischen Tabellen der Version 1 und Version 2 finden Sie unter Formatversionsänderungen in der Apache-Iceberg-Dokumentation.

- Nur Abfragen – Amazon Redshift unterstützt den schreibgeschützten Zugriff auf Apache-Iceberg-Tabellen. Es unterstützt transaktionskonsistente Auswahlabfragen. Sie können einen Service wie Amazon Athena verwenden, um das Schema von Iceberg-Tabellen in AWS Glue Data Catalog zu definieren und zu aktualisieren.
- Hinzufügen von Partitionen – Sie müssen Partitionen für Ihre Apache-Iceberg-Tabellen nicht manuell hinzufügen. Neue Partitionen in Apache-Iceberg-Tabellen werden automatisch von Amazon Redshift erkannt und es ist kein manueller Vorgang erforderlich, um Partitionen in der Tabellendefinition zu aktualisieren. Alle Änderungen der Partitionsspezifikation werden auch automatisch auf Ihre Abfragen angewendet, ohne dass der Benutzer eingreifen muss.
- Erfassen von Iceberg-Daten in Amazon Redshift – Sie können die Befehle INSERT INTO oder CREATE TABLE AS verwenden, um Daten aus Ihrer Iceberg-Tabelle in eine lokale Amazon-Redshift-Tabelle zu importieren. Derzeit können Sie den Befehl COPY nicht verwenden, um den Inhalt einer Apache-Iceberg-Tabelle in eine lokale Amazon-Redshift-Tabelle zu importieren.
- Materialisierte Ansichten – Sie können materialisierte Ansichten für Apache-Iceberg-Tabellen wie jede andere externe Tabelle in Amazon Redshift erstellen. Die gleichen Überlegungen wie für andere Data-Lake-Tabellenformate gelten auch für Apache-Iceberg-Tabellen. Inkrementelle Updates, automatische Aktualisierungen, automatisches Umschreiben von Abfragen und automatische MVs in Data-Lake-Tabellen werden derzeit nicht unterstützt.
- AWS Lake Formation feinkörnige Zugriffskontrolle — Amazon Redshift unterstützt eine AWS Lake Formation differenzierte Zugriffskontrolle für Apache Iceberg-Tabellen.
- Benutzerdefinierte Datenverarbeitungsparameter – Amazon Redshift unterstützt benutzerdefinierte Datenverarbeitungsparameter für Apache-Iceberg-Tabellen. Sie verwenden benutzerdefinierte Datenverarbeitungsparameter für vorhandene Dateien, um die Daten, die in externen Tabellen abgefragt werden, anzupassen und Scanfehler zu vermeiden. Diese Parameter bieten die Möglichkeit, Diskrepanzen zwischen dem Tabellenschema und den tatsächlichen Daten in Dateien zu bearbeiten. Sie können benutzerdefinierte Datenverarbeitungsparameter auch für Apache-Iceberg-Tabellen verwenden.
- Datenfreigabe – Die Datenfreigabe von Amazon Redshift unterstützt derzeit keine Data Lake-Tabellen, einschließlich Apache-Iceberg-Tabellen.
- Zeitreiseabfragen – Zeitreiseabfragen werden derzeit mit Apache-Iceberg-Tabellen nicht unterstützt.

- Preise – Wenn Sie von einem Cluster aus auf Iceberg-Tabellen zugreifen, werden Ihnen die Redshift-Spectrum-Preise berechnet. Wenn Sie von einer Arbeitsgruppe aus auf Iceberg-Tabellen zugreifen, werden Ihnen die Redshift-Serverless-Preise berechnet. Weitere Informationen zu Preisen von Redshift Spectrum und Redshift Serverless finden Sie unter Amazon Redshift – Preise.

Themen

Mit Apache-Iceberg-Tabellen unterstützte Datentypen

Amazon Redshift kann Iceberg-Tabellen abfragen, welche die folgenden Datentypen enthalten:

```
binary
boolean
date
decimal
double
float
int
list
long
map
string
struct
timestamp without time zone
```

Weitere Informationen zu Iceberg-Datentypen finden Sie unter Schemata für Iceberg in der Apache-Iceberg-Dokumentation.

Die folgende Tabelle zeigt das Verhältnis zwischen Amazon-Redshift-Datentypen und Datentypen von Iceberg-Tabellen.

Iceberg-Typ	Amazon-Redshift-Typ	Hinweise
`boolean`	`boolean`	
-	`tinyint`	Wird für Iceberg-Tabellen in Amazon Redshift nicht unterstützt.

Iceberg-Typ	Amazon-Redshift-Typ	Hinweise
-	`smallint`	Wird für Iceberg-Tabellen in Amazon Redshift nicht unterstützt.
`int`	`int`	In SQL-Anweisungen von Amazon Redshift lautet dieser Typ `INTEGER`.
`long`	`bigint`	
`double`	`double`	
`float`	`float`	
`decimal(P, S)`	`decimal(P, S)`	`P` ist Präzision, `S` ist Skalieren.
-	`char`	Wird für Iceberg-Tabellen in Redshift Spectrum nicht unterstützt.
`string`	`string`	In SQL-Anweisungen von Amazon Redshift lautet dieser Typ `VARCHAR`.
`binary`	`binary`	
`date`	`date`	
`time`	-	
`timestamp`	`timestamp`	
`timestamptz`	-	Der Typ timestamptz wird derzeit in Redshift Spectrum nicht unterstützt.
`list<E>`	`array`	
`map<K,V>`	`map`	
`struct<...>`	`struct`	

Iceberg-Typ	Amazon-Redshift-Typ	Hinweise
`fixed(L)`	-	Der Typ `fixed(L)` wird derzeit in Redshift Spectrum nicht unterstützt.

Weitere Informationen zu Datentypen in Amazon Redshift finden Sie unter Datentypen.

Verbesserung der Amazon-Redshift-Spectrum-Abfrageleistung

Sehen Sie sich den Abfrageplan an, um zu ermitteln, welche Schritte auf die Amazon-Redshift-Spectrum-Ebene übertragen wurden.

Die folgenden Schritte beziehen sich auf die Redshift Spectrum-Abfrage:

- S3 Seq Scan
- S3 HashAggregate
- S3 Query Scan
- Seq-Scan PartitionInfo
- Partition Loop

Das folgende Beispiel zeigt den Abfrageplan für eine Abfrage, die eine externe mit einer lokalen Tabelle verbindet. Beachten Sie die HashAggregate Schritte S3 Seq Scan und S3, die für die Daten auf Amazon S3 ausgeführt wurden.

```
explain
select top 10 spectrum.sales.eventid, sum(spectrum.sales.pricepaid)
from spectrum.sales, event
where spectrum.sales.eventid = event.eventid
and spectrum.sales.pricepaid > 30
group by spectrum.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader


              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


                    Sort Key: sum(sales.derived_col2)


                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99 rows=200
 width=31)


                          ->  XN Hash Join DS_BCAST_INNER  (cost=3119.97..1055769620.49
 rows=200000 width=31)


                                Hash Cond: ("outer".derived_col1 = "inner".eventid)


                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
 rows=200000 width=31)


                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
 rows=200000 width=16)


                                            ->  S3 Seq Scan spectrum.sales
 location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
 (cost=0.00..2150.00 rows=172000 width=16)
```

```
                                                Filter: (pricepaid > 30.00)

                                ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)

                                      ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Beachten Sie die folgenden Elemente in dem Abfrageplan:

- Der Knoten `S3 Seq Scan` zeigt, dass der Filter `pricepaid > 30.00` auf der Redshift Spectrum-Ebene verarbeitet wurde.

  Ein Filterknoten unter dem Knoten `XN S3 Query Scan` zeigt die Prädikatverarbeitung in Amazon Redshift auf den von der Redshift-Spectrum-Ebene zurückgegebenen Daten an.
- Der Knoten `S3 HashAggregate` verweist auf die Aggregierung auf der Redshift Spectrum-Ebene für die Group By-Klausel (`group by spectrum.sales.eventid`).

Nachfolgend finden Sie einige Möglichkeiten zur Verbesserung der Leistung von Redshift Spectrum:

- Verwenden Sie im Apache Parquet-Format formatierte Datendateien. Parquet speichert Daten im Spaltenformat, so dass Redshift Spectrum nicht benötigte Spalten aus dem Scan ausschließen kann. Wenn die Daten das Textfile-Format aufweisen, muss Redshift Spectrum die gesamte Datei scannen.
- Verwenden Sie mehrere Dateien zur Optimierung für die parallele Verarbeitung. Halten Sie die Größe Ihrer Dateien größer als 64 MB. Vermeiden Sie Datenverzerrungen, indem Sie die Dateien annähernd gleich groß halten. Informationen zu Apache Parquet-Dateien und Konfigurationsempfehlungen finden Sie unter Dateiformat: Konfigurationen in der Apache Parquet-Dokumentation.
- Verwenden Sie in Ihren Abfragen so wenige Spalten wie möglich.
- Speichern Sie Ihre großen Faktentabellen in Amazon S3 und Ihre häufig verwendeten und kleineren Dimensionstabellen in Ihrer lokalen Amazon-Redshift-Datenbank.
- Aktualisieren Sie die Statistiken für externe Tabellen durch Einstellung des TABLE PROPERTIES-numRows-Parameters. Verwenden Sie CREATE EXTERNAL TABLE oder ALTER TABLE, um den numRows-Parameter TABLE PROPERTIES so einzustellen, dass er die Anzahl der Zeilen in der Tabelle widerspiegelt. Amazon Redshift analysiert keine externen Tabellen, um die

Tabellenstatistiken zu generieren, die der Abfrageoptimierer verwendet, um einen Abfrageplan zu erstellen. Wenn für eine externe Tabelle keine Tabellenstatistiken festgelegt sind, generiert Amazon Redshift einen Abfrageausführungsplan. Amazon Redshift generiert diesen Plan unter der Annahme, dass externe Tabellen die größeren Tabellen und lokale Tabellen die kleineren Tabellen sind.

- Der Amazon-Redshift-Abfrageplaner pusht Prädikate und Aggregationen nach Möglichkeit auf die Redshift-Spectrum-Abfrageebene. Wenn große Mengen von Daten von Amazon S3 zurückgegeben werden, ist die Verarbeitung durch die Ressourcen Ihres Clusters begrenzt. Redshift Spectrum wird für die Verarbeitung sehr großer Anfragen automatisch skaliert. Daher steigt Ihre allgemeine Leistung, wenn Sie die Verarbeitung auf die Redshift Spectrum-Ebene verschieben können.
- Schreiben Sie Ihre Abfragen so, dass sie Filter und Aggregierungen verwenden, die die Verschiebung auf die Redshift Spectrum-Ebene ermöglichen.

  Nachfolgend finden Sie einige Beispiele für Operationen, die auf die Redshift Spectrum-Ebene verschoben werden können:

  - GROUP BY-Klauseln
  - Vergleichsbedingungen und Musterabgleichbedingungen, wie etwa LIKE.
  - Aggregierungsfunktionen, wie etwa COUNT, SUM, AVG, MIN und MAX.
  - Zeichenfolgefunktionen.

  Operationen, die nicht auf die Redshift Spectrum-Ebene verschoben werden können, sind etwa DISTINCT und ORDER BY.
- Verwenden Sie Partitionen, um die Menge der gescannten Daten zu begrenzen. Partitionieren Sie Ihre Daten auf der Grundlage der häufigsten Abfrageprädikate, und verkürzen Sie dann die Partitionen durch die Filterung auf Partitionierungsspalten. Weitere Informationen finden Sie unter Partitionierung externer Redshift-Spectrum-Tabellen.

  Fragen Sie SVL_S3PARTITION ab, um alle Partitionen und qualifizierte Partitionen anzuzeigen.
- Verwenden Sie AWS Glue den Statistikgenerator, um Statistiken auf Spaltenebene für AWS Glue Data Catalog Tabellen zu berechnen. Sobald Statistiken für Tabellen im Datenkatalog AWS Glue generiert wurden, verwendet Amazon Redshift Spectrum diese Statistiken automatisch, um den Abfrageplan zu optimieren. Weitere Informationen zur Berechnung von Statistiken auf Spaltenebene mithilfe von AWS Glue finden Sie unter Arbeiten mit Spaltenstatistiken im AWS Glue Entwicklerhandbuch.

Einstellung der Datenverarbeitungsoptionen

Sie können Tabellenparameter einstellen, wenn Sie externe Tabellen erstellen, um die in externen Tabellen abgefragten Daten anzupassen. Andernfalls können Scanfehler auftreten. Weitere Informationen finden Sie unter TABLE PROPERTIES in CREATE EXTERNAL TABLE. Beispiele finden Sie unter Beispiele für die Datenverarbeitung. Eine Liste von Fehlern finden Sie unter SVL_SPECTRUM_SCAN_ERROR.

Sie können die folgenden TABLE PROPERTIES einstellen, wenn Sie externe Tabellen erstellen, um die Eingabebehandlung für Daten anzugeben, die in externen Tabellen abgefragt werden.

- `column_count_mismatch_handling`, um festzustellen, ob die Datei weniger oder mehr Werte für eine Zeile enthält als die in der externen Tabellendefinition angegebene Anzahl von Spalten.
- `invalid_char_handling`, um die Eingabebehandlung für ungültige Zeichen in Spalten anzugeben, die VARCHAR-, CHAR- und String-Daten enthalten. Wenn Sie REPLACE für `invalid_char_handling` angeben, können Sie das zu verwendende Ersatzzeichen angeben.
- `numeric_overflow_handling`, um die Verarbeitung von Umwandlungsüberläufen in Spalten anzugeben, die Ganzzahl- und Dezimaldaten enthalten.
- `surplus_bytes_handling`, um die Eingabebehandlung für überschüssige Byte in Spalten anzugeben, die VARBYTE-Daten enthalten.
- `surplus_char_handling`, um die Eingabebehandlung für überschüssige Zeichen in Spalten anzugeben, die VARCHAR-, CHAR- und String-Daten enthalten.

Sie können eine Konfigurationsoption festlegen, um Abfragen abzubrechen, die eine maximale Anzahl von Fehlern überschreiten. Weitere Informationen finden Sie unter spectrum_query_maxerror.

Beispiel: Durchführen korrelierter Unterabfragen in Redshift Spectrum

Sie können korrelierte Unterabfragen in Redshift Spectrum durchführen. Die Pseudospalte `$spectrum_oid` bietet die Möglichkeit, korrelierte Abfragen mit Redshift Spectrum durchzuführen. Um eine korrelierte Unterabfrage durchzuführen, muss die Pseudospalte `$spectrum_oid` aktiviert sein, wird aber nicht in der SQL-Anweisung angezeigt. Weitere Informationen finden Sie unter Pseudospalten.

Informationen zum Erstellen des externen Schemas und der externen Tabellen für dieses Beispiel finden Sie unter Erste Schritte mit Amazon Redshift Spectrum.

Im Folgenden sehen Sie ein Beispiel für eine korrelierte Unterabfrage in Redshift Spectrum.

```
select *
from myspectrum_schema.sales s
where exists
( select *
from myspectrum_schema.listing l
where l.listid = s.listid )
order by salesid
limit 5;
```

```
salesid  listid   sellerid   buyerid   eventid   dateid  qtysold  pricepaid
 commission   saletime
1        1        36861      21191     7872      1875    4        728         109.2
    2008-02-18 02:36:48
2        4        8117       11498     4337      1983    2        76          11.4
    2008-06-06 05:00:16
3        5        1616       17433     8647      1983    2        350         52.5
    2008-06-06 08:26:17
4        5        1616       19715     8647      1986    1        175         26.25
    2008-06-09 08:38:52
5        6        47402      14115     8240      2069    2        154         23.1
    2008-08-31 09:17:02
```

Überwachen von Metriken in Amazon Redshift Spectrum

Sie können die Amazon-Redshift-Spectrum-Abfragen mithilfe der folgenden Systemansichten überwachen:

- SVL_S3QUERY

  Verwenden Sie die Ansicht SVL_S3QUERY um Details zu Redshift Spectrum-Abfragen (S3-Abfragen) auf Segment- und Knotenslice-Ebene zu erhalten.
- SVL_S3QUERY_SUMMARY

  Verwenden Sie die Ansicht SVL_S3QUERY_SUMMARY zum Erhalt einer Zusammenfassung aller Amazon-Redshift-Spectrum-Abfragen (S3-Abfragen), die auf dem System ausgeführt wurden.

Nachfolgend finden Sie einige Aspekte, auf die Sie in SVL_S3QUERY_SUMMARY achten sollten:

- Die Anzahl der Dateien, die von der Redshift Spectrum-Abfrage verarbeitet wurden.
- Die Anzahl der von Amazon S3 gescannten Bytes. Die Kosten für eine Redshift-Spectrum-Abfrage richten sich nach der Menge der von Amazon S3 gescannten Daten.
- Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Bytes. Eine große Menge zurückgegebener Daten kann sich auf die Systemleistung auswirken.
- Die maximale und die durchschnittliche Dauer von Redshift Spectrum-Anfragen. Länger dauernde Anfragen können Anzeichen für einen Engpass sein.

Fehlerbehebung bei Abfragen in Amazon Redshift Spectrum

Im Folgenden finden Sie eine Kurzreferenz, die einige häufige Probleme identifiziert und behebt, die bei Amazon-Redshift-Spectrum-Abfragen auftreten können. Fragen Sie zur Anzeige der von Redshift Spectrum generierten Fehler die Systemtabelle SVL_S3LOG ab.

Themen

Überschreitung der Anzahl erneuter Versuche

Wenn bei einer Amazon-Redshift-Spectrum-Anfrage eine Zeitüberschreitung auftritt, wird die Anfrage abgebrochen und erneut übermittelt. Nach fünf fehlgeschlagenen erneuten Versuchen schlägt die Abfrage mit dem folgenden Fehler fehl.

```
error:  Spectrum Scan Error: Retries exceeded
```

Dafür gibt es die folgenden möglichen Ursachen:

- Große Dateien (über 1 GB). Prüfen Sie die Größe Ihrer Dateien in Amazon S3, und suchen Sie nach großen Dateien und Dateigrößenverzerrungen. Teilen Sie große Dateien in kleinere auf, zwischen 100 MB und 1 GB. Versuchen Sie, die Dateien etwa gleich groß zu halten.
- Langsamer Netzwerkdurchsatz. Versuchen Sie Ihre Abfrage später erneut.

Zugriff gedrosselt

Amazon Redshift Spectrum unterliegt den Servicekontingenten anderer AWS Services. Bei hoher Auslastung müssen Redshift-Spectrum-Anforderungen möglicherweise verlangsamt werden, was zum folgenden Fehler führt.

```
error:  Spectrum Scan Error: Access throttled
```

Es können zwei Arten der Drosselung auftreten:

- Zugriff gedrosselt von Amazon S3.
- Der Zugriff wurde gedrosselt von. AWS KMS

Der Fehlerkontext enthält weitere Details zum Typ der Drosselung. Nachfolgend können Sie Ursachen und mögliche Lösungen für diese Drosselung finden.

Zugriff gedrosselt von Amazon S3

Amazon S3 drosselt möglicherweise eine Redshift-Spectrum-Anforderung, wenn die Leseanforderungsrate für ein Präfix zu hoch ist. Informationen zu einer GET/HEAD-Anforderungsrate, die Sie in Amazon S3 erreichen können, finden Sie unter Optimieren der Leistung von Amazon S3 im Amazon-Simple-Storage-Service-Benutzerhandbuch. Die Amazon-S3-GET/HEAD Anforderungsrate berücksichtigt alle GET/HEAD-Anforderungen an ein Präfix, sodass verschiedene Anwendungen, die auf dasselbe Präfix zugreifen, die Gesamtanforderungsrate teilen.

Wenn Ihre Redshift-Spectrum-Anforderungen häufig von Amazon S3 gedrosselt werden, reduzieren Sie die Anzahl der Amazon-S3-GET/HEAD-Anforderungen, die Redshift Spectrum an Amazon S3

richtet. Versuchen Sie dazu, kleine Dateien zu größeren Dateien zusammenzufassen. Wir empfehlen, Dateigrößen von mindestens 64 MB zu verwenden.

Ziehen Sie auch die Partitionierung Ihrer Redshift-Spectrum-Tabellen in Betracht, um von einer frühen Filterung zu profitieren und die Anzahl der Dateien zu reduzieren, auf die in Amazon S3 zugegriffen wird. Weitere Informationen finden Sie unter Partitionierung externer Redshift-Spectrum-Tabellen.

Zugriff gedrosselt von AWS KMS

Wenn Sie Ihre Daten mit serverseitiger Verschlüsselung (SSE-S3 oder SSE-KMS) in Amazon S3 speichern, ruft Amazon S3 AWS KMS für jede Datei, auf die Redshift Spectrum zugreift, eine API-Operation auf. Diese Anforderungen zählen für Ihr kryptografisches Operationskontingent. Weitere Informationen finden Sie unter AWS KMS -Anforderungskontingente. Weitere Informationen zu SSE-S3 und SSE-KMS finden Sie unter Schutz von Daten mit serverseitiger Verschlüsselung und Schutz von Daten mit serverseitiger Verschlüsselung mit in AWS KMS gespeicherten KMS-Schlüsseln im Amazon-Simple-Storage-Service-Benutzerhandbuch.

Ein erster Schritt zur Reduzierung der Anzahl der Anfragen, an die Redshift Spectrum stellt, AWS KMS besteht darin, die Anzahl der abgerufenen Dateien zu reduzieren. Versuchen Sie dazu, kleine Dateien zu größeren Dateien zusammenzufassen. Wir empfehlen, Dateigrößen von mindestens 64 MB zu verwenden.

Wenn Ihre Redshift Spectrum-Anfragen häufig gedrosselt werden, sollten Sie erwägen AWS KMS, eine Erhöhung des Kontingents für Ihre Anforderungsrate für AWS KMS kryptografische Operationen zu beantragen. Informationen zum Anfordern einer Kontingenterhöhung finden Sie unter AWS - Service Limits im Allgemeine Amazon Web Services-Referenz.

Ressourcenlimit überschritten

Redshift Spectrum erzwingt eine Obergrenze für die Speichermenge, die eine Anforderung verwenden kann. Eine Redshift-Spectrum-Anforderung, die mehr Speicher benötigt, schlägt fehl, was zu dem folgenden Fehler führt.

```
error:  Spectrum Scan Error: Resource limit exceeded
```

Es gibt zwei häufige Gründe, die dazu führen können, dass eine Redshift-Spectrum-Anforderung ihr Speicherkontingent überschreitet:

- Redshift Spectrum verarbeitet einen großen Datenblock, der nicht in kleinere Datenblöcke aufgeteilt werden kann.
- Ein großer Aggregationsschritt wird von Redshift Spectrum verarbeitet.

Es wird empfohlen, ein Dateiformat zu verwenden, das parallele Lesevorgänge mit geteilten Größen von 128 MB oder weniger unterstützt. Unterstützte Dateiformate und allgemeine Richtlinien für die Erstellung von Datendateien finden Sie unter Erstellen von Datendateien für Abfragen in Amazon Redshift Spectrum. Bei Verwendung von Dateiformaten oder Komprimierungsalgorithmen, die keine parallelen Lesevorgänge unterstützen, empfehlen wir, die Dateigrößen zwischen 64 MB und 128 MB zu halten.

Für eine partitionierte Tabelle werden keine Zeilen ausgegeben.

Wenn Ihre Abfrage von einer partitionierten externen Tabelle null Zeilen ausgibt, prüfen Sie, ob für diese externe Tabelle eine Partition hinzugefügt wurde. Redshift Spectrum scannt nur Dateien an einem Amazon-S3-Speicherort, der ausdrücklich mit hinzugefügt wurde `ALTER TABLE … ADD PARTITION`. Fragen Sie die Ansicht SVV_EXTERNAL_PARTITIONS ab, um die vorhandenen Partitionen zu finden. Führen Sie für jede fehlende Partition `ALTER TABLE … ADD PARTITION` aus.

„Nicht autorisiert“-Fehler

Prüfen Sie, ob die IAM-Rolle für den Cluster den Zugriff auf die Amazon-S3-Dateiobjekte erlaubt. Wenn sich Ihre externe Datenbank auf Amazon Athena befindet, prüfen Sie, ob die IAM-Rolle den Zugriff auf Athena-Ressourcen erlaubt. Weitere Informationen finden Sie unter IAM-Richtlinien für Amazon Redshift Spectrum.

Inkompatible Datenformate

Bei einem Spaltenformat wie Apache Parquet ist der Spaltentyp mit den Daten eingebettet. Der Spaltentyp in der CREATE EXTERNAL TABLE-Definition muss dem Spaltentyp der Datendatei entsprechen. Wenn dies nicht der Fall ist, erhalten Sie eine Fehlermeldung wie die folgende:

```
File 'https://s3bucket/location/file has an incompatible Parquet schema
for column ‘s3://s3bucket/location.col1'. Column type: VARCHAR, Par
```

Die Fehlermeldung kann aus Längengründen verkürzt angezeigt werden. Um die gesamte Fehlermeldung, einschließlich Spaltenname und Spaltentyp, abzurufen, fragen Sie die Systemansicht SVL_S3LOG ab.

Im folgenden Beispiel wird SVL_S3LOG nach der letzten abgeschlossene Abfrage abgefragt.

```
select message
from svl_s3log
where query = pg_last_query_id()
order by query,segment,slice;
```

nachfolgend sehen Sie ein Beispiel für ein Ergebnis mit der vollständigen Fehlermeldung.

```
                            message
-----------------------------------------------------------------------------------------
Spectrum Scan Error. File 'https://s3bucket/location/file has an incompatible
Parquet schema for column ' s3bucket/location.col1'.
Column type: VARCHAR, Parquet schema:\noptional int64 l_orderkey [i:0 d:1 r:0]\n
```

Um den Fehler zu korrigieren, ändern Sie die externe Tabelle so, dass sie dem Spaltentyp der Parquet-Datei entspricht.

Syntaxfehler bei der Verwendung von Hive-DDL in Amazon Redshift

Amazon Redshift unterstützt die Data Definition Language (DDL) für CREATE EXTERNAL TABLE, die ähnlich der Hive-DDL ist. Die beiden DDL-Typen sind aber nicht überall gleich. Wenn Sie Hive-DDL zur Erstellung oder Änderung externer Amazon-Redshift-Tabellen kopieren, können Syntaxfehler auftreten. Nachfolgend sehen Sie einige Beispiele für Unterschiede zwischen Amazon Redshift und Hive-DDL:

- Amazon Redshift erfordert einfache Anführungszeichen ('), während Hive-DDL doppelte Anführungszeichen unterstützt (").
- Amazon Redshift unterstützt nicht den Datentyp STRING. Verwenden Sie stattdessen VARCHAR.

Berechtigungen zum Erstellen temporärer Tabellen

Um Redshift Spectrum-Abfragen auszuführen, benötigt der Datenbankbenutzer die Berechtigung, temporäre Tabellen in der Datenbank zu erstellen. Das folgende Beispiel erteilt der Benutzergruppe `spectrumdb` temporäre Berechtigungen für die Datenbank `spectrumusers`.

```
grant temp on database spectrumdb  to group spectrumusers;
```

Weitere Informationen finden Sie unter GRANT.

Ungültiger Bereich

Redshift Spectrum erwartet, dass Dateien in Amazon S3, die zu einer externen Tabelle gehören, bei einer Abfrage nicht überschrieben werden. Sollte dies doch geschehen, kann der folgende Fehler auftreten.

```
Error: HTTP response error code: 416 Message: InvalidRange The requested range is not
 satisfiable
```

Um den Fehler zu vermeiden, stellen Sie sicher, dass Amazon S3-Dateien nicht überschrieben werden, während sie mit Redshift Spectrum abgefragt werden.

Ungültige Parquet-Versionsnummer

Redshift Spectrum überprüft die Metadaten jeder Apache Parquet-Datei, auf die zugegriffen wird. Wenn die Prüfung fehlschlägt, kann dies zu einem Fehler wie dem folgenden führen:

```
File 'https://s3.region.amazonaws.com/s3bucket/location/file has an invalid version
 number
```

Es gibt zwei häufige Gründe, die dazu führen können, dass die Überprüfung fehlschlägt:

- Die Parquet-Datei wurde bei der Abfrage überschrieben (siehe Ungültiger Bereich).
- Die Parquett-Datei ist beschädigt.

Tutorial: Abfragen verschachtelter Daten mit Amazon Redshift Spectrum

Übersicht

Amazon Redshift Spectrum unterstützt das Abfragen verschachtelter Daten in den Dateiformaten ORC, JSON und Ion. Redshift Spectrum greift mittels externer Tabellen auf die Daten zu. Sie können externe Tabellen erstellen, die die komplexen Datentypen `struct`, `array` und `map` verwenden.

Angenommen, Ihre Datendatei enthält die folgenden Daten in Amazon S3 in einem Ordner mit dem Namen `customers`. Obwohl kein einziges Stammelement vorhanden ist, stellt jedes JSON-Objekt in diesen Stichprobendaten eine Zeile in einer Tabelle dar.

```
{"id": 1,
 "name": {"given": "John", "family": "Smith"},
 "phones": ["123-457789"],
 "orders": [{"shipdate": "2018-03-01T11:59:59.000Z", "price": 100.50},
            {"shipdate": "2018-03-01T09:10:00.000Z", "price": 99.12}]
}
{"id": 2,
 "name": {"given": "Jenny", "family": "Doe"},
 "phones": ["858-8675309", "415-9876543"],
 "orders": []
}
{"id": 3,
 "name": {"given": "Andy", "family": "Jones"},
 "phones": [],
 "orders": [{"shipdate": "2018-03-02T08:02:15.000Z", "price": 13.50}]
}
```

Sie können mithilfe von Amazon Redshift Spectrum verschachtelte Daten in Dateien abfragen. Das folgende Tutorial veranschaulicht die Vorgehensweise hierfür mit Apache-Parquet-Daten.

Voraussetzungen für das Tutorial, Verfahrensschritte und Anwendungsfälle verschachtelter Daten finden Sie in den folgenden Themen:

- Voraussetzungen
- Schritt 1: Erstellen einer externen Tabelle mit verschachtelten Daten
- Schritt 2: Abfragen Ihrer verschachtelten Daten in Amazon S3 mit SQL-Erweiterungen
- Anwendungsfälle für verschachtelte Daten
- Einschränkungen bei verschachtelten Daten (Vorschau)
- Serialisieren komplexer verschachtelter JSON-Datentypen

Voraussetzungen

Wenn Sie noch nicht mit Redshift Spectrum vertraut sind, befolgen Sie die Schritte unter Erste Schritte mit Amazon Redshift Spectrum, bevor Sie fortfahren.

Um ein externes Schema zu erstellen, ersetzen Sie den ARN der IAM-Rolle im folgenden Befehl durch den Rollen-ARN, den Sie im Abschnitt Erstellen einer IAM-Rolle festgelegt haben. Führen Sie dann den Befehl in Ihrem SQL-Client aus.

```
create external schema spectrum
from data catalog
database 'myspectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
create external database if not exists;
```

Schritt 1: Erstellen einer externen Tabelle mit verschachtelten Daten

Sie können die Quelldaten anzeigen, indem Sie sie von Amazon S3 herunterladen.

Um die externe Tabelle für dieses Tutorial zu erstellen, führen Sie den folgenden Befehl aus.

```
CREATE EXTERNAL TABLE spectrum.customers (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones array<varchar(20)>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

Im voranstehenden Beispiel verwendet die externe Tabelle `spectrum.customers` die Datentypen `struct` und `array`, um Spalten mit verschachtelten Daten zu definieren. Amazon Redshift Spectrum unterstützt das Abfragen verschachtelter Daten in den Dateiformaten ORC, JSON und Ion. Der Parameter `STORED AS` für Apache-Parquet-Dateien lautet `PARQUET`. Der Parameter `LOCATION` muss auf den Amazon S3-Ordner verweisen, der die verschachtelten Daten oder Dateien enthält. Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.

Sie können die Typen `array` und `struct` auf jeder beliebigen Ebene verschachteln. So können Sie z. B. eine Spalte mit dem Namen `toparray` definieren; siehe folgendes Beispiel.

```
toparray array<struct<nestedarray:
         array<struct<morenestedarray:
         array<string>>>>>
```

Außerdem können Sie wie für Spalte `struct` im folgenden Beispiel veranschaulicht auch `x`-Typen verschachteln.

```
x struct<a: string,
         b: struct<c: integer,
                   d: struct<e: string>
```

```
                >
        >
```

Schritt 2: Abfragen Ihrer verschachtelten Daten in Amazon S3 mit SQL-Erweiterungen

Redshift Spectrum unterstützt das Abfragen komplexer `array`-, `map`- und `struct`-Typen durch Erweiterungen der Amazon-Redshift-SQL-Syntax.

Erweiterung 1: Zugriff auf Structs-Spalten

Sie können Daten aus `struct`-Spalten extrahieren. Dazu verwenden Sie eine punktierte Schreibweise, die Feldnamen in Pfade verkettet. Die folgende Abfrage gibt z. B. Vor- und Nachnamen von Kunden zurück. Auf den Vornamen wird über den langen Pfad `c.name.given` zugegriffen. Auf den Nachnamen wird über den langen Pfad `c.name.family` zugegriffen.

```
SELECT c.id, c.name.given, c.name.family
FROM   spectrum.customers c;
```

Die vorhergehende Abfrage gibt die folgenden Daten zurück.

```
id | given | family
---|-------|-------
1  | John  | Smith
2  | Jenny | Doe
3  | Andy  | Jones
(3 rows)
```

Ein `struct`-Element kann eine Spalte eines anderen `struct`-Elements sein, das wiederum auf einer beliebigen anderen Ebene eine Spalte eines anderen `struct`-Elements sein kann. Die Pfade, die auf Spalten in solchen tief verschachtelten `struct`-Elementen zugreifen können willkürlich lang sein. Betrachten Sie sich z. B. die Definition der Spalte `x` im folgenden Beispiel.

```
x struct<a: string,
         b: struct<c: integer,
                   d: struct<e: string>
                  >
        >
```

Sie können auf die Daten in `e` als `x.b.d.e` zugreifen.

Erweiterung 2: Übergreifende Arrays in einer FROM-Klausel

Sie können Daten aus `array`-Spalten (und damit auch aus `map`-Spalten) extrahieren, indem Sie die `array`-Spalten in einer `FROM`-Klausel anstelle von Tabellennamen angeben. Die Erweiterung gilt für die `FROM`-Klausel der Hauptabfrage sowie auch für die `FROM`-Klauseln von Unterabfragen.

Sie können auf `array`-Elemente nach Position verweisen, also beispielsweise `c.orders[0]` angeben (Vorschau).

Durch das Kombinieren von übergreifenden `arrays` mit Joins können Sie die Verschachtelung auf verschiedene Weise aufheben, wie in den folgenden Anwendungsfällen beschrieben.

Aufheben der Verschachtelung mit Inner Joins

Die folgende Abfrage wählt Kunden-IDs und Auftragslieferdaten für Kunden aus, für die Aufträgen vorhanden sind. Die SQL-Erweiterung in der FROM-Klausel `c.orders o` ist vom Alias `c` abhängig.

```
SELECT c.id, o.shipdate
FROM   spectrum.customers c, c.orders o
```

Für jeden Kunden `c`, für den Aufträge vorhanden sind, gibt die `FROM`-Klausel für jeden Auftrag `o` des Kunden `c` eine Zeile zurück. Diese Zeile kombiniert die Kundenzeile `c` und die Auftragszeile `o`. Die `SELECT`-Klausel behält dann nur `c.id` und `o.shipdate` bei. Dies führt zu folgendem Ergebnis.

```
id|      shipdate
--|----------------------
1 |2018-03-01  11:59:59
1 |2018-03-01  09:10:00
3 |2018-03-02  08:02:15
(3 rows)
```

Der Alias `c` bietet Zugriff auf die Kundenfelder, und der Alias `o` bietet Zugriff auf die Auftragsfelder.

Die Semantik ist mit Standard-SQL vergleichbar. Sie können sich die `FROM`-Klausel so vorstellen, als würde die folgende verschachtelte Schleife ausgeführt werden, gefolgt von `SELECT` zur Auswahl der auszugebenden Felder.

```
for each customer c in spectrum.customers
  for each order o in c.orders
     output c.id and o.shipdate
```

Daher erscheinen Kunden, für die keine Aufträge vorhanden sind, nicht im Ergebnis.

Sie können sich das auch so vorstellen, als ob die FROM-Klausel einen JOIN mit der customers-Tabelle und dem orders-Array durchführen wurde. Sie können die Abfrage daher auch wie im folgenden Beispiel veranschaulicht schreiben.

```
SELECT c.id, o.shipdate
FROM   spectrum.customers c INNER JOIN c.orders o ON true
```

> **Note**
>
> Wenn ein Schema mit dem Namen c mit einer Tabelle mit dem Namen orders vorhanden ist, dann bezieht sich c.orders auf die Tabelle orders und nicht auf die Array-Spalte von customers.

Aufheben der Verschachtelung mit Left Joins

Die folgende Abfrage gibt alle Kundennamen und deren Aufträge aus. Für Kunden, die keinen Auftrag erteilt haben, wird dennoch der Kundenname zurückgegeben. In diesem Fall ist der Wert der Auftragsspalten jedoch NULL, wie im folgenden Beispiel für Jenny Doe veranschaulicht.

```
SELECT c.id, c.name.given, c.name.family, o.shipdate, o.price
FROM   spectrum.customers c LEFT JOIN c.orders o ON true
```

Die vorhergehende Abfrage gibt die folgenden Daten zurück.

```
id  |  given  | family  |    shipdate          | price
----|---------|---------|----------------------|--------
 1  |  John   | Smith   | 2018-03-01  11:59:59 | 100.5
 1  |  John   | Smith   | 2018-03-01  09:10:00 |  99.12
 2  |  Jenny  | Doe     |                      |
 3  |  Andy   | Jones   | 2018-03-02  08:02:15 |  13.5
 (4 rows)
```

Erweiterung 3: Direkter Zugriff auf ein Array von Skalaren über einen Alias

Wenn sich der Alias p in einer FROM-Klausel über ein Array von Skalaren erstreckt, bezeichnet die Abfrage die p-Werte als p. Die folgende Abfrage ergibt z. B. Paare aus Kundennamen und Telefonnummern.

```
SELECT c.name.given, c.name.family, p AS phone
FROM   spectrum.customers c LEFT JOIN c.phones p ON true
```

Die vorhergehende Abfrage gibt die folgenden Daten zurück.

```
given  |  family  |  phone
-------|----------|-----------
John   | Smith    | 123-4577891
Jenny  | Doe      | 858-8675309
Jenny  | Doe      | 415-9876543
Andy   | Jones    |
(4 rows)
```

Erweiterung 4: Zugriff auf Zuordnungselemente

Redshift Spectrum verarbeitet den `map`-Datentyp als einen `array`-Typ, der `struct`-Typen mit einer `key`-Spalte und einer `value`-Spalte enthält. `key` muss vom Typ `scalar` sein; der Wert kann ein beliebiger Datentyp sein.

Der folgende Code erstellt z. B. eine externe Tabelle mit einem `map`-Element zum Speichern von Telefonnummern.

```
CREATE EXTERNAL TABLE spectrum.customers2 (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones map<varchar(20), varchar(20)>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

Da sich ein `map`-Typ wie ein `array`-Typ mit den Spalten `key` und `value` verhält, können Sie sich die vorhergehenden Schemen so vorstellen, als wären sie die folgenden Schemen.

```
CREATE EXTERNAL TABLE spectrum.customers3 (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones array<struct<key:varchar(20), value:varchar(20)>>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
```

```
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

Die folgende Abfrage gibt die Namen von Kunden mit Mobiltelefonnummer und die Nummer für jeden Namen zurück. Die Zuordnungsabfrage wird genauso wie die Abfrage eines verschachtelten `array` aus `struct`-Typen verarbeitet. Die folgende Abfrage gibt nur Daten zurück, wenn Sie die externe Tabelle wie zuvor beschrieben erstellt haben.

```
SELECT c.name.given, c.name.family, p.value
FROM   spectrum.customers c, c.phones p
WHERE  p.key = 'mobile';
```

> **Note**
>
> Der `key`-Wert für ein `map`-Element ist bei den Dateitypen Ion und JSON vom Typ `string`.

Anwendungsfälle für verschachtelte Daten

Sie können die zuvor beschriebenen Erweiterungen mit den gewöhnlichen SQL-Funktionen kombinieren. Die folgenden Anwendungsfälle veranschaulichen einige häufige Kombinationen. Anhand dieser Beispiele soll demonstriert werden, wie Sie verschachtelte Daten anwenden können. Sie sind nicht Teil des Tutorials.

Themen

- Aufnehmen verschachtelter Daten
- Aggregieren verschachtelter Daten mit Unterabfragen
- Verknüpfen von Amazon Redshift und verschachtelten Daten

Aufnehmen verschachtelter Daten

Sie können eine `CREATE TABLE AS`-Anweisung verwenden, um Daten aus einer externen Tabelle mit komplexen Datentypen aufzunehmen. Die folgende Abfrage extrahiert alle Kunden und ihre Telefonnummern aus der externen Tabelle mithilfe von `LEFT JOIN` und speichert sie in der Amazon-Redshift-Tabelle `CustomerPhones`.

```
CREATE TABLE CustomerPhones AS
SELECT  c.name.given, c.name.family, p AS phone
```

```
FROM    spectrum.customers c LEFT JOIN c.phones p ON true;
```

Aggregieren verschachtelter Daten mit Unterabfragen

Sie können eine Unterabfrage zum Aggregieren von verschachtelten Daten verwenden. Die folgende Abbildung veranschaulicht dieses Konzept.

```
SELECT c.name.given, c.name.family, (SELECT COUNT(*) FROM c.orders o) AS ordercount
FROM   spectrum.customers c;
```

Die folgenden Daten werden zurückgegeben.

```
given   |  family  |  ordercount
--------|----------|--------------
 Jenny  |  Doe     |       0
 John   |  Smith   |       2
 Andy   |  Jones   |       1
 (3 rows)
```

> **Note**
>
> Wenn Sie verschachtelte Daten durch Gruppieren nach der übergeordneten Zeile aggregieren, ist die effizienteste Methode dafür die, die im vorherigen Beispiel veranschaulicht wird. Im diesem Beispiel werden die verschachtelten `c.orders`-Zeilen nach der ihnen übergeordneten Zeile `c` gruppiert. Wenn Ihnen bekannt ist, dass `id` für jeden `customer`-Eintrag eindeutig und jeder `o.shipdate`-Eintrag niemals null ist, können Sie alternativ dazu wie im folgenden Beispiel gezeigt aggregieren. Dieser Ansatz ist in der Regel jedoch nicht so effizient wie das vorherige Beispiel.

```
SELECT    c.name.given, c.name.family, COUNT(o.shipdate) AS ordercount
FROM      spectrum.customers c LEFT JOIN c.orders o ON true
GROUP BY  c.id, c.name.given, c.name.family;
```

Sie können die Abfrage auch mit einer Unterabfrage in der `FROM`-Klausel schreiben, die sich auf einen Alias (`c`) der Vorgängerabfrage bezieht und Array-Daten extrahiert. Das folgende Beispiel illustriert diese Herangehensweise.

```
SELECT c.name.given, c.name.family, s.count AS ordercount
```

```
FROM   spectrum.customers c, (SELECT count(*) AS count FROM c.orders o) s;
```

Verknüpfen von Amazon Redshift und verschachtelten Daten

Sie können Amazon-Redshift-Daten auch mit verschachtelten Daten in einer externen Tabelle verknüpfen. Angenommen, Sie haben die folgenden verschachtelten Daten in Amazon S3.

```
CREATE EXTERNAL TABLE spectrum.customers2 (
  id      int,
  name    struct<given:varchar(20), family:varchar(20)>,
  phones  array<varchar(20)>,
  orders  array<struct<shipdate:timestamp, item:int>>
);
```

Nehmen wir außerdem an, Sie haben die folgende Tabelle in Amazon Redshift.

```
CREATE TABLE prices (
  id int,
  price double precision
);
```

Die folgende Abfrage findet basierend auf dem Vorangehenden die Gesamtzahl und Gesamtmenge der Käufe eines jeden Kundens. Das folgende Beispiel dient nur zur Veranschaulichung. Es gibt nur Daten zurück, wenn Sie die Tabellen wie zuvor beschrieben erstellt haben.

```
SELECT   c.name.given, c.name.family, COUNT(o.date) AS ordercount, SUM(p.price) AS
 ordersum
FROM     spectrum.customers2 c, c.orders o, prices p ON o.item = p.id
GROUP BY c.id, c.name.given, c.name.family;
```

Einschränkungen bei verschachtelten Daten (Vorschau)

> **Note**
>
> Die in der folgenden Liste (mit Vorschau) gekennzeichneten Einschränkungen gelten nur für Vorschau-Cluster und Vorschau-Arbeitsgruppen, die in den folgenden Regionen erstellt wurden.
>
> - USA Ost (Ohio): (us-east-2)
> - USA Ost (Nord-Virginia): (us-east-1)

- USA West (Nordkalifornien) (us-west-1)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Weitere Informationen zum Einrichten von Vorschau-Clustern finden Sie unter Erstellen eines Vorschau-Clusters im Amazon-Redshift-Verwaltungshandbuch. Weitere Informationen zum Einrichten von Vorschau-Arbeitsgruppen finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe im Amazon-Redshift-Verwaltungshandbuch.

Die folgenden Einschränkungen gelten für verschachtelte Daten:

- Ein `array`- oder `map`-Typ kann andere `array`- oder `map`-Typen enthalten, sofern Abfragen für die verschachtelten `arrays` oder `maps` keine `scalar`-Werte zurückgeben. (Vorschau)
- Amazon Redshift Spectrum unterstützt komplexe Datentypen nur als externe Tabellen.
- Die Ergebnisspalten der Unterabfrage müssen sich auf oberster Ebene befinden. (Vorschau)
- Wenn sich ein `OUTER JOIN`-Ausdruck auf eine verschachtelte Tabelle bezieht, darf er sich nur auf die betreffende Tabelle und ihre verschachtelten Arrays (und Zuordnungen) beziehen. Wenn sich ein `OUTER JOIN`-Ausdruck nicht auf eine verschachtelte Tabelle bezieht, kann er sich auf eine beliebige Anzahl nicht verschachtelter Tabellen beziehen.
- Wenn sich eine `FROM`-Klausel in einer Unterabfrage auf eine verschachtelte Tabelle bezieht, darf sie sich auf keine andere Tabelle beziehen.
- Wenn eine Unterabfrage von einer verschachtelten Tabelle abhängig ist, die sich auf eine übergeordnete Tabelle bezieht, kann die Unterabfrage die übergeordnete Tabelle nur in der `FROM`-Klausel verwenden. Sie können die übergeordnete Tabelle in keiner anderen Klausel, wie z. B. einer `SELECT`- oder `WHERE`-Klausel, verwenden. Die folgende Abfrage wird beispielsweise nicht ausgeführt, da sich die `SELECT`-Klausel der Unterabfrage auf die übergeordnete Tabelle `c` bezieht.

```
SELECT c.name.given
FROM   spectrum.customers c
WHERE (SELECT COUNT(c.id) FROM c.phones p WHERE p LIKE '858%') > 1;
```

Die folgende Abfrage funktioniert, da die übergeordnete Tabelle `c` nur in der `FROM`-Klausel der Unterabfrage verwendet wird.

```
SELECT c.name.given
FROM   spectrum.customers c
WHERE (SELECT COUNT(*) FROM c.phones p WHERE p LIKE '858%') > 1;
```

- Eine Unterabfrage, die an anderer Stelle als die FROM-Klausel auf verschachtelte Daten zugreift, muss einen einzelnen Wert zurückgeben. Die einzigen Ausnahmen sind `(NOT) EXISTS`-Operatoren in einer WHERE-Klausel.
- `(NOT) IN` wird nicht unterstützt.
- Die maximale Verschachtelungstiefe aller verschachtelter Typen ist 100. Diese Einschränkung gilt für alle Dateiformate (Parquet, ORC, Ion und JSON).
- Aggregations-Unterabfragen, die auf verschachtelte Daten zugreifen, dürfen nur auf `arrays` und `maps` in ihrer FROM-Klausel verweisen, nicht auf eine externe Tabelle.
- Das Abfragen der Pseudospalten verschachtelter Daten in einer Redshift-Spectrum-Tabelle wird nicht unterstützt. Weitere Informationen finden Sie unter Pseudospalten.
- Wenn Sie Daten aus Array- oder Map-Spalten extrahieren, indem Sie diese in einer FROM-Klausel angeben, können Sie nur Werte aus diesen Spalten auswählen, wenn es sich um `scalar`-Werte handelt. Die folgenden Abfragen versuchen beispielsweise beide, Elemente aus einem Array auszuwählen (SELECT). Die Abfrage, die `arr.a` auswählt, funktioniert, da es sich bei `arr.a` um einen `scalar`-Wert handelt. Die zweite Abfrage funktioniert nicht, da `array` ein Array ist, das in der FROM-Klausel aus `s3.nested table` extrahiert wurde. (Vorschau)

```
SELECT array_column FROM s3.nested_table;

array_column
-----------------
[{"a":1},{"b":2}]

SELECT arr.a FROM s3.nested_table t, t.array_column arr;

arr.a
-----
1

--This query fails to run.
SELECT array FROM s3.nested_table tab, tab.array_column array;
```

Sie können in der FROM-Klausel keine Arrays oder Maps verwenden, die selbst aus anderen Arrays oder Maps stammen. Um Arrays oder andere komplexe Strukturen auszuwählen, die in anderen Arrays verschachtelt sind, könnten Sie Indizes in der SELECT-Anweisung verwenden.

Serialisieren komplexer verschachtelter JSON-Datentypen

Eine Alternative zu den in diesem Tutorial demonstrierten Methoden besteht darin, verschachtelte Sammlungsspalten der obersten Ebene als serialisierten JSON-Datentyp abzufragen. Sie können die Serialisierung verwenden, um verschachtelte Daten als JSON mit Redshift Spectrum zu untersuchen, zu konvertieren und aufzunehmen. Diese Methode wird für ORC-, JSON-, Ion- und Parquet-Formate unterstützt. Verwenden Sie den Sitzungskonfigurationsparameter `json_serialization_enable`, um das Serialisierungsverhalten zu konfigurieren. Wenn diese Einstellung festgelegt ist, werden komplexe JSON-Datentypen in VARCHAR(65535) serialisiert. Auf den verschachtelten JSON-Datentyp kann mit JSON-Funktionen zugegriffen werden. Weitere Informationen finden Sie unter json_serialization_enable.

Beispiel: Ohne die Einstellung von `json_serialization_enable` schlagen die folgenden Abfragen, die auf verschachtelte Spalten direkt zugreifen, fehl.

```
SELECT * FROM spectrum.customers LIMIT 1;

=> ERROR:  Nested tables do not support '*' in the SELECT clause.

SELECT name FROM spectrum.customers LIMIT 1;

=> ERROR:  column "name" does not exist in customers
```

Das Einstellen von `json_serialization_enable` ermöglicht das direkte Abfragen von Sammlungen der obersten Ebene.

```
SET json_serialization_enable TO true;

SELECT * FROM spectrum.customers order by id LIMIT 1;

id | name                                | phones         | orders
---+-------------------------------------+----------------
+-----------------------------------------------------------------------------------------------
```

```
1  | {"given": "John", "family": "Smith"} | ["123-457789"] | [{"shipdate":
 "2018-03-01T11:59:59.000Z", "price": 100.50}, {"shipdate": "2018-03-01T09:10:00.000Z",
 "price": 99.12}]

SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": "John", "family": "Smith"}
```

Beachten Sie die folgenden Elemente, wenn Sie verschachtelte JSON-Datentypen serialisieren.

- Wenn Sammlungsspalten als VARCHAR(65535) serialisiert werden, kann auf ihre verschachtelten Unterfelder nicht direkt als Teil der Abfragesyntax zugegriffen werden (z. B. in der Filter-Klausel). JSON-Funktionen können jedoch für den Zugriff auf verschachtelte JSON-Datentypen verwendet werden.
- Die folgenden spezialisierten Darstellungen werden nicht unterstützt:
  - ORC-Vereinigungen
  - ORC-Zuordnungen mit komplexen Typenschlüsseln
  - Ion-Datagramme
  - Ion SEXP
- Zeitstempel werden als serialisierte ISO-Zeichenfolgen zurückgegeben.
- Primitive Zuordnungsschlüssel werden zu einer Zeichenfolge heraufgestuft (z. B. 1 zu `"1"`).
- Nullwerte der obersten Ebene werden als NULL serialisiert.
- Wenn die Serialisierung die maximale VARCHAR-Größe von 65535 übersteigt, wird die Zelle auf NULL gesetzt.

Serialisieren komplexer Typen, die JSON-Zeichenfolgen enthalten

Standardmäßig werden Zeichenfolgenwerte, die in verschachtelten Sammlungen enthalten sind, als Escape-JSON-Zeichenfolgen serialisiert. Escaping kann unerwünscht sein, wenn die Zeichenfolgen gültige JSON-Ausdrücke sind. Stattdessen sollten Sie verschachtelte VARCHAR-serialisierte Unterelemente oder Felder direkt als JSON schreiben. Aktivieren Sie dieses Verhalten mit der Konfiguration `json_serialization_parse_nested_strings` auf Sitzungsebene. Wenn `json_serialization_enable` und `json_serialization_parse_nested_strings` eingestellt sind, werden gültige JSON-Werte inline ohne Escape-Zeichen serialisiert.

Wenn der Wert kein gültiger JSON-Ausdruck ist, wird er escaped, als ob der `json_serialization_parse_nested_strings`-Konfigurationswert nicht festgelegt wäre. Weitere Informationen finden Sie unter json_serialization_parse_nested_strings.

Angenommen, die Daten aus dem vorherigen Beispiel enthielten JSON als komplexen `structs`-Typ in im VARCHAR(20)-Feld `name`:

```
name
---------
{"given": "{\"first\":\"John\",\"middle\":\"James\"}", "family": "Smith"}
```

Wenn `json_serialization_parse_nested_strings` eingestellt ist, wird die Spalte `name` wie folgt serialisiert:

```
SET json_serialization_enable TO true;
SET json_serialization_parse_nested_strings TO true;
SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": {"first":"John","middle":"James"}, "family": "Smith"}
```

Anstatt wie folgt escaped zu werden:

```
SET json_serialization_enable TO true;
SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": "{\"first\":\"John\",\"middle\":\"James\"}", "family": "Smith"}
```

Verwenden von HyperLogLog Skizzen in Amazon Redshift

HyperLogLog ist ein Algorithmus, der zur Schätzung der Kardinalität eines Multisets verwendet wird. Kardinalität bezieht sich auf die Anzahl der einzelnen Werte in einer Multimenge. Zum Beispiel: In der Menge {4,3,6,2,2,6,4,3,6,2,2,3} ist die Kardinalität 4, da die einzelnen Werte 4, 3, 6 und 2 sind.

Die Genauigkeit des HyperLogLog Algorithmus (auch als M-Wert bezeichnet) kann sich auf die Genauigkeit der geschätzten Kardinalität auswirken. Während der Kardinalitätsschätzung verwendet Amazon Redshift einen Standardgenauigkeitswert von 15. Bei kleineren Datensätzen ist ein Wert von bis zu 26 möglich. Die durchschnittliche relative Fehlerquote liegt also zwischen 0,01 und 0,6 %.

Bei der Berechnung der Kardinalität einer Mehrfachmenge generiert der HyperLogLog Algorithmus ein Konstrukt, das als HLL-Skizze bezeichnet wird. HLL-Skizzen kapseln Informationen zu den unterschiedlichen Werte in einem Multiset. Der Amazon-Redshift-Datentyp HLLSKETCH repräsentiert diese Skizzenwerte. Der Datentyp kann verwendet werden, um Skizzen in einer Amazon-Redshift-Tabelle zu speichern. Darüber hinaus unterstützt Amazon Redshift Vorgänge, die als Aggregation- und Skalarfunktionen auf HLLSKETCH-Werte angewendet werden können. Mit diesen Funktionen können Sie die Kardinalität einer HLLSKETCH extrahieren und mehrere HLLSKETCH-Werte kombinieren.

Der Datentyp HLLSKETCH bietet erhebliche Abfrageleistungsvorteile beim Extrahieren der Kardinalität aus großen Datensätzen. Sie können eine Voraggregation dieser Datensätze vornehmen, indem Sie HLLSKETCH-Werte verwenden und diese in Tabellen speichern. Amazon Redshift kann die Kardinalität direkt aus den gespeicherten HLLSKETCH-Werten extrahieren, ohne auf die zugrundeliegenden Datensätze zuzugreifen.

Bei der Verarbeitung von HLL-Skizzen führt Amazon Redshift Optimierungen durch, die den Speicherbedarf der Skizze minimieren und die Genauigkeit der extrahierten Kardinalität maximieren. Amazon Redshift verwendet zwei Darstellungen für HLL-Skizzen, rudimentär und ausführlich. Zu Beginn hat eine HLLSKETCH ein rudimentäres Format. Wenn neue Werte in eingefügt werden, nimmt die Größe zu. Wenn die Größe die Größe der ausführlichen Darstellung erreicht hat, konvertiert Amazon Redshift die Skizze automatisch von rudimentär zu ausführlich.

Amazon Redshift importiert, exportiert und gibt eine HLLSKETCH als JSON aus, wenn die Skizze ein rudimentäres Format hat. Amazon Redshift importiert, exportiert und gibt eine HLLSKETCH als Base64-Zeichenfolge aus, wenn die Skizze ein ausführliches Format hat. Weitere Informationen zu UNLOAD finden Sie unter Entladen des Datentyps HLLSKETCH. Verwenden Sie den COPY-Befehl,

um Text- oder CSV-Daten (Comma-Separated Value) in Amazon Redshift zu importieren. Weitere Informationen finden Sie unter Laden des Datentyps HLLSKETCH.

Hinweise zu Funktionen, die mit verwendet werden, finden Sie unter. HyperLogLog HyperLogLog Funktionen

Themen

- Überlegungen
- Einschränkungen
- Beispiele

Überlegungen

Die folgenden Überlegungen sollten bei der Verwendung HyperLogLog in Amazon Redshift beachtet werden:

- Die folgenden HyperLogLog Nichtfunktionen können eine Eingabe des Typs HLLSKETCH oder Spalten des Typs HLLSKETCH akzeptieren:
  - Die Aggregatfunktion COUNT
  - Die bedingten Ausdrücke COALESCE und NVL
  - CASE-Ausdrücke
- Die unterstützte Kodierung ist RAW.
- Sie können einen UNLOAD-Vorgang für eine Tabelle mit HLLSKETCH-Spalten in Text oder CSV-Spalten durchführen. Sie können die UNLOAD-HLLSKETCH-Spalten verwenden, um HLLSKETCH-Daten zu schreiben. Amazon Redshift zeigt Daten für eine rudimentäre Darstellung im JSON-Format und für eine ausführliche Darstellung im Base64-Format an. Weitere Informationen zu UNLOAD finden Sie unter Entladen des Datentyps HLLSKETCH.

  Im Folgenden wird das Format gezeigt, das für eine dünn besetzte HyperLogLog Skizze verwendet wird, die in einem JSON-Format dargestellt wird.

  ```
  {"version":1,"logm":15,"sparse":{"indices":
  [15099259,33107846,37891580,50065963],"values":[2,3,2,1]}}
  ```
- Sie können Text- oder CSV-Daten mit dem Befehl COPY in Amazon Redshift importieren. Weitere Informationen finden Sie unter Laden des Datentyps HLLSKETCH.

- Die Standardkodierung für HLLSKETCH ist RAW. Weitere Informationen finden Sie unter Kompressionskodierungen.

Einschränkungen

Die folgenden Einschränkungen gelten für die Verwendung HyperLogLog in Amazon Redshift:

- Amazon-Redshift-Tabellen unterstützen keine HLLSKETCH-Spalte als Sortierschlüssel oder Verteilungsschlüssel.
- Amazon Redshift unterstützt keine HLLSKETCH-Spalten in ORDER-BY-, GROUP-BY- oder DISTINCT-Klauseln.
- Sie können HLLSKETCH-Spalten nur im Text- oder CSV-Format entladen. Amazon Redshift schreibt die HLLSKETCH-Daten dann entweder im JSON- oder im Base64-Format. Weitere Informationen zu UNLOAD finden Sie unter UNLOAD.
- Amazon Redshift unterstützt nur HyperLogLog Skizzen mit einer Genauigkeit (Logm-Wert) von 15.
- JDBC- und ODBC-Treiber unterstützen den HLLSKETCH-Datentyp nicht. Daher verwendet der Ergebnissatz VARCHAR, um die HLLSKETCH-Werte darzustellen.
- Amazon Redshift Spectrum unterstützt die HLLSKETCH-Daten nicht standardmäßig. Daher können Sie keine externe Tabelle mit einer HLLSKETCH-Spalte erstellen oder ändern.
- Datentypen für benutzerdefinierte Python-Funktionen (UDFs) unterstützen den Datentyp HLLSKETCH nicht. Weitere Informationen zu Python-UDFs finden Sie unter Erstellung einer skalaren Python-UDF.

Beispiele

Beispiel: Rückgabe der Kardinalität in einer Unterabfrage

Im folgenden Beispiel wird die Kardinalität für jede Skizze in einer Unterabfrage für eine Tabelle mit dem Namen Sales (Vertrieb) zurückgegeben.

```
CREATE TABLE Sales (customer VARCHAR, country VARCHAR, amount BIGINT);
INSERT INTO Sales VALUES ('David Joe', 'Greece', 14.5),  ('David Joe', 'Greece',
 19.95), ('John Doe', 'USA', 29.95), ('John Doe', 'USA', 19.95), ('George Spanos',
 'Greece', 9.95), ('George Spanos', 'Greece', 2.95);
```

Die folgende Abfrage generiert eine HLL-Skizze für die Kunden jedes Landes und extrahiert die Kardinalität. So werden die eindeutigen Kunden aus jedem Land angezeigt.

```
SELECT hll_cardinality(sketch), country
FROM (SELECT hll_create_sketch(customer) AS sketch, country
        FROM Sales
        GROUP BY country) AS hll_subquery;

hll_cardinality | country
----------------+---------
            1   | USA
            2   | Greece
 ...
```

Beispiel: Rückgabe eines HLLSKETCH-Typs von kombinierten Skizzen in einer Unterabfrage

Im folgenden Beispiel wird ein einzelner HLLSKETCH-Typ zurückgegeben, der die Kombination einzelner Skizzen aus einer Unterabfrage darstellt. Die Skizzen werden mit der Aggregationsfunktion HLL_COMBINE kombiniert.

```
SELECT hll_combine(sketch)
FROM (SELECT hll_create_sketch(customers) AS sketch
        FROM Sales
        GROUP BY country) AS hll_subquery

                                        hll_combine
-------------------------------------------------------------------------------------------
 {"version":1,"logm":15,"sparse":{"indices":[29808639,35021072,47612452],"values":
[1,1,1]}}
(1 row)
```

Beispiel: Rückgabe einer HyperLogLog Skizze aus der Kombination mehrerer Skizzen

Im folgenden Beispiel nehmen wir an, dass die Tabelle `page-users` die voraggregierten Skizzen für jede Seite speichert, die Benutzer auf einer bestimmten Website besucht haben. Jede Zeile in dieser Tabelle enthält eine HyperLogLog Skizze, die alle Benutzer-IDs darstellt, die die besuchten Seiten zeigen.

```
page_users
-- +----------------+-------------+--------------+
-- | _PARTITIONTIME | page         | sketch |
-- +----------------+-------------+--------------+
-- | 2019-07-28     | homepage     | CHAQkAQYA... |
-- | 2019-07-28     | Product A    | CHAQxPnYB... |
-- +----------------+-------------+--------------+
```

Im folgenden Beispiel werden die voraggregierten Skizzen zu einer einzelnen Skizze zusammengefasst. Diese Skizze kapselt die kollektive Kardinalität, die jede Skizze kapselt.

```
SELECT hll_combine(sketch) as sketch
FROM page_users
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
-- +----------------------------------------+
-- | sketch |
-- +----------------------------------------+
-- | CHAQ3sGoCxgCIAuCB4iAIBgTIBgqgIAgAwY.... |
-- +----------------------------------------+
```

Wenn eine neue Skizze erstellt wird, können Sie die Funktion HLL_CARDINALITY verwenden, um die kollektiven eindeutigen Werte abzurufen, wie im Folgenden gezeigt.

```
SELECT hll_cardinality(sketch)
FROM (
  SELECT
  hll_combine(sketch) as sketch
  FROM page_users
) AS hll_subquery
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
-- +-------+
-- | count |
-- +-------+
-- | 54356 |
-- +-------+
```

Beispiel: Generieren Sie mithilfe externer HyperLogLog Tabellen Skizzen anhand von S3-Daten

In den folgenden Beispielen werden HyperLogLog Skizzen zwischengespeichert, um zu vermeiden, dass zur Kardinalitätsschätzung direkt auf Amazon S3 zugegriffen wird.

Sie können HyperLogLog Skizzen vorab aggregieren und in externen Tabellen zwischenspeichern, die für Amazon S3 S3-Daten definiert sind. Auf diese Weise können Sie Kardinalitätsschätzungen extrahieren, ohne auf die zugrundeliegenden Basisdaten zuzugreifen.

Angenommen, Sie haben eine Reihe von Textdateien mit Tabulatortrennzeichen in Amazon S3 entladen. Sie führen die folgende Abfrage aus, um eine externe Tabelle mit Namen `sales` im externen Amazon-Redshift-Schema mit dem Namen `spectrum` zu definieren. Der Amazon S3 S3-Bucket für dieses Beispiel befindet sich im Osten der USA (Nord-Virginia) AWS-Region.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid smallint,
buyerid smallint,
eventid integer,
dateid integer,
qtysold integer,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t' stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/';
```

Angenommen, Sie möchten die einzelnen Käufer berechnen, die einen Artikel zu einem beliebigen Datum gekauft haben. Dazu werden im folgenden Beispiel Skizzen für die Käufer-IDs für jeden Tag des Jahres generiert und das Ergebnis wird in der Amazon-Redshift-Tabelle `hll_sales` gespeichert.

```
CREATE TABLE hll_sales AS
SELECT saletime, hll_create_sketch(buyerid) AS sketch
FROM spectrum.sales
GROUP BY saletime;
```

```
SELECT TOP 5 * FROM hll_sales;
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
-- hll_sales

-- | saletime        | sketch
    |
-- +-----------------
+-------------------------------------------------------------------+
-- | 7/22/2008 8:30  | {"version":1,"logm":15,"sparse":{"indices":[9281416],"values":
[1]}}
-- | 2/19/2008 0:38  | {"version":1,"logm":15,"sparse":{"indices":[48735497],"values":
[3]}}
-- | 11/5/2008 4:49  | {"version":1,"logm":15,"sparse":{"indices":[27858661],"values":
[1]}}
-- | 10/27/2008 4:08 | {"version":1,"logm":15,"sparse":{"indices":[65295430],"values":
[2]}}
-- | 2/16/2008 9:37  | {"version":1,"logm":15,"sparse":{"indices":[56869618],"values":
[2]}}
-- +----------------
 +-------------------------------------------------------------------+
```

Die folgende Abfrage zeigt die geschätzte Anzahl verschiedener Käufer, die am Freitag nach Thanksgiving im Jahr 2008 einen Artikel gekauft haben.

```
SELECT hll_cardinality(hll_combine(sketch)) as distinct_buyers
FROM hll_sales
WHERE trunc(saletime) = '2008-11-28';
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
distinct_buyers
---------------
386
```

Angenommen, Sie möchten die Anzahl der eindeutigen Benutzer, die einen Artikel an einem bestimmten Datumsbereich gekauft haben abrufen. Der Zeitraum könnte sich zum Beispiel vom Black Friday bis zum darauffolgenden Montag erstrecken. Um diese Information abzurufen, verwendet die folgende Abfrage die Aggregationsfunktion `hll_combine`. Mit dieser Funktion können Sie

vermeiden, dass Käufer doppelt gezählt werden, die einen Artikel an mehr als einem Tag des ausgewählten Bereichs gekauft haben.

```
SELECT hll_cardinality(hll_combine(sketch)) as distinct_buyers
FROM hll_sales
WHERE saletime BETWEEN '2008-11-28' AND '2008-12-01';
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
distinct_buyers
---------------
1166
```

Um die `hll_sales` Tabelle beizubehalten up-to-date, führen Sie am Ende jedes Tages die folgende Abfrage aus. Dadurch wird auf der Grundlage der IDs der Käufer, die heute einen Artikel gekauft haben, eine HyperLogLog Skizze erstellt und der `hll_sales` Tabelle hinzugefügt.

```
INSERT INTO hll_sales
SELECT saletime, hll_create_sketch(buyerid)
FROM spectrum.sales
WHERE TRUNC(saletime) = to_char(GETDATE(), 'YYYY-MM-DD')
GROUP BY saletime;
```

Datenbankübergreifendes Abfragen von Daten

Durch Verwendung von datenbankübergreifenden Abfragen in Amazon Redshift können Sie datenbankübergreifend in einem Amazon-Redshift-Cluster abfragen. Bei datenbankübergreifenden Abfragen können Sie Daten aus jeder Datenbank im Amazon-Redshift-Cluster abfragen, unabhängig davon, mit welcher Datenbank Sie verbunden sind. Datenbankübergreifende Abfragen eliminieren Datenkopien und vereinfachen Ihre Datenorganisation, um mehrere Geschäftsgruppen aus demselben Data Warehouse zu unterstützen.

Mit datenbankübergreifenden Abfragen können Sie folgende Aktionen durchführen:

- Daten datenbankübergreifend in Ihrem Amazon-Redshift-Cluster abfragen.

  Sie können nicht nur Datenbanken abfragen, mit denen Sie verbunden sind, sondern auch aus allen anderen Datenbanken lesen, für die Sie über Berechtigungen verfügen.

  Wenn Sie Datenbankobjekte in anderen nicht verbundenen Datenbanken abfragen, haben Sie nur Lesezugriff auf diese Datenbankobjekte. Sie können datenbankübergreifende Abfragen verwenden, um auf Daten aus allen Datenbanken Ihres Amazon-Redshift-Clusters zuzugreifen, ohne eine Verbindung zu dieser bestimmten Datenbank herstellen zu müssen. Auf diese Weise können Sie schnell und einfach Daten abfragen und verknüpfen, die über mehrere Datenbanken in Ihrem Amazon-Redshift-Cluster verteilt sind.

  Sie können auch Datasets aus mehreren Datenbanken in einer einzigen Abfrage verknüpfen und die Daten mithilfe von Business Intelligence (BI) oder Analysetools analysieren. Sie können weiterhin detaillierte Zugriffskontrollen auf Tabellenebene für Benutzer einrichten, indem Sie die Standardbefehle von Amazon Redshift SQL verwenden. Auf diese Weise können Sie sicherstellen, dass Benutzer nur die relevanten Teilmengen der Daten sehen, für die sie über Berechtigungen verfügen.
- Abfragen von Objekten.

  Sie können andere Datenbankobjekte mit vollqualifizierten Objektnamen abfragen, die mit der dreiteiligen Schreibweise ausgedrückt werden. Der vollständige Pfad zu einem Datenbankobjekt besteht aus drei Komponenten: Datenbankname, Schema und Name des Objekts. Sie können auf jedes Objekt von jeder anderen Datenbank aus zugreifen, indem Sie die vollständige Pfadnotation *database_name*.*schema_name*.*object_name* verwenden. Um auf eine bestimmte Spalte zuzugreifen, verwenden Sie *database_name*.*schema_name*.*object_name*.*column_name*.

Sie können auch einen Alias für ein Schema in einer anderen Datenbank erstellen, indem Sie die externe Schemanotation verwenden. Dieses externe Schema verweist auf eine andere Datenbank und ein Schemapaar. Die Abfrage kann mit der externen Schemanotation *external_schema_name*.*object_name* auf das andere Datenbankobjekt zugreifen.

In derselben schreibgeschützten Abfrage können Sie verschiedene Datenbankobjekte wie Benutzertabellen, reguläre Ansichten, materialisierte Ansichten und späte Bindungsansichten aus anderen Datenbanken abfragen.

- Verwalten von Berechtigungen.

  Benutzer mit Zugriffsberechtigungen für Objekte in Datenbanken in einem Amazon-Redshift-Cluster können diese Objekte abfragen. Sie erteilen Berechtigungen für Benutzer und Benutzergruppen mit dem Befehl GRANT. Sie können Berechtigungen auch mit dem Befehl REVOKE aufheben, wenn ein Benutzer keinen Zugriff mehr auf bestimmte Datenbankobjekte benötigt.

- Arbeiten Sie mit Metadaten und BI-Tools.

  Sie können ein externes Schema erstellen, um auf ein Schema in einer anderen Amazon-Redshift-Datenbank innerhalb desselben Amazon-Redshift-Clusters zu verweisen. Weitere Informationen zum Befehl finden Sie unter CREATE EXTERNAL SCHEMA.

  Nachdem externe Schemareferenzen erstellt wurden, zeigt Amazon Redshift die Tabellen unter dem Schema der anderen Datenbank inSVV_EXTERNAL_TABLES und SVV_EXTERNAL_COLUMNS für die Tools, um die Metadaten zu erkunden.

  Um datenbankübergreifende Abfrage in BI-Tools zu integrieren, können Sie die folgenden Systemansichten verwenden. Diese helfen Ihnen beim Anzeigen von Informationen zu den Metadaten von Objekten in den verbundenen Datenbanken und anderen Datenbanken im Amazon-Redshift-Cluster.

  Im Folgenden finden Sie Systemansichten, die alle Amazon-Redshift-Objekte und externen Objekte aller Datenbanken in Ihrem Amazon-Redshift-Cluster anzeigen:

  - SVV_ALL_COLUMNS
  - SVV_ALL_SCHEMAS
  - SVV_ALL_TABLES

Im Folgenden finden Sie Systemansichten, die alle Amazon-Redshift-Objekte aller Datenbanken in Ihrem Amazon-Redshift-Cluster anzeigen:

- SVV_REDSHIFT_COLUMNS
- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES

Themen

Überlegungen

Wenn Sie mit der Funktion für datenbankübergreifende Abfragen in Amazon Redshift arbeiten, sollten Sie Folgendes beachten:

- Amazon Redshift unterstützt datenbankübergreifende Abfrage für die Knotentypen ra3.4xlarge, ra3.16xlarge und ra3.xlplus.
- Amazon Redshift unterstützt das Verbinden von Daten aus Tabellen oder Ansichten über eine oder mehrere Datenbanken im selben Amazon-Redshift-Cluster.
- Amazon Redshift Serverless unterstützt dieselben datenbankübergreifenden Funktionen wie Amazon-Redshift-Cluster. Sie können also Daten aus Tabellen oder Ansichten über eine oder mehrere Datenbanken hinweg in einem Serverless-Namespace verknüpfen.
- Alle Abfragen in einer Transaktion in der verbundenen Datenbank lesen Daten im gleichen Zustand der anderen Datenbank wie die Daten zu Beginn der Transaktion. Dieser Ansatz hilft bei der Bereitstellung von Abfragetransaktionskonsistenz über Datenbanken hinweg. Amazon Redshift unterstützt transaktionale Konsistenz für datenbankübergreifende Abfragen.
- Um Metadaten über Datenbanken hinweg abzurufen, verwenden Sie die Metadatenansichten SVV_ALL* und SVV_REDSHIFT*. Sie können die dreiteilige Notation oder externe Schemas nicht verwenden, um datenbankübergreifende Metadatentabellen oder -ansichten unter information_schema und pg_catalog abzufragen.

Einschränkungen

Wenn Sie mit der Funktion für datenbankübergreifende Abfragen in Amazon Redshift arbeiten, sollten Sie folgende Einschränkungen beachten:

- Wenn Sie Datenbankobjekte in anderen nicht verbundenen Datenbanken abfragen, haben Sie nur Lesezugriff auf diese Datenbankobjekte.
- Sie können keine Ansichten abfragen, die in anderen Datenbanken erstellt wurden, die sich auf Objekte einer anderen Datenbank beziehen.
- Sie können nur spätbindende und materialisierte Ansichten für Objekte anderer Datenbanken im Cluster erstellen. Sie können keine regulären Ansichten für Objekte anderer Datenbanken im Cluster erstellen.
- Amazon Redshift unterstützt keine Tabellen mit Berechtigungen auf Spaltenebene für datenbankübergreifende Abfragen.
- Amazon Redshift unterstützt keine Abfragekatalogobjekte in AWS Glue oder verbundenen Datenbanken. Um diese Objekte abzufragen, erstellen Sie zunächst externe Schemas, die auf diese externen Datenquellen in jeder Datenbank verweisen.
- Das Ausführen von datenbankübergreifenden Abfragen für Tabellen mit verschachtelten Sortierschlüsseln wird nicht unterstützt.

Beispiele für die Verwendung einer datenbankübergreifenden Abfrage

Anhand der folgenden Beispiele erfahren Sie, wie Sie eine datenbankübergreifende Abfrage einrichten, die auf eine Amazon-Redshift-Datenbank verweist.

Erstellen Sie zum Starten die Datenbanken `db1` und `db2` sowie die Benutzer `user1` und `user2` in Ihrem Amazon-Redshift-Cluster. Weitere Informationen erhalten Sie unter CREATE DATABASE und CREATE USER.

```
--As user1 on db1
CREATE DATABASE db1;

CREATE DATABASE db2;

CREATE USER user1 PASSWORD 'Redshift01';
```

```
CREATE USER user2 PASSWORD 'Redshift01';
```

Erstellen Sie als `user1` in db1 eine Tabelle, erteilen Sie Zugriffsberechtigungen für `user2` und fügen Sie Werte in `table1` ein. Weitere Informationen erhalten Sie unter GRANT und INSERT.

```
--As user1 on db1
CREATE TABLE table1 (c1 int, c2 int, c3 int);

GRANT SELECT ON table1 TO user2;

INSERT INTO table1 VALUES (1,2,3),(4,5,6),(7,8,9);
```

Führen Sie als `user2` in db2 eine datenbankübergreifende Abfrage in db2unter Verwendung der dreiteiligen Notation durch.

```
--As user2 on db2
SELECT * from db1.public.table1 ORDER BY c1;
c1 | c2  | c3
---+-----+----
1  |  2  | 3
4  |  5  | 6
7  |  8  | 9
(3 rows)
```

Erstellen Sie als `user2` in db2 ein externes Schema und führen Sie eine datenbankübergreifende Abfrage in db2 unter Verwendung der externen Schemanotation durch.

```
--As user2 on db2
CREATE EXTERNAL SCHEMA db1_public_sch
FROM REDSHIFT DATABASE 'db1' SCHEMA 'public';

SELECT * FROM db1_public_sch.table1 ORDER BY c1;

c1  | c2 | c3
----+----+----
1   | 2  | 3
4   | 5  | 6
7   | 8  | 9
(3 rows)
```

Gehen Sie wie folgt vor, um als `user1` in `db1` verschiedene Ansichten zu erstellen und diesen Ansichten Berechtigungen zu erteilen.

```
--As user1 on db1
CREATE VIEW regular_view AS SELECT c1 FROM table1;

GRANT SELECT ON regular_view TO user2;


CREATE MATERIALIZED VIEW mat_view AS SELECT c2 FROM table1;

GRANT SELECT ON mat_view TO user2;


CREATE VIEW late_bind_view AS SELECT c3 FROM public.table1 WITH NO SCHEMA BINDING;

GRANT SELECT ON late_bind_view TO user2;
```

Führen Sie als `user2` in `db2` die folgende datenbankübergreifende Abfrage mit der dreiteiligen Notation aus, um die bestimmte Ansicht anzuzeigen.

```
--As user2 on db2
SELECT * FROM db1.public.regular_view;
c1
----
1
4
7
(3 rows)

SELECT * FROM db1.public.mat_view;
c2
----
8
5
2
(3 rows)

SELECT * FROM db1.public.late_bind_view;
c3
----
3
6
```

```
9
(3 rows)
```

Führen Sie als user2 in db2 die folgende datenbankübergreifende Abfrage mithilfe der externen Schemanotation aus, um die späte Bindungsansicht abzufragen.

```
--As user2 on db2
SELECT * FROM db1_public_sch.late_bind_view;
c3
----
3
6
9
(3 rows)
```

Führen Sie als user2 in db2 den folgenden Befehl aus, um verbundene Tabellen in einer einzelnen Abfrage zu verwenden.

```
--As user2 on db2
CREATE TABLE table1 (a int, b int, c int);

INSERT INTO table1 VALUES (1,2,3), (4,5,6), (7,8,9);

SELECT a AS col_1, (db1.public.table1.c2 + b) AS sum_col2, (db1.public.table1.c3 + c)
 AS sum_col3 FROM db1.public.table1, table1 WHERE db1.public.table1.c1 = a;
col_1 | sum_col2 | sum_col3
------+----------+----------
1     | 4        | 6
4     | 10       | 12
7     | 16       | 18
(3 rows)
```

Im folgenden Beispiel werden alle Datenbanken auf dem Cluster aufgelistet.

```
select database_name, database_owner, database_type
from svv_redshift_databases
where database_name in ('db1', 'db2');

 database_name | database_owner | database_type
---------------+----------------+---------------
 db1           |            100 | local
 db2           |            100 | local
```

```
(2 rows)
```

Im folgenden Beispiel werden alle Amazon-Redshift-Schemas aller Datenbanken im Cluster aufgelistet.

```
select database_name, schema_name, schema_owner, schema_type
from svv_redshift_schemas
where database_name in ('db1', 'db2');

 database_name |    schema_name     | schema_owner | schema_type
---------------+--------------------+--------------+-------------
 db1           | pg_catalog         |            1 | local
 db1           | public             |            1 | local
 db1           | information_schema |            1 | local
 db2           | pg_catalog         |            1 | local
 db2           | public             |            1 | local
 db2           | information_schema |            1 | local
(6 rows)
```

Im folgenden Beispiel werden alle Amazon-Redshift-Tabellen aller Datenbanken im Cluster aufgelistet.

```
select database_name, schema_name, table_name, table_type
from svv_redshift_tables
where database_name in ('db1', 'db2') and schema_name in ('public');

 database_name | schema_name |     table_name      | table_type
---------------+-------------+---------------------+------------
 db1           | public      | late_bind_view      | VIEW
 db1           | public      | mat_view            | VIEW
 db1           | public      | mv_tbl__mat_view__0 | TABLE
 db1           | public      | regular_view        | VIEW
 db1           | public      | table1              | TABLE
 db2           | public      | table2              | TABLE
(6 rows)
```

Im folgenden Beispiel werden alle Amazon-Redshift- und externen Schemas aller Datenbanken im Cluster aufgelistet.

```
select database_name, schema_name, schema_owner, schema_type
from svv_all_schemas where database_name in ('db1', 'db2') ;
```

```
 database_name |    schema_name     | schema_owner | schema_type
---------------+--------------------+--------------+-------------
 db1           | pg_catalog         |            1 | local
 db1           | public             |            1 | local
 db1           | information_schema |            1 | local
 db2           | pg_catalog         |            1 | local
 db2           | public             |            1 | local
 db2           | information_schema |            1 | local
 db2           | db1_public_sch     |            1 | external
(7 rows)
```

Im folgenden Beispiel werden alle Amazon-Redshift- und externen Tabellen aller Datenbanken im Cluster aufgelistet.

```
select database_name, schema_name, table_name, table_type
from svv_all_tables
where database_name in ('db1', 'db2') and schema_name in ('public');

 database_name | schema_name |     table_name      | table_type
---------------+-------------+---------------------+------------
 db1           | public      | regular_view        | VIEW
 db1           | public      | mv_tbl__mat_view__0 | TABLE
 db1           | public      | mat_view            | VIEW
 db1           | public      | late_bind_view      | VIEW
 db1           | public      | table1              | TABLE
 db2           | public      | table2              | TABLE
(6 rows)
```

Verwendung datenbankübergreifender Abfragen mit dem Abfrage-Editor

Sie können datenbankübergreifende Abfragen verwenden, um auf Daten aus allen Datenbanken Ihres Amazon-Redshift-Clusters zuzugreifen, ohne eine Verbindung zu dieser bestimmten Datenbank herstellen zu müssen. Wenn Sie datenbankübergreifende Abfragen für andere nicht verbundene Datenbanken ausführen, haben Sie nur Lesezugriff auf diese Datenbankobjekte.

Sie können andere Datenbankobjekte mit vollqualifizierten Objektnamen abfragen, die mit der dreiteiligen Schreibweise ausgedrückt werden. Der vollständige Pfad zu einem Datenbankobjekt besteht aus drei Komponenten: Datenbankname, Schema und Name des Objekts. Ein Beispiel ist *database_name.schema_name.object_name*.

So verwenden Sie datenbankübergreifende Abfragen mit dem Abfrage-Editor v2

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Erstellen Sie einen Cluster, um datenbankübergreifende Abfragen im Abfrage-Editor v2 von Amazon Redshift zu verwenden. Weitere Informationen finden Sie unter Creating a cluster (Erstellen eines Cluster) im Amazon-Redshift-Verwaltungshandbuch.
3. Aktivieren Sie den Zugriff auf den Abfrage-Editor mit den entsprechenden Berechtigungen. Weitere Informationen finden Sie unter Abfragen einer Datenbank mit dem Abfrage-Editor v2 im Amazon-Redshift-Verwaltungshandbuch.
4. Wählen Sie im Navigationsmenü die Option Abfrage-Editor v2 und stellen Sie dann eine Verbindung zu einer Datenbank in Ihrem Cluster her.

   Wenn Sie zum ersten Mal eine Verbindung mit dem Abfrage-Editor v2 herstellen, zeigt Amazon Redshift standardmäßig die Ressourcen für die verbundene Datenbank an.
5. Wählen Sie die anderen Datenbanken aus, auf die Sie zugreifen können, um Datenbankobjekte für diese anderen Datenbanken anzuzeigen. Stellen Sie zum Anzeigen von Objekten sicher, dass Sie über die entsprechenden Berechtigungen verfügen. Nachdem Sie eine Datenbank ausgewählt haben, zeigt Amazon Redshift die Liste der Schemas aus der Datenbank an.

   Wählen Sie ein Schema aus, um die Liste der Datenbankobjekte innerhalb dieses Schemas anzuzeigen.

   > **Note**
   >
   > Amazon Redshift unterstützt keine Abfragekatalogobjekte, die Teil von AWS Glue - oder Verbunddatenbanken sind. Um diese abzufragen, erstellen Sie zunächst externe Schemas, die auf diese externen Datenquellen in jeder Datenbank verweisen. Datenbankübergreifende Amazon-Redshift-Abfragen mit der dreiteiligen Notation unterstützen keine Metadatentabellen unter den Schemas `information_schema` und `pg_catalog`, da diese Metadatenansichten spezifisch für eine Datenbank sind.
6. (Optional) Filtern Sie die Liste der Tabellen oder Ansichten für das ausgewählte Schema.

Daten in Amazon Redshift teilen

Mit Amazon Redshift Data Sharing können Sie den Zugriff auf Live-Daten sicher über Amazon Redshift Redshift-Cluster und Arbeitsgruppen hinweg teilen AWS-Konten, AWS-Regionen ohne die Daten manuell verschieben oder kopieren zu müssen. Da die Daten live sind, können alle Benutzer die meisten up-to-date und konsistentesten Informationen in Amazon Redshift sehen, sobald sie aktualisiert sind.

Sie können Daten für bereitgestellte Cluster, serverlose Arbeitsgruppen, Availability Zones und gemeinsam nutzen. AWS-Konten AWS-Regionen Die Daten können zwischen Cluster-Typen sowie zwischen bereitgestellten Clustern und Serverless geteilt werden.

Multi-Warehouse-Schreibvorgänge in Amazon Redshift (Vorschau)

Sie können Datenbankobjekte sowohl für Lese- als auch für Schreibvorgänge in verschiedenen Amazon Redshift-Clustern oder Amazon Redshift Serverless-Arbeitsgruppen innerhalb derselben AWS-Konto oder von einer zur anderen gemeinsam nutzen. AWS-Konto Sie können Daten auch regionsübergreifend schreiben. Sie können Berechtigungen wie SELECT, INSERT und UPDATE für verschiedene Tabellen und USAGE und CREATE für verschiedene Schemata gewähren. Die Daten sind live und für alle Warehouses verfügbar, sobald eine Schreibtransaktion festgeschrieben wurde.

Weitere Informationen zur Konfiguration der Funktionen für die gemeinsame Nutzung von Daten im Track PREVIEW_2023 finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau).

> **Note**
>
> Multi-Warehouse-Schreibvorgänge über gemeinsame Datennutzung sind derzeit auf ra3.xlplus-Clustern nicht verfügbar. Um diese Funktion zu verwenden, erstellen Sie ra3.4xl-Cluster, ra3.16xl-Cluster oder Amazon Redshift Serverless-Arbeitsgruppen.

Überblick über die Datenfreigabe in Amazon Redshift

Mit Data Sharing können Sie Live-Daten sicher und einfach über Amazon Redshift Redshift-Cluster hinweg teilen.

Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten und zur Verwaltung von Datenfreigaben mithilfe von finden Sie unter. AWS Management ConsoleVerwalten von Aufgaben bei der Datenfreigabe

Anwendungsfälle für Datenfreigabe mit Amazon Redshift

Die Amazon-Redshift-Datenfreigabe ist besonders für folgende Anwendungsfälle nützlich:

- Unterstützung verschiedener Arten geschäftskritischer Workloads – Verwenden Sie einen zentralen Extract, Transform, Load (ETL)-Cluster, der Daten mit mehreren Business Intelligence (BI) oder Analyse-Clustern gemeinsam verwendet. Dieser Ansatz bietet Lese-Workload-Isolation und Rückbelastung für einzelne Workloads. Sie können Ihre individuelle Workload-Rechenleistung entsprechend den Workload-spezifischen Preis- und Leistungsanforderungen anpassen und skalieren.
- Gruppenübergreifende Zusammenarbeit – Ermöglicht nahtlose Zusammenarbeit zwischen Teams und Unternehmensgruppen für umfassendere Analysen, Datenwissenschaft und produktübergreifende Wirkungsanalysen.
- Bereitstellung von Daten als Service – Teilen Sie Daten als Service in Ihrer gesamten Organisation.
- Freigabe von Daten zwischen Umgebungen – Teilen Sie Daten in Entwicklungs-, Test- und Produktionsumgebungen. Sie können die Teamagilität verbessern, indem Sie Daten auf verschiedenen Granularitätsstufen teilen.
- Lizenzierung des Zugriffs auf Daten in Amazon Redshift — Listet Amazon Redshift Redshift-Datensätze im AWS Data Exchange Katalog auf, die Kunden innerhalb von Minuten finden, abonnieren und abfragen können.

Anwendungsfälle für die gemeinsame Nutzung von Daten und Schreibzugriff (Vorschau)

Die gemeinsame Nutzung von Daten für Schreibvorgänge hat mehrere wichtige Anwendungsfälle:

- Geschäftsquellendaten auf dem Hersteller aktualisieren — Sie können Daten als Service in Ihrer gesamten Organisation gemeinsam nutzen, aber dann können Verbraucher auch Aktionen an den Quelldaten durchführen. Sie können beispielsweise up-to-date Rückwerte mitteilen oder den Empfang von Daten bestätigen. Dies sind nur einige mögliche Geschäftsanwendungsfälle.
- Fügen Sie zusätzliche Datensätze zum Hersteller hinzu — Verbraucher können den ursprünglichen Quelldaten Datensätze hinzufügen. Diese können bei Bedarf als vom Verbraucher stammend gekennzeichnet werden.

Spezifische Informationen zur Ausführung von Schreibvorgängen auf einer Datenfreigabe finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau).

Freigeben von Daten auf verschiedenen Ebenen in Amazon Redshift

Mit Amazon Redshift können Sie Daten auf verschiedenen Ebenen teilen. Diese Ebenen umfassen Datenbanken, Schemas, Tabellen, Ansichten (einschließlich regulärer, späterbindender und materialisierter Ansichten) sowie benutzerdefinierte SQL-Funktionen (UDFs). Sie können mehrere Datashares für eine bestimmte Datenbank erstellen. Ein Datashare kann Objekte aus mehreren Schemas in der Datenbank enthalten, für die eine Freigabe erfolgt.

Durch diese Flexibilität beim Teilen von Daten erhalten Sie eine differenzierte Zugriffskontrolle. Sie können dieses Steuerelement für verschiedene Benutzer und Unternehmen anpassen, die Zugriff auf Amazon-Redshift-Daten benötigen.

Verwalten der Datenkonsistenz in Amazon Redshift

Amazon Redshift bietet Transaktionskonsistenz auf allen Produzenten- und Verbraucherclustern und teilt up-to-date und konsistente Ansichten der Daten mit allen Verbrauchern.

Sie können Daten im Produzenten-Cluster kontinuierlich aktualisieren. Alle Abfragen in einem Konsumenten-Cluster innerhalb einer Transaktion lesen den gleichen Status der freigegebenen Daten. Amazon Redshift berücksichtigt nicht die Daten, die durch eine andere Transaktion im Produzenten-Cluster geändert wurden, die nach dem Beginn der Transaktion im Konsumenten-Cluster festgeschrieben wurden. Nachdem die Datenänderung im Produzenten-Cluster festgeschrieben wurde, können neue Transaktionen im Konsumenten-Cluster die aktualisierten Daten sofort abfragen.

Durch die starke Konsistenz werden die Risiken von Geschäftsberichten mit niedrigerer Detailtreue entfernt, die während der Freigabe von Daten möglicherweise ungültige Ergebnisse enthalten. Dieser Faktor ist besonders wichtig für Finanzanalysen oder wo die Ergebnisse verwendet werden könnten, um Datensätze vorzubereiten, die zum Trainieren von Modellen für Machine Learning verwendet werden.

Überlegungen zur Verwendung der Datenfreigabe in Amazon Redshift

Im Folgenden finden Sie Überlegungen für das Arbeiten mit der Amazon-Redshift-Datenfreigabe. Weitere Informationen zu Einschränkungen bei der Datenfreigabe finden Sie unter Einschränkungen für die Freigabe von Daten.

- Der regionsübergreifende Datenaustausch beinhaltet zusätzliche Gebühren für die regionsübergreifende Datenübertragung. Diese Datenübertragungsgebühren fallen nicht innerhalb derselben Region an, sondern nur regionsübergreifend. Weitere Informationen finden Sie unter Verwaltung der Kostenkontrolle für regionsübergreifende Datashares.
- Als Datashare-Benutzer stellen Sie weiterhin nur eine Verbindung zu Ihrer lokalen Cluster-Datenbank her. Sie können keine Verbindung zu den Datenbanken herstellen, die aus einem Datashare erstellt wurden, aber aus diesen Datenbanken lesen.
- Dem Konsumenten werden alle Gebühren für die Datenverarbeitung und die regionsübergreifende Datenübertragung in Rechnung gestellt, die für die Abfrage der Daten des Produzenten erforderlich sind. Dem Produzenten wird die zugrunde liegende Datenspeicherung in seinem bereitgestellten Cluster oder Serverless-Namespace in Rechnung gestellt.
- Die Leistung der Abfragen auf gemeinsam genutzten Daten hängt von der Rechenkapazität der Konsumenten-Cluster ab.

Verwalten der Cluster-Verschlüsselung

Um Daten gemeinsam nutzen zu können AWS-Konto, müssen sowohl die Produzenten- als auch die Verbrauchercluster verschlüsselt werden.

In Amazon Redshift können Sie die Datenbankverschlüsselung für Ihre Cluster aktivieren, um Data-at-Rest besser zu schützen. Wenn Sie die Verschlüsselung für einen Cluster aktivieren, werden die Datenblöcke und die Metadaten des Systems für den Cluster und Snapshots des Clusters verschlüsselt. Sie können die Verschlüsselung aktivieren, wenn Sie Ihren Cluster starten, oder Sie können einen unverschlüsselten Cluster so ändern, dass er AWS Key Management Service -Verschlüsselung (AWS KMS-Verschlüsselung) verwendet. Weitere Informationen zur Datenbankverschlüsselung von Amazon-Redshift finden Sie unter Amazon-Redshift-Datenbankverschlüsselung im Amazon-Redshift-Verwaltungshandbuch.

Um Daten während der Übertragung zu schützen, werden alle Daten während der Übertragung durch das Verschlüsselungsschema des Produzentenclusters verschlüsselt. Der Konsumenten-Cluster übernimmt dieses Verschlüsselungsschema, wenn Daten geladen werden. Der Konsumenten-Cluster arbeitet dann als normaler verschlüsselter Cluster. Die Kommunikation zwischen Produzent und Konsument wird ebenfalls mit einem Shared Key-Schema verschlüsselt. Weitere Informationen zur Verschlüsselung während der Übertragung finden Sie unter Verschlüsselung während der Übertragung.

Einschränkungen für die Freigabe von Daten

Nachfolgend sehen Sie die Einschränkungen bei der Verwendung von Datashares in Amazon Redshift:

- Die gemeinsame Nutzung von Daten wird für alle bereitgestellten ra3-Clustertypen (ra3.16xlarge, ra3.4xlarge und ra3.xlplus) und Amazon Redshift Serverless unterstützt. Es wird für andere Clustertypen nicht unterstützt.
- Für die konto- und regionsübergreifende Datenfreigabe müssen sowohl der Produzenten- und der Konsumenten-Cluster als auch die Serverless-Namespaces verschlüsselt sein. Dies dient der Sicherheit. Sie müssen jedoch nicht den gleichen Verschlüsselungsschlüssel verwenden.
- Sie können SQL-UDFs nur über Datashares freigeben. Python- und Lambda-UDFs werden nicht unterstützt.
- Wenn die Produzentendatenbank über eine bestimmte Kollation verfügt, verwenden Sie dieselben Kollationseinstellungen für die Konsumentendatenbank.
- Amazon Redshift unterstützt nicht das Hinzufügen von externen Schemata, Tabellen oder spätbindenden Ansichten in externen Tabellen zu Datashares.
- Amazon Redshift unterstützt keine verschachtelten benutzerdefinierten SQL-Funktionen auf Erzeugerclustern.
- Amazon Redshift unterstützt keine Freigabe von Tabellen mit verschachtelten Sortierschlüsseln und Ansichten, die sich auf Tabellen mit verschachtelten Sortierschlüsseln beziehen.
- Konsumenten können Datashare-Objekte keinem anderen Datashare hinzufügen. Darüber hinaus können Konsumenten keine Ansichten, die auf Datashare-Objekte verweisen, einem anderen Datashare hinzufügen.
- Amazon Redshift unterstützt nicht den Zugriff auf ein Datashare-Objekt, bei dem eine gleichzeitige DDL zwischen der Vorbereitung und der Ausführung des Zugriffs aufgetreten ist.
- Amazon Redshift unterstützt die gemeinsame Nutzung von gespeicherten Prozeduren im Rahmen von Datashares nicht.
- Amazon Redshift unterstützt die gemeinsame Nutzung von Metadaten, Systemansichten und Systemtabellen nicht.

Regionen, in denen Datenfreigabe verfügbar ist

In der folgenden Tabelle ist die Verfügbarkeit von Funktionen zur gemeinsamen Nutzung von Daten aufgeführt.

Region	Gemeinsame Nutzung von Daten in derselben Region	Regionsübergreifender Datenaustausch	AWS Lake Formation geregelter Datenaustausch
USA Ost (Nord-Virginia): (us-east-1)	Ja	Ja	Ja
USA Ost (Ohio): (us-east-2)	Ja	Ja	Ja
USA West (Nordkalifornien) (us-west-1)	Ja	Ja	Ja
USA West (Oregon): (us-west-2)	Ja	Ja	Ja
Asien-Pazifik (Mumbai): (ap-south-1)	Ja	Ja	Ja
Asien-Pazifik (Hyderabad) (ap-south-2)	Ja	Nein	Nein
Asien-Pazifik (Tokyo) (ap-northeast-1)	Ja	Ja	Ja
Asien-Pazifik (Singapur): (ap-southeast-1)	Ja	Ja	Ja
Asien-Pazifik (Sydney): (ap-southeast-2)	Ja	Ja	Ja
Asien-Pazifik (Jakarta); (ap-southeast-3)	Ja	Nein	Nein

Region	Gemeinsame Nutzung von Daten in derselben Region	Regionsübergreifender Datenaustausch	AWS Lake Formation geregelter Datenaustausch
Asien-Pazifik (Melbourne) (ap-southeast-4)	Ja	Nein	Nein
Asien-Pazifik (Seoul): (ap-northeast-2)	Ja	Ja	Ja
Asien-Pazifik (Osaka) (ap-northeast-3)	Ja	Nein	Nein
Afrika (Kapstadt) (af-south-1)	Ja	Ja	Nein
Kanada West (Calgary) (ca-west-1)	Ja	Nein	Nein
Kanada (Zentral): (ca-central-1)	Ja	Ja	Ja
Europa (Frankfurt) (eu-central-1)	Ja	Ja	Ja
Europa (Zürich) (eu-central-2)	Ja	Nein	Nein
Europa (Irland) (eu-west-1)	Ja	Ja	Ja
Europa (London) (eu-west-2)	Ja	Ja	Ja
Europa (Paris) (eu-west-3)	Ja	Ja	Ja

Region	Gemeinsame Nutzung von Daten in derselben Region	Regionsübergreifender Datenaustausch	AWS Lake Formation geregelter Datenaustausch
Europa (Mailand) (eu-south-1)	Ja	Nein	Nein
Europa (Spanien) (eu-south-2)	Ja	Nein	Nein
Europa (Stockholm) (eu-north-1)	Ja	Ja	Ja
Naher Osten (VAE) (me-central-1)	Ja	Nein	Nein
Naher Osten (Bahrain) (me-south-1)	Ja	Nein	Nein
Israel (Tel Aviv) (il-central-1)	Ja	Nein	Nein
Südamerika (São Paulo) (sa-east-1)	Ja	Ja	Ja
AWS GovCloud (US-Ost) (us-gov-east-1)	Ja	Nein	Ja
AWS GovCloud (US-West) (US-Regierung West-1)	Ja	Nein	Ja

Regionale Verfügbarkeit für Schreibvorgänge in mehreren Warehouses für die gemeinsame Nutzung von Daten

Im Track PREVIEW_2023 bietet die gemeinsame Nutzung von Daten die Möglichkeit für Schreibvorgänge und detailliertere gemeinsame Nutzung. Weitere Informationen zur Konfiguration dieser Funktionen finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau). Informationen

zu Regionen, in denen Vorschaufunktionen verfügbar sind, finden Sie unter Regionen, in denen die gemeinsame Nutzung von Daten verfügbar ist (Vorschau).

Was ist ein Datashare?

Ein Datashare ist die Einheit zum Freigeben von Daten in Amazon Redshift. Verwenden Sie Datashares, um Daten auf demselben AWS-Konto oder einem anderen Datenträger gemeinsam zu nutzen. AWS-Konten Sie können Daten auch für Lesezwecke in verschiedenen Amazon-Redshift-Clustern freigeben.

Jedes Datashare ist einer bestimmten Datenbank in Ihrem Amazon-Redshift-Cluster zugeordnet.

Ein Cluster-Administrator kann Datashares erstellen, um Daten mit anderen Clustern gemeinsam zu nutzen, die als ausgehende Datenfreigaben bezeichnet werden. Ein Verbraucher-Administrator kann Datashares von anderen Clustern empfangen, die als eingehende Datenfreigaben bezeichnet werden. Einzelheiten zu Herstellern und Verbrauchern finden Sie unter Hersteller und Verbraucher von Datashares.

Datashare-Objekte sind Objekte aus bestimmten Datenbanken in einem Cluster, die Erzeugercluster-Administratoren zu Datashares hinzufügen können, die für Datenverbraucher freigegeben werden. Datashare-Objekte sind schreibgeschützt für Datenverbraucher. Beispiele für Datashare-Objekte sind Tabellen, Ansichten und benutzerdefinierte Funktionen. Sie können Datashare-Objekte zu Datashares hinzufügen, während Sie Datashares erstellen oder bearbeiten.

Die Datenfreigabe funktioniert weiterhin, wenn die Cluster-Größe angepasst oder der Produzenten-Cluster angehalten wird.

Es gibt verschiedene Arten von Datashares.

Themen

- Standard-Datashares
- AWS Data Exchange Datashares
- Von AWS Lake Formation verwaltete Datashares
- Hersteller und Verbraucher von Datashares

Standard-Datashares

Mit Standard-Datashares können Sie Daten für bereitgestellte Cluster, serverlose Arbeitsgruppen, Availability Zones und gemeinsam nutzen. AWS-Konten AWS-Regionen Die Daten können zwischen Cluster-Typen sowie zwischen bereitgestellten Clustern und Amazon Redshift Serverless geteilt werden.

Beachten Sie bei der gemeinsamen Nutzung von Daten die folgenden bereitgestellten Cluster, den folgenden serverlosen Namespace und die folgenden Identifikatoren: AWS-Konto

- Bereitgestellte Cluster-Namespaces sind Kennungen, die bereitgestellten Amazon-Redshift-Cluster identifizieren. Bei der Erstellung des bereitgestellten Clusters wird automatisch ein Namespace Globally Unique Identifier (GUID) erstellt und an den Cluster angehängt. Ein Amazon-Ressourcenname (ARN) hat das Format arn:{partition}:redshift:{region}:{account-id}:namespace:{namespace-guid}. Sie können den Namespace eines bereitgestellten Clusters auf der Seite mit den Cluster-Details auf der Amazon-Redshift-Konsole sehen.

  Im Workflow für die Datenfreigabe werden der Namespace-GUID-Wert und die Cluster-Namespace-ARN verwendet, um Daten mit Clustern im AWS-Konto freizugeben. Sie können auch den Namespace für den aktuellen Cluster mithilfe der Funktion `current_namespace` finden.
- Serverless Namespaces sind Kennungen, die Amazon Redshift Serverless identifizieren. Bei der Erstellung von Amazon Redshift Serverless wird automatisch ein Namespace Globally Unique Identifier (GUID) erstellt und an die Instance angehängt. Ein Serverless-Namespace-ARN hat das Format arn:{partition}:redshift-serverless:{region}:{account-id}:namespace/{namespace-guid}.
- AWS-Konten können Nutzer von Datashares sein und werden jeweils durch eine 12-stellige ID repräsentiert. AWS-Konto

Berücksichtigen Sie für Standard-Datashares Folgendes:

- Wenn ein Produzenten-Cluster gelöscht wird, löscht Amazon Redshift die vom Produzenten-Cluster erstellten Datashares. Wenn ein Produzenten-Cluster gesichert und wiederhergestellt wird, bleiben die erstellten Datashares weiterhin auf dem wiederhergestellten Cluster erhalten. Berechtigungen für den Datashare, die auf anderen Clustern gewährt wurden, gelten auf dem wiederhergestellten Cluster jedoch nicht mehr. Erteilen Sie den gewünschten Konsumenten-Clustern erneut Nutzungsberechtigungen für Datashares. Die Konsumenten-Datenbank auf dem Konsumenten-Cluster verweist auf den Datashare aus dem ursprünglichen Cluster, in dem der Snapshot erstellt wird. Um die freigegebenen Daten aus dem wiederhergestellten Cluster

abzufragen, erstellt der Administrator des Konsumenten-Clusters eine andere Datenbank. Oder der Administrator kann eine vorhandene Verbraucherdatenbank löschen und neu erstellen, um das Datashare aus dem neu wiederhergestellten Cluster zu verwenden.

- Wenn ein Konsumenten-Cluster gelöscht und aus einem Snapshot wiederhergestellt wird, ist der vorherige freigegebene Zugriff für diesen Cluster nicht mehr gültig und sichtbar. Wenn auf dem wiederhergestellten Konsumenten-Cluster weiterhin Zugriff auf Datashares erforderlich ist, muss der Administrator des Produzenten-Clusters dem wiederhergestellten Konsumenten-Cluster die Verwendung von Datashares erneut gewähren. Der Consumer-Cluster-Administrator muss alle veralteten VKonsumentendatenbanken löschen, die aus den inaktiven Datashares erstellt wurden. Dann muss der Administrator die Konsumentendatenbank aus dem Datashare neu erstellen, nachdem der Produzent die Berechtigungen erneut erteilt hat. Da sich die Cluster-Namespace-GUID auf einem wiederhergestellten Cluster von dem ursprünglichen Cluster unterscheidet, gewähren Sie Datashare-Berechtigungen erneut, wenn der Konsumenten- oder Produzenten-Cluster aus dem Backup wiederhergestellt wird.

AWS Data Exchange Datashares

Ein AWS Data Exchange Datashare ist eine Lizenzeinheit für die gemeinsame Nutzung Ihrer Daten. AWS Data Exchange AWS verwaltet alle Abrechnungen und Zahlungen im Zusammenhang mit Abonnements AWS Data Exchange und der Nutzung von Amazon Redshift Data Sharing. Zugelassene Datenanbieter können AWS Data Exchange Datashares zu Produkten hinzufügen. AWS Data Exchange Wenn Kunden ein Produkt mit AWS Data Exchange Datashares abonnieren, erhalten sie Zugriff auf die Datashares im Produkt.

AWS Data Exchange für Amazon Redshift macht es bequem, den Zugriff auf Ihre Amazon Redshift Redshift-Daten über zu lizenzieren. AWS Data Exchange Wenn ein Kunde ein Produkt mit AWS Data Exchange Datashares abonniert, AWS Data Exchange wird der Kunde automatisch als Datenverbraucher für alle AWS Data Exchange im Produkt enthaltenen Datenfreigaben hinzugefügt. Rechnungen werden automatisch generiert und Zahlungen werden zentral erfasst und automatisch ausgezahlt. AWS Marketplace Entitlement Service

Anbieter können Daten in Amazon Redshift auf granularer Ebene lizenzieren, wie z. B. Schemas, Tabellen, Ansichten und benutzerdefinierte Funktionen. Sie können denselben AWS Data Exchange Datenaustausch für mehrere Produkte verwenden. AWS Data Exchange Alle dem AWS Data Exchange Datashare hinzugefügten Objekte stehen Verbrauchern zur Verfügung. Produzenten können alle AWS Data Exchange Datenfreigaben einsehen, die in ihrem Namen mithilfe von AWS Data Exchange Amazon Redshift Redshift-API-Operationen, SQL-Befehlen und der Amazon Redshift

Redshift-Konsole verwaltet werden. Kunden, die Datashares eines Produkts abonnieren, haben nur Lesezugriff auf die Objekte in den AWS Data Exchange Datashares.

Kunden, die Herstellerdaten von Drittanbietern nutzen möchten, können den AWS Data Exchange Katalog durchsuchen, um Datensätze in Amazon Redshift zu finden und zu abonnieren. Nachdem ihr AWS Data Exchange Abonnement aktiv ist, können sie aus dem Datashare in ihrem Cluster eine Datenbank erstellen und die Daten in Amazon Redshift abfragen.

Wie funktionieren Datashares AWS Data Exchange

AWS Data Exchange Datashares als Producer-Administrator verwalten

Wenn Sie ein Datenproduzent sind (auch als Anbieter bezeichnet AWS Data Exchange), können Sie AWS Data Exchange Datashares erstellen, die eine Verbindung zu Ihren Amazon Redshift Redshift-Datenbanken herstellen. Um AWS Data Exchange Datashares zu Produkten auf hinzufügen zu können AWS Data Exchange, müssen Sie ein registrierter Anbieter sein. AWS Data Exchange

Weitere Informationen zu den ersten Schritten mit AWS Data Exchange Datashares finden Sie unter. Teilen lizenzierter Amazon Redshift Redshift-Daten auf AWS Data Exchange

AWS Data Exchange Datashares als Verbraucher mit einem aktiven Abonnement verwenden AWS Data Exchange

Wenn Sie ein Verbraucher mit einem aktiven AWS Data Exchange Abonnement sind (auch als Abonnent bezeichnet AWS Data Exchange), können Sie den AWS Data Exchange Katalog auf der AWS Data Exchange Konsole nach Produkten durchsuchen, die Datashares enthalten AWS Data Exchange .

Nachdem Sie ein Produkt abonniert haben, das AWS Data Exchange Datashares enthält, erstellen Sie eine Datenbank aus dem Datashare in Ihrem Cluster. Sie können die Daten in Amazon Redshift dann direkt abfragen, ohne die Daten extrahieren, transformieren und laden zu müssen.

Weitere Informationen zu den ersten Schritten mit AWS Data Exchange Datashares finden Sie unter. Teilen lizenzierter Amazon Redshift Redshift-Daten auf AWS Data Exchange

Für AWS Data Exchange -Datashares, beachten Sie Folgendes:

- Wenn ein Produzenten-Cluster gelöscht wird, löscht Amazon Redshift die vom Produzenten-Cluster erstellten Datashares. Wenn ein Produzenten-Cluster gesichert und wiederhergestellt wird, bleiben die erstellten Datashares weiterhin auf dem wiederhergestellten Cluster erhalten.

Damit Datenabonnenten weiterhin auf die Daten zugreifen können, müssen Sie die AWS Data Exchange Datashares erneut erstellen und sie in den Datensätzen des Produkts veröffentlichen. Die Konsumenten-Datenbank auf dem Konsumenten-Cluster verweist auf den Datashare aus dem ursprünglichen Cluster, in dem der Snapshot erstellt wird. Um die gemeinsam genutzten Daten aus dem wiederhergestellten Cluster abzufragen, erstellt der Administrator des Consumer-Clusters eine andere Datenbank oder löscht eine bestehende Consumer-Datenbank und erstellt sie neu, um die neu erstellte AWS Data Exchange Datenfreigabe aus dem neu wiederhergestellten Cluster zu verwenden.

- Wenn ein Konsumenten-Cluster gelöscht und aus einem Snapshot wiederhergestellt wird, bleibt der vorherige freigegebene Zugriff für diesen Cluster gültig und sichtbar. Der Administrator des Konsumenten-Clusters muss alle veralteten Konsumenten-Datenbanken löschen, die aus inaktiven Datashares erstellt wurden, und die Konsumenten-Datenbank aus dem Datashare neu erstellen, nachdem der Hersteller die Berechtigungen erneut erteilt hat. Da sich die Cluster-Namespace-GUID auf einem wiederhergestellten Cluster von dem ursprünglichen Cluster unterscheidet, gewähren Sie Datashare-Berechtigungen erneut, wenn der Produzenten-Cluster aus dem Backup wiederhergestellt wird.
- Wir empfehlen, dass Sie Ihren Cluster nicht löschen, wenn Sie über Datenfreigaben verfügen. AWS Data Exchange Diese Art der Änderung kann außerdem zu einer Verletzung der Datenproduktbedingungen in AWS Data Exchange führen.

Überlegungen bei der Verwendung AWS Data Exchange für Amazon Redshift

Beachten Sie bei der Verwendung AWS Data Exchange für Amazon Redshift Folgendes:

- Sowohl Produzenten als auch Konsumenten müssen die RA3-Instance-Typen nutzen, um Amazon-Redshift-Datashares verwenden zu können. Produzenten müssen die RA3-Instance-Typen mit der neuesten Amazon-Redshift-Clusterversion verwenden.
- Sowohl die Produzenten- als auch die Konsumenten-Cluster müssen verschlüsselt sein.
- Sie müssen als AWS Data Exchange Anbieter registriert sein, um Produkte anbieten zu können AWS Data Exchange, einschließlich Produkte, die AWS Data Exchange Datashares enthalten. Weitere Informationen finden Sie unter Erste Schritte als Anbieter.
- Sie müssen kein registrierter AWS Data Exchange Anbieter sein, um Amazon Redshift Redshift-Daten zu finden, zu abonnieren und abzufragen. AWS Data Exchange
- Um den Zugriff auf Ihre Daten zu kontrollieren, erstellen Sie AWS Data Exchange Datashares, bei denen die Einstellung „Öffentlich zugänglich“ aktiviert ist. Um eine AWS Data Exchange

Datenfreigabe so zu ändern, dass die Einstellung für öffentlich zugänglich deaktiviert wird, legen Sie die Sitzungsvariable so fest, dass ALTER DATASHARE SET PUBLICACCESSIBLE FALSE zulässig ist. Weitere Informationen finden Sie unter Nutzungshinweise für ALTER DATASHARE.

- Hersteller können Verbraucher nicht manuell zu Datenfreigaben hinzufügen oder aus ihnen entfernen, da der Zugriff auf die AWS Data Exchange Datenfreigaben nur gewährt wird, wenn sie über ein aktives Abonnement für ein Produkt verfügen, das den Datashare enthält. AWS Data Exchange AWS Data Exchange
- Produzenten können die SQL-Abfragen, die Konsumenten ausführen, nicht einsehen. Sie können lediglich Metadaten wie die Anzahl der Abfragen oder die Objekte, die Verbraucher abfragen, über Amazon-Redshift-Tabellen anzeigen, auf die nur der Produzent zugreifen kann. Weitere Informationen finden Sie unter Überwachen und Prüfen bei der Datenfreigabe in Amazon Redshift.
- Wir empfehlen Ihnen, Ihre Datashares öffentlich zugänglich zu machen. Wenn Sie dies nicht tun, können Abonnenten AWS Data Exchange mit öffentlich zugänglichen Verbraucherclustern Ihren Datashare nicht nutzen.
- Wir empfehlen, Datenfreigaben, die mit anderen geteilt wurden, nicht AWS-Konten mithilfe AWS Data Exchange der DROP DATASHARE-Anweisung zu löschen. Wenn Sie dies tun, verlieren diejenigen, AWS-Konten die Zugriff auf den Datashare haben, den Zugriff. Diese Aktion ist unumkehrbar. Diese Art der Änderung kann außerdem zu einer Verletzung der Datenproduktbedingungen in AWS Data Exchange führen. Wenn Sie ein AWS Data Exchange Datashare löschen möchten, finden Sie weitere Informationen unter. Nutzungshinweise für DROP DATASHARE
- Für die regionsübergreifende gemeinsame Nutzung von Daten können Sie AWS Data Exchange Datashares erstellen, um lizenzierte Daten gemeinsam zu nutzen.
- Beim Verbrauchen von Daten aus einer anderen Region zahlt der Verbraucher die regionsübergreifende Datenübertragungsgebühr von der Produzentenregion zur Konsumentenregion.

Von AWS Lake Formation verwaltete Datashares

Mithilfe AWS Lake Formation können Sie Zugriffsberechtigungen für Amazon Redshift Redshift-Datenfreigaben auf Datenbank-, Tabellen-, Spalten- und Zeilenebene zentral definieren und durchsetzen und den Benutzerzugriff auf Objekte innerhalb einer Datenfreigabe einschränken. Wenn Sie Daten über Lake Formation freigeben, können Sie Berechtigungen in Lake Formation definieren und diese auf jedes Datashare und seine Objekte anwenden. Wenn Sie beispielsweise eine Tabelle mit Mitarbeiterinformationen haben, können Sie mithilfe der Filter auf Spaltenebene von

Lake Formation verhindern, dass Mitarbeiter, die nicht in der Personalabteilung arbeiten, persönlich identifizierbare Informationen (PII) wie beispielsweise Sozialversicherungsnummern sehen. Weitere Informationen zu Datenfiltern finden Sie unter Datenfilterung und Sicherheit auf Zellebene in Lake Formation im Entwicklerhandbuch für AWS Lake Formation .

Sie können in Lake Formation auch Tags verwenden, um Berechtigungen für Lake-Formation-Ressourcen zu konfigurieren. Weitere Informationen finden Sie unter Tag-basierte Zugriffskontrolle in Lake Formation.

Amazon Redshift unterstützt derzeit die Datenfreigabe über Lake Formation, wenn die Freigabe innerhalb desselben Kontos oder kontenübergreifend erfolgt. Eine regionsübergreifende Freigabe wird zurzeit nicht unterstützt.

Nachfolgend finden Sie allgemeine Informationen zur Verwendung von Lake Formation zur Steuerung von Datashare-Berechtigungen:

1. In Amazon Redshift erstellt der Administrator des Produzenten-Clusters oder der Arbeitsgruppe ein Datashare auf dem Produzenten-Cluster oder der Arbeitsgruppe und gewährt einem Lake-Formation-Konto die Nutzungsberechtigung hierfür.
2. Der Administrator des Produzenten-Clusters oder der Arbeitsgruppe berechtigt das Lake-Formation-Konto zum Zugriff auf das Datashare.
3. Der Lake-Formation-Administrator ermittelt und registriert die Datashares. Sie müssen auch die AWS Glue ARNs ermitteln, auf die sie Zugriff haben, und die Datenfreigaben einem ARN zuordnen. AWS Glue Data Catalog Wenn Sie das verwenden, können AWS CLI Sie Datenfreigaben mit den Redshift-CLI-Operationen und erkennen und akzeptieren. `describe-data-shares` `associate-data-share-consumer` Verwenden Sie die CLI Operation `register-resource` in Lake Formation, um ein Datashare zu registrieren.
4. Der Lake Formation-Administrator erstellt eine Verbunddatenbank im und konfiguriert Lake Formation Formation-Berechtigungen AWS Glue Data Catalog, um den Benutzerzugriff auf Objekte innerhalb des Datashare zu steuern. Weitere Informationen zu Verbunddatenbanken finden Sie unter Verwaltung von Berechtigungen für Daten in AWS Glue einer Amazon Redshift Redshift-Datenfreigabe.
5. Der Lake Formation-Administrator erkennt die AWS Glue Datenbanken, auf die er Zugriff hat, und ordnet den Datashare einem ARN zu. AWS Glue Data Catalog
6. Der Redshift-Administrator erkennt die AWS Glue Datenbank-ARNs, auf die er Zugriff hat, erstellt mithilfe eines Datenbank-ARN eine externe Datenbank im Amazon Redshift Redshift-Consumer-Cluster und gewährt AWS Glue Datenbankbenutzern, die mit IAM-Anmeldeinformationen

authentifiziert sind, die Nutzung, um mit der Abfrage der Amazon Redshift Redshift-Datenbank zu beginnen.

7. Datenbankbenutzer können die Ansichten SVV_EXTERNAL_TABLES und SVV_EXTERNAL_COLUMNS verwenden, um alle Tabellen oder Spalten in der AWS Glue Datenbank zu finden, auf die sie Zugriff haben, und dann die Tabellen der AWS Glue Datenbank abfragen.
8. Wenn der Administrator des Produzenten-Clusters oder der Arbeitsgruppe beschließt, die Daten nicht mehr für den Konsumenten-Cluster freizugeben, kann er die Nutzung des Datashare widerrufen, die Autorisierung aufheben oder das Datashare in Redshift löschen. Die zugehörigen Berechtigungen und Objekte in Lake Formation werden nicht automatisch gelöscht.

Weitere Hinweise zur gemeinsamen Nutzung einer Datenfreigabe mit einem Producer-Cluster- oder Arbeitsgruppenadministrator finden Sie unter. AWS Lake Formation Arbeiten mit von Lake Formation verwalteten Datashares als Produzent Informationen zur Verwendung der gemeinsam genutzten Daten aus dem Produzenten-Cluster oder der Arbeitsgruppe finden Sie unter Arbeiten mit von Lake Formation verwalteten Datashares als Konsument.

Überlegungen und Einschränkungen bei der Verwendung AWS Lake Formation mit Amazon Redshift

Im Folgenden finden Sie Überlegungen und Einschränkungen in Bezug auf die Freigabe von Amazon-Redshift-Daten über Lake Formation. Informationen zu Überlegungen und Einschränkungen für die Datenfreigabe finden Sie unter Überlegungen zur Freigabe von Daten in Amazon Redshift. Informationen zu den Einschränkungen von Lake Formation finden Sie unter Hinweise zur Arbeit mit Amazon-Redshift-Datashares in Lake Formation.

- Die regionsübergreifende Freigabe eines Datashares an Lake Formation wird derzeit nicht unterstützt.
- Wenn Filter auf Spaltenebene für einen Benutzer in einer freigegebenen Beziehung definiert sind, werden beim Ausführen der Operation `SELECT *` nur die Spalten zurückgegeben, auf die der Benutzer Zugriff hat.
- Filter auf Zellenebene von Lake Formation werden nicht unterstützt.
- Wenn Sie eine Ansicht und die zugehörigen Tabellen erstellt und für Lake Formation freigegeben haben, können Sie Filter konfigurieren, um den Tabellenzugriff zu verwalten. Amazon Redshift setzt die von Lake Formation definierten Richtlinien durch, wenn Benutzer eines Konsumenten-Clusters auf freigegebene Objekte zugreifen. Wenn ein Benutzer auf eine für Lake Formation

freigegebene Ansicht zugreift, setzt Redshift nur die für die Ansicht definierten Lake-Formation-Richtlinien durch, und nicht die für die in der Ansicht enthaltenen Tabellen. Wenn Benutzer jedoch direkt auf die Tabelle zugreifen, setzt Redshift die definierten Lake-Formation-Richtlinien für die Tabelle durch.

- Sie können keine materialisierten Ansichten für den Verbraucher auf der Grundlage einer freigegebenen Tabelle erstellen, wenn für die Tabelle Lake-Formation-Filter konfiguriert sind.
- Der Lake-Formation-Administrator muss über Berechtigungen für Data-Lake-Administratoren und die erforderlichen Berechtigungen zum Akzeptieren eines Datashare verfügen.
- Der Produzenten-Konsumenten-Cluster muss ein RA3-Cluster mit der neuesten Amazon-Redshift-Clusterversion oder eine Serverless-Arbeitsgruppe sein, um Datashares über Lake Formation freigeben zu können.
- Sowohl die Produzenten- als auch die Konsumenten-Cluster müssen verschlüsselt sein.
- Die im Produzenten-Cluster oder der Arbeitsgruppe implementierten Redshift-Zugriffskontrollrichtlinien auf Zeilen- und Spaltenebene werden ignoriert, wenn das Datashare für Lake Formation freigegeben wird. Der Lake-Formation-Administrator muss diese Richtlinien in Lake Formation konfigurieren. Der Administrator des Produzenten-Clusters oder der Arbeitsgruppe kann RLS für eine Tabelle mithilfe des Befehls ALTER TABLE deaktivieren.
- Die Freigabe von Datashares über Lake Formation ist nur für Benutzer verfügbar, die sowohl Zugriff auf Redshift als auch auf Lake Formation haben.

Hersteller und Verbraucher von Datashares

Datenproduzenten (auch bekannt als Datenfreigabeproduzent oder Datashare-Produzent) sind Cluster, in denen Sie Daten freigeben möchten. Produzentencluster-Administratoren und Datenbankbesitzer können Datashares mit dem Befehl CREATE DATASHARE erstellen. Sie können Objekte wie Schemas, Tabellen, Ansichten und benutzerdefinierte SQL-Funktionen (UDFs) aus einer Datenbank hinzufügen, die der Producer-Cluster mit Consumer-Clustern gemeinsam nutzen soll.

Datenproduzenten (auch als Anbieter bezeichnet AWS Data Exchange) für AWS Data Exchange Datashares können Daten lizenzieren. AWS Data Exchange Zugelassene Anbieter können AWS Data Exchange Datashares zu Produkten hinzufügen. AWS Data Exchange

Wenn ein Kunde ein Produkt mit AWS Data Exchange Datashares abonniert, AWS Data Exchange wird der Kunde automatisch als Datenverbraucher für alle AWS Data Exchange im Produkt enthaltenen Datenfreigaben hinzugefügt. AWS Data Exchange entfernt außerdem alle Kunden aus AWS Data Exchange Datashares, wenn ihr Abonnement endet. AWS Data Exchange verwaltet

außerdem automatisch Abrechnung, Rechnungsstellung, Zahlungseinzug und Zahlungsverteilung für kostenpflichtige Produkte mit Datashares. AWS Data Exchange Weitere Informationen finden Sie unter AWS Data Exchange Datashares. Weitere Informationen zur Registrierung als AWS Data Exchange -Datenanbieter finden Sie unter Erste Schritte als Anbieter.

Datenkonsumenten (auch bekannt als Datenfreigabekonsumenten oder Datashare-Konsumenten) sind Cluster, die Datashares von Erzeugerclustern empfangen.

Amazon Redshift Redshift-Cluster, die Daten gemeinsam nutzen, können sich auf demselben, einem anderen AWS-Konten oder einem anderen System befinden AWS-Regionen, sodass Sie Daten organisationsübergreifend gemeinsam nutzen und mit anderen Parteien zusammenarbeiten können. Konsumenten-Cluster-Administratoren erhalten den Datashare, für den sie verwendet werden, und überprüfen den Inhalt der einzelnen Datenspeicher. Um gemeinsam genutzte Daten zu nutzen, erstellt der Konsumenten-Cluster-Administrator eine Amazon-Redshift-Datenbank aus dem Datashare. Der Administrator weist dann Benutzern und Rollen im Konsumenten-Cluster Berechtigungen für die Datenbank zu. Nachdem Berechtigungen gewährt wurden, können Benutzer und Rollen die gemeinsam genutzten Objekte als Teil der standardmäßigen Metadatenabfragen zusammen mit den lokalen Daten im Konsumenten-Cluster auflisten. Sie können sofort mit der Abfrage beginnen.

Wenn Sie ein Verbraucher mit einem aktiven AWS Data Exchange Abonnement sind (auch als Abonnenten bezeichnet AWS Data Exchange), können Sie detaillierte up-to-date Daten in Amazon Redshift suchen, abonnieren und abfragen, ohne die Daten extrahieren, transformieren und laden zu müssen. Weitere Informationen finden Sie unter AWS Data Exchange Datashares.

Funktionsweise der Datenfreigabe in Amazon Redshift

Verwalten von Datashare mit unterschiedlichem Status

Bei kontoübergreifenden Datashares gibt es Datashares mit unterschiedlichem Status, für die eine Aktion erforderlich ist. Ihr Datashare kann den Status „aktiv“, „Aktion erforderlich“ oder „inaktiv“ haben.

Im Folgenden sehen Sie eine Beschreibung der einzelnen Datashare-Status und der erforderlichen Aktionen:

- Wenn ein Administrator eines Produzenten-Clusters einen Datashare erstellt, lautet der Status des Datashares auf dem Produzenten-Cluster Autorisierung ausstehend. Der Administrator des

Produzenten-Clusters kann Datenkonsumenten für den Zugriff auf den Datashare berechtigen. Der Administrator des Konsumenten-Clusters muss keine Aktion ausführen.

- Wenn ein Administrator eines Produzenten-Clusters einen Datashare autorisiert, lautet der Status des Datashares auf dem Produzenten-Cluster Autorisiert. Der Administrator des Produzenten-Clusters muss keine Aktion ausführen. Wenn mindestens eine Assoziation mit einem Datenverbraucher für den Datashare vorhanden ist, ändert sich der Status des Datashares von Autorisiert zu Aktiv.

  Der Status des Datashares wird dann auf dem Konsumenten-Cluster Verfügbar (Aktion auf der Amazon-Redshift-Konsole erforderlich). Der Administrator des Konsumenten-Clusters kann den Datashare für Datenverbraucher freigeben oder den Datashare ablehnen. Der Administrator des Konsumenten-Clusters kann auch den AWS CLI -Befehl `describeDatashareforConsumer` verwenden, um den Status eines Datashares anzuzeigen. Oder der Administrator kann den CLI-Befehl `describeDatashare` verwenden und den Amazon-Ressourcenname (ARN) für den Datashare angeben, um den Status des Datashares anzuzeigen.
- Wenn der Administrator des Konsumenten-Clusters Datenverbrauchern einen Datashare zuordnet, wird der Status des Datashares auf dem Produzenten-Cluster als Aktiv angezeigt. Wenn mindestens eine Assoziation mit einem Datenverbraucher für den Datashare vorhanden ist, ändert sich der Status des Datashares von Autorisiert zu Aktiv. Für den Administrator des Produzenten-Cluster ist keine Aktion erforderlich.

  Der Status des Datashares wird im Konsumenten-Cluster als Aktiv angezeigt. Der Administrator des Konsumenten-Clusters muss keine Aktion ausführen.
- Wenn der Administrator des Konsumenten-Clusters eine Verbraucherzuordnung aus einem Datashare entfernt, wird der Status des Datashares entweder als Aktiv oder als Autorisiert angezeigt. Der Status ist Aktiv, wenn für den Datashare mit einem anderen Datenverbraucher mindestens eine Zuordnung besteht. Der Status ist Autorisiert, wenn keine Verbraucherzuordnung zum Datashare auf dem Produzenten-Cluster besteht. Der Administrator des Produzenten-Clusters muss keine Aktion ausführen.

  Der Status des Datashares wird als Aktion erforderlichauf dem Consumer-Cluster angezeigt, wenn alle Zuordnungen entfernt werden. Der Administrator des Konsumenten-Clusters kann den Datashare für Datenverbraucher neu zuordnen, wenn der Datashare Verbrauchern zur Verfügung steht.
- Wenn ein Administrator eines Konsumenten-Clusters einen Datashare ablehnt, wird der Status des Datashares auf dem Produzenten-Cluster als Aktion erforderlich und Abgelehnt angezeigt. Der

Administrator des Produzenten-Clusters kann den Datashare erneut autorisieren. Der Administrator des Konsumenten-Clusters muss keine Aktion ausführen.

- Wenn der Administrator eines Produzenten-Clusters die Autorisierung von einem Datashare entfernt, wird der Status des Datashares im Produzenten-Cluster als Action required (Aktion erforderlich) angezeigt. Der Administrator des Produzenten-Clusters kann auswählen, ob er das Datashare erneut autorisieren will, wenn erforderlich. Der Administrator des Konsumenten-Clusters muss keine Aktion ausführen.

Freigeben von Datashares

Sie benötigen Datashares nur, wenn Sie Daten zwischen verschiedenen von Amazon Redshift bereitgestellten Clustern oder Serverless-Arbeitsgruppen freigeben. Innerhalb desselben Clusters können Sie eine andere Datenbank mithilfe der einfachen dreiteiligen Notation `database.schema.table` abfragen, sofern Sie über die erforderlichen Berechtigungen für die Objekte in der anderen Datenbank verfügen.

Verwalten von Berechtigungen für Datashares in Amazon Redshift

Als Administrator eines Produzenten-Clusters behalten Sie die Kontrolle über die Datensätze, die Sie freigeben. Sie können neue Objekte hinzufügen oder sie aus dem Datashare entfernen. Sie können auch den Zugriff auf Datashares als Ganzes für die Kundencluster, AWS Konten oder Regionen gewähren oder entziehen. AWS Wenn Berechtigungen aufgehoben werden, verlieren Konsumenten-Cluster umgehend den Zugriff auf freigegebene Objekte. Sie sehen diese auch nicht mehr in der Liste EINGEHENDER Datashares in SVV_DATASHARES.

Im folgenden Beispiel wird der Datashare erstellt`salesshare`, das Schema hinzugefügt und die `public` Tabelle hinzugefügt. `public.tickit_sales_redshift` `salesshare` Außerdem werden Nutzungsberechtigungen für den angegebenen `salesshare` Cluster-Namespace gewährt.

```
CREATE DATASHARE salesshare;

ALTER DATASHARE salesshare ADD SCHEMA public;

ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;

GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Für CREATE DATASHARE können Superuser und Datenbankbesitzer Datashares erstellen. Weitere Informationen finden Sie unter DATASHARE ERSTELLEN. Bei ALTER DATASHARE kann der Besitzer des Datashares mit den erforderlichen Berechtigungen für die hinzuzufügenden oder zu entfernenden Datashare-Objekte den Datashare ändern. Weitere Informationen finden Sie unter ALTER DATASHARE.

Wenn Sie als Administrator eines Produzenten-Clusters einen Datashare entfernen, wird er nicht mehr im Konsumenten-Cluster aufgeführt. Die Datenbanken und Schemareferenzen, die auf dem Consumer-Cluster aus dem gelöschten Datashare erstellt wurden, sind weiterhin ohne Objekte vorhanden. Der Administrator des Consumer Clusters muss diese Datenbanken manuell löschen.

Auf der Konsumentenseite kann ein Administrator für Konsumenten-Cluster durch Erstellen einer Datenbank aus dem Datashare bestimmen, welche Benutzer und Rollen Zugriff auf die freigegebenen Daten erhalten sollen. Abhängig von den Optionen, die Sie bei der Erstellung der Datenbank wählen, können Sie den Zugriff auf die Datenbank wie folgt steuern. Weitere Informationen zum Erstellen einer Datenbank aus einem Datashare finden Sie unter CREATE DATABASE.

Erstellen der Datenbank ohne die WITH PERMISSIONS-Klausel

Ein Administrator kann den Zugriff auf Datenbank- oder Schemaebene steuern. Um den Zugriff auf Schemaebene zu steuern, muss der Administrator ein externes Schema aus der Amazon-Redshift-Datenbank erstellen, die aus dem Datashare erstellt wurde.

Im folgenden Beispiel werden Berechtigungen für den Zugriff auf eine freigegebene Tabelle auf Datenbank- und Schemaebene erteilt.

```
GRANT USAGE ON DATABASE sales_db TO Bob;

CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE sales_db SCHEMA 'public';

GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Um den Zugriff weiter einzuschränken, können Sie Ansichten über freigegebene Objekte erstellen, wobei nur die erforderlichen Daten verfügbar sind. Sie können diese Ansichten dann verwenden, um den Benutzern und Rollen Zugriff zu gewähren.

Sobald den Benutzern Zugriff auf die Datenbank oder das Schema gewährt wurde, haben sie Zugriff auf alle freigegebenen Objekte in der betreffenden Datenbank bzw. dem Schema.

Erstellen der Datenbank mit der WITH PERMISSIONS-Klausel

Nach der Gewährung von Nutzungsrechten für die Datenbank oder das Schema kann ein Administrator den Zugriff mit demselben Verfahren weiter kontrollieren, wie er es für eine lokale Datenbank oder ein lokales Schema tun würde. Ohne individuelle Objektberechtigungen können Benutzer auch dann nicht auf Objekte in der gemeinsam genutzten Datenbank oder dem Schema zugreifen, wenn ihnen die USAGE-Berechtigung erteilt wurde.

Im folgenden Beispiel werden Berechtigungen für den Zugriff auf eine gemeinsam genutzte Tabelle auf Datenbankebene erteilt.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
GRANT USAGE FOR SCHEMAS IN DATABASE sales_db TO Bob;
GRANT SELECT ON sales_db.public.tickit_sales_redshift TO Bob;
```

Nachdem Benutzern der Zugriff auf die Datenbank oder das Schema gewährt wurde, müssen sie weiterhin die entsprechenden Berechtigungen für alle Objekte in der Datenbank oder in dem Schema erhalten, auf die sie zugreifen sollen.

Granulares Teilen mithilfe von WITH PERMISSIONS (Vorschau)

Clustern oder Serverless-Arbeitsgruppen die Abfrage des Datashare ermöglichen

In diesem Schritt wird davon ausgegangen, dass das Datashare aus einem anderen Cluster oder Amazon-Redshift-Serverless-Namespace in Ihrem Konto oder von einem anderen Konto stammt und mit dem von Ihnen verwendeten Namespace verknüpft wurde.

1. Der Administrator der Konsumenten-Datenbank kann eine Datenbank aus dem Datashare erstellen.

   ```
   CREATE DATABASE my_ds_db [WITH PERMISSIONS] FROM DATASHARE my_datashare OF
    NAMESPACE 'abc123def';
   ```

   Wenn Sie eine Datenbank mit der Klausel WITH PERMISSIONS erstellen, können Sie verschiedenen Benutzern und Rollen detaillierte Berechtigungen für Datashare-Objekte gewähren. Andernfalls erhalten alle Benutzer und Rollen, denen die USAGE-Berechtigung für die Datashare-Datenbank gewährt wurde, alle Berechtigungen für alle Objekte in der Datashare-Datenbank.
2. Im Folgenden wird gezeigt, wie einem Redshift-Datenbankbenutzer oder -rolle Berechtigungen erteilt werden. Sie müssen mit einer lokalen Datenbank verbunden sein, um diese Anweisungen

auszuführen. Sie können diese Anweisungen nicht ausführen, wenn Sie vor der Ausführung der Grant-Anweisungen einen USE-Befehl in der Datashare-Datenbank ausführen.

```
GRANT USAGE ON DATABASE my_ds_db TO ROLE data_eng;
GRANT CREATE, USAGE ON SCHEMA my_ds_db.my_shared_schema TO ROLE data_eng;
GRANT ALL ON ALL TABLES IN SCHEMA my_ds_db.my_shared_schema TO ROLE data_eng;

GRANT USAGE ON DATABASE my_ds_db TO bi_user;
GRANT USAGE ON SCHEMA my_ds_db.my_shared_schema TO bi_user;
GRANT SELECT ON my_ds_db.my_shared_schema.table1 TO bi_user;
```

Arbeiten mit Ansichten in der Amazon-Redshift-Datenfreigabe

Ein Produzenten-Cluster kann reguläre, spätbindende und materialisierte Ansichten freigeben. Wenn Sie reguläre oder spätbindende Ansichten freigeben, müssen Sie die Basistabellen nicht freigeben. Die folgende Tabelle zeigt, wie Ansichten bei der Datenfreigabe unterstützt werden.

Name der Ansicht	Kann diese Ansicht einem Datashare hinzugefügt werden?	Kann ein Konsument diese Ansicht auf Datashare-Objekten in verschiedenen Clustern erstellen?	
Reguläre Ansicht	Ja	Nein	
Ansicht mit später Bindung	Ja	Ja	
Materialisierte Ansicht	Ja	Ja, aber nur mit einer vollständigen Aktualisierung.	

Die folgende Abfrage zeigt die Ausgabe einer regulären Ansicht, die mit der Datenfreigabe unterstützt wird. Weitere Informationen zur Definition einer regulären Ansicht finden Sie unter CREATE VIEW.

```
SELECT * FROM tickit_db.public.myevent_regular_vw
ORDER BY eventid LIMIT 5;
```

```
   eventid  |  eventname
  ----------+-------------
     3835   | LeAnn Rimes
     3967   | LeAnn Rimes
     4856   | LeAnn Rimes
     4948   | LeAnn Rimes
     5131   | LeAnn Rimes
```

Die folgende Abfrage zeigt die Ausgabe einer spätbindenden Ansicht, die mit der Datenfreigabe unterstützt wird. Weitere Informationen zur Definition spätbindender Ansichten finden Sie unter CREATE VIEW.

```
SELECT * FROM tickit_db.public.event_lbv
ORDER BY eventid LIMIT 5;

 eventid | venueid | catid | dateid |          eventname           |      starttime
 --------+---------+-------+--------+------------------------------
+---------------------
     1   |   305   |   8   |  1851  |        Gotterdammerung       | 2008-01-25
 14:30:00
     2   |   306   |   8   |  2114  |           Boris Godunov      | 2008-10-15
 20:00:00
     3   |   302   |   8   |  1935  |              Salome          | 2008-04-19
 14:30:00
     4   |   309   |   8   |  2090  |  La Cenerentola (Cinderella) | 2008-09-21
 14:30:00
     5   |   302   |   8   |  1982  |          Il Trovatore        | 2008-06-05
 19:00:00
```

Die folgende Abfrage zeigt die Ausgabe einer materialisierten Ansicht, die mit der Datenfreigabe unterstützt wird. Weitere Hinweise zur Definition materialisierter Ansichten finden Sie unter CREATE MATERIALIZED VIEW.

```
SELECT * FROM tickit_db.public.tickets_mv;

  catgroup | qtysold
 ----------+---------
  Concerts | 195444
    Shows  | 149905
```

Sie können gemeinsame Tabellen für alle Mandanten in einem Produzenten-Cluster verwalten. Sie können auch Teilmengen von Daten in Konsumenten-Clustern freigeben, die nach

Dimensionsspalten gefiltert wurden, z. B.`tenant_id` (`account_id` oder `namespace_id`). Dazu können Sie eine Ansicht in der Basistabelle mit einem Filter für diese ID-Spalten definieren, z. B. `current_aws_account = tenant_id`. Auf der Verbraucherseite sehen Sie beim Abfragen der Ansicht nur die Zeilen, die für Ihr Konto qualifiziert sind. Dazu können Sie die Amazon-Redshift-Kontextfunktionen `current_aws_account` und `current_namespace` verwenden.

Die folgende Abfrage gibt die Konto-ID zurück, in der sich der aktuelle Amazon-Redshift-Cluster befindet. Sie können diese Abfrage ausführen, wenn Sie mit Amazon Redshift verbunden sind.

```
select current_user, current_aws_account;

current_user | current_aws_account
-------------+--------------------
dwuser       |    111111111111
(1row)
```

Die folgende Abfrage gibt den Namespace zurück, in dem sich der aktuelle Amazon-Redshift-Cluster befindet. Sie können diese Abfrage ausführen, wenn Sie mit der Datenbank verbunden sind.

```
select current_user, current_namespace;

current_user | current_namespace
-------------+--------------------------------------
dwuser       | 86b5169f-01dc-4a6f-9fbb-e2e24359e9a8
(1 row)
```

Inkrementelle Aktualisierung für materialisierte Ansichten in einem Datashare

Amazon Redshift unterstützt die inkrementelle Aktualisierung für materialisierte Ansichten in einem Consumer-Datashare, wenn die Basistabellen gemeinsam genutzt werden. Inkrementelle Aktualisierung ist ein Vorgang, bei dem Amazon Redshift Änderungen in der Basistabelle oder den Basistabellen identifiziert, die nach der vorherigen Aktualisierung vorgenommen wurden, und nur die entsprechenden Datensätze in der materialisierten Ansicht aktualisiert. Weitere Informationen zu diesem Verhalten finden Sie unter CREATE MATERIALIZED VIEW.

Verwalten des Zugriffs auf Datenfreigabe-API-Vorgänge mit IAM-Richtlinien

Um den Zugriff auf die API-Operationen für die Datenfreigabe zu steuern, verwenden Sie aktionsbasierte IAM-Richtlinien. Weitere Informationen zum Verwalten von IAM-Richtlinien finden Sie unter Verwalten von IAM-Richtlinien im IAM-Benutzerhandbuch.

Informationen zu den Berechtigungen, die für die Verwendung der API-Vorgänge für die Datenfreigabe erforderlich sind, finden Sie unter Für die Verwendung der API-Vorgänge zur Datenfreigabe erforderliche Berechtigungen im Amazon-Redshift-Clusterverwaltungshandbuch.

Um die kontoübergreifende Datenfreigabe sicherer zu gestalten, können Sie einen bedingten Schlüssel `ConsumerIdentifier` für die API-Vorgänge `AuthorizeDataShare` und `DeauthorizeDataShare` verwenden. Auf diese Weise können Sie explizit steuern, AWS-Konten wer die beiden API-Operationen aufrufen kann.

Sie können die Autorisierung der Datenfreigabe oder deren Aufhebung für jeden Konsumenten verweigern, der sich nicht in Ihrem eigenen Konto befindet. Geben Sie dazu die AWS-Konto Nummer in der IAM-Richtlinie an.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "redshift:ConsumerIdentifier": "555555555555"
                }
            }
        }
    ]
}
```

In der IAM-Richtlinie können Sie einem Hersteller mit a DataShareArn **`testshare2`** die ausdrückliche Weitergabe an einen Verbraucher mit AWS-Konto der Nummer 111122223333 gestatten.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "arn:aws:redshift:us-
east-1:666666666666:datashare:af06285e-8a45-4ee9-b598-648c218c8ff1/testshare2",
            "Condition": {
                "StringEquals": {
                    "redshift:ConsumerIdentifier": "111122223333"
                }
            }
        }
    ]
}
```

Abfragen von Datashares

Zugreifen auf freigegebene Daten in Amazon Redshift

Sie können freigegebene Daten mithilfe von Standard-SQL-Schnittstellen, JDBC- oder ODBC-Treibern und der Daten-API ermitteln. Sie können auch Daten mit hoher Leistung aus vertrauten Business Intelligence (BI)- und Analyse-Tools abfragen. Sie können auf Ihrem Cluster Abfragen durchführen, indem Sie auf die Objekte aus anderen Amazon-Redshift-Datenbanken verweisen, die sowohl lokal als auch remote zur Verfügung stehen und für die Sie über Zugriffsberechtigungen verfügen.

Sie können dies einfach tun, indem Sie mit lokalen Datenbanken in Ihrem Cluster verbunden bleiben. Dann können Sie Verbraucherdatenbanken aus Datashares erstellen, um gemeinsam genutzte Daten zu nutzen.

Anschließend können Sie datenbankübergreifende Abfragen durchführen, welche die Datensätze verbinden. Sie können Objekte in Konsumentendatenbanken mit der dreiteiligen Notation (*consumer_database_name*.*schema_name*.*table_name*) abfragen. Über externe Schema-Links zu Schemas können Sie ebenfalls in der Konsumentendatenbank abfragen. Es lassen sich sowohl lokale Daten als auch Daten, die von anderen Clustern freigegeben wurden, innerhalb derselben Abfrage abrufen. Eine solche Abfrage kann auf Objekte aus der aktuell verbundenen Datenbank und aus anderen nicht verbundenen Datenbanken verweisen, einschließlich Konsumentendatenbanken, die aus Datashares erstellt wurden.

Zugreifen auf Metadaten für Datashares in Amazon Redshift

Um Cluster-Administratoren beim Auffinden von Datashares zu unterstützen, bietet Amazon Redshift eine Reihe von Metadatenansichten zur Auflistung von Datashares. In diesen Ansichten werden Datashares aufgelistet, die in Ihrem Cluster erstellt wurden, sowie diejenigen, die von anderen Clustern innerhalb desselben Kontos, von anderen Konten oder anderen AWS -Regionen empfangen wurden. Diese Ansichten zeigen die folgenden Informationen an:

- Datashares, die von den Clustern freigegeben und empfangen werden
- Inhalt von Datenbankobjekten in den Datashares, einschließlich der grundlegenden Freigabe-Metadaten,-Objekte und -Konsumenten.

Verwenden Sie SVV_DATAHARES, um eine Liste aller Datashares anzuzeigen, die in Ihrem Cluster erstellt (ausgehend) und von anderen freigegeben wurden (eingehend). Weitere Informationen finden Sie unter SVV_DATASHARES.

Verwenden Sie SVV_DATASHARE_CONSUMER, um eine Liste der Datenkonsumenten anzuzeigen. Weitere Informationen finden Sie unter SVV_DATASHARE_CONSUMERS.

Verwenden Sie SVV_DATASHARE_OBJECTS, um eine Liste der Objekte in allen Datashares anzuzeigen, die in Ihrem Cluster erstellt (ausgehend) und von anderen freigegeben wurden (eingehend). Weitere Informationen finden Sie unter SVV_DATASHARE_OBJECTS.

Integration der Amazon-Redshift-Datenfreigabe in Business-Intelligence-Tools

Um die Datenfreigabe für Business Intelligence (BI)-Tools zu integrieren, empfehlen wir Ihnen, die Amazon-Redshift-JDBC- oder ODBC-Treiber zu verwenden.

Amazon-Redshift-JDBC- und ODBC-Treiber unterstützen den `GetCatalogs`-API-Vorgang in den Treibern, der die Liste aller Datenbanken einschließlich der Datenbanken, die aus Datashares erstellt wurden, zurückgibt. Die Treiber unterstützen auch nachgelagerte Operationen, wie `GetSchemas`, `GetTables` usw., die Daten aus allen Datenbanken zurückgeben, die `GetCatalogs` ausgibt. Die Treiber bieten diese Unterstützung auch dann, wenn der Katalog nicht explizit im Aufruf angegeben ist. Weitere Informationen zu JDBC- oder ODBC-Treibern finden Sie unter Configuring connections in Amazon Redshift (Konfigurieren von Verbindungen in Amazon Redshift) im Amazon-Redshift-Verwaltungshandbuch.

Sie können keine direkte Verbindung zu Konsumenten-Datenbanken herstellen, die aus Datashares erstellt wurden. Verbindung mit lokalen Datenbanken in Ihrem Cluster. Wenn Sie in Ihrem Tool über eine Benutzeroberfläche für den Verbindungswechsel verfügen, sollte die Liste der Datenbanken nur die lokalen Clusterdatenbanken enthalten. Die Liste sollte Konsumenten-Datenbanken ausschließen, die aus Datashares erstellt wurden, um eine optimale Erfahrung zu bieten. Sie können eine Option in der Ansicht SVV_REDSHIFT_DATABASS verwenden, um Datenbanken zu filtern.

Überwachen und Prüfen bei der Datenfreigabe in Amazon Redshift

Durch Prüfung des Datashares können Produzenten die Evolution des Datashares verfolgen. Mithilfe von Auditing kann beispielsweise nachverfolgt werden, wann Datenfreigaben erstellt, Objekte hinzugefügt oder entfernt und Berechtigungen für Amazon Redshift Redshift-Cluster, AWS Konten oder Regionen erteilt oder entzogen wurden. AWS

Zusätzlich zur Prüfung zeichnen Produzenten und Verbraucher die Nutzung des Datashares in unterschiedlicher Aufschlüsselung auf, wie Konto-, Cluster- und Objektebenen. Weitere Informationen zum Nachverfolgen von Nutzungs- und Prüfungsansichten finden Sie unter SVL_DATASHARE_CHANGE_LOG und SVL_DATASHARE_USAGE_PRODUCER.

Sie können Datashares überwachen, indem Sie Systemansichten abfragen.

1. Der Administrator des Produzenten-Cluster, der Daten gemeinsam nutzen möchte, erstellt ein Amazon-Redshift-Datashare. Der Administrator des Produzenten-Clusters fügt dann die benötigten Datenbankobjekte hinzu. Das können Schemas, Tabellen und Ansichten für das Datashare sein und eine Liste der Konsumenten angeben, mit denen die Objekte geteilt werden sollen.

   Verwenden Sie die folgenden Systemansichten, um konsolidierte Ansichten anzuzeigen, mit denen Sie Änderungen an und die Nutzung von Datashares in Produzenten- und/oder Verbraucher-Clustern verfolgen können:

   - SYS_DATESHARE_CHANGE_LOG
   - SYS_DATASHARE_USAGE_CONSUMER
   - SYS_DATASHARE_USAGE_PRODUCER

   Verwenden Sie die folgenden Systemansichten, um Datashare-Objekte und Datenverbraucherinformationen für ausgehende Datashares anzuzeigen:

   - SVV_DATASHARES
   - SVV_DATASHARE_CONSUMERS

- SVV_DATASHARE_OBJECTS

2. Die Administratoren für Konsumenten-Cluster betrachten die Datashares, für die sie Nutzungsrechte erhalten, und überprüfen den Inhalt jedes Datashares, indem sie eingehende Datashares mit SVV_DATASHARES überwachen.

   Um gemeinsam genutzte Daten zu nutzen, erstellt der Administrator des Konsumenten-Clusters eine Amazon-Redshift-Datenbank aus dem Datashare. Der Administrator weist dann Benutzern und Rollen im Konsumenten-Cluster Berechtigungen zu. Benutzer und Rollen können die gemeinsam genutzten Objekte als Teil der standardmäßigen Metadatenabfragen auflisten, indem sie die folgenden Metadaten-Systemansichten anzeigen und sofort mit der Datenabfrage beginnen.

   - SVV_REDSHIFT_COLUMNS
   - SVV_REDSHIFT_DATABASES
   - SVV_REDSHIFT_FUNCTIONS
   - SVV_REDSHIFT_SCHEMAS
   - SVV_REDSHIFT_TABLES

   Um Objekte von lokalen und freigegebenen Schemas und externen Schemas von Amazon Redshift anzuzeigen, verwenden Sie die folgenden Metadaten-Systemansichten, um sie abzufragen.

   - SVV_ALL_COLUMNS
   - SVV_ALL_SCHEMAS
   - SVV_ALL_TABLES

Integration der Amazon Redshift Redshift-Datenfreigabe mit AWS CloudTrail

Die gemeinsame Nutzung von Daten ist integriert in. AWS CloudTrail CloudTrail ist ein Service, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon Redshift ausgeführt wurden. CloudTrail erfasst alle API-Aufrufe für den Datenaustausch als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der AWS CloudTrail Konsole und Codeaufrufen für die Datenaustauschvorgänge. Weitere Informationen zur Amazon Redshift Redshift-Integration mit finden Sie AWS CloudTrail unter Logging with CloudTrail.

Weitere Informationen zu finden Sie CloudTrail unter So CloudTrail funktioniert es.

Verwalten von Aufgaben bei der Datenfreigabe

Sie können mit der Datenfreigabe beginnen, indem Sie entweder die SQL-Schnittstelle oder die Amazon-Redshift-Konsole verwenden.

Themen

Verwalten der Datenfreigabe mithilfe der SQL-Schnittstelle

Sie können Daten für Lesezwecke in verschiedenen Amazon-Redshift-Clustern innerhalb oder über AWS-Konten oder AWS-Regionen hinweg freigeben.

Themen

Teilen des Lesezugriffs auf Daten innerhalb eines AWS-Konto

Sie können Daten für Lesezwecke in verschiedenen Amazon-Redshift-Clustern innerhalb eines AWS-Konto freigeben.

So geben Sie Daten für Lesezwecke als Administrator eines Produzenten-Clusters oder als Datenbankbesitzer frei:

1. Erstellen von Datashares im Cluster Weitere Informationen finden Sie unter DATASHARE ERSTELLEN.

```
CREATE DATASHARE salesshare;
```

Cluster-Superuser und Datenbankbesitzer können Datashares erstellen. Jedes Datashare wird während der Erstellung einer Datenbank zugeordnet. Nur Objekte aus dieser Datenbank können in diesem Datashare freigegeben werden. Mehrere Datashares können in derselben Datenbank mit derselben oder unterschiedlicher Granularität von Objekten erstellt werden. Es gibt keine Beschränkung für die Anzahl der Datashares, die ein Cluster erstellen kann.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu erstellen. Weitere Informationen finden Sie unter Erstellen von Datenaustauschen.

2. Delegieren Sie Berechtigungen, um mit Datashares zu arbeiten. Weitere Informationen finden Sie unter GRANT oder REVOKE.

   Im folgenden Beispiel werden `dbuser` Berechtigungen in `salesshare` gewährt.

   ```
   GRANT ALTER, SHARE ON DATASHARE salesshare TO dbuser;
   ```

   Cluster-Superuser und die Besitzer des Datashares können zusätzlichen Benutzern Änderungsberechtigungen für den Datashare erteilen oder widerrufen.

3. Hinzufügen zu oder Entfernen von Objekten in Datashares. Um Objekte zu einem Datashare hinzuzufügen, fügen Sie das Schema vor den Objekten hinzu. Wenn Sie ein Schema hinzufügen, fügt Amazon Redshift nicht alle untergeordneten Objekte hinzu. Fügen Sie diese explizit hinzu. Weitere Informationen finden Sie unter ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

   Sie können auch Ansichten zu einem Datashare hinzufügen.

   ```
   CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
    public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;
   ```

   Verwenden Sie ALTER DATASHARE, um Schemas und Tabellen, Ansichten und Funktionen in einem bestimmten Schema freizugeben. Superuser, Datashare-Besitzer oder Benutzer, die ALTER- oder ALL-Berechtigungen für den Datashare besitzen, können den Datashare ändern,

um Objekte hinzuzufügen oder daraus zu entfernen. Benutzer sollten über die Berechtigung verfügen, Objekte zum Datashare hinzuzufügen oder daraus zu entfernen. Benutzer sollten auch Eigentümer der Objekte sein oder SELECT, USAGE- oder ALL-Berechtigungen für die Objekte haben.

Sie können GRANT auch verwenden, um Objekte zum Datashare hinzuzufügen. Dieses Beispiel zeigt, wie:

```
GRANT SELECT ON TABLE public.tickit_sales_redshift TO DATASHARE salesshare;
```

Diese Syntax entspricht funktionell. `ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;`

Verwenden Sie die INCLUDENEW-Klausel, um neue Tabellen, Ansichten oder benutzerdefinierte SQL-Funktionen (UDFs) zum Datashare hinzuzufügen, die in einem angegebenen Schema erstellt wurden. Nur Superuser können diese Eigenschaft für jedes Datashare-Schema-Paar ändern.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares hinzuzufügen oder zu entfernen. Weitere Informationen finden Sie unter Hinzufügen von Datashare-Objekten zu Datashares, Entfernen von Datashare-Objekten aus Datashares und Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden.

4. Hinzufügen oder Entfernen von Verbrauchern zu Datashares. Im folgenden Beispiel wird der Konsumenten-Cluster-Namespace zu `salesshare` hinzugefügt. Der Namespace ist der global eindeutige Namespace-Bezeichner (Globally Unique Identifier, GUID) des Konsumenten-Clusters im Konto. Weitere Informationen finden Sie unter GRANT oder REVOKE.

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Sie können nur einem Datashare-Konsumenten in einer GRANT-Anweisung Berechtigungen erteilen.

Cluster-Superuser und die Eigentümer von Datashare-Objekten oder Benutzer mit SHARE-Berechtigung für das Datashare können Konsumenten zu einem Datashare hinzufügen oder daraus entfernen. Um dies zu tun, verwenden sie GRANT USAGE oder REVOKE USAGE.

Um den Namespace des aktuell angezeigten Clusters zu finden, können Sie den Befehl SELECT CURRENT_NAMESPACE verwenden. Um den Namespace eines anderen Clusters innerhalb desselben Clusters zu finden AWS-Konto, gehen Sie zur Detailseite des Amazon Redshift Redshift-Konsolen-Clusters. Suchen Sie auf dieser Seite das neu hinzugefügte Namespace-Feld.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datenkonsumenten für Datashares zu entfernen oder sie hinzuzufügen. Weitere Informationen erhalten Sie unter Hinzufügen von Datenverbrauchern zu Datashares und Entfernen von Datenkonsumenten aus Datashares.

5. (Optional) Fügen Sie Sicherheitseinschränkungen zum Datashare hinzu. Das folgende Beispiel zeigt, dass der Konsumenten-Cluster mit einem öffentlichen IP-Zugriff den Datashare lesen darf. Weitere Informationen finden Sie unter ALTER DATASHARE.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE = TRUE;
```

Nach der Erstellung des Datashares können Sie die Eigenschaften des Konsumententyps bearbeiten. Sie können beispielsweise festlegen, dass Cluster, die Daten aus einem bestimmten Datashare verwenden möchten, nicht öffentlich zugänglich sein können. Abfragen von Konsumenten-Clustern, die die im Datashare angegebenen Sicherheitseinschränkungen nicht erfüllen, werden bei der Abfragenlaufzeit abgelehnt.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu bearbeiten. Weitere Informationen finden Sie unter Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden.

6. Listen Sie im Cluster erstellte Datashares auf, und schauen Sie sich den Inhalt des Datashares an.

Im folgenden Beispiel werden die Informationen eines Datashares mit dem Namen `salesshare` angezeigt. Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

```
DESC DATASHARE salesshare;
```

```
 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+-------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

Im folgenden Beispiel werden die ausgehenden Datashares in einem Produzenten-Cluster angezeigt.

```
SHOW DATASHARES LIKE 'sales%';
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
share_name | share_owner  | source_database | consumer_database | share_type |
 createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+---------------------+----------------------+-----------+------------------
+--------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

Sie können auch SVV_DATASHARES, SVV_DATASHARE_CONSUMERS und SVV_DATASHARE_OBJECTS verwenden, um die Datashares, die Objekte im Datashare und die Datashare-Konsumenten anzuzeigen.

7. Löschen Sie Datashares. Weitere Informationen finden Sie unter DROP DATASHARE.

   Sie können die Datashare-Objekte jederzeit mit DROP DATASHARE löschen. Cluster-Superuser und Besitzer von Datashares können Datashares löschen.

   Im folgenden Beispiel wird ein Datashare namens `salesshare` entfernt.

   ```
   DROP DATASHARE salesshare;
   ```

   Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu löschen. Weitere Informationen finden Sie unter Löschen von -Datashares, die in Ihrem Konto erstellt wurden.

8. Verwenden Sie ALTER DATASHARE, um Objekte aus Datashares zu einem beliebigen Zeitpunkt aus dem Datashare zu entfernen. Verwenden Sie REVOKE USAGE ON, um Berechtigungen für den Datashare für bestimmte Konsumenten zu widerrufen. Es entzieht USAGE-Berechtigungen für Objekte innerhalb eines Datashares und stoppt sofort den Zugriff auf alle Konsumenten-Cluster. Das Auflisten von Datashares und Metadatenabfragen, wie z. B. das Auflisten von Datenbanken und Tabellen, gibt die freigegebenen Objekte nach dem Widerrufen des Zugriffs nicht zurück.

   ```
   ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
   ```

   Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu bearbeiten. Weitere Informationen finden Sie unter Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden.

9. Widerrufen Sie den Zugriff auf das Datashare aus Namespaces, wenn Sie die Daten nicht mehr mit den Konsumenten teilen möchten.

   ```
   REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
    '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
   ```

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu bearbeiten. Weitere Informationen finden Sie unter Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden.

So geben Sie Daten für Lesezwecke als Konsumenten-Cluster-Administrator frei:

1. Listen Sie die Datashares auf, die Ihnen zur Verfügung gestellt werden, und zeigen Sie deren Inhalt an. Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

   Im folgenden Beispiel werden die Informationen zu eingehenden Datashares eines angegebenen Produzenten-Namespaces angezeigt. Wenn Sie DESC DATASHARE als Administrator eines Konsumenten-Clusters ausführen, müssen Sie die Option NAMESPACE angeben, um eingehende Datashares anzuzeigen.

```
DESC DATASHARE salesshare OF NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';


 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name           |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_users_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | view        | public.sales_data_summary_view  |
```

Nur Cluster-Superuser können dies tun. Sie können SVV_DATASHARES auch verwenden, um die Datashares anzuzeigen, und SVV_DATASHARE_OBJECTS, um die Objekte innerhalb des Datashares anzuzeigen.

Im folgenden Beispiel werden die eingehenden Datashares in einem Konsumenten-Cluster angezeigt.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+-------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

2. Als Datenbank-Superuser können Sie lokale Datenbanken erstellen, die auf die Datashares verweisen. Weitere Informationen finden Sie unter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Wenn Sie eine genauere Kontrolle über den Zugriff auf die Objekte in der lokalen Datenbank wünschen, verwenden Sie beim Erstellen der Datenbank die WITH PERMISSIONS-Klausel. Auf diese Weise können Sie in Schritt 4 Berechtigungen auf Objektebene für Objekte in der Datenbank gewähren.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Sie können Datenbanken sehen, die Sie durch Abfrage der SVV_REDSHIFT_DATABASES-Ansicht aus dem Datashare erstellt haben. Sie können keine Verbindung zu diesen schreibgeschützten Datenbanken herstellen, die aus Datashares erstellt wurden. Sie können jedoch eine Verbindung zu einer lokalen Datenbank in Ihrem Konsumenten-Cluster herstellen und eine datenbankübergreifende Abfrage durchführen, um die Daten aus den

Datenbanken abzufragen, die aus Datashares erstellt wurden. Sie können keinen Datashare auf Datenbankobjekten erstellen, die aus einem vorhandenen Datashare erstellt wurden. Sie können die Daten jedoch in eine separate Tabelle im Konsumenten-Cluster kopieren, alle erforderlichen Verarbeitungsvorgänge durchführen und dann die neu erstellten Objekte freigeben.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datenbanken aus Datashares zu erstellen. Weitere Informationen finden Sie unter Erstellen von Datenbanken aus Datashares.

3. (Optional) Erstellen Sie externe Schemas, um auf bestimmte Schemas in der Konsumenten-Datenbank zu verweisen, die im Konsumenten-Cluster importiert wurden, und weisen Sie granulare Berechtigungen zu. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

4. Gewähren Sie Benutzern und Rollen im Konsumenten-Cluster nach Bedarf Berechtigungen für Datenbanken und Schemareferenzen, die aus den Datashares erstellt wurden. Weitere Informationen finden Sie unter GRANT oder REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Wenn Sie Ihre Datenbank ohne WITH PERMISSIONS erstellt haben, können Sie Ihren Benutzern und Rollen nur Berechtigungen für die gesamte aus dem Datashare erstellte Datenbank gewähren. In einigen Fällen benötigen Sie feinkörnige Steuerelemente für eine Teilmenge von Datenbankobjekten, die aus dem Datashare erstellt wurden. Wenn dies der Fall ist, können Sie eine externe Schemareferenz erstellen, die auf bestimmte Schemas im Datashare verweist (wie im vorherigen Schritt beschrieben) und detaillierte Berechtigungen auf Schemaebene bereitstellen.

Sie können auch spätbindende Ansichten über freigegebene Objekte erstellen und diese verwenden, um detaillierte Berechtigungen zuzuweisen. Sie können auch erwägen, dass Produzenten-Cluster zusätzliche Datashares mit der erforderlichen Granularität erstellen.

Wenn Sie Ihre Datenbank in Schritt 2 mit WITH PERMISSIONS erstellt haben, müssen Sie Objekten in der gemeinsam genutzten Datenbank Berechtigungen auf Objektebene zuweisen. Ein Benutzer, der nur über die USAGE-Berechtigung verfügt, kann auf Objekte in einer

Datenbank, die mit WITH PERMISSIONS erstellt wurde, erst zugreifen, wenn ihm zusätzliche Berechtigungen auf Objektebene erteilt wurden.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

5. Abfragen von Daten in den freigegebenen Objekten in Datashares.

   Benutzer und Rollen mit Berechtigungen für Konsumenten-Datenbanken und Schemata in Konsumenten-Clustern können die Metadaten beliebiger freigegebener Objekte durchsuchen und navigieren. Sie können auch lokale Objekte in einem Konsumenten-Cluster untersuchen und navigieren. Dazu verwenden sie JDBC- oder ODBC-Treiber oder SVV_ALL- und SVV_REDSHIFT-Ansichten.

   Produzenten-Cluster können viele Schemas in der Datenbank, den Tabellen und Ansichten innerhalb jedes Schemas enthalten. Die Benutzer auf der Konsumenten-Seite können nur die Teilmenge der Objekte sehen, die über den Datashare zur Verfügung gestellt werden. Diese Benutzer können nicht die gesamten Metadaten aus dem Produzenten-Cluster sehen. Dieser Ansatz ermöglicht eine detaillierte Metadaten-Sicherheitskontrolle bei der Datenfreigabe.

   Sie verbinden sich weiterhin mit lokalen Cluster-Datenbanken. Nun können Sie aber auch aus den Datenbanken und Schemas lesen, die aus dem Datashare mit der dreiteiligen database.schema.table-Notation erstellt werden. Sie können Abfragen durchführen, die sich über alle Datenbanken erstrecken, die für Sie sichtbar sind. Dies können lokale Datenbanken auf dem Cluster oder Datenbanken sein, die aus den Datashares erstellt wurden. Konsumenten-Cluster können keine direkte Verbindung zur Datenbank herstellen, die aus dem Datashare erstellt wurde.

   Sie können auf die Daten mit vollständiger Qualifikation zugreifen. Weitere Informationen finden Sie unter Beispiele für die Verwendung einer datenbankübergreifenden Abfrage.

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
   109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
    11.40 | 2008-06-06 05:00:16
```

```
   3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
52.50 | 2008-06-06 08:26:17
   4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
26.25 | 2008-06-09 08:38:52
   5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
23.10 | 2008-08-31 09:17:02
```

Sie können nur SELECT-Anweisungen für gemeinsam genutzte Objekte verwenden. Sie können jedoch Tabellen im Konsumenten-Cluster erstellen, indem Sie die Daten von den freigegebenen Objekten in einer anderen lokalen Datenbank abfragen.

Zusätzlich zu Abfragen können Konsumenten Ansichten zu freigegebenen Objekten erstellen. Nur spätbindende Ansichten oder materialisierte Ansichten werden unterstützt. Amazon Redshift unterstützt keine regelmäßigen Ansichten zu freigegebenen Daten. Ansichten, die Konsumenten erstellen, können sich über mehrere lokale Datenbanken oder Datenbanken erstrecken, die aus Datashares erstellt wurden. Weitere Informationen finden Sie unter CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Gemeinsamer Schreibzugriff auf Daten (Vorschau)

Sie können Datenbankobjekte sowohl für Lese- als auch für Schreibvorgänge in verschiedenen Amazon Redshift-Clustern oder Amazon Redshift Serverless-Arbeitsgruppen innerhalb derselben AWS-Konto, konto- und regionsübergreifend gemeinsam nutzen. Die Verfahren in diesem Thema zeigen, wie Sie die gemeinsame Nutzung von Daten einrichten, einschließlich Schreibberechtigungen. Sie können Berechtigungen wie SELECT, INSERT und UPDATE für verschiedene Tabellen und USAGE und CREATE für Schemas gewähren. Die Daten sind live und für alle Warehouses verfügbar, sobald eine Schreibtransaktion festgeschrieben wurde. Administratoren von Produzentenkonten können festlegen, ob bestimmte Namespaces oder Regionen schreibgeschützt sind oder ob sie Zugriff auf die Daten read-and-write haben.

In den folgenden Abschnitten wird gezeigt, wie die gemeinsame Nutzung von Daten konfiguriert wird. Die Prozeduren setzen voraus, dass Sie in einer Datenbank in einem bereitgestellten Cluster oder einer Amazon-Redshift-Serverless-Arbeitsgruppe arbeiten.

Datenfreigabe im schreibgeschützten Modus im Vergleich zur Freigabe von Daten für Lese- und Schreibvorgänge

Bisher waren Objekte in Datashares unter allen Umständen schreibgeschützt. Das Schreiben in ein Objekt in einem Datashare ist ein neues Feature. Objekte in Datashares sind nur schreibfähig, wenn ein Produzent dem Datashare ausdrücklich Schreibberechtigungen wie INSERT oder CREATE für Objekte gewährt. Für die kontenübergreifende gemeinsame Nutzung muss ein Hersteller außerdem die gemeinsame Nutzung von Daten für Schreibvorgänge autorisieren, und der Verbraucher muss bestimmte Cluster und Arbeitsgruppen für Schreibvorgänge zuordnen. Einzelheiten finden Sie in den nachfolgenden Abschnitten dieses Themas.

Berechtigungen, die Sie für Datenfreigaben gewähren können (Vorschau)

Verschiedene Objekttypen und verschiedene Berechtigungen, die Sie ihnen in einem Datenaustauschkontext gewähren können.

Schemata:

- USAGE
- CREATE

Tabellen:

- SELECT
- INSERT
- AKTUALISIERUNG
- DELETE
- TRUNCATE
- DROP
- REFERENCES

Funktionen:

- EXECUTE

Datenbanken:

- CREATE

Anforderungen und Einschränkungen für Datashares in Vorschau

- Verbindungen — Sie müssen direkt mit einer Datashare-Datenbank verbunden sein oder den USE-Befehl ausführen, um in Datashares zu schreiben. Wir werden jedoch bald die Möglichkeit bieten, dies mit dreiteiliger Notation zu tun.
- Verfügbarkeit — Sie müssen serverlose Arbeitsgruppen, ra3.4xl-Cluster oder ra3.16xl-Cluster verwenden, um diese Funktion nutzen zu können. Die Unterstützung für ra3.xlplus-Cluster ist geplant.
- Erkennung von Metadaten — Wenn Sie ein Verbraucher sind, der über die Redshift-JDBC-, ODBC- oder Python-Treiber direkt mit einer Datashare-Datenbank verbunden ist, können Sie Katalogdaten auf folgende Weise anzeigen:
  - SQL SHOW-Befehle.
  - Abfrage von information_schema-Tabellen und Ansichten.
  - Abfragen von SVV-Metadatenansichten.
- Daten-API — Sie können über die Daten-API keine Verbindung zu Datashare-Datenbanken herstellen. Unterstützung dafür wird in Kürze verfügbar sein.
- Sichtbarkeit der Berechtigungen — Verbraucher können nicht sehen, welche Berechtigungen den Datashares gewährt wurden. Wir werden dies bald hinzufügen.
- Verschlüsselung — Für die kontenübergreifende gemeinsame Nutzung von Daten müssen sowohl der Producer- als auch der Consumer-Cluster verschlüsselt sein.
- Isolationsstufe — Die Isolationsstufe Ihrer Datenbank muss Snapshot-Isolation sein, damit andere serverlose Arbeitsgruppen und Cluster darauf schreiben können.
- auto Operationen — Verbraucher, die in Datashare-Objekte schreiben, lösen keinen automatischen Analysevorgang aus. Aus diesem Grund muss der Produzent nach dem Einfügen von Daten in die Tabelle die Analyse manuell ausführen, um die Tabellenstatistiken zu aktualisieren. Andernfalls sind Abfragepläne möglicherweise nicht optimal.
- Abfragen und Transaktionen mit mehreren Anweisungen — Abfragen mit mehreren Anweisungen außerhalb eines Transaktionsblocks werden derzeit nicht unterstützt. Wenn Sie also einen Abfrage-

Editor wie dbeaver verwenden und mehrere Schreibabfragen haben, müssen Sie Ihre Abfragen daher in eine explizite BEGIN... END-Transaktionsanweisung einbinden.

Unterstützte SQL-Anweisungen

Diese Anweisungen werden für die öffentliche Vorschauversion der Datenfreigabe mit Schreibvorgängen unterstützt:

- BEGIN | START TRANSACTION
- END | COMMIT | ROLLBACK
- COPY ohne COMPUPDATE
- { CREATE | DROP } SCHEMA
- { CREATE | DROP | SHOW } TABLE
- CREATE TABLE table_name AS
- DELETE
- { GRANT | REVOKE } privilege_name ON OBJECT_TYPE object_name TO consumer_user
- INSERT
- SELECT
- INSERT INTO SELECT
- TRUNCATE
- UPDATE
- Spalten des Datentyps Super

Nicht unterstützte Anweisungstypen – Die folgenden Anweisungstypen werden nicht unterstützt:

- Abfragen mit mehreren Anweisungen an Konsumenten-Warehouses, wenn zu Produzenten geschrieben wird.
- Abfragen mit Parallelitätsskalierung, die von Konsumenten zu Produzenten geschrieben werden.
- Auto-Copy-Aufträge, die von Konsumenten zu Produzenten geschrieben werden.
- Streaming-Aufträge, die von Konsumenten zu Produzenten geschrieben werden.
- Konsumenten, die Null-ETL-Integrationstabellen auf Produzenten-Clustern erstellen. Weitere Informationen zu Null-ETL-Integrationen finden Sie unter Arbeiten mit Null-ETL-Integrationen.

- Schreiben in eine Tabelle mit überlappendem Sortierschlüssel.

Teilen von Daten innerhalb eines Kontos mit Schreibberechtigungen als Administrator des Produzentenkontos (Vorschau)

Bisher waren Objekte in Datashares unter allen Umständen schreibgeschützt. Das Schreiben in ein Objekt in einem Datashare ist ein neues Feature. Objekte in Datashares sind nur schreibfähig, wenn ein Produzent dem Datashare ausdrücklich Schreibberechtigungen wie INSERT oder CREATE für Objekte gewährt. Einzelheiten finden Sie in den nachfolgenden Abschnitten dieses Themas.

Wenn Sie nach der vorhandenen Dokumentation für schreibgeschützte Datashares suchen, finden Sie diese unter Cluster-übergreifende Freigabe von Daten in Amazon Redshift.

Um die gemeinsame Nutzung von Daten zu starten, erstellt der Administrator auf dem Produzenten ein Datashare und fügt diesem Objekte hinzu:

1. Der Eigentümer oder Superuser der Produzenten-Datenbank erstellt ein Datashare.
   Ein Datashare ist ein logischer Container mit Datenbankobjekten, Berechtigungen und Konsumenten. (Konsumenten sind Cluster oder Amazon-Redshift-Serverless-Namespaces in Ihrem Konto und anderen Konten.) Jedes Datashare ist mit der Datenbank verknüpft, in der es erstellt wurde, und es können nur Objekte aus dieser Datenbank hinzugefügt werden. Der folgende Befehl erstellt ein Datashare:

   ```
   CREATE DATASHARE my_datashare [PUBLICACCESSIBLE = TRUE];
   ```

   Wenn PUBLICACCESSIBLE = TRUE gesetzt wird, können Konsumenten Ihren Datashare von öffentlich zugänglichen Clustern und bereitgestellten Arbeitsgruppen aus abfragen. Lassen Sie dies weg oder setzen Sie es explizit auf „false“, wenn Sie dies nicht zulassen möchten.

   Der Eigentümer des Datashares muss die USAGE-Berechtigung für die Schemata gewähren, die dem Datashare hinzugefügt werden sollen. Der Befehl GRANT ist neu. Er wird verwendet, um verschiedene Aktionen für das Schema zu gewähren, darunter CREATE und USAGE. Die Schemata enthalten gemeinsam genutzte Objekte:

   ```
   CREATE SCHEMA myshared_schema1;
   CREATE SCHEMA myshared_schema2;

   GRANT USAGE ON SCHEMA myshared_schema1 TO DATASHARE my_datashare;
   GRANT CREATE, USAGE ON SCHEMA myshared_schema2 TO DATASHARE my_datashare;
   ```

Alternativ kann der Administrator weiterhin ALTER-Befehle ausführen, um dem Datashare ein Schema hinzuzufügen. Wenn ein Schema auf diese Weise hinzugefügt wird, werden nur USAGE-Berechtigungen gewährt.

```
ALTER DATASHARE my_datashare ADD SCHEMA myshared_schema1;
```

2. Nachdem der Administrator Schemata hinzugefügt hat, kann er Datashare-Berechtigungen für Objekte in dem Schema gewähren. Dies können Lese- und Schreibberechtigungen sein. Das GRANT ALL-Beispiel zeigt, wie alle Berechtigungen gewährt werden.

```
GRANT SELECT, INSERT ON TABLE myshared_schema1.table1, myshared_schema1.table2,
 myshared_schema2.table1
TO DATASHARE my_datashare;

GRANT ALL ON TABLE myshared_schema1.table4 TO DATASHARE my_datashare;
```

   Sie können weiterhin Befehle wie ALTER DATASHARE ausführen, um Tabellen hinzuzufügen. Wenn Sie dies tun, werden nur SELECT-Berechtigungen für die hinzugefügten Objekte gewährt.

```
ALTER DATASHARE my_datashare ADD TABLE myshared_schema1.table1,
 myshared_schema1.table2, myshared_schema2.table1;
```

3. Der Administrator gestattet einem bestimmten Namespace in dem Konto die Nutzung des Datashare. Sie finden die Namespace-ID als Teil des ARN auf der Cluster-Detailseite, auf der Detailseite des Amazon Redshift Serverless Namespace oder indem Sie den Befehl `SELECT current_namespace;` ausführen. Weitere Informationen finden Sie unter CURRENT_NAMESPACE.

```
GRANT USAGE ON DATASHARE my_datashare TO NAMESPACE '86b5169f-012a-234b-9fbb-
e2e24359e9a8';
```

Teilen von Schreibberechtigungen für Daten über mehrere Konten hinweg (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu

verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Wenn Sie auf dem Track PREVIEW_2023 noch kein Datashare erstellt haben, gehen Sie zu Teilen des Schreibzugriffs auf Daten (Vorschau), um loszulegen.

Zuordnen gemeinsam genutzter Daten als Konsumenten-Sicherheitsadministrator (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern.
Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Falls Sie im Track PREVIEW_2023 noch kein Datashare erstellt haben, gehen Sie zu Teilen des Schreibzugriffs auf Daten (Vorschau), um loszulegen.

Voraussetzungen: Die Schritte in diesem Abschnitt werden ausgeführt, nachdem der Produzenten-Administrator bestimmte Aktionen für die gemeinsam genutzten Datenbankobjekte gewährt hat. Wenn das Datashare für ein anderes Konto freigegeben ist, autorisiert der Produzenten-Sicherheitsadministrator den Zugriff.

Der Sicherheitsadministrator des Konsumenten legt Folgendes fest:

- Ob alle Namespaces in einem Konto, Namespaces in bestimmten Regionen des Kontos oder bestimmte Namespaces Zugriff auf das Datashare haben oder nicht.
- Wenn Namespaces Zugriff auf das Datashare haben, ob diese Namespaces Schreibberechtigungen haben oder nicht.

Der Sicherheitsadministrator des Konsumenten kann das Datashare über die Konsole, die CLI oder über die API zuordnen. Bei Verwendung der CLI verwendet der Administrator den folgenden Befehl:

```
associate-data-share-consumer
--data-share-arn <value>
--consumer-identifier <value>
```

```
[--allow-writes | --no-allow-writes]
```

Weitere Informationen über den Befehl finden Sie unter associate-data-share-consumer.

Der Sicherheitsadministraor des Konsumenten muss den Wert `allow-writes` explizit auf „true“ setzen, wenn er ein Datashare einem Namespace zuordnet, um die Verwendung der Befehle INSERT und UPDATE zu ermöglichen. Andernfalls können die Benutzer nur Lesevorgänge wie mit SELECT-, USAGE- oder EXECUTE-Berechtigungen ausführen.

Sie können die Zuordnung eines Namespaces für ein Datashare ändern, indem Sie `associate-data-share-consumer` erneut mit einem anderen Wert aufrufen. Die alte Assoziation wird durch die neue Assoziation überschrieben. Wenn Sie also ursprünglich `allow-writes` zuordnen und festlegen, aber `no-allow-writes` verknüpfen und angeben oder einfach keinen Wert angeben, werden dem Konsumenten die Schreibberechtigungen entzogen.

Autorisieren von Datashares für Schreibvorgänge als Sicherheitsadministrator des Produzenten (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern.
Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Falls Sie im Track PREVIEW_2023 noch kein Datashare erstellt haben, gehen Sie zu Teilen des Schreibzugriffs auf Daten (Vorschau), um loszulegen.

> **Note**
>
> Dies gilt nur, wenn das Datashare von mehreren Konten gemeinsam genutzt wird.

Der Sicherheitsadministrator des Produzenten legt Folgendes fest:

- Ob ein anderes Konto Zugriff auf das Datashare hat oder nicht.
- Ob ein Konto über Schreibberechtigungen verfügt oder nicht, wenn es Zugriff auf das Datashare hat.

Die folgenden IAM-Berechtigungen sind erforderlich, um ein Datashare zu autorisieren:

redshift: Teilen AuthorizeData

Sie können Nutzung und Schreibvorgänge entweder mit einem CLI-Aufruf oder mit der API autorisieren:

```
authorize-data-share
--data-share-arn <value>
--consumer-identifier <value>
[--allow-writes | --no-allow-writes]
```

Weitere Informationen über den Befehl finden Sie unter authorize-data-share.

Die Konsumenten-ID kann sein:

- Eine zwölfstellige AWS Konto-ID.
- Der Namespace-Bezeichner-ARN.

Beachten Sie, dass beim Autorisierungsschritt keine Schreibberechtigungen erteilt werden. Durch die Autorisierung eines Datashares für Schreibvorgänge erhält das Konto lediglich Schreibberechtigungen, die vom Datashare-Administrator erteilt wurden. Wenn ein Administrator keine Schreibvorgänge zulässt, sind die einzigen Berechtigungen, die dem jeweiligen Konsumenten zur Verfügung stehen, SELECT, USAGE und EXECUTE.

Sie können die Autorisierung eines Datashare-Konsumenten ändern, indem Sie `authorize-data-share` erneut aufrufen, jedoch mit einem anderen Wert. Die alte Autorisierung wird durch die neue Autorisierung überschrieben. Wenn Sie also ursprünglich Schreibvorgänge autorisieren und zulassen, aber `no-allow-writes` erneut autorisieren und angeben oder einfach gar keinen Wert angeben, werden dem Konsumenten die Schreibberechtigungen entzogen.

Regionen, in denen die Datenfreigabe verfügbar ist (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Wenn Sie auf dem Track PREVIEW_2023 noch kein Datashare erstellt haben, gehen Sie zu Teilen des Schreibzugriffs auf Daten (Vorschau), um loszulegen.

In den folgenden Regionen steht die Datenfreigabe zur Verfügung (in Vorschau):

- USA Ost (Nord-Virginia): (us-east-1)
- USA Ost (Ohio): (us-east-2)
- USA West (Oregon): (us-west-2)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Daten gemeinsam nutzen AWS-Konten

Sie können Daten für Lesezwecke über AWS-Konten hinweg freigeben. Die gemeinsame Nutzung von Daten AWS-Konten funktioniert ähnlich wie die gemeinsame Nutzung von Daten innerhalb eines Kontos. Der Unterschied besteht darin, dass ein bidirektionaler Handshake erforderlich ist, um Daten für mehrere AWS-Konten freizugeben. Administratoren eines Produzentenkontos können entweder Konsumentenkonten für den Zugriff auf Datashares autorisieren oder keinen Zugriff autorisieren. Um ein autorisiertes Datashare zu verwenden, kann ein Administrator eines Konsumentenkontos das Datashare zuordnen. Der Administrator kann die Datenfreigabe einem ganzen Cluster AWS-Konto oder bestimmten Clustern im Kundenkonto zuordnen oder die Datenfreigabe ablehnen. Weitere Informationen zum Freigeben von Daten in einem Konto finden Sie unter Teilen des Lesezugriffs auf Daten innerhalb eines AWS-Konto.

Ein Datashare kann Datenkonsumenten haben, die entweder Cluster-Namespaces im gleichen oder in verschiedenen AWS-Konten sind. Sie müssen keine separaten Datashares für die Freigabe innerhalb eines Kontos und der kontoübergreifenden Freigabe erstellen.

Für die kontoübergreifende Datenfreigabe müssen sowohl der Produzenten- als auch der Konsumenten-Cluster verschlüsselt sein.

Beim Teilen von Daten mit AWS-Konten Producer-Clusteradministratoren teilen sie diese als Entität mit. AWS-Konto Ein Administrator des Konsumenten-Cluster kann entscheiden, welche Cluster-Namespaces im Konsumentenkonto Zugriff auf ein Datashare erhalten.

Themen

- Aktionen des Administrators des Produzenten-Clusters

- Aktionen des Administrators des Konsumentenkontos
- Aktionen für Administratoren von Konsumenten-Clustern

Aktionen des Administrators des Produzenten-Clusters

Wenn Sie ein Administrator eines Produzentenkontos oder ein Datenbankbesitzer sind, gehen Sie wie folgt vor:

1. Erstellen Sie Datashares in Ihrem Cluster und fügen Sie Datashare-Objekte zu den Datashares hinzu. Ausführlichere Schritte zum Erstellen von Datashares und zum Hinzufügen von Datashare-Objekten zu Datashares finden Sie unter Teilen des Lesezugriffs auf Daten innerhalb eines AWS-Konto. Weitere Informationen zu CREATE DATASHARE und ALTER DATASHARE finden Sie unter DATASHARE ERSTELLEN und ALTER DATASHARE.

   Im folgenden Beispiel werden dem Datashare `salesshare` verschiedene Datashare-Objekte hinzugefügt:

   ```
   -- Add schema to datashare
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;

   -- Add table under schema to datashare
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;

   -- Add view to datashare
   ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;

   -- Add all existing tables and views under schema to datashare (does not include
    future table)
   ALTER DATASHARE salesshare ADD ALL TABLES in schema public;
   ```

   Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu erstellen oder zu bearbeiten. Weitere Informationen erhalten Sie unter Erstellen von Datenaustauschen und Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden.
2. Delegieren Sie Berechtigungen, um mit Datashares zu arbeiten. Weitere Informationen finden Sie unter GRANT oder REVOKE.

   Im folgenden Beispiel werden `dbuser` Berechtigungen in `salesshare` gewährt.

   ```
   GRANT ALTER, SHARE ON DATASHARE salesshare TO dbuser;
   ```

Cluster-Superuser und die Besitzer des Datashares können zusätzlichen Benutzern Änderungsberechtigungen für das Datashare erteilen oder widerrufen.

3. Hinzufügen oder Entfernen von Verbrauchern zu Datashares. Im folgenden Beispiel wird die AWS-Konto -ID zu `salesshare` hinzugefügt. Weitere Informationen finden Sie unter GRANT oder REVOKE.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012';
```

Sie können nur einem Daten-Konsumenten in einer GRANT-Anweisung Berechtigungen erteilen.

Cluster-Superuser und die Besitzer von Datashare-Objekten oder Benutzer mit SHARE-Berechtigungen für das Datashare können Konsumenten zu einem Datashare hinzufügen oder daraus entfernen. Um dies zu tun, verwenden sie GRANT USAGE oder REVOKE USAGE.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datenkonsumenten für Datashares zu entfernen oder sie hinzuzufügen. Weitere Informationen erhalten Sie unter Hinzufügen von Datenverbrauchern zu Datashares und Entfernen von Datenkonsumenten aus Datashares.

4. (Optional) Widerrufen Sie den Zugriff auf die Datenfreigabe von, AWS-Konten wenn Sie die Daten nicht mehr mit den Verbrauchern teilen möchten.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012';
```

Wenn Sie ein Administrator eines Produzenten-Kontos sind, gehen Sie wie folgt vor:

Nachdem Sie dem die Nutzung gewährt haben AWS-Konto, lautet der Status des Datenaustauschs. `pending_authorization` Der Administrator des Produzentenkontos sollte Datashares über die Amazon-Redshift-Konsole autorisieren und die Datenkonsumenten auswählen.

Melden Sie sich auf https://console.aws.amazon.com/redshiftv2/ an. Dann wählen Sie aus, welchen Datenkonsumenten Sie eine Autorisierung für Datashares gewähren oder entziehen möchten. Berechtigte Datenkonsumenten erhalten Benachrichtigungen, um Maßnahmen auf Datashares zu ergreifen. Wenn Sie einen Cluster-Namespace als Datenverbraucher hinzufügen, müssen Sie keine Autorisierung durchführen. Nachdem Datenkonsumenten autorisiert wurden, können sie auf Datashare-Objekte zugreifen und eine Konsumenten-Datenbank erstellen, um die Daten

abzufragen. Weitere Informationen finden Sie unter Autorisierung oder Entfernen von Autorisierung aus Datashares.

Aktionen des Administrators des Konsumentenkontos

Wenn Sie ein Administrator eines Konsumentenkontos sind, gehen Sie wie folgt vor:

Verwenden Sie die Amazon Redshift Redshift-Konsole, um eine oder mehrere Datenfreigaben, die von anderen Konten gemeinsam genutzt werden, Ihren gesamten AWS-Konto oder bestimmten Cluster-Namespaces in Ihrem Konto zuzuordnen.

Melden Sie sich auf https://console.aws.amazon.com/redshiftv2/ an. Ordnen Sie dann einen oder mehrere Datashares, die von anderen Konten gemeinsam genutzt werden, Ihren gesamten AWS-Konto oder bestimmten Cluster-Namespaces in Ihrem Konto zu. Weitere Informationen finden Sie unter Zuordnen von Datashares.

Nachdem die AWS-Konto oder bestimmte Cluster-Namespaces verknüpft wurden, stehen die Datenfreigaben zur Nutzung zur Verfügung. Sie können die Zuordnung des Datashares auch jederzeit ändern. Wenn die Zuordnung von einzelnen Cluster-Namespaces zu einem geändert wird AWS-Konto, überschreibt Amazon Redshift die Cluster-Namespaces mit den Informationen. AWS-Konto Wenn die Zuordnung von einem AWS-Konto zu bestimmten Cluster-Namespaces geändert wird, überschreibt Amazon Redshift die AWS-Konto Informationen mit den Cluster-Namespace-Informationen. Alle Cluster-Namespaces im Konto erhalten Zugriff auf die Daten.

Aktionen für Administratoren von Konsumenten-Clustern

Wenn Sie ein Administrator eines Konsumenten-Clusters sind, gehen Sie wie folgt vor:

1. Listen Sie die Datashares auf, die Ihnen zur Verfügung gestellt werden, und zeigen Sie deren Inhalt an. Der Inhalt von Datashares ist nur verfügbar, wenn der Administrator des Produzenten-Clusters die Datashares autorisiert und der Administrator des Konsumenten-Cluster die Datashares akzeptiert und zugeordnet hat. Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

   Im folgenden Beispiel werden die Informationen zu eingehenden Datashares eines angegebenen Produzenten-Namespaces angezeigt. Wenn Sie DESC DATAHSARE als Administrator eines Konsumenten-Clusters ausführen, müssen Sie NAMESPACE und Konto-ID angeben, um eingehende Datashares anzuzeigen. Geben Sie für ausgehende Datashares den Datashare-Namen an.

   ```
   SHOW DATASHARES LIKE 'sales%';
   ```

```
share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
salesshare |             |                 |                   | INBOUND    |
            |        t            |           | 123456789012    | 'dd8772e1-
d792-4fa4-996b-1870577efc0d'
```

```
DESC DATASHARE salesshare OF ACCOUNT '123456789012' NAMESPACE 'dd8772e1-
d792-4fa4-996b-1870577efc0d';


 producer_account |          producer_namespace          | share_type | share_name |
 object_type |           object_name
-----------------+--------------------------------------+------------+------------
+-------------+---------------------------------
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_users_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_venue_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_category_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_date_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_event_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_listing_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_sales_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 schema      | public
(8 rows)
```

Nur Cluster-Superuser können dies tun. Sie können SVV_DATASHARES auch verwenden, um die Datashares anzuzeigen, und SVV_DATASHARE_OBJECTS, um die Objekte innerhalb des Datashares anzuzeigen.

Im folgenden Beispiel werden die eingehenden Datashares in einem Konsumenten-Cluster angezeigt.

```
SELECT * FROM SVV_DATASHARES WHERE share_name LIKE 'sales%';

share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
salesshare |             |                 |                   | INBOUND    |
            |        t            |           | 123456789012      | 'dd8772e1-
d792-4fa4-996b-1870577efc0d'
```

```
SELECT * FROM SVV_DATASHARE_OBJECTS WHERE share_name LIKE 'sales%';
 share_type | share_name | object_type |           object_name           |
 producer_account |          producer_namespace
------------+------------+-------------+---------------------------------
+------------------+--------------------------------------
 INBOUND    | salesshare | table       | public.tickit_users_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_venue_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_category_redshift |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_date_redshift     |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_event_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_listing_redshift  |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_sales_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | schema      | public                          |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
(8 rows)
```

2. Erstellen Sie lokale Datenbanken, die auf die Datashares verweisen. Geben Sie den NAMESPACE und die Konto-ID an, wenn Sie die Datenbank aus dem Datashare erstellen. Weitere Informationen finden Sie unter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF ACCOUNT '123456789012'
 NAMESPACE 'dd8772e1-d792-4fa4-996b-1870577efc0d';
```

Wenn Sie eine genauere Kontrolle über den Zugriff auf die Objekte in der lokalen Datenbank wünschen, verwenden Sie beim Erstellen der Datenbank die WITH PERMISSIONS-Klausel. Auf diese Weise können Sie in Schritt 4 Berechtigungen auf Objektebene für Objekte in der Datenbank gewähren.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF ACCOUNT
 '123456789012' NAMESPACE 'dd8772e1-d792-4fa4-996b-1870577efc0d';
```

Sie können Datenbanken sehen, die Sie aus dem Datashare erstellt haben, indem Sie eine Abfrage in der SVV_REDSHIFT_DATABASES-Ansicht durchführen. Sie können keine Verbindung zu diesen schreibgeschützten Datenbanken herstellen, die aus Datashares erstellt wurden. Sie können jedoch eine Verbindung zu einer lokalen Datenbank in Ihrem Konsumenten-Cluster herstellen und eine datenbankübergreifende Abfrage der Daten aus den Datenbanken durchführen, die anhand von Datashares erstellt wurden. Sie können keinen Datashare auf Datenbankobjekten erstellen, die aus einem vorhandenen Datashare erstellt wurden. Sie können die Daten jedoch in eine separate Tabelle im Konsumenten-Cluster kopieren, alle erforderlichen Verarbeitungsvorgänge durchführen und dann die neu erstellten Objekte freigeben.

3. (Optional) Erstellen Sie externe Schemas, um auf bestimmte Schemas in der Konsumenten-Datenbank zu verweisen, die im Konsumenten-Cluster importiert wurden, und weisen Sie granulare Berechtigungen zu. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

   ```
   CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
    'public';
   ```

4. Gewähren Sie Benutzern oder Rollen im Konsumenten-Cluster nach Bedarf Berechtigungen für Datenbanken und Schemareferenzen, die aus den Datashares erstellt wurden. Weitere Informationen finden Sie unter GRANT oder REVOKE.

   ```
   GRANT USAGE ON DATABASE sales_db TO Bob;
   ```

   ```
   GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
   ```

Wenn Sie Ihre Datenbank ohne WITH PERMISSIONS erstellt haben, können Sie Ihren Benutzern oder Rollen nur Berechtigungen für die gesamte aus dem Datashare erstellte Datenbank gewähren. In einigen Fällen benötigen Sie feinkörnige Steuerelemente für eine Teilmenge von Datenbankobjekten, die aus dem Datashare erstellt wurden. Wenn dies der Fall ist, können Sie eine externe Schemareferenz erstellen, die auf bestimmte Schemas im Datashare verweist (wie im vorherigen Schritt beschrieben). Sie können dann auf Schemaebene detaillierte Berechtigungen bereitstellen. Sie können auch spätbindende Ansichten über freigegebene Objekte erstellen und diese verwenden, um detaillierte Berechtigungen zuzuweisen. Sie können auch erwägen, dass Produzenten-Cluster zusätzliche Datashares mit der erforderlichen Granularität erstellen. Sie können so viele Schemareferenzen aus dem Datashare für die Datenbank erstellen wie Sie möchten.

Wenn Sie Ihre Datenbank in Schritt 2 mit WITH PERMISSIONS erstellt haben, müssen Sie Objekten in der gemeinsam genutzten Datenbank Berechtigungen auf Objektebene zuweisen. Ein Benutzer, der nur über die USAGE-Berechtigung verfügt, kann auf Objekte in einer Datenbank, die mit WITH PERMISSIONS erstellt wurde, erst zugreifen, wenn ihm zusätzliche Berechtigungen auf Objektebene erteilt wurden.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

5. Abfragen von Daten in den freigegebenen Objekten in Datashares.

   Benutzer und Rollen mit Berechtigungen für Konsumenten-Datenbanken und Schemata in Konsumenten-Clustern können die Metadaten beliebiger freigegebener Objekte durchsuchen und navigieren. Sie können auch lokale Objekte in einem Konsumenten-Cluster untersuchen und navigieren. Verwenden Sie dazu JDBC- oder ODBC-Treiber oder SVV_ALL- und SVV_REDSHIFT-Ansichten.

   Produzenten-Cluster können viele Schemas in der Datenbank, den Tabellen und Ansichten innerhalb jedes Schemas enthalten. Die Benutzer auf der Konsumenten-Seite können nur die Teilmenge der Objekte sehen, die über den Datashare zur Verfügung gestellt werden. Diese Benutzer können nicht alle Metadaten aus dem Produzenten-Cluster sehen. Dieser Ansatz ermöglicht eine detaillierte Metadaten-Sicherheitskontrolle bei der Datenfreigabe.

   Sie verbinden sich weiterhin mit lokalen Cluster-Datenbanken. Nun können Sie aber auch aus den Datenbanken und Schemas lesen, die aus dem Datashare mit der dreiteiligen database.schema.table-Notation erstellt werden. Sie können Abfragen durchführen, die sich über alle Datenbanken erstrecken, die für Sie sichtbar sind. Dies können lokale Datenbanken auf dem

Cluster oder Datenbanken sein, die aus den Datashares erstellt wurden. Konsumenten-Cluster können keine direkte Verbindung zur Datenbank herstellen, die aus dem Datashare erstellt wurde.

Sie können auf die Daten mit vollständiger Qualifikation zugreifen. Weitere Informationen finden Sie unter Beispiele für die Verwendung einer datenbankübergreifenden Abfrage.

```
SELECT * FROM sales_db.public.tickit_sales_redshift;
```

Sie können nur SELECT-Anweisungen für gemeinsam genutzte Objekte verwenden. Sie können jedoch Tabellen im Konsumenten-Cluster erstellen, indem Sie die Daten von den freigegebenen Objekten in einer anderen lokalen Datenbank abfragen.

Zusätzlich zur Durchführung von Abfragen können Konsumenten Ansichten zu freigegebenen Objekten erstellen. Nur spätbindende Ansichten und materialisierte Ansichten werden unterstützt. Amazon Redshift unterstützt keine regelmäßigen Ansichten zu freigegebenen Daten. Ansichten, die Konsumenten erstellen, können sich über mehrere lokale Datenbanken oder Datenbanken erstrecken, die aus Datashares erstellt wurden. Weitere Informationen finden Sie unter CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Gemeinsame Nutzung von Daten zwischen AWS-Regionen

Sie können Daten für Lesezwecke übergreifend über Amazon-Redshift-Cluster in AWS-Regionen freigeben. Durch die regionsübergreifende gemeinsame Nutzung von Daten können Sie Daten gemeinsam nutzen, AWS-Regionen ohne Daten manuell kopieren zu müssen. Sie müssen für Ihre Daten kein Unload in Amazon S3 durchführen und die Daten nicht in einen neuen Amazon-Redshift-Cluster kopieren oder eine regionsübergreifende Snapshot-Kopie durchführen.

Mit der regionsübergreifenden Datenfreigabe können Sie Daten zwischen Clustern in derselben oder in verschiedenen Clustern gemeinsam nutzen AWS-Konto, AWS-Konten selbst wenn sich die

Cluster in verschiedenen Regionen befinden. Wenn Sie Daten mit Amazon Redshift Redshift-Clustern teilen, die sich in demselben, AWS-Konto aber unterschiedlichen befinden AWS-Regionen, folgen Sie demselben Workflow wie beim Teilen von Daten innerhalb eines AWS-Konto. Weitere Informationen finden Sie unter Teilen des Lesezugriffs auf Daten innerhalb eines AWS-Konto.

Wenn sich Cluster, die Daten gemeinsam nutzen, in unterschiedlichen AWS-Konten Bereichen befinden AWS-Regionen, können Sie denselben Arbeitsablauf wie bei der gemeinsamen Nutzung von Daten verwenden AWS-Konten und Verknüpfungen auf regionaler Ebene in den Consumer-Cluster einbeziehen. Die regionsübergreifende Datenfreigabe unterstützt die Datenfreigabezuweisung mit dem gesamten AWS-Konto, dem gesamten oder bestimmten AWS-Region Cluster-Namespaces innerhalb eines. AWS-Region Weitere Informationen zur gemeinsamen Nutzung von Daten finden Sie unter. AWS-KontenDaten gemeinsam nutzen AWS-Konten

Beim Verbrauchen von Daten aus einer anderen Region zahlt der Verbraucher die regionsübergreifende Datenübertragungsgebühr von der Produzentenregion zur Konsumentenregion.

Um das Datashare zu verwenden, kann ein Administrator des Konsumentenkontos das Datashare auf eine der folgenden drei Arten zuordnen.

- Assoziation mit einem Ganzen, das AWS-Konto all seine Elemente umfasst AWS-Regionen
- Assoziation mit einem bestimmten AWS-Region in einem AWS-Konto
- Assoziation mit bestimmten Cluster-Namespaces innerhalb eines AWS-Region

Wenn der Administrator das Ganze auswählt AWS-Konto, haben alle vorhandenen und future Cluster-Namespaces AWS-Regionen in verschiedenen Bereichen des Kontos Zugriff auf die Datashares. Ein Administrator eines Verbraucherkontos kann auch bestimmte Namespaces AWS-Regionen oder Cluster-Namespaces innerhalb einer Region auswählen, um ihm Zugriff auf die Datashares zu gewähren.

Wenn Sie ein Administrator eines Produzenten-Clusters oder Datenbankbesitzer sind, erstellen Sie ein Datashare, fügen Sie dem Datashare Datenbankobjekte und Datenkonsumenten hinzu und erteilen Sie den Datenkonsumenten Berechtigungen. Weitere Informationen finden Sie unter Aktionen des Administrators des Produzenten-Clusters.

Wenn Sie ein Producer-Kontoadministrator sind, autorisieren Sie Datashares mit der AWS Command Line Interface (AWS CLI) oder der Amazon Redshift Redshift-Konsole und wählen Sie die Datenverbraucher aus.

Wenn Sie in Administrator eines Konsumentenkontos sind, gehen Sie wie folgt vor:

Verwenden Sie die Amazon Redshift Redshift-Konsole, um eine oder mehrere Datenfreigaben, die von anderen Konten gemeinsam genutzt werden, Ihren gesamten AWS-Konto oder bestimmten Namespaces AWS-Regionen oder Cluster-Namespaces innerhalb eines AWS-Region zuzuordnen.

Mit regionsübergreifendem Datenaustausch können Sie AWS-Region mithilfe der AWS Command Line Interface (AWS CLI) oder der Amazon Redshift Redshift-Konsole Cluster in einem bestimmten Bereich hinzufügen.

Um eine oder mehrere AWS Regionen anzugeben, können Sie den `associate-data-share-consumer` CLI-Befehl mit der optionalen `consumer-region` Option verwenden.

Mit der CLI verknüpft das folgende Beispiel die Option `Salesshare` with the entire AWS-Konto mit der `associate-entire-account` Option. Sie können jeweils nur eine Region zuordnen.

```
aws redshift associate-data-share-consumer
--region {PRODUCER_REGION}
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--associate-entire-account
```

Im folgenden Beispiel wird `Salesshare` der Region USA Ost (Ohio) (`us-east-2`) zugeordnet.

```
aws redshift associate-data-share-consumer
--region {PRODUCER_REGION}
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:0123456789012:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--consumer-region 'us-east-2'
```

Im folgenden Beispiel wird der `Salesshare` einem bestimmten Consumer-Cluster-Namespace in einem anderen AWS-Konto in der Region Asien-Pazifik (Sydney) zugeordnet (`ap-southeast-2`).

```
aws redshift associate-data-share-consumer
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--consumer-arn 'arn:aws:redshift:ap-southeast-2:{CONSUMER_ACCOUNT}:namespace:
{ConsumerImmutableClusterId}'
```

Sie können die Amazon Redshift Redshift-Konsole verwenden, um Datashares mit Ihren gesamten AWS-Konto oder bestimmten Namespaces AWS-Regionen oder Cluster-Namespaces innerhalb eines zu verknüpfen. AWS-RegionMelden Sie sich dazu bei https://console.aws.amazon.com/redshiftv2/ an. Ordnen Sie dann einen oder mehrere Datashares zu, die von anderen Konten für Ihr

gesamtes AWS-Konto, die gesamte AWS-Region oder oder bestimmte Cluster-Namespaces in einer AWS-Region freigegeben werden. Weitere Informationen finden Sie unter Zuordnen von Datashares.

Nachdem die AWS-Konto oder bestimmte Cluster-Namespaces verknüpft wurden, stehen die Datashares zur Nutzung zur Verfügung. Sie können die Zuordnung des Datashares auch jederzeit ändern. Wenn die Zuordnung von einzelnen Cluster-Namespaces zu einem geändert wird AWS-Konto, überschreibt Amazon Redshift die Cluster-Namespaces mit den Informationen. AWS-Konto Wenn die Zuordnung von einem AWS-Konto zu bestimmten Cluster-Namespaces geändert wird, überschreibt Amazon Redshift die AWS-Konto Informationen mit den Cluster-Namespace-Informationen. Wenn die Zuordnung von einer ganzen AWS-Konto zu bestimmten AWS Regionen und Cluster-Namespaces geändert wird, überschreibt Amazon Redshift die AWS-Konto Informationen mit den spezifischen Regions- und Cluster-Namespace-Informationen.

Wenn Sie Administrator für einen Konsumenten-Cluster sind, können Sie lokale Datenbanken erstellen, die auf die Datashares verweisen, und Benutzern oder Rollen im Konsumenten-Cluster nach Bedarf Berechtigungen für Datenbanken gewähren, die aus den Datashares erstellt wurden. Sie können auch Ansichten für freigegebene Objekte und externe Schemas erstellen, um auf bestimmte Schemas in der Konsumenten-Datenbank zu verweisen, die im Konsumenten-Cluster importiert wurden, und granulare Berechtigungen zuweisen. Weitere Informationen finden Sie unter Aktionen für Administratoren von Konsumenten-Clustern.

Verwaltung der Kostenkontrolle für regionsübergreifende Datashares

Beim Verbrauchen von Daten aus einer anderen Region zahlt der Verbraucher die regionsübergreifende Datenübertragungsgebühr von der Produzentenregion zur Konsumentenregion. Der Preis für die Datenübertragung unterscheidet sich für verschiedene Regionen. Die Gebühr basiert auf den Datenbytes, die für jede erfolgreiche Abfrageausführung gescannt wurden. Weitere Informationen zu Amazon-Redshift-Preisen finden Sie unter Amazon-Redshift-Preise.

Ihnen wird die Anzahl der Bytes in Rechnung gestellt – auf das nächste Megabyte aufgerundet und mit einem Minimum von 10 MB pro Abfrage. Sie können Kostenkontrollen für Ihre Abfragenutzung festlegen und die Datenmenge anzeigen, die pro Abfrage in Ihrem Cluster übertragen wird.

Um Ihre Nutzung der regionsübergreifenden Datenfreigabe und die damit verbundenen Nutzungskosten zu überwachen und zu kontrollieren, können Sie tägliche, wöchentliche und monatliche Nutzungslimits erstellen. Außerdem können Sie Aktionen definieren, die Amazon Redshift automatisch ausführt, wenn diese Limits erreicht werden. So lässt sich Ihr Budget einfacher planen. Weitere Informationen zu Nutzungslimits in Amazon Redshift finden Sie unter Verwalten von Nutzungslimits in Amazon Redshift.

Abhängig von den von Ihnen festgelegten Nutzungsbeschränkungen kann Amazon Redshift darin bestehen, ein Ereignis in einer Systemtabelle zu protokollieren, einen CloudWatch Alarm zu senden und einen Administrator mit einem Amazon SNS zu benachrichtigen oder den regionsübergreifenden Datenaustausch für die weitere Verwendung zu deaktivieren. Weitere Informationen zu den Aktionen finden Sie unter Verwalten von Nutzungslimits in Amazon Redshift.

Um Nutzungslimits in der Amazon Redshift-Konsole zu erstellen, wählen Sie für Ihr Cluster Actions (Aktionen) die Option Configure usage limit (Nutzungslimit konfigurieren) aus. Mit automatisch generierten CloudWatch Metriken auf den Registerkarten Cluster-Leistung oder Überwachung können Sie Ihre Nutzungstrends überwachen und Benachrichtigungen erhalten, wenn die Nutzung Ihre definierten Grenzwerte überschreitet. Sie können Nutzungslimits auch programmgesteuert erstellen, ändern und löschen, indem Sie die AWS CLI oder Amazon Redshift-API-Operationen verwenden. Weitere Informationen finden Sie unter Verwalten von Nutzungslimits in Amazon Redshift.

Teilen lizenzierter Amazon Redshift Redshift-Daten auf AWS Data Exchange

Beim Erstellen von AWS Data Exchange Datashares und deren Hinzufügen zu einem AWS Data Exchange Produkt können Anbieter Daten in Amazon Redshift lizenzieren, sodass Verbraucher up-to-date Daten in Amazon Redshift finden, abonnieren und abfragen können, wenn sie über aktive Abonnements verfügen. AWS Data Exchange

Wenn AWS Data Exchange Datashares zu einem AWS Data Exchange Produkt hinzugefügt werden, haben Verbraucher zu Beginn ihres Abonnements automatisch Zugriff auf die Datenfreigaben eines Produkts und behalten ihren Zugriff, solange ihr Abonnement aktiv ist.

Als Produzent mit Datashares arbeiten AWS Data Exchange

Wenn Sie ein Producer-Cluster-Administrator sind, gehen Sie wie folgt vor, um AWS Data Exchange Datenfreigaben auf der Amazon Redshift Redshift-Konsole zu verwalten:

1. Erstellen Sie Datashares in Ihrem Cluster, um Daten zu teilen AWS Data Exchange und Zugriff auf die Datashares zu gewähren. AWS Data Exchange

   Cluster-Superuser und Datenbankbesitzer können Datashares erstellen. Jedes Datashare wird während der Erstellung einer Datenbank zugeordnet. Nur Objekte aus dieser Datenbank können in diesem Datashare freigegeben werden. Mehrere Datashares können in derselben Datenbank mit derselben oder unterschiedlicher Granularität von Objekten erstellt werden. Es gibt keine Beschränkung für die Anzahl der Datashares, die Sie für einen Cluster erstellen können.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu erstellen. Weitere Informationen finden Sie unter Erstellen von Datenaustauschen.

Verwenden Sie die ADX-Option MANAGEDBY, um implizit Zugriff auf die Datenfreigabe zu gewähren, wenn Sie die CREATE DATASHARE-Anweisung ausführen. AWS Data Exchange Dies bedeutet AWS Data Exchange , dass dieser Datashare verwaltet wird. Sie können die Option MANAGEDBY ADX nur verwenden, wenn Sie ein neues Datashare erstellen. Die Anweisung ALTER DATASHARE lässt sich nicht verwenden, um ein vorhandenes Datashare zu ändern und die Option MANAGEDBY ADX hinzuzufügen. Sobald ein Datashare mit der Option MANAGEDBY ADX erstellt wurde, kann nur AWS Data Exchange auf das Datashare zugreifen und es verwalten.

```
CREATE DATASHARE salesshare
[[SET] MANAGEDBY [=] {ADX} ];
```

2. Sie können Objekte zu Datashares hinzufügen. Der Producer-Administrator verwaltet weiterhin Datashare-Objekte, die in einem Datashare verfügbar sind. AWS Data Exchange

   Um Objekte zu einem Datashare hinzuzufügen, fügen Sie das Schema vor den Objekten hinzu. Wenn Sie ein Schema hinzufügen, fügt Amazon Redshift nicht alle untergeordneten Objekte hinzu. Sie müssen sie explizit hinzufügen. Weitere Informationen finden Sie unter ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

   Sie können auch Ansichten zu einem Datashare hinzufügen.

   ```
   CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
    public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;
   ```

   Verwenden Sie ALTER DATASHARE, um Schemas und Tabellen, Ansichten und Funktionen in einem bestimmten Schema freizugeben. Superuser, Datashare-Besitzer oder Benutzer mit ALTER- oder ALL-Berechtigungen für das Datashare können das Datashare ändern, um Objekte hinzuzufügen oder daraus zu entfernen. Benutzer sollten über die Berechtigung verfügen, Objekte zum Datashare hinzuzufügen oder daraus zu entfernen. Benutzer sollten auch

Eigentümer der Objekte sein oder SELECT, USAGE- oder ALL-Berechtigungen für die Objekte haben.

Verwenden Sie die INCLUDENEW-Klausel, um neue Tabellen, Ansichten oder benutzerdefinierte SQL-Funktionen (UDFs) zum Datashare hinzuzufügen, die in einem angegebenen Schema erstellt wurden. Nur Superuser können diese Eigenschaft für jedes Datashare-Schema-Paar ändern.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares hinzuzufügen oder zu entfernen. Weitere Informationen finden Sie unter Hinzufügen von Datashare-Objekten zu Datashares, Entfernen von Datashare-Objekten aus Datashares und AWS Data Exchange Datenfreigaben bearbeiten.

3. Gehen Sie wie folgt vor, um den Zugriff auf die Datashares für AWS Data Exchange zu autorisieren:

   - Autorisieren Sie explizit den Zugriff auf den Datashare für, AWS Data Exchange indem Sie das Schlüsselwort in der API verwenden. ADX `aws redshift authorize-data-share` Dies ermöglicht es AWS Data Exchange , den Datenaustausch im Dienstkonto zu erkennen und die Zuordnung von Verbrauchern zum Datashare zu verwalten.

     ```
     aws redshift authorize-data-share
     --data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
     {PRODUCER_CLUSTER_NAMESPACE}/salesshare
     --consumer-identifier ADX
     ```

     Sie können einen bedingten Schlüssel `ConsumerIdentifier` für die `AuthorizeDataShare` und `DeauthorizeDataShare` APIs verwenden, um für AWS Data Exchange explizit das Aufrufen beider APIs in der IAM-Richtlinie zuzulassen oder zu verweigern.

     ```
     {
         "Version": "2012-10-17",
         "Statement": [
             {
                 "Sid": "VisualEditor0",
                 "Effect": "Deny",
     ```

```
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "redshift:ConsumerIdentifier": "ADX"
                }
            }
        }
    ]
}
```

- Verwenden Sie die Amazon Redshift Redshift-Konsole, um die Autorisierung von AWS Data Exchange Datashares zu autorisieren oder zu entfernen. Weitere Informationen finden Sie unter Autorisierung oder Entfernen von Autorisierung aus Datashares.
- Optional können Sie den Zugriff auf den Datashare implizit autorisieren, wenn Sie den AWS Data Exchange Datashare in einen Datensatz importieren. AWS Data Exchange

Um die Autorisierung für den Zugriff auf die AWS Data Exchange Datashares zu entfernen, verwenden Sie das Schlüsselwort in der API-Operation. `ADX` `aws redshift deauthorize-data-share` Dadurch lassen Sie zu, dass AWS Data Exchange das Datashare im Dienstkonto erkennen und das Entfernen der Zuordnung vom Datashare verwalten kann.

```
aws redshift deauthorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier ADX
```

4. Listen Sie im Cluster erstellte Datashares auf, und schauen Sie sich den Inhalt des Datashares an.

   Im folgenden Beispiel werden die Informationen eines Datashares mit dem Namen Salesshare angezeigt. Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

```
DESC DATASHARE salesshare;
```

```
 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
-------------------+--------------------------------------+------------
+------------+------------+--------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

Im folgenden Beispiel werden die ausgehenden Datashares in einem Produzenten-Cluster angezeigt.

```
SHOW DATASHARES LIKE 'sales%';
```

Die Ausgabe sieht folgendermaßen oder ähnlich aus.

```
share_name | share_owner  | source_database | consumer_database | share_type |
 createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
-----------+--------------+-----------------+-------------------+------------
+---------------------+----------------------+-----------+-----------------
+--------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

Sie können auch SVV_DATASHARES, SVV_DATASHARE_CONSUMERS und SVV_DATASHARE_OBJECTS verwenden, um die Datashares, die Objekte im Datashare und die Datashare-Konsumenten anzuzeigen.

5. Löschen Sie Datashares. Wir empfehlen, Datenfreigaben, die mit anderen geteilt wurden, nicht AWS-Konten mithilfe AWS Data Exchange der DROP DATASHARE-Anweisung zu löschen. Diese Konten verlieren den Zugriff auf das Datashare. Diese Aktion ist unumkehrbar. Dies könnte gegen die Bedingungen des Datenproduktangebots in verstoßen. AWS Data Exchange Informationen zum Löschen einer AWS Data Exchange Datenfreigabe finden Sie unter. Nutzungshinweise für DROP DATASHARE

   Im folgenden Beispiel wird ein Datashare namens Salesshare entfernt.

   ```
   DROP DATASHARE salesshare;
   ERROR:  Drop of ADX-managed datashare salesshare requires session variable
    datashare_break_glass_session_var to be set to value '620c871f890c49'
   ```

   Um das Löschen einer AWS Data Exchange Datenfreigabe zu ermöglichen, legen Sie die Variable datashare_break_glass_session_var fest und führen Sie die DROP DATASHARE-Anweisung erneut aus. Wenn Sie ein Datashare löschen möchten, finden Sie Nutzungshinweise für DROP DATASHARE weitere Informationen unter. AWS Data Exchange

   Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu löschen. Weitere Informationen finden Sie unter Löschen von AWS Data Exchange Datashares, die in Ihrem Konto erstellt wurden.

6. Verwenden Sie ALTER DATASHARE, um Objekte aus Datashares zu einem beliebigen Zeitpunkt aus dem Datashare zu entfernen. Verwenden Sie REVOKE USAGE ON, um Berechtigungen für den Datashare für bestimmte Konsumenten zu widerrufen. Es entzieht USAGE-Berechtigungen für Objekte innerhalb eines Datashares und stoppt sofort den Zugriff auf alle Konsumenten-Cluster. Das Auflisten von Datashares und Metadatenabfragen, wie z. B. das Auflisten von Datenbanken und Tabellen, gibt die freigegebenen Objekte nach dem Widerrufen des Zugriffs nicht zurück.

   ```
   ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
   ```

   Sie können auch die Amazon-Redshift-Konsole verwenden, um Datashares zu bearbeiten. Weitere Informationen finden Sie unter AWS Data Exchange Datenfreigaben bearbeiten.

7. Gewähren oder widerrufen Sie GRANT USAGE für Datashares. AWS Data Exchange Sie können GRANT USAGE für AWS Data Exchange Datashare nicht gewähren oder widerrufen. Das folgende Beispiel zeigt einen Fehler, wenn die GRANT USAGE-Berechtigung einem AWS-Konto für einen Datenaustausch erteilt wird, der verwaltet. AWS Data Exchange

```
CREATE DATASHARE salesshare MANAGEDBY ADX;
```

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '012345678910';
ERROR:  Permission denied to add/remove consumer to/from datashare salesshare.
 Datashare consumers are managed by ADX.
```

   Weitere Informationen finden Sie unter GRANT oder REVOKE.

Wenn Sie ein Producer-Cluster-Administrator sind, gehen Sie wie folgt vor, um ein Datashare-Produkt auf der Konsole zu erstellen und zu veröffentlichen: AWS Data Exchange

- Wenn das AWS Data Exchange Datashare erstellt wurde, erstellt der Producer einen neuen Datensatz, importiert Assets, erstellt eine Revision und erstellt und veröffentlicht ein neues Produkt.

  Verwenden Sie die Amazon-Redshift-Konsole verwenden, um Datensätze zu erstellen. Weitere Informationen finden Sie unter Datensätze erstellen auf AWS Data Exchange.

  Weitere Informationen finden Sie unter Bereitstellung von Datenprodukten am. AWS Data Exchange

Als Verbraucher mit AWS Data Exchange Datashares arbeiten

Wenn Sie ein Verbraucher sind, gehen Sie wie folgt vor, um Datenprodukte zu finden, die AWS Data Exchange Datashares enthalten und Amazon Redshift Redshift-Daten abfragen:

1. Entdecken und abonnieren Sie auf der AWS Data Exchange Konsole Datenprodukte, die Datenfreigaben enthalten. AWS Data Exchange

   Sobald Ihr Abonnement beginnt, können Sie auf lizenzierte Amazon Redshift Redshift-Daten zugreifen, die als Assets in Datensätze importiert werden, die Datashares enthalten AWS Data Exchange .

Weitere Informationen zu den ersten Schritten mit der Verwendung von Datenprodukten, die AWS Data Exchange Datashares enthalten, finden Sie unter Abonnieren von Datenprodukten auf. AWS Data Exchange

2. Erstellen Sie auf der Amazon-Redshift-Konsole bei Bedarf einen Amazon-Redshift-Cluster.

   Weitere Informationen zum Erstellen eines Clusters finden Sie unter Erstellen eines Clusters.

3. Listen Sie die Datashares auf, die Ihnen zur Verfügung gestellt werden, und zeigen Sie deren Inhalt an. Weitere Informationen erhalten Sie unter DESC DATASHARE und SHOW DATASHARES.

   Im folgenden Beispiel werden die Informationen zu eingehenden Datashares eines angegebenen Produzenten-Namespaces angezeigt. Wenn Sie DESC DATASHARE als Administrator eines Konsumenten-Clusters ausführen, müssen Sie die Option ACCOUNT und NAMESPACE angeben, um eingehende Datashares anzuzeigen.

```
DESC DATASHARE salesshare of ACCOUNT '123456789012' NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name           |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_users_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | view        | public.sales_data_summary_view  |
```

Nur Cluster-Superuser können dies tun. Sie können SVV_DATASHARES auch verwenden, um die Datashares anzuzeigen, und SVV_DATASHARE_OBJECTS, um die Objekte innerhalb des Datashares anzuzeigen.

Im folgenden Beispiel werden die eingehenden Datashares in einem Konsumenten-Cluster angezeigt.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+-------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

4. Erstellen Sie lokale Datenbanken, die auf die Datashares verweisen. Sie müssen die Optionen ACCOUNT und NAMESPACE angeben, um lokale Datenbanken für Datashares zu erstellen. AWS Data Exchange Weitere Informationen finden Sie unter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF ACCOUNT '123456789012'
 NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Wenn Sie eine genauere Kontrolle über den Zugriff auf die Objekte in der lokalen Datenbank wünschen, verwenden Sie beim Erstellen der Datenbank die WITH PERMISSIONS-Klausel. Auf diese Weise können Sie in Schritt 6 Berechtigungen auf Objektebene für Objekte in der Datenbank gewähren.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF ACCOUNT
 '123456789012' NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Sie können Datenbanken sehen, die Sie durch Abfrage der SVV_REDSHIFT_DATABASES-Ansicht aus dem Datashare erstellt haben. Sie können keine Verbindung zu diesen schreibgeschützten Datenbanken herstellen, die aus Datashares erstellt wurden. Sie können jedoch eine Verbindung zu einer lokalen Datenbank in Ihrem Konsumenten-Cluster herstellen

und eine datenbankübergreifende Abfrage der Daten aus den Datenbanken durchführen, die anhand von Datashares erstellt wurden. Sie können keinen Datashare auf Datenbankobjekten erstellen, die aus einem vorhandenen Datashare erstellt wurden. Sie können die Daten jedoch in eine separate Tabelle im Konsumenten-Cluster kopieren, alle erforderlichen Verarbeitungsvorgänge durchführen und dann die neu erstellten Objekte freigeben.

Sie können auch die Amazon-Redshift-Konsole verwenden, um Datenbanken aus Datashares zu erstellen. Weitere Informationen finden Sie unter Erstellen von Datenbanken aus Datashares.

5. (Optional) Erstellen Sie externe Schemas, um auf bestimmte Schemas in der Konsumenten-Datenbank zu verweisen, die im Konsumenten-Cluster importiert wurden, und weisen Sie granulare Berechtigungen zu. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

6. Gewähren Sie Benutzern oder Rollen im Konsumenten-Cluster nach Bedarf Berechtigungen für Datenbanken und Schemareferenzen, die aus den Datashares erstellt wurden. Weitere Informationen finden Sie unter GRANT oder REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Wenn Sie Ihre Datenbank ohne WITH PERMISSIONS erstellt haben, können Sie Ihren Benutzern und Rollen nur Berechtigungen für die gesamte aus dem Datashare erstellte Datenbank gewähren. In einigen Fällen benötigen Sie feinkörnige Steuerelemente für eine Teilmenge von Datenbankobjekten, die aus dem Datashare erstellt wurden. Wenn dies der Fall ist, können Sie eine externe Schemareferenz erstellen, die auf bestimmte Schemas im Datashare verweist (wie im vorherigen Schritt beschrieben) und detaillierte Berechtigungen auf Schemaebene bereitstellen.

Sie können auch spätbindende Ansichten über freigegebene Objekte erstellen und diese verwenden, um detaillierte Berechtigungen zuzuweisen. Sie können auch erwägen, dass Produzenten-Cluster zusätzliche Datashares mit der erforderlichen Granularität erstellen. Sie können so viele Schemareferenzen aus dem Datashare für die Datenbank erstellen wie Sie möchten.

Wenn Sie Ihre Datenbank in Schritt 4 mit WITH PERMISSIONS erstellt haben, müssen Sie Objekten in der gemeinsam genutzten Datenbank Berechtigungen auf Objektebene zuweisen. Ein Benutzer, der nur über die USAGE-Berechtigung verfügt, kann auf Objekte in einer Datenbank, die mit WITH PERMISSIONS erstellt wurde, erst zugreifen, wenn ihm zusätzliche Berechtigungen auf Objektebene erteilt wurden.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

7. Abfragen von Daten in den freigegebenen Objekten in Datashares.

   Benutzer und Rollen mit Berechtigungen für Konsumenten-Datenbanken und Schemata in Konsumenten-Clustern können die Metadaten beliebiger freigegebener Objekte durchsuchen und navigieren. Sie können auch lokale Objekte in einem Konsumenten-Cluster untersuchen und navigieren. Dazu verwenden sie JDBC- oder ODBC-Treiber oder SVV_ALL- und SVV_REDSHIFT-Ansichten.

   Produzenten-Cluster können viele Schemas in der Datenbank, den Tabellen und Ansichten innerhalb jedes Schemas enthalten. Die Benutzer auf der Konsumenten-Seite können nur die Teilmenge der Objekte sehen, die über den Datashare zur Verfügung gestellt werden. Diese Benutzer können nicht die gesamten Metadaten aus dem Produzenten-Cluster sehen. Dieser Ansatz ermöglicht eine detaillierte Metadaten-Sicherheitskontrolle bei der Datenfreigabe.

   Sie verbinden sich weiterhin mit lokalen Cluster-Datenbanken. Nun können Sie aber auch aus den Datenbanken und Schemas lesen, die aus dem Datashare mit der dreiteiligen database.schema.table-Notation erstellt werden. Sie können Abfragen durchführen, die sich über alle Datenbanken erstrecken, die für Sie sichtbar sind. Dies können lokale Datenbanken auf dem Cluster oder Datenbanken sein, die aus den Datashares erstellt wurden. Konsumenten-Cluster können keine direkte Verbindung zur Datenbank herstellen, die aus dem Datashare erstellt wurde.

   Sie können auf die Daten mit vollständiger Qualifikation zugreifen. Weitere Informationen finden Sie unter Beispiele für die Verwendung einer datenbankübergreifenden Abfrage.

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
```

```
      1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
   109.20 | 2008-02-18 02:36:48
      2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
    11.40 | 2008-06-06 05:00:16
      3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
    52.50 | 2008-06-06 08:26:17
      4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
    26.25 | 2008-06-09 08:38:52
      5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
    23.10 | 2008-08-31 09:17:02
```

Sie können nur SELECT-Anweisungen für gemeinsam genutzte Objekte verwenden. Sie können jedoch Tabellen im Konsumenten-Cluster erstellen, indem Sie die Daten von den freigegebenen Objekten in einer anderen lokalen Datenbank abfragen.

Zusätzlich zu Abfragen können Konsumenten Ansichten zu freigegebenen Objekten erstellen. Nur spätbindende Ansichten oder materialisierte Ansichten werden unterstützt. Amazon Redshift unterstützt keine regelmäßigen Ansichten zu freigegebenen Daten. Ansichten, die Konsumenten erstellen, können sich über mehrere lokale Datenbanken oder Datenbanken erstrecken, die aus Datashares erstellt wurden. Weitere Informationen finden Sie unter CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Mit -verwalteten Datenfreigaben arbeiten AWS Lake Formation

Durch die gemeinsame Nutzung von Daten AWS Lake Formation können Sie die AWS Lake Formation Berechtigungen für Amazon Redshift Redshift-Datenfreigaben zentral definieren und den Benutzerzugriff auf Objekte innerhalb eines Datashare einschränken.

Arbeiten mit von Lake Formation verwalteten Datashares als Produzent

Wenn Sie Administrator eines Produzenten-Clusters oder einer Arbeitsgruppe sind, führen Sie diese Schritte aus, um Datashares für Lake Formation freizugeben:

1. Erstellen Sie Datashares in Ihrem Cluster und autorisieren Sie den Zugriff auf die Datashares. AWS Lake Formation

   Nur Cluster-Superuser und Datenbankbesitzer können Datashares erstellen. Jedes Datashare wird während der Erstellung einer Datenbank zugeordnet. Nur Objekte aus dieser Datenbank können in diesem Datashare freigegeben werden. Mehrere Datashares können in derselben Datenbank mit derselben oder unterschiedlicher Granularität von Objekten erstellt werden. Es gibt keine Beschränkung für die Anzahl der Datashares, die Sie für einen Cluster erstellen können.

   ```
   CREATE DATASHARE salesshare;
   ```

2. Fügen Sie Objekte zu dem Datashare hinzu. Der Administrator des Produzenten-Clusters oder der Arbeitsgruppe verwaltet weiterhin die verfügbaren Datashare-Objekte. Um Objekte zu einem Datashare hinzuzufügen, fügen Sie das Schema vor den Objekten hinzu. Wenn Sie ein Schema hinzufügen, fügt Amazon Redshift nicht alle untergeordneten Objekte hinzu. Sie müssen sie explizit hinzufügen. Weitere Informationen finden Sie unter ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

   Sie können auch Ansichten zu einem Datashare hinzufügen. Unterstützt werden Standardansichten, spätbindende Ansichten und materialisierte Ansichten.

   ```
   CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
    public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ```

   Verwenden Sie ALTER DATASHARE, um Schemas, Tabellen und Ansichten in einem bestimmten Schema freizugeben. Superuser, Datashare-Besitzer oder Benutzer mit ALTER- oder ALL-Berechtigungen für das Datashare können das Datashare ändern, um Objekte hinzuzufügen oder daraus zu entfernen. Datenbankbenutzer sollten Besitzer der Objekte sein oder SELECT, USAGE- oder ALL-Berechtigungen für die Objekte haben.

Verwenden Sie die INCLUDENEW-Klausel, um dem Datashare neue Tabellen und Ansichten hinzuzufügen, die in einem angegebenen Schema erstellt wurden. Nur Superuser können diese Eigenschaft für jedes Datashare-Schema-Paar ändern.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

3. Gewähren Sie einem Lake-Formation-Administratorkonto Zugriff auf das Datashare.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '012345678910' VIA DATA CATALOG;
```

Verwenden Sie den folgenden Befehl, um die Nutzungsberechtigung zu widerrufen.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '012345678910' VIA DATA CATALOG;
```

4. Gewähren Sie Lake Formation mithilfe der API-Operation `aws redshift authorize-data-share` Zugriff auf das Datashare. Dadurch kann Lake Formation das Datashare im Servicekonto erkennen und die Zuordnung von Konsumenten zum Datashare verwalten.

```
aws redshift authorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier {"DataCatalog/<consumer-account-id>"}
```

Verwenden Sie die API-Operation `aws redshift deauthorize-data-share`, um die Berechtigung für von Lake Formation verwaltete Datashares zu entfernen. Auf diese Weise ermöglichen Sie es, den Datashare im AWS Lake Formation Dienstkonto zu erkennen und die Autorisierung zu entfernen.

```
aws redshift deauthorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier {"DataCatalog/<consumer-account-id>"}
```

Wenn der Administrator des Produzenten-Clusters oder der Arbeitsgruppe irgendwann beschließt, dass keine Freigabe von Daten für den Konsumenten-Cluster mehr erforderlich ist, kann er das Datashare mithilfe von DROP DATASHARE löschen, die Autorisierung aufheben

oder die Berechtigungen für das Datashare widerrufen. Die zugehörigen Berechtigungen und Objekte in Lake Formation werden nicht automatisch gelöscht.

```
DROP DATASHARE salesshare;
```

Nach der Autorisierung des Lake Formation-Kontos zur Verwaltung des Datashare kann der Lake Formation-Administrator den gemeinsam genutzten Datashare ermitteln, den Dateshare einem Datenkatalog-ARN zuordnen und eine Datenbank in der Verknüpfung mit dem Datashare erstellen. AWS Glue Data Catalog Verwenden Sie den Befehl, um Datashares mithilfe von zu verknüpfen. AWS CLI associate-data-share-consumer Um einen Datenaustausch gemeinsam zu nutzen AWS-Regionen, geben Sie den `--region` Parameter im `associate-data-share-consumer` Befehl an oder verwenden Sie die AWS Konsole, um Ihre Datenverbraucher auszuwählen. Das folgende Beispiel zeigt, wie Sie eine von Lake Formation verwalteten Datashare in Regionen freigeben.

```
aws redshift associate-data-share-consumer --region <region-1>
--data-share-arn 'arn:aws:redshift:us-
east-1:12345678912:datashare:035c45ea-61ce-86f0-8b75-19ac6102c3b7/sample_share'
--consumer-arn 'arn:aws:glue:<region-1>:111912345678:catalog'
```

Der Lake-Formation-Administrator muss außerdem lokale Ressourcen erstellen, die festlegen, wie Objekte innerhalb des Datashares Objekten in Lake Formation zugeordnet werden sollen. Weitere Informationen zum Ermitteln von Datashares und zum Erstellen von lokalen Ressourcen finden Sie unter Verwaltung von Berechtigungen für Daten in einem Amazon-Redshift-Datashare.

Arbeiten mit von Lake Formation verwalteten Datashares als Konsument

Nachdem der AWS Lake Formation Administrator die Datashare-Einladung entdeckt und darin eine Datenbank erstellt hat, die AWS Glue Data Catalog mit dem Datashare verknüpft ist, kann der Consumer-Cluster- oder Arbeitsgruppenadministrator den Cluster mit dem Datashare und der Datenbank in der verknüpfen AWS Glue Data Catalog, eine lokale Datenbank für den Consumer-Cluster oder die Arbeitsgruppe erstellen und Benutzern und Rollen im Amazon Redshift Redshift-Consumer-Cluster oder der Arbeitsgruppe Zugriff gewähren, um mit der Abfrage zu beginnen. Führen Sie diese Schritte aus, um Abfrageberechtigungen einzurichten.

1. Erstellen Sie in der Amazon-Redshift-Konsole bei Bedarf einen Redshift-Cluster, der als Konsumenten-Cluster oder Arbeitsgruppe fungieren soll. Weitere Informationen zum Erstellen eines Clusters finden Sie unter Erstellen eines Clusters.
2. Um aufzulisten, auf welche Datenbanken im AWS Glue Data Catalog Consumer-Cluster oder in der Arbeitsgruppe Benutzer Zugriff haben, führen Sie den Befehl SHOW DATABASES aus.

   ```
   SHOW DATABASES FROM DATA CATALOG [ACCOUNT <account-id>,<account-id2>] [LIKE
    <expression>]
   ```

   Dadurch werden die Ressourcen aufgelistet, die im Datenkatalog verfügbar sind, z. B. der ARN der AWS Glue Datenbank, der Datenbankname und Informationen zum Datashare.
3. Erstellen Sie mit dem AWS Glue Datenbank-ARN von SHOW DATABASES eine lokale Datenbank im Consumer-Cluster oder in der Arbeitsgruppe. Weitere Informationen finden Sie unter CREATE DATABASE.

   ```
   CREATE DATABASE lf_db FROM ARN <lake-formation-database-ARN> WITH [NO] DATA CATALOG
    SCHEMA [<schema>];
   ```
4. Gewähren Sie Benutzern und Rollen im Konsumenten-Cluster oder der Arbeitsgruppe nach Bedarf Zugriffsberechtigungen für Datenbanken und Schemaverweise, die aus den Datashares erstellt wurden. Weitere Informationen finden Sie unter GRANT oder REVOKE. Beachten Sie, dass Benutzer, die mit dem Befehl CREATE USER (Benutzer erstellen) erstellt wurden, nicht auf Objekte in Datashares zugreifen können, die für Lake Formation freigegeben wurden. Nur Benutzer mit Zugriff auf Redshift und Lake Formation können auf Datashares zugreifen, die für Lake Formation freigegeben wurden.

   ```
   GRANT USAGE ON DATABASE sales_db TO IAM:Bob;
   ```

   Als Administrator eines Konsumenten-Clusters können Sie Benutzern und Gruppen nur Berechtigungen für die gesamte aus dem Datashare erstellte Datenbank zuweisen. In einigen Fällen benötigen Sie feinkörnige Steuerelemente für eine Teilmenge von Datenbankobjekten, die aus dem Datashare erstellt wurden.

   Sie können auch spätbindende Ansichten über freigegebene Objekte erstellen und diese verwenden, um detaillierte Berechtigungen zuzuweisen. Sie können auch erwägen, dass Produzenten-Cluster oder Arbeitsgruppen zusätzliche Datashares mit der erforderlichen

Granularität erstellen. Sie können so viele Schemareferenzen aus dem Datashare für die Datenbank erstellen wie Sie möchten.

5. Datenbankbenutzer können die Ansichten SVV_EXTERNAL_TABLES und SVV_EXTERNAL_COLUMNS verwenden, um alle gemeinsam genutzten Tabellen oder Spalten in der Datenbank zu finden AWS Glue

```
SELECT * from svv_external_tables WHERE redshift_database_name = 'lf_db';

SELECT * from svv_external_columns WHERE redshift_database_name = 'lf_db';
```

6. Abfragen von Daten in den freigegebenen Objekten in Datashares.

   Benutzer und Rollen mit Berechtigungen für Konsumenten-Datenbanken und Schemata in Konsumenten-Clustern oder Arbeitsgruppen können die Metadaten beliebiger freigegebener Objekte durchsuchen und navigieren. Sie können auch lokale Objekte in einem Konsumenten-Cluster oder einer Arbeitsgruppe untersuchen und darin navigieren. Hierfür können sie die JDBC- oder ODBC-Treiber oder die Ansichten SVV_ALL und SVV_EXTERNAL verwenden.

```
SELECT * FROM lf_db.schema.table;
```

   Sie können nur SELECT-Anweisungen für gemeinsam genutzte Objekte verwenden. Sie können jedoch Tabellen im Konsumenten-Cluster erstellen, indem Sie die Daten von den freigegebenen Objekten in einer anderen lokalen Datenbank abfragen.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.

CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Verwalten der Datenfreigabe mithilfe der Konsole

Verwenden Sie die Amazon-Redshift-Konsole, um Datashares zu verwalten, die in Ihrem Konto erstellt oder von anderen Konten freigegeben wurden.

Sie benötigen Berechtigungen, um Datashares zu erstellen, zu bearbeiten oder zu löschen. Weitere Informationen finden Sie unter Verwalten von Berechtigungen für Datashares in Amazon Redshift.

- Wenn Sie ein Administrator eines Produzenten-Clusters sind, können Sie Datashares erstellen, Datenkonsumenten hinzufügen, Datashare-Objekte hinzufügen, Datenbanken aus Datashares erstellen, Datashares bearbeiten oder sie auf der Registerkarte CLUSTER löschen.

  Navigieren Sie im Navigationsmenü zur Registerkarte Cluster und wählen Sie einen Cluster aus der Clusterliste aus. Führen Sie dann einen der folgenden Schritte aus:

  - Wählen Sie die Registerkarte Datashares im Abschnitt Datashares created in my namespace (Datashares, die in meinem Namespace erstellt wurden) aus. Führen Sie dann einen der folgenden Schritte aus:

    - Erstellen von Datenaustauschen

      Wenn ein Datenspeicher erstellt wird, können Sie Datashare-Objekte oder Datenkonsumenten hinzufügen. Weitere Informationen erhalten Sie unter Hinzufügen von Datashare-Objekten zu Datashares und Hinzufügen von Datenverbrauchern zu Datashares.

    - Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden
    - Löschen von -Datashares, die in Ihrem Konto erstellt wurden

  - Klicken Sie auf Datashares und wählen Sie ein Datashare aus dem Abschnitt Datashares from other clusters (Datashare von anderen Clustern) aus. Führen Sie dann einen der folgenden Schritte aus:

    - Erstellen von Datenaustauschen
    - Erstellen von Datenbanken aus Datashares

  - Klicken Sie auf Databases (Datenbanken) und wählen Sie eine Datenbank aus dem Abschnitt Databases (Datenbanken) aus. Wählen Sie dann Create datashare (Datashare erstellen) aus. Weitere Informationen finden Sie unter Erstellen von Datenbanken aus Datashares.

> **Note**
>
> Stellen Sie eine Verbindung zu einer Datenbank her, um Datenbanken und Objekte in Datenbanken oder Datashares im Cluster anzuzeigen. Weitere Informationen finden Sie unter Verbinden mit einer Datenbank.

Verbinden mit einer Datenbank

Verbinden Sie sich mit einer Datenbank, um Datenbanken und Objekte in Datenbanken in diesem Cluster oder Datashares anzuzeigen.

Die Benutzeranmeldeinformationen, die zum Herstellen einer Verbindung mit einer angegebenen Datenbank verwendet werden, müssen über die erforderlichen Berechtigungen verfügen, um alle Datashares anzuzeigen.

Führen Sie einen der folgenden Schritte aus, wenn keine lokale Verbindung besteht:

- Klicken Sie auf der Detailseite des Clusters in der Registerkarte Databases (Datenbanken) auf den Abschnitt Databases (Datenbanken) oder Datashare objects (Datashare-Objekte) und wählen Sie Connect to database (Mit Datenbank verbinden) aus, um Datenbankobjekte im Cluster anzuzeigen.
- Gehen Sie auf der Seite Cluster-Details auf der Registerkarte Datashares wie folgt vor:
  - Wählen Sie im Abschnitt Datashares from other clusters (Datashare von anderen Clustern) Connect to database (Mit Datenbank verbinden) aus, um Datashares von anderen Clustern anzuzeigen.
  - Wählen Sie im Abschnitt Datashares created in my cluster (In meinem Cluster erstellte Datashares) Connect to database (Mit Datenbank verbinden) aus, um Datashares in Ihrem Cluster anzuzeigen.
- Gehen Sie im Fenster Connect to database (Mit Datenbank verbinden) wie folgt vor:
  - Wenn Sie Create a new connection (Neue Verbindung herstellen) auswählen, wählen Sie AWS Secrets Manager aus, um ein gespeichertes Secret zur Authentifizierung des Zugriffs auf die Verbindung auszuwählen.

    Oder wählen Sie Temporary credentials (Temporäre Anmeldeinformationen), um Datenbank-Anmeldeinformationen zum Authentifizieren des Zugriffs für die Verbindung zu verwenden. Geben Sie Werte für den Datenbanknamen und den Datenbankbenutzer an.

    Wählen Sie Connect (Verbinden) aus.

- Klicken Sie auf Use a recent connection (Verwenden einer kürzlich verwendeten Verbindung), um eine Verbindung zu einer anderen Datenbank herzustellen, für die Sie über die erforderlichen Berechtigungen verfügen.

  Amazon Redshift stellt die Verbindung automatisch her.

Nachdem die Datenbankverbindung hergestellt wurde, können Sie mit dem Erstellen von Datashares, dem Abfragen von Datashares oder dem Erstellen von Datenbanken aus Datashares beginnen.

Erstellen von Datashares

Erstellen von Datenaustauschen

Als Administrator eines Produzenten-Clusters können Sie auf der Detailseite zu Clustern Datashare auf den Registerkarten Databases (Datenbanken) oder Datashares erstellen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Führen Sie auf der Seite zu den Cluster-Details die folgenden Schritte aus:
   - Wählen Sie im Abschnitt Database (Datenbank) auf der Registerkarte Databases (Datenbanken) eine Datenbank aus. Die Seite mit den Datenbankdetails wird angezeigt.

     Wählen Sie Create datashare (Datashare erstellen) aus. Sie können einen Datashare nur aus einer lokalen Datenbank erstellen. Falls Sie noch keine Verbindung zur Datenbank hergestellt haben, wird die Seite Verbindung zur Datenbank herstellen geöffnet. Befolgen Sie die Schritte in Verbinden mit einer Datenbank, um eine Verbindung zu einer Datenbank herzustellen. Wenn eine kürzlich hergestellte Verbindung besteht, wird die Seite Datashare erstellen angezeigt.
   - Verbinden Sie sich auf der Registerkarte Datashares im Abschnitt Datashares mit einer Datenbank, sofern noch keine Verbindung hergestellt wurde.

     Wählen Sie im Abschnitt Datashares created in my cluster (In meinem Cluster erstellte Datashare) Create datashare (Datashare erstellen) aus. Die Seite „Create datashare“ (Datashare erstellen) wird angezeigt.
4. Wählen Sie im Abschnitt Datashare information (Datashare-Informationen) eine der folgenden Optionen aus:

- Klicken Sie auf Datashare, um Datashares zu erstellen und Daten für Lesezwecke über verschiedene Amazon-Redshift-Cluster oder im selben AWS-Konto oder verschiedenen AWS-Konten freizugeben.
- Wählen Sie AWS Data Exchange Datashare, um Datashares zu erstellen, über die Sie Ihre Daten lizenzieren können. AWS Data Exchange

5. Geben Sie Werte für Datashare name (Datashare-Namen), Database name (Datenbanknamen) und Publicly accessible (Öffentlicher Zugriff) an.

   Wenn Sie den Datenbanknamen ändern, stellen Sie eine neue Datenbankverbindung her.
6. Wählen Sie im Abschnitt Datashare-Objekte Option Hinzufügen aus. Die Seite „Datashare hinzufügen“ wird angezeigt. Folgen Sie Hinzufügen von Datashare-Objekten zu Datashares, um Objekte zu einem Datashare hinzuzufügen.
7. Im Bereich Datenverbraucher können Sie wählen, ob Sie auf einem Redshift-Konto oder auf dem veröffentlichen möchten AWS Glue Data Catalog, wodurch der Prozess des Datenaustauschs über Lake Formation gestartet wird. Wenn Sie Ihr Datashare in Redshift-Konten veröffentlichen, geben Sie Ihre Daten für ein anderes Redshift-Konto frei, das als Konsumenten-Cluster fungiert.

   > **Note**
   >
   > Sobald das Datashare erstellt wurde, können Sie die Konfiguration nicht mehr bearbeiten, um die andere Veröffentlichungsoption zu verwenden.
8. Wählen Sie Create datashare (Datashare erstellen) aus.

Amazon Redshift erstellt das Datashare. Nachdem das Datashare erstellt wurde, können Sie Datenbanken aus dem Datashare erstellen.

Hinzufügen von Datashare-Objekten zu Datashares

Fügen Sie ein oder mehrere Objekte zum Datashare hinzu. Datashare-Objekte sind schreibgeschützt für Datenverbraucher.

Sie können ein Datashare erstellen, ohne Datashare-Objekte hinzuzufügen, und Objekte später hinzufügen.

Ein Datashare wird nur dann aktiv, wenn Sie mindestens ein Objekt zum Datashare hinzufügen.

1. Wählen Sie das Datashare, dem Sie Objekte hinzufügen möchten, aus der Datashare-Liste aus.

2. Wählen Sie Hinzufügen aus. Die Seite „Datenfreigabe-Objekte hinzufügen“ wird angezeigt.
3. Fügen Sie dem Datashare mindestens ein Schema hinzu, bevor Sie weitere Datashare-Objekte hinzufügen. Fügen Sie mehrere Schemata hinzu, indem Sie Add and repeat (Hinzufügen und Wiederholen) auswählen.
4. Sie können alle bestehenden Objekte ausgewählter Objekttypen aus dem angegebenen Schema oder bestimmte individuelle Objekte hinzufügen. Wählen Sie die Objekttypen aus, wie Tabellen und Ansichten oder benutzerdefinierte Funktionen.
5. Sie können Add and repeat (Hinzufügen und Wiederholen) auswählen, um die spezifischen Schemas und Datashare-Objekte und weitere Objekte hinzuzufügen.

Hinzufügen von Datenverbrauchern zu Datashares

Sie können einen oder mehrere Datenkonsumenten zu den Datashares hinzufügen. Datenkonsumenten können Cluster-Namespaces sein, die Amazon-Redshift-Cluster oder AWS-Konten eindeutig identifiziert haben.

Sie müssen die Freigabe von Datashares mit Clustern mit öffentlichem Zugriff explizit ein- oder ausschalten.

- Klicken Sie auf Add cluster namespaces to the datashare (Cluster-Namespaces zum Datashare hinzufügen). Namespaces sind GUIDs für Amazon-Redshift-Cluster.
- Wählen Sie AWS-Konten hinzufügen, um Konten zum Datashare hinzuzufügen. Die angegebenen Benutzer AWS-Konten müssen über Zugriffsberechtigungen für den Datashare verfügen.

Autorisierung oder Entfernen von Autorisierung aus Datashares

Wählen Sie als Administrator eines Produzenten-Clusters aus, welchen Datenkonsumenten Sie eine Autorisierung für Datashares gewähren oder entziehen möchten. Berechtigte Datenkonsumenten erhalten Benachrichtigungen, um Maßnahmen auf Datashare zu ergreifen. Wenn Sie einen Cluster-Namespace als Datenverbraucher hinzufügen, müssen Sie keine Autorisierung durchführen.

Voraussetzung: Um die Autorisierung für das Datashare zu autorisieren oder zu entfernen, muss mindestens ein Datenkonsument zum Datashare hinzugefügt werden.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datashares aus. Die Seite mit den Datashares wird angezeigt.

3. Wählen Sie In my account (In meinem Konto) aus.
4. Führen Sie im Abschnitt Datashares in my account (Datashare in meinem Konto) eine der folgenden Aktionen aus:
   - Wählen Sie einen oder mehrere Konsumenten-Cluster aus, die Sie autorisieren möchten. Die Seite „Authorize data consumers“ (Datenkonsumenten autorisieren) wird angezeigt. Klicken Sie dann auf Authorize (Autorisieren).

     Wenn Sie beim Erstellen des Datashares die Option In AWS Glue Data Catalog veröffentlichen ausgewählt haben, können Sie die Berechtigung für das Datashare nur einem Lake-Formation-Konto gewähren.

     Für AWS Data Exchange Datashare können Sie jeweils nur einen Datashare autorisieren.

     Wenn Sie eine Datenfreigabe autorisieren, teilen Sie AWS Data Exchange die Datenfreigabe mit dem AWS Data Exchange Dienst und ermöglichen so AWS Data Exchange , den Zugriff auf die Datenfreigabe in Ihrem Namen zu verwalten. AWS Data Exchange ermöglicht Verbrauchern den Zugriff, indem beim Abonnieren der Produkte Verbraucherkonten als Datenverbraucher zum AWS Data Exchange Datenaustausch hinzugefügt werden. AWS Data Exchange hat keinen Lesezugriff auf den Datashare.
   - Wählen Sie einen oder mehrere Konsumenten-Cluster aus, deren Autorisierung Sie entfernen möchten. Wählen Sie dann Remove authorization (Autorisierung entfernen) aus.

Nachdem Datenkonsumenten autorisiert wurden, können sie auf Datashare-Objekte zugreifen und eine Konsumenten-Datenbank erstellen, um die Daten abzufragen.

Nachdem die Autorisierung entfernt wurde, verlieren Datenkonsumenten sofort den Zugriff auf das Datashare.

Verwalten von Datashares von anderen Konten als ein Verbraucher

Zuordnen von Datashares

Als Consumer-Cluster-Administrator können Sie eine oder mehrere Datashares, die von anderen Konten gemeinsam genutzt werden, Ihrem gesamten AWS Konto oder bestimmten Cluster-Namespaces in Ihrem Konto zuordnen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.

2. Wählen Sie im Navigationsmenü Datashares aus. Die Seite mit den Datashares wird angezeigt.
3. Wählen Sie From other accounts (Von anderen Konten) aus.
4. Wählen Sie im Abschnitt Datashares from other accounts (Datashares von anderen Konten) den Datashare aus, den Sie verknüpfen möchten. Wählen Sie dazu Associate (Verknüpfen) aus. Wenn die Seite „Associate datashare“ (Datashare zuordnen) angezeigt wird, wählen Sie einen der folgenden Zuordnungstypen aus:
   - Wählen Sie Gesamtes AWS Konto aus, um alle vorhandenen und future Cluster-Namespaces in verschiedenen AWS Regionen in Ihrem AWS Konto mit dem Datashare zu verknüpfen. Wählen Sie dann Associate (Zuordnen) aus.

     Wenn das Datashare auf dem veröffentlicht wird AWS Glue Data Catalog, können Sie das Datashare nur dem gesamten Konto zuordnen. AWS
   - Wählen Sie Bestimmte AWS Regionen und Cluster-Namespaces aus, um dem Datashare eine oder mehrere AWS Regionen und bestimmte Cluster-Namespaces zuzuordnen.
     a. Wählen Sie Region hinzufügen, um dem Datashare bestimmte AWS Regionen und Cluster-Namespaces hinzuzufügen. Die Seite „Region hinzufügen“ wird angezeigt. AWS
     b. Wählen Sie eine AWS Region aus.
     c. Führen Sie eine der folgenden Aufgaben aus:
        - Wählen Sie Add all cluster namespaces (Alle Cluster-Namespaces hinzufügen) aus, um alle vorhandenen und zukünftigen Cluster-Namespaces in dieser Region zum Datashare hinzuzufügen.
        - Wählen Sie Add specific cluster namespace (Bestimmten Cluster-Namespace hinzufügen) aus, um einen oder mehrere bestimmte Cluster-Namespaces dem Datashare hinzuzufügen.
        - Wählen Sie einen oder mehrere Cluster-Namespaces und wählen Sie Region hinzufügen. AWS
     d. Wählen Sie Associate aus.

Wenn Sie das Datashare einem Lake-Formation-Konto zuordnen, wechseln Sie zur Lake-Formation-Konsole, um eine Datenbank zu erstellen, und definieren Sie dann die Berechtigungen über die Datenbank. Weitere Informationen finden Sie unter Einrichten von Berechtigungen für Amazon Redshift Redshift-Datenfreigaben im AWS Lake Formation Entwicklerhandbuch. Sobald Sie eine AWS Glue Datenbank oder eine Verbunddatenbank erstellt haben, können Sie den Abfrage-Editor v2 oder einen beliebigen beliebigen SQL-Client mit Ihrem Consumer-Cluster verwenden, um die Daten

abzufragen. Weitere Informationen finden Sie unter Arbeiten mit von Lake Formation verwalteten Datashares als Konsument.

Nachdem das Datashare verknüpft wurde, werden die Datashares verfügbar.

Sie können die Zuordnung des Datashares auch jederzeit ändern. Wenn die Zuordnung von bestimmten AWS Regionen und Cluster-Namespaces zum gesamten AWS Konto geändert wird, überschreibt Amazon Redshift die spezifischen Regions- und Cluster-Namespace-Informationen mit Kontoinformationen. AWS Alle AWS Regionen und Cluster-Namespaces im Konto haben dann Zugriff auf den Datashare. AWS

Wenn die Zuordnung von bestimmten Cluster-Namespaces zu allen Cluster-Namespaces in der angegebenen AWS Region geändert wird, haben dann alle Cluster-Namespaces in dieser Region Zugriff auf den Datashare.

Entfernen der Zuordnung des Datashares zu Datenkonsumenten

Als Administrator eines Konsumenten-Clusters können Sie die Zuordnung von Datashares von Datenkonsumenten entfernen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datashares aus. Die Seite mit den Datashares wird angezeigt.
3. Wählen Sie From other accounts (Von anderen Konten) aus.
4. Im Abschnitt Datashares from other accounts (Datashares von anderen Konten) wählen Sie das Datashare, um die Zuordnung zu Datenkonsumenten zu löschen.
5. Wählen Sie im Abschnitt Data consumers (Datenkonsumenten) einen oder mehrere Datenkonsumenten, um die Zuordnung davon zu entfernen. Wählen Sie dann Remove association (Zuordnung entfernen) aus.
6. Wählen Sie auf der Seite „Remove association“ (Zuordnung entfernen) Remove association (Zuordnung entfernen) aus.

Nachdem die Zuordnung entfernt wurde, verlieren Datenkonsumenten sofort den Zugriff auf den Datashare. Sie können jederzeit Änderungen an der Zuordnung der Datenkonsumenten vornehmen.

Ablehnen von Datashares

Als Administrator eines Konsumenten-Clusters können Sie alle Datashares mit dem Status verfügbar oder aktiv ablehnen. Nachdem Sie ein Datashare abgelehnt haben, können Benutzer

von Konsumenten-Clustern nicht mehr auf das Datashare zugreifen. Amazon Redshift gibt das abgelehnte Datashare nicht zurück, wenn die API-Operation `DescribeDataSharesForConsumer` aufgerufen wird. Wenn der Administrator des Produzenten-Clusters die API-Operation `DescribeDataSharesForProducer` ausführt, sieht er, dass das Datashare abgelehnt wurde. Sobald ein Datashare abgelehnt wurde, kann der Producer-Cluster-Administrator das Datashare erneut für einen Consumer-Cluster autorisieren, und der Consumer-Cluster-Administrator kann wählen, ob er sein AWS Konto dem Datashare zuordnen oder es ablehnen möchte.

Wenn Ihr AWS Konto eine Zuordnung zu einem Datashare und eine ausstehende Verknüpfung zu einem von Lake Formation verwalteten Datashare hat, wird durch das Ablehnen der von Lake Formation verwalteten Datenfreigabe auch das ursprüngliche Datashare zurückgewiesen. Um eine spezifische Zuordnung abzulehnen, kann der Administrator des Produzenten-Clusters die Autorisierung für ein spezifisches Datashare entfernen. Diese Aktion wirkt sich nicht auf andere Datashares aus.

Um eine Datenfreigabe abzulehnen, verwenden Sie die Konsole, den API-Vorgang oder in der. AWS `RejectDataShare` `reject-datashare` AWS CLI

So lehnen Sie eine Datenfreigabe mithilfe der Konsole ab: AWS

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datenfreigaben aus.
3. Wählen Sie From other accounts (Von anderen Konten) aus.
4. Wählen Sie im Abschnitt Datashares from other accounts (Datashare von anderen Konten) das Datashare aus, das Sie ablehnen möchten. Wenn die Seite Decline datashare (Datashare ablehnen) angezeigt wird, wählen Sie Decline (Ablehnen) aus.

Nachdem Sie die Datashares abgelehnt haben, können Sie die Änderung nicht rückgängig machen. Amazon Redshift entfernt die Datashares aus der Liste. Damit das Datashare wieder angezeigt wird, muss der Administrator des Produzenten-Clusters es erneut autorisieren.

Verwalten vorhandener Datashares

Anzeigen von Datashares

Anzeigen von Datashares auf den Registerkarten DATASHARES oder CLUSTER.

- Verwenden Sie die Registerkarte DATASHARES, um Datashares in Ihrem oder anderen Konten aufzulisten.
  - Wenn Sie Datashares anzeigen möchten, die in Ihrem Konto erstellt wurden, wählen Sie In my account (In meinem Konto) und anschließend das Datashare, das Sie anzeigen möchten.
  - Um Datashares anzuzeigen, die von anderen Konten freigegeben werden, wählen Sie From other accounts (Von anderen Konten) und dann das Datashare aus, den Sie anzeigen möchten.
- Nutzen Sie die Registerkarte CLUSTER, um Datashares in Ihrem Cluster oder von anderen Clustern aufzulisten.

  Verbindung zu einer Datenbank herstellen Weitere Informationen finden Sie unter Verbinden mit einer Datenbank.

  Wählen Sie anschließend ein Datashare aus dem Abschnitt Datashares from other clusters (Datashares von anderen Clustern) oder dem Abschnitt Datashares created in my cluster (In meinem Cluster erstellte Datashares) aus, um detaillierte Informationen dazu anzuzeigen.

Entfernen von Datashare-Objekten aus Datashares

Sie können eines oder mehrere Objekte mithilfe des folgenden Verfahrens aus einem Datashare entfernen.

Entfernen eines oder mehrerer Objekte aus einem Datashare

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares.
4. Wählen Sie im Abschnitt Datashares created in my account (In meinem Konto erstellte Datashares) Connect to database (Mit Datenbank verbinden). Weitere Informationen finden Sie unter Verbinden mit einer Datenbank.
5. Wählen Sie ein Datashare aus, das Sie bearbeiten möchten, und anschließend Edit (Bearbeiten). Die Detailseite zum Datashare wird angezeigt.
6. Gehen Sie wie folgt vor, um eines oder mehrere Datashare-Objekte aus dem Datashare zu entfernen.

- Wählen Sie eines oder mehr Schemas aus, um Schemas aus dem Datashare zu entfernen. Wählen Sie dann Remove (Entfernen) aus. Amazon Redshift entfernt die angegebenen Schemas und alle Objekte der angegebenen Schemas aus dem Datashare.
- Um Tabellen und Ansichten aus dem Datashare zu entfernen, wählen Sie eine oder mehrere Tabellen und Ansichten aus. Wählen Sie dann Remove (Entfernen) aus. Alternativ können Sie Remove by schema (Nach Schema entfernen) auswählen, um alle Tabellen und Ansichten in den angegebenen Schemas zu entfernen.
- Um benutzerdefinierte Funktionen aus dem Datashare zu entfernen, wählen Sie eine oder mehrere benutzerdefinierte Funktionen aus. Wählen Sie dann Remove (Entfernen) aus. Alternativ können Sie Remove by schema (Nach Schema entfernen) auswählen, um alle benutzerdefinierten Funktionen in den angegebenen Schemas zu entfernen.

Entfernen von Datenkonsumenten aus Datashares

Sie können einen oder mehrere Datenkonsumenten aus einem Datashare entfernen. Datenverbraucher können Cluster-Namespaces sein, die Amazon Redshift Redshift-Cluster oder -Konten eindeutig identifizieren. AWS

Wählen Sie einen oder mehrere Datenverbraucher entweder aus den Cluster-Namespace-IDs oder dem AWS Konto aus und wählen Sie dann Entfernen.

Amazon Redshift entfernt die angegebenen Datenkonsumenten aus dem Datashare. Sie können ab sofort nicht mehr auf das Datashare zugreifen.

Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden

Bearbeiten Sie über die Konsole Datashares, die in Ihrem Konto erstellt wurden. Stellen Sie zunächst eine Verbindung zu einer Datenbank her, um die Liste der in Ihrem Konto erstellten Datashares anzuzeigen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares.

4. Wählen Sie im Abschnitt Datashares created in my account (In meinem Konto erstellte Datashares) Connect to database (Mit Datenbank verbinden). Weitere Informationen finden Sie unter Verbinden mit einer Datenbank.
5. Wählen Sie ein Datashare aus, das Sie bearbeiten möchten, und anschließend Edit (Bearbeiten). Die Detailseite zum Datashare wird angezeigt.
6. Nehmen Sie Änderungen im Abschnitt Datashare objects (Datashare-Objekte) oder im Abschnitt Data consumers (Datenkonsumenten) vor.

> **Note**
>
> Wenn Sie sich dafür entschieden haben, Ihr Datashare auf dem zu veröffentlichen AWS Glue Data Catalog, können Sie die Konfiguration nicht bearbeiten, um das Datashare auf anderen Amazon Redshift Redshift-Konten zu veröffentlichen.

7. Wählen Sie Save Changes (Änderungen speichern).

Amazon Redshift aktualisiert Ihren Datenaustausch mit den Änderungen.

Löschen von -Datashares, die in Ihrem Konto erstellt wurden

Löschen Sie über die Konsole Datashares, die in Ihrem Konto erstellt wurden. Stellen Sie zunächst eine Verbindung zu einer Datenbank her, um die Liste der in Ihrem Konto erstellten Datashares anzuzeigen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares. Die Liste mit den Datashares wird angezeigt.
4. Wählen Sie im Abschnitt Datashares created in my account (In meinem Konto erstellte Datashares) Connect to database (Mit Datenbank verbinden). Weitere Informationen finden Sie unter Verbinden mit einer Datenbank.
5. Wählen Sie einen oder mehrere Datashares aus, die Sie löschen möchten, und wählen Sie dann Delete (Löschen) aus. Die Seite „Delete datashares“ (Datashares löschen) wird angezeigt.

Beim Löschen eines Datashares, das für Lake Formation freigegeben wurde, werden nicht automatisch die zugehörigen Berechtigungen in Lake Formation entfernt. Um sie zu entfernen, wechseln Sie zur Lake-Formation-Konsole.

6. Geben Sie Delete (Löschen) ein, um die ausgewählten Datashares zu löschen.
7. Wählen Sie Delete (Löschen).

Nachdem Datashares gelöscht wurden, verlieren Konsumenten von Datashares den Zugriff auf die Datashares.

Abfragen von Datashares

Erstellen von Datenbanken aus Datashares

Erstellen Sie eine Datenbank aus dem Datashare, um mit der Abfrage von Daten im Datashare zu beginnen. Sie können nur eine Datenbank aus einem angegebenen Datashare erstellen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares. Die Liste mit den Datashares wird angezeigt.
4. Wählen Sie im Abschnitt Datashares from other clusters (Datashares von anderen Clustern) Connect to database (Mit Datenbank verbinden) aus. Weitere Informationen finden Sie unter Verbinden mit einer Datenbank.
5. Wählen Sie ein Datashare aus, aus dem Sie Datenbanken erstellen möchten, und wählen Sie anschließend Create database from datashare (Datenbank aus Datashare erstellen) aus. Auf der Seite des Datashares wird „Create database“ (Datenbank erstellen) angezeigt.
6. Geben Sie bei Datenbankname einen Datenbanknamen an. Der Datenbankname muss aus 1–64 alphanumerischen Zeichen bestehen (nur in Kleinbuchstaben) und es darf kein reserviertes Wort sein.
7. Wählen Sie Create (Erstellen) aus.

Nachdem die Datenbank erstellt wurde, können Sie Daten aus der Datenbank abrufen.

Datashares verwalten AWS Data Exchange

Datensätze erstellen auf AWS Data Exchange

Datensätze erstellen am AWS Data Exchange.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares.
4. Wählen Sie im Bereich Datashares created in my account (In meinem Konto erstellte Datenfreigaben) einen Datashare aus. AWS Data Exchange
5. Wählen Sie Datensatz erstellen am. AWS Data Exchange Weitere Informationen finden Sie unter Veröffentlichen eines neuen Produkts.

AWS Data Exchange Datenfreigaben bearbeiten

Bearbeiten Sie AWS Data Exchange Datashares mit der Konsole. Stellen Sie zunächst eine Verbindung zu einer Datenbank her, um die Liste der in Ihrem Konto erstellten Datashares anzuzeigen.

Bei AWS Data Exchange Datenfreigaben können Sie keine Änderungen an Datenverbrauchern vornehmen.

Verwenden Sie den Abfrage-Editor v2, um die öffentlich zugängliche Einstellung für AWS Data Exchange Datashares zu bearbeiten. Amazon Redshift generiert einen zufälligen Einmalwert zur Festlegung der Sitzungsvariable, um das Deaktivieren dieser Einstellung zu erlauben. Weitere Informationen finden Sie unter Nutzungshinweise für ALTER DATASHARE.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Wählen Sie im Navigator-Menü Editor und dann Query editor v2 (Abfrage-Editor v2) aus.
4. Wenn Sie den Abfrage-Editor v2 zum ersten Mal verwenden, konfigurieren Sie Ihr AWS-Konto. Standardmäßig wird ein AWS eigener Schlüssel zum Verschlüsseln von Ressourcen verwendet.

Weitere Informationen zur Konfiguration Ihres AWS-Konto finden Sie unter Konfiguration Ihres AWS-Konto im Amazon Redshift Management Guide.

5. Um eine Verbindung zu dem Cluster herzustellen, in dem sich Ihr AWS Data Exchange Datashare befindet, wählen Sie Datenbank und den Clusternamen in der Strukturansicht aus. Geben Sie bei Aufforderung die Verbindungsparameter ein.
6. Kopieren Sie die folgende SQL-Anweisung. Im folgenden Beispiel wird die Einstellung für den öffentlichen Zugriff auf das Datashare Salesshare geändert.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

7. Um die kopierte SQL-Anweisung auszuführen, wählen Sie Queries (Abfragen) aus und fügen Sie die kopierte SQL-Anweisung in den Abfragebereich ein. Wählen Sie anschließend Run (Ausführen) aus.

   Ein Fehler tritt folgendermaßen auf:

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
ERROR:  Alter of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value 'c670ba4db22f4b'
```

   Der Wert „c670ba4db22f4b“ ist ein zufälliger Einmalwert, den Amazon Redshift generiert, wenn ein nicht empfohlener Vorgang auftritt.
8. Kopieren Sie die folgende Beispielanweisung und fügen Sie sie in den Abfragebereich ein. Führen Sie dann den Befehl aus. `SET datashare_break_glass_session_var`Mit dem Befehl wird eine Berechtigung erteilt, um einen nicht empfohlenen Vorgang für eine Datenfreigabe zuzulassen. AWS Data Exchange

```
SET datashare_break_glass_session_var to 'c670ba4db22f4b';
```

9. Führen Sie die Anweisung ALTER DATASHARE erneut aus.

```
ALTER DATASHARE salesshare;
```

Amazon Redshift aktualisiert Ihren Datenaustausch mit den Änderungen.

Löschen von AWS Data Exchange Datashares, die in Ihrem Konto erstellt wurden

Löschen Sie die in Ihrem Konto AWS Data Exchange erstellten Datenfreigaben mithilfe der Konsole. Stellen Sie zunächst eine Verbindung zu einer Datenbank her, um die Liste der in Ihrem Konto erstellten Datashares anzuzeigen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Wählen Sie im Navigator-Menü Editor und dann Query editor v2 (Abfrage-Editor v2) aus.
4. Wenn Sie den Abfrage-Editor v2 zum ersten Mal verwenden, konfigurieren Sie Ihr AWS-Konto. Standardmäßig wird ein AWS eigener Schlüssel zum Verschlüsseln von Ressourcen verwendet. Weitere Informationen zur Konfiguration Ihres AWS-Konto finden Sie unter Konfiguration Ihres AWS-Konto im Amazon Redshift Management Guide.
5. Um eine Verbindung zu dem Cluster herzustellen, in dem sich Ihr AWS Data Exchange Datashare befindet, wählen Sie Datenbank und den Clusternamen in der Strukturansicht aus. Geben Sie bei Aufforderung die Verbindungsparameter ein.
6. Kopieren Sie die folgende SQL-Anweisung. Im folgenden Beispiel wird ein Datashare namens Salesshare entfernt.

   ```
   DROP DATASHARE salesshare
   ```

7. Um die kopierte SQL-Anweisung auszuführen, wählen Sie Queries (Abfragen) aus und fügen Sie die kopierte SQL-Anweisung in den Abfragebereich ein. Wählen Sie anschließend Run (Ausführen) aus.

   Ein Fehler tritt folgendermaßen auf:

   ```
   ERROR:  Drop of ADX-managed datashare salesshare requires session variable
    datashare_break_glass_session_var to be set to value '620c871f890c49'
   ```

   Der Wert „620c871f890c49“ ist ein zufälliger Einmalwert, den Amazon Redshift generiert, wenn ein nicht empfohlener Vorgang auftritt.
8. Kopieren Sie die folgende Beispielanweisung und fügen Sie sie in den Abfragebereich ein. Führen Sie dann den Befehl aus. `SET datashare_break_glass_session_var`Mit dem Befehl wird

eine Berechtigung erteilt, um einen nicht empfohlenen Vorgang für eine Datenfreigabe zuzulassen. AWS Data Exchange

```
SET datashare_break_glass_session_var to '620c871f890c49';
```

9. Führen Sie die Anweisung DROP DATASHARE erneut aus.

```
DROP DATASHARE salesshare;
```

Nachdem das Datashare gelöscht wurden, verlieren Konsumenten von Datashares den Zugriff auf das Datashare.

Das Löschen eines gemeinsam genutzten AWS Data Exchange Datenaustauschs kann gegen die Nutzungsbedingungen von Datenprodukt verstoßen. AWS Data Exchange

Verwalten der Datenfreigabe mit AWS CloudFormation

Sie können die Einrichtung der gemeinsamen Nutzung von Daten automatisieren, indem Sie einen AWS CloudFormation Stack verwenden, der AWS Ressourcen bereitstellt. Der CloudFormation Stack richtet den Datenaustausch zwischen zwei Amazon Redshift Redshift-Clustern im selben AWS Konto ein. Sie können mit der Datenfreigabe beginnen, ohne dass Sie SQL-Anweisungen zur Bereitstellung Ihrer Ressourcen ausführen.

Der Stack erstellt ein Datashare auf dem von Ihnen angegebenen Cluster. Das Datashare enthält eine Tabelle und schreibgeschützte Beispieldaten. Diese Daten können von Ihrem anderen Amazon-Redshift-Cluster gelesen werden.

Wenn Sie mit der gemeinsamen Nutzung von Daten in einem AWS Konto beginnen möchten, indem Sie SQL-Anweisungen ausführen, um eine Datenfreigabe einzurichten und Berechtigungen zu gewähren, ohne sie zu verwenden CloudFormation, finden Sie unter. Teilen des Lesezugriffs auf Daten innerhalb eines AWS-Konto

Bevor Sie den Data CloudFormation Sharing-Stack ausführen können, müssen Sie mit einem Benutzer angemeldet sein, der berechtigt ist, eine IAM-Rolle und eine Lambda-Funktion zu erstellen. Sie benötigen auch zwei Amazon-Redshift-Cluster im selben Konto. Der Produzent dient zur Freigabe der Beispieldaten und der Konsument liest die Daten. Die Hauptvoraussetzung für diese Cluster ist, dass für jeden RA3-Knoten verwendet werden. Zusätzliche Anforderungen finden Sie unter Überlegungen zur Verwendung der Datenfreigabe in Amazon Redshift.

Weitere Informationen zu den ersten Schritten beim Einrichten eines Amazon Redshift Redshift-Clusters finden Sie unter Bereitgestellte Amazon Redshift Redshift-Cluster. Weitere Informationen zur Automatisierung der Einrichtung mit finden Sie unter Was ist CloudFormation? AWS CloudFormation

> **Important**
>
> Bevor Sie Ihren CloudFormation Stack starten, stellen Sie sicher, dass Sie zwei Amazon Redshift Redshift-Cluster in demselben Konto haben und dass die Cluster RA3-Knoten verwenden. Stellen Sie sicher, dass jeder Cluster über eine Datenbank und einen Superuser verfügt. Weitere Informationen finden Sie unter CREATE DATABASE und superuser.

So starten Sie Ihren CloudFormation Stack für die gemeinsame Nutzung von Amazon Redshift Redshift-Daten:

1. Klicken Sie auf CFN-Stack starten, wodurch Sie zu dem CloudFormation Service im gelangen. AWS Management Console

   Melden Sie sich an, wenn Sie dazu aufgefordert werden.

   Der Stack-Erstellungsprozess beginnt und verweist auf eine CloudFormation Vorlagendatei, die in Amazon S3 gespeichert ist. Eine CloudFormation Vorlage ist eine Textdatei im JSON-Format, die AWS Ressourcen deklariert, aus denen ein Stack besteht. Weitere Informationen zu CloudFormation Vorlagen finden Sie unter Grundlagen von Vorlagen lernen.
2. Klicken Sie auf Next (Weiter) und geben Sie die Stack-Details ein.
3. Geben Sie unter Parameter für jeden Cluster Folgendes ein:

   - Der Namen Ihres Amazon-Redshift-Clusters, zum Beispiel **`ra3-consumer-cluster`**
   - Der Name Ihrer Datenbank, zum Beispiel **`dev`**
   - Der Name eines Benutzers Ihrer Datenbank, zum Beispiel **`consumeruser`**

   Wir empfehlen die Verwendung von Testclustern, da der Stack mehrere Datenbankobjekte erstellt.

   Wählen Sie Next (Weiter).
4. Die Stack-Optionen werden angezeigt.

   Klicken Sie auf Next (Weiter), um die Standardeinstellungen zu übernehmen.

5. Wählen Sie unter Funktionen die Option Ich bestätige, dass AWS CloudFormation möglicherweise IAM-Ressourcen erstellt werden.
6. Wählen Sie Stack erstellen aus.

CloudFormation dauert etwa 10 Minuten, um den Amazon Redshift Redshift-Stack mithilfe der Vorlage zu erstellen, wobei ein Datashare namens erstellt wird. `myproducer_share` Der Stack erstellt das Datashare in der Datenbank, die in den Stack-Details angegeben ist. Nur Objekte aus dieser Datenbank können freigegeben werden.

Falls während der Erstellung des Stacks ein Fehler auftritt, unternehmen Sie folgende Schritte:

- Stellen Sie sicher, dass Sie den richtigen Cluster-Namen, den Datenbanknamen und den Namen des Datenbankbenutzers für jeden Redshift-Cluster eingegeben haben.
- Stellen Sie sicher, dass Ihr Cluster RA3-Knoten hat.
- Stellen Sie sicher, dass Sie mit einem Benutzer angemeldet sind, der die Berechtigung für eine IAM-Rolle und eine Lambda-Funktion hat. Weitere Informationen zum Erstellen von IAM-Rollen finden Sie unter Erstellen von IAM-Rollen. Weitere Informationen zu den Richtlinien für die Erstellung der Λ-Funktion finden Sie unter Funktionsentwicklung.

Abfragen des von Ihnen erstellten Datashares

Stellen Sie für das folgende Verfahren sicher, dass Sie über die erforderlichen Berechtigungen zum Ausführen von Abfragen in jedem beschriebenen Cluster verfügen.

So fragen Sie Ihr Datashare ab:

1. Stellen Sie mithilfe eines Clienttools wie dem Amazon Redshift Query Editor v2 eine Connect zum Producer-Cluster in der Datenbank her, die Sie bei der Erstellung Ihres CloudFormation Stacks eingegeben haben.
2. Starten Sie eine Abfrage nach Datashares.

```
SHOW DATASHARES;

+------------------+-------------+-----------------+-------------------
+------------+------------+---------------------+-----------+------------------
+-------------------------------------+
```

```
|    share_name    | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace          |
+------------------+-------------+-----------------+-------------------
+------------+------------+--------------------+-----------+-----------------
+-------------------------------------+
| myproducer_share | 100         | sample_data_dev | myconsumer_db     | INBOUND
  | NULL       | true               | NULL      |  producer-acct   |        your-
producer-namespace       |
+------------------+-------------+-----------------+-------------------
+------------+------------+--------------------+-----------+-----------------
+-------------------------------------+
```

Der vorhergehende Befehl gibt den Namen des Datashares zurück, das vom Stack erstellt wurde, namens `myproducer_share`. Es gibt auch den Namen der Datenbank zurück, die mit der Datashare verknüpft ist, `myconsumer_db`.

Kopieren Sie die Namespace-Kennung des Produzenten; sie wird in einem späteren Schritt benötigt.

3. Beschreiben Sie Objekte im Datashare.

```
DESC DATASHARE myproducer_share;

+------------------+-------------------------------------+------------
+------------------+-------------+-------------------------------------
+-------------+
| producer_account |          producer_namespace          | share_type |
 share_name    | object_type |             object_name             | include_new |
+------------------+-------------------------------------+------------
+------------------+-------------+-------------------------------------
+-------------+
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | schema      | myproducer_schema                   | true
 |
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | table       | myproducer_schema.tickit_sales      | NULL
 |
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | view        | myproducer_schema.ticket_sales_view | NULL
 |
```

```
+-----------------+--------------------------------------+-----------
+-----------------+-------------+-----------------------------------
+-------------+
```

Wenn Sie das Datashare beschreiben, werden Eigenschaften für Tabellen und Ansichten zurückgegeben. Der Stack fügt beispielsweise Tabellen und Ansichten mit Beispieldaten zur Produzentendatenbank hinzu, zum Beispiel `tickit_sales` und `tickit_sales_view`. Weitere Informationen zur TICKIT-Beispieldatenbank finden Sie unter Beispieldatenbank.

Sie müssen keine Berechtigungen für das Datashare delegieren, um Abfragen auszuführen. Der Stack gewährt die erforderlichen Berechtigungen.

4. Verbinden Sie sich mit Ihrem Client-Tool mit dem Konsumenten-Cluster. Beschreiben Sie das Datashare und geben Sie den Namespace des Produzenten an.

```
DESC DATASHARE myproducer_share OF NAMESPACE '<namespace id>'; --specify the unique
 identifier for the producer namespace

+-----------------+--------------------------------------+-----------
+-----------------+-------------+-----------------------------------
+-------------+
| producer_account |          producer_namespace          | share_type |
 share_name    | object_type |             object_name             | include_new |
+-----------------+--------------------------------------+-----------
+-----------------+-------------+-----------------------------------
+-------------+
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | schema      | myproducer_schema                   | NULL
 |
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | table       | myproducer_schema.tickit_sales      | NULL
 |
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | view        | myproducer_schema.ticket_sales_view | NULL
 |
+-----------------+--------------------------------------+-----------
+-----------------+-------------+-----------------------------------
+-------------+
```

5. Sie können Tabellen im Datashare abfragen, indem Sie die Datenbank und das Schema des Datashares angeben. Weitere Informationen finden Sie unter Beispiele für die Verwendung einer datenbankübergreifenden Abfrage. Die folgenden Abfragen geben Daten zu Verkäufen

und Verkäufern aus der SALES-Tabelle in der TICKIT-Beispieldatenbank zurück. Weitere Informationen finden Sie unter Tabelle SALES.

```
SELECT * FROM myconsumer_db.myproducer_schema.tickit_sales_view;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |       728 |
     109.2 | 2008-02-18 02:36:48 |
|       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |        76 |
      11.4 | 2008-06-06 05:00:16 |
|       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |       350 |
      52.5 | 2008-06-06 08:26:17 |
|       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |       175 |
     26.25 | 2008-06-09 08:38:52 |
|       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |       154 |
      23.1 | 2008-08-31 09:17:02 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

> **Note**
>
> Die Abfrage wird mit der Ansicht im freigegebenen Schema ausgeführt. Sie können keine direkte Verbindung zu Datenbanken herstellen, die aus Datashares erstellt wurden. Sie sind schreibgeschützt.

6. Wenn Sie eine Abfrage ausführen möchten, die Aggregationen enthält, verwenden Sie das folgende Beispiel.

```
SELECT * FROM myconsumer_db.myproducer_schema.tickit_sales ORDER BY 1,2 LIMIT 5;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

```
|       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |       728 |
     109.2 | 2008-02-18 02:36:48 |
|       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |        76 |
      11.4 | 2008-06-06 05:00:16 |
|       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |       350 |
      52.5 | 2008-06-06 08:26:17 |
|       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |       175 |
     26.25 | 2008-06-09 08:38:52 |
|       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |       154 |
      23.1 | 2008-08-31 09:17:02 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Die Abfrage gibt Verkaufs- und Verkäuferdaten aus den Beispieldaten für TICKIT zurück.

Weitere Beispiele für Datashare-Abfragen finden Sie unter Teilen des Lesezugriffs auf Daten innerhalb eines AWS-Konto.

Verwaltung der Datenfreigabe mit Schreibvorgängen über die Konsole (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zur Einrichtung des Tracks PREVIEW_2023 finden Sie in einer der folgenden Seiten:

- Für Amazon Redshift Serverless Vorschau: Erstellen einer Vorschau-Arbeitsgruppe
- Vorschau für von Amazon Redshift bereitgestellte Cluster: Erstellen eines Vorschau-Clusters

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau).

Verwenden Sie die Amazon-Redshift-Konsole, um Datashares zu verwalten, die in Ihrem Konto erstellt oder von anderen Konten freigegeben wurden.

Verbinden mit einer Datenbank (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau).

Verbinden Sie sich mit einer Datenbank, um Datenbanken und Objekte in Datenbanken in diesem Cluster oder Datashares anzuzeigen.

Die Benutzeranmeldeinformationen, die zum Herstellen einer Verbindung mit einer angegebenen Datenbank verwendet werden, müssen über die erforderlichen Berechtigungen verfügen, um alle Datashares anzuzeigen.

Führen Sie einen der folgenden Schritte aus, wenn keine lokale Verbindung besteht:

- Klicken Sie auf der Detailseite des Clusters in der Registerkarte Databases (Datenbanken) auf den Abschnitt Databases (Datenbanken) oder Datashare objects (Datashare-Objekte) und wählen Sie Connect to database (Mit Datenbank verbinden) aus, um Datenbankobjekte im Cluster anzuzeigen.
- Gehen Sie auf der Seite Cluster-Details auf der Registerkarte Datashares wie folgt vor:
  - Wählen Sie im Abschnitt Datashares from other clusters (Datashare von anderen Clustern) Connect to database (Mit Datenbank verbinden) aus, um Datashares von anderen Clustern anzuzeigen.
  - Wählen Sie im Abschnitt Datashares created in my cluster (In meinem Cluster erstellte Datashares) Connect to database (Mit Datenbank verbinden) aus, um Datashares in Ihrem Cluster anzuzeigen.
- Gehen Sie im Fenster Connect to database (Mit Datenbank verbinden) wie folgt vor:

- Wenn Sie Create a new connection (Neue Verbindung herstellen) auswählen, wählen Sie AWS Secrets Manager aus, um ein gespeichertes Secret zur Authentifizierung des Zugriffs auf die Verbindung auszuwählen.

  Oder wählen Sie Temporary credentials (Temporäre Anmeldeinformationen), um Datenbank-Anmeldeinformationen zum Authentifizieren des Zugriffs für die Verbindung zu verwenden. Geben Sie Werte für den Datenbanknamen und den Datenbankbenutzer an.

  Wählen Sie Connect (Verbinden) aus.
- Klicken Sie auf Use a recent connection (Verwenden einer kürzlich verwendeten Verbindung), um eine Verbindung zu einer anderen Datenbank herzustellen, für die Sie über die erforderlichen Berechtigungen verfügen.

  Amazon Redshift stellt die Verbindung automatisch her.

Nachdem die Datenbankverbindung hergestellt wurde, können Sie mit dem Erstellen von Datashares, dem Abfragen von Datashares oder dem Erstellen von Datenbanken aus Datashares beginnen.

Erstellen von Datashares und Hinzufügen von Objekten (Vorschau)

Erstellen von Datenaustauschen

> Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern.
> Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau).

Als Administrator eines Produzenten-Clusters können Sie auf der Detailseite zu Clustern Datashare auf den Registerkarten Databases (Datenbanken) oder Datashares erstellen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.

2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Führen Sie auf der Seite zu den Cluster-Details die folgenden Schritte aus:
   - Wählen Sie im Abschnitt Database (Datenbank) auf der Registerkarte Databases (Datenbanken) eine Datenbank aus. Die Seite mit den Datenbankdetails wird angezeigt.

     Wählen Sie Create datashare (Datashare erstellen) aus. Sie können einen Datashare nur aus einer lokalen Datenbank erstellen. Falls Sie noch keine Verbindung zur Datenbank hergestellt haben, wird die Seite Verbindung zur Datenbank herstellen geöffnet. Befolgen Sie die Schritte in Verbinden mit einer Datenbank (Vorschau), um eine Verbindung zu einer Datenbank herzustellen. Wenn eine kürzlich hergestellte Verbindung besteht, wird die Seite Datashare erstellen angezeigt.
   - Verbinden Sie sich auf der Registerkarte Datashares im Abschnitt Datashares mit einer Datenbank, sofern noch keine Verbindung hergestellt wurde.

     Wählen Sie im Abschnitt Datashares created in my cluster (In meinem Cluster erstellte Datashare) Create datashare (Datashare erstellen) aus. Die Seite Datashare erstellen wird angezeigt.
4. Von hier aus können Sie Datenbankobjekte verschiedener Typen hinzufügen. Wählen Sie die Schaltfläche Hinzufügen, um Objekte hinzuzufügen. Es wird ein Dialogfeld angezeigt. Führen Sie die folgenden Schritte aus:

   1. Wählen Sie ein Schema oder mehrere Schemata aus. Dadurch können Objekte aus den Schemata hinzugefügt werden.
   2. Wählen Sie Objekttypen aus den Schemata aus.

      Von hier aus können Sie einige Optionen zum Hinzufügen von Objekten wählen:

      - Hinzufügen bestimmter Objekte aus Schemata – Wenn Sie diese Option wählen, werden einzelne Objekte nach Namen aufgelistet. Sie können Objekte auswählen und sie dem Datashare hinzufügen. Sie können beispielsweise bestimmte Tabellen und Gespeicherte Prozeduren hinzufügen, wenn Sie möchten. Dann sind die Tabellen und gespeicherten Prozeduren aus dem Schema, das Sie ausgewählt haben, in dem Datashare vorhanden. Das Festlegen von Berechtigungen wird in den nachfolgenden Schritten näher erläutert. Fahren Sie mit Ansichten und anderen Typen fort und wählen Sie die hinzuzufügenden Objekte aus.

- Hinzufügen aller vorhandenen Objekte aus den ausgewählten Objekttypen zu dem Schema – Damit werden alle Objekte hinzugefügt.

3. Sie können auch Zukünftige Objekte hinzufügen auswählen. Wenn Sie sich dafür entscheiden, dem Schema hinzugefügte Datashare-Objekte einzubeziehen, bedeutet dies, dass dem Schema hinzugefügte Objekte automatisch dem Datashare hinzugefügt werden.
4. Wählen Sie Hinzufügen, um den Abschnitt zu vervollständigen und die Objekte hinzuzufügen. Sie sind unter den Datashare-Objekten aufgeführt.
5. Nachdem Sie Objekte hinzugefügt haben, können Sie einzelne Objekte auswählen und deren Berechtigungen bearbeiten. Wenn Sie ein Schema auswählen, wird ein Dialogfeld angezeigt, in dem Sie gefragt werden, ob Sie bereichsbezogene Berechtigungen hinzufügen möchten. Dadurch verfügt jedes vorhandene oder dem Schema hinzugefügte Objekt über einen vorausgewählten Satz von Berechtigungen, die für den Objekttyp geeignet sind. Der Administrator kann beispielsweise festlegen, dass alle hinzugefügten Tabellen über SELECT- und UPDATE-Berechtigungen verfügen.
6. Nachdem Sie Schemaberechtigungen konfiguriert haben, können Sie sich weitere Objekttypen ansehen und deren Berechtigungen auswählen. Sie können beispielsweise einer bestimmten Tabelle UPDATE-Berechtigungen hinzufügen.
7. Im Bereich Datenverbraucher können Sie Namespaces oder AWS Konten als Nutzer des Datenaustauschs hinzufügen.
8. Wählen Sie Datashare erstellen aus, um Ihre Änderungen zu speichern.

Nachdem Sie das Datashare erstellt haben, wird es in der Liste unter In meinem Namespace erstellte Datashares angezeigt. Wenn Sie ein Datashare aus der Liste auswählen, können Sie dessen Nutzer, Objekte und andere Eigenschaften anzeigen.

Hinzufügen von Datenverbrauchern zu Datashares

Sie können einen oder mehrere Datenkonsumenten zu den Datashares hinzufügen. Datenkonsumenten können Cluster-Namespaces sein, die Amazon-Redshift-Cluster oder AWS-Konten eindeutig identifiziert haben.

Sie müssen die Freigabe von Datashares mit Clustern mit öffentlichem Zugriff explizit ein- oder ausschalten.

- Klicken Sie auf Add cluster namespaces to the datashare (Cluster-Namespaces zum Datashare hinzufügen). Namespaces sind GUIDs für Amazon-Redshift-Cluster.

- Wählen Sie AWS-Konten hinzufügen, um Konten zum Datashare hinzuzufügen. Die angegebenen Benutzer AWS-Konten müssen über Zugriffsberechtigungen für den Datashare verfügen.

Autorisierung oder Entfernen von Autorisierung aus Datenaustauschen (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern.
Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Teilen des Schreibzugriffs auf Daten (Vorschau).

Wählen Sie als Administrator eines Produzenten-Clusters aus, welchen Datenkonsumenten Sie eine Autorisierung für Datashares gewähren oder entziehen möchten. Berechtigte Datenkonsumenten erhalten Benachrichtigungen, um Maßnahmen auf Datashare zu ergreifen. Wenn Sie einen Cluster-Namespace als Datenverbraucher hinzufügen, müssen Sie keine Autorisierung durchführen.

Voraussetzung: Um die Autorisierung für das Datashare zu autorisieren oder zu entfernen, muss mindestens ein Datenkonsument zum Datashare hinzugefügt werden.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datashares aus. Von hier aus können Sie eine Liste mit dem Namen Datashares-Konsumenten einsehen. Wählen Sie einen oder mehrere Konsumenten-Cluster aus, die Sie autorisieren möchten. Klicken Sie dann auf Authorize (Autorisieren).
3. Das Dialogfeld Konto autorisieren wird angezeigt. Sie können zwischen mehreren Autorisierungstypen wählen.

   - Schreibgeschützt auf [Cluster-Name oder Arbeitsgruppenname] – Dies bedeutet, dass für den Benutzer keine Schreibberechtigungen verfügbar sind, auch wenn der Ersteller des Datashares Schreibberechtigungen gewährt hat.
   - Schreiben und Lesen auf [Cluster-Name oder Arbeitsgruppenname] – Dies bedeutet, dass für den Benutzer alle Berechtigungen, einschließlich Schreibberechtigungen, verfügbar sind.

4. Wählen Sie Speichern.

Sie können sich auch AWS Data Exchange als Verbraucher autorisieren.

1. Wenn Sie beim Erstellen des Datashares die Option In AWS Glue Data Catalog veröffentlichen ausgewählt haben, können Sie die Berechtigung für das Datashare nur einem Lake-Formation-Konto gewähren.

   Bei AWS Data Exchange Datashare können Sie jeweils nur eine Datenfreigabe autorisieren.

   Wenn Sie eine Datenfreigabe autorisieren, teilen Sie AWS Data Exchange die Datenfreigabe mit dem AWS Data Exchange Dienst und ermöglichen so AWS Data Exchange , den Zugriff auf die Datenfreigabe in Ihrem Namen zu verwalten. AWS Data Exchange ermöglicht Verbrauchern den Zugriff, indem beim Abonnieren der Produkte Verbraucherkonten als Datenverbraucher zum AWS Data Exchange Datenaustausch hinzugefügt werden. AWS Data Exchange hat keinen Lesezugriff auf den Datashare.
2. Wählen Sie Speichern.

Nachdem Datenkonsumenten autorisiert wurden, können sie auf Datashare-Objekte zugreifen und eine Konsumenten-Datenbank erstellen, um die Daten abzufragen.

Entfernen einer Autorisierung:

Wählen Sie einen oder mehrere Konsumenten-Cluster aus, deren Autorisierung Sie entfernen möchten. Wählen Sie dann Remove authorization (Autorisierung entfernen) aus.

Nachdem die Autorisierung entfernt wurde, verlieren Datenkonsumenten sofort den Zugriff auf das Datashare.

Zuordnen oder Ablehnen von Datashares als Konsument (Vorschau)

Zuordnen von Datashares

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu

verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Schreibzugriff auf Daten teilen (Vorschau).

Als Consumer-Cluster-Administrator können Sie eine oder mehrere Datenfreigaben, die von anderen Konten gemeinsam genutzt werden, Ihrem gesamten AWS Konto oder bestimmten Cluster-Namespaces in Ihrem Konto zuordnen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datashares aus. Die Seite mit den Datashares wird angezeigt. Wählen Sie From other accounts (Von anderen Konten) aus.
3. Wählen Sie im Abschnitt Datashares from other accounts (Datashares von anderen Konten) den Datashare aus, den Sie verknüpfen möchten. Wählen Sie dazu Associate (Verknüpfen) aus. Wenn die Seite Datashare zuordnen angezeigt wird, wählen Sie einen der folgenden Zuordnungstypen aus:

   - Wählen Sie Gesamtes AWS Konto aus, um alle vorhandenen und future Cluster-Namespaces in verschiedenen AWS Regionen in Ihrem AWS Konto mit dem Datashare zu verknüpfen.

     Wenn das Datashare auf dem veröffentlicht wird AWS Glue Data Catalog, können Sie das Datashare nur dem gesamten Konto zuordnen. AWS
4. Von hier aus können Sie Zulässige Berechtigungen auswählen. Es stehen folgende Optionen zur Verfügung:

   - Schreibgeschützt – Wenn Sie „Schreibgeschützt“ wählen, sind Schreibberechtigungen wie UPDATE oder INSERT für den Konsumenten nicht verfügbar, auch wenn diese Berechtigungen vom Produzenten erteilt und autorisiert wurden.
   - Lesen und Schreiben – Benutzer von Konsumenten-Datashares verfügen über alle Berechtigungen, sowohl Lese- als auch Schreibberechtigungen, die vom Produzenten gewährt und autorisiert wurden.
5. Oder wählen Sie Bestimmte AWS Regionen und Cluster-Namespaces, um dem Datashare eine oder mehrere AWS Regionen und bestimmte Cluster-Namespaces zuzuordnen. Wählen Sie Region hinzufügen, um dem Datashare bestimmte AWS Regionen und Cluster-Namespaces hinzuzufügen. Die Seite „Region hinzufügen“ wird angezeigt. AWS

6. Wählen Sie eine AWS Region aus.
7. Führen Sie eine der folgenden Aufgaben aus:

   - Wählen Sie Add all cluster namespaces (Alle Cluster-Namespaces hinzufügen) aus, um alle vorhandenen und zukünftigen Cluster-Namespaces in dieser Region zum Datashare hinzuzufügen.
   - Wählen Sie Add specific cluster namespace (Bestimmten Cluster-Namespace hinzufügen) aus, um einen oder mehrere bestimmte Cluster-Namespaces dem Datashare hinzuzufügen.
   - Wählen Sie einen oder mehrere Cluster-Namespaces und wählen Sie Region hinzufügen. AWS
8. Wählen Sie Associate aus.

Der Produzent kann zurückgehen und die Einstellungen für eine Autorisierung ändern, was sich auf die Zuordnungseinstellungen der Konsumenten auswirken kann.

Wenn Sie das Datashare einem Lake-Formation-Konto zuordnen, wechseln Sie zur Lake-Formation-Konsole, um eine Datenbank zu erstellen, und definieren Sie dann die Berechtigungen über die Datenbank. Weitere Informationen finden Sie unter Einrichten von Berechtigungen für Amazon Redshift Redshift-Datenfreigaben im AWS Lake Formation Entwicklerhandbuch. Sobald Sie eine AWS Glue Datenbank oder eine Verbunddatenbank erstellt haben, können Sie den Abfrage-Editor v2 oder einen beliebigen beliebigen SQL-Client mit Ihrem Consumer-Cluster verwenden, um die Daten abzufragen.

Nachdem das Datashare verknüpft wurde, werden die Datashares verfügbar.

Sie können die Zuordnung des Datashares auch jederzeit ändern. Wenn die Zuordnung von bestimmten AWS Regionen und Cluster-Namespaces zum gesamten AWS Konto geändert wird, überschreibt Amazon Redshift die spezifischen Regions- und Cluster-Namespace-Informationen mit Kontoinformationen. AWS Alle AWS Regionen und Cluster-Namespaces im Konto haben dann Zugriff auf den Datashare. AWS

Wenn die Zuordnung von bestimmten Cluster-Namespaces zu allen Cluster-Namespaces in der angegebenen AWS Region geändert wird, haben dann alle Cluster-Namespaces in dieser Region Zugriff auf den Datashare.

Entfernen der Zuordnung des Datashares zu Datenkonsumenten

Als Administrator eines Konsumenten-Clusters können Sie die Zuordnung von Datashares von Datenkonsumenten entfernen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datashares aus. Die Seite mit den Datashares wird angezeigt.
3. Wählen Sie From other accounts (Von anderen Konten) aus.
4. Im Abschnitt Datashares from other accounts (Datashares von anderen Konten) wählen Sie das Datashare, um die Zuordnung zu Datenkonsumenten zu löschen.
5. Wählen Sie im Abschnitt Data consumers (Datenkonsumenten) einen oder mehrere Datenkonsumenten, um die Zuordnung davon zu entfernen. Wählen Sie dann Remove association (Zuordnung entfernen) aus.
6. Wählen Sie auf der Seite „Remove association“ (Zuordnung entfernen) Remove association (Zuordnung entfernen) aus.

Nachdem die Zuordnung entfernt wurde, verlieren Datenkonsumenten sofort den Zugriff auf den Datashare. Sie können jederzeit Änderungen an der Zuordnung der Datenkonsumenten vornehmen.

Ablehnen von Datashares

Als Administrator eines Konsumenten-Clusters können Sie alle Datashares mit dem Status verfügbar oder aktiv ablehnen. Nachdem Sie ein Datashare abgelehnt haben, können Benutzer von Konsumenten-Clustern nicht mehr auf das Datashare zugreifen. Amazon Redshift gibt das abgelehnte Datashare nicht zurück, wenn die API-Operation `DescribeDataSharesForConsumer` aufgerufen wird. Wenn der Administrator des Produzenten-Clusters die API-Operation `DescribeDataSharesForProducer` ausführt, sieht er, dass das Datashare abgelehnt wurde. Sobald ein Datashare abgelehnt wurde, kann der Producer-Cluster-Administrator das Datashare erneut für einen Consumer-Cluster autorisieren, und der Consumer-Cluster-Administrator kann wählen, ob er sein AWS Konto dem Datashare zuordnen oder es ablehnen möchte.

Wenn Ihr AWS Konto eine Zuordnung zu einem Datashare und eine ausstehende Verknüpfung zu einem von Lake Formation verwalteten Datashare hat, wird durch das Ablehnen der von Lake Formation verwalteten Datenfreigabe auch das ursprüngliche Datashare zurückgewiesen. Um eine spezifische Zuordnung abzulehnen, kann der Administrator des Produzenten-Clusters die

Autorisierung für ein spezifisches Datashare entfernen. Diese Aktion wirkt sich nicht auf andere Datashares aus.

Um eine Datenfreigabe abzulehnen, verwenden Sie die Konsole, den API-Vorgang oder in der. AWS `RejectDataShare` `reject-datashare` AWS CLI

So lehnen Sie eine Datenfreigabe mithilfe der Konsole ab: AWS

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Datenfreigaben aus.
3. Wählen Sie From other accounts (Von anderen Konten) aus.
4. Wählen Sie im Abschnitt Datashares from other accounts (Datashare von anderen Konten) das Datashare aus, das Sie ablehnen möchten. Wenn die Seite Decline datashare (Datashare ablehnen) angezeigt wird, wählen Sie Decline (Ablehnen) aus.

Nachdem Sie die Datashares abgelehnt haben, können Sie die Änderung nicht rückgängig machen. Amazon Redshift entfernt die Datashares aus der Liste. Damit das Datashare wieder angezeigt wird, muss der Administrator des Produzenten-Clusters es erneut autorisieren.

Verwalten vorhandener Datashares (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern.
Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Schreibzugriff auf Daten teilen (Vorschau).

Anzeigen von Datashares

Anzeigen von Datashares auf den Registerkarten DATASHARES oder CLUSTER.

- Verwenden Sie die Registerkarte DATASHARES, um Datashares in Ihrem oder anderen Konten aufzulisten.
  - Wenn Sie Datashares anzeigen möchten, die in Ihrem Konto erstellt wurden, wählen Sie In my account (In meinem Konto) und anschließend das Datashare, das Sie anzeigen möchten.
  - Um Datashares anzuzeigen, die von anderen Konten freigegeben werden, wählen Sie From other accounts (Von anderen Konten) und dann das Datashare aus, den Sie anzeigen möchten.
- Nutzen Sie die Registerkarte CLUSTER, um Datashares in Ihrem Cluster oder von anderen Clustern aufzulisten.

  Verbindung zu einer Datenbank herstellen Weitere Informationen finden Sie unter Verbinden mit einer Datenbank (Vorschau).

  Wählen Sie anschließend ein Datashare aus dem Abschnitt Datashares from other clusters (Datashares von anderen Clustern) oder dem Abschnitt Datashares created in my cluster (In meinem Cluster erstellte Datashares) aus, um detaillierte Informationen dazu anzuzeigen.

Entfernen von Datashare-Objekten aus Datashares

Sie können eines oder mehrere Objekte mithilfe des folgenden Verfahrens aus einem Datashare entfernen.

Entfernen eines oder mehrerer Objekte aus einem Datashare

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares.
4. Wählen Sie im Abschnitt Datashares created in my account (In meinem Konto erstellte Datashares) Connect to database (Mit Datenbank verbinden). Weitere Informationen finden Sie unter Verbinden mit einer Datenbank (Vorschau).
5. Wählen Sie ein Datashare aus, das Sie bearbeiten möchten, und anschließend Edit (Bearbeiten). Die Detailseite zum Datashare wird angezeigt.
6. Gehen Sie wie folgt vor, um eines oder mehrere Datashare-Objekte aus dem Datashare zu entfernen.

- Wählen Sie eines oder mehr Schemas aus, um Schemas aus dem Datashare zu entfernen. Wählen Sie dann Remove (Entfernen) aus. Amazon Redshift entfernt die angegebenen Schemas und alle Objekte der angegebenen Schemas aus dem Datashare.
- Um Tabellen und Ansichten aus dem Datashare zu entfernen, wählen Sie eine oder mehrere Tabellen und Ansichten aus. Wählen Sie dann Remove (Entfernen) aus. Alternativ können Sie Remove by schema (Nach Schema entfernen) auswählen, um alle Tabellen und Ansichten in den angegebenen Schemas zu entfernen.
- Um benutzerdefinierte Funktionen aus dem Datashare zu entfernen, wählen Sie eine oder mehrere benutzerdefinierte Funktionen aus. Wählen Sie dann Remove (Entfernen) aus. Alternativ können Sie Remove by schema (Nach Schema entfernen) auswählen, um alle benutzerdefinierten Funktionen in den angegebenen Schemas zu entfernen.

Entfernen von Datenkonsumenten aus Datashares

Sie können einen oder mehrere Datenkonsumenten aus einem Datashare entfernen. Datenverbraucher können Cluster-Namespaces sein, die Amazon Redshift Redshift-Cluster oder -Konten eindeutig identifizieren. AWS

Wählen Sie einen oder mehrere Datenverbraucher entweder aus den Cluster-Namespace-IDs oder dem AWS Konto aus und wählen Sie dann Entfernen.

Amazon Redshift entfernt die angegebenen Datenkonsumenten aus dem Datashare. Sie können ab sofort nicht mehr auf das Datashare zugreifen.

Bearbeiten von Datashares, die in Ihrem Konto erstellt wurden

Bearbeiten Sie über die Konsole Datashares, die in Ihrem Konto erstellt wurden. Stellen Sie zunächst eine Verbindung zu einer Datenbank her, um die Liste der in Ihrem Konto erstellten Datashares anzuzeigen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares.
4. Wählen Sie im Abschnitt In meinem Konto erstellte Datashares Mit Datenbank verbinden.

5. Wählen Sie ein Datashare aus, das Sie bearbeiten möchten, und anschließend Edit (Bearbeiten). Die Detailseite zum Datashare wird angezeigt.
6. Nehmen Sie Änderungen im Abschnitt Datashare objects (Datashare-Objekte) oder im Abschnitt Data consumers (Datenkonsumenten) vor.

> **Note**
>
> Wenn Sie sich dafür entschieden haben, Ihr Datashare auf dem zu veröffentlichen AWS Glue Data Catalog, können Sie die Konfiguration nicht bearbeiten, um das Datashare auf anderen Amazon Redshift Redshift-Konten zu veröffentlichen.

7. Wählen Sie Save Changes (Änderungen speichern).

Amazon Redshift aktualisiert Ihren Datenaustausch mit den Änderungen.

Löschen von -Datashares, die in Ihrem Konto erstellt wurden

Löschen Sie über die Konsole Datashares, die in Ihrem Konto erstellt wurden. Stellen Sie zunächst eine Verbindung zu einer Datenbank her, um die Liste der in Ihrem Konto erstellten Datashares anzuzeigen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares. Die Liste mit den Datashares wird angezeigt.
4. Wählen Sie im Abschnitt In meinem Konto erstellte Datashares Mit Datenbank verbinden.
5. Wählen Sie einen oder mehrere Datashares aus, die Sie löschen möchten, und wählen Sie dann Delete (Löschen) aus. Die Seite „Delete datashares“ (Datashares löschen) wird angezeigt.

   Beim Löschen eines Datashares, das für Lake Formation freigegeben wurde, werden nicht automatisch die zugehörigen Berechtigungen in Lake Formation entfernt. Um sie zu entfernen, wechseln Sie zur Lake-Formation-Konsole.
6. Geben Sie Delete (Löschen) ein, um die ausgewählten Datashares zu löschen.
7. Wählen Sie Delete (Löschen).

Nachdem Datashares gelöscht wurden, verlieren Konsumenten von Datashares den Zugriff auf die Datashares.

Abfragen von Datashares (Vorschau)

Dies ist eine Vorabdokumentation für das Multi-Data-Warehouse-Feature über die gemeinsame Nutzung von Daten für Amazon Redshift, die in der öffentlichen Vorschau im Track PREVIEW_2023 verfügbar ist. Sowohl die Dokumentation als auch die Funktion können sich ändern.
Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Die Bedingungen für Vorversionen finden Sie unter Beta-Service-Teilnahme in den AWS -Servicebedingungen.

Weitere Informationen zu den ersten Schritten mit der gemeinsamen Nutzung von Daten finden Sie unter Schreibzugriff auf Daten teilen (Vorschau).

Erstellen von Datenbanken aus Datashares

Erstellen Sie eine Datenbank aus dem Datashare, um mit der Abfrage von Daten im Datashare zu beginnen. Sie können nur eine Datenbank aus einem angegebenen Datashare erstellen.

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Clusters (Cluster) und dann Ihren Cluster aus. Die Cluster-Detailseite wird angezeigt.
3. Klicken Sie auf Datashares. Die Liste mit den Datashares wird angezeigt.
4. Wählen Sie im Abschnitt Datashares from other clusters (Datashares von anderen Clustern) Connect to database (Mit Datenbank verbinden) aus. Weitere Informationen finden Sie unter Verbinden mit einer Datenbank (Vorschau).
5. Wählen Sie ein Datashare aus, aus dem Sie Datenbanken erstellen möchten, und wählen Sie anschließend Create database from datashare (Datenbank aus Datashare erstellen) aus. Auf der Seite des Datashares wird „Create database“ (Datenbank erstellen) angezeigt.
6. Geben Sie bei Datenbankname einen Datenbanknamen an. Der Datenbankname muss aus 1–64 alphanumerischen Zeichen bestehen (nur in Kleinbuchstaben) und es darf kein reserviertes Wort sein.
7. Wählen Sie Create (Erstellen) aus.

Nachdem die Datenbank erstellt wurde, können Sie Daten in der Datenbank abfragen oder Schreibvorgänge ausführen, sofern diese vom Konsumentenadministrator genehmigt, autorisiert und zugeordnet wurden.

Erfassen und Abfragen von halbstrukturierten Daten in Amazon Redshift

Durch die Unterstützung halbstrukturierter Daten in Amazon Redshift können Sie halbstrukturierte Daten in Ihren Amazon-Redshift-Data-Warehouses erfassen und speichern. Mithilfe des SUPER-Datentyps und der PartiQL-Sprache erweitert Amazon Redshift die Data-Warehouse-Funktion, um sowohl SQL- als auch NoSQL-Datenquellen zu integrieren. Auf diese Weise ermöglicht Amazon Redshift effiziente Analysen von relationalen und halbstrukturierten gespeicherten Daten wie JSON.

Amazon Redshift unterstützt halbstrukturierte Daten auf zwei Weisen: den SUPER-Datentyp und Amazon Redshift Spectrum.

Verwenden Sie den SUPER-Datentyp, wenn Sie kleine Batches von JSON-Daten mit geringer Latenz einfügen oder aktualisieren müssen. Verwenden Sie auch SUPER, wenn Ihre Abfrage starke Konsistenz, vorhersehbare Abfrageleistung, komplexe Abfrageunterstützung und Benutzerfreundlichkeit mit sich entwickelnden Schemata und schemalosen Daten erfordert.

Verwenden Sie Amazon Redshift Spectrum dagegen mit einem offenen Dateiformat, wenn Ihre Datenabfrage die Integration mit anderen AWS Diensten und mit Daten erfordert, die hauptsächlich zu Archivierungszwecken in Amazon S3 gespeichert werden.

Anwendungsfälle für den SUPER-Datentyp

Die Unterstützung halbstrukturierter Daten mithilfe des SUPER-Datentyps in Amazon Redshift bietet überlegene Leistung, Flexibilität und Benutzerfreundlichkeit. Anhand der folgenden Anwendungsfälle soll demonstriert werden, wie Sie die Unterstützung für halbstrukturierte Daten mit SUPER nutzen können.

Schnelles und flexibles Einfügen von JSON-Daten – Amazon Redshift unterstützt schnelle Transaktionen, die JSON parsen und als SUPER-Wert speichern können. Die Insert-Transaktionen können bis zu fünf Mal schneller ausgeführt werden als die gleichen Einfügungen in Tabellen, bei denen die Attribute von SUPER in herkömmliche Spalten aufgeteilt sind. Angenommen, die eingehenden JSON-Daten haben die Form “a”:.., “b”:.., “c”“..., ...}. Sie können das Einfügen um ein Vielfaches beschleunigen, indem Sie die eingehenden JSON-Daten in einer Tabelle TJ mit einer einzigen SUPER-Spalte S speichern, anstatt sie in einer herkömmlichen Tabelle TR mit den Spalten ”a’, ‘b’, “c’ usw. zu speichern. Wenn die JSON-Daten Hunderte von Attributen enthalten, wird der Leistungsvorteil des SUPER-Datentyps beträchtlich.

Außerdem benötigt der SUPER-Datentyp kein reguläres Schema. Sie müssen die eingehenden JSON-Daten vor dem Speichern nicht analysieren und bereinigen. Angenommen, eingehende JSON-Daten haben ein Zeichenfolgeattribut „c“ und andere ein Integerattribut „c“, ohne den SUPER-Datentyp. In diesem Fall müssen Sie entweder die Spalten c_string und c_int trennen oder die Daten bereinigen. Im Gegensatz dazu werden beim SUPER-Datentyp alle JSON-Daten während der Aufnahme ohne Informationsverlust gespeichert. Später können Sie die PartiQL-Erweiterung von SQL verwenden, um die Informationen zu analysieren.

Flexible Abfragen für die Erkennung – Nachdem Sie Ihre halbstrukturierten Daten (z. B. JSON) in einem SUPER-Datenwert gespeichert haben, können Sie sie abfragen, ohne ein Schema vorschreiben zu müssen. Sie können die dynamische Typisierung von PartiQL und Lax-Semantik verwenden, um Ihre Abfragen auszuführen und die tief verschachtelten Daten zu ermitteln, die Sie benötigen, ohne dass Sie vor der Abfrage ein Schema vorschreiben müssen.

Flexible Abfragen ETL-Operationen (Extract, Transform, Load) in konventionelle materialisierte Ansichten – Nachdem Sie Ihre schemalosen und halbstrukturierten Daten in SUPER gespeichert haben, können Sie materialisierte PartiQL-Ansichten verwenden, um die Daten zu analysieren und in materialisierte Ansichten aufzuteilen.

Die materialisierten Ansichten mit den aufgeteilten Daten sind ein gutes Beispiel für Leistungs- und Nutzbarkeitsvorteile für klassische Analysefälle. Wenn Sie Analysen für die aufgeteilten Daten durchführen, bietet die spaltenförmige Organisation der materialisierten Ansichten von Amazon Redshift eine bessere Leistung. Darüber hinaus können Benutzer und Business Intelligence (BI)-Tools, die ein konventionelles Schema für aufgenommene Daten erfordern, Ansichten (entweder materialisiert oder virtuell) als konventionelle Schemadarstellung der Daten verwenden.

Nachdem Ihre mit PartiQL materialisierten Ansichten die in JSON oder SUPER gefundenen Daten in konventionelle spaltenförmige materialisierte Ansichten extrahiert haben, können Sie die materialisierten Ansichten abfragen. Weitere Informationen über die Funktionsweise des SUPER-Datentyps mit materialisierten Ansichten finden Sie unter Verwenden des SUPER-Datentyps mit materialisierten Ansichten.

Sie können dynamische Datenmaskierungsrichtlinien auf `scalar`-Werte in den Pfaden von Spalten des Typs SUPER anwenden. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung. Informationen zur Verwendung der dynamischen Datenmaskierung mit dem SUPER-Datentyp finden Sie unter Verwendung dynamischer Datenmaskierung mit Pfaden des Datentyps SUPER. (Vorschau)

Informationen zum SUPER-Datentyp finden Sie unter Typ SUPER.

Weitere Informationen zu der Verwendung des SUPER-Datentyps finden Sie in den Unterabschnitten zu diesem Thema, beginnend mit SUPER-Beispieldatensatz.

Konzepte zur Verwendung des SUPER-Datentyps

Im Folgenden finden Sie einige Konzepte zum SUPER-Datentyp in Amazon Redshift.

Verstehen des SUPER-Datentyps in Amazon Redshift – Der SUPER-Datentyp ist ein Amazon-Redshift-Datentyp, der die Speicherung von schemalosen Arrays und Strukturen ermöglicht, die Amazon-Redshift-Skalare und möglicherweise verschachtelte Arrays und Strukturen enthalten. Der SUPER-Datentyp kann nativ verschiedene Formate von halbstrukturierten Daten speichern, z. B. JSON oder Daten, die aus dokumentorientierten Quellen stammen. Sie können eine neue SUPER-Spalte hinzufügen, um halbstrukturierte Daten zu speichern und Abfragen zu schreiben, die auf die SUPER-Spalte zugreifen, zusammen mit den üblichen skalaren Spalten. Weitere Informationen zum SUPER-Datentyp finden Sie unter Typ SUPER.

JSON ohne Schema in SUPER aufnehmen – Mit dem flexiblen halbstrukturierten SUPER-Datentyp kann Amazon Redshift JSON ohne Schema in einen SUPER–Wert aufnehmen. Amazon Redshift kann beispielsweise den JSON-Wert [10.5, "first"] in einen SUPER-Wert [10.5, ‘first’] aufnehmen, d. h. ein Array, das die Amazon-Redshift-Dezimalzahl 10.5 und varchar ‘first’ enthält. Amazon Redshift kann JSON mithilfe des Befehls COPY oder der JSON-Parse-Funktion wie json_parse('[10.5, "first"]') in einen SUPER-Wert umwandeln. Sowohl COPY als auch `json_parse` erfassen JSON standardmäßig mit strikter Parsing-Semantik. Sie können auch SUPER-Werte einschließlich Arrays und Strukturen erstellen, indem Sie die Datenbankdaten selbst verwenden.

Die SUPER-Spalte erfordert keine Schemaänderungen, während die unregelmäßigen Strukturen von schemalosem JSON aufgenommen werden. Wenn Sie beispielsweise einen Clickstream analysieren, speichern Sie zunächst in der SUPER-Spalte „Click“-Strukturen mit den Attributen „IP“ und „Zeit“. Sie können ein Attribut „Kunden-ID“ hinzufügen, ohne Ihr Schema zu ändern, um solche Änderungen aufzunehmen.

Das native Format, das für den SUPER-Datentyp verwendet wird, ist ein Binärformat, das weniger Platz benötigt als der JSON-Wert in seiner Textform. Dies ermöglicht eine schnellere Erfassung und Laufzeitverarbeitung von SUPER-Werten bei der Abfrage.

SUPER-Daten mit PartiQL abfragen — PartiQL ist eine abwärtskompatible Erweiterung von SQL-92, die derzeit von vielen Diensten verwendet wird. AWS Mit der Verwendung von PartiQL verbinden vertraute SQL-Konstrukte nahtlos den Zugriff auf die klassischen, tabellarischen SQL-Daten und die

halbstrukturierten Daten von SUPER. Sie können Objekt- und Array-Navigation und unnest-Arrays durchführen. PartiQL erweitert die SQL-Standardsprache, um verschachtelte und mehrwertige Daten deklarativ auszudrücken und zu verarbeiten.

PartiQL ist eine Erweiterung von SQL, bei der verschachtelte und schemalose Daten aus SUPER-Spalten als First-Class-Objekte gelten. PartiQL erfordert nicht, dass alle Abfrageausdrücke bei der Abfragekompilierung typgeprüft werden. Mit diesem Ansatz können Abfrageausdrücke, die den SUPER-Datentyp enthalten, während der Abfrageausführung dynamisch typisiert werden, wenn auf die tatsächlichen Datentypen innerhalb der SUPER-Spalten zugegriffen wird. Außerdem arbeitet PartiQL in einem Lax-Modus, in dem Typ-Inkonsistenzen keine Fehler verursachen, sondern Nullwerte zurückgeben. Die Kombination aus schemaloser und Lax-Abfrageverarbeitung macht PartiQL ideal für ELT-Anwendungen (Extract, Load, Transfer), bei denen Ihre SQL-Abfrage die JSON-Daten auswertet, die in den SUPER-Spalten aufgenommen werden.

Integration mit Redshift Spectrum – Amazon Redshift unterstützt mehrere Aspekte von PartiQL beim Ausführen von Redshift-Spectrum-Abfragen über JSON, Parquet und andere Formate, die verschachtelte Daten aufweisen. Redshift Spectrum unterstützt nur verschachtelte Daten mit Schemata. Mit Redshift Spectrum können Sie beispielsweise deklarieren, dass Ihre JSON-Daten ein Attributnested_schemaful_example in einem Schema ARRAY<STRUCT<a:INTEGER, b:DECIMAL(5,2)>> haben. Das Schema dieses Attributs bestimmt, dass die Daten immer ein Array enthalten, das eine Struktur mit Integer a und Dezimalwert b enthält. Wenn sich die Daten so ändern, dass sie weitere Attribute enthalten, ändert sich auch der Typ. Im Gegensatz dazu benötigt der SUPER-Datentyp kein Schema. Sie können Arrays mit Strukturelementen speichern, die unterschiedliche Attribute oder Typen haben. Außerdem können einige Werte außerhalb von Arrays gespeichert werden.

Weitere Informationen zu Funktionen, die den SUPER-Datentyp unterstützen, finden Sie im Folgenden:

- Funktion ABS
- Die Funktion CEILING (oder CEIL)
- Die Funktion FLOOR
- Die Funktion ROUND
- Die Funktion SIGN
- Die Funktion TRUNC

Überlegungen für SUPER-Daten

Bei der Arbeit mit SUPER-Daten sollten Sie Folgendes berücksichtigen:

- Verwenden Sie Version 1.2.50 des JDBC-Treibers, eine Version des ODBC-Treibers ab 1.4.17 und eine Version des Amazon-Redshift-Python-Treibers ab 2.0.872.

  Informationen zu JDBC-Treibern finden Sie unter Konfigurieren einer JDBC-Verbindung.

  Informationen zu ODBC-Treibern finden Sie unter Konfigurieren einer ODBC-Verbindung.
- Die in den folgenden Themen verwendeten Schemabeispiele finden Sie unter SUPER-Beispieldatensatz.
- Alle SQL-Codebeispiele, die in den folgenden Themen verwendet werden, sind mit demselben S3-Präfix zum Download enthalten. Dazu gehören die Datendefinitionssprache (Data Definition Language, DDL) und COPY-Anweisungen sowie bestimmte geänderte TPC-H-Abfragen, die mit SUPER funktionieren.

  Führen Sie einen der folgenden Schritte aus, um die SQL-Dateien anzuzeigen oder herunterzuladen:
  - Laden Sie die SQL-Datei für das SUPER-Tutorial und die TPC-H-Datei herunter.
  - Führen Sie mit der Amazon S3 CLI den folgenden Befehl aus. Sie können Ihren eigenen Zielpfad verwenden.

    ```
    aws s3 cp s3://redshift-downloads/semistructured/tutorialscripts/semistructured-tutorial.sql /target/path
    aws s3 cp s3://redshift-downloads/semistructured/tutorialscripts/super_tpch_queries.sql /target/path
    ```

Weitere Informationen zu SUPER-Konfigurationen finden Sie unter SUPER-Konfigurationen.

SUPER-Beispieldatensatz

Das Tabellenschema und das Datenmodell für Erfassungs- und Abfragebeispiele sind wie folgt definiert.

```
/*customer-orders-lineitem*/
CREATE TABLE customer_orders_lineitem
(c_custkey bigint
```

```
,c_name varchar
,c_address varchar
,c_nationkey smallint
,c_phone varchar
,c_acctbal decimal(12,2)
,c_mktsegment varchar
,c_comment varchar
,c_orders super
);

/* Datamodel of documents to be stored in c_orders Super column would be as follows*/
ARRAY < STRUCT < o_orderkey:bigint
                          ,o_orderstatus:string
                          ,o_totalprice:double
                          ,o_orderdate:string
                          ,o_orderpriority:string
                          ,o_clerk:string
                          ,o_shippriority:int
                          ,o_comment:string
                          ,o_lineitems:ARRAY < STRUCT < l_partkey:bigint
                                                       ,l_suppkey:bigint
                                                       ,l_linenumber:int
                                                       ,l_quantity:double
                                                       ,l_extendedprice:double
                                                       ,l_discount:double
                                                       ,l_tax:double
                                                       ,l_returnflag:string
                                                       ,l_linestatus:string
                                                       ,l_shipdate:string
                                                       ,l_commitdate:string
                                                       ,l_receiptdate:string
                                                       ,l_shipinstruct:string
                                                       ,l_shipmode:string
                                                       ,l_comment:string
                                                       > >
                           > >

/*part*/
CREATE TABLE part
(
 p_partkey bigint
 ,p_name varchar
 ,p_mfgr varchar
 ,p_brand varchar
```

```
 ,p_type varchar
 ,p_size int
 ,p_container varchar
 ,p_retailprice decimal(12,2)
 ,p_comment varchar
 );

/*region-nations*/
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );

 /* Datamodel of documents to be stored in r_nations Super column would be as follows*/
 ARRAY < STRUCT < n_nationkey:int,n_name:string,n_comment:string > >

/*supplier-partsupp*/
CREATE TABLE supplier_partsupp
(
 s_suppkey bigint
 ,s_name varchar
 ,s_address varchar
 ,s_nationkey smallint
 ,s_phone varchar
 ,s_acctbal double precision
 ,s_comment varchar
 ,s_partsupps super
 );

 /* Datamodel of documents to be stored in s_partsupps Super column would be as
 follows*/
 ARRAY < STRUCT <
 ps_partkey:bigint,ps_availqty:int,ps_supplycost:double,ps_comment:string > >
```

Laden von halbstrukturierten Daten in Amazon Redshift

Verwenden Sie den SUPER-Datentyp, um hierarchische und generische Daten in Amazon Redshift beizubehalten und abzufragen. Amazon Redshift stellt die `json_parse`-Funktion bereit, um Daten im JSON-Format zu analysieren und in die SUPER-Repräsentation zu konvertieren. Amazon

Redshift unterstützt auch das Laden von SUPER-Spalten mit dem COPY-Befehl. Die unterstützten Dateiformate sind JSON, Avro, Text, CSV-Format (durch Kommas getrennte Wert), Parquet und ORC.

Informationen zu den Tabellen, die in den folgenden Beispielen verwendet werden, finden Sie unter SUPER-Beispieldatensatz.

Informationen zur Funktion `json_parse` finden Sie unter Funktion JSON_PARSE.

Die Standardkodierung für den SUPER-Datentyp ist ZSTD.

Parsen von JSON-Dokumenten in SUPER-Spalten

Sie können JSON-Daten in eine SUPER-Spalte einfügen oder aktualisieren, indem Sie die Funktion `json_parse` verwenden. Die Funktion analysiert Daten im JSON-Format und konvertiert sie in den SUPER-Datentyp, den Sie in INSERT- oder UPDATE-Anweisungen verwenden können.

Im folgenden Beispiel werden JSON-Daten in eine SUPER-Spalte eingefügt. Wenn das die Funktion `json_parse` in der Abfrage fehlt, behandelt Amazon Redshift den Wert als eine einzelne Zeichenfolge anstelle einer JSON-formatierten Zeichenfolge, die analysiert werden muss.

Wenn Sie eine SUPER-Datenspalte aktualisieren, erfordert Amazon Redshift, dass das vollständige Dokument an Spaltenwerte übergeben wird. Amazon Redshift unterstützt keine teilweisen Aktualisierungen.

```
INSERT INTO region_nations VALUES(0,
   'lar deposits. blithely final packages cajole. regular waters are final requests.
 regular accounts are according to',
   'AFRICA',
   JSON_PARSE('{"r_nations":[
      {"n_comment":" haggle. carefully final deposits detect slyly agai",
         "n_nationkey":0,
         "n_name":"ALGERIA"
      },
      {"n_comment":"ven packages wake quickly. regu",
         "n_nationkey":5,
         "n_name":"ETHIOPIA"
      },
      {"n_comment":" pending excuses haggle furiously deposits. pending, express pinto
 beans wake fluffily past t",
         "n_nationkey":14,
         "n_name":"KENYA"
```

```
      },
      {"n_comment":"rns. blithely bold courts among the closely regular packages use
 furiously bold platelets?",
         "n_nationkey":15,
         "n_name":"MOROCCO"
      },
      {"n_comment":"s. ironic, unusual asymptotes wake blithely r",
         "n_nationkey":16,
         "n_name":"MOZAMBIQUE"
      }
   ]
}'));
```

Verwenden von COPY zum Laden von SUPER-Spalten in Amazon Redshift

In den folgenden Abschnitten erfahren Sie, wie Sie den Befehl COPY verwenden können, um JSON-Daten in Amazon Redshift zu laden.

Kopieren von Daten aus JSON und Avro

Durch die Unterstützung halbstrukturierter Daten in Amazon Redshift können Sie ein JSON-Dokument laden, ohne die Attribute seiner JSON-Strukturen in mehrere Spalten aufzuteilen.

Amazon Redshift bietet zwei Methoden zur Aufnahme von JSON-Dokumenten mit COPY, selbst bei einer JSON-Struktur, die vollständig oder teilweise unbekannt ist:

1. Speichern Sie die Daten, die aus einem JSON-Dokument abgeleitet werden, in einer einzelnen SUPER-Datenspalte mit der `noshred`-Option. Diese Methode ist nützlich, wenn das Schema nicht bekannt ist oder voraussichtlich geändert wird. Daher erleichtert diese Methode das Speichern des gesamten Tupels in einer einzigen SUPER-Spalte.
2. Teilen Sie das JSON-Dokument mit der Option `auto` oder `jsonpaths` in mehrere Amazon-Redshift-Spalten auf. Attribute können Amazon-Redshift-Skalare oder SUPER-Werte sein.

Sie können diese Optionen mit den Formaten JSON oder Avro verwenden.

Die maximale Größe für ein JSON-Objekt vor dem Aufteilen ist 4 MB.

Kopieren eines JSON-Dokuments in eine einzelne SUPER-Datenspalte

Um ein JSON-Dokument in eine einzelne SUPER-Datenspalte zu kopieren, erstellen Sie eine Tabelle mit einer einzelnen SUPER-Datenspalte.

```
CREATE TABLE region_nations_noshred (rdata SUPER);
```

Kopieren Sie die Daten aus Amazon S3 in die einzelne SUPER-Datenspalte. Um die JSON-Quelldaten in eine einzelne SUPER-Datenspalte aufzunehmen, geben Sie die Option `noshred` in der Klausel FORMAT JSON an.

```
COPY region_nations_noshred FROM 's3://redshift-downloads/semistructured/tpch-nested/
data/json/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'noshred';
```

Nachdem COPY die JSON-Daten erfolgreich aufgenommen hat, verfügt Ihre Tabelle über eine `rdata`SUPER-Datenspalte, die die Daten des gesamten JSON-Objekts enthält. Die aufgenommenen Daten behalten alle Eigenschaften der JSON-Hierarchie bei. Für eine effiziente Abfrageverarbeitung werden die Blätter jedoch in skalare Amazon-Redshift-Typen konvertiert.

Verwenden Sie die folgende Abfrage, um die ursprüngliche JSON-Zeichenfolge abzurufen.

```
SELECT rdata FROM region_nations_noshred;
```

Wenn Amazon Redshift eine SUPER-Datenspalte generiert, wird sie mithilfe von JDBC als Zeichenfolge über die JSON-Serialisierung zugänglich. Weitere Informationen finden Sie unter Serialisieren komplexer verschachtelter JSON-Datentypen.

Kopieren eines JSON-Dokuments in mehrere SUPER-Datenspalten

Sie können ein JSON-Dokument in mehrere Spalten aufteilen, die entweder SUPER-Datenspalten oder Amazon-Redshift-Skalartypen sein können. Amazon Redshift verteilt verschiedene Teile des JSON-Objekts auf verschiedene Spalten.

```
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );
```

Um die Daten des vorherigen Beispiels in die Tabelle zu kopieren, geben Sie die Option AUTO in der Klausel FORMAT JSON an, um den JSON-Wert auf mehrere Spalten aufzuteilen. COPY ordnet die

JSON-Attribute der obersten Ebene mit Spaltennamen zu und ermöglicht, dass verschachtelte Werte als SUPER-Werte wie JSON-Arrays und -Objekte aufgenommen werden können.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'auto';
```

Wenn die JSON-Attributnamen gemischte Groß- und Kleinschreibung enthalten, geben Sie die Option `auto ignorecase` in der Klausel FORMAT JSON an. Weitere Informationen zur Verwendung des COPY-Befehls finden Sie unter Laden von JSON-Daten unter Verwendung der Option „auto ignorecase“.

In einigen Fällen gibt es eine Diskrepanz zwischen Spaltennamen und JSON-Attributen oder das zu ladende Attribut ist mehr als eine Ebene tief verschachtelt. Wenn ja, verwenden Sie eine `jsonpaths`-Datei, um JSON-Attribute manuell Amazon-Redshift-Spalten zuzuordnen.

```
CREATE TABLE nations
(
 regionkey smallint
 ,name varchar
 ,comment super
 ,nations super
 );
```

Angenommen, Sie möchten Daten in eine Tabelle laden, in der die Spaltennamen nicht mit den JSON-Attributen übereinstimmen. Im folgenden Beispiel ist das bei der Tabelle `nations` der Fall. Sie können eine `jsonpaths`-Datei erstellen, die die Pfade von Attributen den Tabellenspalten anhand ihrer Position im `jsonpaths`-Array zuordnet.

```
{"jsonpaths": [
       "$.r_regionkey",
       "$.r_name",
       "$.r_comment",
       "$.r_nations
    ]
}
```

Der Speicherort der `jsonpaths`-Datei wird als Argument für FORMAT JSON verwendet.

```
COPY nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 's3://redshift-downloads/semistructured/tpch-nested/data/jsonpaths/
nations_jsonpaths.json';
```

Verwenden Sie die folgende Abfrage, um auf die Tabelle zuzugreifen, die die Datenverteilung auf mehrere Spalten anzeigt. Die SUPER-Datenspalten werden im JSON-Format ausgegeben.

```
SELECT r_regionkey,r_name,r_comment,r_nations[0].n_nationkey FROM region_nations ORDER
 BY 1,2,3 LIMIT 1;
```

Jsonpaths-Dateien ordnen Felder im JSON-Dokument Tabellenspalten zu. Sie können zusätzliche Spalten, wie Verteilungs- und Sortierschlüssel, extrahieren und dabei zugleich das gesamte Dokument als SUPER-Spalte laden. Die folgende Abfrage lädt das vollständige Dokument in die Spalte „nations“. Die Spalte `name` ist der Sortierschlüssel und die Spalte `regionkey` ist der Verteilungsschlüssel.

```
CREATE TABLE nations_sorted (
    regionkey smallint,
    name varchar,
    nations super
) DISTKEY(regionkey) SORTKEY(name);
```

Der Root-jsonpath „$“ wird dem Root des Dokuments wie folgt zugeordnet:

```
{"jsonpaths": [
       "$.r_regionkey",
       "$.r_name",
       "$"
    ]
}
```

Der Speicherort der jsonpaths-Datei wird als das Argument für FORMAT JSON verwendet.

```
COPY nations_sorted FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 's3://redshift-downloads/semistructured/tpch-nested/data/jsonpaths/
nations_sorted_jsonpaths.json';
```

Kopieren von Daten aus Text und CSV

Amazon Redshift stellt SUPER-Spalten in Text- und CSV-Formaten als serialisiertes JSON dar. Eine gültige JSON-Formatierung ist erforderlich, damit SUPER-Spalten mit den korrekten Typinformationen geladen werden. Verwenden Sie Unquote für Objekte, Arrays, Zahlen, Boolesche Werte und Nullwerte. Setzen Sie Zeichenfolgenwerte in doppelte Anführungszeichen. SUPER-Spalten verwenden Standard-Escape-Zeichenregeln für Text- und CSV-Formate. Beim CSV-Format werden Trennzeichen gemäß CSV-Standard mit Escape-Zeichen versehen. Verwenden Sie bei Text die ESCAPE-Option während COPY und UNLOAD, wenn das gewählte Trennzeichen auch in einem SUPER-Feld vorkommen könnte.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/csv/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT CSV;
```

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/text/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
DELIMITER ','
ESCAPE;
```

Kopieren von Daten aus dem Spaltenformat Parquet und ORC

Wenn Ihre halbstrukturierten oder verschachtelten Daten bereits im Format Apache Parquet oder Apache ORC verfügbar sind, können Sie den Befehl COPY verwenden, um Daten in Amazon Redshift zu erfassen.

Die Amazon-Redshift-Tabellenstruktur sollte mit der Anzahl der Spalten und den Spaltendatentypen der Parquet- oder ORC-Dateien übereinstimmen. Indem Sie SERIALIZETOJSON im Befehl COPY angeben, können Sie jeden Spaltentyp in der Datei laden, der einer SUPER-Spalte in der Tabelle zugeordnet ist. Dazu gehören Struktur- und Array-Typen.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/
parquet/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT PARQUET SERIALIZETOJSON;
```

Im folgenden Beispiel wird ein ORC-Format verwendet.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/orc/
region_nation'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT ORC SERIALIZETOJSON;
```

Wenn die Attribute der Datentypen Datum oder Uhrzeit in ORC enthalten sind, konvertiert Amazon Redshift sie nach der Kodierung in SUPER zu varchar.

Entladen von halbstrukturierten Daten

Sie können Tabellen mit SUPER-Datenspalten in verschiedenen Formaten in Amazon S3 entladen.

Themen

- Entladen von semistrukturierten Daten in CSV- oder Textformaten
- Entladen von semistrukturierten Daten im Parquet-Format

Entladen von semistrukturierten Daten in CSV- oder Textformaten

Es ist möglich, Tabellen mit SUPER-Datenspalten in einem CSV- oder Textformat in Amazon S3 zu entladen. Amazon Redshift entlädt hierarchische Daten im SUPER-Datenformat im CSV- oder Textformat mit einer Kombination aus Navigations- und Unnest-Klauseln in Amazon S3. Anschließend können Sie externe Tabellen mit entladenen Daten erstellen und mit Redshift Spectrum abfragen. Informationen zur Verwendung von UNLOAD und den erforderlichen IAM-Berechtigungen finden Sie unter UNLOAD.

Bevor Sie das folgende Beispiel ausführen, füllen Sie die Tabelle region_nations mit den Prozessen in Laden von halbstrukturierten Daten in Amazon Redshift. Weitere Hinweise zu den Tabellen, die in den folgenden Beispielen verwendet werden, finden Sie unter SUPER-Beispieldatensatz.

Im folgenden Beispiel werden Daten in Amazon S3 entladen.

```
UNLOAD ('SELECT * FROM region_nations')
TO 's3://xxxxxx/'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3-Write'
DELIMITER AS '|'
GZIP
ALLOWOVERWRITE;
```

Im Gegensatz zu anderen Datentypen, bei denen eine benutzerdefinierte Zeichenfolge einen Nullwert darstellt, exportiert Amazon Redshift die SUPER-Datenspalten im JSON-Format und stellt sie als null dar, wie durch das JSON-Format bestimmt. Daher ignorieren SUPER-Datenspalten die Option NULL [AS], die in UNLOAD-Befehlen verwendet wird.

Entladen von semistrukturierten Daten im Parquet-Format

Sie können Tabellen mit SUPER-Datenspalten im Parquet-Format in Amazon S3 entladen. Amazon Redshift stellt SUPER-Spalten in Parquet als JSON-Datentyp dar. Auf diese Weise können semistrukturierte Daten in Parquet dargestellt werden. Sie können diese Spalten mit Redshift Spectrum abfragen oder sie mit dem Befehl COPY wieder in Amazon Redshift erfassen. Informationen zur Verwendung von UNLOAD und den erforderlichen IAM-Berechtigungen finden Sie unter UNLOAD.

Im folgenden Beispiel werden Daten im Parquet-Format in Amazon S3 entladen.

```
UNLOAD ('SELECT * FROM region_nations')
TO 's3://xxxxxx/'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3-Write'
FORMAT PARQUET;
```

Abfragen von halbstrukturierten Daten

Amazon Redshift verwendet die PartiQL-Sprache, um SQL-kompatiblen Zugriff auf relationale, halbstrukturierte und verschachtelte Daten zu bieten.

PartiQL arbeitet mit dynamischen Typen. Dies ermöglicht eine intuitive Filterung, Verknüpfung und Aggregation für die Kombination strukturierter, semistrukturierter und verschachtelter Datensätze. Die PartiQL-Syntax verwendet Punktschreibweise und Array-Subscript für die Pfadnavigation beim Zugriff auf verschachtelte Daten. Es ermöglicht auch die FROM-Klauselelemente über Arrays zu iterieren und für Unnest-Operationen zu verwenden. Nachfolgend werden die verschiedenen Abfragemuster beschrieben, die die Verwendung des SUPER-Datentyps mit Pfad- und Array-Navigation, Aufheben der Verschachtelung, Entpivotieren und Joins kombinieren.

Weitere Hinweise zu den Tabellen, die in den folgenden Beispielen verwendet werden, finden Sie unter SUPER-Beispieldatensatz.

Navigation

Amazon Redshift verwendet PartiQL, um die Navigation in Arrays und Strukturen mithilfe der [...]-Klammer bzw. Punktschreibweise zu ermöglichen. Darüber hinaus können Sie die Navigation mithilfe von Punktschreibweise und Arrays mithilfe der Klammernotation in Strukturen mischen. Im folgenden Beispiel wird angenommen, dass die SUPER-Datenspalte `c_orders` ein Array mit einer Struktur ist und ein Attribut `o_orderkey` lautet.

Um Daten in der Spalte `customer_orders_lineitem` zu erfassen, führen Sie den folgenden Befehl aus. Ersetzen Sie die IAM-Rolle durch Ihre eigenen Anmeldeinformationen.

```
COPY customer_orders_lineitem FROM 's3://redshift-downloads/semistructured/tpch-nested/
data/json/customer_orders_lineitem'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'auto';

SELECT c_orders[0].o_orderkey FROM customer_orders_lineitem;
```

Amazon Redshift verwendet auch einen Tabellenalias als Präfix für die Notation. Das folgende Beispiel zeigt dieselbe Abfrage wie im vorherigen Beispiel.

```
SELECT cust.c_orders[0].o_orderkey FROM customer_orders_lineitem AS cust;
```

Sie können die Punkt- und Klammernotationen in allen Arten von Abfragen verwenden, z. B. Filtern, Verknüpfen und Aggregation. Sie können diese Notationen in einer Abfrage verwenden, in der normalerweise Spaltenverweise vorhanden sind. Im folgenden Beispiel wird eine SELECT-Anweisung verwendet, die Ergebnisse filtert.

```
SELECT count(*) FROM customer_orders_lineitem WHERE c_orders[0]. o_orderkey IS NOT
 NULL;
```

Im folgenden Beispiel wird die Klammer- und Punktnavigation in den Klauseln GROUP BY und ORDER BY verwendet.

```
SELECT c_orders[0].o_orderdate,
       c_orders[0].o_orderstatus,
       count(*)
FROM customer_orders_lineitem
```

```
WHERE c_orders[0].o_orderkey IS NOT NULL
GROUP BY c_orders[0].o_orderstatus,
         c_orders[0].o_orderdate
ORDER BY c_orders[0].o_orderdate;
```

Aufheben der Verschachtelung von Abfragen

Zur Aufhebung der Verschachtelung von Abfragen verwendet Amazon Redshift die PartiQL-Syntax, um über SUPER-Arrays zu iterieren. Dazu navigiert es im Array mithilfe der FROM-Klausel einer Abfrage. Das folgende Beispiel nutzt das vorherige Beispiel und iteriert über die Attributwerte für `c_orders`.

```
SELECT c.*, o FROM customer_orders_lineitem c, c.c_orders o;
```

Die Unnesting-Syntax ist eine Erweiterung der FROM-Klausel. In Standard-SQL bedeutet die FROM-Klausel `x (AS) y`, dass `y` über jedes Tupel in Beziehung `x` iteriert. In diesem Fall bezieht sich `x` auf eine Beziehung und `y` bezieht sich auf einen Alias für Beziehung `x`. Entsprechend bedeutet die PartiQL-Syntax zum Aufheben der Verschachtelung mithilfe des FROM-Klauselelements `x (AS) y`, dass `y` über jeden (SUPER)-Wert in (SUPER)-Array-Ausdruck `x` iteriert. In diesem Fall ist `x` ein SUPER-Ausdruck und `y` ist ein Alias für `x`.

Der linke Operand kann auch die Punkt- und Klammernotation für die reguläre Navigation verwenden. Im vorherigen Beispiel ist `customer_orders_lineitem c` die Iteration über die Basistabelle `customer_order_lineitem` und `c.c_orders o` ist die Iteration über das Array `c.c_orders`. Um über das Attribut `o_lineitems` zu iterieren, also ein Array innerhalb eines Arrays, fügen Sie mehrere Klauseln hinzu.

```
SELECT c.*, o, l FROM customer_orders_lineitem c, c.c_orders o, o.o_lineitems l;
```

Amazon Redshift unterstützt auch einen Array-Index, wenn mit dem AT-Schlüsselwort über das Array iteriert wird. Die Klausel `x AS y AT z` iteriert über Array `x` und generiert das Feld `z,`, das der Array-Index ist. Das folgende Beispiel zeigt die Funktionsweise eines Array-Index.

```
SELECT c_name,
       orders.o_orderkey AS orderkey,
       index AS orderkey_index
FROM customer_orders_lineitem c, c.c_orders AS orders AT index
ORDER BY orderkey_index;
```

```
c_name             | orderkey | orderkey_index
-------------------+----------+----------------
Customer#000008251 | 3020007  |        0
Customer#000009452 | 4043971  |        0
  (2 rows)
```

Das folgende Beispiel iteriert über ein skalares Array.

```
CREATE TABLE bar AS SELECT json_parse('{"scalar_array": [1, 2.3, 45000000]}') AS data;

SELECT index, element FROM bar AS b, b.data.scalar_array AS element AT index;

 index | element
-------+----------
     0 | 1
     1 | 2.3
     2 | 45000000
(3 rows)
```

Im folgenden Beispiel wird über ein Array mit mehreren Ebenen iteriert. Das Beispiel nutzt mehrere Klauseln zum Aufheben der Verschachtelung, um in die innersten Arrays zu iterieren. Das AS-Array `f.multi_level_array` iteriert über `multi_level_array`. Das Array-AS-Element ist die Iteration über die Arrays innerhalb von `multi_level_array`.

```
CREATE TABLE foo AS SELECT json_parse('[[1.1, 1.2], [2.1, 2.2], [3.1, 3.2]]') AS
 multi_level_array;

SELECT array, element FROM foo AS f, f.multi_level_array AS array, array AS element;

   array   | element
-----------+---------
 [1.1,1.2] | 1.1
 [1.1,1.2] | 1.2
 [2.1,2.2] | 2.1
 [2.1,2.2] | 2.2
 [3.1,3.2] | 3.1
 [3.1,3.2] | 3.2
(6 rows)
```

Weitere Informationen über die FROM-Klausel finden Sie unter FROM-Klausel.

Entpivotieren von Objekten

Um Objekte zu entpivotieren, verwendet Amazon Redshift die PartiQL-Syntax, um über SUPER-Objekte zu iterieren. Dazu verwendet es die FROM-Klausel einer Abfrage mit dem Schlüsselwort UNPIVOT. In diesem Fall ist der Ausdruck das Objekt. `c.c_orders[0]` Die Beispielabfrage iteriert über jedes vom Objekt zurückgegebene Attribut.

```
SELECT attr as attribute_name, json_typeof(val) as value_type
FROM customer_orders_lineitem c, UNPIVOT c.c_orders[0] AS val AT attr
WHERE c_custkey = 9451;

 attribute_name  | value_type
-----------------+------------
 o_orderstatus   | string
 o_clerk         | string
 o_lineitems     | array
 o_orderdate     | string
 o_shippriority  | number
 o_totalprice    | number
 o_orderkey      | number
 o_comment       | string
 o_orderpriority | string
(9 rows)
```

Wie beim Aufheben der Verschachtelung ist die Syntax zum Entpivotieren eine Erweiterung der FROM-Klausel. Der Unterschied ist, dass die Syntax zum Entpivotieren das Schlüsselwort UNPIVOT verwendet, um anzuzeigen, dass es über ein Objekt anstelle eines Arrays iteriert. Es verwendet das AS `value_alias` zur Iteration über alle Werte innerhalb eines Objekts und das AT `attribute_alias` zum Iterieren über alle Attribute. Betrachten Sie das folgende Syntaxfragment:

```
UNPIVOT expression AS value_alias [ AT attribute_alias ]
```

Amazon Redshift unterstützt die Verwendung von Object Unpivoting und Array-Unnesting in einer einzigen FROM-Klausel wie folgt:

```
SELECT attr as attribute_name, val as object_value
FROM customer_orders_lineitem c, c.c_orders AS o, UNPIVOT o AS val AT attr
WHERE c_custkey = 9451;
```

Wenn Sie das Entpivotieren von Objekten verwenden, unterstützt Amazon Redshift kein korreliertes Entpivotieren. Angenommen den Fall, es gibt mehrere Beispiele für Entpivotieren in verschiedenen Abfrageebenen und das innere Entpivotieren verweist auf das äußere. Amazon Redshift unterstützt diese Art von mehrfachem Entpivotieren nicht.

Weitere Informationen über die FROM-Klausel finden Sie unter FROM-Klausel. Beispiele, die Abfragen für strukturierte Daten mit PIVOT und UNPIVOT veranschaulichen, finden Sie unter Beispiele für PIVOT und UNPIVOT.

Dynamische Typisierung

Die dynamische Eingabe erfordert keine explizite Umwandlung von Daten, die aus den Punkt- und Klammerpfaden extrahiert werden. Amazon Redshift verwendet die dynamische Typisierung, um schemalose SUPER-Daten zu verarbeiten, ohne dass die Datentypen deklariert werden müssen, bevor Sie sie in Ihrer Abfrage verwenden. Bei der dynamischen Typisierung werden die Ergebnisse der Navigation in SUPER-Datenspalten verwendet, ohne sie explizit in Amazon-Redshift-Typen umwandeln zu müssen. Dynamische Typisierung ist am nützlichsten in Joins und GROUP-BY-Klauseln. Im folgenden Beispiel wird eine SELECT-Anweisung verwendet, die keine explizite Umwandlung der Punkt- und Klammerausdrücke in die üblichen Amazon-Redshift-Typen erfordert. Informationen zur Typkompatibilität und Konvertierung finden Sie unter Kompatibilität von Typen und Umwandlung zwischen Typen.

```
SELECT c_orders[0].o_orderkey
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderstatus = 'P';
```

Das Gleichheitszeichen in dieser Abfrage wird als `true` evaluiert, wenn c_orders[0].o_orderstatus die Zeichenfolge ‘P’. In allen anderen Fällen wird das Gleichheitszeichen als `false` evaluiert, einschließlich der Fälle, in denen die Argumente der Gleichheit unterschiedliche Typen sind.

Dynamische und statische Typisierung

Ohne dynamische Typisierung können Sie nicht bestimmen, ob c_orders[0].o_orderstatus eine Zeichenfolge, eine Ganzzahl oder eine Struktur ist. Sie können nur feststellen, dass c_orders[0].o_orderstatus ein SUPER-Datentyp ist, bei dem es sich um einen Amazon-Redshift-Skalar, ein Array oder eine Struktur handeln kann. Der statische Typ von c_orders[0].o_orderstatus ist ein SUPER-Datentyp. Üblicherweise ist ein Typ implizit ein statischer Typ in SQL.

Amazon Redshift verwendet dynamische Typisierung für die Verarbeitung von schemalosen Daten. Wenn die Abfrage die Daten auswertet, erweist sich c_orders[0].o_orderstatus als ein bestimmter Typ. Beispielsweise kann die Auswertung von c_orders[0].o_orderstatus auf dem ersten Datensatz von customer_orders_lineitem zu einer Ganzzahl führen. Die Auswertung des zweiten Datensatzes kann zu einer Zeichenfolge führen. Dies sind die dynamischen Typen des Ausdrucks.

Wenn Sie einen SQL-Operator oder eine SQL-Funktion mit Punkt- und Klammerausdrücken verwenden, die dynamische Typen haben, erzeugt Amazon Redshift ähnliche Ergebnisse wie bei der Verwendung des Standard-SQL-Operators bzw. der Standard-SQL-Funktion mit den jeweiligen statischen Typen. Wenn in diesem Beispiel der dynamische Typ des Pfadausdrucks eine Zeichenfolge ist, ist der Vergleich mit der Zeichenfolge „P“ sinnvoll. Immer wenn der dynamische Typ von c_orders[0].o_orderstatus ein anderer Datentyp außer eine Zeichenfolge ist, gibt die Gleichheit false zurück. Andere Funktionen geben null zurück, wenn falsch eingegebene Argumente verwendet werden.

Im folgenden Beispiel wird die vorherige Abfrage mit statischer Eingabe geschrieben:

```
SELECT c_custkey
FROM customer_orders_lineitem
WHERE CASE WHEN JSON_TYPEOF(c_orders[0].o_orderstatus) = 'string'
           THEN c_orders[0].o_orderstatus::VARCHAR = 'P'
           ELSE FALSE END;
```

Beachten Sie die folgende Unterscheidung zwischen Gleichheitsprädikaten und Vergleichsprädikaten. Wenn Sie im vorherigen Beispiel das Gleichheitsprädikat durch ein Prädikat ersetzen, ergibt die less-than-or-equal Semantik Null statt False.

```
SELECT c_orders[0]. o_orderkey
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderstatus <= 'P';
```

Wenn in diesem Beispiel c_orders[0].o_orderstatus eine Zeichenfolge ist, gibt Amazon Redshift true zurück, wenn sie alphabetisch gleich oder kleiner als „P“ ist. Amazon Redshift gibt false zurück, wenn sie alphabetisch größer als „P“ ist. Wenn c_orders[0].o_orderstatus jedoch keine Zeichenfolge ist, gibt Amazon Redshift null zurück, da Amazon Redshift Werte verschiedener Typen nicht vergleichen kann, wie in der folgenden Abfrage dargestellt:

```
SELECT c_custkey
```

```
FROM customer_orders_lineitem
WHERE CASE WHEN JSON_TYPEOF(c_orders[0].o_orderstatus) = 'string'
           THEN c_orders[0].o_orderstatus::VARCHAR <= 'P'
           ELSE NULL END;
```

Dynamische Typisierung schließt keine Vergleiche von Typen aus, die minimal vergleichbar sind. Beispielsweise können Sie Amazon-Redshift-Skalartypen von sowohl CHAR als auch VARCHAR in SUPER konvertieren. Sie sind als Zeichenfolgen vergleichbar, einschließlich des Ignorierens nachstehender Leerzeichen, ähnlich wie bei CHAR- und VARCHAR-Typen von Amazon Redshift. In ähnlicher Weise sind Ganzzahlen, Dezimalzahlen und Gleitkommawerte als SUPER-Werte vergleichbar. Speziell für Dezimalspalten kann jeder Wert auch einen anderen Maßstab haben. In Amazon Redshift gelten sie weiterhin als dynamische Typen.

Amazon Redshift unterstützt auch die Gleichheit von Objekten und Arrays, die als deep equal ausgewertet werden, z. B. die ausführliche Auswertung von Objekten oder Arrays und der Vergleich aller Attribute. Verwenden Sie deep equal mit Vorsicht, da die Durchführung zeitaufwendig sein kann.

Verwenden der dynamischen Typisierung für Joins

Bei Joins passt die dynamische Typisierung automatisch Werte mit unterschiedlichen dynamischen Typen an, ohne eine lange CASE-WHEN-Analyse durchzuführen, um herauszufinden, welche Datentypen möglicherweise angezeigt werden. Nehmen wir beispielsweise an, dass Ihre Organisation das Format geändert hat, das sie für Teileschlüssel verwendet hat.

Die ursprünglichen ganzzahligen Teilschlüssel werden durch Zeichenfolgen-Teilschlüssel ersetzt, wie „A55“, und später wieder durch Array-Teilschlüssel ersetzt, wie ['X', 10], die eine Zeichenfolge und eine Zahl kombinieren. Amazon Redshift muss keine langwierige Fallanalyse zu Teileschlüsseln durchführen und kann Joins verwenden, wie im folgenden Beispiel gezeigt.

```
SELECT c.c_name
    ,l.l_extendedprice
    ,l.l_discount
FROM customer_orders_lineitem c
    ,c.c_orders o
    ,o.o_lineitems l
    ,supplier_partsupp s
    ,s.s_partsupps ps
WHERE l.l_partkey = ps.ps_partkey
AND c.c_nationkey = s.s_nationkey
ORDER BY c.c_name;
```

Das folgende Beispiel zeigt, wie komplex und ineffizient dieselbe Abfrage ohne dynamische Typisierung sein kann:

```
SELECT c.c_name
    ,l.l_extendedprice
    ,l.l_discount
FROM customer_orders_lineitem c
    ,c.c_orders o
    ,o.o_lineitems l
    ,supplier_partsupp s
    ,s.s_partsupps ps
WHERE CASE WHEN IS_INTEGER(l.l_partkey) AND IS_INTEGER(ps.ps_partkey)
           THEN l.l_partkey::integer = ps.ps_partkey::integer
           WHEN IS_VARCHAR(l.l_partkey) AND IS_VARCHAR(ps.ps_partkey)
           THEN l.l_partkey::varchar = ps.ps_partkey::varchar
           WHEN IS_ARRAY(l.l_partkey) AND IS_ARRAY(ps.ps_partkey)
                AND IS_VARCHAR(l.l_partkey[0]) AND IS_VARCHAR(ps.ps_partkey[0])
                AND IS_INTEGER(l.l_partkey[1]) AND IS_INTEGER(ps.ps_partkey[1])
           THEN l.l_partkey[0]::varchar = ps.ps_partkey[0]::varchar
                AND l.l_partkey[1]::integer = ps.ps_partkey[1]::integer
           ELSE FALSE END
AND c.c_nationkey = s.s_nationkey
ORDER BY c.c_name;
```

Lax-Semantik

Standardmäßig geben Navigationsvorgänge für SUPER-Werte null zurück, anstatt einen Fehler zurückzugeben, wenn die Navigation ungültig ist. Die Objektnavigation ist ungültig, wenn der SUPER-Wert kein Objekt ist oder wenn der SUPER-Wert ein Objekt ist, aber nicht den Attributnamen enthält, der in der Abfrage verwendet wird. Die folgende Abfrage greift beispielsweise auf einen ungültigen Attributnamen in der SUPER-Datenspalte cdata zu:

```
SELECT c.c_orders.something FROM customer_orders_lineitem c;
```

Die Array-Navigation gibt null zurück, wenn der SUPER-Wert kein Array ist oder der Array-Index außerhalb der Grenzen liegt. Die folgende Abfrage gibt null zurück, da c_orders[1][1] außerhalb der Grenzen liegt.

```
SELECT c.c_orders[1][1] FROM customer_orders_lineitem c;
```

Lax-Semantik ist besonders nützlich, wenn dynamische Typisierung verwendet wird, um einen SUPER Wert zu konvertieren. Wenn ein SUPER Wert in den falschen Typ umgewandelt wird, wird null anstelle eines Fehlers zurückgegeben, wenn die Umwandlung ungültig ist. Die folgende Abfrage gibt beispielsweise null zurück, da sie den Zeichenfolgenwert 'Good' des Objektattributs o_orderstatus nicht in INTEGER umwandeln kann. Amazon Redshift gibt einen Fehler für eine Umwandlung von VARCHAR zu INTEGER zurück, aber nicht für eine SUPER-Umwandlung.

```
SELECT c.c_orders.o_orderstatus::integer FROM customer_orders_lineitem c;
```

Arten der Introspektion

SUPER-Datenspalten unterstützen Inspektionsfunktionen, die den dynamischen Typ und andere Typinformationen über den SUPER Wert zurückgeben. Das gängigste Beispiel ist die Skalarfunktion JSON_TYPEOF, die einen VARCHAR mit den Werten „boolean“, „number“, „string“, „object“, „array“ oder „null“ zurückgibt, abhängig vom dynamischen Typ des SUPER-Wertes. Amazon Redshift unterstützt die folgenden booleschen Funktionen für SUPER-Datenspalten:

- DECIMAL_PRECISION
- DECIMAL_SCALE
- IS_ARRAY
- IS_BIGINT
- IS_CHAR
- IS_DECIMAL
- IS_FLOAT
- IS_INTEGER
- IS_OBJECT
- IS_SCALAR
- IS_SMALLINT
- IS_VARCHAR
- JSON_TYPEOF

Alle diese Funktionen geben false zurück, wenn der Eingabewert null ist. IS_SCALAR, IS_OBJECT und IS_ARRAY schließen sich gegenseitig aus und decken alle möglichen Werte mit Ausnahme von null ab.

Um die den Daten entsprechenden Typen abzuleiten, verwendet Amazon Redshift die Funktion JSON_TYPEOF, die den Typ (die oberste Ebene) des SUPER-Werts zurückgibt, wie im folgenden Beispiel gezeigt:

```
SELECT JSON_TYPEOF(r_nations) FROM region_nations;
 json_typeof
 -------------
 array
(1 row)
```

```
SELECT JSON_TYPEOF(r_nations[0].n_nationkey) FROM region_nations;
 json_typeof
 -------------
 number
```

Amazon Redshift sieht dies als eine einzelne lange Zeichenfolge, ähnlich dem Einfügen dieses Werts in eine VARCHAR-Spalte anstelle eines SUPER. Da die Spalte SUPER ist, ist die einzelne Zeichenfolge immer noch ein gültiger SUPER-Wert und der Unterschied wird in JSON_TYPEOF notiert:

```
SELECT IS_VARCHAR(r_nations[0].n_name) FROM region_nations;
 is_varchar
-------------
 true
(1 row)
```

```
SELECT r_nations[4].n_name FROM region_nations
WHERE CASE WHEN IS_INTEGER(r_nations[4].n_nationkey)
           THEN r_nations[4].n_nationkey::INTEGER = 15
           ELSE false END;
```

Order by (Sortieren nach)

Amazon Redshift definiert keine SUPER-Vergleiche zwischen Werten mit verschiedenen dynamischen Typen. Ein SUPER-Wert, der eine Zeichenfolge ist, ist weder kleiner noch größer als ein SUPER-Wert, der eine Zahl ist. Um ORDER-BY-Klauseln mit SUPER-Spalten zu verwenden, definiert Amazon Redshift eine Gesamtsortierung zwischen verschiedenen Typen, die beachtet werden müssen, wenn Amazon Redshift SUPER-Werte mithilfe von ORDER-BY-Klauseln anordnet.

Die Reihenfolge zwischen dynamischen Typen ist boolesch, Zahl, Zeichenfolge, Array, Objekt. Das folgende Beispiel zeigt die Reihenfolge der verschiedenen Typen:

```
INSERT INTO region_nations VALUES
(100,'name1','comment1','AWS'),
(200,'name2','comment2',1),
(300,'name3','comment3',ARRAY(1, 'abc', null)),
(400,'name4','comment4',-2.5),
(500,'name5','comment5','Amazon');

SELECT r_nations FROM region_nations order by r_nations;

r_nations
----------------
 -2.5
 1
 "Amazon"
 "AWS"
 [1,"abc",null]
(5 rows)
```

Weitere Informationen über die ORDER-BY-Klausel finden Sie unter ORDER BY-Klausel.

Operatoren und Funktionen

Amazon Redshift bietet die folgende Funktionsunterstützung von SUPER-Operatoren und -Funktionen.

Arithmetische Operatoren

SUPER-Werte unterstützen alle grundlegenden arithmetischen Operatoren +, -, *, /, % mit dynamischer Typisierung. Der resultierende Typ der Operation bleibt SUPER. Für alle Operatoren, außer für den binären Operator +, müssen die Eingabeoperanden Zahlen sein. Andernfalls gibt Amazon Redshift null zurück. Die Unterscheidung zwischen Dezimalzahlen und Gleitkommawerten wird beibehalten, wenn Amazon Redshift diese Operatoren ausführt und sich der dynamische Typ nicht ändert. Die Dezimalskalierung ändert sich jedoch, wenn Sie Multiplikationen und Teilungen verwenden. Arithmetische Überläufe verursachen immer noch Abfragefehler, sie werden nicht in null geändert. Der binäre Operator + führt eine Addition aus, wenn die Eingaben Zahlen sind, oder eine Verkettung, wenn die Eingaben Zeichenfolgen sind. Wenn ein Operand eine Zeichenfolge ist und der

andere Operand eine Zahl ist, ist das Ergebnis null Die unären Präfix-Operatoren + und - geben null zurück, wenn der SUPER-Wert keine Zahl ist, wie im folgenden Beispiel gezeigt:

```
SELECT (c_orders[0]. o_orderkey + 0.5) * c_orders[0]. o_orderkey / 10 AS math FROM
 customer_orders_lineitem;
            math
----------------------------
 1757958232200.1500
(1 row)
```

Durch die dynamische Typisierung können Dezimalwerte in SUPER unterschiedliche Maßstäbe haben. Amazon Redshift behandelt Dezimalwerte, als ob es sich um unterschiedliche statische Typen handelt und erlaubt alle mathematischen Operationen. Amazon Redshift berechnet die resultierende Skalierung dynamisch basierend auf den Skalierungen der Operanden. Wenn einer der Operanden eine Gleitkommazahl ist, stuft Amazon Redshift den anderen Operanden zu einer Gleitkommazahl herauf und generiert das Ergebnis als Gleitkommazahl.

Arithmetische Funktionen

Amazon Redshift unterstützt die folgenden arithmetischen Funktionen für SUPER-Spalten. Sie geben null zurück, wenn die Eingabe keine Zahl ist:

- FLOOR. Weitere Informationen finden Sie unter Die Funktion FLOOR.
- CEIL und CEILING. Weitere Informationen finden Sie unter Die Funktion CEILING (oder CEIL).
- ROUND. Weitere Informationen finden Sie unter Die Funktion ROUND.
- TRUNC. Weitere Informationen finden Sie unter Die Funktion TRUNC.
- ABS. Weitere Informationen finden Sie unter Funktion ABS.

Im folgenden Beispiel werden arithmetische Funktionen zum Abfragen von Daten verwendet:

```
SELECT x, FLOOR(x), CEIL(x), ROUND(x)
FROM (
    SELECT (c_orders[0]. o_orderkey + 0.5) * c_orders[0].o_orderkey / 10 AS x
    FROM customer_orders_lineitem
    );

         x          |     floor     |     ceil      |     round
--------------------+---------------+---------------+---------------
```

```
 1389636795898.0500  | 1389636795898  | 1389636795899  | 1389636795898
```

Die ABS-Funktion behält die Skala der Eingabe-Dezimalzahl während FLOOR, CEIL. ROUND eliminiert die Skala der Eingabe-Dezimalzahl.

Array-Funktionen

Amazon Redshift unterstützt die folgenden Array-Kompositions- und Dienstprogrammfunktionen: array, array_concat, subarray, array_flatten, get_array_length und split_to_array.

Sie können SUPER-Arrays aus Werten in Amazon-Redshift-Datentypen mithilfe der ARRAY-Funktion erstellen, einschließlich anderer SUPER-Werte. Im folgenden Beispiel wird die variadische Funktion ARRAY verwendet:

```
SELECT ARRAY(1, c.c_custkey, NULL, c.c_name, 'abc') FROM customer_orders_lineitem c;
                              array
 ------------------------------------------------------
[1,8401,null,""Customer#000008401"",""abc""]
[1,9452,null,""Customer#000009452"",""abc""]
[1,9451,null,""Customer#000009451"",""abc""]
[1,8251,null,""Customer#000008251"",""abc""]
[1,5851,null,""Customer#000005851"",""abc""]
(5 rows)
```

Im folgenden Beispiel wird die Array-Verkettung mit der Funktion ARRAY_CONCAT verwendet:

```
SELECT ARRAY_CONCAT(JSON_PARSE('[10001,10002]'),JSON_PARSE('[10003,10004]'));

             array_concat
-----------------------------------
 [10001,10002,10003,10004]
(1 row)
```

Das folgende Beispiel verwendet Array-Manipulation mit der SUBARRAY-Funktion, die eine Teilmenge des Eingabe-Arrays zurückgibt.

```
SELECT SUBARRAY(ARRAY('a', 'b', 'c', 'd', 'e', 'f'), 2, 3);

   subarray
---------------
 ["c","d","e"]
```

```
(1 row))
```

Im folgenden Beispiel werden mehrere Ebenen von Arrays mit ARRAY_FLATTEN zu einem einzelnen Array zusammengeführt:

```
SELECT x, ARRAY_FLATTEN(x) FROM (SELECT ARRAY(1, ARRAY(2, ARRAY(3, ARRAY()))) AS x);

     x           | array_flatten
 ----------------+---------------
 [1,[2,[3,[]]]]  | [1,2,3]
(1 row)
```

Array-Funktionen ARRAY_CONCAT und ARRAY_FLATTEN verwenden dynamische Typisierungsregeln. Sie geben eine Null anstelle eines Fehlers zurück, wenn die Eingabe kein Array ist. Die Funktion GET_ARRAY_LENGTH gibt die Länge eines an ein bestimmtes Objekt übergebenen SUPER-Arrays oder eines Array-Pfads an.

```
SELECT c_name
FROM customer_orders_lineitem
WHERE GET_ARRAY_LENGTH(c_orders) = (
    SELECT MAX(GET_ARRAY_LENGTH(c_orders))
    FROM customer_orders_lineitem
    );
```

Im folgenden Beispiel wird eine Zeichenfolge mithilfe von SPLIT_TO_ARRAY in ein Array von Zeichenfolgen aufgeteilt. Die Funktion verwendet ein Trennzeichen als optionalen Parameter. Wenn kein Trennzeichen fehlt, ist der Standardwert ein Komma.

```
SELECT SPLIT_TO_ARRAY('12|345|6789', '|');

   split_to_array
---------------------
 ["12","345","6789"]
(1 row)
```

SUPER-Konfigurationen

Beachten Sie die folgenden Überlegungen zu SUPER-Konfigurationen, wenn Sie den SUPER-Datentyp von Amazon Redshift und PartiQL verwenden.

Lax und strenge Modi für SUPER

Wenn Sie SUPER-Daten abfragen, stimmt der Pfadausdruck möglicherweise nicht mit der tatsächlichen SUPER-Datenstruktur überein. Wenn Sie versuchen, auf ein nicht vorhandenes Mitglied eines Objekts oder Elements eines Arrays zuzugreifen, gibt Amazon Redshift einen NULL-Wert zurück, wenn Ihre Abfrage im Standard-Lax-Modus. Wenn Sie Ihre Abfrage im strikten Modus ausführen, gibt Amazon Redshift einen Fehler zurück. Die folgenden Sitzungsparameter können eingestellt werden, um den Lax-Modus ein- oder auszuschalten.

Im folgenden Beispiel werden Sitzungsparameter verwendet, um den Lax-Modus zu aktivieren.

```
SET navigate_super_null_on_error=ON;  --default lax mode for navigation

SET cast_super_null_on_error=ON;  --default lax mode for casting

SET parse_super_null_on_error=OFF;  --default strict mode for ingestion
```

Zugriff auf JSON-Felder mit Feldnamen oder Attributen in Großschreibung und gemischter Groß-/Kleinschreibung

Wenn Ihre JSON-Attributnamen Groß- oder gemischte Groß-/Kleinschreibung aufweisen, müssen Sie in der Lage sein, unter Beachtung der Groß- und Kleinschreibung in SUPER-Typstrukturen zu navigieren. Dazu können Sie `enable_case_sensitive_identifier` auf TRUE setzen und die Attributnamen in Groß- und gemischter Groß-/Kleinschreibung in doppelte Anführungszeichen setzen. Sie können auch `enable_case_sensitive_super_attribute` auf TRUE setzen. In diesem Fall können Sie in Ihren Abfragen Attributnamen in Großschreibung und gemischter Groß-/Kleinschreibung verwenden, ohne sie in doppelte Anführungszeichen zu setzen.

Das folgende Beispiel zeigt, wie Sie `enable_case_sensitive_identifier` festlegen, um Daten abzufragen.

```
SET enable_case_sensitive_identifier to TRUE;

-- Accessing JSON attribute names with uppercase and mixedcase names
SELECT json_table.data."ITEMS"."Name",
       json_table.data."price"
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;
```

```
 Name | price
------+-------
 "TV" | 345
(1 row)

RESET enable_case_sensitive_identifier;

-- After resetting the above configuration, the following query accessing JSON
 attribute names with uppercase and mixedcase names should return null (if in lax
 mode).
SELECT json_table.data."ITEMS"."Name",
       json_table.data."price"
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
------+-------
      | 345
(1 row)
```

Das folgende Beispiel zeigt, wie Sie `enable_case_sensitive_super_attribute` festlegen, um Daten abzufragen.

```
SET enable_case_sensitive_super_attribute to TRUE;
 -- Accessing JSON attribute names with uppercase and mixedcase names

SELECT json_table.data.ITEMS.Name,
       json_table.data.price
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
 -----+-------
 "TV" | 345
(1 row)

 RESET enable_case_sensitive_super_attribute;

 -- After resetting enable_case_sensitive_super_attribute, the query now returns NULL
 for ITEMS.Name (if in lax mode).

SELECT json_table.data.ITEMS.Name,
       json_table.data.price
```

```
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
 -----+-------
      | 345
(1 row)
```

Parsing-Optionen für SUPER

Wenn Sie die Funktion JSON_PARSE zum Parsing von JSON-Zeichenfolgen in SUPER-Werte verwenden, gelten bestimmte Einschränkungen:

- Derselbe Attributname kann nicht in demselben Objekt vorkommen, aber in einem verschachtelten Objekt verwendet werden. Die Konfigurationsoption `json_parse_dedup_attributes` bietet JSON_PARSE die Möglichkeit, nur das letzte Auftreten doppelter Attribute beizubehalten, anstatt einen Fehler zurückzugeben.
- Zeichenfolgenwerte dürfen die maximale Varchar-Größe des Systems von 65535 Byte nicht überschreiten. Die Konfigurationsoption `json_parse_truncate_strings` bietet JSON_PARSE() die Möglichkeit, Zeichenfolgen, die länger als dieser Grenzwert sind, automatisch zu verkürzen, ohne einen Fehler zurückzugeben. Dieses Verhalten wirkt sich nur auf Zeichenfolgenwerte und nicht auf Attributnamen aus.

Weitere Informationen zur Funktion JSON_PARSE finden Sie unter Funktion JSON_PARSE.

Im folgenden Beispiel wird gezeigt, wie Sie für die Konfigurationsoption `json_parse_dedup_attributes` das Standardverhalten, also die Rückgabe eines Fehlers bei doppelten Attributen, festlegen.

```
SET json_parse_dedup_attributes=OFF;  --default behavior of returning error instead of
 de-duplicating attributes
```

Im folgenden Beispiel wird gezeigt, wie Sie für die Konfigurationsoption `json_parse_truncate_strings` das Standardverhalten, also die Rückgabe eines Fehlers bei Zeichenfolgen, die diesen Grenzwert überschreiten, festlegen.

```
SET json_parse_truncate_strings=OFF;  --default behavior of returning error instead of
 truncating strings
```

Einschränkungen

Beachten Sie bei der Verwendung des Datentyps SUPER die folgenden Einschränkungen:

- Sie können SUPER-Spalten weder als Verteilungs- noch als Sortierschlüssel definieren.
- Ein einzelnes SUPER-Objekt kann bis zu 16 MB an Daten enthalten.
- Ein einzelner Wert innerhalb eines SUPER-Objekts ist auf die maximale Länge des entsprechenden Amazon-Redshift-Typs begrenzt. Beispielsweise ist ein einzelner, in SUPER geladener Zeichenfolgenwert auf die maximale VARCHAR-Länge von 65 535 Bytes beschränkt.
- Sie können keine Teilaktualisierungs- oder Transformationsvorgänge für SUPER-Spalten durchführen.
- Sie können den SUPER-Datentyp und seinen Alias nicht in Right Joins oder Full Outer Joins verwenden.
- Der SUPER Datentyp unterstützt XML nicht als eingehendes oder ausgehendes Serialisierungsformat.
- In der FROM-Klausel einer Unterabfrage (die korreliert ist oder nicht), die eine Tabellenvariable zum Aufheben der Verschachtelung referenziert, kann die Abfrage nur auf ihre übergeordnete Tabelle und nicht auf andere Tabellen verweisen.
- Einschränkungen für die Umwandlung

  SUPER-Werte können in andere Datentypen umgewandelt werden und umgekehrt, mit folgenden Ausnahmen:

  - Amazon Redshift unterscheidet keine Ganzzahlen und Dezimalstellen der Skala 0.
  - Wenn die Skalierung nicht Null ist, hat der SUPER-Datentyp dasselbe Verhalten wie andere Amazon-Redshift-Datentypen, mit der Ausnahme, dass Amazon Redshift Super-bezogene Fehler in null konvertiert, wie im folgenden Beispiel gezeigt.

```
SELECT 5::bool;
 bool
-------
 True
(1 row)

SELECT 5::decimal::bool;
ERROR:  cannot cast type numeric to boolean

SELECT 5::super::bool;
```

```
 bool
-------
 True
(1 row)

SELECT 5.0::bool;
ERROR:  cannot cast type numeric to boolean

SELECT 5.0::super::bool;
 bool
-------
(1 row)
```

- Amazon Redshift wandelt die Datums- und Uhrzeittypen nicht in den SUPER-Datentyp um. Amazon Redshift kann nur Datums- und Uhrzeitdatentypen vom SUPER-Datentyp übertragen, wie im folgenden Beispiel gezeigt.

```
SELECT o.o_orderdate FROM customer_orders_lineitem c,c.c_orders o;
  order_date
------------------
 "2001-09-08"
(1 row)


SELECT JSON_TYPEOF(o.o_orderdate) FROM customer_orders_lineitem c,c.c_orders o;
 json_typeof
-----------------
 string
(1 row)


SELECT o.o_orderdate::date FROM customer_orders_lineitem c,c.c_orders o;
 order_date
----------------
 2001-09-08
(1 row)


--date/time cannot be cast to super
SELECT '2019-09-09'::date::super;
ERROR:  cannot cast type date to super
```

- Die Umwandlung von nicht-skalaren Werten (Objekt und Array) in eine Zeichenfolge gibt NULL zurück. Um diese nicht skalaren Werte korrekt zu serialisieren, sollten Sie sie nicht umwandeln. Verwenden Sie stattdessen `json_serialize`, um nicht-skalare Werte umzuwandeln. Die `json_serialize`-Funktion gibt einen varchar zurück. Normalerweise müssen Sie nicht skalare Werte in varchar umwandeln, da Amazon Redshift implizit serialisiert, wie im folgenden ersten Beispiel gezeigt.

```
SELECT r_nations FROM region_nations WHERE r_regionkey=300;
   r_nations
----------------
 [1,"abc",null]
(1 row)

SELECT r_nations::varchar FROM region_nations WHERE r_regionkey=300;
 r_nations
-------------
(1 row)

SELECT JSON_SERIALIZE(r_nations) FROM region_nations WHERE r_regionkey=300;
 json_serialize
-----------------
 [1,"abc",null]
(1 row)
```

  - Bei Datenbanken ohne Berücksichtigung der Groß- und Kleinschreibung unterstützt Amazon Redshift den SUPER-Datentyp nicht. Bei Spalten ohne Berücksichtigung der Groß- und Kleinschreibung werden sie von Amazon Redshift nicht in den SUPER-Typ umgewandelt. Daher unterstützt Amazon Redshift keine SUPER-Spalten, die mit Spalten ohne Berücksichtigung der Groß- und Kleinschreibung interagieren, die die Umwandlung auslösen.
- Amazon Redshift unterstützt keine flüchtigen Funktionen wie RANDOM ( ) oder TIMEOFDAY ( ) in Unterabfragen, die die Verschachtelung einer äußeren Tabelle oder linken Seite (LHS) von IN-Funktionen mit solchen Unterabfragen aufheben.

Verwenden des SUPER-Datentyps mit materialisierten Ansichten

Amazon Redshift erweitert die Funktion materialisierter Ansichten, um mit dem SUPER-Datentyp und PartiQL in materialisierten Ansichten zu funktionieren. SQL- und PartiQL-Abfragen können mit inkrementellen materialisierten Ansichten vorberechnet werden. Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

Sobald Sie Ihre schemalosen und halbstrukturierten Daten in SUPER gespeichert haben, können Sie mit PartiQL materialisierte Ansichten verwenden, um die Daten zu analysieren und in materialisierte Ansichten aufzuteilen.

Beschleunigen von PartiQL-Abfragen

Sie können materialisierte Ansichten verwenden, um PartiQL-Abfragen zu beschleunigen, die hierarchische Daten in SUPER-Spalten navigieren und/oder deren Verschachtelung aufheben. Erstellen Sie eine oder mehrere materialisierte Ansichten, um die SUPER-Werte in mehrere Spalten aufzuteilen und die spaltenförmige Organisation von Amazon-Redshift-Analyseabfragen zu verwenden. Folglich verwenden Abfragen die materialisierten Ansichten.

Die materialisierte Ansicht extrahiert und normalisiert die verschachtelten Daten im Wesentlichen. Der Grad der Normalisierung hängt davon ab, wie viel Aufwand Sie bei der Umwandlung der SUPER-Daten in herkömmliche spaltenförmige Daten aufwenden.

Aufteilen in SUPER-Spalten mit materialisierten Ansichten

Das folgende Beispiel zeigt eine materialisierte Ansicht, die die verschachtelten Daten aufteilt, wobei die resultierenden Spalten immer noch der SUPER-Datentyp sind.

```
SELECT c.c_name, o.o_orderstatus
FROM customer_orders_lineitem c, c.c_orders o;
```

Das folgende Beispiel zeigt eine materialisierte Ansicht, die herkömmliche skalare Amazon-Redshift-Spalten aus den aufgeteilten Daten erstellt.

```
SELECT c.c_name, c.c_orders[0].o_totalprice
FROM customer_orders_lineitem c;
```

Sie können eine einzelne materialisierte Ansicht super_mv erstellen, um beide Abfragen zu beschleunigen.

Um die erste Abfrage zu beantworten, müssen Sie das Attribut o_orderstatus materialisieren. Sie können das Attribut c_name weglassen, da es weder verschachtelte Navigation noch Aufheben der Verschachtelung beinhaltet. Sie müssen auch das Attribut c_custkey von customer_orders_lineitem in die materialisierte Ansicht aufnehmen, um die Basistabelle mit der materialisierten Ansicht verbinden zu können.

Um die zweite Abfrage zu beantworten, müssen Sie auch das Attribut o_totalprice und den Array-Index o_idx von c_orders materialisieren. Daher können Sie auf den Index 0 von c_orders zugreifen.

```
CREATE MATERIALIZED VIEW super_mv distkey(c_custkey) sortkey(c_custkey) AS (
  SELECT c_custkey, o.o_orderstatus, o.o_totalprice, o_idx
  FROM customer_orders_lineitem c, c.c_orders o AT o_idx
);
```

Die Attribute o_orderstatus und o_totalprice der materialisierten Ansicht super_mv sind SUPER.

Die materialisierte Ansicht super_mv wird bei Änderungen an der Basistabelle customer_orders_lineitem inkrementell aktualisiert.

```
REFRESH MATERIALIZED VIEW super_mv;
INFO: Materialized view super_mv was incrementally updated successfully.
```

Um die erste PartiQL Abfrage als reguläre SQL-Abfrage neu zu schreiben, verbinden Sie customer_orders_lineitem mit super_mv wie folgt.

```
SELECT c.c_name, v.o_orderstatus
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey;
```

In ähnlicher Weise können Sie die zweite PartiQL-Abfrage umschreiben. Im folgenden Beispiel wird ein Filter für o_idx = 0 verwendet.

```
SELECT c.c_name, v.o_totalprice
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey
WHERE v.o_idx = 0;
```

Geben Sie im Befehl CREATE MATERIALIZED VIEW c_custkey als Verteilungsschlüssel und Sortierschlüssel für super_mv an. Amazon Redshift führt einen effizienten Merge-Join durch, vorausgesetzt, dass c_custkey auch der Verteilungsschlüssel und Sortierschlüssel von customer_orders_lineitem ist. Wenn dies nicht der Fall ist, können Sie c_custkey als Sortierschlüssel und Verteilungsschlüssel von customer_orders_lineitem wie folgt angeben.

```
ALTER TABLE customer_orders_lineitem
ALTER DISTKEY c_custkey, ALTER SORTKEY (c_custkey);
```

Verwenden Sie die EXPLAIN-Anweisung, um zu überprüfen, ob Amazon Redshift einen Merge-Join für die neu geschriebenen Abfragen durchführt.

```
EXPLAIN
      SELECT c.c_name, v.o_orderstatus
      FROM customer_orders_lineitem c JOIN super_mv v ON c.c_custkey = v.c_custkey;

      QUERY PLAN

 ------------------------------------------------------------------------------------------------
      XN Merge Join DS_DIST_NONE  (cost=0.00..34701.82 rows=1470776 width=27)
      Merge Cond: ("outer".c_custkey = "inner".c_custkey)
      ->  XN Seq Scan on mv_tbl__super_mv__0 derived_table2  (cost=0.00..14999.86
 rows=1499986 width=13)
      ->  XN Seq Scan on customer_orders_lineitem c  (cost=0.00..999.96 rows=99996
 width=30)
      (4 rows)
```

Erstellen von skalaren Amazon-Redshift-Spalten aus aufgeteilten Daten

Schemalose Daten, die in SUPER gespeichert sind, können die Leistung von Amazon Redshift beeinflussen. Beispiel: Filterprädikate oder Join-Bedingungen als Scans mit eingeschränkter Reichweite können Zonenzuordnungen nicht effektiv verwenden. Benutzer und BI-Tools können materialisierte Ansichten als konventionelle Darstellung der Daten verwenden und die Leistung analytischer Abfragen steigern.

Die folgende Abfrage scannt die materialisierte Ansicht `super_mv` und filtert nach `o_orderstatus`.

```
SELECT c.c_name, v.o_totalprice
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey
WHERE v.o_orderstatus = 'F';
```

Untersuchen Sie `stl_scan`, um zu überprüfen, ob Amazon Redshift Zonenzuordnungen beim Scannen von `o_orderstatus` mit eingeschränkter Reichweite nicht effektiv verwenden kann.

```
SELECT slice, is_rrscan FROM stl_scan
WHERE query = pg_last_query_id() AND perm_table_name LIKE '%super_mv%';

 slice | is_rrscan
-------+-----------
```

```
     0 | f
     1 | f
     5 | f
     4 | f
     2 | f
     3 | f
(6 rows)
```

Im folgenden Beispiel wird die materialisierte Ansicht `super_mv` angepasst, um skalare Spalten aus den aufgeteilten Daten zu erstellen. In diesem Fall wandelt Amazon Redshift `o_orderstatus` von SUPER zu VARCHAR um. Geben Sie außerdem `o_orderstatus` als Sortierschlüssel für `super_mv` aus.

```
CREATE MATERIALIZED VIEW super_mv distkey(c_custkey) sortkey(c_custkey, o_orderstatus)
 AS (
  SELECT c_custkey, o.o_orderstatus::VARCHAR AS o_orderstatus, o.o_totalprice, o_idx
  FROM customer_orders_lineitem c, c.c_orders o AT o_idx
);
```

Stellen Sie nach dem erneuten Ausführen der Abfrage sicher, dass Amazon Redshift jetzt Zonenzuordnungen verwenden kann.

```
SELECT v.o_totalprice
FROM super_mv v
WHERE v.o_orderstatus = 'F';
```

Sie können überprüfen, ob der Scan mit eingeschränkter Reichweite jetzt Zonenzuordnungen wie folgt verwendet.

```
SELECT slice, is_rrscan FROM stl_scan
WHERE query = pg_last_query_id() AND perm_table_name LIKE '%super_mv%';

 slice | is_rrscan
-------+-----------
     0 | t
     1 | t
     2 | t
     3 | t
     4 | t
     5 | t
(6 rows)
```

Einschränkungen bei der Verwendung des SUPER-Datentyps mit materialisierten Ansichten

Wenn Sie den SUPER-Datentyp mit materialisierten Ansichten verwenden, beachten Sie folgende Einschränkungen.

Materialisierte Ansichten in Amazon Redshift haben keine spezifischen Einschränkungen in Bezug auf PartiQL oder SUPER.

Hinweise zu allgemeinen SQL-Einschränkungen beim Erstellen materialisierter Ansichten finden Sie unter Einschränkungen.

Hinweise zu allgemeinen SQL-Einschränkungen beim inkrementellen Aktualisieren materialisierter Ansichten finden Sie unter Einschränkungen für die inkrementelle Aktualisierung.

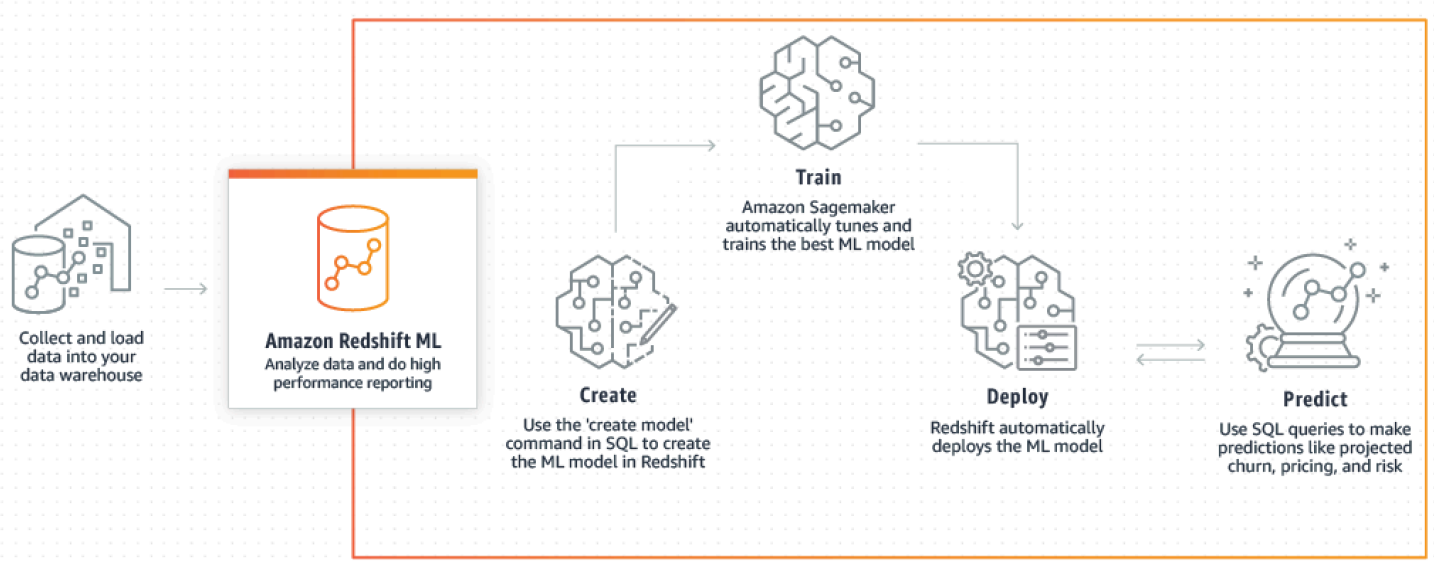
Verwendung von Machine Learning in Amazon Redshift

Amazon Redshift Machine Learning (Amazon Redshift ML) ist ein robuster, cloudbasierter Service, der Analysten und Datenwissenschaftlern aller Kenntnisstufen die Arbeit mit Machine-Learning-Technologien erleichtert. Sie geben den Daten, die Sie trainieren möchten, ein Model und Metadaten, die mit den Dateneingaben in Amazon Redshift verbunden sind. Dann erstellt Amazon Redshift ML Modelle, die Muster in den Eingabedaten erfassen. Sie können diese Modelle dann verwenden, um Prognosen für neue Eingabedaten zu generieren, ohne dass zusätzliche Kosten entstehen.

Funktionsweise von Amazon Redshift ML mit Amazon SageMaker

Amazon Redshift kann mit Amazon SageMaker Autopilot verwendet werden, um automatisch das beste Modell abzurufen und die Prognosefunktion in Amazon Redshift verfügbar zu machen.

Das folgende Diagramm veranschaulicht, wie Amazon Redshift ML funktioniert.



Im Allgemeinen sieht der Workflow wie folgt aus:

1. Amazon Redshift exportiert die Trainingsdaten in Amazon S3.
2. Amazon SageMaker Autopilot verarbeitet die Trainingsdaten vor. Die Vorverarbeitung umfasst wichtige Funktionen, wie zum Beispiel die Imputation fehlender Werte. Im Rahmen der Vorverarbeitung werden außerdem kategorische Spalten (zum Beispiel die Postleitzahl) erfasst und für das Training formatiert sowie weitere Funktionen durchgeführt. Die Auswahl der besten Präprozessoren für den Trainingsdatensatz ist ein Problem an sich, und Amazon SageMaker Autopilot automatisiert seine Lösung.

3. Amazon SageMaker Autopilot findet den Algorithmus und die Algorithmus-Hyperparameter, die dem Modell die genauesten Vorhersagen bieten.
4. Amazon Redshift registriert die Prognosefunktion als SQL-Funktion in Ihrem Amazon-Redshift-Cluster.
5. Wenn Sie CREATE MODEL-Anweisungen ausführen, verwendet Amazon Redshift Amazon SageMaker für das Training. Daher fallen Kosten für das Training Ihres Modells an. Dies ist ein separater Einzelposten für Amazon SageMaker in Ihrer AWS Rechnung. Sie zahlen außerdem für den Speicher, der in Amazon S3 für die Speicherung Ihrer Trainingsdaten verwendet wird. Inferenzen mit Modellen, die mit CREATE MODEL erstellt wurden, die Sie kompilieren und auf Ihrem Redshift-Cluster ausführen können, werden nicht berechnet. Es fallen keine zusätzlichen Amazon-Redshift-Gebühren für die Verwendung von Amazon Redshift ML an.

Themen

Übersicht zum Machine Learning

Mithilfe von Amazon Redshift ML können Sie Machine-Learning-Modelle mithilfe von SQL-Anweisungen trainieren und sie in SQL-Abfragen für Prognosen aufrufen.

Mehr über die Verwendung von Amazon Redshift ML erfahren Sie in folgendem Video: Amazon Redshift ML.

Informationen zu den Voraussetzungen für die Einrichtung Ihres Redshift-Clusters, Berechtigungen und Eigentümerschaft für die Verwendung von Amazon Redshift ML finden Sie in den folgenden Abschnitten. In diesen Abschnitten wird auch beschrieben, wie einfache Trainings und Prognosen in Amazon Redshift ML funktionieren.

Wie Machine Learning dabei hilft, ein Problem zu lösen

Machine-Learning-Modelle generieren Prognosen, indem Muster in Ihren Trainingsdaten gefunden und diese Muster dann auf neue Daten angewendet werden. Beim Machine Learning trainieren

Sie diese Modelle, indem Muster gelernt werden, die ihre Daten am besten beschreiben. Dann verwenden Sie diese Modelle, um Prognosen (auch als Inferenzen bezeichnet) für neue Daten zu stellen. Machine Learning ist normalerweise ein iterativer Prozess, bei dem Sie die Genauigkeit der Prognosen weiter verbessern können, indem Sie die Parameter ändern und Ihre Trainingsdaten verbessern. Wenn sich Daten ändern, erfolgt ein erneutes Training neuer Modelle mit dem neuen Datensatz.

Um verschiedene Geschäftsziele zu erreichen, gibt es verschiedene grundlegende Machine-Learning-Ansätze.

Supervised Learning in Amazon Redshift ML

Amazon Redshift unterstützt Supervised Learning – den gängigsten Ansatz für fortschrittliche Unternehmensanalysen. Supervised Learning ist der bevorzugte Machine-Learning-Ansatz, wenn Sie einen festgelegten Datensatz haben und verstehen, wie bestimmte Eingabedaten Prognosen zu verschiedenen Geschäftsergebnissen erstellen. Diese Ergebnisse werden manchmal als Labels bezeichnet. Ihr Datensatz ist im Wesentlichen eine Tabelle mit Attributen, die aus Merkmalen (Eingaben) und Zielen (Ausgaben) bestehen. Stellen Sie sich zum Beispiel eine Tabelle vor, die das Alter und die Postleitzahl für vergangene und gegenwärtige Kunden enthält. Nehmen Sie außerdem an, dass es das Feld „aktiv“ gibt, das für gegenwärtige Kunden den Wert „true“ hat und für vergangene Kunden, die ihre Mitgliedschaft ausgesetzt haben, den Wert „false“. Das Ziel des Supervised Machine Learning ist es, Muster im Alter und den Postleitzahlen zu erkennen, die zur Kundenabwanderung führen, die von Kunden mit dem Wert „false“ dargestellt wird. Sie können dieses Modell verwenden, um vorherzusagen, welche Kunden wahrscheinlich abwandern werden, zum Beispiel indem sie ihre Mitgliedschaft aussetzen, und gegebenenfalls Initiativen zur Bindung anzubieten.

Amazon Redshift unterstützt Supervised Learning. Dies umfasst Regression, binäre Klassifizierung und Mehrklassen-Klassifizierung. Regression bezieht sich auf das Problem der Prognose kontinuierlicher Werte, wie zum Beispiel der Gesamtausgaben der Kunden. Binäre Klassifizierung bezieht sich auf das Problem der Prognose eines von zwei Ergebnissen, z. B. die Prognose, ob ein Kunde abwandert oder nicht. Mehrklassen-Klassifizierung bezieht sich auf das Problem der Prognose eines von vielen Ergebnissen, z. B. die Prognose des Artikels, der für einen Kunden interessant sein könnte. Datenanalysten und Datenwissenschaftler können sie verwenden, um Supervised Learning durchzuführen, um Probleme bei Prognosen, der Personalisierung und der Vorhersage der Kundenabwanderung zu lösen. Sie können Supervised Learning auch bei Problemen wie der Prognose, welche Verkäufe abgeschlossen werden, Umsatzprognose, Betrugserkennung und Prognose des Kundenlebenszeitwerts nutzen.

Unsupervised Learning in Amazon Redshift ML

Unsupervised Learning verwendet Machine-Learning-Algorithmen, um nicht gekennzeichnete Trainingsdaten zu analysieren und zu gruppieren. Die Algorithmen erkennen versteckte Muster oder Gruppierungen. Ziel ist es, die zugrunde liegende Struktur oder Verteilung in den Daten zu modellieren, um weitere Informationen über die Daten zu erfahren.

Amazon Redshift unterstützt den K-Means-Clustering-Algorithmus für die Lösung von Unsupervised-Learning-Problemen. Dieser Algorithmus löst Clusterprobleme, bei denen Sie Gruppierungen in den Daten erkennen möchten. Der K-Means-Algorithmus versucht, separate Gruppierungen innerhalb der Daten zu finden. Nicht klassifizierte Daten werden aufgrund ihrer Ähnlichkeiten und Unterschiede gruppiert und partitioniert. Durch die Gruppierung ermittelt der K-Means-Algorithmus iterativ die besten Schwerpunkte und weist jedes Element dem nächstgelegenen Schwerpunkt zu. Mitglieder mit dem gleichen nächstgelegenen Schwerpunkt gehören zur selben Gruppe. Mitglieder einer Gruppe sind anderen Mitgliedern derselben Gruppe so ähnlich wie möglich und unterscheiden sich so stark wie möglich von Mitgliedern anderer Gruppen. Zum Beispiel lassen sich mithilfe des K-Means-Clustering-Algorithmus anhand der Beliebtheit von Konsumgütern Städte klassifizieren, die von einer Pandemie betroffen sind.

Bei Verwendung des K-Means-Algorithmus legen Sie eine Eingabe k fest. Diese gibt die Anzahl der Cluster an, die in den Daten gefunden werden sollen. Dieser Algorithmus gibt einen Satz von k-Schwerpunkten aus. Jeder Datenpunkt gehört zu einem der k-Cluster, der ihm am nächsten liegt. Beschrieben wird jeder Cluster durch seinen Schwerpunkt. Der Schwerpunkt kann als mehrdimensionaler Durchschnitt des Clusters bertrachtet werden. Der K-Means-Algorithmus vergleicht die Entfernungen, um zu sehen, wie sehr sich die Cluster voneinander unterscheiden. Ein größerer Abstand weist allgemein auf einen größeren Unterschied zwischen den Clustern hin.

Die Vorverarbeitung der Daten ist für K-Means wichtig, weil damit die Merkmale des Modells im gleichen Maßstab bleiben und zuverlässige Ergebnisse geliefert werden. Amazon Redshift unterstützt einige K-Means-Präprozessoren für die CREATE MODEL StandardScaler-Anweisung, z. B. MinMax, und NumericPassthrough. Wenn Sie keine Vorverarbeitung für K-Means anwenden möchten, wählen Sie NumericPassthrough explizit als Transformator aus. Informationen zum Festlegen von K-Means-Parametern finden Sie unter CREATE MODEL mit K-MEANS – Parameter.

Mehr über die Durchführung von unbeaufsichtigtem Training mit K-Means-Clustering erfahren Sie in folgendem Video:Unüberwachtes Training mit K-Means-Clustering.

Begriffe und Konzepte von Amazon Redshift ML

Die folgenden Beschreibungen werden verwendet, um einige Konzepte von Amazon Redshift ML zu beschreiben:

- Machine Learning in Amazon Redshift trainiert ein Modell mit einem SQL-Befehl. Amazon Redshift ML und Amazon SageMaker verwalten alle Datenkonvertierungen, Berechtigungen, Ressourcennutzung und Erkennung des richtigen Modells.
- Training ist die Phase, in der Amazon Redshift ein Machine-Learning-Modell erstellt, indem eine bestimmte Teilmenge von Daten im Modell ausgeführt wird. Amazon Redshift startet automatisch einen Trainingsauftrag in Amazon SageMaker und generiert ein Modell.
- Prognose (auch Inferenz genannt) ist die Verwendung des Modells in Amazon-Redshift-SQL-Abfragen, um Ergebnisse vorherzusagen. Zur Inferenzzeit verwendet Amazon Redshift eine modellbasierte Prognosefunktion als Teil einer größeren Abfrage, um Prognosen zu erstellen. Die Prognosen werden lokal im Redshift-Cluster berechnet und bieten somit einen hohen Durchsatz, eine niedrige Latenz und keine Zusatzkosten.
- Mit Bring Your Own Model (BYOM) können Sie ein Modell, das außerhalb von Amazon Redshift mit Amazon trainiert wurde, SageMaker für die lokale Inferenz in Amazon Redshift verwenden. Amazon Redshift ML unterstützt die Verwendung von BYOM in der lokalen Inferenz.
- Lokale Inferenz wird verwendet, wenn Modelle in Amazon vortrainiert SageMaker, von Amazon SageMaker Neo kompiliert und in Amazon Redshift ML lokalisiert sind. Um Modelle zu importieren, die für lokale Inferenz in Amazon Redshift unterstützt werden, verwenden Sie den Befehl CREATE MODEL. Amazon Redshift importiert die vortrainierten SageMaker Modelle, indem es Amazon SageMaker Neo aufruft. Sie kompilieren das Modell dort und importieren das kompilierte Modell in Amazon Redshift. Nutzen Sie lokale Inferenz für höhere Geschwindigkeit und niedrigere Kosten.
- Remote-Inferenz wird verwendet, wenn Amazon Redshift einen in bereitgestellten Modellendpunkt aufruft SageMaker. Remote-Inferenz bietet die Flexibilität, alle Arten von benutzerdefinierten Modellen und Deep-Learning-Modellen aufzurufen, z. B. TensorFlow Modelle, die Sie in Amazon erstellt und bereitgestellt haben SageMaker.

Wichtig sind außerdem:

- Amazon SageMaker ist ein vollständig verwalteter Machine-Learning-Service. Mit Amazon können SageMakerDatenwissenschaftler und Entwickler problemlos Modelle erstellen, trainieren und direkt in einer produktionsbereiten gehosteten Umgebung bereitstellen. Weitere Informationen

zu Amazon SageMakerfinden Sie unter Was ist Amazon SageMaker? im Amazon- SageMaker Entwicklerhandbuch.

- Amazon SageMaker Autopilot ist ein Funktionssatz, der basierend auf Ihren Daten automatisch die besten Machine-Learning-Modelle für Klassifizierung oder Regression trainiert und optimiert. Sie behalten die volle Kontrolle und Transparenz. Amazon SageMaker Autopilot unterstützt Eingabedaten im Tabellenformat. Amazon SageMaker Autopilot bietet automatische Datenbereinigung und Vorverarbeitung, automatische Algorithmusauswahl für lineare Regression, binäre Klassifikation und Mehrklassenklassifizierung. Es unterstützt auch die automatische Hyperparameteroptimierung (HPO), verteiltes Training, automatische Instances und Clustergrößenauswahl. Weitere Informationen zu Amazon SageMaker Autopilot finden Sie unter Automatisieren der Modellentwicklung mit Amazon SageMaker Autopilot im Amazon- SageMaker Entwicklerhandbuch.

Machine Learning für Anfänger und Experten

Amazon Redshift ML ermöglicht es Ihnen, Modelle mit einem einzigen SQL-CREATE-MODEL-Befehl zu trainieren. Der Befehl CREATE MODEL erstellt ein Modell, das Amazon Redshift verwendet, um modellbasierte Prognosen mit vertrauten SQL-Konstrukten zu generieren.

Amazon Redshift ML besonders nützlich, wenn Sie über keine Fachkenntnisse in Bereichen Machine Learning, Tools, Sprachen, Algorithmus und APIs verfügen. Mit Amazon Redshift ML sparen Sie sich den Aufwand, der mit der Integration eines externen Machine-Learning-Services verbunden ist. Amazon Redshift spart Ihnen Zeit, um Daten zu formatieren und zu verschieben, Berechtigungskontrollen zu verwalten und benutzerdefinierte Integrationen, Workflows und Skripts zu entwickeln. Sie können ganz einfach gängige Machine-Learning-Algorithmen verwenden und Trainingsanforderungen vereinfachen, die häufige Iterationen vom Training bis zur Prognose erfordern. Amazon Redshift erkennt automatisch den besten Algorithmus und passt das beste Modell für Ihr Problem an. Sie können Prognosen innerhalb des Amazon-Redshift-Clusters erstellen, ohne dass Daten aus Amazon Redshift verschoben werden müssen. Da Sie sich nicht mit einem anderen Service verbinden müssen, sparen Sie außerdem die damit verbundenen Kosten.

Amazon Redshift ML unterstützt Datenanalysten und Datenwissenschaftler bei der Nutzung von Machine Learning. Damit können Experten für Machine Learning auch ihr Wissen dazu nutzen, mit der Anweisung CREATE MODEL nur die von ihnen angegebenen Aspekte zu verwenden. Dadurch können Sie die Zeit verkürzen, die CREATE MODEL benötigt, um den besten Kandidaten zu finden, die Genauigkeit des Modells zu verbessern oder beides zu erreichen.

Die Anweisung CREATE MODEL bietet Ihnen Flexibilität beim Festlegen der Parameter für den Trainingsauftrag. Dank dieser Flexibilität können sowohl Anfänger als auch Experten im Bereich Machine Learning ihre bevorzugten Präprozessoren, Algorithmen, Problemtypen und Hyperparameter auswählen. So kann zum Beispiel ein Benutzer, der mehr über die Kundenabwanderung erfahren möchte, für die CREATE-MODEL-ANWEISUNG festlegen, dass der Problemtyp eine binäre Klassifizierung sein soll, die für den Bereich Kundenabwanderung geeignet ist. Die CREATE-MODEL-Anweisung schränkt die Suche nach dem besten Modell dann auf Modelle mit binärer Klassifizierung ein. Selbst wenn der Benutzer den Problemtyp frei auswählen kann, bleiben immer noch viele Optionen übrig, mit denen die CREATE-MODEL-Anweisung arbeiten kann. Beispielsweise kann CREATE MODEL die besten Vorverarbeitungstransformationen finden und anwenden sowie die besten Hyperparametereinstellungen ermitteln.

Amazon Redshift ML erleichtert das Training, indem es automatisch das beste Modell mit Amazon SageMaker Autopilot findet. Amazon SageMaker Autopilot trainiert und optimiert im Hintergrund automatisch das beste Machine-Learning-Modell basierend auf Ihren bereitgestellten Daten. Amazon SageMaker Neo kompiliert dann das Trainingsmodell und stellt es für Prognosen in Ihrem Redshift-Cluster zur Verfügung. Wenn Sie eine Machine-Learning-Inferenzabfrage mit einem trainierten Modell ausführen, kann die Abfrage die MPP-Funktionen (Massively Parallel Processing) von Amazon Redshift nutzen. Gleichzeitig kann die Abfrage auf Machine Learning basierende Vorhersage verwenden.

- Als Anfänger im Bereich Machine Learning mit allgemeinen Kenntnissen zu unterschiedlichen ML-Aspekten wie Präprozessoren, Algorithmen und Hyperparametern können Sie die CREATE-MODEL-Anweisung für die von Ihnen angegebenen Aspekte verwenden. Sie können dann die Zeit verkürzen, die CREATE MODEL benötigt, um den besten Kandidaten zu finden oder die Genauigkeit des Modells zu verbessern. Außerdem können Sie den geschäftlichen Wert der Prognosen erhöhen, indem Sie zusätzliche Domäneninformationen wie den Problemtyp oder das Ziel hinzufügen. Wenn beispielsweise in einem Kundenabwanderungsszenario das Ergebnis „Kunde ist nicht aktiv“ selten ist, wird das F1-Ziel häufig dem Ziel „Genauigkeit“ vorgezogen. Modelle mit einer hohen Genauigkeit sagen wahrscheinlich immer „Kunde ist aktiv“ voraus. Dies ergibt zwar eine hohe Genauigkeit, jedoch ist der geschäftliche Wert gering. Weitere Informationen zu F1-Zielen finden Sie unter AutoMLJobObjective in der Amazon SageMaker -API-Referenz .

  Weitere Informationen zu den grundlegenden Optionen für die CREATE-MODEL-Anweisung finden Sie unter Einfaches CREATE MODEL.
- Als Fortgeschrittener im Bereich Machine Learning können Sie den Problemtyp und die Präprozessoren für manche (nicht alle) Merkmale angeben. Dann folgt CREATE MODEL Ihren

Vorschlägen zu den angegebenen Aspekten. Gleichzeitig ermittelt CREATE MODEL noch die besten Präprozessoren für die verbleibenden Funktionen und die besten Hyperparameter. Weitere Informationen zum Einschränken von einem oder mehreren Aspekten der Trainings-Pipeline finden Sie unter CREATE MODEL mit Benutzerführung.

- Als Experte im Bereich Machine Learning haben Sie die volle Kontrolle über das Training und die Optimierung der Hyperparameter. CREATE MODEL versucht dann nicht, die besten Präprozessoren, Algorithmen und Hyperparameter zu ermitteln, da Sie alle Entscheidungen treffen. Weitere Informationen zur Verwendung von CREATE MODEL mit AUTO OFF finden Sie unter CREATE XGBoost-Modelle mit AUTO OFF.
- Als Dateningenieur können Sie ein vortrainiertes XGBoost-Modell in Amazon verwenden SageMaker und es zur lokalen Inferenz in Amazon Redshift importieren. Mit Bring Your Own Model (BYOM) können Sie ein Modell verwenden, das außerhalb von Amazon Redshift mit Amazon SageMaker für die lokale Inferenz in Amazon Redshift trainiert wurde. Amazon Redshift ML unterstützt die Verwendung von BYOM in der lokalen oder Remote-Inferenz.

  Weitere Informationen zur Verwendung der CREATE-MODEL-Anweisung für die lokale oder Remote-Inferenz finden Sie unter Bring Your Own Model (BYOM) - lokale Inferenz.

Als Benutzer von Amazon Redshift ML können Sie eine der folgenden Optionen auswählen, um Ihr Modell zu trainieren und bereitzustellen:

- Problemtypen, siehe CREATE MODEL mit Benutzerführung.
- Ziele, siehe CREATE MODEL mit Benutzerführung oder CREATE XGBoost-Modelle mit AUTO OFF.
- Modelltypen, siehe CREATE XGBoost-Modelle mit AUTO OFF.
- Präprozessoren, siehe CREATE MODEL mit Benutzerführung.
- Hyperparameter, siehe CREATE XGBoost-Modelle mit AUTO OFF.
- Bring your own model (BYOM), siehe Bring Your Own Model (BYOM) - lokale Inferenz.

Kosten für die Verwendung von Amazon Redshift ML

Amazon Redshift ML verwendet Ihre vorhandenen Clusterressourcen für Prognosen, sodass Sie zusätzliche Amazon-Redshift-Gebühren vermeiden können. Es fallen keine zusätzlichen Amazon-Redshift-Gebühren für das Erstellen oder Verwenden eines Modells an. Die Prognose erfolgt lokal in Ihrem Redshift-Cluster, sodass Sie nicht extra bezahlen müssen, es sei denn, Sie müssen die Größe

Ihres Clusters ändern. Amazon Redshift ML verwendet Amazon SageMaker für das Training Ihres Modells, für das zusätzliche Kosten anfallen.

Für Prognosefunktionen, die in Ihrem Amazon-Redshift-Cluster ausgeführt werden, fallen keine zusätzlichen Gebühren an. Die CREATE MODEL-Anweisung verwendet Amazon SageMaker und verursacht zusätzliche Kosten. Die Kosten steigen mit der Anzahl der Zellen in Ihren Trainingsdaten. Die Anzahl der Zellen ist das Produkt aus der Anzahl der Datensätze (in der Trainingsabfrage oder Tabellenzeiten) mal der Anzahl der Spalten. Wenn beispielsweise eine SELECT-Abfrage der CREATE-MODEL-Anweisung 10 000 Datensätze und 5 Spalten erstellt, beträgt die Anzahl der erstellten Zellen 50 000.

In einigen Fällen überschreiten die von der SELECT-Abfrage von CREATE MODEL erzeugten Trainingsdaten das von Ihnen angegebene MAX_CELLS-Limit (oder die Standardeinstellung 1 Million, wenn Sie kein Limit angegeben haben). In diesen Fällen wählt CREATE MODEL MAX_CELLS (d. h. die „Anzahl der Spalten“-Datensätze aus dem Trainingsdatensatz) nach dem Zufallsprinzip aus. CREATE MODEL führt dann das Training mit diesen zufällig ausgewählten Tupeln durch. Diese zufällige Auswahl stellt sicher, dass der reduzierte Trainingsdatensatz keine Verzerrung aufweist. Sie können also durch die Festlegung von MAX_CELLS die Trainingskosten kontrollieren.

Wenn Sie die Anweisung CREATE MODEL verwenden, können Sie die Optionen MAX_CELLS und MAX_RUNTIME verwenden, um die Kosten, Zeit und potenzielle Modellgenauigkeit zu kontrollieren.

MAX_RUNTIME gibt die maximale Zeit an, die das Training in Anspruch nehmen kann SageMaker , wenn die Option AUTO ON oder OFF verwendet wird. Je nach Größe des Datensatzes werden Trainingsaufträge früher als MAX_RUNTIME abgeschlossen. Nachdem ein Modell trainiert wurde, führt Amazon Redshift zusätzliche Aufgaben im Hintergrund durch, um Ihre Modelle in Ihrem Cluster zu kompilieren und zu installieren. Daher kann CREATE MODEL länger dauern als MAX_RUNTIME. MAX_RUNTIME begrenzt jedoch den Umfang der Berechnung und die Zeit, die in SageMaker zum Trainieren Ihres Modells verwendet werden. Sie können den Status Ihres Modells jederzeit mit SHOW MODEL überprüfen.

Wenn Sie CREATE MODEL mit AUTO ON ausführen, verwendet Amazon Redshift ML SageMaker Autopilot, um automatisch und intelligent verschiedene Modelle (oder Kandidaten) zu untersuchen, um das beste Modell zu finden. MAX_RUNTIME begrenzt den Zeitaufwand und die Berechnung. Wenn MAX_RUNTIME zu niedrig eingestellt ist, reicht die Zeit möglicherweise nicht einmal aus, um einen einzigen Kandidaten zu untersuchen. Wenn der Fehler „Autopilot candidate has no models“ (Autopilot-Kandidat hat keine Modelle) angezeigt wird, führen Sie CREATE MODEL erneut mit einem höheren MAX_RUNTIME-Wert aus. Weitere Informationen zu diesem Parameter finden Sie unter MaxAutoMLJobRuntimeInSeconds in der Amazon SageMaker -API-Referenz .

Wenn Sie CREATE MODEL mit AUTO OFF ausführen, entspricht MAX_RUNTIME einem Limit für die Dauer der Ausführung des Trainingsauftrags in SageMaker. Trainingsaufträge werden oft früher abgeschlossen, abhängig von der Größe des Datensatzes und anderen verwendeten Parametern, wie z. B. num_rounds in MODEL_TYPE XGBOOST.

Sie können auch Kosten kontrollieren oder die Trainingszeit reduzieren, indem Sie einen kleineren MAX_CELLS-Wert angeben, wenn Sie CREATE MODEL ausführen. Eine Zelle ist ein Eintrag in der Datenbank. Jede Zeile entspricht so vielen Zellen wie vorhandenen Spalten, die eine feste oder unterschiedliche Breite haben können. MAX_CELLS begrenzt die Anzahl der Zellen und damit die Anzahl der Trainingsbeispiele, die zum Trainieren Ihres Modells verwendet werden. Standardmäßig ist MAX_CELLS auf 1 Million Zellen festgelegt. Die Reduzierung von MAX_CELLS reduziert die Anzahl der Zeilen aus dem Ergebnis der SELECT-Abfrage in CREATE MODEL, die Amazon Redshift exportiert und an sendet, um ein Modell SageMaker zu trainieren. Die Reduzierung von MAX_CELLS reduziert somit die Größe des Datensatzes, der zum Trainieren von Modellen mit AUTO ON und AUTO OFF verwendet wird. Dieser Ansatz trägt dazu bei, die Kosten und die Zeit für das Training von Modellen zu reduzieren. Um Informationen zu Trainings- und Abrechnungszeiten eines bestimmten Trainingsauftrags anzuzeigen, wählen Sie Trainingsjobs in Amazon aus SageMaker.

Die Erhöhung von MAX_RUNTIME und MAX_CELLS verbessert häufig die Modellqualität SageMaker , da mehr Kandidaten untersucht werden können. Auf diese Weise SageMaker kann es mehr Zeit dauern, jeden Kandidaten zu trainieren und mehr Daten zu verwenden, um bessere Modelle zu trainieren. Wenn eine schnellere Iteration oder Untersuchung Ihres Datensatzes erforderlich ist, reduzieren Sie die MAX_RUNTIME und MAX_CELLS. Wenn eine höhere Genauigkeit von Modellen erforderlich ist, erhöhen Sie die MAX_RUNTIME und MAX_CELLS.

Weitere Informationen zu den Kosten für unterschiedliche Zellanzahlen und kostenlosen Testversionen finden Sie unter Amazon-Redshift-Preise.

Erste Schritte mit Amazon Redshift ML

Amazon Redshift ML erleichtert SQL-Benutzern das Erstellen, Trainieren und Bereitstellen von Machine-Learning-Modellen mit vertrauten SQL-Befehlen. Mit Amazon Redshift ML können Sie Ihre Daten in Ihrem Redshift-Cluster verwenden, um das Modell mit Amazon zu trainieren SageMaker. Danach können Sie die Modelle lokalisieren und Prognosen innerhalb einer Amazon-Redshift-Datenbank erstellen. Amazon Redshift ML unterstützt derzeit die Machine-Learning-Algorithmen XGBoost (AUTO ON und OFF), Multi-Layer-Perceptron (AUTO ON), K-Means (AUTO OFF) und den Algorithmus für lineares Lernen.

Themen

Einrichtung für die Amazon-Redshift-ML-Administration in Cluster aufteilen und konfigurieren

Bevor Sie mit Amazon Redshift ML arbeiten, schließen Sie die Cluster-Einrichtung ab und konfigurieren Sie die Berechtigungen für die Verwendung von Amazon Redshift ML.

Cluster-Einrichtung für die Verwendung von Amazon Redshift ML

Bevor Sie mit Amazon Redshift ML arbeiten können, müssen die folgenden Voraussetzungen erfüllt sein.

Die einmalige Einrichtung durch einen Amazon-Redshift-Administrator läuft wie folgt ab.

So führen Sie eine einmalige ClusterEinrichtung für Amazon Redshift ML durch

1. Erstellen Sie einen Redshift-Cluster mit der AWS Management Console oder der AWS Command Line Interface (AWS CLI). Stellen Sie sicher, dass Sie beim Erstellen des Clusters die AWS Identity and Access Management (IAM)-Richtlinie anfügen. Weitere Informationen zu Berechtigungen, die für die Verwendung von Amazon Redshift ML mit Amazon erforderlich sind SageMaker, finden Sie unter Für die Verwendung von Amazon Redshift Machine Learning (ML) mit Amazon erforderliche Berechtigungen SageMaker.
2. Erstellen Sie die IAM-Rolle, die für die Nutzung von Amazon Redshift ML erforderlich ist, mit einer der folgenden Methoden:

   - Ein einfacher Vorgang ist die Erstellung einer IAM-Rolle mit `AmazonS3FullAccess`- und-Richtlinien `AmazonSageMakerFullAccess` zur Nutzung mit Amazon Redshift ML. Wenn Sie planen, auch Prognosemodelle zu erstellen, fügen Sie die Richtlinie `AmazonForecastFullAccess` ebenfalls Ihrer Rolle hinzu.
   - Wir empfehlen, dass Sie eine IAM-Rolle über die Amazon-Redshift-Konsole erstellen, die über die `AmazonRedshiftAllCommandsFullAccess`-Richtlinie mit Berechtigungen zum Ausführen von SQL-Befehlen wie z. B. CREATE MODEL verfügt. Amazon Redshift verwendet

einen nahtlosen API-basierten Mechanismus, um in Ihrem Namen programmgesteuert IAM-AWS-Konto Rollen in Ihrem zu erstellen. Amazon Redshift fügt der IAM-Rolle automatisch vorhandene AWS verwaltete Richtlinien an. Bei dieser Methode können Sie in der Amazon-Redshift-Konsole bleiben und müssen zur Rollenerstellung nicht zur IAM-Konsole wechseln. Weitere Informationen finden Sie unter Erstellen einer IAM-Rolle als Standard für Amazon Redshift.

Wenn eine IAM-Rolle als Standard für Ihren Cluster erstellt wird, schließen Sie `redshift` als Teil des Ressourcennamens ein oder verwenden Sie ein Redshift-spezifisches Tag, um diese Ressourcen zu kennzeichnen.

Wenn für Ihren Cluster erweitertes Amazon-VPC-Routing aktiviert wurde, können Sie eine IAM-Rolle verwenden, die über die Amazon-Redshift-Konsole erstellt wurde. Dieser IAM-Rolle wurde die `AmazonRedshiftAllCommandsFullAccess`-Richtlinie angehängt und Sie fügen die folgenden Berechtigungen zur Richtlinie hinzu. Diese zusätzlichen Berechtigungen ermöglichen Amazon Redshift, eine Elastic-Network-Schnittstelle (ENI) in Ihrem Konto zu erstellen und zu löschen und an Kompilierungsaufgaben anzuhängen, die auf Amazon EC2 oder Amazon ECS ausgeführt werden. Auf diese Weise kann auf Objekte in Ihren Amazon S3 Buckets nur von innerhalb einer Virtual Private Cloud (VPC) zugegriffen werden, wobei der Internetzugang gesperrt ist.

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterfacePermission",
        "ec2:DeleteNetworkInterface",
        "ec2:CreateNetworkInterfacePermission",
        "ec2:CreateNetworkInterface",
        "ec2:ModifyNetworkInterfaceAttribute"
    ],
    "Resource": "*"
}
```

- Wenn Sie eine IAM-Rolle mit einer restriktiveren Richtlinie erstellen möchten, können Sie die folgende Richtlinie verwenden. Sie können diese Richtlinie auch entsprechend Ihren Anforderungen anpassen.

  Der Amazon-S3-Bucket `redshift-downloads/redshift-ml/` ist der Ort, an dem die Beispieldaten für andere Schritte und Beispiele gespeichert werden. Sie können ihn entfernen, wenn Sie keine Daten aus Amazon S3 laden müssen. Sie können ihn auch durch andere Amazon-S3-Buckets ersetzen, die Sie zum Laden von Daten in Amazon Redshift verwenden.

  Die Werte *`your-account-id`*, *`your-role`* und *`your-s3-bucket`* sind die, die Sie als Teil Ihres Befehls CREATE MODEL angeben.

  (Optional) Verwenden Sie den AWS KMS Schlüsselabschnitt der Beispielrichtlinie, wenn Sie bei der Verwendung von Amazon Redshift ML einen - AWS KMS Schlüssel angeben. Der Wert *`your-kms-key`* ist der Schlüssel, den Sie als Teil Ihres CREATE-MODEL-Befehls verwenden.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData",
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetAuthorizationToken",
                "ecr:GetDownloadUrlForLayer",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents",
                "sagemaker:*Job*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "s3:AbortMultipartUpload",
                "s3:GetObject",
                "s3:DeleteObject",
```

```
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:iam::<your-account-id>:role/<your-role>",
                "arn:aws:s3:::<your-s3-bucket>/*",
                "arn:aws:s3:::redshift-downloads/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::<your-s3-bucket>",
                "arn:aws:s3:::redshift-downloads"
            ]
        }
        // Optional section needed if you use AWS KMS keys.
        ,{
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:Encrypt",
                "kms:GenerateDataKey*"
             ],
             "Resource": [
                "arn:aws:kms:<your-region>:<your-account-id>:key/<your-kms-key>"
             ]
        }
    ]
}
```

3. Damit Amazon Redshift und die Rolle für SageMaker die Interaktion mit anderen -Services übernehmen können, fügen Sie der IAM-Rolle die folgende Vertrauensrichtlinie hinzu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "redshift.amazonaws.com",
          "sagemaker.amazonaws.com",
          "forecast.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

4. (Optional) Erstellen Sie einen Amazon S3-Bucket und einen - AWS KMS Schlüssel. Diese werden von Amazon Redshift verwendet, um die an Amazon gesendeten Trainingsdaten zu speichern SageMaker und das trainierte Modell von Amazon zu erhalten SageMaker.
5. (Optional) Erstellen Sie verschiedene Kombinationen von IAM-Rollen und Amazon-S3-Buckets, um den Zugriff auf verschiedene Benutzergruppen zu steuern.
6. (Optional) Wenn Sie das VPC-Routing für Ihren Redshift-Cluster aktivieren, erstellen Sie einen Amazon S3-Endpunkt und einen SageMaker Endpunkt für die VPC, in der sich Ihr Redshift-Cluster befindet. Auf diese Weise kann der Datenverkehr zwischen Diensten während der Ausführung von CREATE MODEL durch Ihre VPC laufen. Weitere Informationen zum VPC-Routing finden Sie unter Enhanced VPC Routing in Amazon Redshift.

   Weitere Informationen zu Berechtigungen, die zum Angeben einer privaten VPC für Ihren Hyperparameter-Optimierungsauftrag erforderlich sind, finden Sie unter Erforderliche Berechtigungen für die Verwendung von Amazon Redshift ML mit Amazon SageMaker.

Informationen zur Verwendung der CREATE-MODEL-Anweisung zum Erstellen von Modellen für verschiedene Anwendungsfälle finden Sie unter CREATE MODEL.

Verwalten von Berechtigungen und Eigentümerschaft

Wie bei anderen Datenbankobjekten wie z. B. Tabellen oder Funktionen bindet Amazon Redshift das Erstellen und Verwenden von ML-Modellen an Mechanismen für die Zugriffssteuerung. Für die Erstellung eines Modells, das Prognosefunktionen ausführt, sind separate Berechtigungen erforderlich.

In den folgenden Beispielen werden zwei Benutzergruppen verwendet: `retention_analyst_grp` (Modellersteller) und `marketing_analyst_grp` (Modellbenutzer), um zu veranschaulichen, wie

Amazon Redshift die Zugriffskontrolle verwaltet. Der Retention Analyst erstellt Machine-Learning-Modelle, die andere Benutzer verwenden können, wenn sie über entsprechende Berechtigungen verfügen.

Ein Superuser kann GRANT USER- oder GROUP-Berechtigungen für die Erstellung von Machine-Learning-Anweisungen erteilen. Dafür wird die folgende Anweisung verwendet.

```
GRANT CREATE MODEL TO GROUP retention_analyst_grp;
```

Benutzer oder Gruppen mit dieser Berechtigung können ein Modell in einem beliebigen Schema im Cluster erstellen, wenn ein Benutzer über die übliche CREATE-Berechtigung für das SCHEMA verfügt. Das Machine-Learning-Modell ist Teil der Schemahierarchie, ähnlich wie bei Tabellen, Ansichten, Prozeduren und benutzerdefinierte Funktionen.

Wenn beispielsweise das Schema `demo_ml` bereits vorhanden ist, können Sie zwei Benutzergruppen Berechtigungen für das Schema erteilten. Dies sieht so aus:

```
GRANT CREATE, USAGE ON SCHEMA demo_ml TO GROUP retention_analyst_grp;
```

```
GRANT USAGE ON SCHEMA demo_ml TO GROUP marketing_analyst_grp;
```

Um anderen Benutzern die Verwendung der Machine-Learning-Inferenzfunktion zu ermöglichen, erteilen Sie die EXECUTE-Berechtigung. Im folgenden Beispiel wird die EXECUTE-Berechtigung verwendet, um der marketing_analyst_grp GROUP die Berechtigung zur Verwendung des Modells zu erteilen.

```
GRANT EXECUTE ON MODEL demo_ml.customer_churn_auto_model TO GROUP
 marketing_analyst_grp;
```

Verwenden Sie die REVOKE-Anweisung mit CREATE MODEL und EXECUTE, um Benutzern oder Gruppen diese Berechtigungen wieder zu entziehen. Weitere Informationen zu Befehlen für die Berechtigung finden Sie unter GRANT und REVOKE.

Verwenden der Modellerklärbarkeit mit Amazon Redshift ML

Mit der Modellerklärbarkeit in Amazon Redshift ML verstehen Sie anhand von Werten für die Funktionsbedeutung, wie jedes Attribut in Ihren Trainingsdaten zum prognostizierten Ergebnis beiträgt.

Die Erklärbarkeit von Modellen hilft dabei, Ihre Machine-Learning-Modelle (ML) zu verbessern, indem die von den Modellen getätigten Prognosen näher erläutert werden. Die Erklärbarkeit des Modells hilft mit der Zuordnung von Funktionen zu erläutern, wie diese Modelle Prognosen erstellen.

Amazon Redshift ML nutzt die Erklärbarkeit von Modellen, um Amazon-Redshift-ML-Benutzern Funktionen zur Modellerklärung bereitzustellen. Weitere Informationen zur Modellerklärbarkeit finden Sie unter Was sind Fairness und Modellerklärbarkeit für Machine Learning-Prognosen? im Amazon-SageMaker Entwicklerhandbuch.

Die Modellerklärbarkeit überwacht auch die Inferenzen, zu denen es bei in der Produktion eingesetzten Modellen kommt, auf eine Drift der Funktionszuordnung. Sie bietet auch Tools, mit denen Sie Modell-Governance-Berichte zur Information von Risiko- und Compliance-Teams sowie externer Aufsichtsbehörden erstellen können.

Wenn Sie bei Verwendung der Anweisung CREATE MODEL die Option AUTO ON oder AUTO OFF angeben, SageMaker erstellt nach Abschluss des Modelltrainingsauftrags die Erklärungsausgabe. Sie können mithilfe der Funktion EXPLAIN_MODEL den Erklärbarkeitsbericht in einem JSON-Format abfragen. Weitere Informationen finden Sie unter Machine-Learning-Funktionen.

Wahrscheinlichkeitsmetriken von Amazon Redshift ML

Bei Supervised-Learning-Problemen sind Klassen-Labels die Ergebnisse von Vorhersagen, die Eingabedaten verwenden. Wenn Sie beispielsweise ein Modell verwenden, um vorherzusagen, ob ein Kunde einen Streaming-Dienst erneut abonnieren wird, sind mögliche Labels „likely“ (wahrscheinlich) und „unlikely“ (unwahrscheinlich). Redshift ML bietet die Möglichkeit zur Verwendung von Wahrscheinlichkeitsmetriken, die jedem Label eine Wahrscheinlichkeit zuweisen, um die Wahrscheinlichkeit seines Eintretens anzugeben. Dies hilft Ihnen, fundiertere Entscheidungen auf der Grundlage der prognostizierten Ergebnisse zu treffen. In Amazon Redshift ML sind Wahrscheinlichkeitsmetriken verfügbar, wenn AUTO-ON-Modelle mit dem Problemtyp binäre Klassifikation oder Mehrklassen-Klassifizierung erstellt werden. Wenn Sie den Parameter AUTO ON weglassen, geht Redshift ML davon aus, dass für das Modell AUTO ON gelten sollte.

Erstellen des Modells

Beim Erstellen eines Modells erkennt Amazon Redshift automatisch den Modelltyp und den Problemtyp. Wenn es sich um ein Klassifizierungsproblem handelt, erstellt Redshift automatisch eine zweite Inferenzfunktion, mit deren Hilfe Sie Wahrscheinlichkeiten in Relation zu den einzelnen Labels ausgeben können. Der Name dieser zweiten Inferenzfunktion ist der Name Ihrer angegebenen

Inferenzfunktion, gefolgt von der Zeichenfolge `_probabilities`. Wenn Sie beispielsweise Ihre Inferenzfunktion als `customer_churn_predict` bezeichnen, lautet der Name der zweiten Inferenzfunktion `customer_churn_predict_probabilities`. Sie können diese Funktion dann abfragen, um die Wahrscheinlichkeiten der einzelnen Labels zu erhalten.

```
CREATE MODEL customer_churn_model
FROM customer_activity
    PROBLEM_TYPE BINARY_CLASSIFICATION
TARGET churn
FUNCTION customer_churn_predict
IAM_ROLE {default}
AUTO ON
SETTINGS ( S3_BUCKET '<DOC-EXAMPLE-BUCKET>'
```

Abrufen von Wahrscheinlichkeiten

Wenn die Wahrscheinlichkeitsfunktion einsatzbereit ist, wird bei Ausführung des Befehls ein SUPER-Typ zurückgegeben, der Arrays der zurückgegebenen Wahrscheinlichkeiten und der zugehörigen Labels enthält. Das Ergebnis `"probabilities" : [0.7, 0.3], "labels" : ["False.", "True."]` bedeutet beispielsweise, dass das Label „False“ (Falsch) eine Wahrscheinlichkeit von 0,7 und das Label „True“ (Wahr) eine Wahrscheinlichkeit von 0,3 aufweist.

```
SELECT customer_churn_predict_probabilities(Account_length, Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins, Intl_calls,
            Intl_charge, Cust_serv_calls)
FROM customer_activity;

customer_churn_predict_probabilities
 --------------------
 {"probabilities" : [0.7, 0.3], "labels" : ["False.", "True."]}
 {"probabilities" : [0.8, 0.2], "labels" : ["False.", "True."]}
 {"probabilities" : [0.75, 0.25], "labels" : ["True.", "False"]}
```

Die Arrays der Wahrscheinlichkeiten und Labels sind immer in absteigender Reihenfolge nach ihren Wahrscheinlichkeiten sortiert. Sie können eine Abfrage schreiben, die nur das prognostizierte Label mit der höchsten Wahrscheinlichkeit zurückgibt, indem Sie die Verschachtelung der von SUPER zurückgegebenen Ergebnisse der Wahrscheinlichkeitsfunktion aufheben.

```
SELECT prediction.labels[0], prediction.probabilities[0]
```

```
            FROM (SELECT customer_churn_predict_probabilities(Account_length,
 Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins, Intl_calls,
            Intl_charge, Cust_serv_calls) AS prediction
FROM customer_activity);

  labels   | probabilities
-----------+--------------
 "False."  | 0.7
 "False."  | 0.8
 "True."   | 0.75
```

Um die Abfragen zu vereinfachen, können Sie die Ergebnisse der Vorhersagefunktion in einer Tabelle speichern.

```
CREATE TABLE churn_auto_predict_probabilities AS
            (SELECT customer_churn_predict_probabilities(Account_length, Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins,
            Intl_calls, Intl_charge, Cust_serv_calls) AS prediction
FROM customer_activity);
```

Sie können die Tabelle mit den Ergebnissen abfragen, um nur Vorhersagen mit einer Wahrscheinlichkeit von mehr als 0,7 zurückzugeben.

```
SELECT prediction.labels[0], prediction.probabilities[0]
FROM churn_auto_predict_probabilities
WHERE prediction.probabilities[0] > 0.7;

  labels   | probabilities
-----------+--------------
 "False."  | 0.8
 "True."   | 0.75
```

Mithilfe der Indexnotation können Sie die Wahrscheinlichkeit eines bestimmten Labels ermitteln. Im folgenden Beispiel werden die Wahrscheinlichkeiten aller `True.`-Labels zurückgegeben.

```
SELECT label, index, p.prediction.probabilities[index]
FROM churn_auto_predict_probabilities p, p.prediction.labels AS label AT index
WHERE label='True.';
```

```
  label  | index | probabilities
---------+-------+---------------
 "True." |     0 | 0.3
 "True." |     0 | 0.2
 "True." |     0 | 0.75
```

Im folgenden Beispiel werden alle Zeilen mit einem `True`-Label mit einer Wahrscheinlichkeit von mehr als 0,7 zurückgegeben, einem Hinweis dafür, dass der Kunde wahrscheinlich abwandern wird.

```
SELECT prediction.labels[0], prediction.probabilities[0]
FROM churn_auto_predict_probabilities
WHERE prediction.probabilities[0] > 0.7 AND prediction.labels[0] = "True.";

labels     | probabilities
-----------+--------------
 "True."   | 0.75
```

Tutorials für Amazon Redshift ML

Mithilfe von Amazon Redshift ML können Sie Machine-Learning-Modelle mit SQL-Anweisungen trainieren und sie in SQL-Abfragen für Prognosen aufrufen. Machine Learning in Amazon Redshift trainiert ein Modell mit einem SQL-Befehl. Amazon Redshift startet automatisch einen Trainingsauftrag in Amazon SageMaker und generiert ein Modell. Nachdem ein Modell erstellt wurde, können Sie mithilfe der Voraussagefunktion des Modells Prognosen in Amazon Redshift erstellen.

Befolgen Sie die Schritte in diesen Tutorials, um mehr über die Funktionen von Amazon Redshift ML zu erfahren:

- Tutorial: Erstellen von Kundenabwanderungsmodellen
- Tutorial: Erstellen von Remote-Inferenzmodellen
- Tutorial: k-Means-Clustering-Modelle erstellen
- Tutorial: Erstellen von Mehrklassen-Klassifizierungsmodellen
- Tutorial: Erstellen von XGBoost-Modellen
- Tutorial: Erstellen von Regressionsmodellen
- Tutorial: Erstellen von Regressionsmodellen mit linearem Lernen
- Tutorial: Erstellen von Mehrklassen-Klassifizierungsmodellen mit linearem Lernen

Tutorial: Erstellen von Kundenabwanderungsmodellen

In diesem Tutorial verwenden Sie Amazon Redshift ML, um mit dem Befehl CREATE MODEL ein Kundenabwanderungsmodell zu erstellen und Voraussageabfragen für Benutzerszenarien auszuführen. Anschließend implementieren Sie Abfragen mit der SQL-Funktion, die der Befehl CREATE MODEL generiert.

Sie können einen einfachen CREATE-MODEL-Befehl verwenden, um Trainingsdaten zu exportieren, ein Modell zu trainieren, das Modell zu importieren und eine Amazon-Redshift-Prognosefunktion vorzubereiten. Verwenden Sie die Anweisung CREATE MODEL, um Trainingsdaten entweder als Tabelle oder SELECT-Anweisung anzugeben.

Dieses Beispiel verwendet historische Informationen, um ein Machine-Learning-Modell für die Abwanderung von Kunden eines Mobilfunkbetreibers zu erstellen. Zuerst SageMaker trainiert Ihr Machine-Learning-Modell und testet dann Ihr Modell anhand der Profilinformationen eines beliebigen Kunden. Nachdem das Modell validiert wurde, SageMaker stellt Amazon das Modell und die Vorhersagefunktion in Amazon Redshift bereit. Mit der Voraussagefunktion können Sie prognostizieren, ob ein Kunde abwandern wird oder nicht.

Beispielanwendungsfälle

Mit Amazon Redshift ML können Sie andere binäre Klassifikationsprobleme lösen, z. B. voraussagen, ob ein Vertriebs-Lead abgeschlossen wird oder nicht. Sie können auch voraussagen, ob eine Finanztransaktion einen Betrug darstellt oder nicht.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Erstellen von Prognosen mit dem Modell

Voraussetzungen

Für dieses Tutorial benötigen Sie Folgendes:

- Sie müssen einen Amazon-Redshift-Cluster für Amazon Redshift ML einrichten. Verwenden Sie dazu die Dokumentation zum Thema Einrichtung für die Amazon-Redshift-ML-Administration in Cluster aufteilen und konfigurieren.

- Der Amazon-Redshift-Cluster, den Sie zum Erstellen des Modells verwenden, und der Amazon-S3-Bucket, der zur Bereitstellung der Trainingsdaten und zum Speichern der Modellartefakte verwendet wird, müssen sich in derselben AWS -Region befinden.
- Führen Sie einen der folgenden Schritte aus, um die in dieser Dokumentation verwendeten SQL-Befehle und den Beispieldatensatz anzuzeigen oder herunterzuladen:
  - Laden Sie die SQL-Befehle, die Datei customer_activity und die Datei abalone herunter.
  - Führen Sie mit der AWS CLI für Amazon S3 den folgenden Befehl aus. Sie können Ihren eigenen Zielpfad verwenden.

```
aws s3 cp s3://redshift-downloads/redshift-ml/tutorial-scripts/redshift-ml-
tutorial.sql </target/path>
aws s3 cp s3://redshift-downloads/redshift-ml/customer_activity/
customer_activity.csv </target/path>
aws s3 cp s3://redshift-downloads/redshift-ml/abalone_xgb/abalone_xgb.csv </target/
path>
```

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um Abfragen zu bearbeiten und auszuführen und Ergebnisse visuell darzustellen.

Durch Ausführen der folgenden Abfragen werden eine Tabelle mit dem Namen `customer_activity` erstellt und der Beispieldatensatz von Amazon S3 erfasst.

```
DROP TABLE IF EXISTS customer_activity;

CREATE TABLE customer_activity (
state varchar(2),
account_length int,
area_code int,
phone varchar(8),
intl_plan varchar(3),
vMail_plan varchar(3),
vMail_message int,
day_mins float,
day_calls int,
day_charge float,
total_charge float,
eve_mins float,
eve_calls int,
```

```
eve_charge float,
night_mins float,
night_calls int,
night_charge float,
intl_mins float,
intl_calls int,
intl_charge float,
cust_serv_calls int,
churn varchar(6),
record_date date
);

COPY customer_activity
FROM 's3://redshift-downloads/redshift-ml/customer_activity/'
REGION 'us-east-1' IAM_ROLE default
FORMAT AS CSV IGNOREHEADER 1;
```

Schritt 2: Erstellen des Machine-Learning-Modells

Die Abwanderung ist unsere Zieleingabe in diesem Modell. Alle anderen Eingaben für das Modell sind Attribute, die helfen, eine Funktion zur Voraussage der Abwanderung zu erstellen.

Im folgenden Beispiel wird die Operation CREATE MODEL verwendet, um ein Modell bereitzustellen, das voraussagen kann, ob ein Kunde aktiv sein wird. Hierfür werden Eingaben wie das Alter des Kunden, die Postleitzahl, die Ausgaben und die Fälle herangezogen. Ersetzen Sie im folgenden Beispiel *DOC-EXAMPLE-BUCKET* durch Ihren eigenen Amazon-S3-Bucket.

```
CREATE MODEL customer_churn_auto_model
FROM
    (
      SELECT state,
             account_length,
             area_code,
             total_charge/account_length AS average_daily_spend,
             cust_serv_calls/account_length AS average_daily_cases,
             churn
      FROM customer_activity
      WHERE  record_date < '2020-01-01'
     )
TARGET churn FUNCTION ml_fn_customer_churn_auto
IAM_ROLE default SETTINGS (
  S3_BUCKET '<DOC-EXAMPLE-BUCKET>'
```

```
);
```

Mit der SELECT-Abfrage im vorherigen Beispiel werden die Trainingsdaten erstellt. Die TARGET-Klausel gibt an, welche Spalte das Machine-Learning-Label ist, das die Operation CREATE MODEL zum Erlernen des Voraussagens verwendet. Die Zielspalte „churn“ (Abwanderung) gibt an, ob der Kunde noch eine aktive Mitgliedschaft hat oder die Mitgliedschaft beendet hat. Das Feld S3_BUCKET ist der Name des Amazon-S3-Buckets, den Sie zuvor erstellt haben. Der Amazon S3-Bucket wird verwendet, um Trainingsdaten und Artefakte zwischen Amazon Redshift und Amazon freizugeben SageMaker. Die restlichen Spalten sind die Merkmale, die für die Prognose verwendet werden.

Eine Zusammenfassung der Syntax und Merkmale eines einfachen Anwendungsfalls des Befehls CREATE MODEL finden Sie unter Einfaches CREATE MODEL.

Hinzufügen von Berechtigungen für die serverseitige Verschlüsselung (optional)

Amazon Redshift verwendet standardmäßig Amazon SageMaker Autopilot für das Training. Insbesondere exportiert Amazon Redshift Trainingsdaten sicher in den vom Kunden angegebenen Amazon-S3-Bucket. Wenn Sie keine `KMS_KEY_ID` angeben, werden die Daten standardmäßig unter Verwendung der serverseitigen Verschlüsselung SSE-S3 verschlüsselt.

Wenn Sie Ihre Eingabe mit serverseitiger Verschlüsselung mit einem AWS KMS verwalteten Schlüssel (SSE-MMS) verschlüsseln, fügen Sie die folgenden Berechtigungen hinzu:

```
{
    "Effect": "Allow",
    "Action": [
    "kms:Encrypt"
    "kms:Decrypt"
    ]
}
```

Weitere Informationen zu Amazon- SageMaker Rollen finden Sie unter Amazon- SageMaker Rollen im Amazon- SageMaker Entwicklerhandbuch.

Überprüfen des Status des Modelltrainings (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die Operation, um den Status des Modells zu überprüfen.

```
SHOW MODEL customer_churn_auto_model;
```

Es folgt ein Beispiel für die Ausgabe der vorherigen Operation.

```
+--------------------------
+-----------------------------------------------------------------------------------------------------------
+
|           Key            |
                 Value
             |
+--------------------------
+-----------------------------------------------------------------------------------------------------------
+
|        Model Name        |
        customer_churn_auto_model
             |
|       Schema Name        |
                 public
             |
|          Owner           |
                 awsuser
             |
|      Creation Time       |
        Tue, 14.06.2022 17:15:52
             |
|       Model State        |
                TRAINING
             |
|                          |

             |
|      TRAINING DATA:      |

             |
|          Query           | SELECT STATE, ACCOUNT_LENGTH, AREA_CODE, TOTAL_CHARGE /
 ACCOUNT_LENGTH AS AVERAGE_DAILY_SPEND, CUST_SERV_CALLS / ACCOUNT_LENGTH AS
 AVERAGE_DAILY_CASES, CHURN |
|                          |
         FROM CUSTOMER_ACTIVITY
             |
|                          |
    WHERE RECORD_DATE < '2020-01-01'
             |
|      Target Column       |
                 CHURN
             |
```

```
|                          |

             |
|       PARAMETERS:        |

             |
|        Model Type        |
                   auto
             |
|       Problem Type       |

             |
|        Objective         |

             |
|     AutoML Job Name      |
     redshiftml-20220614171552640901
             |
|      Function Name       |
        ml_fn_customer_churn_auto
             |
|   Function Parameters    |                                                  state
 account_length area_code average_daily_spend average_daily_cases
                        |
| Function Parameter Types |
      varchar int4 int4 float8 int4
             |
|         IAM Role         |
          default-aws-iam-role
             |
|        S3 Bucket         |
             DOC-EXAMPLE-BUCKET
             |
|       Max Runtime        |
                   5400
             |
+--------------------------
+-----------------------------------------------------------------------------------------------------------------
+
```

Wenn das Modelltraining abgeschlossen ist, wird die Variable `model_state` zu `Model is Ready` und die Voraussagefunktion wird verfügbar.

Schritt 3: Erstellen von Prognosen mit dem Modell

Sie können SQL-Anweisungen verwenden, um die vom Prognosemodell getroffenen Voraussagen anzuzeigen. In diesem Beispiel erhält die von der Operation CREATE MODEL erstellte Voraussagefunktion den Namen `ml_fn_customer_churn_auto`. Die Eingabeargumente für die Voraussagefunktion entsprechen den Merkmaltypen, z. B. varchar für `state` und Ganzzahl für `account_length`. Die Ausgabe der Prognosefunktion ist derselbe Typ wie die TARGET-Spalte der CREATE-MODEL-Anweisung.

1. Sie haben das Modell anhand von Daten aus der Zeit vor dem 01.01.2020 trainiert, sodass Sie jetzt die Voraussagefunktion für den Testsatz verwenden. Die folgende Abfrage zeigt die Prognosen an, ob Kunden, die sich nach dem 01.01.2020 registriert haben, von Abwanderung betroffen sein werden oder nicht.

```
SELECT
    phone,
    ml_fn_customer_churn_auto(
        state,
        account_length,
        area_code,
        total_charge / account_length,
        cust_serv_calls / account_length
    ) AS active
FROM
    customer_activity
WHERE
    record_date > '2020-01-01';
```

2. Im folgenden Beispiel wird dieselbe Voraussagefunktion für einen anderen Anwendungsfall verwendet. In diesem Fall prognostiziert Amazon Redshift das Verhältnis der abwandernden und nicht abwandernden Kunden aus unterschiedlichen Staaten mit einem Erfassungsdatum nach 01.01.2020.

```
WITH predicted AS (
    SELECT
        state,
        ml_fn_customer_churn_auto(
            state,
            account_length,
            area_code,
            total_charge / account_length,
```

```
            cust_serv_calls / account_length
        ) :: varchar(6) AS active
    FROM
        customer_activity
    WHERE
        record_date > '2020-01-01'
)
SELECT
    state,
    SUM(
        CASE
            WHEN active = 'True.' THEN 1
            ELSE 0
        END
    ) AS churners,
    SUM(
        CASE
            WHEN active = 'False.' THEN 1
            ELSE 0
        END
    ) AS nonchurners,
    COUNT(*) AS total_per_state
FROM
    predicted
GROUP BY
    state
ORDER BY
    state;
```

3. Im folgenden Beispiel wird die Voraussagefunktion für den Anwendungsfall zur Prognose des Prozentsatzes der Kunden verwendet, die in einem Bundesstaat abwandern. In diesem Fall prognostiziert Amazon Redshift den Prozentsatz der abwandernden Kunden, bei denen das Erfassungsdatum nach dem 01.01.2020 liegt.

```
WITH predicted AS (
    SELECT
        state,
        ml_fn_customer_churn_auto(
            state,
            account_length,
            area_code,
            total_charge / account_length,
            cust_serv_calls / account_length
```

```
        ) :: varchar(6) AS active
    FROM
        customer_activity
    WHERE
        record_date > '2020-01-01'
)
SELECT
    state,
    CAST((CAST((SUM(
        CASE
            WHEN active = 'True.' THEN 1
            ELSE 0
        END
    )) AS FLOAT) / CAST(COUNT(*) AS FLOAT)) AS DECIMAL (3, 2)) AS pct_churn,
    COUNT(*) AS total_customers_per_state
FROM
    predicted
GROUP BY
    state
ORDER BY
    3 DESC;
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Befehl CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: Erstellen von Remote-Inferenzmodellen

Im folgenden Tutorial werden die Schritte zum Erstellen eines Random-Cut-Forest-Modells beschrieben, das zuvor in Amazon SageMakeraußerhalb von Amazon Redshift trainiert und bereitgestellt wurde. Der Random-Cut-Forest-Algorithmus erkennt anormale Datenpunkte in einem Datensatz. Wenn Sie ein Modell mit Remote-Inferenz erstellen, können Sie Ihr Random-Cut-Forest-SageMaker Modell in Amazon Redshift integrieren. Anschließend verwenden Sie in Amazon Redshift SQL, um Vorhersagen auf einem Remote- SageMaker Endpunkt durchzuführen.

Sie können einen Befehl CREATE MODEL verwenden, um ein Machine-Learning-Modell von einem Amazon- SageMaker Endpunkt zu importieren und eine Amazon-Redshift-Vorhersagefunktion vorzubereiten. Wenn Sie die Operation CREATE MODEL verwenden, geben Sie den Endpunktnamen des SageMaker Machine-Learning-Modells an.

In diesem Tutorial erstellen Sie ein Machine-Learning-Modell von Amazon Redshift mithilfe eines SageMaker Modellendpunkts. Sobald Ihr Machine-Learning-Modell bereit ist, können Sie es verwenden, um Prognosen in Amazon Redshift zu treffen. Zuerst trainieren und erstellen Sie einen Endpunkt in Amazon SageMakerund erhalten dann den Endpunktnamen. Anschließend verwenden Sie den Befehl CREATE MODEL, um ein Modell mit Amazon Redshift ML zu erstellen. Schließlich treffen Sie Prognosen für das Modell mithilfe der Voraussagefunktion, die mit dem Befehl CREATE MODEL generiert wird.

Beispielanwendungsfälle

Sie können Random-Cut-Forest-Modelle und Remote-Inferenz zur Erkennung von Anomalien in anderen Datensätzen verwenden, z. B. zur Prognose eines raschen Anstiegs oder Rückgangs von E-Commerce-Transaktionen. Sie können auch signifikante Änderungen des Wetters oder der seismischen Aktivität voraussagen.

Aufgaben

- Voraussetzungen
- Schritt 1: Bereitstellen des Amazon- SageMaker Modells
- Schritt 2: Abrufen des SageMaker Modellendpunkts
- Schritt 3: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 4: Erstellen eines Modells mit Amazon Redshift ML
- Schritt 5: Erstellen von Prognosen mit dem Modell

Voraussetzungen

Für dieses Tutorial benötigen Sie Folgendes:

- Sie haben die administrative Einrichtung für Amazon Redshift ML abgeschlossen.
- Sie haben den NYC-Taxi-Datensatz heruntergeladen, einen Amazon-S3-Bucket erstellt und die Daten in den Amazon-S3-Bucket hochgeladen.
- Sie müssen das SageMaker Modell und den Endpunkt trainieren, bereitstellen und den Namen des SageMaker Endpunkts abrufen. Verwenden Sie diese AWS CloudFormation Vorlage, um alle SageMaker Ressourcen in Ihrem AWS Konto automatisch bereitzustellen.

Schritt 1: Bereitstellen des Amazon- SageMaker Modells

1. Um das Modell bereitzustellen, navigieren Sie zur Amazon- SageMaker Konsole und wählen Sie im Navigationsbereich unter Notebook die Option Notebook-Instances aus.
2. Wählen Sie Jupyter öffnen für das Jupyter-Notebook aus, das von der CloudFormation Vorlage erstellt wurde.
3. Wählen Sie `bring-your-own-model-remote-inference.ipynb`.
4. Richten Sie die Parameter zum Speichern der Trainingseingabe- und -ausgabe in Amazon S3 ein, indem Sie die folgenden Zeilen durch Ihren Amazon-S3-Bucket und Ihr Präfix ersetzen.

```
data_location=f"s3://{bucket}/{prefix}/",
output_path=f"s3://{bucket}/{prefix}/output",
```

5. Wählen Sie die fast-forward-Schaltfläche aus, um alle Zellen auszuführen.

Schritt 2: Abrufen des SageMaker Modellendpunkts

Wählen Sie in der Amazon- SageMaker Konsole unter Inferenz im Navigationsbereich Endpunkte und suchen Sie Ihren Modellnamen. Sie müssen den Endpunktnamen Ihres Modells kopieren, wenn Sie das Remote-Inferenzmodell in Amazon Redshift erstellen.

Schritt 3: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgenden SQL-Befehle in Amazon Redshift auszuführen. Mit diesen Befehlen werden die Tabelle `rcf_taxi_data` verworfen, falls sie vorhanden ist, eine gleichnamige Tabelle erstellt und der Beispieldatensatz in die Tabelle geladen.

```
DROP TABLE IF EXISTS public.rcf_taxi_data CASCADE;

CREATE TABLE public.rcf_taxi_data (ride_timestamp timestamp, nbr_passengers int);

COPY public.rcf_taxi_data
FROM
    's3://sagemaker-sample-files/datasets/tabular/anomaly_benchmark_taxi/
NAB_nyc_taxi.csv'
    IAM_ROLE default
    IGNOREHEADER 1
    FORMAT AS CSV;
```

Schritt 4: Erstellen eines Modells mit Amazon Redshift ML

Führen Sie die folgende Abfrage aus, um ein Modell in Amazon Redshift ML mit dem SageMaker Modellendpunkt zu erstellen, den Sie im vorherigen Schritt erhalten haben. Ersetzen Sie durch den Namen *randomcutforest-xxxxxxxxx* Ihres eigenen SageMaker Endpunkts.

```
CREATE MODEL public.remote_random_cut_forest
FUNCTION remote_fn_rcf(int)
RETURNS decimal(10, 6) SAGEMAKER '<randomcutforest-xxxxxxxxx>' IAM_ROLE default;
```

Überprüfen des Modellstatus (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die folgende Operation SHOW MODEL, um den Status des Modells zu überprüfen.

```
SHOW MODEL public.remote_random_cut_forest
```

Die Ausgabe zeigt den SageMaker Endpunkt und den Funktionsnamen.

```
+-------------------------+----------------------------------------+
|        Model Name       |       remote_random_cut_forest         |
+-------------------------+----------------------------------------+
|       Schema Name       |                public                  |
|          Owner          |                awsuser                 |
|      Creation Time      |       Wed, 15.06.2022 17:58:21         |
|       Model State       |                 READY                  |
|                         |                                        |
|       PARAMETERS:       |                                        |
```

```
|         Endpoint          |      <randomcutforest-xxxxxxxxx>        |
|      Function Name        |              remote_fn_rcf              |
|      Inference Type       |                 Remote                  |
| Function Parameter Types  |                  int4                   |
|         IAM Role          |          default-aws-iam-role           |
+---------------------------+-----------------------------------------+
```

Schritt 5: Erstellen von Prognosen mit dem Modell

Der Amazon SageMaker Random Cut Forest-Algorithmus ist darauf ausgelegt, ungewöhnliche Datenpunkte innerhalb eines Datensatzes zu erkennen. In diesem Beispiel wurde Ihr Modell entwickelt, um Spitzen bei Taxifahrten aufgrund wichtiger Veranstaltungen zu erkennen. Sie können das Modell verwenden, um anomale Ereignisse vorherzusagen, indem Sie für jeden Datenpunkt einen Anomaliewert generieren.

Verwenden Sie die folgende Abfrage, um die Anomaliewerte für den gesamten Taxi-Datensatz zu berechnen. Beachten Sie, dass Sie auf die Funktion verweisen, die Sie im vorherigen Schritt in Ihrer Anweisung CREATE MODEL verwendet haben.

```
SELECT
    ride_timestamp,
    nbr_passengers,
    public.remote_fn_rcf(nbr_passengers) AS score
FROM
    public.rcf_taxi_data;
```

Prüfen auf hohe und niedrige Anomalien (optional)

Führen Sie die folgende Abfrage aus, um Datenpunkte zu finden, deren Ergebnisse größer als drei Standardabweichungen vom Mittelwert sind.

```
WITH score_cutoff AS (
    SELECT
        STDDEV(public.remote_fn_rcf(nbr_passengers)) AS std,
        AVG(public.remote_fn_rcf(nbr_passengers)) AS mean,
        (mean + 3 * std) AS score_cutoff_value
    FROM
        public.rcf_taxi_data
)
SELECT
    ride_timestamp,
    nbr_passengers,
```

```
    public.remote_fn_rcf(nbr_passengers) AS score
FROM
    public.rcf_taxi_data
WHERE
    score > (
        SELECT
            score_cutoff_value
        FROM
            score_cutoff
    )
ORDER BY
    2 DESC;
```

Führen Sie die folgende Abfrage aus, um Datenpunkte zu finden, deren Ergebnisse größer als drei Standardabweichungen vom Mittelwert sind.

```
WITH score_cutoff AS (
    SELECT
        STDDEV(public.remote_fn_rcf(nbr_passengers)) AS std,
        AVG(public.remote_fn_rcf(nbr_passengers)) AS mean,
        (mean - 3 * std) AS score_cutoff_value
    FROM
        public.rcf_taxi_data
)
SELECT
    ride_timestamp,
    nbr_passengers,
    public.remote_fn_rcf(nbr_passengers) AS score
FROM
    public.rcf_taxi_data
WHERE
    score < (
        SELECT
            score_cutoff_value
        FROM
            score_cutoff
    )
ORDER BY
    2 DESC;
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: k-Means-Clustering-Modelle erstellen

In diesem Tutorial verwenden Sie Amazon Redshift ML, um ein Machine-Learning-Modell auf der Grundlage des k-Means-Algorithmus zu erstellen, zu trainieren und bereitzustellen. Dieser Algorithmus löst Clusterprobleme, bei denen Sie Gruppierungen in den Daten erkennen möchten. k-Means hilft beim Gruppieren von Daten, die noch nicht beschriftet wurden. Weitere Informationen zum Clustering von K-Means finden Sie unter Funktionsweise des Clusterings von K-Means im Amazon- SageMaker Entwicklerhandbuch.

Sie verwenden eine CREATE-MODEL-Operation, um ein k-Means-Modell aus einem Amazon-Redshift-Cluster zu erstellen. Sie können einen CREATE-MODEL-Befehl verwenden, um Trainingsdaten zu exportieren, ein Modell zu trainieren, das Modell zu importieren und eine Amazon-Redshift-Prognosefunktion vorzubereiten. Verwenden Sie die Operation CREATE MODEL, um Trainingsdaten entweder als Tabelle oder SELECT-Anweisung anzugeben.

In diesem Tutorial verwenden Sie k-Means für den Datensatz Global Database of Events, Language, and Tone (GDELT), der Weltnachrichten auf der ganzen Welt überwacht, und die Daten werden jede Sekunde jeden Tag gespeichert. k-Means gruppiert Ereignisse mit ähnlichem Ton, Akteuren oder Orten. Die Daten werden in zwei verschiedenen Ordnern in mehreren Dateien im Amazon Simple Storage Service gespeichert. Die Ordner sind historisch, d. h., sie decken die Jahre 1979–2013 ab, und enthalten tägliche Updates, die sich auf die Jahre 2013 und später beziehen. In diesem Beispiel verwenden wir das historische Format und greifen auf Daten von 1979 zurück.

Beispielanwendungsfälle

Sie können andere Clustering-Probleme mit Amazon Redshift ML lösen, z. B. das Gruppieren von Kunden mit ähnlichen Sehgewohnheiten bei einem Streaming-Dienst. Sie können Redshift ML auch verwenden, um die optimale Anzahl von Versandzentren für einen Lieferservice zu prognostizieren.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Erstellen von Prognosen mit dem Modell

Voraussetzungen

Zum Durchführen dieses Tutorials müssen Sie die administrative Einrichtung für Amazon Redshift ML abschließen.

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

1. Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgende Abfrage auszuführen. Die Abfrage entfernt die Tabelle `gdelt_data` aus dem öffentlichen Schema, falls sie vorhanden ist, und erstellt eine Tabelle mit demselben Namen im öffentlichen Schema.

```
DROP TABLE IF EXISTS gdelt_data CASCADE;

CREATE TABLE gdelt_data (
    GlobalEventId bigint,
    SqlDate bigint,
    MonthYear bigint,
    Year bigint,
    FractionDate double precision,
    Actor1Code varchar(256),
    Actor1Name varchar(256),
    Actor1CountryCode varchar(256),
    Actor1KnownGroupCode varchar(256),
    Actor1EthnicCode varchar(256),
    Actor1Religion1Code varchar(256),
    Actor1Religion2Code varchar(256),
    Actor1Type1Code varchar(256),
    Actor1Type2Code varchar(256),
    Actor1Type3Code varchar(256),
    Actor2Code varchar(256),
    Actor2Name varchar(256),
    Actor2CountryCode varchar(256),
    Actor2KnownGroupCode varchar(256),
    Actor2EthnicCode varchar(256),
```

```
    Actor2Religion1Code varchar(256),
    Actor2Religion2Code varchar(256),
    Actor2Type1Code varchar(256),
    Actor2Type2Code varchar(256),
    Actor2Type3Code varchar(256),
    IsRootEvent bigint,
    EventCode bigint,
    EventBaseCode bigint,
    EventRootCode bigint,
    QuadClass bigint,
    GoldsteinScale double precision,
    NumMentions bigint,
    NumSources bigint,
    NumArticles bigint,
    AvgTone double precision,
    Actor1Geo_Type bigint,
    Actor1Geo_FullName varchar(256),
    Actor1Geo_CountryCode varchar(256),
    Actor1Geo_ADM1Code varchar(256),
    Actor1Geo_Lat double precision,
    Actor1Geo_Long double precision,
    Actor1Geo_FeatureID bigint,
    Actor2Geo_Type bigint,
    Actor2Geo_FullName varchar(256),
    Actor2Geo_CountryCode varchar(256),
    Actor2Geo_ADM1Code varchar(256),
    Actor2Geo_Lat double precision,
    Actor2Geo_Long double precision,
    Actor2Geo_FeatureID bigint,
    ActionGeo_Type bigint,
    ActionGeo_FullName varchar(256),
    ActionGeo_CountryCode varchar(256),
    ActionGeo_ADM1Code varchar(256),
    ActionGeo_Lat double precision,
    ActionGeo_Long double precision,
    ActionGeo_FeatureID bigint,
    DATEADDED bigint
);
```

2. Die folgende Abfrage lädt die Beispieldaten in die Tabelle `gdelt_data`.

```
COPY gdelt_data
FROM 's3://gdelt-open-data/events/1979.csv'
REGION 'us-east-1'
```

```
IAM_ROLE default
CSV
DELIMITER '\t';
```

Untersuchen der Trainingsdaten (optional)

Verwenden Sie die folgende Abfrage, um zu sehen, mit welchen Daten Ihr Modell trainiert wird.

```
SELECT
    AvgTone,
    EventCode,
    NumArticles,
    Actor1Geo_Lat,
    Actor1Geo_Long,
    Actor2Geo_Lat,
    Actor2Geo_Long
FROM
    gdelt_data LIMIT 100;
```

Schritt 2: Erstellen des Machine-Learning-Modells

Im folgenden Beispiel wird der Befehl CREATE MODEL verwendet, um ein Modell zu erstellen, das die Daten in sieben Cluster gruppiert. Der K-Wert entspricht der Anzahl der Cluster, in die Ihre Datenpunkte unterteilt sind. Das Modell klassifiziert Ihre Datenpunkte in Cluster, in denen Datenpunkte eine größere Ähnlichkeit haben. Durch Clustering der Datenpunkte in Gruppen ermittelt der k-Means-Algorithmus iterativ das beste Clusterzentrum. Der Algorithmus weist dann jeden Datenpunkt dem nächstgelegenen Clusterzentrum zu. Mitglieder mit dem gleichen nächstgelegenen Clusterzentrum gehören zur selben Gruppe. Mitglieder einer Gruppe sind anderen Mitgliedern derselben Gruppe so ähnlich wie möglich und unterscheiden sich so stark wie möglich von Mitgliedern anderer Gruppen. Der K-Wert ist subjektiv und hängt von Methoden ab, die die Ähnlichkeiten zwischen Datenpunkten messen. Sie können den K-Wert ändern, um Clustergrößen auszugleichen, wenn die Cluster ungleichmäßig verteilt sind.

Ersetzen Sie im folgenden Beispiel *DOC-EXAMPLE-BUCKET* durch Ihren eigenen Amazon-S3-Bucket.

```
CREATE MODEL news_data_clusters
FROM
    (
        SELECT
            AvgTone,
```

```
            EventCode,
            NumArticles,
            Actor1Geo_Lat,
            Actor1Geo_Long,
            Actor2Geo_Lat,
            Actor2Geo_Long
        FROM
            gdelt_data
    ) FUNCTION news_monitoring_cluster
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE KMEANS
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT
    EXCEPT
    (K '7')
    SETTINGS (S3_BUCKET '<DOC-EXAMPLE-BUCKET>');
```

Überprüfen des Status des Modelltrainings (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die folgende Operation SHOW MODEL, um den Status des Modells zu überprüfen und herauszufinden, ob der `Model State` `Ready` lautet.

```
SHOW MODEL NEWS_DATA_CLUSTERS;
```

Wenn das Modell bereit ist, sollte die Ausgabe der vorherigen Operation zeigen, dass der `Model State` `Ready` lautet. Es folgt ein Beispiel für die Ausgabe der Operation SHOW MODEL.

```
+--------------------------
+-----------------------------------------------------------------------------------------
+
|        Model Name        |
 news_data_clusters                                    |
+--------------------------
+-----------------------------------------------------------------------------------------
+
|       Schema Name        |                                                 public
                                          |
|          Owner           |                                               awsuser
                                          |
```

```
|      Creation Time       |                                        Fri, 17.06.2022
 16:32:19                                  |
|       Model State        |                                            READY
                                          |
|        train:msd         |                                          2973.822754
                                          |
|      train:progress      |                                           100.000000
                                          |
|     train:throughput     |                                         237114.875000
                                          |
|      Estimated Cost      |                                            0.004983
                                          |
|                          |
                                          |
|      TRAINING DATA:      |
                                          |
|          Query           | SELECT AVGTONE, EVENTCODE, NUMARTICLES, ACTOR1GEO_LAT,
 ACTOR1GEO_LONG, ACTOR2GEO_LAT, ACTOR2GEO_LONG |
|                          |                                            FROM GDELT_DATA
                                          |
|                          |
                                          |
|       PARAMETERS:        |
                                          |
|        Model Type        |                                               kmeans
                                          |
|    Training Job Name     |
 redshiftml-20220617163219978978-kmeans                                  |
|      Function Name       |
 news_monitoring_cluster                                        |
|   Function Parameters    |       avgtone eventcode numarticles actor1geo_lat
 actor1geo_long actor2geo_lat actor2geo_long        |
| Function Parameter Types |                              float8 int8 int8 float8 float8
 float8 float8                            |
|         IAM Role         |                                       default-aws-iam-
role                                        |
|        S3 Bucket         |                                          DOC-EXAMPLE-
BUCKET                                       |
|       Max Runtime        |                                               5400
                                          |
|                          |
                                          |
|     HYPERPARAMETERS:     |
                                          |
```

```
|       feature_dim        |                                                 7
                                        |
|            k             |                                                 7
                                        |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
```

Schritt 3: Erstellen von Prognosen mit dem Modell

Identifizieren der Cluster

Sie können diskrete Gruppierungen, auch Cluster genannt, finden, die von Ihrem Modell in den Daten identifiziert wurden. Ein Cluster ist ein Satz von Datenpunkten, der näher an seinem Clusterzentrum als an allen anderen Clusterzentren liegt. Da der K-Wert die Anzahl der Cluster im Modell darstellt, stellt er auch die Anzahl der Clusterzentren dar. Die folgende Abfrage identifiziert die Cluster, indem sie den Cluster anzeigt, der der jeweiligen `globaleventid` zugeordnet ist.

```
SELECT
    globaleventid,
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS cluster
FROM
    gdelt_data;
```

Überprüfen der Verteilung der Daten

Sie können die Verteilung der Daten auf die Cluster überprüfen, um festzustellen, ob der von Ihnen ausgewählte K-Wert dazu geführt hat, dass die Daten etwas gleichmäßiger verteilt wurden. Verwenden Sie die folgende Abfrage, um zu ermitteln, ob die Daten gleichmäßig auf Ihre Cluster verteilt sind.

```
SELECT
    events_cluster,
    COUNT(*) AS nbr_events
```

```
FROM
    (
        SELECT
            globaleventid,
            news_monitoring_cluster(
                AvgTone,
                EventCode,
                NumArticles,
                Actor1Geo_Lat,
                Actor1Geo_Long,
                Actor2Geo_Lat,
                Actor2Geo_Long
            ) AS events_cluster
        FROM
            gdelt_data
    )
GROUP BY
    1;
```

Hinweis: Sie können den K-Wert ändern, um Clustergrößen auszugleichen, wenn die Cluster ungleichmäßig verteilt sind.

Ermitteln der Clusterzentren

Ein Datenpunkt liegt näher an seinem Clusterzentrum als an allen anderen Clusterzentren. Das Auffinden der Clusterzentren hilft Ihnen daher, die Cluster zu definieren.

Führen Sie die folgende Abfrage aus, um die Zentren der Cluster basierend auf der Anzahl der Artikel nach Ereigniscode zu ermitteln.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    eventcode,
    SUM(numArticles) AS numArticles
FROM
```

```
    gdelt_data
GROUP BY
    1,
    2;
```

Anzeigen der Informationen über Datenpunkte in einem Cluster

Verwenden Sie die folgende Abfrage, um die Daten für die Punkte zurückzugeben, die dem fünften Cluster zugewiesen sind. Die ausgewählten Artikel müssen zwei Akteure haben.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    eventcode,
    actor1name,
    actor2name,
    SUM(numarticles) AS totalarticles
FROM
    gdelt_data
WHERE
    events_cluster = 5
    AND actor1name <> ' '
    AND actor2name <> ' '
GROUP BY
    1,
    2,
    3,
    4
ORDER BY
    5 desc;
```

Anzeigen von Daten über Ereignisse mit Akteuren desselben ethnischen Codes

Die folgende Abfrage zählt die Anzahl der Artikel, die positiv über Ereignisse berichten. Die Abfrage erfordert auch, dass die beiden Akteure denselben ethnischen Code haben, und sie gibt zurück, welchem Cluster jedes Ereignis zugewiesen ist.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    SUM(numarticles) AS total_articles,
    eventcode AS event_code,
    Actor1EthnicCode AS ethnic_code
FROM
    gdelt_data
WHERE
    Actor1EthnicCode = Actor2EthnicCode
    AND Actor1EthnicCode <> ' '
    AND Actor2EthnicCode <> ' '
    AND AvgTone > 0
GROUP BY
    1,
    3,
    4
HAVING
    (total_articles) > 4
ORDER BY
    1,
    2 ASC;
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning

- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: Erstellen von Mehrklassen-Klassifizierungsmodellen

In diesem Tutorial verwenden Sie Amazon Redshift ML, um ein Machine-Learning-Modell zu erstellen, das Mehrklassen-Klassifizierungsprobleme löst. Der Mehrklassen-Klassifizierungsalgorithmus klassifiziert Datenpunkte in eine von drei oder mehr Klassen. Anschließend implementieren Sie Abfragen mit der SQL-Funktion, die der Befehl CREATE MODEL generiert.

Sie können einen CREATE-MODEL-Befehl verwenden, um Trainingsdaten zu exportieren, ein Modell zu trainieren, das Modell zu importieren und eine Amazon-Redshift-Prognosefunktion vorzubereiten. Verwenden Sie die Operation CREATE MODEL, um Trainingsdaten entweder als Tabelle oder SELECT-Anweisung anzugeben.

Um dem Tutorial zu folgen, verwenden Sie den öffentlichen Datensatz E-Commerce Sales Forecast, der Verkaufsdaten eines britischen Online-Einzelhändlers enthält. Das von Ihnen generierte Modell richtet sich an die aktivsten Kunden für ein spezielles Kundenbindungsprogramm. Mit der Mehrklassen-Klassifizierung können Sie anhand des Modells voraussagen, wie viele Monate ein Kunde über einen Zeitraum von 13 Monaten aktiv sein wird. Die Prognosefunktion bestimmt Kunden, die voraussichtlich 7 oder mehr Monate aktiv sind, um zum Programm zugelassen zu werden.

Beispielanwendungsfälle

Sie können andere Mehrklassen-Klassifizierungsprobleme mit Amazon Redshift ML lösen, z. B. das meistverkaufte Produkt einer Produktlinie voraussagen. Sie können auch prognostizieren, welche Früchte ein Bild enthält, z. B. Äpfel, Birnen oder Orangen auswählen.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Erstellen von Prognosen mit dem Modell

Voraussetzungen

Zum Durchführen dieses Tutorials müssen Sie die administrative Einrichtung für Amazon Redshift ML abschließen.

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgenden Abfragen auszuführen. Diese Abfragen laden die Beispieldaten in Amazon Redshift.

1. Mit der folgenden Abfrage wird eine Tabelle mit dem Namen `ecommerce_sales` erstellt.

```
CREATE TABLE IF NOT EXISTS ecommerce_sales (
    invoiceno VARCHAR(30),
    stockcode VARCHAR(30),
    description VARCHAR(60),
    quantity DOUBLE PRECISION,
    invoicedate VARCHAR(30),
    unitprice DOUBLE PRECISION,
    customerid BIGINT,
    country VARCHAR(25)
);
```

2. Die folgende Abfrage kopiert die Beispieldaten aus dem Datensatz „E-Commerce Sales Forecast“ in die `ecommerce_sales`-Tabelle.

```
COPY ecommerce_sales
FROM
    's3://redshift-ml-multiclass/ecommerce_data.txt'
IAM_ROLE default
DELIMITER '\t'
IGNOREHEADER 1
REGION 'us-east-1'
MAXERROR 100;
```

Aufteilen der Daten

Wenn Sie ein Modell in Amazon Redshift ML erstellen, teilt Ihre Daten SageMaker automatisch in Trainings- und Testsätze auf, sodass die Modellgenauigkeit bestimmen SageMaker kann. Durch manuelles Aufteilen der Daten in diesem Schritt können Sie die Genauigkeit des Modells überprüfen, indem Sie einen zusätzlichen Prognosesatz zuweisen.

Verwenden Sie die folgende SQL-Anweisung, um die Daten für Training, Validierung und Voraussage auf drei Sätze aufzuteilen.

```
--creates table with all data
CREATE TABLE ecommerce_sales_data AS (
    SELECT
        t1.stockcode,
        t1.description,
        t1.invoicedate,
        t1.customerid,
        t1.country,
        t1.sales_amt,
        CAST(RANDOM() * 100 AS INT) AS data_group_id
    FROM
        (
            SELECT
                stockcode,
                description,
                invoicedate,
                customerid,
                country,
                SUM(quantity * unitprice) AS sales_amt
            FROM
                ecommerce_sales
            GROUP BY
                1,
                2,
                3,
                4,
                5
        ) t1
);

--creates training set
CREATE TABLE ecommerce_sales_training AS (
    SELECT
        a.customerid,
        a.country,
        a.stockcode,
        a.description,
        a.invoicedate,
        a.sales_amt,
        (b.nbr_months_active) AS nbr_months_active
```

```
    FROM
        ecommerce_sales_data a
        INNER JOIN (
            SELECT
                customerid,
                COUNT(
                    DISTINCT(
                        DATE_PART(y, CAST(invoicedate AS DATE)) || '-' || LPAD(
                            DATE_PART(mon, CAST(invoicedate AS DATE)),
                            2,
                            '00'
                        )
                    )
                ) AS nbr_months_active
            FROM
                ecommerce_sales_data
            GROUP BY
                1
        ) b ON a.customerid = b.customerid
    WHERE
        a.data_group_id < 80
);

--creates validation set
CREATE TABLE ecommerce_sales_validation AS (
    SELECT
        a.customerid,
        a.country,
        a.stockcode,
        a.description,
        a.invoicedate,
        a.sales_amt,
        (b.nbr_months_active) AS nbr_months_active
    FROM
        ecommerce_sales_data a
        INNER JOIN (
            SELECT
                customerid,
                COUNT(
                    DISTINCT(
                        DATE_PART(y, CAST(invoicedate AS DATE)) || '-' || LPAD(
                            DATE_PART(mon, CAST(invoicedate AS DATE)),
                            2,
                            '00'
```

```
                        )
                    )
                ) AS nbr_months_active
            FROM
                ecommerce_sales_data
            GROUP BY
                1
        ) b ON a.customerid = b.customerid
    WHERE
        a.data_group_id BETWEEN 80
        AND 90
);

--creates prediction set
CREATE TABLE ecommerce_sales_prediction AS (
    SELECT
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    FROM
        ecommerce_sales_data
    WHERE
        data_group_id > 90);
```

Schritt 2: Erstellen des Machine-Learning-Modells

In diesem Schritt verwenden Sie die Anweisung CREATE MODEL, um Ihr Machine-Learning-Modell mithilfe der Mehrklassen-Klassifizierung zu erstellen.

Die folgende Abfrage erstellt das Mehrklassen-Klassifizierungsmodell mit dem Trainingssatz unter Verwendung der Operation CREATE MODEL. Ersetzen Sie *DOC-EXAMPLE-BUCKET* durch Ihren eigenen Amazon-S3-Bucket.

```
CREATE MODEL ecommerce_customer_activity
FROM
    (
        SELECT
            customerid,
            country,
            stockcode,
```

```
            description,
            invoicedate,
            sales_amt,
            nbr_months_active
        FROM
            ecommerce_sales_training
    ) TARGET nbr_months_active FUNCTION predict_customer_activity IAM_ROLE default
 PROBLEM_TYPE MULTICLASS_CLASSIFICATION SETTINGS (
        S3_BUCKET '<DOC-EXAMPLE-BUCKET>',
        S3_GARBAGE_COLLECT OFF
    );
```

In dieser Abfrage geben Sie den Problemtyp als `Multiclass_Classification` an. Das Ziel, das Sie für das Modell voraussagen, ist `nbr_months_active`. Wenn das Training des Modells SageMaker abgeschlossen hat, wird die Funktion erstelltpredict_customer_activity, mit der Sie Vorhersagen in Amazon Redshift treffen.

Anzeigen des Status des Modelltrainings (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die folgende Abfrage, um verschiedene Metriken des Modells zurückzugeben, einschließlich Modellstatus und Genauigkeit.

```
SHOW MODEL ecommerce_customer_activity;
```

Wenn das Modell bereit ist, sollte die Ausgabe der vorherigen Operation zeigen, dass der `Model State` `Ready` lautet. Es folgt ein Beispiel für die Ausgabe der Operation SHOW MODEL.

```
+--------------------------
+-----------------------------------------------------------------------------------------
+
|        Model Name        |
 ecommerce_customer_activity                                |
+--------------------------
+-----------------------------------------------------------------------------------------
+
|       Schema Name        |                                                 public
                                  |
|          Owner           |                                                 awsuser
                                  |
```

```
|      Creation Time       |                                  Fri, 17.06.2022 19:02:15
                                    |
|       Model State        |                                            READY
                                    |
|   Training Job Status    |
 MaxAutoMLJobRuntimeReached                                  |
|   validation:accuracy    |                                         0.991280
                                    |
|      Estimated Cost      |                                         7.897689
                                    |
|                          |
                                    |
|      TRAINING DATA:      |
                                    |
|          Query           | SELECT CUSTOMERID, COUNTRY, STOCKCODE, DESCRIPTION,
 INVOICEDATE, SALES_AMT, NBR_MONTHS_ACTIVE |
|                          |                                    FROM
 ECOMMERCE_SALES_TRAINING                                |
|      Target Column       |                                        NBR_MONTHS_ACTIVE
                                    |
|                          |
                                    |
|       PARAMETERS:        |
                                    |
|        Model Type        |                                          xgboost
                                    |
|       Problem Type       |                                 MulticlassClassification
                                    |
|        Objective         |                                         Accuracy
                                    |
|     AutoML Job Name      |
 redshiftml-20220617190215268770                              |
|      Function Name       |
 predict_customer_activity                                |
|   Function Parameters    |               customerid country stockcode description
 invoicedate sales_amt               |
| Function Parameter Types |                           int8 varchar varchar varchar
 varchar float8                        |
|         IAM Role         |                                  default-aws-iam-role
                                    |
|        S3 Bucket         |                                      DOC-EXAMPLE-BUCKET
                                    |
|       Max Runtime        |                                           5400
                                    |
```

```
+-------------------------
+-------------------------------------------------------------------------------------------------------------------
+
```

Schritt 3: Erstellen von Prognosen mit dem Modell

Die folgende Abfrage zeigt, welche Kunden sich für Ihr Kundenbindungsprogramm qualifizieren. Wenn das Modell voraussagt, dass der Kunde mindestens sieben Monate lang aktiv sein wird, wählt das Modell den Kunden für das Treueprogramm aus.

```
SELECT
    customerid,
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) AS predicted_months_active
FROM
    ecommerce_sales_prediction
WHERE
    predicted_months_active >= 7
GROUP BY
    1,
    2
LIMIT
    10;
```

Ausführen von Voraussageabfragen für die Validierungsdaten (optional)

Führen Sie die folgenden Voraussageabfragen für die Validierungsdaten aus, um den Genauigkeitsgrad des Modells zu ermitteln.

```
SELECT
    CAST(SUM(t1.match) AS decimal(7, 2)) AS predicted_matches,
    CAST(SUM(t1.nonmatch) AS decimal(7, 2)) AS predicted_non_matches,
    CAST(SUM(t1.match + t1.nonmatch) AS decimal(7, 2)) AS total_predictions,
    predicted_matches / total_predictions AS pct_accuracy
FROM
    (
        SELECT
```

```
            customerid,
            country,
            stockcode,
            description,
            invoicedate,
            sales_amt,
            nbr_months_active,
            predict_customer_activity(
                customerid,
                country,
                stockcode,
                description,
                invoicedate,
                sales_amt
            ) AS predicted_months_active,
            CASE
                WHEN nbr_months_active = predicted_months_active THEN 1
                ELSE 0
            END AS match,
            CASE
                WHEN nbr_months_active <> predicted_months_active THEN 1
                ELSE 0
            END AS nonmatch
        FROM
            ecommerce_sales_validation
    )t1;
```

Prognostizieren, wie viele Kunden nicht aufgenommen werden (optional)

Die folgende Abfrage vergleicht die Anzahl der Kunden, die voraussichtlich nur 5 oder 6 Monate aktiv sind. Das Modell prognostiziert, dass sich diese Kunden nicht für das Treueprogramm qualifizieren werden. Die Abfrage vergleicht dann die Anzahl der nicht in das Programm aufgenommenen Kunden mit der Anzahl, die sich laut Prognose für das Treueprogramm qualifizieren. Diese Abfrage könnte verwendet werden, um eine Entscheidung darüber zu treffen, ob der Schwellenwert für das Treueprogramm gesenkt werden soll. Sie können auch feststellen, ob es eine erhebliche Anzahl von Kunden gibt, die sich laut Prognose nicht für das Programm qualifizieren werden. Sie können diese Kunden dann ermutigen, ihre Aktivität zu steigern, um in das Treueprogramm aufgenommen zu werden.

```
SELECT
    predict_customer_activity(
        customerid,
```

```
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) AS predicted_months_active,
    COUNT(customerid)
FROM
    ecommerce_sales_prediction
WHERE
    predicted_months_active BETWEEN 5 AND 6
GROUP BY
    1
ORDER BY
    1 ASC
LIMIT
    10)
UNION
(SELECT
      NULL AS predicted_months_active,
    COUNT (customerid)
FROM
    ecommerce_sales_prediction
WHERE
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) >=7);
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: Erstellen von XGBoost-Modellen

In diesem Tutorial erstellen Sie ein Modell mit Daten aus Amazon S3 und führen Prognoseabfragen mit dem Modell mithilfe von Amazon Redshift ML aus. Der XGBoost-Algorithmus ist eine optimierte Implementierung eines Baumalgorithmus mit Gradient Boosting. XGBoost verarbeitet mehr Datentypen, Beziehungen und Verteilungen als andere Baumalgorithmen mit Gradient Boosting. Sie können XGBoost für Regressions-, Binär- und Mehrklassen-Klassifizierungs- und Ranglistenprobleme verwenden. Weitere Informationen zum XGBoost-Algorithmus finden Sie unter XGBoost-Algorithmus im Amazon- SageMaker Entwicklerhandbuch.

Die Operation `CREATE MODEL` von Amazon Redshift ML mit der Option `AUTO OFF` unterstützt derzeit XGBoost als `MODEL_TYPE`. Sie können relevante Informationen wie das Ziel und die Hyperparameter als Teil des Befehls `CREATE MODEL` basierend auf Ihrem Anwendungsfall angeben.

In diesem Tutorial verwenden Sie den Banknotenauthentifizierungs-Datensatz, ein binäres Klassifizierungsproblem, um vorauszusagen, ob eine bestimmte Banknote echt oder gefälscht ist.

Beispielanwendungsfälle

Mit Amazon Redshift ML können Sie andere binäre Klassifikationsprobleme lösen, z. B. voraussagen, ob ein Patient gesund ist oder an einer Krankheit leidet. Sie können auch prognostizieren, ob es sich bei einer E-Mail um Spam handelt oder nicht.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Erstellen von Prognosen mit dem Modell

Voraussetzungen

Zum Durchführen dieses Tutorials müssen Sie die administrative Einrichtung für Amazon Redshift ML abschließen.

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgenden Abfragen auszuführen.

Mit der folgenden Abfrage werden zwei Tabellen erstellt, die Daten aus Amazon S3 geladen und auf einen Trainingsatz und einen Testsatz aufgeteilt. Sie verwenden das Trainingsset, um Ihr Modell zu trainieren und die Voraussagefunktion zu erstellen. Anschließend testen Sie die Voraussagefunktion mit dem Testsatz.

```
--create training set table
CREATE TABLE banknoteauthentication_train(
    variance FLOAT,
    skewness FLOAT,
    curtosis FLOAT,
    entropy FLOAT,
    class INT
);

--Load into training table
COPY banknoteauthentication_train
FROM
    's3://redshiftbucket-ml-sagemaker/banknote_authentication/train_data/' IAM_ROLE
 default REGION 'us-west-2' IGNOREHEADER 1 CSV;

--create testing set table
CREATE TABLE banknoteauthentication_test(
    variance FLOAT,
    skewness FLOAT,
    curtosis FLOAT,
    entropy FLOAT,
    class INT
);

--Load data into testing table
COPY banknoteauthentication_test
FROM
    's3://redshiftbucket-ml-sagemaker/banknote_authentication/test_data/'
    IAM_ROLE default
    REGION 'us-west-2'
    IGNOREHEADER 1
    CSV;
```

Schritt 2: Erstellen des Machine-Learning-Modells

Die folgende Abfrage erstellt das XGBoost-Modell in Amazon Redshift ML aus dem Trainingssatz, den Sie im vorherigen Schritt erstellt haben. Ersetzen Sie `DOC-EXAMPLE-BUCKET` mit Ihrem eigenen `S3_BUCKET`, in dem Ihre Eingabedatensätze und andere Redshift-ML-Artefakte gespeichert werden.

```
CREATE MODEL model_banknoteauthentication_xgboost_binary
FROM
    banknoteauthentication_train
    TARGET class
    FUNCTION func_model_banknoteauthentication_xgboost_binary
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE xgboost
    OBJECTIVE 'binary:logistic'
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT
EXCEPT(NUM_ROUND '100')
SETTINGS(S3_BUCKET '<DOC-EXAMPLE-BUCKET>');
```

Anzeigen des Status des Modelltrainings (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die folgende Abfrage, um den Trainingsfortschritt des Modells zu überwachen.

```
SHOW MODEL model_banknoteauthentication_xgboost_binary;
```

Wenn das Modell `READY` ist, bietet die Operation SHOW MODEL auch die `train:error`-Metrik, wie im folgenden Beispiel für die Ausgabe gezeigt. Die `train:error`-Metrik ist ein Maß für die Genauigkeit Ihres Modells, das bis zu sechs Dezimalstellen misst. Ein Wert von 0 ist am genauesten und ein Wert von 1 ist am wenigsten genau.

```
+--------------------------+-------------------------------------------------+
|        Model Name        |   model_banknoteauthentication_xgboost_binary   |
+--------------------------+-------------------------------------------------+
| Schema Name              | public                                          |
| Owner                    | awsuser                                         |
| Creation Time            | Tue, 21.06.2022 19:07:35                        |
| Model State              | READY                                           |
| train:error              |                                        0.000000 |
```

```
| Estimated Cost           |                                                0.006197 |
|                          |                                                         |
| TRAINING DATA:           |                                                         |
| Query                    | SELECT *                                                |
|                          | FROM "BANKNOTEAUTHENTICATION_TRAIN"                     |
| Target Column            | CLASS                                                   |
|                          |                                                         |
| PARAMETERS:              |                                                         |
| Model Type               | xgboost                                                 |
| Training Job Name        | redshiftml-20220621190735686935-xgboost                 |
| Function Name            | func_model_banknoteauthentication_xgboost_binary        |
| Function Parameters      | variance skewness curtosis entropy                      |
| Function Parameter Types | float8 float8 float8 float8                             |
| IAM Role                 | default-aws-iam-role                                    |
| S3 Bucket                | DOC-EXAMPLE-BUCKET    |
| Max Runtime              |                                                    5400 |
|                          |                                                         |
| HYPERPARAMETERS:         |                                                         |
| num_round                |                                                     100 |
| objective                | binary:logistic                                         |
+--------------------------+---------------------------------------------------------+
```

Schritt 3: Erstellen von Prognosen mit dem Modell

Überprüfen der Genauigkeit des Modells

Die folgende Voraussageabfrage verwendet die im vorherigen Schritt erstellte Prognosefunktion, um die Genauigkeit Ihres Modells zu überprüfen. Führen Sie diese Abfrage für den Testsatz aus, um sicherzustellen, dass das Modell dem Trainingssatz nicht zu genau entspricht. Diese enge Übereinstimmung wird auch als Überanpassung bezeichnet. Eine Überanpassung kann dazu führen, dass das Modell unzuverlässige Voraussagen trifft.

```
WITH predict_data AS (
    SELECT
        class AS label,
        func_model_banknoteauthentication_xgboost_binary (variance, skewness, curtosis,
 entropy) AS predicted,
        CASE
            WHEN label IS NULL THEN 0
            ELSE label
        END AS actual,
```

```
        CASE
            WHEN actual = predicted THEN 1 :: INT
            ELSE 0 :: INT
        END AS correct
    FROM
        banknoteauthentication_test
),
aggr_data AS (
    SELECT
        SUM(correct) AS num_correct,
        COUNT(*) AS total
    FROM
        predict_data
)
SELECT
    (num_correct :: FLOAT / total :: FLOAT) AS accuracy
FROM
    aggr_data;
```

Prognostizieren der Menge der echten und der gefälschten Banknoten

Die folgende Prognoseabfrage gibt die prognostizierte Anzahl der echten und gefälschten Banknoten im Testsatz zurück.

```
WITH predict_data AS (
    SELECT
        func_model_banknoteauthentication_xgboost_binary(variance, skewness, curtosis,
 entropy) AS predicted
    FROM
        banknoteauthentication_test
)
SELECT
    CASE
        WHEN predicted = '0' THEN 'Original banknote'
        WHEN predicted = '1' THEN 'Counterfeit banknote'
        ELSE 'NA'
    END AS banknote_authentication,
    COUNT(1) AS count
FROM
    predict_data
GROUP BY
    1;
```

Ermitteln des Durchschnitts der Beobachtungen echter und gefälschter Banknoten

Die folgende Prognoseabfrage gibt den Durchschnittswert jedes Merkmals für Banknoten zurück, bei denen im Testsatz vorausgesagt wird, dass sie echt und gefälscht sind.

```
WITH predict_data AS (
    SELECT
        func_model_banknoteauthentication_xgboost_binary(variance, skewness, curtosis,
 entropy) AS predicted,
          variance,
          skewness,
          curtosis,
          entropy
    FROM
        banknoteauthentication_test
)
SELECT
    CASE
        WHEN predicted = '0' THEN 'Original banknote'
        WHEN predicted = '1' THEN 'Counterfeit banknote'
        ELSE 'NA'
    END AS banknote_authentication,
    TRUNC(AVG(variance), 2) AS avg_variance,
    TRUNC(AVG(skewness), 2) AS avg_skewness,
    TRUNC(AVG(curtosis), 2) AS avg_curtosis,
    TRUNC(AVG(entropy), 2) AS avg_entropy
FROM
    predict_data
GROUP BY
    1
ORDER BY
    2;
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: Erstellen von Regressionsmodellen

In diesem Tutorial verwenden Sie Amazon Redshift ML, um ein Regressionsmodell für Machine Learning zu erstellen und Voraussageabfragen für das Modell auszuführen. Mit Regressionsmodellen können Sie numerische Ergebnisse vorhersagen, z. B. den Preis eines Hauses oder wie viele Personen den Fahrradverleih einer Stadt nutzen werden. Sie verwenden den Befehl CREATE MODEL in Amazon Redshift mit Ihren Trainingsdaten. Anschließend kompiliert Amazon Redshift ML das Modell, importiert das trainierte Modell in Redshift und bereitet eine SQL-Voraussagefunktion vor. Sie können die Voraussagefunktion in SQL-Abfragen in Amazon Redshift verwenden.

In diesem Tutorial erstellen Sie mithilfe von Amazon Redshift ML ein Regressionsmodell, das die Anzahl der Personen vorhersagt, die den Bike-Sharing-Service der Stadt Toronto zu einer bestimmten Tageszeit nutzen. Die Eingaben für das Modell umfassen Feiertage und Wetterbedingungen. Sie verwenden ein Regressionsmodell, da für dieses Problem ein numerisches Ergebnis erwünscht ist.

Sie können den Befehl CREATE MODEL verwenden, um Trainingsdaten zu exportieren, ein Modell zu trainieren und es in Amazon Redshift als SQL-Funktion verfügbar zu machen. Verwenden Sie die Operation CREATE MODEL, um Trainingsdaten entweder als Tabelle oder SELECT-Anweisung anzugeben.

Beispielanwendungsfälle

Sie können andere Regressionsprobleme mit Amazon Redshift ML lösen, z. B. den Wert für die Lebensdauer eines Kunden prognostizieren. Sie können Redshift ML auch verwenden, um den profitabelsten Preis und den daraus resultierenden Umsatz eines Produkts vorauszusagen.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Validieren des Modells

Voraussetzungen

Zum Durchführen dieses Tutorials müssen Sie die administrative Einrichtung für Amazon Redshift ML abschließen.

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgenden Abfragen auszuführen.

1. Sie müssen drei Tabellen erstellen, um die drei öffentlichen Datensätze in Amazon Redshift zu laden. Die Datensätze sind Toronto Bike Ridership Data, Historical Weather Data und Historical Holidays Data. Führen Sie die folgende Abfrage im Abfrage-Editor von Amazon Redshift aus, um Tabellen mit dem Namen `ridership`, `weather` und `holiday` zu erstellen.

```
CREATE TABLE IF NOT EXISTS ridership (
    trip_id INT,
    trip_duration_seconds INT,
    trip_start_time timestamp,
    trip_stop_time timestamp,
    from_station_name VARCHAR(50),
    to_station_name VARCHAR(50),
    from_station_id SMALLINT,
    to_station_id SMALLINT,
    user_type VARCHAR(20)
);

CREATE TABLE IF NOT EXISTS weather (
    longitude_x DECIMAL(5, 2),
    latitude_y DECIMAL(5, 2),
    station_name VARCHAR(20),
    climate_id BIGINT,
    datetime_utc TIMESTAMP,
    weather_year SMALLINT,
    weather_month SMALLINT,
    weather_day SMALLINT,
    time_utc VARCHAR(5),
    temp_c DECIMAL(5, 2),
    temp_flag VARCHAR(1),
    dew_point_temp_c DECIMAL(5, 2),
    dew_point_temp_flag VARCHAR(1),
    rel_hum SMALLINT,
    rel_hum_flag VARCHAR(1),
```

```
    precip_amount_mm DECIMAL(5, 2),
    precip_amount_flag VARCHAR(1),
    wind_dir_10s_deg VARCHAR(10),
    wind_dir_flag VARCHAR(1),
    wind_spd_kmh VARCHAR(10),
    wind_spd_flag VARCHAR(1),
    visibility_km VARCHAR(10),
    visibility_flag VARCHAR(1),
    stn_press_kpa DECIMAL(5, 2),
    stn_press_flag VARCHAR(1),
    hmdx SMALLINT,
    hmdx_flag VARCHAR(1),
    wind_chill VARCHAR(10),
    wind_chill_flag VARCHAR(1),
    weather VARCHAR(10)
);

CREATE TABLE IF NOT EXISTS holiday (holiday_date DATE, description VARCHAR(100));
```

2. Die folgende Abfrage lädt die Beispieldaten in die Tabellen, die Sie im vorherigen Schritt erstellt haben.

```
COPY ridership
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/ridership/'
    IAM_ROLE default
    FORMAT CSV
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;

COPY weather
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/weather/'
    IAM_ROLE default
    FORMAT csv
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;
```

```
COPY holiday
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/holiday/'
    IAM_ROLE default
    FORMAT csv
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;
```

3. Die folgende Abfrage führt Transformationen für die Datensätze `ridership` und `weather` zur Beseitigung von Verzerrungen oder Anomalien aus. Das Entfernen von Verzerrungen und Anomalien führt zu einer verbesserten Modellgenauigkeit. Die Abfrage vereinfacht die Tabellen, indem sie zwei neue Ansichten mit dem Namen `ridership_view` und `weather_view` erstellt.

```
CREATE
OR REPLACE VIEW ridership_view AS
SELECT
    trip_time,
    trip_count,
    TO_CHAR(trip_time, 'hh24') :: INT trip_hour,
    TO_CHAR(trip_time, 'dd') :: INT trip_day,
    TO_CHAR(trip_time, 'mm') :: INT trip_month,
    TO_CHAR(trip_time, 'yy') :: INT trip_year,
    TO_CHAR(trip_time, 'q') :: INT trip_quarter,
    TO_CHAR(trip_time, 'w') :: INT trip_month_week,
    TO_CHAR(trip_time, 'd') :: INT trip_week_day
FROM
    (
        SELECT
            CASE
                WHEN TRUNC(r.trip_start_time) < '2017-07-01' :: DATE THEN
 CONVERT_TIMEZONE(
                    'US/Eastern',
                    DATE_TRUNC('hour', r.trip_start_time)
                )
                ELSE DATE_TRUNC('hour', r.trip_start_time)
            END trip_time,
            COUNT(1) trip_count
        FROM
            ridership r
```

```
        WHERE
            r.trip_duration_seconds BETWEEN 60
            AND 60 * 60 * 24
        GROUP BY
            1
    );

CREATE
OR REPLACE VIEW weather_view AS
SELECT
    CONVERT_TIMEZONE(
        'US/Eastern',
        DATE_TRUNC('hour', datetime_utc)
    ) daytime,
    ROUND(AVG(temp_c)) temp_c,
    ROUND(AVG(precip_amount_mm)) precip_amount_mm
FROM
    weather
GROUP BY
    1;
```

4. Die folgende Abfrage erstellt eine Tabelle, die alle relevanten Eingabeattribute aus `ridership_view` und `weather_view` in Tabelle `trip_data` kombiniert.

```
CREATE TABLE trip_data AS
SELECT
    r.trip_time,
    r.trip_count,
    r.trip_hour,
    r.trip_day,
    r.trip_month,
    r.trip_year,
    r.trip_quarter,
    r.trip_month_week,
    r.trip_week_day,
    w.temp_c,
    w.precip_amount_mm,CASE
        WHEN h.holiday_date IS NOT NULL THEN 1
        WHEN TO_CHAR(r.trip_time, 'D') :: INT IN (1, 7) THEN 1
        ELSE 0
    END is_holiday,
    ROW_NUMBER() OVER (
        ORDER BY
```

```
            RANDOM()
    ) serial_number
FROM
    ridership_view r
    JOIN weather_view w ON (r.trip_time = w.daytime)
    LEFT OUTER JOIN holiday h ON (TRUNC(r.trip_time) = h.holiday_date);
```

Anzeigen der Beispieldaten (optional)

Die folgende Abfrage zeigt Einträge aus der Tabelle. Sie können diesen Vorgang ausführen, um sicherzustellen, dass die Tabelle korrekt erstellt wurde.

```
SELECT *
FROM trip_data
LIMIT 5;
```

Es folgt ein Beispiel für die Ausgabe der vorherigen Operation.

```
+---------------------+------------+-----------+----------+------------+-----------
+--------------+-----------------+---------------+--------+-----------------
+------------+---------------+
|      trip_time      | trip_count | trip_hour | trip_day | trip_month | trip_year
 | trip_quarter | trip_month_week | trip_week_day | temp_c | precip_amount_mm |
 is_holiday | serial_number |
+---------------------+------------+-----------+----------+------------+-----------
+--------------+-----------------+---------------+--------+-----------------
+------------+---------------+
| 2017-03-21 22:00:00 |         47 |        22 |       21 |          3 |        17 |
         1 |               3 |             3 |      1 |                0 |          0 |
             1 |
| 2018-05-04 01:00:00 |         19 |         1 |        4 |          5 |        18 |
         2 |               1 |             6 |     12 |                0 |          0 |
             3 |
| 2018-01-11 10:00:00 |         93 |        10 |       11 |          1 |        18 |
         1 |               2 |             5 |      9 |                0 |          0 |
             5 |
| 2017-10-28 04:00:00 |         20 |         4 |       28 |         10 |        17 |
         4 |               4 |             7 |     11 |                0 |          1 |
             7 |
```

```
| 2017-12-31 21:00:00 |         11 |        21 |       31 |         12 |        17 |
         4 |               5 |             1 |    -15 |                0 |          1 |
             9 |
+---------------------+------------+-----------+----------+------------+-----------
+--------------+-----------------+---------------+---------+------------------
+------------+---------------+
```

Anzeigen der Korrelation zwischen Attributen (optional)

Die Bestimmung der Korrelation hilft Ihnen, die Stärke der Assoziation zwischen Attributen zu messen. Mithilfe der Zuordnungsebene können Sie feststellen, was Ihre Zielausgabe beeinflusst. In diesem Tutorial lautet die Zielausgabe `trip_count`.

Die folgende Abfrage erstellt oder ersetzt das Verfahren `sp_correlation`. Sie verwenden die gespeicherte Prozedur namens `sp_correlation`, um die Korrelation zwischen einem Attribut und anderen Attributen in einer Tabelle in Amazon Redshift anzuzeigen.

```
CREATE OR REPLACE PROCEDURE sp_correlation(source_schema_name in varchar(255),
 source_table_name in varchar(255), target_column_name in varchar(255),
 output_temp_table_name inout varchar(255)) AS $$
DECLARE
  v_sql varchar(max);
  v_generated_sql varchar(max);
  v_source_schema_name varchar(255)=lower(source_schema_name);
  v_source_table_name varchar(255)=lower(source_table_name);
  v_target_column_name varchar(255)=lower(target_column_name);
BEGIN
  EXECUTE 'DROP TABLE IF EXISTS ' || output_temp_table_name;
  v_sql = '
SELECT
  ''CREATE temp table '|| output_temp_table_name||' AS SELECT ''|| outer_calculation||
  '' FROM (SELECT COUNT(1) number_of_items, SUM('||v_target_column_name||')
 sum_target, SUM(POW('||v_target_column_name||',2)) sum_square_target, POW(SUM('||
v_target_column_name||'),2) square_sum_target,''||
  inner_calculation||
  '' FROM (SELECT ''||
  column_name||
  '' FROM '||v_source_table_name||'))''
FROM
  (
  SELECT
    DISTINCT
    LISTAGG(outer_calculation,'','') OVER () outer_calculation
```

```
    ,LISTAGG(inner_calculation,'','') OVER () inner_calculation
    ,LISTAGG(column_name,'','') OVER () column_name
  FROM
    (
    SELECT
      CASE WHEN atttypid=16 THEN ''DECODE(''||column_name||'',true,1,0)'' ELSE
 column_name END column_name
     ,atttypid
     ,''CAST(DECODE(number_of_items * sum_square_''||rn||'' - square_sum_''||
rn||'',0,null,(number_of_items*sum_target_''||rn||'' - sum_target * sum_''||rn||
       '')/SQRT((number_of_items * sum_square_target - square_sum_target) *
 (number_of_items * sum_square_''||rn||
       '' - square_sum_''||rn||''))) AS numeric(5,2)) ''||column_name
 outer_calculation
     ,''sum(''||column_name||'') sum_''||rn||'','''||
           ''SUM(trip_count*''||column_name||'') sum_target_''||rn||'','''||
           ''SUM(POW(''||column_name||'',2)) sum_square_''||rn||'','''||
           ''POW(SUM(''||column_name||''),2) square_sum_''||rn inner_calculation
   FROM
     (
     SELECT
       row_number() OVER (order by a.attnum) rn
       ,a.attname::VARCHAR column_name
       ,a.atttypid
     FROM pg_namespace AS n
       INNER JOIN pg_class AS c ON n.oid = c.relnamespace
       INNER JOIN pg_attribute AS a ON c.oid = a.attrelid
     WHERE a.attnum > 0
       AND n.nspname = '''||v_source_schema_name||'''
       AND c.relname = '''||v_source_table_name||'''
       AND a.atttypid IN (16,20,21,23,700,701,1700)
     )
   )
)';
  EXECUTE v_sql INTO v_generated_sql;
  EXECUTE  v_generated_sql;
END;
$$ LANGUAGE plpgsql;
```

Die folgende Abfrage zeigt die Korrelation zwischen der Zielspalte `trip_count` und anderen numerischen Attributen in unserem Datensatz.

```
call sp_correlation(
```

```
    'public',
    'trip_data',
    'trip_count',
    'tmp_corr_table'
);

SELECT
    *
FROM
    tmp_corr_table;
```

Das folgende Beispiel zeigt die Ausgabe der vorherigen Operation `sp_correlation`.

```
+------------+-----------+----------+------------+-----------+-------------
+-----------------+---------------+--------+------------------+------------
+---------------+
| trip_count | trip_hour | trip_day | trip_month | trip_year | trip_quarter
 | trip_month_week | trip_week_day | temp_c | precip_amount_mm | is_holiday |
 serial_number |
+------------+-----------+----------+------------+-----------+-------------
+-----------------+---------------+--------+------------------+------------
+---------------+
|          1 |      0.32 |     0.01 |       0.18 |      0.12 |         0.18 |
     0 |          0.02 |   0.53 |            -0.07 |      -0.13 |             0 |
+------------+-----------+----------+------------+-----------+-------------
+-----------------+---------------+--------+------------------+------------
+---------------+
```

Schritt 2: Erstellen des Machine-Learning-Modells

1. Die folgende Abfrage teilt Ihre Daten in einen Trainingssatz und einen Validierungssatz auf, indem 80 % des Datensatzes für das Training und 20 % für die Validierung bestimmt werden. Der Trainingssatz ist die Eingabe für das ML-Modell, um den bestmöglichen Algorithmus für das Modell zu ermitteln. Nachdem das Modell erstellt wurde, verwenden Sie den Validierungssatz, um die Modellgenauigkeit zu überprüfen.

   ```
   CREATE TABLE training_data AS
   SELECT
       trip_count,
       trip_hour,
       trip_day,
       trip_month,
   ```

```
    trip_year,
    trip_quarter,
    trip_month_week,
    trip_week_day,
    temp_c,
    precip_amount_mm,
    is_holiday
FROM
    trip_data
WHERE
    serial_number > (
        SELECT
            COUNT(1) * 0.2
        FROM
            trip_data
    );

CREATE TABLE validation_data AS
SELECT
    trip_count,
    trip_hour,
    trip_day,
    trip_month,
    trip_year,
    trip_quarter,
    trip_month_week,
    trip_week_day,
    temp_c,
    precip_amount_mm,
    is_holiday,
    trip_time
FROM
    trip_data
WHERE
    serial_number <= (
        SELECT
            COUNT(1) * 0.2
        FROM
            trip_data
    );
```

2. Die folgende Abfrage erstellt ein Regressionsmodell zur Voraussage des Werts `trip_count` für ein beliebiges Eingabedatum und eine Uhrzeit. Ersetzen Sie im folgenden Beispiel *DOC-EXAMPLE-BUCKET* durch Ihren eigenen S3-Bucket.

```
CREATE MODEL predict_rental_count
FROM
    training_data TARGET trip_count FUNCTION predict_rental_count
    IAM_ROLE default
    PROBLEM_TYPE regression
    OBJECTIVE 'mse'
    SETTINGS (
        s3_bucket '<DOC-EXAMPLE-BUCKET>',
        s3_garbage_collect off,
        max_runtime 5000
    );
```

Schritt 3: Validieren des Modells

1. Verwenden Sie die folgende Abfrage, um Aspekte des Modells auszugeben und die Mean-Square-Error-Metrik in der Ausgabe zu ermitteln. Der Mean Square Error ist eine typische Genauigkeitsmetrik für Regressionsprobleme.

```
show model predict_rental_count;
```

2. Führen Sie die folgenden Voraussageabfragen für die Validierungsdaten aus, um die prognostizierte Anzahl der Fahrten mit der tatsächlichen Anzahl der Fahrten zu vergleichen.

```
SELECT
    trip_time,
    actual_count,
    predicted_count,
    (actual_count - predicted_count) difference
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
```

```
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    )
LIMIT
    5;
```

3. Die folgende Abfrage berechnet den Mean Square Error und den Root Mean Square Error basierend auf Ihren Validierungsdaten. Sie verwenden den Mean Square Error und den Root Mean Square Error, um die Abweichung zwischen dem vorausgesagten numerischen Ziel und der tatsächlichen numerischen Antwort zu messen. Ein gutes Modell erzielt bei beiden Metriken ein niedriges Ergebnis. Die folgende Abfrage gibt den Wert beider Metriken zurück.

```
SELECT
    ROUND(
        AVG(POWER((actual_count - predicted_count), 2)),
        2
    ) mse,
    ROUND(
        SQRT(AVG(POWER((actual_count - predicted_count), 2))),
        2
    ) rmse
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
```

```
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    );
```

4. Die folgende Abfrage berechnet den prozentualen Fehler bei der Anzahl der Fahrten für jede Fahrtzeit am 01.01.2017. Die Abfrage ordnet die Fahrtzeiten von der Zeit mit dem niedrigsten prozentualen Fehler bis zur Zeit mit dem höchsten prozentualen Fehler an.

```
SELECT
    trip_time,
    CAST(ABS(((actual_count - predicted_count) / actual_count)) * 100 AS DECIMAL
 (7,2)) AS pct_error
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    )
WHERE
   trip_time LIKE '2017-01-01 %%:%%:%%'
ORDER BY
   2 ASC;
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: Erstellen von Regressionsmodellen mit linearem Lernen

In diesem Tutorial erstellen Sie ein lineares Lernmodell mit Daten aus Amazon S3 und führen Voraussageabfragen mit dem Modell mithilfe von Amazon Redshift ML aus. Der SageMaker lineare Lernalgorithmus löst entweder Regressions- oder Mehrklassen-Klassifizierungsprobleme. Weitere Informationen zu Regressions- und Mehrklassen-Klassifizierungsproblemen finden Sie unter Problemtypen für die Machine-Learning-Parameter im Amazon- SageMaker Entwicklerhandbuch. In diesem Tutorial lösen Sie ein Regressionsproblem. Der Algorithmus für lineares Lernen trainiert viele Modelle parallel und ermittelt automatisch das am besten optimierte Modell. Sie verwenden die Operation CREATE MODEL in Amazon Redshift, die Ihr lineares Lernmodell mit erstellt SageMaker und eine Vorhersagefunktion an Amazon Redshift sendet. Weitere Informationen zum Algorithmus für lineares Lernen finden Sie unter Algorithmus für lineares Lernen im Amazon- SageMaker Entwicklerhandbuch.

Sie können einen CREATE-MODEL-Befehl verwenden, um Trainingsdaten zu exportieren, ein Modell zu trainieren, das Modell zu importieren und eine Amazon-Redshift-Prognosefunktion vorzubereiten. Verwenden Sie die Operation CREATE MODEL, um Trainingsdaten entweder als Tabelle oder SELECT-Anweisung anzugeben.

Lineare Lernmodelle optimieren entweder kontinuierliche oder diskrete Ziele. Kontinuierliche Ziele werden für die Regression verwendet, während diskrete Variablen für die Klassifizierung genutzt werden. Einige Methoden, wie die Regressionsmethode, bieten eine Lösung nur für kontinuierliche Ziele. Der Algorithmus für lineares Lernen bietet eine Steigerung der Geschwindigkeit

gegenüber naiven Hyperparameter-Optimierungstechniken wie der Naive-Bayes-Technik. Eine naive Optimierungstechnik geht davon aus, dass jede Eingabevariable unabhängig ist. Um den Algorithmus für lineares Lernen verwenden zu können, müssen Sie Spalten, die die Dimensionen der Eingaben darstellen, und Zeilen, die Beobachtungen repräsentieren, bereitstellen. Weitere Informationen zum Algorithmus für lineares Lernen finden Sie im Algorithmus für lineares Lernen im Amazon-SageMaker Entwicklerhandbuch.

In diesem Tutorial erstellen Sie ein lineares Lernmodell, das das Alter von Abalonen voraussagen kann. Sie verwenden den Befehl CREATE MODEL für den Abalone-Datensatz, um die Beziehung zwischen den verschiedenen physikalischen Messungen der Abalone zu bestimmen. Anschließend verwenden Sie das Modell, um das Alter der Abalone zu bestimmen.

Beispielanwendungsfälle

Sie können andere Regressionsprobleme mit dem linearen Lernmodell und Amazon Redshift ML lösen, z. B. den Preis eines Hauses prognostizieren. Sie können Redshift ML auch verwenden, um die Anzahl der Personen vorauszusagen, die den Fahrradverleih einer Stadt nutzen werden.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Validieren des Modells

Voraussetzungen

Zum Durchführen dieses Tutorials müssen Sie die administrative Einrichtung für Amazon Redshift ML abschließen.

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgenden Abfragen auszuführen. Diese Abfragen laden die Beispieldaten in Redshift und teilen die Daten in einen Trainingssatz und einen Validierungssatz auf.

1. Mit der folgenden Abfrage wird die Tabelle `abalone_dataset` erstellt.

```
CREATE TABLE abalone_dataset (
```

```
    id INT IDENTITY(1, 1),
    Sex CHAR(1),
    Length float,
    Diameter float,
    Height float,
    Whole float,
    Shucked float,
    Viscera float,
    Shell float,
    Rings integer
);
```

2. Die folgende Abfrage kopiert die Beispieldaten aus dem Abalone-Datensatz in Amazon S3 in die Tabelle `abalone_dataset`, die Sie zuvor in Amazon Redshift erstellt haben.

```
COPY abalone_dataset
FROM
    's3://redshift-ml-multiclass/abalone.csv' REGION 'us-east-1' IAM_ROLE default CSV
 IGNOREHEADER 1 NULL AS 'NULL';
```

3. Durch manuelles Aufteilen der Daten können Sie die Genauigkeit des Modells überprüfen, indem Sie einen zusätzlichen Prognosesatz zuweisen. Die folgende Abfrage teilt die Daten in zwei Sätze auf. Die Tabelle `abalone_training` ist für das Training und die Tabelle `abalone_validation` für die Validierung bestimmt.

```
CREATE TABLE abalone_training as
SELECT
    *
FROM
    abalone_dataset
WHERE
    mod(id, 10) < 8;

CREATE TABLE abalone_validation as
SELECT
    *
FROM
    abalone_dataset
WHERE
    mod(id, 10) >= 8;
```

Schritt 2: Erstellen des Machine-Learning-Modells

In diesem Schritt verwenden Sie die Anweisung CREATE MODEL, um Ihr Machine-Learning-Modell mit dem linearen Lernalgorithmus zu erstellen.

Die folgende Abfrage erstellt das lineare Lernmodell mit der Operation CREATE MODEL unter Verwendung Ihres S3-Buckets. Ersetzen Sie *DOC-EXAMPLE-BUCKET* durch Ihren eigenen S3-Bucket.

```
CREATE MODEL model_abalone_ring_prediction
FROM
    (
        SELECT
            Sex,
            Length,
            Diameter,
            Height,
            Whole,
            Shucked,
            Viscera,
            Shell,
            Rings AS target_label
        FROM
            abalone_training
    ) TARGET target_label FUNCTION f_abalone_ring_prediction IAM_ROLE default
 MODEL_TYPE LINEAR_LEARNER PROBLEM_TYPE REGRESSION OBJECTIVE 'MSE' SETTINGS (
        S3_BUCKET 'DOC-EXAMPLE-BUCKET',
        MAX_RUNTIME 15000
    );
```

Anzeigen des Status des Modelltrainings (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die folgende Abfrage, um den Trainingsfortschritt des Modells zu überwachen.

```
SHOW MODEL model_abalone_ring_prediction;
```

Wenn das Modell bereit ist, sollte die Ausgabe der vorherigen Operation dem folgenden Beispiel ähneln. Beachten Sie, dass die Ausgabe die Metrik `validation:mse` angibt, die dem Mean Square Error entspricht. Sie verwenden den Mean Square Error im nächsten Schritt, um die Genauigkeit des Modells zu überprüfen.

```
+--------------------------
+-----------------------------------------------------------------------------------------------------
+
|        Model Name        |
 model_abalone_ring_prediction                                    |
+--------------------------
+-----------------------------------------------------------------------------------------------------
+
| Schema Name              | public
                                        |
| Owner                    | awsuser
                                        |
| Creation Time            | Thu, 30.06.2022 18:00:10
                                        |
| Model State              | READY
                                        |
| validation:mse           |
                                   4.168633 |
| Estimated Cost           |
                                   4.291608 |
|                          |
                                        |
| TRAINING DATA:           |
                                        |
| Query                    | SELECT SEX , LENGTH , DIAMETER , HEIGHT , WHOLE ,
 SHUCKED , VISCERA , SHELL, RINGS AS TARGET_LABEL |
|                          | FROM ABALONE_TRAINING
                                        |
| Target Column            | TARGET_LABEL
                                        |
|                          |
                                        |
| PARAMETERS:              |
                                        |
| Model Type               | linear_learner
                                        |
| Problem Type             | Regression
                                        |
| Objective                | MSE
                                        |
| AutoML Job Name          | redshiftml-20220630180010947843
                                        |
```

```
| Function Name            | f_abalone_ring_prediction
                                        |
| Function Parameters      | sex length diameter height whole shucked viscera shell
                                        |
| Function Parameter Types | bpchar float8 float8 float8 float8 float8 float8 float8
                                        |
| IAM Role                 | default-aws-iam-role
                                        |
| S3 Bucket                | DOC-EXAMPLE-BUCKET
                                        |
| Max Runtime              |
                                   15000 |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
```

Schritt 3: Validieren des Modells

1. Die folgende Voraussageabfrage validiert die Genauigkeit des Modells für den Datensatz `abalone_validation` durch Berechnung des Mean Square Errors und des Root Mean Square Errors.

```
SELECT
    ROUND(AVG(POWER((tgt_label - predicted), 2)), 2) mse,
    ROUND(SQRT(AVG(POWER((tgt_label - predicted), 2))), 2) rmse
FROM
    (
        SELECT
            Sex,
            Length,
            Diameter,
            Height,
            Whole,
            Shucked,
            Viscera,
            Shell,
            Rings AS tgt_label,
            f_abalone_ring_prediction(
                Sex,
                Length,
                Diameter,
                Height,
                Whole,
```

```
                Shucked,
                Viscera,
                Shell
            ) AS predicted,
            CASE
                WHEN tgt_label = predicted then 1
                ELSE 0
            END AS match,
            CASE
                WHEN tgt_label <> predicted then 1
                ELSE 0
            END AS nonmatch
        FROM
            abalone_validation
    ) t1;
```

Die Ausgabe der vorherigen Abfrage sollte dem folgenden Beispiel ähneln. Der Wert der Mean-Square-Error-Metrik sollte ähnlich wie die Metrik `validation:mse` sein, die in der Ausgabe der Operation SHOW MODEL angezeigt wird.

```
+-----+--------------------+
| mse |        rmse        |
+-----+--------------------+
| 5.1 | 2.2600000000000002 |
+-----+--------------------+
```

2. Verwenden Sie die folgende Abfrage, um die Operation EXPLAIN_MODEL für Ihre Prognosefunktion auszuführen. Die Operation gibt einen Modellerklärbarkeitsbericht zurück. Weitere Informationen über die Operation EXPLAIN_MODEL finden Sie unter Funktion EXPLAIN_MODEL im Datenbankentwicklerhandbuch zu Amazon Redshift.

```
SELECT
    EXPLAIN_MODEL ('model_abalone_ring_prediction');
```

Die folgenden Informationen sind ein Beispiel für den Modellerklärbarkeitsbericht, der mit der vorherigen Operation EXPLAIN_MODEL erstellt wurde. Die Werte für die einzelnen Eingaben sind Shapley-Werte. Die Shapley-Werte stellen den Effekt dar, den jede Eingabe auf die Voraussage Ihres Modells hat, wobei höherwertige Eingaben einen größeren Einfluss auf die Voraussage haben. In diesem Beispiel haben die höherwertigen Eingaben einen größeren Einfluss auf die Voraussage des Alters der Abalone.

```
{
    "explanations": {
        "kernel_shap": {
            "label0": {
                "expected_value" :10.290688514709473,
                "global_shap_values": {
                    "diameter" :0.6856910187882492,
                    "height" :0.4415323937124035,
                    "length" :0.21507476107609084,
                    "sex" :0.448611774505744,
                    "shell" :1.70426496893776,
                    "shucked" :2.1181392924386994,
                    "viscera" :0.342220754059912,
                    "whole" :0.6711906974084011
                }
            }
        }
    },
    "version" :"1.0"
};
```

3. Verwenden Sie die folgende Abfrage, um den Prozentsatz korrekter Voraussagen für Abalonen zu berechnen, die das Modell für noch nicht ausgereifte Abalonen trifft. Unreife Abalonen haben 10 Ringe oder weniger und eine korrekte Voraussage ist auf einen Ring Abweichung von der tatsächlichen Anzahl der Ringe genau.

```
SELECT
    TRUNC(
        SUM(
            CASE
                WHEN ROUND(
                    f_abalone_ring_prediction(
                        Sex,
                        Length,
                        Diameter,
                        Height,
                        Whole,
                        Shucked,
                        Viscera,
                        Shell
                    ),
                    0
```

```
                ) BETWEEN Rings - 1
                AND Rings + 1 THEN 1
                ELSE 0
            END
        ) / CAST(COUNT(SHELL) AS FLOAT),
        4
    ) AS prediction_pct
FROM
    abalone_validation
WHERE
    Rings <= 10;
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten
- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Tutorial: Erstellen von Mehrklassen-Klassifizierungsmodellen mit linearem Lernen

In diesem Tutorial erstellen Sie ein lineares Lernmodell mit Daten aus Amazon S3 und führen dann mithilfe von Amazon Redshift ML Voraussageabfragen mit dem Modell aus. Der SageMaker lineare Lernalgorithmus löst entweder Regressions- oder Klassifizierungsprobleme. Weitere Informationen zu Regressions- und Mehrklassen-Klassifizierungsproblemen finden Sie unter Problemtypen für die Machine-Learning-Parameter im Amazon- SageMaker Entwicklerhandbuch. In diesem Tutorial lösen Sie ein Mehrklassen-Klassifizierungsproblem. Der Algorithmus für lineares Lernen trainiert viele Modelle parallel und ermittelt automatisch das am besten optimierte Modell. Sie verwenden die Operation CREATE MODEL in Amazon Redshift, die Ihr lineares Lernmodell mit erstellt SageMaker und die Prognosefunktion an Amazon Redshift sendet. Weitere Informationen

zum Algorithmus für lineares Lernen finden Sie unter Algorithmus für lineares Lernen im Amazon-SageMaker Entwicklerhandbuch.

Sie können einen CREATE-MODEL-Befehl verwenden, um Trainingsdaten zu exportieren, ein Modell zu trainieren, das Modell zu importieren und eine Amazon-Redshift-Prognosefunktion vorzubereiten. Verwenden Sie die Operation CREATE MODEL, um Trainingsdaten entweder als Tabelle oder SELECT-Anweisung anzugeben.

Lineare Lernmodelle optimieren entweder kontinuierliche oder diskrete Ziele. Kontinuierliche Ziele werden für die Regression verwendet, während diskrete Variablen für die Klassifizierung genutzt werden. Einige Methoden, wie eine Regressionsmethode, bieten eine Lösung nur für kontinuierliche Ziele. Der Algorithmus für lineares Lernen bietet eine Steigerung der Geschwindigkeit gegenüber naiven Hyperparameter-Optimierungstechniken wie der Naive-Bayes-Technik. Eine naive Optimierungstechnik geht davon aus, dass jede Eingabevariable unabhängig ist. Der Algorithmus für lineares Lernen trainiert viele Modelle parallel und wählt das am besten optimierte Modell aus. Ein ähnlicher Algorithmus ist XGBoost, der Schätzungen aus einer Reihe einfacherer und schwächerer Modelle kombiniert, um Voraussagen zu treffen. Weitere Informationen zu XGBoost finden Sie unter XGBoost-Algorithmus im Amazon- SageMaker Entwicklerhandbuch.

Um den Algorithmus für lineares Lernen verwenden zu können, müssen Sie Spalten, die die Dimensionen der Eingaben darstellen, und Zeilen, die Beobachtungen repräsentieren, bereitstellen. Weitere Informationen zum Algorithmus für lineares Lernen finden Sie im Algorithmus für lineares Lernen im Amazon- SageMaker Entwicklerhandbuch.

In diesem Tutorial erstellen Sie ein lineares Lernmodell, das die Art der Bewaldung für ein bestimmtes Gebiet voraussagen kann. Sie verwenden den Befehl CREATE MODEL für den Covertype-Datensatz aus dem UCI Machine Learning Repository. Anschließend verwenden Sie die durch den Befehl erstellte Voraussagefunktion, um die Bewaldungsarten in einem Wildnisgebiet zu bestimmen. Ein Bewaldungstyp ist normalerweise eine Baumart. Zu den Eingaben, die Redshift ML zum Erstellen des Modells verwendet, gehören der Bodentyp, die Entfernung zu Straßen und die Bezeichnung des Wildnisgebiets. Weitere Informationen zu dem Datensatz finden Sie im Covertype-Datensatz im UCI Machine Learning Repository.

Beispielanwendungsfälle

Mit dem linearen Lernmodell können Sie weitere Mehrklassen-Klassifizierungsprobleme lösen, z. B. die Pflanzenart anhand eines Bildes voraussagen. Sie können auch die Menge eines Produkts prognostizieren, die ein Kunde kaufen wird.

Aufgaben

- Voraussetzungen
- Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift
- Schritt 2: Erstellen des Machine-Learning-Modells
- Schritt 3: Validieren des Modells

Voraussetzungen

Zum Durchführen dieses Tutorials müssen Sie die administrative Einrichtung für Amazon Redshift ML abschließen.

Schritt 1: Laden von Daten aus Amazon S3 in Amazon Redshift

Verwenden Sie den Abfrage-Editor v2 von Amazon Redshift, um die folgenden Abfragen auszuführen. Diese Abfragen laden die Beispieldaten in Redshift und teilen die Daten in einen Trainingssatz und einen Validierungssatz auf.

1. Mit der folgenden Abfrage wird die Tabelle `covertype_data` erstellt.

```
CREATE TABLE public.covertype_data (
    elevation bigint ENCODE az64,
    aspect bigint ENCODE az64,
    slope bigint ENCODE az64,
    horizontal_distance_to_hydrology bigint ENCODE az64,
    vertical_distance_to_hydrology bigint ENCODE az64,
    horizontal_distance_to_roadways bigint ENCODE az64,
    hillshade_9am bigint ENCODE az64,
    hillshade_noon bigint ENCODE az64,
    hillshade_3pm bigint ENCODE az64,
    horizontal_distance_to_fire_points bigint ENCODE az64,
    wilderness_area1 bigint ENCODE az64,
    wilderness_area2 bigint ENCODE az64,
    wilderness_area3 bigint ENCODE az64,
    wilderness_area4 bigint ENCODE az64,
    soil_type1 bigint ENCODE az64,
    soil_type2 bigint ENCODE az64,
    soil_type3 bigint ENCODE az64,
    soil_type4 bigint ENCODE az64,
    soil_type5 bigint ENCODE az64,
    soil_type6 bigint ENCODE az64,
    soil_type7 bigint ENCODE az64,
    soil_type8 bigint ENCODE az64,
```

```
    soil_type9 bigint ENCODE az64,
    soil_type10 bigint ENCODE az64,
    soil_type11 bigint ENCODE az64,
    soil_type12 bigint ENCODE az64,
    soil_type13 bigint ENCODE az64,
    soil_type14 bigint ENCODE az64,
    soil_type15 bigint ENCODE az64,
    soil_type16 bigint ENCODE az64,
    soil_type17 bigint ENCODE az64,
    soil_type18 bigint ENCODE az64,
    soil_type19 bigint ENCODE az64,
    soil_type20 bigint ENCODE az64,
    soil_type21 bigint ENCODE az64,
    soil_type22 bigint ENCODE az64,
    soil_type23 bigint ENCODE az64,
    soil_type24 bigint ENCODE az64,
    soil_type25 bigint ENCODE az64,
    soil_type26 bigint ENCODE az64,
    soil_type27 bigint ENCODE az64,
    soil_type28 bigint ENCODE az64,
    soil_type29 bigint ENCODE az64,
    soil_type30 bigint ENCODE az64,
    soil_type31 bigint ENCODE az64,
    soil_type32 bigint ENCODE az64,
    soil_type33 bigint ENCODE az64,
    soil_type34 bigint ENCODE az64,
    soil_type35 bigint ENCODE az64,
    soil_type36 bigint ENCODE az64,
    soil_type37 bigint ENCODE az64,
    soil_type38 bigint ENCODE az64,
    soil_type39 bigint ENCODE az64,
    soil_type40 bigint ENCODE az64,
    cover_type bigint ENCODE az64
) DISTSTYLE AUTO;
```

2. Die folgende Abfrage kopiert die Beispieldaten aus dem Covertype-Datensatz in Amazon S3 in die Tabelle `covertype_data`, die Sie zuvor in Amazon Redshift erstellt haben.

```
COPY public.covertype_data
FROM
    's3://redshift-ml-multiclass/covtype.data.gz' IAM_ROLE DEFAULT gzip DELIMITER ','
 REGION 'us-east-1';
```

3. Durch manuelles Aufteilen der Daten können Sie die Genauigkeit des Modells überprüfen, indem Sie einen zusätzlichen Testsatz zuweisen. Die folgende Abfrage teilt die Daten in drei Sätze auf. Die Tabelle `covertype_training` ist für das Training, die Tabelle `covertype_validation` zur Validierung und die Tabelle `covertype_test` zum Testen Ihres Modells bestimmt. Sie verwenden den Trainingsatz, um Ihr Modell zu trainieren, und den Validierungssatz, um die Entwicklung des Modells zu validieren. Anschließend verwenden Sie den Testsatz, um die Leistung des Modells zu testen und festzustellen, ob das Modell den Datensatz über- oder unterpasst.

```
CREATE TABLE public.covertype_data_prep AS
SELECT
    a.*,
    CAST (random() * 100 AS int) AS data_group_id
FROM
    public.covertype_data a;

--training dataset
 CREATE TABLE public.covertype_training as
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id < 80;

--validation dataset
 CREATE TABLE public.covertype_validation AS
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id BETWEEN 80
    AND 89;

--test dataset
 CREATE TABLE public.covertype_test AS
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
```

```
    data_group_id > 89;
```

Schritt 2: Erstellen des Machine-Learning-Modells

In diesem Schritt verwenden Sie die Anweisung CREATE MODEL, um Ihr Machine-Learning-Modell mit dem linearen Lernalgorithmus zu erstellen.

Die folgende Abfrage erstellt das lineare Lernmodell mit der Operation CREATE MODEL unter Verwendung Ihres S3-Buckets. Ersetzen Sie *DOC-EXAMPLE-BUCKET* durch Ihren eigenen S3-Bucket.

```
CREATE MODEL forest_cover_type_model
FROM
    (
        SELECT
            Elevation,
            Aspect,
            Slope,
            Horizontal_distance_to_hydrology,
            Vertical_distance_to_hydrology,
            Horizontal_distance_to_roadways,
            HIllshade_9am,
            Hillshade_noon,
            Hillshade_3pm,
            Horizontal_Distance_To_Fire_Points,
            Wilderness_Area1,
            Wilderness_Area2,
            Wilderness_Area3,
            Wilderness_Area4,
            soil_type1,
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
```

```
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40,
            Cover_type
        from
            public.covertype_training
    ) TARGET cover_type FUNCTION predict_cover_type IAM_ROLE default MODEL_TYPE
LINEAR_LEARNER PROBLEM_TYPE MULTICLASS_CLASSIFICATION OBJECTIVE 'Accuracy' SETTINGS (
        S3_BUCKET '<DOC-EXAMPLE-BUCKET>',
        S3_GARBAGE_COLLECT OFF,
        MAX_RUNTIME 15000
    );
```

Anzeigen des Status des Modelltrainings (optional)

Sie können den Befehl SHOW MODEL verwenden, um festzustellen, wann Ihr Modell bereit ist.

Verwenden Sie die folgende Abfrage, um den Trainingsfortschritt des Modells zu überwachen.

```
SHOW MODEL forest_cover_type_model;
```

Wenn das Modell bereit ist, sollte die Ausgabe der vorherigen Operation dem folgenden Beispiel ähneln. Beachten Sie, dass die Ausgabe die Metrik `validation:multiclass_accuracy` bereitstellt, die Sie auf der rechten Seite des folgenden Beispiels sehen können. Die Mehrklassen-Genauigkeit misst den Prozentsatz der Datenpunkte, die vom Modell korrekt klassifiziert werden. Sie verwenden die Mehrklassen-Genauigkeit, um die Genauigkeit des Modells im nächsten Schritt zu überprüfen.

```
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
|              Key               |




 Value




                                                          |
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
| Model Name                     | forest_cover_type_model

                                                          |
| Schema Name                    | public

                                                          |
```

```
| Owner                          | awsuser
                                                                  |
| Creation Time                  | Tue, 12.07.2022 20:24:32
                                                                  |
| Model State                    | READY
                                                                  |
| validation:multiclass_accuracy |
                                                 0.724952 |
| Estimated Cost                 |
```

```
                                        5.341750 |
|                              |

                                                 |
| TRAINING DATA:               |

                                                 |
| Query                        | SELECT ELEVATION, ASPECT, SLOPE,
 HORIZONTAL_DISTANCE_TO_HYDROLOGY, VERTICAL_DISTANCE_TO_HYDROLOGY,
 HORIZONTAL_DISTANCE_TO_ROADWAYS, HILLSHADE_9AM, HILLSHADE_NOON, HILLSHADE_3PM ,
 HORIZONTAL_DISTANCE_TO_FIRE_POINTS, WILDERNESS_AREA1, WILDERNESS_AREA2,
 WILDERNESS_AREA3, WILDERNESS_AREA4, SOIL_TYPE1, SOIL_TYPE2, SOIL_TYPE3, SOIL_TYPE4,
 SOIL_TYPE5, SOIL_TYPE6, SOIL_TYPE7, SOIL_TYPE8, SOIL_TYPE9, SOIL_TYPE10 , SOIL_TYPE11,
 SOIL_TYPE12 , SOIL_TYPE13 , SOIL_TYPE14, SOIL_TYPE15, SOIL_TYPE16, SOIL_TYPE17,
 SOIL_TYPE18, SOIL_TYPE19, SOIL_TYPE20, SOIL_TYPE21, SOIL_TYPE22, SOIL_TYPE23,
 SOIL_TYPE24, SOIL_TYPE25, SOIL_TYPE26, SOIL_TYPE27, SOIL_TYPE28, SOIL_TYPE29,
 SOIL_TYPE30, SOIL_TYPE31, SOIL_TYPE32, SOIL_TYPE33, SOIL_TYPE34, SOIL_TYPE36,
 SOIL_TYPE37, SOIL_TYPE38, SOIL_TYPE39, SOIL_TYPE40, COVER_TYPE |
|                              | FROM PUBLIC.COVERTYPE_TRAINING
```

```
                                                    |
| Target Column                  | COVER_TYPE

                                                    |
|                                |

                                                    |
| PARAMETERS:                    |

                                                    |
| Model Type                     | linear_learner

                                                    |
```

```
| Problem Type                    | MulticlassClassification

                                                          |
| Objective                       | Accuracy

                                                          |
| AutoML Job Name                 | redshiftml-20220712202432187659

                                                          |
| Function Name                   | predict_cover_type

                                                          |
| Function Parameters             | elevation aspect slope
 horizontal_distance_to_hydrology vertical_distance_to_hydrology
 horizontal_distance_to_roadways hillshade_9am hillshade_noon hillshade_3pm
 horizontal_distance_to_fire_points wilderness_area1 wilderness_area2 wilderness_area3
```

```
 wilderness_area4 soil_type1 soil_type2 soil_type3 soil_type4 soil_type5 soil_type6
 soil_type7 soil_type8 soil_type9 soil_type10 soil_type11 soil_type12 soil_type13
 soil_type14 soil_type15 soil_type16 soil_type17 soil_type18 soil_type19 soil_type20
 soil_type21 soil_type22 soil_type23 soil_type24 soil_type25 soil_type26 soil_type27
 soil_type28 soil_type29 soil_type30 soil_type31 soil_type32 soil_type33 soil_type34
 soil_type36 soil_type37 soil_type38 soil_type39 soil_type40
                                                 |
| Function Parameter Types       | int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8
                                                                    |
| IAM Role                       | default-aws-iam-role
                                                 |
| S3 Bucket                      | DOC-EXAMPLE-BUCKET
                                                 |
| Max Runtime                    |
```

```
                                          15000 |
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
```

Schritt 3: Validieren des Modells

1. Die folgende Voraussageabfrage validiert die Genauigkeit des Modells für den Datensatz `covertype_validation` durch Berechnung der Mehrklassen-Genauigkeit. Die Mehrklassen-Genauigkeit ist der Prozentsatz der korrekten Voraussagen des Modells.

```
SELECT
    CAST(sum(t1.match) AS decimal(7, 2)) AS predicted_matches,
    CAST(sum(t1.nonmatch) AS decimal(7, 2)) AS predicted_non_matches,
    CAST(sum(t1.match + t1.nonmatch) AS decimal(7, 2)) AS total_predictions,
    predicted_matches / total_predictions AS pct_accuracy
FROM
    (
        SELECT
            Elevation,
            Aspect,
            Slope,
            Horizontal_distance_to_hydrology,
            Vertical_distance_to_hydrology,
            Horizontal_distance_to_roadways,
            HIllshade_9am,
            Hillshade_noon,
            Hillshade_3pm,
            Horizontal_Distance_To_Fire_Points,
            Wilderness_Area1,
            Wilderness_Area2,
            Wilderness_Area3,
            Wilderness_Area4,
            soil_type1,
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
```

```
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40,
            Cover_type AS actual_cover_type,
            predict_cover_type(
                Elevation,
                Aspect,
                Slope,
                Horizontal_distance_to_hydrology,
                Vertical_distance_to_hydrology,
                Horizontal_distance_to_roadways,
                HIllshade_9am,
                Hillshade_noon,
                Hillshade_3pm,
                Horizontal_Distance_To_Fire_Points,
                Wilderness_Area1,
```

```
            Wilderness_Area2,
            Wilderness_Area3,
            Wilderness_Area4,
            soil_type1,
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40
        ) AS predicted_cover_type,
        CASE
```

```
                WHEN actual_cover_type = predicted_cover_type THEN 1
                ELSE 0
            END AS match,
            CASE
                WHEN actual_cover_type <> predicted_cover_type THEN 1
                ELSE 0
            END AS nonmatch
        FROM
            public.covertype_validation
    ) t1;
```

Die Ausgabe der vorherigen Abfrage sollte dem folgenden Beispiel ähneln. Der Wert der Mehrklassen-Genauigkeitsmetrik sollte ähnlich wie der Wert der Metrik `validation:multiclass_accuracy` sein, die in der Ausgabe der Operation SHOW MODEL angezeigt wird.

```
+------------------+----------------------+------------------+--------------+
| predicted_matches | predicted_non_matches | total_predictions | pct_accuracy |
+------------------+----------------------+------------------+--------------+
|             41211 |                16324 |             57535 |   0.71627704 |
+------------------+----------------------+------------------+--------------+
```

2. Die folgende Abfrage prognostiziert die gängigste Bewaldungsart für `wilderness_area2`. Dieser Datensatz umfasst vier Wildnisgebiete und sieben Bewaldungsarten. Ein Wildnisgebiet kann mehrere Bewaldungsarten aufweisen.

```
SELECT t1. predicted_cover_type, COUNT(*)
FROM
(
SELECT
   Elevation,
   Aspect,
   Slope,
   Horizontal_distance_to_hydrology,
   Vertical_distance_to_hydrology,
   Horizontal_distance_to_roadways,
   HIllshade_9am,
   Hillshade_noon,
   Hillshade_3pm ,
   Horizontal_Distance_To_Fire_Points,
   Wilderness_Area1,
   Wilderness_Area2,
```

```
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10 ,
    Soil_Type11,
    Soil_Type12 ,
    Soil_Type13 ,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
    Soil_Type32,
    Soil_Type33,
    Soil_Type34,
    Soil_Type36,
    Soil_Type37,
    Soil_Type38,
    Soil_Type39,
    Soil_Type40,
    predict_cover_type( Elevation,
    Aspect,
    Slope,
```

```
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10,
    Soil_Type11,
    Soil_Type12,
    Soil_Type13,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
    Soil_Type32,
    Soil_Type33,
```

```
   Soil_Type34,
   Soil_Type36,
   Soil_Type37,
   Soil_Type38,
   Soil_Type39,
   Soil_Type40) AS predicted_cover_type

FROM public.covertype_test
WHERE wilderness_area2 = 1)
t1
GROUP BY 1;
```

Die Ausgabe der vorherigen Operation sollte dem folgenden Beispiel ähneln. Diese Ausgabe bedeutet, dass das Modell vorausgesagt hat, dass der Großteil der Bewaldung Bewaldungsart 1 ist und die Bewaldung in einigen Gebieten den Bewaldungsarten 2 und 7 entspricht.

```
+----------------------+-------+
| predicted_cover_type | count |
+----------------------+-------+
|                    2 |   564 |
|                    7 |    97 |
|                    1 |  2309 |
+----------------------+-------+
```

3. Die folgende Abfrage zeigt die häufigste Bewaldungsart in einem einzelnen Wildnisgebiet. Die Abfrage zeigt das Aufkommen dieser Bewaldungsart und das Wildnisgebiet der Bewaldungsart an.

```
SELECT t1. predicted_cover_type, COUNT(*), wilderness_area
FROM
(
SELECT
   Elevation,
   Aspect,
   Slope,
   Horizontal_distance_to_hydrology,
   Vertical_distance_to_hydrology,
   Horizontal_distance_to_roadways,
   HIllshade_9am,
   Hillshade_noon,
   Hillshade_3pm ,
   Horizontal_Distance_To_Fire_Points,
   Wilderness_Area1,
```

```
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10 ,
    Soil_Type11,
    Soil_Type12 ,
    Soil_Type13 ,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
    Soil_Type32,
    Soil_Type33,
    Soil_Type34,
    Soil_Type36,
    Soil_Type37,
    Soil_Type38,
    Soil_Type39,
    Soil_Type40,
    predict_cover_type( Elevation,
    Aspect,
```

```
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10,
    Soil_Type11,
    Soil_Type12,
    Soil_Type13,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
    Soil_Type32,
```

```
   Soil_Type33,
   Soil_Type34,
   Soil_Type36,
   Soil_Type37,
   Soil_Type38,
   Soil_Type39,
   Soil_Type40) AS predicted_cover_type,
   CASE WHEN Wilderness_Area1 = 1 THEN 1
        WHEN Wilderness_Area2 = 1 THEN 2
        WHEN Wilderness_Area3 = 1 THEN 3
        WHEN Wilderness_Area4 = 1 THEN 4
        ELSE 0
   END AS wilderness_area

FROM public.covertype_test)
t1
GROUP BY 1, 3
ORDER BY 2 DESC
LIMIT 1;
```

Die Ausgabe der vorherigen Operation sollte dem folgenden Beispiel ähneln.

```
+----------------------+-------+-----------------+
| predicted_cover_type | count | wilderness_area |
+----------------------+-------+-----------------+
|                    2 | 15738 |               1 |
+----------------------+-------+-----------------+
```

Verwandte Themen

Weitere Informationen zu Amazon Redshift ML finden Sie in der folgenden Dokumentation:

- Kosten für die Verwendung von Amazon Redshift ML
- Operation CREATE MODEL
- Funktion EXPLAIN_MODEL

Weitere Informationen über Machine Learning finden Sie in der folgenden Dokumentation:

- Übersicht zum Machine Learning
- Machine Learning für Anfänger und Experten

- Was ist Fairness und Modellerklärbarkeit für Prognosen mit Machine Learning?

Optimieren der Abfrageleistung

Amazon Redshift verwendet zur Interaktion mit Daten und Objekten im System Abfragen auf der Basis von SQL (Structured Query Language). DML (Data Manipulation Language) ist die Teilmenge von SQL, mit der Daten angezeigt, hinzugefügt, geändert und gelöscht werden. DDL (Data Definition Language) ist die Teilmenge von SQL, mit der Datenbankobjekte wie Tabellen und Sichten angezeigt, hinzugefügt, geändert und gelöscht werden.

Wenn Ihr System eingerichtet ist, arbeiten Sie normalerweise vor allem mit DML, und insbesondere mit dem Befehl SELECT zum Abrufen und Anzeigen von Daten. Um effiziente Abfragen zum Abrufen von Daten in Amazon Redshift zu erstellen, müssen Sie sich mit SELECT vertraut machen und die Tipps in Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift für die Maximierung der Abfrageeffizienz anwenden.

Informationen zur Verarbeitung von Abfragen durch Amazon Redshift finden Sie in den Abschnitten Verarbeitung von Abfragen und Analysieren und Verbessern von Abfragen. Mithilfe dieser Informationen und geeigneten Diagnosetools lassen sich Leistungsprobleme bei Abfragen identifizieren und entfernen.

Hinweise zur Identifizierung und Behandlung einiger der häufigsten und schwerwiegendsten Probleme, die bei Amazon-Redshift-Abfragen auftreten können, finden Sie im Abschnitt Fehlerbehebung bei Abfragen.

Themen

Verarbeitung von Abfragen

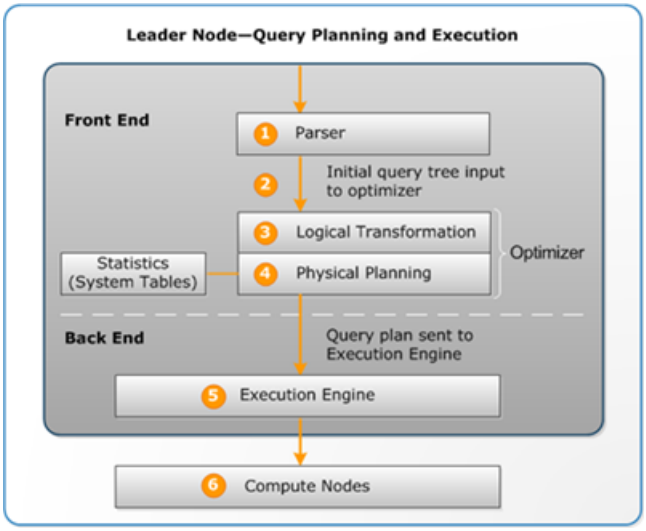
Amazon Redshift leitet eingegebene SQL-Abfragen an den Parser und den Optimierer weiter, der einen Abfrageplan erstellt. Anschließend übersetzt die Ausführungs-Engine den Abfrageplan in einen ausführbaren Code und sendet diesen zur Ausführung an die Verarbeitungsknoten.

Themen

- Abfrageplan
- Übersicht über die Schritte des Abfrageplans
- Für die Abfrageleistung relevante Faktoren

Workflow der Abfrageplanung und -ausführung

Die folgende Abbildung gibt eine allgemeine Übersicht über den Workflow der Abfrageplanung und -ausführung.



Der Workflow der Abfrageplanung und -ausführung folgt diesen Schritten:

1. Die Abfrage wird an den Führungsknoten übermittelt und dort geparst.
2. Der Parser erzeugt eine erste Version einer Abfragestruktur, die eine logische Repräsentation der ursprünglichen Abfrage ist. Amazon Redshift gibt dann diese Abfragestruktur in den Abfrageoptimierer ein.
3. Der Optimierer bewertet die Abfragestruktur und transformiert sie ggf., um sie maximal effizient zu machen. Eine solche Transformation zur Optimierung der Effizienz kann beispielsweise so aussehen, dass anstelle einer Abfrage mehrere zusammengehörige Abfragen ausgeführt werden.
4. Der Optimierer erzeugt für diese transformierten Abfragestrukturen einen Abfrageplan, in dem der Ausführungsablauf mit der besten Ausführungsperformance. In dem Abfrageplan sind bestimmte Ausführungsoptionen angegeben, über die beispielsweise Art und Reihenfolge von Join-Operationen, Aggregierungsoptionen und Anforderungen an die Datenverteilung gesteuert werden.

Sie können den Befehl EXPLAIN verwenden, um den Abfrageplan anzuzeigen. Der Abfrageplan ist ein grundlegendes Werkzeug für die Analyse und Optimierung komplexer Abfragen. Weitere Informationen finden Sie unter Abfrageplan.

5. Die Ausführungs-Engine übersetzt den Abfrageplan in Schritte, Segmente und Streams:

   Schritt

   Ein Schritt ist eine Einzeloperation, die zur Ausführung der Abfrage durchgeführt werden muss. Schritte können zusammengefasst werden und ermöglichen so den Verarbeitungsknoten, Abfragen, Join- und andere Datenbankoperationen auszuführen.

   Segment

   Eine Kombination von Schritten, der von einem einzelnen Prozess verarbeitet werden kann; Ein Segment ist auch die kleinste Kompilierungseinheit, die von einem Verarbeitungsknoten in einer Slice verarbeitet werden kann. Ein Slice ist die Parallelverarbeitungseinheit in Amazon Redshift. Die Segmente in einem Stream werden parallel ausgeführt.

   Stream

   Eine Sammlung von Segmenten, die an die verfügbaren Slices in Verarbeitungsknoten übermittelt werden.

   Die Ausführungs-Engine erzeugt anhand der Schritte, Segmente und Streams kompilierten Code. Kompilierter Code wird schneller ausgeführt als interpretierter Code und verbraucht auch weniger Rechenkapazität. Anschließend wird dieser kompilierte Code an alle Verarbeitungsknoten gesendet.

   > **Note**
   >
   > Wenn Sie die Abfragen anhand verschiedener Benchmark-Metriken testen, sollten Sie jeweils die zweite Ausführung der Abfrage messen, da bei der ersten Ausführung der Verwaltungsaufwand zum Kompilieren des Codes mitgezählt wird. Weitere Informationen finden Sie unter Für die Abfrageleistung relevante Faktoren.

6. Die Abfragesegmente werden in den Slices der Verarbeitungsknoten parallel ausgeführt. Bei diesem Vorgang nutzt Amazon Redshift eine optimierte Netzwerkkommunikation und optimiertes Speicher- und Datenträgermanagement, um Zwischenergebnisse von einem Schritt des Abfrageplans in den nächsten zu übernehmen. Dadurch lässt sich auch die Abfrageausführung beschleunigen.

Die Schritte 5 und 6 erfolgen für jeden Stream getrennt. Die Engine erstellt die ausführbaren Segmente für einen Stream sendet sie an die Verarbeitungsknoten. Wenn die Segmente eines Streams vollständig erzeugt sind, generiert die Engine die Segmente für den nächsten Stream. Dies ermöglicht der Engine eine Analyse des jeweiligen vorangehenden Streams, beispielsweise, ob die Operationen datenträgerbasiert waren, und eine Optimierung der Generierung der Segmente für den nächsten Stream.

Wenn die Verarbeitungsknoten die Berechnungen abgeschlossen haben, senden Sie die Abfrageergebnisse zur Weiterverarbeitung an den Führungsknoten zurück. Der Führungsknoten fügt die Daten zu einem Ergebnissatz zusammen und sortiert und gruppiert das Gesamtergebnis. Dann sendet der Führungsknoten das Ergebnis an den Client zurück.

> **Note**
>
> Die Verarbeitungsknoten können während der Ausführung der Abfrage bei Bedarf bestimmte Daten an den Führungsknoten zurücksenden. Wenn beispielsweise in einer Unterabfrage eine LIMIT-Klausel vorkommt, wird der Grenzwert auf dem Führungsknoten angewendet, bevor die Daten zur Weiterverarbeitung an den Cluster verteilt werden.

Abfrageplan

Sie können den Abfrageplan verwenden, um Informationen über die Einzeloperationen zu erhalten, die zur Ausführung einer Abfrage benötigt werden. Wir empfehlen, sich zunächst mit den Grundlagen der Verarbeitung von Abfragen und der Erstellung von Abfrageplänen durch Amazon Redshift vertraut zu machen, bevor Sie mit Abfrageplänen arbeiten. Weitere Informationen finden Sie unter Workflow der Abfrageplanung und -ausführung.

Um einen Abfrageplan zu erstellen, führen Sie einen EXPLAIN-Befehl, gefolgt von der Abfragetext, aus. Der Abfrageplan enthält die folgenden Informationen:

- Von unten nach oben gelesen erhalten Sie die Operationen, die die Ausführungs-Engine durchführt.
- Welche Art von Schritten jede Operation ausführt
- Für jede Operation die verwendeten Tabellen und Spalten
- Für jede Operation, welches Datenvolumen verarbeitet wird (als Anzahl Spalten, sowie die Datenbreite in Bytes)

- Die relativen Kosten der Operation. Die Kosten sind ein Maß für den Vergleich der relativen Ausführungsdauer der Schritte in einem Plan. Die Kosten geben keinen Aufschluss über die tatsächliche Ausführungsdauer oder den Speicherverbrauch und ermöglichen auch keinen Vergleich zwischen Ausführungsplänen. Die Kosten weisen vielmehr darauf hin, welche Operationen in einer Abfrage die meisten Ressourcen verbrauchen.

Der Befehl EXPLAIN führt die Abfrage nicht aus. Der Befehl zeigt lediglich den Plan an, den Amazon Redshift ausführt, wenn die Abfrage unter den aktuellen Betriebsbedingungen ausgeführt wird. Wenn Sie für eine Tabelle das Schema oder Daten ändern und dann erneut ANALYZE ausführen, um die statistischen Metadaten zu aktualisieren, sieht der Abfrageplan möglicherweise anders aus.

Der von EXPLAIN ausgegebene Abfrageplan ist eine vereinfachte Übersicht der Abfrageausführung. Die Details zur parallelen Verarbeitung der Abfragen werden nicht angezeigt. Wenn Sie alle Informationen detailliert anzeigen möchten, müssen Sie die Abfrage ausführen und dann die Zusammenfassung der Abfrage über die Ansicht SVL_QUERY_SUMMARY oder SVL_QUERY_REPORT abrufen. Weitere Informationen zur Verwendung dieser Ansichten finden Sie unter Analysieren des Abfragezusammenfassung.

In dem folgenden Beispiel ist die Ausgabe von EXPLAIN für eine einfache GROUP BY-Abfrage über der Tabelle EVENT dargestellt:

```
explain select eventname, count(*) from event group by eventname;

                            QUERY PLAN
------------------------------------------------------------------------
XN HashAggregate  (cost=131.97..133.41 rows=576 width=17)
  ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=17)
```

EXPLAIN gibt für jede Operation die folgenden Metriken zurück:

Kosten

Ein relativer Wert, mit dem sich die Operationen in einem Plan vergleichen lassen. Die kosten werden repräsentiert als zwei Dezimalzahlen, die durch zwei Punkte getrennt sind, beispielsweise `cost=131.97..133.41`. Der erste Wert (in diesem Fall 131,97) gibt die relativen Kosten für die Rückgabe der ersten Zeile für diese Operation an. Der zweite Wert (in diesem Fall 133,41) gibt die relativen Kosten für die Ausführung der gesamten Operation an. Die Kosten im Abfrageplan sind beim Lesen des Plans kumulativ, sodass die HashAggregate Kosten in diesem Beispiel (131.97...133.41) die Kosten des darunterliegenden Seq Scans beinhalten (0.00...87.98).

Rows

Die geschätzte Anzahl der zurückzugebenden Zeilen. In diesem Beispiel wird erwartet, dass der Scan 8798 Zeilen zurückgibt. Es wird erwartet, dass der HashAggregate Operator selbst 576 Zeilen zurückgibt (nachdem doppelte Ereignisnamen aus der Ergebnismenge verworfen wurden).

> **Note**
>
> Die Schätzung der zurückgegebenen Zeilen basiert auf der durch den Befehl ANALYZE generierten Statistik. Wenn ANALYZE längere Zeit nicht ausgeführt wurde, sind die Schätzungen weniger zuverlässig.

Width

Die geschätzte Breite der Zeilen, in Bytes. In diesem Beispiel wird eine durchschnittliche Zeilenbreite von 17 Bytes erwartet.

EXPLAIN-Operatoren

In diesem Abschnitt werden kurz die verschiedenen Operatoren beschrieben, die von EXPLAIN am häufigsten ausgegeben werden. Eine vollständige Liste der Operatoren finden Sie unter EXPLAIN im Abschnitt „SQL-Befehle“.

Sequenzieller Scan-Operator

Der sequenzielle Scan-Operator (Seq Scan) scannt eine Tabelle. Seq Scan scannt alle Spalten in der Tabelle sequenziell von Anfang bis Ende und bewertet für jede Zeile die Bedingungen der Abfrage in der WHERE-Klausel.

Join-Operatoren

Amazon Redshift wählt Join-Operatoren abhängig vom physischen Design der Tabellen, für die die Join-Operation ausgeführt wird, vom Speicherort der Daten, die für den Join benötigt werden, und von den spezifischen Anforderungen der Abfrage selbst aus.

- Nested Loop

  Dies ist der am wenigsten optimale Join; wird vor allem für Kreuz-Joins (kartesische Produkte) und einige Ungleichheits-Joins verwendet.

- Hash-Join und Hash

  Hash-Joins und Hashes werden in der Regel schneller ausgeführt als ein Join über eine verschachtelte Schleife, und werden für innere linke Joins sowie für äußere linke und rechte Joins verwendet. Diese Operatoren werden bei Join-Operationen für Tabellen verwendet, bei denen die Join-Spalten nicht sowohl Verteilungsschlüssel als auch Sortierschlüssel sind. Hash-Joins erstellen die Hash-Tabelle für die innere Tabelle des Verbunds. Der Hash-Join-Operator liest die äußere Tabelle, erstellt einen Hash für die Spalte, über die verbunden wird, durchsucht die innere Hash-Tabelle nach Übereinstimmungen.

- Merge Join

  In der Regel die schnellste Join-Variante, wird für innere und äußere Joins verwendet. Der Zusammenführungs-Join ermöglicht nicht die Erstellung eines vollständigen Joins. Der Operator wird bei Join-Operationen für Tabellen verwendet, bei denen die Join-Spalten sowohl Verteilungsschlüssel als auch Sortierschlüssel sind und weniger als 20 % der Tabellen für die Join-Operation unsortiert sind. Bei dieser Operation werden zwei sortierte Tabellen sequentiell eingelesen und die übereinstimmenden Zeilen gesucht. Um den Prozentsatz unsortierter Zeilen anzuzeigen, führen Sie eine Abfrage über der Systemtabelle SVV_TABLE_INFO aus.

- Räumlicher Join

  In der Regel ein schneller Join, der auf der Nähe von räumlichen Daten basiert und für die Datentypen `GEOMETRY` und `GEOGRAPHY` verwendet wird.

Aggregat-Operatoren

In Abfrageplänen werden die folgenden Operatoren bei Abfragen verwendet, die Aggregat-Funktionen und GROUP BY-Operationen verwenden.

- Aggregate

  Operator für skalare Aggregat-Funktionen wie AVG oder SUM.

- HashAggregate

  Operator für unsortierte, gruppierte Aggregat-Funktionen.

- GroupAggregate

  Operator für sortierte, gruppierte Aggregat-Funktionen.

Sortieroperatoren

In dem Abfrageplan werden die folgenden Operatoren verwendet, wenn in Abfragen Ergebnissätze sortiert oder zusammengeführt werden sollen.

- Sortierung

  Bewertet die ORDER BY-Klausel und andere Sortieroperationen, beispielsweise Sortiervorgänge aufgrund von UNION-Abfragen und Join-Operationen, SELECT DISTINCT-Abfragen und Fensterfunktionen.
- Mischen von

  Erstellt die abschließenden sortierten Ergebnisse auf der Basis zwischenzeitlicher sortierter Ergebnisse, die aus den parallel ausgeführten Operationen abgeleitet werden.

Die Operatoren UNION, INTERSECT und EXCEPT

In Abfrageplänen werden die folgenden Operatoren bei Abfragen verwendet, die Operationen über Mengen (UNION, INTERSECT und EXCEPT) verwenden.

- Unterabfrage

  Wird zur Ausführung von UNION-Abfragen verwendet.
- Hash Intersect Distinct

  Wird für INTERSECT -Abfragen verwendet.
- SetOp Ausnahme

  Wird zur Ausführung von EXCEPT- (bzw. MINUS-)Abfragen verwendet.

Andere Operatoren

Die folgenden Operatoren werden häufig bei Routineabfragen in der Ausgabe von EXPLAIN verwendet.

- Eindeutig

  Beseitigt Duplikate bei SELECT DISTINCT- und UNION-Abfragen.
- Limit

Verarbeitet die LIMIT-Klausel.

- Window

  Führt Fensterfunktionen aus.

- Ergebnis

  Führt skalare Funktionen aus, für die kein Tabellenzugriff erforderlich ist.

- Subplan

  Wird für bestimmte Unterabfragen verwendet.

- Netzwerk

  Sendet Zwischenergebnisse zur weiteren Verarbeitung an den Führungsknoten zurück.

- Materialize

  Speichert Zeilen als Eingabe für Joins mit verschachtelten Schleifen und einige Zusammenführungs-Joins.

Joins in EXPLAIN

Der Abfrageoptimierer verwendet verschiedene Varianten von Join-Operationen, je nachdem, wie die Abfrage und die zugrundeliegenden Tabellen strukturiert sind. In der Ausgabe von EXPLAIN sind die Join-Variante, die verwendeten Tabellen und die Verteilung der Daten über den Cluster aufgeführt, um die Verarbeitung der Abfrage zu beschreiben.

Beispiele für Join-Varianten

Die folgenden Beispiele zeigen die verschiedene Join-Varianten, die dem Abfrageoptimierer zur Verfügung stehen. Die Join-Variante in dem Abfrageplan hängt vom physischen Design der beteiligten Tabellen ab.

Beispiel: Hash-Join zweier Tabellen

Mit der folgenden Abfrage werden EVENT und CATEGORY über die Spalte CATID verbunden. CATID ist der Verteilungsschlüssel und der Sortierschlüssel für CATEGORY, aber nicht für EVENT. Es wird ein Hash-Join mit EVENT als äußerer Tabelle und CATEGORY als innerer Tabelle durchgeführt. Weil CATEGORY die kleinere der beiden Tabellen ist, sendet der Planer bei der

Ausführung der Abfrage mit DS_BCAST_INNER eine Kopie der Tabelle an die Verarbeitungsknoten. Die Kosten für die Join-Operation machen den Großteil der kumulierten Kosten für den Plan aus.

```
explain select * from category, event where category.catid=event.catid;

                              QUERY PLAN
----------------------------------------------------------------------
 XN Hash Join DS_BCAST_INNER  (cost=0.14..6600286.07 rows=8798 width=84)
   Hash Cond: ("outer".catid = "inner".catid)
   ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=35)
   ->  XN Hash  (cost=0.11..0.11 rows=11 width=49)
         ->  XN Seq Scan on category  (cost=0.00..0.11 rows=11 width=49)
```

> **Note**
>
> Wenn in der Ausgabe von EXPLAIN zwei Operatoren gleich weit eingerückt sind, kann dies bedeuten, dass diese Operationen voneinander unabhängig sind und parallel gestartet werden können. In dem vorangehenden Beispiel jedoch ist dies nicht möglich: Zwar befinden sich die Scan-Operation für die Tabelle EVENT und die Hash-Operation auf derselben Ebene, aber der Scan von EVENT kann erst gestartet werden, nachdem die Hash-Operation vollständig abgeschlossen ist.

Beispiel: Zusammenführungs-Join zweier Tabellen

In der folgenden Abfrage wird ebenfalls SELECT * verwendet, aber es wird der Verbund von SALES und LISTING über die Spalte LISTID durchgeführt, und LISTID ist für beide Tabellen sowohl Verteilungs- als auch Sortierschlüssel. Der Optimierer wählt als Join-Variante Merge-Join, und für den Join ist keine Umverteilung von Daten erforderlich (DS_DIST_NONE).

```
explain select * from sales, listing where sales.listid = listing.listid;
QUERY PLAN
------------------------------------------------------------------------------
XN Merge Join DS_DIST_NONE  (cost=0.00..6285.93 rows=172456 width=97)
  Merge Cond: ("outer".listid = "inner".listid)
  ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=44)
  ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=53)
```

Das folgende Beispiele illustriert die verschiedene Join-Varianten in derselben Abfrage. Wie in dem vorangehenden Beispiel werden ebenfalls SALES und LISTING per Zusammenführungs-

Join verbunden, aber die dritte Tabelle, EVENT, muss per Hash-Join mit dem Ergebnis des Zusammenführungs-Joins verbunden werden. Auch hier fallen für den Hash-Join Kosten für die Rundsendung von Daten an.

```
explain select * from sales, listing, event
where sales.listid = listing.listid and sales.eventid = event.eventid;
                                  QUERY PLAN
------------------------------------------------------------------------------
XN Hash Join DS_BCAST_INNER  (cost=109.98..3871130276.17 rows=172456 width=132)
  Hash Cond: ("outer".eventid = "inner".eventid)
  ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6285.93 rows=172456 width=97)
        Merge Cond: ("outer".listid = "inner".listid)
        ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=44)
        ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=53)
  ->  XN Hash  (cost=87.98..87.98 rows=8798 width=35)
        ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=35)
```

Beispiel: Join-, Aggregat- und Sortieroperationen

In der folgenden Abfrage wird ein Hash-Join für die Tabellen SALES und EVENT durchgeführt, gefolgt von Aggregat- und Sortieroperationen für die gruppierte SUM-Funktion und die ORDER BY-Klausel. Der erste Sortieroperator wird parallel auf den Datenverarbeitungsknoten ausgeführt. Anschließend sendet der Network-Operator die Ergebnisse an den Führungsknoten, wo sie mit einer Merge-Operation zu einem sortierten Endergebnis zusammengeführt werden.

```
explain select eventname, sum(pricepaid) from sales, event
where sales.eventid=event.eventid group by eventname
order by 2 desc;
                                            QUERY PLAN
------------------------------------------------------------------------------
 XN Merge  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
  Merge Key: sum(sales.pricepaid)
  ->  XN Network  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
        Send to leader
        ->  XN Sort  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
              Sort Key: sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=2815366577.07..2815366578.51 rows=576
 width=27)
                    ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..2815365714.80
 rows=172456 width=27)
                          Hash Cond: ("outer".eventid = "inner".eventid)
```

```
                        ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456
width=14)
                        ->  XN Hash  (cost=87.98..87.98 rows=8798 width=21)
                              ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798
width=21)
```

Datenumverteilung

Die Ausgabe für EXPLAIN enthält bei Join-Operationen die Angabe der Methode, mit der Daten in einem Cluster bewegt werden, um die Join-Operation durchzuführen. Diese Datenbewegung kann entweder eine Rundsendung oder eine Umverteilung sein. Bei einer Rundsendung werden die Datenwerte von der einen Seite des Joins von jedem Verarbeitungsknoten an jeden anderen Verarbeitungsknoten gesendet, sodass jeder Verarbeitungsknoten am Ende der Operation eine vollständige Kopie der Daten hat. Bei einer Umverteilung werden die jeweiligen Datenwerte von ihrer aktuellen Slice an eine neue Slice gesendet, die sich möglicherweise auf einem anderen Knoten befindet. Die Daten werden normalerweise entsprechend dem Verteilungsschlüssel der anderen Tabelle des Joins umverteilt, wenn eine der Spalten der Join-Operation dieser Verteilungsschlüssel ist. Wenn keine der beiden Tabellen Verteilungsschlüssel in der jeweils anderen Tabelle hat, werden entweder beide Tabellen verteilt, oder die innere Tabelle wird an alle Knoten rundgesendet.

Die Ausgabe von EXPLAIN enthält auch Angaben zu inneren und äußeren Tabellen. Die innere Tabelle wird zuerst gescannt und wird weiter unten im Abfrageplan angezeigt. Die innere Tabelle ist die Tabelle, in der nach Übereinstimmungen gesucht wird. Sie wird normalerweise im Speicher gehalten und als Quelltabelle für Hash-Operationen verwendet. Falls möglich wird die kleiner der beiden Tabellen als innere Tabelle verwendet. Der äußeren Tabelle werden die Zeilen entnommen für die in der inneren Tabelle nach Übereinstimmungen gesucht wird. Sie wird normalerweise vom Datenträger gelesen. Der Abfrageoptimierer wählt auf der Grundlage der Datenbankstatistik, die bei der letzten Ausführung des Befehls ANALYZE erstellt wurde, aus, welche Tabelle als innere und welche als äußere verwendet wird. Die Reihenfolge der Tabellen in der FROM-Klausel der Abfrage beeinflusst diese Auswahl nicht.

Die folgenden Attribute in Abfrageplänen ermöglichen zu identifizieren, welche Daten bei der Ausführung einer Abfrage bewegt werden:

- DS_BCAST_INNER

  Es wird eine Kopie der gesamten inneren Tabelle an alle Verarbeitungsknoten rundgesendet.
- DS_DIST_ALL_NONE

Es ist keine Umverteilung erforderlich, weil die innere Tabelle bereits mit DISTSTYLE ALL an alle Knoten verteilt wurde.

- DS_DIST_NONE

  Es werden keine Tabellen umverteilt. Eine Zusammenstellung der Joins ist bereits möglich, weil die entsprechenden Slices verbunden werden, ohne dass Daten zwischen Knoten bewegt werden müssen.

- DS_DIST_INNER

  Die innere Tabelle wird umverteilt.

- DS_DIST_OUTER

  Die äußere Tabelle wird umverteilt.

- DS_DIST_ALL_INNER

  Die gesamte innere Tabelle wird an eine einzige Slice umverteilt wurde, weil die äußere Tabelle DISTSTYLE ALL verwendet.

- DS_DIST_BOTH

  Beide Tabellen werden umverteilt.

Übersicht über die Schritte des Abfrageplans

Sie können die Schritte von Abfrageplänen anzeigen, indem Sie den Befehls EXPLAIN ausführen. Das folgende Beispiel zeigt eine SQL-Abfrage und erläutert die Ausgabe. Wenn Sie den Abfrageplan von unten nach oben lesen, sehen Sie die einzelnen logischen Operationen, die für die Ausführung der Abfrage verwendet werden. Weitere Informationen finden Sie unter Abfrageplan.

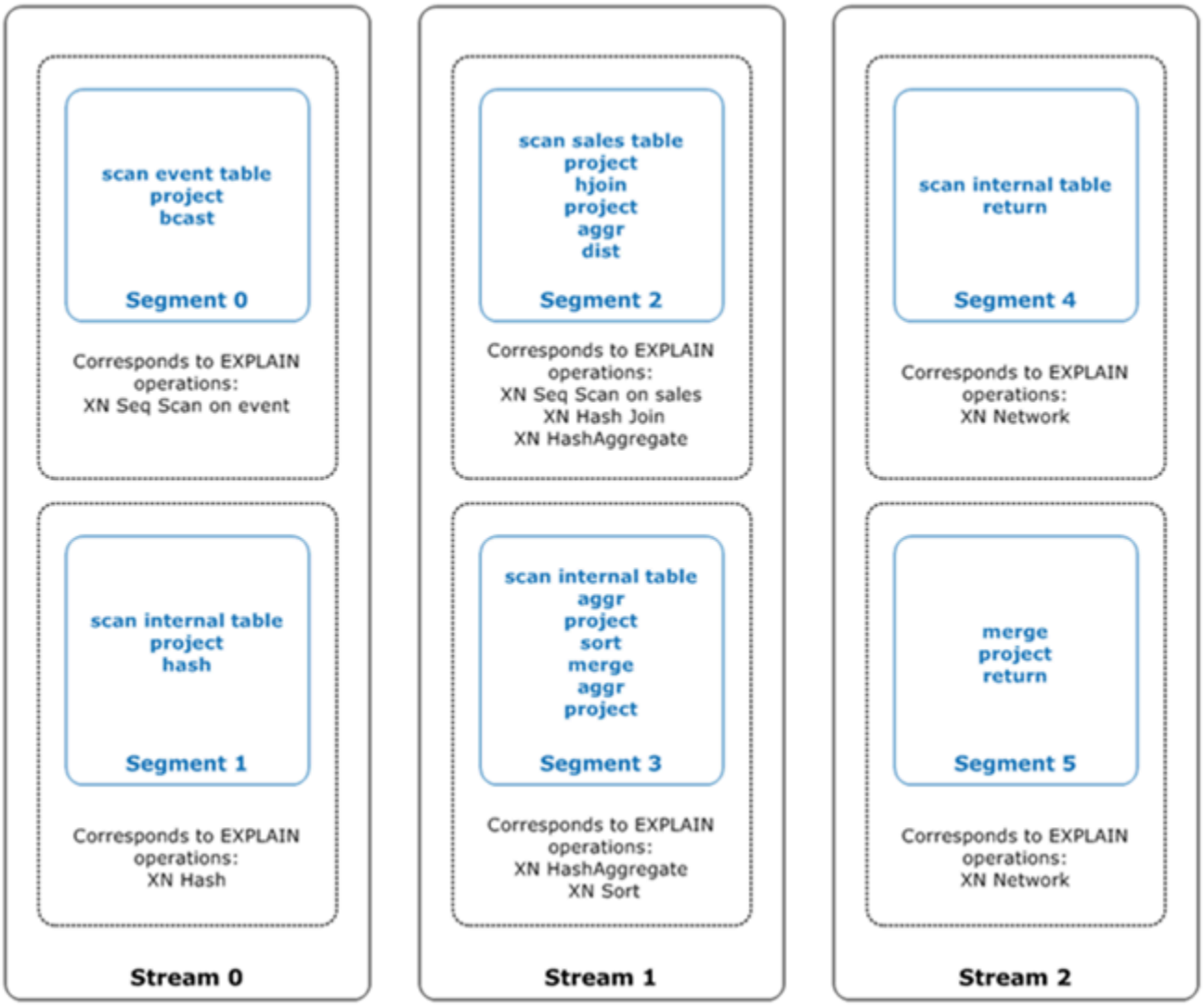
```
explain
select eventname, sum(pricepaid) from sales, event
where sales.eventid = event.eventid
group by eventname
order by 2 desc;
```

```
XN Merge  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
  Merge Key: sum(sales.pricepaid)
  ->  XN Network  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
```

```
        Send to leader
        ->  XN Sort  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
              Sort Key: sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=2815366577.07..2815366578.51 rows=576
width=27)
                    ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..2815365714.80
rows=172456 width=27)
                          Hash Cond: ("outer".eventid = "inner".eventid)
                          ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456
width=14)
                          ->  XN Hash  (cost=87.98..87.98 rows=8798 width=21)
                                ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798
width=21)
```

Beim Generieren eines Abfrageplans gliedert der Abfrageoptimierer den Plan in Streams, Segmente und Schritte auf. Der Abfrageoptimierer unterbricht den Plan, um die Verteilung der Daten und Abfrage-Workload an die Rechenknoten vorzubereiten. Weitere Informationen zu Segmenten und Schritten finden Sie unter Workflow der Abfrageplanung und -ausführung.

Die folgende Abbildung zeigt die vorhergehende Abfrage und den zugehörigen Abfrageplan. Es zeigt an, wie die beteiligten Abfrageoperationen Schritten zugeordnet werden, die Amazon Redshift zum Generieren von kompiliertem Code für die Rechenknoten-Slices verwendet. Den Operationen in dem Abfrageplan werden mehrere Schritte in Segmenten und manchmal auch mehrere Segmente in den Streams zugeordnet.



In dieser Abbildung führt der Abfrageoptimierer den Abfrageplan wie folgt aus:

1. In `Stream 0` führt die Abfrage `Segment 0` mit einem sequenziellen Scanvorgang aus, um die Tabelle `events` zu scannen. Die Abfrage wird bis `Segment 1` mit einem Hash-Vorgang fortgesetzt, um die Hashtabelle für die innere Tabelle in der Verknüpfung zu erstellen.
2. In `Stream 1` führt die Abfrage `Segment 2` mit einem sequenziellen Scanvorgang aus, um die Tabelle `sales` zu scannen. Es wird mit `Segment 2` mit einem Hash-Join fortgefahren, um Tabellen zu verknüpfen, bei denen die Join-Spalten nicht sowohl Verteilungsschlüssel als auch Sortierschlüssel sind. Es wird wieder mit `Segment 2` mit einem Hash-Aggregat fortgefahren, um Ergebnisse zu aggregieren. Dann führt die Abfrage `Segment 3` mit einem Hash-Aggregatvorgang aus, um unsortierte gruppierte Aggregatfunktionen auszuführen, und einen Sortiervorgang, um die ORDER BY-Klausel und andere Sortiervorgänge auszuwerten.

3. In `Stream 2` führt die Abfrage einen Netzwerkvorgang in `Segment 4` und `Segment 5` aus, um Zwischenergebnisse zur weiteren Verarbeitung an den Führerknoten zu senden.

Das letzte Segment einer Abfrage gibt die Daten zurück. Wenn der Rückgabesatz aggregiert oder sortiert ist, senden die Rechenknoten jeweils ihren Teil des Zwischenergebnisses an den Führungsknoten. Der Führungsknoten führt dann die Daten zusammen, sodass das Endergebnis an den anfordernden Client zurückgesendet werden kann.

Weitere Hinweise zu EXPLAIN-Operatoren finden Sie unter EXPLAIN.

Für die Abfrageleistung relevante Faktoren

Die Abfrageleistung kann durch eine Reihe von Faktoren beeinflusst werden. Wie schnell Ihre Abfragen ausgeführt werden, hängt von den folgenden Aspekten Ihrer Daten, Cluster und Datenbankoperationen ab.

- Anzahl der Knoten, Prozessoren oder Slices – Ein Rechenknoten wird in Slices aufgeteilt. Mehr Knoten bedeuten auch mehr Prozessoren und mehr Slices; dadurch können Abfragen schneller verarbeitet werden, weil mehr Teile der Abfrage gleichzeitig auf die Slices verteilt werden können. Mehr Knoten bedeuten jedoch auch, dass die Kosten für Verwaltungsoperationen höher sind. Es muss also bei der Anzahl der zu verwendenden Slices auf ein für Ihr System ausgewogenes Verhältnis zwischen Kosten und Leistung geachtet werden. Weitere Informationen zur Amazon-Redshift-Clusterarchitektur finden Sie unter Architektur des Data Warehouse-Systems.
- Knotentypen – Ein Amazon-Redshift-Cluster kann einen von mehreren Knotentypen verwenden. Die verschiedenen Knotentypen sind in verschiedenen Größen und mit unterschiedlichen Grenzwerten verfügbar, was eine geeignete Skalierung Ihres Clusters ermöglicht. Die Knotengröße bestimmt die Speicherkapazität, den Arbeitsspeicher, die CPU und den Preis für die einzelnen Knoten in dem Cluster. Weitere Informationen zu den einzelnen Knotentypen finden Sie unter Übersicht über Amazon-Redshift-Cluster im Amazon-Redshift-Verwaltungshandbuch.
- Datenverteilung – Amazon Redshift speichert Tabellendaten in den Rechenknoten entsprechend dem Verteilungsstil der Tabelle. Wenn sie eine Abfrage ausführen, führt der Abfrageoptimierer nach Bedarf eine Neuverteilung der Daten auf die Datenverarbeitungsknoten durch, um Join- oder Aggregierungsoperationen durchführen zu können. Durch die Auswahl eines geeigneten Tabellenverteilungsstils können die Auswirkungen des Neuverteilungsschritts dadurch minimiert werden, dass die Daten dort platziert sind, wo sie benötigt werden, bevor die Join-Operationen ausgeführt werden. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

- Datensortierreihenfolge – Amazon Redshift speichert die Tabellendaten auf der Festplatte in sortierter Form gemäß den Sortierschlüsseln einer Tabelle. Der Abfrageoptimierer und der Abfrageprozessor nutzen Informationen dazu, an welcher Stelle die Daten gespeichert sind, aus, um die Anzahl der zu scannenden Blöcke zu verringern und so die Ausführungsgeschwindigkeit der Abfrage zu verbessern. Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.
- Datensatzgröße – Eine größere Datenmenge im Cluster kann Abfragen verlangsamen, da mehr Zeilen gescannt und neu verteilt werden müssen. Sie können diese Effekt verringern, indem Sie die Daten regelmäßig komprimieren und archivieren, aber auch indem Sie mit einem Prädikat das Dataset für die Abfrage beschränken.
- Gleichzeitige Operationen – Die gleichzeitige Ausführung mehrerer Operationen kann die Abfrageleistung beeinträchtigen. Jede Operation belegt einen oder mehrere Plätze in einer verfügbaren Abfragewarteschlange und verbraucht den Speicher für diese Plätze. Wenn gleichzeitig weitere Operationen ausgeführt werden, sind möglicherweise nicht genügend Plätze in Abfragewarteschlangen verfügbar. In diesem Fall kann die Abfrage erst dann verarbeitet werden, wenn wieder Plätze verfügbar sind. Weitere Informationen zum Erstellen und Konfigurieren von Abfragewarteschlangen finden Sie unter Implementierung von Workload Management.
- Abfragestruktur – Die Abfrageleistung richtet sich auch danach, wie Ihre Abfrage geschrieben ist. Achten Sie bei der Formulierung von Abfragen darauf, dass jeweils so wenig Daten wie möglich und nur so viele Daten wie nötig verarbeitet und zurückgegeben werden. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift.
- Codekompilierung – Amazon Redshift generiert und kompiliert Code für jeden Abfrageausführungsplan.

  Der kompilierte Code wird schneller ausgeführt, weil der Verwaltungsaufwand für den Interpreter entfällt. Bei der Generierung und Kompilierung des Codes bei der erstmaligen Ausführung fallen stets auch Kosten für die Verwaltung der Operationen an. Aus diesem Grund kann die Abfrageleistung bei der ersten Ausführung signifikant von nachfolgenden Ausführungen abweichen. Diese Kosten für den Verwaltungsaufwand machen sich insbesondere bemerkbar, wenn Sie nur einmal verwendete Abfragen ausführen. Wenn Sie die typische Ausführungszeit der Abfrage ermitteln möchten, messen Sie die Zeit, die sich bei der zweiten Ausführung ergibt. Amazon Redshift verwendet einen Serverless-Kompilierungsservice, um Abfragekompilierungen über die Rechenressourcen eines Amazon-Redshift-Clusters hinaus zu skalieren. Die kompilierten Codesegmente werden lokal auf dem Cluster und in einem praktisch unbegrenzten Cache gespeichert. Dieser Cache bleibt nach dem Neustart des Clusters bestehen. Nachfolgende Ausführungen derselben Abfrage werden schneller ausgeführt, da sie die Kompilierungsphase überspringen.

Der Cache ist nicht versionsübergreifend mit Amazon Redshift kompatibel, daher wird der Kompilierungs-Cache gelöscht und der Code neu kompiliert, wenn Abfragen nach einem Versions-Upgrade ausgeführt werden. Wenn für Ihre Abfragen strenge SLAs gelten, empfehlen wir Ihnen, Abfragesegmente, die Daten aus Cluster-Tabellen scannen, vorab auszuführen. Dadurch kann Amazon Redshift die Basisdaten der Tabelle zwischenspeichern, wodurch die Planungszeit für Abfragen nach einem Versions-Upgrade reduziert wird. Durch die Verwendung eines skalierbaren Kompilierungsservice ist Amazon Redshift in der Lage, Code parallel zu kompilieren, um eine konstant schnelle Leistung zu bieten. Die Ausdehnung der Workload hängt von der Komplexität und Nebenläufigkeit von Abfragen ab.

Analysieren und Verbessern von Abfragen

Beim Abrufen von Daten aus einem Amazon-Redshift-Data-Warehouse werden komplexe Abfragen über riesigen Datenmengen ausgeführt, was sehr viel Zeit in Anspruch nehmen kann. Um sicherzustellen, dass Abfragen so schnell wie möglich ausgeführt werden, stehen Ihnen eine Reihe von Tools zur Verfügung, die Sie verwenden können, um potenzielle Probleme in Zusammenhang mit der Leistung aufzuspüren.

Themen

Workflow zur Analyse von Abfragen

Wenn die Ausführung einer Abfrage mehr Zeit als erwartet in Anspruch nimmt, verwenden Sie die folgenden Schritte, um Probleme zu identifizieren und zu korrigieren, die die Abfrageleistung potentiell mindern könnten. Wenn Sie nicht genau wissen, welche Abfragen in Ihrem System von der Leistungsoptimierung profitieren könnten, führen Sie zunächst die Diagnoseabfrage in aus Identifizieren der Top-Kandidaten zur Optimierung unter den Abfragen.

1. Stellen Sie sicher, dass Ihre Tabellen entsprechend bewährten Methoden entworfen sind. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift.
2. Überprüfen Sie, ob Sie nicht benötigte Daten in Ihren Tabellen löschen bzw. archivieren können. Beispiel: Nehmen Sie an, dass Ihre Abfragen stets nur Daten aus den letzten 6 Monaten verwenden, aber Ihre Tabellen enthalten die letzten 18 Monate. In diesem Fall können Sie die alten Daten löschen oder archivieren, um die Anzahl der zu scannenden und zu verteilenden Datensätze zu reduzieren.
3. Führen Sie den Befehl VACUUM über den Tabelle in der Abfrage aus, um Speicherbereiche wieder freizugeben und Zeilen neu zu sortieren. Die Ausführung von VACUUM kann dazu beitragen, die Leistung zu verbessern, wenn ein großer Bereich unsortiert ist, und der Sortierschlüssel in der Abfrage in einer Join-Operation oder im Prädikat verwendet wird.
4. Führen Sie den Befehl ANALYZE für die Tabellen in der Abfrage aus, um sicherzustellen, dass die Datenbankstatistik aktuell ist. Die Ausführung von ANALYZE wird empfohlen, wenn sich die Größe der in der Abfrage verwendeten Tabellen in größerem Umfang geändert hat. Gelegentlich kann die Ausführung eines umfassenden ANALYSE-Befehls zu viel Zeit in Anspruch nehmen. In diesem Fall kann jedoch bereits die Ausführung von ANALYZE über einer bestimmten Spalte die Verarbeitungszeit beträchtlich verbessern. Auch bei eines solchen eingeschränkten ANALYZE-Ausführung wird die Statistik zu Tabellengrößen aktualisiert. Die Tabellengröße ist ein signifikanter Faktor bei der Abfrageplanung.
5. Stellen Sie sicher, dass Ihre Abfrage für jeden Clienttyp (also für alle clientseitig verwendeten Verbindungsprotokolle) einmal ausgeführt wurde, damit die Abfrage kompiliert und zwischengespeichert wird. Dieser Ansatz beschleunigt die nachfolgenden Ausführungen der Abfrage. Weitere Informationen finden Sie unter Für die Abfrageleistung relevante Faktoren.
6. Überprüfen Sie die Tabelle STL_ALERT_EVENT_LOG, um mögliche Probleme bei Ihrer Abfrage zu identifizieren und zu korrigieren. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.
7. Um den Abfrageplan abzurufen und mit seiner Hilfe die Abfrage zu optimieren, führen Sie den Befehl EXPLAIN aus. Weitere Informationen finden Sie unter Analysieren des Abfrageplans.
8. Um die Abfragezusammenfassung abzurufen und mit ihrer Hilfe die Abfrage zu optimieren, führen Sie die Befehle SVL_QUERY_SUMMARY bzw. SVL_QUERY_REPORT aus. Weitere Informationen finden Sie unter Analysieren des Abfragezusammenfassung.

Manchmal muss eine Abfrage, die schnell ausgeführt werden sollte, warten, bis eine andere Abfrage mit längerer Ausführungszeit abgeschlossen ist. In diesem Fall braucht möglicherweise an der

Abfrage selbst nichts verbessert werden, aber Sie können die Systemleistung insgesamt verbessern, indem Sie für verschiedene Arten von Abfragen Warteschlangen erstellen und verwenden. Beispiele zu Wartezeiten für verschiedene Abfragen in Warteschlangen finden Sie unter Überprüfen der Wartezeiten von Abfragen in Warteschlangen. Weitere Informationen zum Konfigurieren von Abfragewarteschlangen finden Sie unter Implementierung von Workload Management.

Überprüfen von Abfragewarnungen

Verwenden Sie die Systemtabelle STL_ALERT_EVENT_LOG wie folgt, um mögliche Performanceprobleme bei Ihrer Abfrage zu identifizieren und zu korrigieren.

1. Führen Sie die folgende Abfrage aus, um die Abfrage-ID anzuzeigen:

```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

   Untersuchen Sie den Abfrageausschnitt im Feld `substring`, um herauszufinden, welchen `query`-Wert Sie auswählen müssen. Wenn Sie die Abfrage mehrmals ausgeführt haben, verwenden Sie den `query`-Wert aus der Zeile mit dem niedrigeren `elapsed`-Wert. Dies ist die Zeile für die kompilierte Version. Wenn Sie viele Abfragen ausgeführt haben, können Sie den in der LIMIT-Klausel verwendeten Wert heraufsetzen, um sicherzustellen, dass die Abfrage berücksichtigt wird.

2. Wählen Sie Zeilen aus STL_ALERT_EVENT_LOG für Ihre Abfrage aus:

```
Select * from stl_alert_event_log where query = MyQueryID;
```

userid	query	slice	segment	step	pid	xid	event	solution	event_time
100	32359	4	0	0	8780	71195	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-10 17:40:50
100	32359	5	0	0	8781	71195	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-10 17:40:50
100	109142	4	0	0	8780	302411	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-24 20:32:28
100	109142	5	0	0	8781	302411	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-24 20:32:28
100	109828	4	1	0	8746	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:27:52
100	109828	5	1	0	8747	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:27:52
100	109829	4	1	0	8760	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:28:01
100	109829	5	1	0	8761	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:28:01
100	113910	4	1	0	8774	316848	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-25 17:14:58

3. Werten Sie die Ergebnisse für Ihre Abfrage aus. Verwenden Sie die folgende Tabelle, um mögliche Lösungen für die identifizierten Probleme zu finden.

> **Note**
>
> In STL_ALERT_EVENT_LOG sind nur Abfragen mit erkannten Problemen aufgeführt.

Problem	Ereigniswert	Lösungswert	Empfohlene Lösung
Die Statistiken für die Tabellen in der Abfrage sind nicht vorhanden oder veraltet.	Statistiken des Abfrageplaners fehlen.	Führen Sie den Befehl ANALYZE aus.	Siehe Die Tabellenstatistik fehlt oder ist veraltet..
In dem Abfrageplan wird eine verschachtelter Schleifen-Join (die am wenigsten effiziente Join-Vari ante) angezeigt.	Verschachtelter Schleifen-Join in der Abfrage	Überprüfen Sie, ob die Join-Prädikate so umformuliert werden können, dass kartesisc he Produkte vermieden werden.	Siehe Verschachtelte Schleife.
Der Scan hat eine sehr große Anzahl an Zeilen übersprungen, die als gelöscht markiert sind, jedoch noch nicht bereinigt wurden, oder es wurden Zeilen eingefügt, aber nicht per Commit bestätigt.	Es wurde eine große Menge an gelöschten Zeilen gescannt.	Führen Sie den Befehl VACUUM aus, um den Speicherplatz für die gelöschte n Zeilen zurückzug ewinnen.	Siehe Geisterze ilen und Zeilen mit ausstehen dem Commit.

Problem	Ereigniswert	Lösungswert	Empfohlene Lösung
Für einen Hash-Join oder eine Aggregierung wurden mehr als 1.000.000 Zeilen umverteilt.	Es wurde eine große Anzahl von Zeilen über das Netzwerk verteilt: RowCount Zeilen wurden verteilt, um die Aggregation zu verarbeiten	Überprüfen Sie die Auswahl des Verteilungsschlüssels, mit dem die Join- oder Aggregat-Operation zusammengefasst werden.	Siehe Suboptimale Datenverteilung.
Für einen Hash-Join wurden mehr als 1.000.000 Zeilen rundgesendet.	Es wurden sehr viele Zeilen über das Netzwerk gesendet.	Überprüfen Sie die Auswahl des Verteilungsschlüssels, mit dem die Join-Operation zusammengefasst wird, oder verwenden Sie verteilte Tabellen.	Siehe Suboptimale Datenverteilung.
In dem Abfrageplan wird ein Umverteilungsstil von DS_DIST_ALL_INNER angezeigt, der die serielle Ausführung erzwingt, da die gesamte innere Tabelle an einen einzelnen Knoten umverteilt wurde.	DS_DIST_ALL_INNER für Hash-Join im Abfrageplan	Überprüfen Sie die Auswahl des Verteilungsschlüssels, um sicherzustellen, dass die innere, und nicht die äußere Tabelle verteilt wird.	Siehe Suboptimale Datenverteilung.

Analysieren des Abfrageplans

Um den Abfrageplan zu analysieren, sollten Sie sich zunächst damit vertraut machen, wie diese Pläne zu lesen sind. Wenn Sie sich erstmals mit der Interpretation von Abfrageplänen beschäftigen, empfehlen wir Ihnen, Abfrageplan zu lesen, bevor Sie fortfahren.

Führen Sie den Befehl EXPLAIN aus, um den Abfrageplan abzurufen. Folgen Sie diesen Anweisungen, um die Daten in dem Abfrageplan zu analysieren:

1. Identifizieren Sie die Schritte mit den höchsten Kosten. Konzentrieren Sie sich bei den restlichen Anweisungen darauf, diese Schritte zu optimieren.
2. Achten Sie auf die Join-Varianten:
   - Nested Loop (Verschachtele Schleife): Verschachtelte Schleifen treten häufig auf, weil eine Bedingung für die Join-Operation fehlt. Empfohlene Lösungen finden Sie unter Verschachtelte Schleife.
   - Hash and Hash Join (Hash und Hash-Join): Hash-Joins werden bei Join-Operationen für Tabellen verwendet, bei denen die Join-Spalten weder Verteilungsschlüssel noch Sortierschlüssel sind. Empfohlene Lösungen finden Sie unter Hash join.
   - Merge Join (Zusammenführungs-Join): keine Änderung erforderlich.
3. Achten Sie darauf, welche Tabellen für den inneren Join verwendet werden, und welche für den äußeren. Die Abfrage-Engine verwendet normalerweise die kleinere Tabelle für den inneren Join und die größere für den äußeren. Wenn die Abfrage offensichtlich die beiden Tabellen falsch zuordnet, besteht die Möglichkeit, dass die Datenbankstatistik nicht mehr aktuell ist. Empfohlene Lösungen finden Sie unter Die Tabellenstatistik fehlt oder ist veraltet..
4. Schauen Sie nach, ob bestimmte Sortieroperationen besonders teuer sind. Falls dies der Fall ist, finden Sie empfohlene Lösungen unter Unsortierte oder falsch sortierte Zeilen.
5. Sehen Sie nach den folgenden Broadcastoperatoren, wenn es Operationen mit hohen Kosten gibt:
   - DS_BCAST_INNER: Zeigt an, dass die Tabelle an alle Computingknoten übertragen wird. Dies ist bei kleinen Tabellen unproblematisch, bei großen Tabellen jedoch nicht ideal.
   - DS_DIST_ALL_INNER: Gibt an, dass ein einziger Slice den gesamten Workload ausführt.
   - DS_DIST_BOTH: Gibt an, dass Daten stark umverteilt werden.

   Empfohlene Lösungen für diese Situationen finden Sie unter Suboptimale Datenverteilung.

Analysieren des Abfragezusammenfassung

Wenn Sie die Ausführungsschritte und die Statistik etwas detaillierter benötigen als in dem Abfrageplan, den EXPLAIN ausgibt, verwenden Sie die Systemansichten SVL_QUERY_SUMMARY und SVL_QUERY_REPORT.

SVL_QUERY_SUMMARY zeigt Abfragestatistiken für die Stream an. Sie können die ausgegebenen Informationen verwenden, um Probleme in Zusammenhang mit besonders kostenintensiven Schritten, Schritten mit besonders langer Ausführungszeit und Schritten, bei denen Daten auf den Datenträger geschrieben werden, zu identifizieren.

Die Systemansicht SVL_QUERY_REPORT zeigt Informationen ähnlich denen von SVL_QUERY_SUMMARY an, allerdings nicht nach Streams angeordnet, sondern nach den Slices auf den Verarbeitungsknoten. Sie können diese Informationen auf Slice-Ebene verwenden, um eine ungleichmäßige Datenverteilung in dem Cluster zu erkennen (eine „verzerrte“ Datenverteilung). Eine verzerrte Datenverteilung zwingt einige Knoten dazu, mehr Arbeit zu verrichten als andere, was die Abfrageleistung beeinträchtigt.

Themen

- Verwenden der Ansicht SVL_QUERY_SUMMARY
- Verwenden der Ansicht SVL_QUERY_REPORT
- Zuweisung der Informationen in Abfrageplan und Abfragezusammenfassung

Verwenden der Ansicht SVL_QUERY_SUMMARY

Gehen Sie wie folgt vor, um eine Abfragezusammenfassung nach Streams zu analysieren:

1. Führen Sie die folgende Abfrage aus, um die Abfrage-ID anzuzeigen:

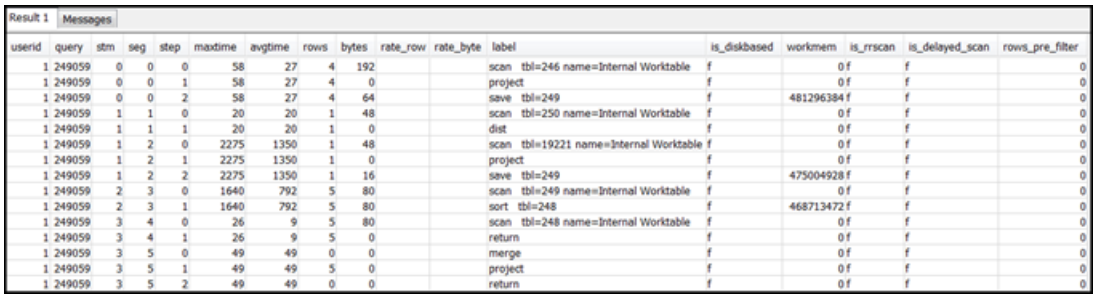
```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

   Untersuchen Sie den Abfrageausschnitt im Feld `substring`, um herauszufinden, welcher `query`-Wert Ihrer Abfrage entspricht. Wenn Sie die Abfrage mehrmals ausgeführt haben, verwenden Sie den `query`-Wert aus der Zeile mit dem niedrigeren `elapsed`-Wert. Dies ist die Zeile für die

kompilierte Version. Wenn Sie viele Abfragen ausgeführt haben, können Sie den in der LIMIT-Klausel verwendeten Wert heraufsetzen, um sicherzustellen, dass die Abfrage berücksichtigt wird.

2. Wählen Sie Zeilen aus SVL_QUERY_SUMMARY für Ihre Abfrage aus. Ordnen Sie die Ergebnisse nach Stream, Segment und Schritt an:

```
select * from svl_query_summary where query = MyQueryID order by stm, seg, step;
```



Result 1 Messages

userid	query	stm	seg	step	maxtime	avgtime	rows	bytes	rate_row	rate_byte	label	is_diskbased	workmem	is_rrscan	is_delayed_scan	rows_pre_filter
1	249059	0	0	0	58	27	4	192			scan tbl=246 name=Internal Worktable	f	0	f	f	0
1	249059	0	0	1	58	27	4	0			project	f	0	f	f	0
1	249059	0	0	2	58	27	4	64			save tbl=249	f	481296384	f	f	0
1	249059	1	1	0	20	20	1	48			scan tbl=250 name=Internal Worktable	f	0	f	f	0
1	249059	1	1	1	20	20	1	0			dist	f	0	f	f	0
1	249059	1	2	0	2275	1350	1	48			scan tbl=19221 name=Internal Worktable	f	0	f	f	0
1	249059	1	2	1	2275	1350	1	0			project	f	0	f	f	0
1	249059	1	2	2	2275	1350	1	16			save tbl=249	f	475004928	f	f	0
1	249059	2	3	0	1640	792	5	80			scan tbl=249 name=Internal Worktable	f	0	f	f	0
1	249059	2	3	1	1640	792	5	80			sort tbl=248	f	468713472	f	f	0
1	249059	3	4	0	26	9	5	80			scan tbl=248 name=Internal Worktable	f	0	f	f	0
1	249059	3	4	1	26	9	5	0			return	f	0	f	f	0
1	249059	3	5	0	49	49	0	0			merge	f	0	f	f	0
1	249059	3	5	1	49	49	5	0			project	f	0	f	f	0
1	249059	3	5	2	49	49	0	0			return	f	0	f	f	0

3. Ordnen Sie die Schritte und die Operationen in dem Abfrageplan einander zu. Verwenden Sie dabei die Informationen in Zuweisung der Informationen in Abfrageplan und Abfragezusammenfassung. Zusammengehörige Elemente sollten näherungsweise die gleichen Werte für Zeilen und Bytes (jeweils Zeilen * Breite aus dem Abfrageplan) haben. Falls sich hier unerwartete Abweichungen ergeben, finden Sie empfohlene Lösungen unter Die Tabellenstatistik fehlt oder ist veraltet..
4. Überprüfen Sie, ob das Feld `is_diskbased` für einen Schritt den Wert `t` (true) hat. Hashes, Aggregat- und Sortierfunktionen sind Operatoren, die mit hoher Wahrscheinlichkeit Daten auf den Datenträger schreiben, falls das System nicht genügend Arbeitsspeicher für die Verarbeitung der Abfrage zugeteilt hat.

   Falls `is_diskbased` wahr ist, finden Sie empfohlene Lösungen unter Ungenügende Speicherzuteilung für die Abfrage.
5. Überprüfen Sie die Werte des Feldes `label` und überprüfen Sie, ob sich in den Schritten eine AGG-DIST-AGG-Sequenz befindet. Eine solche Konstellation weist auf eine Aggregierung in zwei Schritten hin, die kostenintensiv ist. Sie können dieses Problem beheben, indem Sie die GROUP BY-Klausel so ändern, dass der Verteilungsschlüssel verwendet wird (bei mehreren der erste).
6. Überprüfen Sie für jedes Segment den Wert von `maxtime` (ist für alle Schritte in einem Segment gleich). Identifizieren Sie das Segment mit dem höchsten `maxtime`-Wert und durchsuchen Sie die Schritte in diesem Segment nach den folgenden Operatoren.

> **Note**
>
> Ein hoher `maxtime`-Wert muss nicht zwangsläufig ein Problem bei dem Segment bedeuten. Auch Segmente, bei denen dieser Wert hoch ist, können schnell verarbeitet worden sein. Die Zeitmessung wird für alle Segmente in einem Stream gleichzeitig gestartet. Einige nachgelagerte Segmente können aber erst dann ausgeführt werden, wenn Daten aus vorangehenden Segmenten abrufbar sind. Dies führt dazu, dass offenbar viel Zeit benötigt wird, weil der `maxtime`-Wert die Warte- und die Verarbeitungszeit beinhaltet.

- BCAST oder DIST: In diesen Fällen kann ein hoher Wert von `maxtime` aufgrund einer Umverteilung einer großen Anzahl an Zeilen zustande kommen. Empfohlene Lösungen finden Sie unter Suboptimale Datenverteilung.
- HJOIN (Hash-Join): Wenn der betreffende Schritt im Feld `rows` im Vergleich zum `rows`-Wert im abschließenden RETURN-Schritt in der Abfrage einen sehr großen Wert hat, finden Sie empfohlene Lösungen unter Hash join.
- SCAN/SORT: Suchen Sie nach einer einem Join-Schritt vorausgehenden SCAN-SORT-SCAN-MERGE-Schrittfolge. Ein solches Muster weist darauf hin, dass unsortierte Daten gescannt, sortiert und dann mit dem sortierten Bereich der Tabelle zusammengeführt werden.

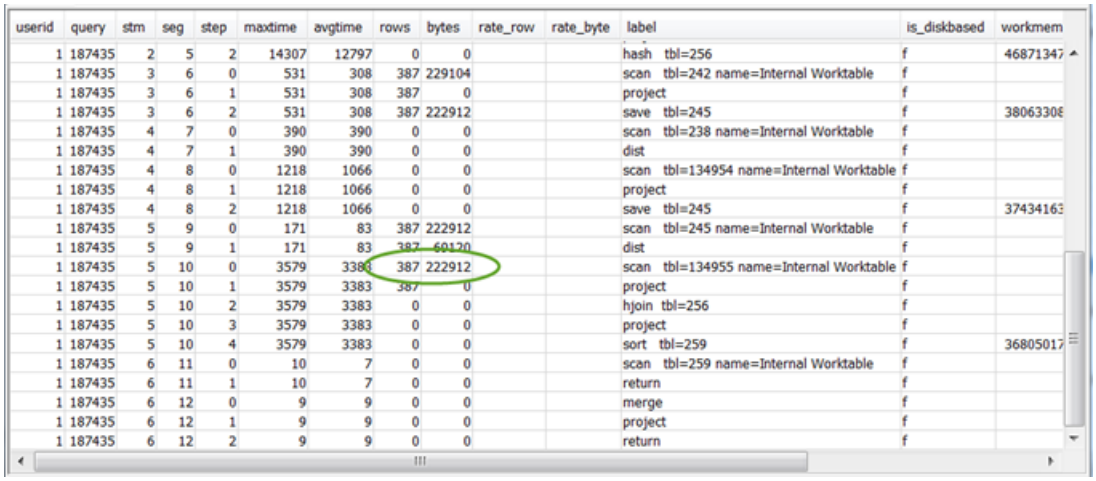
  Überprüfen Sie, ob der Zeilenwert für den SCAN-Schritt im Vergleich zu dem Zeilenwert im abschließenden RETURN-Schritt in der Abfrage einen sehr großen Wert aufweist. Ein solches Muster weist darauf hin, dass die Ausführungs-Engine Zeilen scannt, die später verworfen werden, was ineffizient ist. Empfohlene Lösungen finden Sie unter Nicht ausreichend restriktives Prädikat.

  Falls der `maxtime`-Wert für den SCAN-Schritt besonders hoch ist, finden Sie empfohlene Lösungen unter Suboptimale WHERE-Klausel.

  Falls der `rows`-Wert für den SORT-Schritt ungleich Null ist, finden Sie empfohlene Lösungen unter Unsortierte oder falsch sortierte Zeilen.

7. Überprüfen Sie die Werte für `rows` und `bytes` in den 5 bis 10 Schritten vor dem abschließenden RETURN-Schritt, um sich einen Eindruck davon zu verschaffen, welche Datenmengen an den Client zurückgegeben werden. Dies ist nicht immer ganz einfach.

In der folgenden Zusammenfassungsabfrage sehen Sie beispielsweise, dass der dritte PROJECT-Schritt einen Wert für `rows`, aber nicht für `bytes` hat. Wenn Sie in den vorangehenden Schritten nach demselben `rows`-Wert suchen, finden Sie den SCAN-Schritt mit Zeilen- und Byteinformationen:



userid	query	stm	seg	step	maxtime	avgtime	rows	bytes	rate_row	rate_byte	label	is_diskbased	workmem
1	187435	2	5	2	14307	12797	0	0			hash tbl=256	f	46871347
1	187435	3	6	0	531	308	387	229104			scan tbl=242 name=Internal Worktable	f	
1	187435	3	6	1	531	308	387	0			project	f	
1	187435	3	6	2	531	308	387	222912			save tbl=245	f	3806330
1	187435	4	7	0	390	390	0	0			scan tbl=238 name=Internal Worktable	f	
1	187435	4	7	1	390	390	0	0			dist	f	
1	187435	4	8	0	1218	1066	0	0			scan tbl=134954 name=Internal Worktable	f	
1	187435	4	8	1	1218	1066	0	0			project	f	
1	187435	4	8	2	1218	1066	0	0			save tbl=245	f	37434163
1	187435	5	9	0	171	83	387	222912			scan tbl=245 name=Internal Worktable	f	
1	187435	5	9	1	171	83	387	60120			dist	f	
1	187435	5	10	0	3579	3383	387	222912			scan tbl=134955 name=Internal Worktable	f	
1	187435	5	10	1	3579	3383	387	0			project	f	
1	187435	5	10	2	3579	3383	0	0			hjoin tbl=256	f	
1	187435	5	10	3	3579	3383	0	0			project	f	
1	187435	5	10	4	3579	3383	0	0			sort tbl=259	f	36805017
1	187435	6	11	0	10	7	0	0			scan tbl=259 name=Internal Worktable	f	
1	187435	6	11	1	10	7	0	0			return	f	
1	187435	6	12	0	9	9	0	0			merge	f	
1	187435	6	12	1	9	9	0	0			project	f	
1	187435	6	12	2	9	9	0	0			return	f	

Wenn die zurückgegebene Datenmenge unerwartet groß ist, finden Sie empfohlene Lösungen unter Sehr große Ergebnismengen.

8. Überprüfen Sie, ob der Wert für `bytes` verglichen mit den `rows`-Werten für andere Schritte relativ hoch ist. Dieses Muster kann darauf hinweisen, dass eine große Anzahl an Spalten ausgewählt ist. Empfohlene Lösungen finden Sie unter Große SELECT-Liste.

Verwenden der Ansicht SVL_QUERY_REPORT

Gehen Sie wie folgt vor, um eine Abfragezusammenfassung nach Slices zu analysieren:

1. Führen Sie die folgende Abfrage aus, um die Abfrage-ID anzuzeigen:

```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

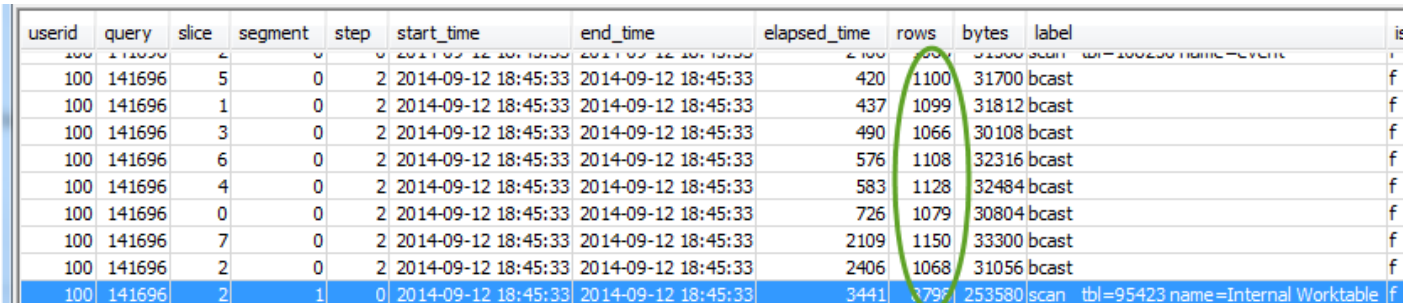
Untersuchen Sie den Abfrageausschnitt im Feld `substring`, um herauszufinden, welcher `query`-Wert Ihrer Abfrage entspricht. Wenn Sie die Abfrage mehrmals ausgeführt haben, verwenden Sie den `query`-Wert aus der Zeile mit dem niedrigeren `elapsed`-Wert. Dies ist die Zeile für die

kompilierte Version. Wenn Sie viele Abfragen ausgeführt haben, können Sie den in der LIMIT-Klausel verwendeten Wert heraufsetzen, um sicherzustellen, dass die Abfrage berücksichtigt wird.

2. Wählen Sie Zeilen aus SVL_QUERY_REPORT für Ihre Abfrage aus. Ordnen Sie die Ergebnisse nach Segment, Schritt, verbrauchter Zeit und Zeilen an:

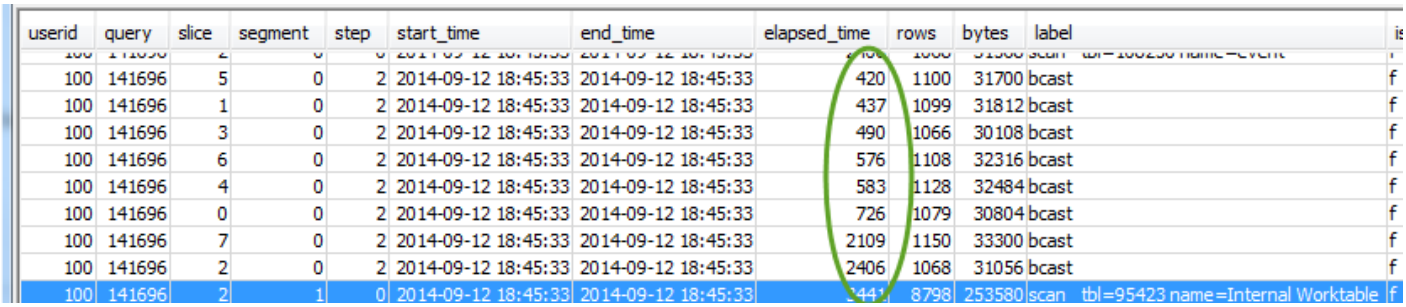
```
select * from svl_query_report where query = MyQueryID order by segment, step,
 elapsed_time, rows;
```

3. Überprüfen Sie, ob auch alle Slices etwa dieselbe Anzahl an Zeilen verarbeiten:



userid	query	slice	segment	step	start_time	end_time	elapsed_time	rows	bytes	label	is
100	141696	5	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	420	1100	31700	bcast	f
100	141696	1	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	437	1099	31812	bcast	f
100	141696	3	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	490	1066	30108	bcast	f
100	141696	6	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	576	1108	32316	bcast	f
100	141696	4	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	583	1128	32484	bcast	f
100	141696	0	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	726	1079	30804	bcast	f
100	141696	7	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2109	1150	33300	bcast	f
100	141696	2	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2406	1068	31056	bcast	f
100	141696	2	1	0	2014-09-12 18:45:33	2014-09-12 18:45:33	3441	8798	253580	scan tbl=95423 name=Internal Worktable	f

Überprüfen Sie auch, ob auch alle Slices etwa dieselbe Verarbeitungsdauer haben:



userid	query	slice	segment	step	start_time	end_time	elapsed_time	rows	bytes	label	is
100	141696	5	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	420	1100	31700	bcast	f
100	141696	1	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	437	1099	31812	bcast	f
100	141696	3	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	490	1066	30108	bcast	f
100	141696	6	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	576	1108	32316	bcast	f
100	141696	4	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	583	1128	32484	bcast	f
100	141696	0	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	726	1079	30804	bcast	f
100	141696	7	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2109	1150	33300	bcast	f
100	141696	2	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2406	1068	31056	bcast	f
100	141696	2	1	0	2014-09-12 18:45:33	2014-09-12 18:45:33	3441	8798	253580	scan tbl=95423 name=Internal Worktable	f

Große Unterschiede zwischen diesen Werten können auf eine verzerrte Datenverteilung hinweisen, weil für diese Abfrage nicht der beste Verteilungsstil verwendet wird. Empfohlene Lösungen finden Sie unter Suboptimale Datenverteilung.

Zuweisung der Informationen in Abfrageplan und Abfragezusammenfassung

Wenn Sie wissen, welche Operationen im Abfrageplan und welche Schritte in der Abfragezusammenfassung (die über die Werte im Feld „Label“ identifiziert werden können) zusammengehören, können Sie weitere Erkenntnisse erhalten:

Operation im Abfrageplan	Wert im Feld „Label“	Beschreibung
Aggregate HashAggregate GroupAggregate	AGGR	Bewertet Aggregationsfunktionen und GROUP BY-Bedingungen.
DS_BCAST_INNER	BCAST (Broadcast)	Sendet eine ganze Tabelle oder eine Teilmenge der Zeilen (beispielsweise eine gefilterte Teilmenge einer Tabelle) an alle Knoten rund.
Kommt im Abfrageplan nicht vor.	DELETE	Löscht Zeilen aus Tabellen.
DS_DIST_NONE DS_DIST_ALL_NONE DS_DIST_INNER DS_DIST_ALL_INNER DS_DIST_ALL_BOTH	DIST (Verteilung)	Verteilt Zeilen zum parallel ausgeführten Joinen oder zu anderen parallel ausgeführten Operationen an die Verarbeitungsknoten.
HASH	HASH	Generiert die Hash-Tabelle zur Verwendung in Hash-Joins.
Hash Join	HJOIN (Hash-Join)	Führt einen Hash-Join für zwei Tabellen oder Zwischenergebnismengen durch.
Kommt im Abfrageplan nicht vor.	INSERT	Fügt Zeilen in Tabellen ein.
Limit	LIMIT	Wendet eine LIMIT-Klausel auf Ergebnismengen an.

Operation im Abfrageplan	Wert im Feld „Label“	Beschreibung
Merge	MERGE	Führt Zeilen aus parallel ausgeführten Sortier- oder Join-Operationen zusammen.
Merge Join	MJOIN (Zusammenführungs-Join)	Führt einen Merge-Join für zwei Tabellen oder Zwischene rgebnismengen durch.
Nested Loop	NLOOP (verschachtelte Schleife)	Führt einen verschachtelten Loop-Join für zwei Tabellen oder Zwischenergebnisme ngen durch.
Kommt im Abfrageplan nicht vor.	PARSE	Parst Zeichenfolgen zum Laden in binäre Werte.
Projekt	PROJECT	Bewertet Ausdrücke.
Netzwerk	RETURN	Gibt Zeilen an den Führungsk noten bzw. an den Client zurück.
Kommt im Abfrageplan nicht vor.	SAVE	Speichert Zeilen zur Verwendung im nächsten Verarbeitungsschrift persistent zwischen.
Seq Scan	SCAN	Scannt Tabellen oder Zwischenergebnismengen.
Sortierung	SORT	Sortiert Zeilen oder Zwischene rgebnistabellen entsprech end den Anforderungen von nachfolgenden Operation en wie Join- oder Aggregat-Operationen, oder von einer ORDER BY-Klausel.

Operation im Abfrageplan	Wert im Feld „Label“	Beschreibung
Unique	UNIQUE	Wendet eine SELECT DISTINCT-Klausel an bzw. entfernt Duplikate, entsprechend den Anforderungen anderer Operationen.
Window	WINDOW	Berechnet Fensterfunktionen für Aggregat-Operationen und zur Rangfestlegung.

Verbessern der Abfrageleistung

Nachstehend finden Sie eine Liste häufig auftretender Probleme, die die Abfrageleistung beeinträchtigen, und Möglichkeiten zu ihrer Diagnose und Behandlung.

Themen

Die Tabellenstatistik fehlt oder ist veraltet.

Wenn die Statistik für die Tabellen nicht vorhanden oder veraltet sind, sieht die Ausgabe wie folgt aus:

- In den Ergebnissen für EXPLAIN eine Warnmeldung.
- In STL_ALERT_EVENT_LOG ein Warnereignis zu fehlender Statistik. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.

Führen Sie den Befehl aus, um dieses Problem zu beheben ANALYZE.

Verschachtelte Schleife

Wenn eine verschachtelte Schleife vorhanden ist, wird häufig ein entsprechendes Warnereignis in STL_ALERT_EVENT_LOG protokolliert. Sie können diese Art von Ereignissen auch identifizieren, indem Sie die Abfrage unter Identifizieren von Abfragen mit verschachtelten Schleifen. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.

Um diesem Verhalten entgegenzuwirken, überprüfen Sie Ihre Abfragen auf Kreuz-Joins und entfernen Sie sie nach Möglichkeit. Kreuz-Joins haben keine Join-Bedingung, daher ist das Ergebnis das kartesische Produkt zweier Tabellen. Sie werden typischerweise als verschachtelte Loop-Joins ausgeführt; dies ist die langsamste der Join-Varianten.

Hash join

Wenn ein Hash-Join vorhanden ist, sieht die Ausgabe wie folgt aus:

- Hash- und Hash-Join-Operationen im Abfrageplan. Weitere Informationen finden Sie unter Analysieren des Abfrageplans.
- Ein HJOIN-Schritt in dem Segment mit dem höchsten Wert für „maxtime“ in SVL_QUERY_SUMMARY. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_SUMMARY.

Um dieses Problem zu beheben, sind eine Reihe von Ansätzen möglich:

- Formulieren Sie die Abfrage nach Möglichkeit so um, dass ein Zusammenführungs-Join verwendet wird. Sie können dies erreichen, indem Sie Spalten für die Join-Operationen angeben, die gleichzeitig Verteilungsschlüssel und Sortierschlüssel sind.
- Wenn der HJOIN-Schritt in SVL_QUERY_SUMMARY im Vergleich zu dem Wert im Feld „rows“ im abschließenden RETURN-Schritt in der Abfrage einen sehr großen Wert aufweist, überprüfen Sie, ob Sie die Abfrage nicht so formulieren können, dass die Join-Operation über eine einzige Spalte durchgeführt werden kann. Wenn eine Abfrage eine Join-Operation über eine Spalte durchführt, die

nicht eindeutig (also beispielsweise kein Primärschlüssel) ist, bedeutet dies, dass bei der Operation mehr Zeilen verarbeitet werden müssen.

Geisterzeilen und Zeilen mit ausstehendem Commit

Wenn es Geisterzeilen oder Zeilen mit ausstehendem Commit gibt, wird möglicherweise ein Warnereignis in STL_ALERT_EVENT_LOG protokolliert, das auf eine große Anzahl von Geisterzeilen hinweist. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.

Um dieses Problem zu beheben, sind eine Reihe von Ansätzen möglich:

- Überprüfen Sie die Registerkarte Loads (Ladevorgänge) Ihrer Amazon-Redshift-Konsole auf aktive Ladevorgänge für eine der Abfragetabellen. Wenn dies der Fall ist, warten Sie, bis diese Operationen abgeschlossen sind, bevor Sie weitere Schritte unternehmen.
- Falls keine Ladeoperationen aktiv sind, führen Sie über den Tabellen der Abfrage den Befehl VACUUM aus, um gelöschte Zeilen ganz aus dem Speicher zu löschen.

Unsortierte oder falsch sortierte Zeilen

Wenn unsortierte oder falsch sortierte Zeilen vorhanden sind, wird häufig ein Warnereignis mit starker Beschränkung in STL_ALERT_EVENT_LOG protokolliert. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.

Sie können auch überprüfen, ob in einer der Tabellen in Ihrer Abfrage große unsortierte Bereiche vorhanden sind, indem Sie die Abfrage in ausführen Identifizieren von Tabellen mit verzerrter Datenverteilung oder unsortierten Zeilen.

Um dieses Problem zu beheben, sind eine Reihe von Ansätzen möglich:

- Führen Sie den Befehl VACUUM über den Tabellen der Abfrage aus, um die Zeilen zu sortieren.
- Überprüfen Sie, ob Verbesserungen an den Sortierschlüsseln der Tabellen in der Abfrage möglich sind. Achten Sie dabei darauf, zwischen der Leistung dieser Abfrage und der von anderen wichtigen Abfragen und der gesamten Systemleistung, falls Sie Modifikationen vornehmen möchten. Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.

Suboptimale Datenverteilung

Wenn die Datenverteilung suboptimale ist, sieht die Ausgabe wie folgt aus:

- Es wird ein Warnereignis zur reihenweisen Ausführung, zu umfangreichen Broadcastoperationen oder zu umfangreichen Verteilungen in STL_ALERT_EVENT_LOG protokolliert. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.
- In einem Schritt verarbeiten bestimmte Slices bedeutend mehr Zeilen als die anderen. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_REPORT.
- In einem Schritt dauert die Verarbeitung bei bestimmten Slices bedeutend länger als bei den anderen. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_REPORT.

Wenn keine der vorangehenden Bedingungen zutrifft, können Sie auch überprüfen, ob in einer der Tabellen in Ihrer Abfrage die Datenverteilung verzerrt ist, indem Sie die Abfrage in ausführen Identifizieren von Tabellen mit verzerrter Datenverteilung oder unsortierten Zeilen.

In diesem Fall können Sie versuchen, das Problem zu beheben, indem Sie sich die Verteilungsstile für die Tabellen in der Abfrage ansehen und prüfen, ob Verbesserungen vorgenommen werden können. Achten Sie dabei darauf, zwischen der Leistung dieser Abfrage und der von anderen wichtigen Abfragen und der gesamten Systemleistung, falls Sie Modifikationen vornehmen möchten. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

Ungenügende Speicherzuteilung für die Abfrage

Wenn Ihrer Abfrage nicht genügend Arbeitsspeicher zugeteilt ist, sehen Sie möglicherweise, dass für einen Schritt in SVL_QUERY_SUMMARY der Wert für `is_diskbased` auf „true“ gesetzt ist. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_SUMMARY.

Sie können in diesem Fall das Problem beheben, indem Sie der Abfrage mehr Speicherplatz zuteilen, indem Sie die Anzahl der für die Abfrage verwendeten Abfrageplätze vorübergehend erhöhen. Workload Management (WLM) reserviert die Plätze in einer Abfragewarteschlange entsprechend der für die Warteschlange festgelegten Parallelausführungsstufe (Gleichzeitigkeitsstufe).
Ein Beispiel: Eine Warteschlange mit einer Gleichzeitigkeitsstufe von 5 hat 5 Plätze. Der der Warteschlange zugewiesene Arbeitsspeicher wird gleichmäßig auf die Plätze verteilt. Wenn Sie einer Abfrage mehrere Plätze zuweisen, erhält die Abfrage den Arbeitsspeicher für alle Plätze, die ihr zugewiesen sind. Weitere Informationen dazu, wie Sie einer Abfrage vorübergehenden mehr Warteschlangenplätze zuweisen, Sie unter wlm_query_slot_count.

Suboptimale WHERE-Klausel

Wenn eine WHERE-Klausel dazu führt, dass umfangreiche Tabellenscans durchgeführt werden müssen, gibt es möglicherweise in SVL_QUERY_SUMMARY einen SCAN-Schritt in dem Segment

mit dem größten `maxtime`-Wert. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_SUMMARY.

Sie können dieses Problem beheben, indem Sie der Abfrage eine WHERE-Klausel auf der Grundlage der primären Sortierspalte der größten Tabelle hinzufügen. Dieser Ansatz hilft Ihnen, die für das Scannen der Tabellen benötigte Zeit zu minimieren. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift.

Nicht ausreichend restriktives Prädikat

Wenn das in der Abfrage verwendete Prädikat nicht ausreichend restriktiv ist, wird möglicherweise in SVL_QUERY_SUMMARY in dem Segment mit dem höchsten `maxtime`-Wert ein SCAN-Schritt angezeigt, der im Feld `rows` im Vergleich zu dem Wert im Feld `rows` im abschließenden RETURN-Schritt in der Abfrage einen sehr großen Wert aufweist. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_SUMMARY.

Sie können versuchen, dieses Problem zu beheben, indem Sie der Abfrage ein Prädikat hinzufügen oder indem Sie ein vorhandenes Prädikat restriktiver formulieren, um die Ausgabe einzuschränken.

Sehr große Ergebnismengen

Wenn Ihre Abfrage einen sehr großen Ergebnissatz zurückgibt, sollten Sie die Abfrage neu schreiben und UNLOAD verwenden, um die Ergebnisse zu Amazon S3 schreiben. Dieser Ansatz verbessert die Leistung des RETURN-Schritts durch Ausnutzung der Parallelverarbeitung. Weitere Informationen zum Überprüfen, ob sehr große Ergebnismengen zurückgegeben werden, finden Sie unter Verwenden der Ansicht SVL_QUERY_SUMMARY.

Große SELECT-Liste

Wenn Ihre Abfrage eine ungewöhnlich große SELECT-Liste erzeugt, wird dies in SVL_QUERY_SUMMARY möglicherweise dadurch ersichtlich, dass in einem bestimmten Schritt der `bytes`-Wert für einen Schritt im Vergleich zu dem `rows`-Wert ungewöhnlich groß ist. Der große Wert für `bytes` kann darauf hinweisen, dass eine große Anzahl an Spalten ausgewählt ist. Weitere Informationen finden Sie unter Verwenden der Ansicht SVL_QUERY_SUMMARY.

Um dieses Problem zu beheben, überprüfen Sie die bei der Abfrage ausgewählten Spalten daraufhin, ob sie aus der Abfrage entfernt werden können.

Diagnoseabfragen zur Abfrageoptimierung

Verwenden Sie die folgenden Abfragen, um Probleme bei Abfragen oder bei den zugrundeliegenden Tabellen zu identifizieren, die zu einer Beeinträchtigung der Abfrageleistung führen können. Wir empfehlen, diese Abfragen zusammen mit dem Abfrageoptimierungsprozess in Analysieren und Verbessern von Abfragen zu verwenden.

Themen

Identifizieren der Top-Kandidaten zur Optimierung unter den Abfragen

Die folgende Abfrage identifiziert die 50 zeitaufwändigsten Anweisungen, die in den letzten 7 Tagen ausgeführt wurden. Sie können die Ergebnisse verwenden, um Abfragen zu identifizieren, die ungewöhnlich lange dauern. Sie können auch Abfragen identifizieren, die häufig ausgeführt wurden (die mehrmals in der Ergebnismenge vorkommen). Diese Abfragen sind häufig gute Kandidaten für Abfragen, deren Optimierung die Systemleistung verbessern kann.

Diese Abfrage zählt außerdem die Warnereignisse für die einzelnen identifizierten Abfragen. Diese Warnmeldungen können Detailinformationen enthalten, mit denen Sie die Abfrageleistung verbessern können. Weitere Informationen finden Sie unter Überprüfen von Abfragewarnungen.

```
select trim(database) as db, count(query) as n_qry,
max(substring (qrytext,1,80)) as qrytext,
min(run_minutes) as "min" ,
max(run_minutes) as "max",
avg(run_minutes) as "avg", sum(run_minutes) as total,
max(query) as max_query_id,
max(starttime)::date as last_run,
sum(alerts) as alerts, aborted
from (select userid, label, stl_query.query,
trim(database) as database,
trim(querytxt) as qrytext,
```

```
md5(trim(querytxt)) as qry_md5,
starttime, endtime,
(datediff(seconds, starttime,endtime)::numeric(12,2))/60 as run_minutes,
alrt.num_events as alerts, aborted
from stl_query
left outer join
(select query, 1 as num_events from stl_alert_event_log group by query ) as alrt
on alrt.query = stl_query.query
where userid <> 1 and starttime >=  dateadd(day, -7, current_date))
group by database, label, qry_md5, aborted
order by total desc limit 50;
```

Identifizieren von Tabellen mit verzerrter Datenverteilung oder unsortierten Zeilen

Mit der folgenden Abfrage können Sie Tabellen identifizieren, die eine ungleichmäßige („verzerrte“) Datenverteilung oder einen hohen Prozentsatz an unsortierten Zeilen aufweisen.

Ein geringer `skew`-Wert weist darauf hin, das die Tabellendaten ordnungsgemäß verteilt sind. Wenn eine Tabelle für `skew` einen Wert von 4,00 oder höher hat, sollten Sie erwägen, den Datenverteilungsstil zu ändern. Weitere Informationen finden Sie unter Suboptimale Datenverteilung.

Wenn eine Tabelle für `pct_unsorted` einen Wert über 20 Prozent hat, sollten Sie ggf. den Befehl VACUUM ausführen. Weitere Informationen finden Sie unter Unsortierte oder falsch sortierte Zeilen.

Sie sollten auch die `mbytes`- und `pct_of_total`-Werte für jede Tabelle überprüfen. Diese Spalten geben die Größe einer Tabelle an, sowie den Prozentsatz des verfügbaren Bruttospeicherplatzes auf dem Datenträger, den die Tabelle verbraucht. Der Roh-Festplattenspeicherplatz schließt den Platz ein, der von Amazon Redshift für die interne Verwendung reserviert ist, ist also größer als die nominelle Festplattenkapazität, bei der es sich um den Festplattenspeicherplatz handelt, der dem Benutzer zur Verfügung steht. Verwenden Sie diese Informationen, um zu verifizieren, dass der verfügbare Speicherplatz auf dem Datenträger mindestens um den Faktor 2,5 größer ist als Ihre größte Tabelle. Dieser verfügbare Speicherplatz reicht dafür aus, dass das System bei der Verarbeitung von komplexen Abfragen Zwischenergebnisse auf den Datenträger schreiben kann.

```
select trim(pgn.nspname) as schema,
trim(a.name) as table, id as tableid,
decode(pgc.reldiststyle,0, 'even',1,det.distkey ,8,'all') as distkey,
 dist_ratio.ratio::decimal(10,4) as skew,
det.head_sort as "sortkey",
det.n_sortkeys as "#sks", b.mbytes,
```

```
decode(b.mbytes,0,0,((b.mbytes/part.total::decimal)*100)::decimal(5,2)) as
 pct_of_total,
decode(det.max_enc,0,'n','y') as enc, a.rows,
decode( det.n_sortkeys, 0, null, a.unsorted_rows ) as unsorted_rows ,
decode( det.n_sortkeys, 0, null, decode( a.rows,0,0, (a.unsorted_rows::decimal(32)/
a.rows)*100) )::decimal(5,2) as pct_unsorted
from (select db_id, id, name, sum(rows) as rows,
sum(rows)-sum(sorted_rows) as unsorted_rows
from stv_tbl_perm a
group by db_id, id, name) as a
join pg_class as pgc on pgc.oid = a.id
join pg_namespace as pgn on pgn.oid = pgc.relnamespace
left outer join (select tbl, count(*) as mbytes
from stv_blocklist group by tbl) b on a.id=b.tbl
inner join (select attrelid,
min(case attisdistkey when 't' then attname else null end) as "distkey",
min(case attsortkeyord when 1 then attname  else null end ) as head_sort ,
max(attsortkeyord) as n_sortkeys,
max(attencodingtype) as max_enc
from pg_attribute group by 1) as det
on det.attrelid = a.id
inner join ( select tbl, max(mbytes)::decimal(32)/min(mbytes) as ratio
from (select tbl, trim(name) as name, slice, count(*) as mbytes
from svv_diskusage group by tbl, name, slice )
group by tbl, name ) as dist_ratio on a.id = dist_ratio.tbl
join ( select sum(capacity) as  total
from stv_partitions where part_begin=0 ) as part on 1=1
where mbytes is not null
order by  mbytes desc;
```

Identifizieren von Abfragen mit verschachtelten Schleifen

Die folgende Abfrage identifiziert Abfragen, für die Warnereignisse zu verschachtelten Schleifen protokolliert wurden. Informationen zur Vorgehensweise, um Probleme bei verschachtelten Schleifen zu beheben, finden Sie unter Verschachtelte Schleife.

```
select query, trim(querytxt) as SQL, starttime
from stl_query
where query in (
select distinct query
from stl_alert_event_log
where event like 'Nested Loop Join in the query plan%')
```

```
order by starttime desc;
```

Überprüfen der Wartezeiten von Abfragen in Warteschlangen

Die folgende Abfrage zeigt, wie lange kürzlich ausgeführte Abfragen auf einen freien Platz in einer Abfragewarteschlange gewartet haben, bis sie ausgeführt wurden. Wenn Sie eine Tendenz erkennen, dass die Wartezeiten eher lang sind, können Sie die Abfragewarteschlangen modifizieren, um den Durchsatz zu verbessern. Weitere Informationen finden Sie unter Implementieren von manuellem WLM.

```
select trim(database) as DB , w.query,
substring(q.querytxt, 1, 100) as querytxt,  w.queue_start_time,
w.service_class as class, w.slot_count as slots,
w.total_queue_time/1000000 as queue_seconds,
w.total_exec_time/1000000 exec_seconds, (w.total_queue_time+w.total_Exec_time)/1000000
 as total_seconds
from stl_wlm_query w
left join stl_query q on q.query = w.query and q.userid = w.userid
where w.queue_start_Time >= dateadd(day, -7, current_Date)
and w.total_queue_Time > 0  and w.userid >1
and q.starttime >= dateadd(day, -7, current_Date)
order by w.total_queue_time desc, w.queue_start_time desc limit 35;
```

Überprüfen von Abfragewarnungen nach Tabelle

Die folgende Abfrage identifiziert Tabellen, für die Warnereignisse protokolliert wurden, und gibt an, welche Arten von Warnungen am häufigsten ausgelöst wurden.

Wenn in einer aufgeführten Tabelle der `minutes`-Wert für eine Zeile besonders hoch ist, überprüfen Sie, ob für die betreffende Tabelle routinemäßige Wartungsaufgaben durchgeführt werden können, beispielsweise durch Ausführen von ANALYZE oder VACUUM für die betreffende Tabelle.

Wenn der Wert `count` für eine Zeile hoch ist, gleichzeitig der Wert `table` aber Null, führen Sie für den zugehörigen `event`-Wert eine Abfrage über STL_ALERT_EVENT_LOG durch, um herauszufinden, warum die Warnung so oft ausgelöst wird.

```
select trim(s.perm_table_name) as table,
(sum(abs(datediff(seconds, s.starttime, s.endtime)))/60)::numeric(24,0) as minutes,
 trim(split_part(l.event,':',1)) as event,  trim(l.solution) as solution,
max(l.query) as sample_query, count(*)
```

```
from stl_alert_event_log as l
left join stl_scan as s on s.query = l.query and s.slice = l.slice
and s.segment = l.segment and s.step = l.step
where l.event_time >=  dateadd(day, -7, current_Date)
group by 1,3,4
order by 2 desc,6 desc;
```

Identifizieren von Tabellen mit fehlender Tabellenstatistik

Die folgende Abfrage zählt die Abfragen, die über Tabellen mit fehlender Tabellenstatistik ausgeführt wurden. Wenn diese Abfrage Zeilen zurückgibt, achten Sie auf den Wert von `plannode`, um die betreffende Tabelle zu ermitteln, und führen Sie dann über dieser Tabelle den Befehl ANALYZE aus.

```
select substring(trim(plannode),1,100) as plannode, count(*)
from stl_explain
where plannode like '%missing statistics%'
group by plannode
order by 2 desc;
```

Fehlerbehebung bei Abfragen

Dieser Abschnitt bietet eine kurze Übersicht über die Vorgehensweise zur Identifizierung und Behebung einiger der häufigsten und schwerwiegendsten Probleme, die bei Amazon-Redshift-Abfragen auftreten können.

Themen

- Verbindungsfehler
- Die Abfrage friert ein
- Die Abfrage dauert zu lange
- Das Laden der Daten schlägt fehl
- Der Ladevorgang dauert zu lange
- Die Ladedaten sind falsch
- Festlegen des JDBC-Parameters für die Abrufgröße

Die Lösungsvorschläge können als Ausgangspunkt für die Fehlerbehebung verwendet werden. Weitergehende Informationen finden Sie auch in den folgenden Ressourcen.

- Zugreifen auf Amazon-Redshift-Cluster und -Datenbanken
- Arbeiten mit automatischer Tabellenoptimierung
- Laden von Daten
- Tutorial: So laden Sie Daten aus Amazon S3

Verbindungsfehler

Die Verbindung der Abfrage kann aus den folgenden Gründen scheitern. Es werden die folgenden Ansätze zur Fehlerbehebung vorgeschlagen.

Client kann keine Verbindung zum Server herstellen

Wenn Sie SSL oder Serverzertifikate verwenden, entfernen Sie zunächst diese Komplexität, während Sie das Verbindungsproblem untersuchen. Wenn Sie das Problem behandelt haben, fügen Sie die SSL-Schicht bzw. die Serverzertifikate wieder ein. Weitere Informationen hierzu finden Sie unter Konfigurieren von Sicherheitsoptionen für Verbindungen im Verwaltungsleitfaden von Amazon Redshift.

Die Verbindung wird zurückgewiesen

Wenn Sie eine Fehlermeldung erhalten, dass eine benötigte Verbindung nicht hergestellt werden kann, bedeutet dies im Allgemeinen, dass ein Problem in Zusammenhang mit fehlenden Berechtigungen für den Zugriff auf den Cluster vorliegt. Weitere Informationen hierzu finden Sie unter Verbindung wird zurückgewiesen oder schlägt fehl im Verwaltungsleitfaden von Amazon Redshift.

Die Abfrage friert ein

Die Abfrage kann aus den folgenden Gründen einfrieren bzw. nicht mehr reagieren. Es werden die folgenden Ansätze zur Fehlerbehebung vorgeschlagen.

Die Datenbankverbindung wird getrennt

Verringern Sie den MTU-Wert (Maximum Transmission Unit, maximale Größe für Übertragungseinheiten). Der MTU-Wert legt die maximale Größe für die Übertragung eines Datenpakets in einem Ethernetframe über die Netzwerkverbindung in Bytes fest.

Weitere Informationen hierzu finden Sie unter Die Datenbankverbindung wird getrennt im Verwaltungsleitfaden von Amazon Redshift.

Zeitüberschreitung bei der Datenbankverbindung

Ihre Client-Verbindung mit der Datenbank scheint zu hängen oder Zeitüberschreitungen zu unterliegen, wenn lange Abfragen wie COPY-Befehle ausgeführt werden. Wenn dies der Fall ist, sehen Sie möglicherweise in der Amazon-Redshift-Konsole, dass die Abfrage abgeschlossen ist, aber das Client-Tool scheint die Abfrage noch nicht abgeschlossen zu haben. Je nachdem, wann die Verbindung unterbrochen wurde, fehlen möglicherweise Abfrageergebnisse oder sind unvollständig. Dieser Effekt kann auftreten, wenn Verbindungen im Leerlauf von zwischengelagerten Netzwerkkomponenten nach einer bestimmten Zeit getrennt werden. Weitere Informationen hierzu finden Sie unter Problem mit Firewall-Zeitüberschreitung im Verwaltungsleitfaden von Amazon Redshift.

Clientseitiger out-of-memory Fehler bei ODBC

Wenn Ihre Clientanwendung eine ODBC-Verbindung verwendet und Ihre Abfrage eine Ergebnismenge erzeugt, die so groß ist, dass sie nicht in den Arbeitsspeicher passt, können Sie die Ergebnismenge unter Verwendung eines Cursors an die Clientanwendung übermitteln. Weitere Informationen erhalten Sie unter DECLARE und Überlegungen in Bezug auf die Leistung bei Verwendung von Cursors.

Clientseitiger out-of-memory Fehler bei JDBC

Wenn Sie versuchen, große Ergebnismengen über eine JDBC-Verbindung abzurufen, können clientseitige out-of-memory Fehler auftreten. Weitere Informationen finden Sie unter Festlegen des JDBC-Parameters für die Abrufgröße.

Potenzielles Deadlock-Problem

Versuchen Sie bei möglichen Deadlock-Problemen, wie folgt vorzugehen:

- Zeigen Sie die Systemtabellen STV_LOCKS und STL_TR_CONFLICT an, um Konflikte zu identifizieren, bei denen mehrere Tabellen aktualisiert werden müssen.
- Verwenden Sie die Funktion PG_CANCEL_BACKEND, um die Abfragen, die den Konflikt verursachen, abzubrechen.
- Mit der Funktion PG_TERMINATE_BACKEND können Sie eine Sitzung beenden. Dabei werden alle Transaktionen, die zurzeit in der beendeten Sitzung ausgeführt werden, gezwungen, alle Sperren aufzuheben und für die Transaktionen ein Rollback auszuführen.

- Planen Sie gleichzeitige Schreibvorgänge sorgfältig. Weitere Informationen finden Sie unter Verwalten gleichzeitiger Schreiboperationen.

Die Abfrage dauert zu lange

Die Ausführung der Abfrage kann aus den folgenden Gründen zu viel Zeit in Anspruch nehmen. Es werden die folgenden Ansätze zur Fehlerbehebung vorgeschlagen.

Die Tabellen sind nicht optimiert

Legen Sie für die Tabellen den Sortierschlüssel, die Verteilungsstile und die Kodierung zur Komprimierung so fest, dass die Parallelverarbeitung optimal genutzt werden kann. Weitere Informationen finden Sie unter Arbeiten mit automatischer Tabellenoptimierung

Die Abfrage schreibt auf den Datenträger

Es kann vorkommen, dass Ihre Abfragen – möglicherweise nur für Teile der Abfrageausführung – Daten auf den Datenträger schreiben. Weitere Informationen finden Sie unter Verbessern der Abfrageleistung.

Die Abfrage muss auf den Abschluss anderer Abfragen warten

Sie können möglicherweise die Systemleistung insgesamt verbessern, indem Sie Warteschlangen erstellen und verschiedene Arten von Abfragen in geeignete Warteschlangen einstellen. Weitere Informationen finden Sie unter Implementierung von Workload Management.

Die Abfragen sind nicht optimiert

Analysieren Sie den mit EXPLAIN erzeugten Abfrageplan, um Möglichkeiten zu erkennen, wie Sie Abfragen umformulieren oder die Datenbank optimieren können. Weitere Informationen finden Sie unter Abfrageplan.

Die Abfrage benötigt zur Ausführung mehr Arbeitsspeicher

Wenn für eine bestimmte mehr Arbeitsspeicher benötigt wird, können Sie den verfügbaren Arbeitsspeicher vergrößern, indem Sie den Wert für heraufsetzen wlm_query_slot_count.

Die Datenbank benötigt zur Ausführung den Befehl VACUUM

Führen Sie den Befehl VACUUM aus, nachdem Sie eine große Anzahl von Zeilen hinzugefügt, gelöscht oder geändert haben, falls Sie Ihre Daten nicht entsprechend dem Sortierschlüssel sortiert

laden. Mit dem Befehl VACUUM werden Ihre Daten entsprechend der Sortierreihenfolge reorganisiert und die Leistung wiederhergestellt. Weitere Informationen finden Sie unter Bereinigen von Tabellen.

Zusätzliche Ressourcen zur Fehlerbehebung bei lang andauernden Abfragen

Im Folgenden finden Sie Themen zur Systemansicht und andere Dokumentationsabschnitte, die für die Abfrageoptimierung hilfreich sind:

- Die Systemansicht STV_INFLIGHT zeigt, welche Abfragen im Cluster ausgeführt werden. Es kann hilfreich sein, sie zusammen mit STV_RECENTS zu verwenden, um festzustellen, welche Abfragen gerade laufen oder kürzlich abgeschlossen wurden.
- SYS_QUERY_HISTORY ist für die Fehlerbehebung nützlich. Sie zeigt DDL- und DML-Abfragen mit relevanten Eigenschaften wie ihrem aktuellen Status, z. B. `running`oder `failed`, die Zeit, die jeweils für die Ausführung benötigt wurde, und ob eine Abfrage in einem Nebenläufigkeitsskalierungs-Cluster ausgeführt wurde.
- STL_QUERYTEXT erfasst den Abfragetext für SQL-Befehle. Zusätzlich gibt die Ansicht SVV_QUERY_INFLIGHT, die STL_QUERYTEXT mit STV_INFLIGHT verbindet, weitere Abfrage-Metadaten an.
- Ein Konflikt zwischen Transaktionssperren kann eine mögliche Ursache für Probleme mit der Abfrageleistung sein. Informationen zu Transaktionen, für die derzeit Tabellen gesperrt sind, finden Sie unter SVV_TRANSACTIONS.
- Identifizieren der Top-Kandidaten zur Optimierung unter den Abfragen stellt eine Fehlerbehebungsabfrage bereit, mit der Sie ermitteln können, welche kürzlich ausgeführten Abfragen am zeitaufwendigsten waren. Dies kann Ihnen helfen, Ihre Bemühungen auf Abfragen zu konzentrieren, die verbessert werden müssen.
- Wenn Sie sich eingehender mit der Abfrageverwaltung befassen und verstehen möchten, wie Abfragewarteschlangen verwaltet werden, finden Sie entsprechende Informationen unter Implementierung von Workload Management. Das Workload-Management ist eine erweiterte Funktion und wir empfehlen in den meisten Fällen ein automatisiertes Workload-Management.

Das Laden der Daten schlägt fehl

Das Laden der Daten kann aus den folgenden Gründen scheitern. Es werden die folgenden Ansätze zur Fehlerbehebung vorgeschlagen.

Die Datenquelle befindet sich in einer anderen AWS Region

Standardmäßig muss sich der im COPY-Befehl angegebene Amazon S3-Bucket oder die Amazon-DynamoDB-Tabelle in derselben - AWS Region wie der Cluster befinden. Wenn sich Ihre Daten und Ihr Cluster in verschiedenen Regionen befinden, wird eine Fehlermeldung wie die folgende angezeigt:

```
The bucket you are attempting to access must be addressed using the specified endpoint.
```

Stellen Sie nach Möglichkeit sicher, dass sich Ihr Cluster und Ihre Datenquelle in derselben Region befinden. Sie können mit der REGION-Option für den Befehl COPY eine abweichende Region angeben.

> **Note**
>
> Wenn sich Ihr Cluster und Ihre Datenquelle in unterschiedlichen - AWS Regionen befinden, fallen Datenübertragungskosten an. Außerdem ist die Latenz höher.

Der Befehl COPY schlägt fehl

Führen Sie eine Abfrage über STL_LOAD_ERRORS aus, um nach Fehlern bei einzelnen Ladevorgängen zu suchen. Weitere Informationen finden Sie unter STL_LOAD_ERRORS.

Der Ladevorgang dauert zu lange

Der Ladevorgang kann aus den folgenden Gründen zu viel Zeit in Anspruch nehmen. Es werden die folgenden Ansätze zur Fehlerbehebung vorgeschlagen.

Der Befehl COPY lädt Daten aus einer einzigen Datei

Aufteilen Ihrer Ladedaten in mehrere Dateien. Wenn Sie alle Daten aus einer Datei laden, kann Amazon Redshift die Daten nur seriell laden, was sehr viel langsamer ist. Die Anzahl der Dateien sollte ein Vielfaches der Anzahl der Slices in Ihrem Cluster sein. Außerdem sollten die Dateien etwa gleich groß und nach der Komprimierung zwischen 1 MB und 1 GB groß sein. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift.

Ladeoperation verwendet mehrere COPY-Befehle

Wenn Sie mehrere gleichzeitige COPY-Befehle verwenden, um eine Tabelle aus mehreren Dateien zu laden, ist Amazon Redshift gezwungen, einen serialisierten Ladevorgang durchzuführen, der bedeutend langsamer ist. Verwenden Sie in diesem Fall nur einen COPY-Befehl.

Die Ladedaten sind falsch

Die Operation COPY kann aus den folgenden Gründen die falschen Daten laden. Es werden die folgenden Ansätze zur Fehlerbehebung vorgeschlagen.

Es werden die falschen Dateien geladen

Wenn Sie für die Angabe der Datendateien ein Objektpräfix verwenden, werden möglicherweise die falschen Dateien gelesen. Verwenden Sie stattdessen eine Manifestdatei, um die zu ladenden Dateien genau anzugeben. Weitere Informationen finden Sie in den Abschnitten zur Option copy_from_s3_manifest_file für den Befehl COPY und in den COPY-Beispielen unter Example: COPY from Amazon S3 using a manifest.

Festlegen des JDBC-Parameters für die Abrufgröße

Der JDBC-Treiber stellt bei Abfragen alle Ergebnisse auf einmal zusammen. Wenn Sie versuchen, eine große Ergebnismenge über eine JDBC-Verbindung abzurufen, kann daher ein clientseitiger out-of-memory Fehler auftreten. Damit Ihr Client Ergebnissätze in Batches statt in einem einzigen all-or-nothing Abruf abrufen kann, legen Sie den Parameter JDBC-Abrufgröße in Ihrer Clientanwendung fest.

> **Note**
>
> Abrufgröße wird für ODBC nicht unterstützt

Legen Sie die Abrufgröße auf den höchsten Wert fest, der nicht zu Fehlern aufgrund von unzureichendem Arbeitsspeicher führt, um die Leistung zu optimieren. Wenn der Wert für die Abrufgröße kleiner gewählt wird, führt dies zu mehr Übertragungsvorgängen zwischen Server und Client, was die Ausführungszeit vergrößert. Der Server reserviert Ressourcen wie den WLM-Abfrageplatz und den zugehörigen Arbeitsspeicher, bis der Client die Ergebnismenge abruft oder die Abfrage abgebrochen wird. Wenn die Abrufgröße richtig eingestellt ist, werden diese Ressourcen schneller wieder freigegeben und sind für andere Abfragen verfügbar.

> **Note**
>
> Wenn Sie große Datensätze extrahieren müssen, empfehlen wir, eine UNLOAD-Anweisung zu verwenden, um die Daten an Amazon S3 zu übertragen. Wenn Sie UNLOAD verwenden,

arbeiten die Datenverarbeitungsknoten parallel, um die Übertragung der Daten zu beschleunigen.

Weitere Informationen zum Festlegen des Parameters für die JDBC-Abrufgröße finden Sie unter Getting results based on a cursor in der PostgreSQL-Dokumentation.

Implementierung von Workload Management

Sie können Workload Management (WLM) zur Definition mehrerer Abfragewarteschlangen und zur Weiterleitung von Abfragen zu den geeigneten Warteschlangen zur Laufzeit verwenden.

Es kann vorkommen, dass Sie mehrere Sitzungen oder Benutzer haben, die gleichzeitig Abfragen ausführen. In diesen Fällen können einige Abfragen über einen längeren Zeitraum Clusterressourcen beanspruchen und die Leistung anderer Abfragen beeinträchtigen. Nehmen Sie beispielsweise an, eine Benutzergruppe übermittelt gelegentlich komplexe, lang andauernde Abfragen, die Zeilen aus mehreren großen Tabellen auswählen und sortieren. Eine andere Gruppe übermittelt häufig kurze Abfragen, die nur wenige Zeilen aus einer oder zwei Tabellen auswählen und nur wenige Sekunden in Anspruch nehmen. In dieser Situation müssen die kürzeren Abfragen möglicherweise in einer Warteschlange warten, bis eine lang andauernde Abfrage abgeschlossen ist. WLM hilft, diese Situation zu steuern.

Sie können Amazon Redshift WLM so konfigurieren, dass es entweder mit automatischem oder manuellem WLM ausgeführt wird.

Automatisches WLM

Um den Systemdurchsatz zu maximieren und Ressourcen effektiv zu verwenden, können Sie Amazon Redshift aktivieren. So können Sie die Aufteilung von Ressourcen für die Ausführung gleichzeitiger Abfragen mit automatischem WLM verwalten. Das automatische WLM verwaltet die für die Ausführung von Abfragen benötigten Ressourcen. Amazon Redshift legt fest, wie viele Abfragen gleichzeitig ausgeführt werden und wie viel Arbeitsspeicher den einzelnen verteilten Abfragen zugewiesen wird. Sie können das automatische WLM über die Amazon-Redshift-Konsole aktivieren, indem Sie Switch WLM mode (WLM-Modus wechseln) und anschließend Auto WLM (Automatisches WLM) auswählen. Bei Wahl dieser Option werden bis zu acht Warteschlangen für die Verwaltung von Abfragen verwendet. Die Felder Memory (Arbeitsspeicher) und Concurrency on main (Nebenläufigkeit für Haupt) werden dabei auf Auto (Automatisch) festgelegt. Sie können eine Priorität angeben, die die geschäftliche Priorität des Workloads angibt, oder die Benutzer, die den einzelnen Warteschlangen zugeordnet sind. Die Standardpriorität der Abfragen ist auf Normal (Normal) festgelegt. Informationen zum Ändern der Priorität von Abfragen in einer Warteschlange finden Sie unter Abfragepriorität. Weitere Informationen finden Sie unter Implementieren von automatischem WLM.

Während der Laufzeit können Sie Abfragen nach Benutzer- oder Abfragegruppe an diese Warteschlangen leiten. Sie können auch eine Abfrageüberwachungsregel (Query Monitoring Rule, QMR) konfigurieren, um lang andauernde Abfragen zu begrenzen.

Mit Nebenläufigkeitsskalierung und automatischem WLM können Sie eine praktisch unbegrenzt Zahl gleichzeitiger Benutzer und Abfragen bei einer konsistent schnellen Abfrageleistung unterstützen. Weitere Informationen finden Sie unter Arbeiten mit Nebenläufigkeitsskalierung.

> **Note**
>
> Wir empfehlen für die Verwaltung von Abfrageressourcen die Erstellung einer Parametergruppe und die Verwendung des automatischen WLM. Für Einzelheiten zur Migration vom manuellen zum automatischen WLM vgl. Migration vom manuellen WLM zum automatischen WLM.

Manuelles WLM

Alternativ können Sie die Systemleistung und die Benutzerfreundlichkeit verwalten, indem Sie Ihre WLM-Konfiguration ändern und separate Warteschlangen für lang andauernde und kürzer laufende Abfragen erstellen. Während der Laufzeit können Sie Abfragen nach Benutzer- oder Abfragegruppe an diese Warteschlangen leiten. Unter Verwendung der Amazon-Redshift-Konsole können Sie diese manuelle Konfiguration aktivieren, indem Sie zu Manual WLM (manuelles WLM) wechseln. Bei dieser Option bestimmen Sie die Warteschlangen, die für die Abfrageverwaltung verwendet werden, sowie die Werte für die Felder Memory (Arbeitsspeicher) und Concurrency on main (Nebenläufigkeit für Haupt). Anhand einer manuellen Konfiguration können Sie bis zu acht Abfragewarteschlangen konfigurieren und die Zahl der in diesen Warteschlangen gleichzeitig möglichen Abfragen einstellen.

Sie können Regeln einrichten, um Abfragen auf der Grundlage des jeweiligen Benutzers oder von Ihnen angegebener Beschriftungen an bestimmte Warteschlangen weiterzuleiten. Sie können auch festlegen, wie viel Speicher jeder Warteschlange zugewiesen wird, so dass große Abfragen in Warteschlangen mit mehr Speicher als bei anderen Warteschlangen ausgeführt werden. Sie können auch eine Abfrageüberwachungsregel (Query Monitoring Rule, QMR) konfigurieren, um lang andauernde Abfragen zu begrenzen. Weitere Informationen finden Sie unter Implementieren von manuellem WLM.

> **Note**
>
> Wir empfehlen, für manuelle WLM-Abfragewartschlangen insgesamt höchstens 15 Abfrage-Slots zu konfigurieren. Weitere Informationen finden Sie unter Nebenläufigkeitsstufe.

Einschränkungen in Bezug auf die WLM-Warteschlangen

Beachten Sie, dass Sie bei einer manuellen WLM-Konfiguration einer Warteschlange maximal 50 Slots zuweisen können. Dies bedeutet jedoch nicht, dass ein Amazon-Redshift-Cluster bei einer automatischen WLM-Konfiguration immer 50 Abfragen gleichzeitig ausführt. Dies kann sich je nach Speicherbedarf oder anderen Arten von Ressourcenzuweisungen im Cluster ändern.

Anwendungsfälle für automatisches WLM und manuelles WLM

Verwenden Sie automatisches WLM, wenn Sie Amazon Redshift aktivieren möchten. So können Sie die Aufteilung von Ressourcen für die Ausführung gleichzeitiger Abfragen verwalten. Die Verwendung von automatischem WLM führt häufig zu einem höheren Durchsatz als manuelles WLM. Mit automatischem WLM können Sie Abfrageprioritäten für Workloads in einer Warteschlange definieren. Weitere Informationen zur Abfragepriorität finden Sie unter Abfragepriorität.

Verwenden Sie manuelles WLM, wenn Sie mehr Kontrolle über die Parallelität haben möchten.

Themen

Modifizieren der WLM-Konfiguration

Die einfachste Möglichkeit zur Modifizierung der WLM-Konfiguration besteht in der Verwendung der Amazon-Redshift-Konsole. Sie können auch die AWS CLI oder die Amazon Redshift Redshift-API verwenden.

Wenn Sie für Ihr Cluster zwischen automatischem und manuellem WLM wechseln, wird für Ihr Cluster der Zustand `pending reboot` festgelegt. Diese Änderung wird erst nach dem nächsten Neustart des Clusters wirksam.

Detaillierte Informationen zum Ändern von WLM-Konfigurationen finden Sie unter Konfigurieren von Workload Management im Amazon-Redshift-Verwaltungshandbuch.

Migration vom manuellen WLM zum automatischen WLM

Um den Systemdurchsatz zu maximieren und die Ressourcen so effektiv wie möglich zu nutzen, sollten Sie für Ihre Warteschlangen das automatische WLM festlegen. Sie sollten dabei den folgenden Ansatz befolgen, um einen problemlosen Übergang vom manuellen WLM zum automatischen WLM sicherzustellen.

Um vom manuellen WLM auf automatisches WLM umzustellen und Abfrageprioritäten zu verwenden, sollten Sie eine neue Parametergruppe erstellen und diese Parametergruppe anschließend an Ihren Cluster anfügen. Weitere Informationen finden Sie unter Amazon Redshift Parameter Groups (Amazon-Redshift-Parametergruppen) im Amazon-Redshift-Verwaltungshandbuch.

> **Important**
>
> Um die Parametergruppe zu ändern oder vom manuellen zum automatischen WLM zu wechseln, ist ein Neustart des Clusters erforderlich. Weitere Informationen finden Sie unter Dynamische und statische WLM-Konfigurationseigenschaften.

Betrachten wir ein Beispiel mit drei manuellen WLM-Warteschlangen. Es gibt jeweils eine Warteschlange für einen ETL-Workload, einen Analytics-Workload und einen Data Science-Workload. Der ETL-Workload wird alle 6 Stunden und der Analytics-Workload während des gesamten Tages ausgeführt. Der Data Science-Workload kann jederzeit ausgeführt werden und dabei einen Spitzenwert erreichen. Bei Verwendung des manuellen WLM geben Sie den Arbeitsspeicher und den Nebenläufigkeitsfaktor an, den die einzelnen Workload-Warteschlangen erhalten, basierend auf

Ihrem Verständnis der Bedeutung der einzelnen Workloads für das Geschäft. Die Festlegung von Arbeitsspeicher und Nebenläufigkeit ist nicht nur schwierig, sondern führt auch zu einer statischen Partitionierung der Cluster-Ressourcen und damit zu Verschwendung, wenn nur ein Teilsatz der Workloads ausgeführt wird.

Sie können automatisches WLM mit Abfrageprioritäten verwenden, um die relativen Prioritäten der Workloads anzugeben, und damit die zuvor beschriebenen Probleme vermeiden. Für dieses Beispiel führen Sie die folgenden Schritte aus:

- Erstellen Sie eine neue Parametergruppe und wechseln Sie zum Modus Auto WLM (Automatisches WLM).
- Fügen Sie für jeden der drei Workloads Warteschlangen hinzu: ETL-Workload, Analytics-Workload und Data Science-Workload. Verwenden Sie dieselben Benutzergruppen für jeden Workload, die im Manual WLM-Modus verwendet wurden.
- Legen Sie die Priorität für den ETL-Workload auf `High`, für den Analytics-Workload auf `Normal` und für den Data Science-Workload auf `Low` fest. Diese Prioritäten geben die geschäftlichen Prioritäten für die verschiedenen Workloads oder Benutzergruppen wieder.
- Optional können Sie die Nebenläufigkeitsskalierung für die Analytics- oder Data-Science-Warteschlange aktivieren, sodass Abfragen in diesen Warteschlangen eine konsistente Leistung erhalten, auch wenn der ETL-Workload alle 6 Stunden ausgeführt wird.

Wenn Sie Abfrageprioritäten verwenden und nur der Analytics-Workload für das Cluster ausgeführt wird, kann er das gesamte System nutzen. Dies führt zu einem hohen Durchsatz bei besserer Systemauslastung. Wenn jedoch der ETL-Workload gestartet wird, erhält dieser Priorität, da er eine höhere Priorität hat. Abfragen, die als Teil des ETL-Workloads ausgeführt werden, erhalten Priorität bei der Annahme. Nach der Annahme werden ihnen zudem bevorzugt Ressourcen zugeteilt. Daher wird der ETL-Workload planbar ausgeführt, unabhängig davon, welche weiteren Aufgaben im System ausgeführt werden. Die planbare Leistung eines Workloads mit hoher Priorität geht zu Lasten anderer Workloads mit niedrigerer Priorität. Diese benötigen eine längere Zeit für die Ausführung, da ihre Abfragen warten müssen, bis wichtigere Abfragen abgeschlossen sind. Ein anderer Grund für die längere Ausführungszeit besteht darin, dass sie einen kleineren Anteil an den Ressourcen erhalten, wenn sie gleichzeitig mit Abfragen höherer Priorität ausgeführt werden. Die von Amazon Redshift verwendeten Planungsalgorithmen sorgen dafür, dass Abfragen mit niedrigerer Priorität nicht angehalten werden, sondern weiter ausgeführt werden, wenn auch langsamer.

Note

- Im automatischen WLM ist das Zeitüberschreitungsfeld nicht verfügbar. Sie müssen stattdessen die QMR-Regel verwenden, `query_execution_time`. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.
- Die QMR-Aktion, HOP, kann nicht auf automatisches WLM angewendet werden. Sie müssen stattdessen die Aktion `change priority` verwenden. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.
- Cluster verwenden automatische WLM-Warteschlangen und manuelle WLM-Warteschlangen unterschiedlich, was zu Verwechslungen mit Ihren Konfigurationen führen kann. Beispielsweise können Sie die Prioritätseigenschaft in automatischen WLM-Warteschlangen konfigurieren, nicht jedoch in manuellen WLM-Warteschlangen. Vermeiden Sie daher die gleichzeitige Verwendung von automatischen und manuellen WLM-Warteschlangen innerhalb einer Parametergruppe. Erstellen Sie stattdessen eine neue Parametergruppe, wenn Sie zum automatischen WLM migrieren.

Implementieren von automatischem WLM

Anhand des automatischen Workload Management (WLM) verwaltet Amazon Redshift die Abfragenebenläufigkeit sowie die Speicherzuweisung. Sie können bis zu acht Warteschlangen mit den Serviceklassen-IDs 100 bis 107 erstellen. Jede Warteschlange besitzt eine Priorität. Weitere Informationen finden Sie unter Abfragepriorität.

Das automatische WLM legt die Menge der Ressourcen fest, die von Abfragen benötigt werden, und passt die Nebenläufigkeit entsprechend dem Workload an. Befinden sich Abfragen im System, die große Mengen an Ressourcen benötigen (z. B. Hash-Joins zwischen großen Tabellen), ist die Nebenläufigkeit geringer. Werden leichtere Abfragen (wie Einfügen, Löschen, Scannen oder einfache Aggregationen) übermittelt, ist die Nebenläufigkeit höher.

Das automatische WLM unterscheidet sich von der Short Query Acceleration (SQA) und wertet Abfragen auf andere Weise aus. Automatisches WLM und SQA arbeiten zusammen, damit kurz ausgeführte und einfache Abfragen abgeschlossen werden können, auch wenn ressourcenintensive Abfragen aktiv sind. Weitere Informationen zu SQA finden Sie unter Arbeiten mit Short Query Acceleration.

Amazon Redshift ermöglicht automatisches WLM über Parametergruppen:

- Wenn Ihre Cluster die Standardparametergruppe verwenden, aktiviert Amazon Redshift das automatische WLM für sie.
- Wenn Ihre Cluster benutzerdefinierte Parametergruppen verwenden, können Sie diese Cluster für die Unterstützung des automatischen WLM konfigurieren. Sie sollten eine eigene Parametergruppe für Ihre Konfiguration des automatischen WLM verwenden.

Um WLM zu konfigurieren, bearbeiten Sie den Parameter `wlm_json_configuration` in einer Parametergruppe, die mit einem oder mehreren Clustern verknüpft werden kann. Weitere Informationen finden Sie unter Modifizieren der WLM-Konfiguration.

Sie können Abfragewarteschlangen innerhalb der WLM-Konfiguration definieren. Sie können der WLM-Standardkonfiguration weitere Abfragewarteschlangen (bis zu acht Benutzerwarteschlangen) hinzufügen. Für jede Abfragewarteschlange kann Folgendes konfiguriert werden:

- Priorität
- Nebenläufigkeitsskalierungsmodus
- Benutzergruppen
- Abfragegruppen
- Abfrageüberwachungsregeln

Priorität

Sie können die relative Wichtigkeit der Abfragen in einem Workload definieren, indem Sie einen Prioritätswert festlegen. Die Priorität wird für eine Warteschlange angegeben und von allen Abfragen geerbt, die mit der Warteschlange verknüpft sind. Weitere Informationen finden Sie unter Abfragepriorität.

Nebenläufigkeitsskalierungsmodus

Bei aktivierter Nebenläufigkeitsskalierung fügt Amazon Redshift automatisch zusätzliche Cluster-Kapazität hinzu, wenn diese benötigt wird, um eine gestiegene Zahl von gleichzeitigen Lese- und Schreibabfragen zu verarbeiten. Ihre Benutzer sehen stets die jeweils aktuellen Daten, unabhängig davon, ob die Abfragen auf dem Haupt- oder einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden.

Sie können verwalten, welche Abfragen an das Nebenläufigkeitsskalierungs-Cluster gesendet werden, indem Sie WLM-Warteschlangen konfigurieren. Wenn die Nebenläufigkeitsskalierung

für eine Warteschlange aktiviert ist, werden entsprechend qualifizierte Abfragen an das Nebenläufigkeitsskalierungs-Cluster übermittelt, anstatt in einer Warteschlange zu warten. Weitere Informationen finden Sie unter Arbeiten mit Nebenläufigkeitsskalierung.

Benutzergruppen

Sie können einer Warteschlange einen Satz von Benutzergruppen zuweisen, indem Sie die einzelnen Namen der Benutzergruppen angeben oder Platzhalter verwenden. Wenn ein Mitglied einer aufgeführten Benutzergruppe eine Abfrage ausführt, wird diese in der entsprechenden Warteschlange ausgeführt. Es gibt keine Grenze für die Anzahl der Benutzergruppen, die einer Warteschlange zugewiesen werden können. Weitere Informationen finden Sie unter Zuweisen von Abfragen zu Warteschlangen auf der Grundlage von Benutzergruppen.

Abfragegruppen

Sie können einer Warteschlange einen Satz von Abfragegruppen zuweisen, indem Sie die einzelnen Namen der Abfragegruppen angeben oder Platzhalter verwenden. Eine Abfragegruppe ist einfach eine Beschriftung. Während der Laufzeit können Sie die Beschriftung der Abfragegruppe einer Serie von Abfragen zuweisen. Alle einer aufgeführten Abfragegruppe zugewiesenen Abfragen werden in der entsprechenden Warteschlange ausgeführt. Es gibt keine Grenze für die Anzahl der Abfragegruppen, die einer Warteschlange zugewiesen werden können. Weitere Informationen finden Sie unter Zuweisen einer Abfrage zu einer Abfragegruppe.

Platzhalter

Wenn in der WLM-Warteschlangenkonfiguration Platzhalter aktiviert sind, können Sie Benutzergruppen und Abfragegruppen einzeln oder mithilfe von Platzhaltern im Unix-Shell-Typ einer Warteschlange zuweisen. Der Musterabgleich beachtet die Groß- und Kleinschreibung.

So entspricht beispielsweise das Platzhalterzeichen „*“ einer beliebigen Anzahl von Zeichen. Wenn Sie dementsprechend `dba_*` zur Liste der Benutzergruppen für eine Warteschlange hinzufügen, wird dieser Warteschlange jede Benutzerabfrage zugeordnet, die zu einer Gruppe gehört, deren Name mit `dba_` beginnt. Beispiele sind `dba_admin` oder `DBA_primary`. Das Platzhalterzeichen „?“ entspricht einem einzelnen Zeichen. Wenn die Warteschlange also die Benutzergruppe `dba?1`enthält, dann passen die Benutzergruppen mit Namen `dba11` und `dba21`, `dba12` jedoch nicht.

Standardmäßig sind Platzhalter nicht aktiviert.

Abfrageüberwachungsregeln

Sie können auf Metriken basierende Leistungsgrenzen für WLM-Warteschlangen definieren und angeben, welche Aktion durchgeführt werden soll, wenn eine Abfrage diese Grenzwerte überschreitet. So können Sie etwa für eine für kurze Abfragen dedizierte Warteschlange eine Regel erstellen, die Abfragen abbricht, die länger als 60 Sekunden ausgeführt werden. Zur Nachverfolgung schlecht gestalteter Abfragen können Sie eine weitere Regel verwenden, die Abfragen mit eingebetteten Schleifen protokolliert. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

Überprüfen auf automatisches WLM

Führen Sie die folgende Abfrage aus, um zu überprüfen, ob das automatische WLM aktiviert ist. Wenn für die Abfrage mindestens eine Zeile zurückgegeben wird, ist das automatische WLM aktiviert.

```
select * from stv_wlm_service_class_config
where service_class >= 100;
```

Die folgende Abfrage zeigt die Anzahl der Abfragen, die die einzelnen Abfragewarteschlangen durchlaufen haben (Service-Klasse). Sie zeigt zudem die durchschnittliche Ausführungsdauer, die Anzahl der Abfragen mit einer Wartezeit ab der 90. Perzentile und die durchschnittliche Wartezeit. Automatische WLM-Abfragen verwenden die Serviceklassen 100 bis 107.

```
select final_state, service_class, count(*), avg(total_exec_time),
percentile_cont(0.9) within group (order by total_queue_time), avg(total_queue_time)
from stl_wlm_query where userid >= 100 group by 1,2 order by 2,1;
```

Um herauszufinden, welche Abfragen von dem automatischen WLM ausgeführt und erfolgreich abgeschlossen wurden, führen Sie die folgende Abfrage aus.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class >= 100 and a.final_state = 'Completed'
order by b.query desc limit 5;
```

Abfragepriorität

Nicht alle Abfragen sind gleich wichtig und häufig ist die Leistung eines bestimmten Workloads oder Benutzersatzes von höherer Wichtigkeit. Wenn Sie automatisches WLM aktiviert haben, können Sie

die relative Wichtigkeit der Abfragen in einem Workload definieren, indem Sie einen Prioritätswert festlegen. Die Priorität wird für eine Warteschlange angegeben und von allen Abfragen geerbt, die mit der Warteschlange verknüpft sind. Sie verknüpfen Abfragen mit einer Warteschlange, indem Sie der Warteschlange Benutzer- und Abfragegruppen zuordnen. Sie können die folgenden Prioritäten festlegen (von höchster zu niedrigster Priorität):

1. `HIGHEST`
2. `HIGH`
3. `NORMAL`
4. `LOW`
5. `LOWEST`

Administratoren verwenden diese Prioritäten, um die relative Wichtigkeit von Workloads anzugeben, wenn es Abfragen mit unterschiedlicher Priorität gibt, die dieselben Ressourcen benötigen. Amazon Redshift verwendet die Priorität bei der Annahme von Abfragen für das System und zur Festlegung der Menge der Ressourcen, die einer Abfrage zugeordnet werden. Standardmäßig wird die Priorität von Abfragen auf festgelegt `NORMAL`.

Für Superuser ist zusätzlich die Priorität `CRITICAL` verfügbar. Diese Priorität hat einen höheren Rang als `HIGHEST`. Um diese Priorität festzulegen, können Sie die Funktionen CHANGE_QUERY_PRIORITY, CHANGE_SESSION_PRIORITY und CHANGE_USER_PRIORITY verwenden. Um einem Datenbankbenutzer die Berechtigung zur Nutzung dieser Funktionen zu erteilen, können Sie eine gespeicherte Prozedur erstellen und diesem Benutzer die entsprechende Berechtigung erteilen. Ein Beispiel finden Sie unter CHANGE_SESSION_PRIORITY.

> **Note**
>
> Es kann jeweils nur eine Abfrage mit der Priorität `CRITICAL` ausgeführt werden.

Betrachten wir ein Beispiel, in dem ein Workload zum Extrahieren, Transformieren und Laden (Extract, Transform, Load; ETL) eine höhere Priorität als der Analytics-Workload hat. Der ETL-Workload wird alle sechs Stunden ausgeführt. Der Analytics-Workload wird während des gesamten Tages ausgeführt. Wenn nur der Analytics-Workload für das Cluster ausgeführt wird, kann er das gesamte System nutzen und einen hohen Durchsatz bei optimaler Systemnutzung erzielen. Wenn jedoch der ETL-Workload gestartet wird, erhält dieser Priorität, da er eine höhere Priorität hat. Abfragen, die als Teil des ETL-Workloads ausgeführt werden, erhalten Priorität bei der Annahme.

Nach der Annahme werden ihnen die Ressourcen bevorzugt zugeteilt. Daher wird der ETL-Workload planbar ausgeführt, unabhängig davon, welche weiteren Aufgaben im System ausgeführt werden. Auf diese Weise wird eine planbare Leistung bereitgestellt und die Administratoren können geschäftlichen Benutzern Service Level Agreements (SLAs) bereitstellen.

Innerhalb des jeweiligen Clusters geht die planbare Leistung eines Workloads mit hoher Priorität zu Lasten anderer Workloads mit einer niedrigeren Priorität. Workloads mit einer niedrigeren Priorität benötigen für die Ausführung möglicherweise mehr Zeit, da ihre Abfragen warten, bis wichtigere Abfragen abgeschlossen sind. Ein anderer Grund für die längere Ausführungszeit kann darin bestehen, dass sie einen kleineren Anteil an den Ressourcen erhalten, wenn sie gleichzeitig mit Abfragen höherer Priorität ausgeführt werden. Abfragen mit einer niedrigeren Priorität werden jedoch nicht angehalten, sondern lediglich langsamer ausgeführt.

Im vorherigen Beispiel kann der Administrator für den Analytics-Workload die Nebenläufigkeitsskalierung aktivieren. Hierdurch kann dieser Workload seinen Durchsatz bewahren, auch wenn der ETL-Workload mit höherer Priorität ausgeführt wird.

Konfigurieren der Warteschlangenpriorität

Wenn Sie automatisches WLM aktiviert haben, besitzt jede Warteschlange einen Prioritätswert. Abfragen werden auf der Grundlage von Benutzer- und Abfragegruppen an Warteschlangen geleitet. Beginnen Sie mit der Warteschlangenpriorität `NORMAL`. Sie können eine höhere oder niedrigere Priorität festlegen, abhängig von dem Workload, der mit den Benutzer- und Abfragegruppen des Workloads verknüpft ist.

Sie können die Priorität einer Warteschlange in der Amazon-Redshift-Konsole ändern. Sie können in der Amazon-Redshift-Konsole auf der Seite Workload Management (Workload-Verwaltung) die Warteschlangen anzeigen und Warteschlangeneigenschaften wie Priority (Priorität) bearbeiten. Um die Priorität über die CLI oder API-Operationen festzulegen, verwenden Sie den Parameter `wlm_json_configuration`. Weitere Informationen finden Sie unter Workload-Management-Konfiguration im Amazon-Redshift-Verwaltungshandbuch.

Im folgenden `wlm_json_configuration`-Beispiel werden drei Benutzergruppen definiert (`ingest`, `reporting` und `analytics`). Die Abfragen, die von den Benutzern dieser Gruppen übermittelt werden, werden jeweils mit den Prioritäten `highest`, `normal` und `low` ausgeführt.

```
[
    {
        "user_group": [
            "ingest"
```

```
        ],
        "priority": "highest",
        "queue_type": "auto"
    },
    {
        "user_group": [
            "reporting"
        ],
        "priority": "normal",
        "queue_type": "auto"
    },
    {
        "user_group": [
            "analytics"
        ],
        "priority": "low",
        "queue_type": "auto",
        "auto_wlm": true
    }
]
```

Ändern der Abfragepriorität mittels Abfrageüberwachungsregeln

Abfrageüberwachungsregeln (Query Monitoring Rules, QMR) ermöglichen Ihnen die Änderung der Priorität einer Abfrage basierend auf ihrem Verhalten während der Ausführung. Hierzu geben Sie zusätzlich zu einer Aktion das Prioritätsattribut in einem QMR-Prädikat an. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

Sie können beispielsweise eine Regel definieren, nach der Abfragen mit der Priorität `high` abgebrochen werden, wenn sie länger als 10 Minuten ausgeführt werden.

```
"rules" :[
  {
    "rule_name":"rule_abort",
    "predicate":[
      {
        "metric_name":"query_cpu_time",
        "operator":">",
        "value":600
      },
      {
        "metric_name":"query_priority",
        "operator":"=",
```

```
        "value":"high"
      }
    ],
    "action":"abort"
  }
]
```

Sie können auch eine Regel definieren, nach der die Priorität aller Abfragen mit der Priorität `lowest` in `normal` geändert wird, wenn sie mehr als 1 TB auf der Festplatte belegen.

```
"rules":[
  {
    "rule_name":"rule_change_priority",
    "predicate":[
      {
        "metric_name":"query_temp_blocks_to_disk",
        "operator":">",
        "value":1000000
      },
      {
        "metric_name":"query_priority",
        "operator":"=",
        "value":"normal"
      }
    ],
    "action":"change_query_priority",
    "value":"lowest"
  }
]
```

Konfigurieren der Abfragepriorität

Um die Priorität für wartende und ausgeführte Abfragen anzuzeigen, zeigen Sie die Spalte `query_priority` in der Systemtabelle „stv_wlm_query_state“ an.

```
query    | service_cl | wlm_start_time             | state            | queue_time |
 query_priority
---------+------------+----------------------------+------------------+------------
+----------------
2673299  | 102        | 2019-06-24 17:35:38.866356 | QueuedWaiting    | 265116     |
 Highest
```

```
2673236  | 101        | 2019-06-24 17:35:33.313854 | Running          | 0          |
 Highest
2673265  | 102        | 2019-06-24 17:35:33.523332 | Running          | 0          |
 High
2673284  | 102        | 2019-06-24 17:35:38.477366 | Running          | 0          |
 Highest
2673288  | 102        | 2019-06-24 17:35:38.621819 | Running          | 0          |
 Highest
2673310  | 103        | 2019-06-24 17:35:39.068513 | QueuedWaiting    | 62970      |
 High
2673303  | 102        | 2019-06-24 17:35:38.968921 | QueuedWaiting    | 162560     |
 Normal
2673306  | 104        | 2019-06-24 17:35:39.002733 | QueuedWaiting    | 128691     |
 Lowest
```

Um die Abfragepriorität abgeschlossener Abfragen anzuzeigen, zeigen Sie die Spalte `query_priority` in der Systemtabelle „stl_wlm_query“ an.

```
select query, service_class as svclass, service_class_start_time as starttime,
 query_priority
from stl_wlm_query order by 3 desc limit 10;
```

```
  query  | svclass |         starttime          |    query_priority
---------+---------+----------------------------+----------------------
 2723254 |     100 | 2019-06-24 18:14:50.780094 | Normal
 2723251 |     102 | 2019-06-24 18:14:50.749961 | Highest
 2723246 |     102 | 2019-06-24 18:14:50.725275 | Highest
 2723244 |     103 | 2019-06-24 18:14:50.719241 | High
 2723243 |     101 | 2019-06-24 18:14:50.699325 | Low
 2723242 |     102 | 2019-06-24 18:14:50.692573 | Highest
 2723239 |     101 | 2019-06-24 18:14:50.668535 | Low
 2723237 |     102 | 2019-06-24 18:14:50.661918 | Highest
 2723236 |     102 | 2019-06-24 18:14:50.643636 | Highest
```

Um den Durchsatz Ihres Workloads zu optimieren, kann Amazon Redshift die Priorität der von Benutzern übermittelten Abfragen ändern. Amazon Redshift verwendet fortschrittliche Machine-Learning-Algorithmen, um zu ermitteln, wann diese Optimierung Ihrem Workload zugutekommt, und wendet sie automatisch an, wenn alle folgenden Bedingungen erfüllt sind.

- Automatisches WLM ist aktiviert.

- Es ist nur eine WLM-Warteschlange definiert.
- Sie haben keine Abfrageüberwachungsregeln (Query Monitoring Rules, QMRS) definiert, die die Abfragepriorität festlegen. Dazu gehören zum Beispiel die QMR-Metrik `query_priority` oder die QMR-Aktion `change_query_priority`. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

Implementieren von manuellem WLM

Bei manuellem WLM können Sie Systemleistung und Benutzererfahrung verwalten, indem Sie die WLM-Konfiguration so ändern, dass separate Warteschlangen für lang andauernde und kürzere Abfragen erstellt werden.

Wenn Benutzer in Amazon Redshift Abfragen ausführen, werden diese zu Abfragewarteschlangen geleitet. Jede Abfragewarteschlange enthält eine Anzahl von Abfrageslots. Jeder Warteschlange ist ein Teil des verfügbaren Speicherplatzes des Clusters zugewiesen. Der Speicher einer Warteschlange wird unter den Abfrageslots der Warteschlange aufgeteilt. Sie können Amazon Redshift aktivieren, um die Nebenläufigkeit von Abfragen über das automatische WLM zu verwalten. Weitere Informationen finden Sie unter Implementieren von automatischem WLM.

Sie können WLM-Eigenschaften auch pro Abfragewarteschlange konfigurieren. Hierzu geben Sie an, wie der Arbeitsspeicher auf die einzelnen Slots verteilt wird und wie Abfragen während der Laufzeit an bestimmte Warteschlangen geleitet werden können. Darüber hinaus können Sie die WLM-Eigenschaften entsprechend konfigurieren, um lang andauernde Abfragen abzubrechen.

Standardmäßig konfiguriert Amazon Redshift die folgenden Abfragewarteschlangen:

- Eine Superuser-Warteschlange

  Die Superuser-Warteschlange ist ausschließlich für Superusers reserviert und kann nicht konfiguriert werden. Verwenden Sie diese Warteschlange nur dann, wenn Sie Abfragen ausführen müssen, die sich auf das System auswirken, oder für Fehlerbehebungszwecke. Nutzen Sie diese Warteschlange beispielsweise, wenn Sie eine lang andauernde Abfrage eines Benutzers abbrechen oder wenn Sie der Datenbank Benutzer hinzufügen müssen. Verwenden Sie sie nicht für Routineabfragen. Die Warteschlange wird nicht in der Konsole angezeigt, sie erscheint aber in den Systemtabellen in der Datenbank als fünfte Spalte. Um eine Abfrage in der Superuser-Warteschlange ausführen zu können, muss ein Benutzer als Superuser angemeldet sein und die Abfrage in der vordefinierten `superuser`-Abfragegruppe ausführen.
- Eine Standard-Benutzer-Warteschlange

Die Standard-Warteschlange ist anfänglich so konfiguriert, dass sie fünf Abfragen gleichzeitig ausführen kann. Beim manuellen WLM können Sie die Eigenschaften für die Nebenläufigkeit, das Timeout und die Speicherzuweisung für die Standard-Warteschlange ändern, jedoch keine Benutzer- oder Abfragegruppen angeben. Die Standard-Warteschlange muss die letzte Warteschlange in der WLM-Konfiguration sein. Alle Abfragen, die nicht zu anderen Warteschlangen geleitet werden, werden in der Standard-Warteschlange ausgeführt.

Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Die WLM-Konfiguration ist ein bearbeitbarer Parameter (`wlm_json_configuration`) in einer Parametergruppe, der mit einem oder mehreren Clustern verbunden werden kann. Weitere Informationen finden Sie unter Workload-Management-Konfiguration im Amazon-Redshift-Verwaltungshandbuch.

Sie können der WLM-Standardkonfiguration weitere Abfragewarteschlangen (bis zu acht Benutzerwarteschlangen) hinzufügen. Für jede Abfragewarteschlange kann Folgendes konfiguriert werden:

- Nebenläufigkeitsskalierungsmodus
- Nebenläufigkeitsstufe
- Benutzergruppen
- Abfragegruppen
- Zu verwendender WLM-Speicherprozentsatz
- WLM-Timeout
- WLM-Abfragewarteschlangen-Hopping
- Abfrageüberwachungsregeln

Nebenläufigkeitsskalierungsmodus

Bei aktivierter Nebenläufigkeitsskalierung fügt Amazon Redshift automatisch zusätzliche Cluster-Kapazität hinzu, wenn diese benötigt wird, um eine gestiegene Zahl von gleichzeitigen Lese- und Schreibabfragen zu verarbeiten. Benutzern werden stets die jeweils aktuellen Daten angezeigt, unabhängig davon, ob die Abfragen auf dem Haupt- oder einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden.

Sie können verwalten, welche Abfragen an das Nebenläufigkeitsskalierungs-Cluster gesendet werden, indem Sie WLM-Warteschlangen konfigurieren. Wenn die Nebenläufigkeitsskalierung

für eine Warteschlange aktiviert ist, werden entsprechend qualifizierte Abfragen an das Nebenläufigkeitsskalierungs-Cluster übermittelt, anstatt in einer Warteschlange zu warten. Weitere Informationen finden Sie unter Arbeiten mit Nebenläufigkeitsskalierung.

Nebenläufigkeitsstufe

Abfragen in einer Warteschlange werden gleichzeitig ausgeführt, bis die für die jeweilige Warteschlange festgelegte Anzahl der WLM-Abfrage-Slots oder die Nebenläufigkeitsstufe erreicht wird. Weitere Abfragen warten dann in der Warteschlange.

> **Note**
>
> Die WLM-Nebenläufigkeitsstufe entspricht nicht der Anzahl der gleichzeitigen Benutzerverbindungen zu einem Cluster. Weitere Informationen finden Sie unter Herstellen von Verbindungen mit einem Cluster im Amazon-Redshift-Verwaltungshandbuch.

Bei einer automatischen WLM-Konfiguration (empfohlen) ist die Nebenläufigkeitsstufe auf Auto festgelegt. Amazon Redshift weist Abfragen dynamisch Speicher zu, wobei anschließend bestimmt wird, wie viele Abfragen gleichzeitig ausgeführt werden sollen. Dies basiert auf den Ressourcen, die für laufende Abfragen und Abfragen in der Warteschlange benötigt werden. Automatisches WLM ist nicht konfigurierbar. Weitere Informationen finden Sie unter Implementieren von automatischem WLM.

Bei einer manuellen WLM-Konfiguration weist Amazon Redshift jeder Warteschlange statisch eine feste Speichermenge zu. Der Speicher der Warteschlange wird gleichmäßig auf die Abfrage-Slots verteilt. Zur Veranschaulichung: Wenn einer Warteschlange 20 % des Speichers eines Clusters zugewiesen sind und die Warteschlange 10 Slots umfasst, werden jeder Abfrage 2 % des Cluster-Speichers zugewiesen. Die Speicherzuweisung bleibt fest, unabhängig davon, wie viele Abfragen gleichzeitig ausgeführt werden. Aufgrund dieser festen Speicherzuweisung kann es vorkommen, dass Abfragen, die bei 5 Slots vollständig im Speicher ausgeführt werden, Zwischenergebnisse auf die Festplatte schreiben, wenn die Anzahl der Slots auf 20 erhöht wird. In diesem Fall verringert sich der Anteil jeder Abfrage am Speicher der Warteschlange von 1/5 auf 1/20. Die zusätzlichen Festplatten-I/O-Operationen können sich nachteilig auf die Leistung auswirken.

Die maximale Slot-Anzahl für alle benutzerdefinierten Warteschlangen ist 50. Dadurch wird die Gesamtzahl der Slots für alle Warteschlangen, einschließlich der Standardwarteschlange, begrenzt. Die einzige Warteschlange, die nicht dem Limit unterliegt, ist die reservierte Superuser-Warteschlange.

Standardmäßig lautet die Nebenläufigkeitsstufe von manuellen WLM-Warteschlangen 5. Ihr Workload kann in manchen Fällen von einer höheren Nebenläufigkeitsstufe profitieren, zu, Beispiel:

- Wenn zahlreiche kleine Abfragen auf länger dauernde Abfragen warten müssen, erstellen Sie eine separate Warteschlange mit einer höheren Slot-Anzahl und weisen Sie die kleineren Abfragen dieser Warteschlange zu. Einer Warteschlange mit einer höheren Nebenläufigkeitsstufe hat weniger Speicherplatz für jeden Abfrageslot, die kleineren Abfragen beanspruchen jedoch auch weniger Speicher.

  > **Note**
  >
  > Wenn Sie Short Query Acceleration (SQA) aktivieren, priorisiert WLM automatisch kurze Abfragen gegenüber längeren Abfragen. In diesem Fall benötigen Sie für die meisten Workflows keine eigene Wartschlange für kurze Abfragen. Weitere Informationen finden Sie unter Arbeiten mit Short Query Acceleration.

- Wenn mehrere Abfragen vorhanden sind, die auf Daten auf einem einzigen Slice zugreifen, richten Sie eine separate WLM-Warteschlange ein, um diese Abfragen gleichzeitig auszuführen. Amazon Redshift weist gleichzeitige Abfragen verschiedenen Slices zu, wodurch mehrere Abfragen parallel auf mehreren Slices ausgeführt werden können. Zum Beispiel: Wenn eine Abfrage ein einfaches Aggregat mit einem Prädikat auf dem Verteilungsschlüssel ist, befinden sich die Daten für die Abfrage auf einem einzelnen Slice.

Beispiel für manuelles WLM

Dieses Beispiel zeigt anhand eines Szenarios mit einfachem manuellem WLM, wie Slots und Speicher zugewiesen werden können. Sie implementieren manuelles WLM mit den folgenden drei Warteschlangen:

- Warteschlange für die Datenerfassung – Diese wird für die Erfassung von Daten eingerichtet. Der Warteschlange sind 20 % des Cluster-Speichers zugewiesen und sie verfügt über 5 Slots. Anschließend können 5 Abfragen gleichzeitig in der Warteschlange ausgeführt werden, denen jeweils 4 % des Speichers zugewiesen werden.
- Warteschlange für Datenwissenschaftler – Diese Warteschlange ist für speicherintensive Abfragen konzipiert. Der Warteschlange sind 40 % des Cluster-Speichers zugewiesen und sie verfügt über 5 Slots. Anschließend können 5 Abfragen gleichzeitig ausgeführt werden, denen jeweils 8 % des Speichers zugewiesen werden.

- Standardwarteschlange – Diese Warteschlange ist für die Mehrheit der Benutzer in der Organisation konzipiert. Hierzu gehören Vertriebs- und Buchhaltungsteams, die in der Regel kürzere oder mittellange Abfragen haben, die nicht kompliziert sind. Dieser Warteschlange werden 40 % des Cluster-Speichers zugewiesen und sie verfügt über 40 Slots. 40 Abfragen können gleichzeitig in dieser Warteschlange ausgeführt werden, wobei jeder Abfrage 1 % des Speichers zugewiesen wird. Dies ist die maximale Anzahl von Slots, die dieser Warteschlange zugewiesen werden können, da das Limit für alle Warteschlangen bei 50 liegt.

Wenn Sie automatisches WLM ausführen und Ihr Workload die parallele Ausführung von mehr als 15 Abfragen erfordert, empfehlen wir, die Nebenläufigkeitsskalierung einzuschalten. Dies liegt daran, dass eine Erhöhung der Anzahl von Abfrage-Slots auf mehr als 15 zu einem Wettbewerb um Systemressourcen führen und den Gesamtdurchsatz eines einzelnen Clusters einschränken könnte. Mit der Nebenläufigkeitsskalierung können Sie Hunderte von Abfragen bis zu einer konfigurierten Anzahl von Nebenläufigkeitsskalierungs-Clustern parallel ausführen. Die Anzahl der Nebenläufigkeitsskalierungs-Cluster wird von max_concurrency_scaling_clusters gesteuert. Weitere Hinweise zur Parallelitätsskalierung finden Sie unter Arbeiten mit Nebenläufigkeitsskalierung.

Weitere Informationen finden Sie unter Verbessern der Abfrageleistung.

Benutzergruppen

Sie können einer Warteschlange einen Satz von Benutzergruppen zuweisen, indem Sie die einzelnen Namen der Benutzergruppen angeben oder Platzhalter verwenden. Wenn ein Mitglied einer aufgeführten Benutzergruppe eine Abfrage ausführt, wird diese in der entsprechenden Warteschlange ausgeführt. Es gibt keine Grenze für die Anzahl der Benutzergruppen, die einer Warteschlange zugewiesen werden können. Weitere Informationen finden Sie unter Zuweisen von Abfragen zu Warteschlangen auf der Grundlage von Benutzergruppen.

Abfragegruppen

Sie können einer Warteschlange einen Satz von Abfragegruppen zuweisen, indem Sie die einzelnen Namen der Abfragegruppen angeben oder Platzhalter verwenden. Eine Abfragegruppe ist einfach eine Beschriftung. Während der Laufzeit können Sie die Beschriftung der Abfragegruppe einer Serie von Abfragen zuweisen. Alle einer aufgeführten Abfragegruppe zugewiesenen Abfragen werden in der entsprechenden Warteschlange ausgeführt. Es gibt keine Grenze für die Anzahl der Abfragegruppen, die einer Warteschlange zugewiesen werden können. Weitere Informationen finden Sie unter Zuweisen einer Abfrage zu einer Abfragegruppe.

Platzhalter

Wenn in der WLM-Warteschlangenkonfiguration Platzhalter aktiviert sind, können Sie Benutzergruppen und Abfragegruppen einzeln oder mithilfe von Platzhaltern im Unix-Shell-Typ einer Warteschlange zuweisen. Der Musterabgleich beachtet die Groß- und Kleinschreibung.

So entspricht beispielsweise das Platzhalterzeichen „*“ einer beliebigen Anzahl von Zeichen. Wenn Sie dementsprechend `dba_*` zur Liste der Benutzergruppen für eine Warteschlange hinzufügen, wird dieser Warteschlange jede Benutzerabfrage zugeordnet, die zu einer Gruppe gehört, deren Name mit `dba_` beginnt. Beispiele sind `dba_admin` oder `DBA_primary`, . Das Platzhalterzeichen „?“ entspricht einem einzelnen Zeichen. Wenn die Warteschlange also die Benutzergruppe `dba?1`enthält, dann passen die Benutzergruppen mit Namen `dba11` und `dba21`, `dba12` jedoch nicht.

Platzhalter sind standardmäßig deaktiviert.

Zu verwendender WLM-Speicherprozentsatz

In einer automatischen WLM-Konfiguration ist der Speicherprozentsatz auf eingestellt **`auto`**. Weitere Informationen finden Sie unter Implementieren von automatischem WLM.

In einer manuellen WLM-Konfiguration können Sie zur Angabe der Menge des verfügbaren Speicherplatzes, der einer Abfrage zugewiesen wird, den Parameter `WLM Memory Percent to Use` einstellen. Standardmäßig wird jeder benutzerdefinierten Warteschlange ein gleicher Anteil des für benutzerdefinierte Abfragen verfügbaren Speicherplatzes zugewiesen. Wenn Sie beispielsweise vier benutzerdefinierte Warteschlangen haben, erhält jede Warteschlange 25 Prozent des verfügbaren Speicherplatzes. Die Superuser-Warteschlange hat ihren eigenen zugewiesenen Speicherplatz und kann nicht modifiziert werden. Zur Änderung der Zuweisung weisen Sie jeder Warteschlange einen ganzzahligen Speicherplatzprozentsatz zu, bis insgesamt 100 Prozent erreicht sind. Nicht zugewiesener Speicherplatz wird von Amazon Redshift verwaltet und kann vorübergehend einer Warteschlange zugewiesen werden, wenn diese zusätzlichen Speicherplatz zur Verarbeitung anfragt.

Wenn Sie beispielsweise vier Warteschlangen konfigurieren, können Sie den Speicherplatz wie folgt zuweisen: 20 Prozent, 30 Prozent, 15 Prozent, 15 Prozent. Die verbleibenden 20 Prozent sind nicht zugewiesen und werden von dem Service verwaltet.

WLM-Timeout

WLM-Timeout (`max_execution_time`) ist veraltet. Erstellen Sie stattdessen eine Abfrageüberwachungsregel (Query Monitoring Rule, QMR) `query_execution_time` zur

Begrenzung der Ausführungszeit für eine Abfrage. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

Um die Zeit zu begrenzen, die Abfragen in einer WLM-Warteschlange nutzen können, können Sie den WLM-Timeout-Wert für jede Warteschlange einstellen. Der Timeout-Parameter gibt die Zeit in Millisekunden an, für die Amazon Redshift auf die Ausführung einer Abfrage wartet, bevor es diese beendet oder zur nächsten Warteschlange springt. Der Timeout-Wert basiert auf der Abfrageausführungszeit und beinhaltet nicht die in einer Warteschlange verbrachte Zeit.

WLM führt Hopping möglichst für CREATE TABLE AS (CTAS)-Anweisungen und schreibgeschützte Abfragen wie SELECT-Anweisungen aus. Abfragen, für die kein Hopping ausgeführt werden kann, werden abgebrochen. Weitere Informationen finden Sie unter WLM-Abfragewarteschlangen-Hopping.

Das WLM-Timeout gilt nicht für eine Abfrage, die den Status „returning“ erreicht hat. Um den Status einer Abfrage anzuzeigen, vgl. die Systemtabelle STV_WLM_QUERY_STATE. COPY-Anweisungen und Wartungsoperationen wie ANALYZE und VACUUM unterliegen keinen WLM-Timeouts.

Die Funktion des WLM-Timeout ist ähnlich wie beim statement_timeout Konfigurationsparameter. Der Unterschied besteht darin, dass, während der Konfigurationsparameter `statement_timeout` für den gesamten Cluster gilt, das WLM-Timeout nur für eine einzelne Warteschlange in der WLM-Konfiguration spezifisch ist.

Wenn statement_timeout auch angegeben wird, wird die untere der Anweisungen „statement_timeout“ und „WLM-Timeout (max_execution_time)“ verwendet.

Abfrageüberwachungsregeln

Sie können auf Metriken basierende Leistungsgrenzen für WLM-Warteschlangen definieren und angeben, welche Aktion durchgeführt werden soll, wenn eine Abfrage diese Grenzwerte überschreitet. So können Sie etwa für eine für kurze Abfragen dedizierte Warteschlange eine Regel erstellen, die Abfragen abbricht, die länger als 60 Sekunden ausgeführt werden. Zur Nachverfolgung schlecht gestalteter Abfragen können Sie eine weitere Regel verwenden, die Abfragen mit eingebetteten Schleifen protokolliert. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

WLM-Abfragewarteschlangen-Hopping

Eine Abfrage kann aufgrund eines WLM-Timeouts oder einer Abfrageüberwachungsregel (Query Monitoring Rule, QMR) gehoppt werden. Sie können Abfragen-Hopping nur in einer manuellen WLM-Konfiguration ausführen.

Wenn für eine Abfrage Hopping ausgeführt wird, versucht WLM, die Abfrage auf Grundlage der WLM-Abfragezuweisungsregeln an die nächste übereinstimmende Warteschlange weiterzuleiten. Wenn die Abfrage keiner anderen Warteschlangendefinition entspricht, wird sie abgebrochen. Sie wird nicht der Standardwarteschlange zugewiesen.

WLM-Timeout-Aktionen

In der folgenden Tabelle ist das Verhalten der verschiedenen Abfragetypen bei einem WLM-Timeout zusammengefasst.

Abfragetyp	Action
INSERT, UPDATE und DELETE	Abbrechen
Benutzerdefinierte Funktionen (User-defined functions, UDFs)	Abbrechen
UNLOAD	Abbrechen
COPY	Ausführung wird fortgesetzt
Wartungsoperationen	Ausführung wird fortgesetzt
Schreibgeschützte Abfragen mit dem Status `returning`	Ausführung wird fortgesetzt
Schreibgeschützte Abfragen mit dem Status `running`	Neuzuweisung oder Neustart
CREATE TABLE AS (CTAS), SELECT INTO	Neuzuweisung oder Neustart

Warteschlangen-Hopping aufgrund von WLM-Timeouts

WLM hoppt die folgenden Abfragen bei einem Timeout:

- Schreibgeschützte Abfragen wie SELECT-Anweisungen mit dem WLM-Status `running`. Sie finden den WLM-Status einer Abfrage in der Spalte STATE in der Systemtabelle STV_WLM_QUERY_STATE.

- CREATE TABLE AS (CTAS)-Anweisungen. WLM-Warteschlangen-Hopping unterstützt sowohl benutzerdefinierte als auch vom System generierte CTAS-Anweisungen.
- SELECT INTO-Anweisungen

Abfragen, die keinen WLM-Timeouts unterliegen, werden weiterhin in der ursprünglichen Warteschlange ausgeführt, bis sie abgeschlossen sind. Die folgenden Abfragetypen unterliegen keinen WLM-Timeouts:

- COPY-Anweisungen
- Wartungsoperationen wie ANALYZE und VACUUM
- Schreibgeschützte Abfragen wie SELECT-Anweisungen mit dem WLM-Status `returning`. Sie finden den WLM-Status einer Abfrage in der Spalte STATE in der Systemtabelle STV_WLM_QUERY_STATE.

Abfragen, die bei einem WLM-Timeout für Hopping qualifiziert sind, werden beim Timeout beendet. Die folgenden Abfragetypen sind bei einem WLM-Timeout nicht für Hopping qualifiziert:

- INSERT-, UPDATE- und DELETE-Anweisungen
- UNLOAD-Anweisungen
- Benutzerdefinierte Funktionen (User-defined functions, UDFs)

Nach WLM-Timeout neu zugewiesene und neu gestartete Abfragen

Wenn für eine Abfrage Hopping ausgeführt, aber keine passende Warteschlange gefunden wird, wird die Abfrage abgebrochen.

Wenn für eine Abfrage Hopping ausgeführt und eine passende Warteschlange gefunden wird, versucht WLM, die Abfrage einer neuen Warteschlange zuzuweisen. Wenn eine Abfrage nicht neu zugewiesen werden kann, wird sie in der neuen Wartschlange wie nachfolgend beschrieben neu gestartet.

Eine Abfrage wird nur dann neu zugewiesen, wenn alle folgenden Bedingungen zutreffen:

- Es wird eine passende Warteschlange gefunden.

- Die neue Warteschlange enthält ausreichend freie Slots zum Ausführen der Warteschlange. Eine Abfrage benötigt möglicherweise mehrere Slots, wenn der wlm_query_slot_count-Parameter einen höheren Wert als 1 enthält.
- Die neue Warteschlange verfügt über mindestens soviel Arbeitsspeicher wie derzeit von der Abfrage verwendet.

Wenn die Abfrage neu zugewiesen wird, wird sie in der neuen Warteschlange weiter ausgeführt. Da Zwischenergebnisse beibehalten werden, gibt es nur minimale Auswirkungen auf die gesamte Ausführungszeit.

Wenn die Abfrage nicht neu zugewiesen werden kann, wird sie abgebrochen und in der neuen Warteschlange neu gestartet. Zwischenergebnisse werden gelöscht. Die Abfrage wird in die Warteschlange eingereiht und ausgeführt, sobald ausreichend Slots verfügbar sind.

QMR-Hopping-Aktionen

In der folgenden Tabelle ist das Verhalten der verschiedenen Abfragetypen bei einer QMR-Hopping-Aktion zusammengefasst.

Abfragetyp	Action
COPY	Ausführung wird fortgesetzt
Wartungsoperationen	Ausführung wird fortgesetzt
Benutzerdefinierte Funktionen (User-defined functions, UDFs)	Ausführung wird fortgesetzt
UNLOAD	Neuzuweisung oder Fortsetzung der Ausführung
INSERT, UPDATE und DELETE	Neuzuweisung oder Fortsetzung der Ausführung
Schreibgeschützte Abfragen mit dem Status `returning`	Neuzuweisung oder Fortsetzung der Ausführung
Schreibgeschützte Abfragen mit dem Status `running`	Neuzuweisung oder Neustart

Abfragetyp	Action
CREATE TABLE AS (CTAS), SELECT INTO	Neuzuweisung oder Neustart

Um herauszufinden, ob eine von QMR gehoppte Abfrage neu zugewiesen, neu gestartet oder abgebrochen wurde, fragen Sie die STL_WLM_RULE_ACTION-Systemprotokolltabelle ab.

Nach QMR-Hopping-Aktionen neu zugewiesene und neu gestartete Abfragen

Wenn für eine Abfrage Hopping ausgeführt, aber keine passende Warteschlange gefunden wird, wird die Abfrage abgebrochen.

Wenn für eine Abfrage Hopping ausgeführt und eine passende Warteschlange gefunden wird, versucht WLM, die Abfrage einer neuen Warteschlange zuzuweisen. Wenn eine Abfrage nicht neu zugewiesen werden kann, wird sie in der neuen Wartschlange neu gestartet oder in der ursprünglichen Warteschlange weiter ausgeführt, wie nachfolgend beschrieben.

Eine Abfrage wird nur dann neu zugewiesen, wenn alle folgenden Bedingungen zutreffen:

- Es wird eine passende Warteschlange gefunden.
- Die neue Warteschlange enthält ausreichend freie Slots zum Ausführen der Warteschlange. Eine Abfrage benötigt möglicherweise mehrere Slots, wenn der wlm_query_slot_count-Parameter einen höheren Wert als 1 enthält.
- Die neue Warteschlange verfügt über mindestens soviel Arbeitsspeicher wie derzeit von der Abfrage verwendet.

Wenn die Abfrage neu zugewiesen wird, wird sie in der neuen Warteschlange weiter ausgeführt. Da Zwischenergebnisse beibehalten werden, gibt es nur minimale Auswirkungen auf die gesamte Ausführungszeit.

Wenn eine Abfrage nicht neu zugewiesen werden kann, wird sie entweder neu gestartet oder in der ursprünglichen Warteschlange weiter ausgeführt. Wenn die Abfrage neu gestartet wird, wird sie abgebrochen und in der neuen Warteschlange neu gestartet. Zwischenergebnisse werden gelöscht. Die Abfrage wartet in der Warteschlange und wird ausgeführt, sobald ausreichend Slots verfügbar sind.

Tutorial: Konfigurieren von manuellen Workload-Management(WLM)-Warteschlangen

Übersicht

Wir empfehlen, automatisches Workload-Management (WLM) in Amazon Redshift zu konfigurieren. Mehr Informationen über automatisches WLM finden Sie unter Implementierung von Workload Management. Wenn Sie jedoch mehrere WLM-Warteschlangen benötigen, werden Sie in diesem Tutorial durch den Prozess des Konfigurierens manuellen Workload-Managements (WLM) in Amazon Redshift geführt. Durch die Konfiguration von manuellem WLM können Sie die Abfrageleistung und die Ressourcenzuweisung in Ihrem Cluster verbessern.

Amazon Redshift leitet Benutzerabfragen zur Verarbeitung an Warteschlangen weiter. WLM legt fest, wie diese Abfragen zu den Warteschlangen geleitet werden. Standardmäßig hat Amazon Redshift zwei Warteschlangen für Abfragen: eine für Superuser und eine für (normale) Benutzer. Die Superuser-Warteschlange kann nicht konfiguriert werden und verarbeitet jeweils nur eine Abfrage zur gleichen Zeit. Sie sollten diese Warteschlange nur für Fehlerbehebungszwecke reservieren. Die Benutzerwarteschlange kann bis zu fünf Abfragen gleichzeitig verarbeiten, Sie können dies jedoch konfigurieren, indem Sie bei Bedarf die Gleichzeitigkeitsstufe der Warteschlange ändern.

Wenn mehrere Benutzer Abfragen gegen die Datenbank ausführen, kann es sein, dass eine andere Konfiguration effizienter ist. Zum Beispiel: Wenn einige Benutzer ressourcenintensive Operationen wie etwa VACUUM durchführen, wirken sich diese möglicherweise negativ auf weniger intensive Abfragen, wie etwa Berichte, aus. Si sollten dann vielleicht weitere Warteschlangen hinzufügen und diese für unterschiedliche Workloads konfigurieren.

Geschätzte Zeit: 75 Minuten

Geschätzte Kosten: 50 Cent

Voraussetzungen

Sie benötigen einen Amazon-Redshift-Cluster, die TICKIT-Beispieldatenbank und das Amazon-Redshift-RSQL-Client-Tool. Wenn diese noch nicht eingerichtet sind, navigieren Sie zum Handbuch Erste Schritte mit Amazon Redshift und zu Amazon Redshift RSQL.

Sections

- Abschnitt 1: Erläuterung des standardmäßigen Verhaltens der Warteschlangenverarbeitung

- Abschnitt 2: Modifizieren der Konfiguration der WLM-Abfragewarteschlange
- Abschnitt 3: Weiterleiten von Abfragen zu Warteschlangen auf der Grundlage von Benutzergruppen und Abfragegruppen
- Abschnitt 4: Verwenden von wlm_query_slot_count zur temporären Übergehung der Gleichzeitigkeitsstufe in einer Warteschlange
- Abschnitt 5: Bereinigen Ihrer Ressourcen

Abschnitt 1: Erläuterung des standardmäßigen Verhaltens der Warteschlangenverarbeitung

Bevor Sie mit der Konfiguration von manuellem WLM beginnen, sollten Sie das standardmäßige Verhalten der Abfrageverarbeitung in Amazon Redshift verstanden haben. In diesem Abschnitt erstellen Sie zwei Datenbankansichten, die Informationen aus verschiedenen Systemtabellen zurückgeben. Dann führen Sie einige Testabfragen aus, um zu sehen, wie Abfragen standardmäßig weitergeleitet werden. Weitere Informationen zu Systemtabellen finden Sie unter Referenz zu Systemtabellen und Ansichten.

Schritt 1: Erstellen der Ansicht WLM_QUEUE_STATE_VW

In diesem Schritt erstellen Sie eine Ansicht mit dem Namen WLM_QUEUE_STATE_VW. Diese Ansicht gibt Informationen aus den folgenden Systemtabellen aus.

- STV_WLM_CLASSIFICATION_CONFIG
- STV_WLM_SERVICE_CLASS_CONFIG
- STV_WLM_SERVICE_CLASS_STATE

Sie verwenden diese Ansicht während des gesamten Tutorials, um zu sehen, was mit Warteschlangen geschieht, nachdem Sie die WLM-Konfiguration geändert haben. Die folgende Tabelle beschreibt die Daten, die die Ansicht WLM_QUEUE_STATE_VW ausgibt.

Spalte	Beschreibung
Warteschlange	Die Nummer der Zeile, die für eine Warteschlange steht. Die Warteschl angennummer bestimmt die Reihenfolge der Warteschlangen in der Datenbank.

Spalte	Beschreibung
description	Ein Wert, der angibt, ob die Warteschlange nur für bestimmte Benutzergruppen, bestimmte Abfragegruppen oder für alle Arten von Abfragen verfügbar ist.
slots	Die Anzahl der der Warteschlange zugewiesenen Slots.
mem	Der der Warteschlange zugewiesene Speicherplatz, in MB pro Slot.
max_execution_time	Die Zeitspanne, für die eine Abfrage laufen kann, bevor sie beendet wird.
Benutzer_*	Ein Wert, der angibt, ob in der WLM-Konfiguration Platzhalterzeichen für Benutzergruppen zulässig sind.
query_*	Ein Wert, der angibt, ob in der WLM-Konfiguration Platzhalterzeichen für Abfragegruppen zulässig sind.
queued	Die Anzahl der Abfragen, die in der Warteschlange auf ihre Verarbeitung warten.
executing	Die Anzahl der Abfragen, die derzeit ausgeführt werden.
executed	Die Anzahl der Abfragen, die ausgeführt wurden.

So erstellen Sie die Ansicht WLM_QUEUE_STATE_VW

1. Öffnen Sie Amazon Redshift RSQL und stellen Sie eine Verbindung zu Ihrer TICKIT-Beispieldatenbank her. Wenn Sie diese Datenbank nicht haben, finden Sie weitere Informationen unter Voraussetzungen.
2. Führen Sie die folgende Abfrage aus, um die Ansicht WLM_QUEUE_STATE_VW zu erstellen.

```
create view WLM_QUEUE_STATE_VW as
select (config.service_class-5) as queue
, trim (class.condition) as description
, config.num_query_tasks as slots
, config.query_working_mem as mem
, config.max_execution_time as max_time
, config.user_group_wild_card as "user_*"
, config.query_group_wild_card as "query_*"
```

```
, state.num_queued_queries queued
, state.num_executing_queries executing
, state.num_executed_queries executed
from
STV_WLM_CLASSIFICATION_CONFIG class,
STV_WLM_SERVICE_CLASS_CONFIG config,
STV_WLM_SERVICE_CLASS_STATE state
where
class.action_service_class = config.service_class
and class.action_service_class = state.service_class
and config.service_class > 4
order by config.service_class;
```

3. Führen Sie die folgende Abfrage aus, um die Informationen in der Ansicht anzuzeigen.

```
select * from wlm_queue_state_vw;
```

Nachfolgend sehen Sie ein Beispielergebnis.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(querytype: any)	5	836	0	false	false	0	1	160

Schritt 2: Erstellen der Ansicht WLM_QUERY_STATE_VW

In diesem Schritt erstellen Sie eine Ansicht mit dem Namen WLM_QUERY_STATE_VW. Diese Ansicht gibt Informationen aus der Systemtabelle STV_WLM_QUERY_STATE aus.

Sie verwenden diese Ansicht während des gesamten Tutorials zur Überwachung der laufenden Abfragen. Die folgende Tabelle beschreibt die Daten, die die Ansicht WLM_Query_STATE_VW ausgibt.

Spalte	Beschreibung
query	Die Abfrage-ID.
Warteschlange	Die Nummer der Warteschlange.
slot_count	Die Anzahl der der Abfrage zugewiesenen Slots.
start_time	Der Zeitpunkt des Beginns der Abfrage.

Spalte	Beschreibung
state	Der Status der Abfrage, etwa „executing“.
queue_time	Die Anzahl der Mikrosekunden, die die Abfrage in der Warteschlange verbracht hat.
exec_time	Die Anzahl der Mikrosekunden, seitdem die Abfrage in die Warteschl ange ausgeführt wird.

So erstellen Sie die Ansicht WLM_QUERY_STATE_VW

1. Führen Sie in RSQL die folgende Abfrage aus, um die Ansicht WLM_QUERY_STATE_VW zu erstellen.

```
create view WLM_QUERY_STATE_VW as
select query, (service_class-5) as queue, slot_count, trim(wlm_start_time) as
 start_time, trim(state) as state, trim(queue_time) as queue_time, trim(exec_time) as
 exec_time
from stv_wlm_query_state;
```

2. Führen Sie die folgende Abfrage aus, um die Informationen in der Ansicht anzuzeigen.

```
select * from wlm_query_state_vw;
```

   Nachfolgend sehen Sie ein Beispielergebnis.

query	queue	slot_count	start_time	state	queue_time	exec_time
1249	1	1	2014-09-24 22:19:16	Executing	0	516

Schritt 3: Ausführen von Testabfragen

In diesem Schritt führen Sie Abfragen von mehreren Verbindungen in RSQL aus und prüfen die Systemtabellen, um festzustellen, wie die Abfragen zur Verarbeitung weitergeleitet wurden.

Für diesen Schritt müssen zwei RSQL-Fenster geöffnet sein:

- In RSQL-Fenster 1 führen Sie Abfragen aus, die den Status der Warteschlangen und Abfragen mithilfe der in diesem Tutorial bereits erstellten Ansichten überwachen.

- In RSQL-Fenster 2 führen Sie lang andauernde Abfragen aus, um die Ergebnisse in RSQL-Fenster 1 zu ändern.

So führen Sie die Testabfragen aus

1. Öffnen Sie zwei RSQL-Fenster. Wenn Sie bereits ein Fenster geöffnet haben, müssen Sie lediglich ein zweites Fenster öffnen. Sie können für beide Verbindungen dasselbe Benutzerkonto verwenden.
2. Führen Sie in RSQL-Fenster 1 die folgende Abfrage aus.

   ```
   select * from wlm_query_state_vw;
   ```

   Nachfolgend sehen Sie ein Beispielergebnis.

query	queue	slot_count	start_time	state	queue_time	exec_time
1258	1	1	2014-09-24 22:21:03	Executing	0	549

   Diese Abfrage gibt ein selbstreferenzielles Ergebnis aus. Die derzeit ausgeführte Abfrage ist die SELECT-Anweisung aus dieser Ansicht. Eine Abfrage in dieser Ansicht gibt immer mindestens ein Ergebnis aus. Vergleichen Sie dieses Ergebnis dann mit dem Ergebnis, das nach dem Start der lang andauernden Abfrage im nächsten Schritt angezeigt wird.
3. Führen Sie in RSQL-Fenster 2 eine Abfrage aus der TICKIT-Beispieldatenbank aus. Diese Abfrage sollte etwa eine Minute lang dauern, so dass Sie Zeit haben, die Ergebnisse der vorher erstellten Ansichten WLM_QUEUE_STATE_VW und WLM_QUERY_STATE_VW zu untersuchen. In manchen Fällen stellen Sie möglicherweise fest, dass die Abfrage nicht lange genug ausgeführt wird, um beide Ansichten abzufragen. In diesen Fällen können Sie den Wert des Filters für `l.listid` erhöhen, um die Ausführungszeit zu verlängern.

   > **Note**
   >
   > Um die Ausführungszeiten von Abfragen zu reduzieren und die Systemleistung zu verbessern, stellt Amazon Redshift die Ergebnisse bestimmter Abfragen in den Speicher auf dem führenden Knoten. Wenn die Ergebniszwischenspeicherung aktiviert ist, werden nachfolgende Abfragen viel schneller ausgeführt. Um zu verhindern, dass die Abfrage zu schnell ausgeführt wird, deaktivieren Sie die Ergebniszwischenspeicherung für die aktuelle Sitzung.

Um die Ergebniszwischenspeicherung für die aktuelle Sitzung zu deaktivieren, setzen Sie den Parameter enable_result_cache_for_session wie im Folgenden dargestellt auf `off`.

```
set enable_result_cache_for_session to off;
```

Führen Sie in RSQL-Fenster 2 die folgende Abfrage aus.

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <
 100000;
```

4. Fragen Sie in RSQL-Fenster 1 WLM_QUEUE_STATE_VW und WLM_QUERY_STATE_VW ab und vergleichen Sie die Ergebnisse mit Ihren früheren Ergebnissen.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Nachfolgend sehen Sie einige Beispielergebnisse.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(querytype: any)	5	836	0	false	false	0	2	163

query	queue	slot_count	start_time	state	queue_time	exec_time
1267	1	1	2014-09-24 22:22:30	Executing	0	684
1265	1	1	2014-09-24 22:22:26	Executing	0	4080859

Beachten Sie die folgenden Unterschiede zwischen Ihren vorherigen Abfragen und den Ergebnissen in diesem Schritt:

- WLM_QUERY_STATE_VW enthält jetzt zwei Zeilen. Ein Ergebnis bezieht sich auf die selbstreferenzielle Abfrage zur Ausführung einer SELECT-Operation für diese Ansicht. Das zweite Ergebnis ist das der länger währenden Abfrage aus dem vorherigen Schritt.
- Der Wert in der Spalte „executing“ in WLM_QUEUE_STATE_VW stieg von 1 zu 2. Dieser Spalteneintrag bedeutet, dass derzeit zwei Abfragen in der Warteschlange ausgeführt werden.
- Der Wert in der Spalte „executed“ wird bei jeder Ausführung einer Abfrage in der Warteschlange erhöht.

Die Ansicht WLM_QUEUE_STATE_VW ist nützlich für einen allgemeinen Überblick über die Warteschlangen und die Zahl der in jeder dieser Warteschlangen ausgeführten Abfragen. Die Ansicht WLM_QUERY_STATE_VW ist nützlich für eine detailliertere Ansicht der einzelnen Abfragen, die derzeit ausgeführt werden.

Abschnitt 2: Modifizieren der Konfiguration der WLM-Abfragewarteschlange

Da Sie jetzt wissen, wie Abfragen standardmäßig funktionieren, können Sie mit der Konfiguration von Abfragewarteschlangen mithilfe von manuellem WLM fortfahren. In diesem Abschnitt erstellen und konfigurieren Sie eine neue Parametergruppe für Ihren Cluster. Sie erstellen zwei zusätzliche Benutzerwarteschlangen und konfigurieren sie so, dass sie Abfragen basierend auf den Bezeichnungen der Benutzergruppe oder der Anfragegruppe der Abfragen entgegennehmen. Alle Anfragen, die nicht an eine dieser beiden Warteschlangen weitergeleitet werden, werden zur Laufzeit an die Standard-Warteschlange weitergeleitet.

So erstellen Sie eine manuelle WLM-Konfiguration in einer Parametergruppe:

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Configurations (Konfigurationen) und dann Workload management (Workload-Management) aus, um die Seite Workload management (Workload-Management) anzuzeigen.
3. Wählen Sie Create (Erstellen) aus, um das Fenster Create parameter group (Parametergruppe erstellen) anzuzeigen.
4. Geben Sie **`WLMTutorial`** für Parameter group name (Parametergruppenname) und Beschreibung) ein. Wählen Sie dann Erstellen aus, um die Parametergruppe zu erstellen.

   > **Note**
   >
   > Der Parameter group name (Parametergruppenname) wird beim Erstellen in alle Kleinbuchstaben umgewandelt.

5. Wählen Sie auf der Seite Workload management (Workload-Management) die Parametergruppe **`wlmtutorial`** aus, um die Detailseite mit Registerkarten für Parameters (Parameter) und Workload management (Workload-Management) anzuzeigen.
6. Vergewissern Sie sich, dass Sie sich auf der Registerkarte Workload Management (Workload-Management) befinden. Wählen Sie dann Switch WLM mode (WLM-Modus wechseln) aus, um das Fenster Concurrency settings (Parallelitätseinstellungen) anzuzeigen.

7. Wählen Sie Manual WLM (Manuelles WLM) und dann Save (Speichern) aus, um zum manuellen WLM zu wechseln.
8. Wählen Sie Edit workload queues (Workload-Warteschlangen bearbeiten) aus.
9. Wählen Sie zweimal Add queue (Warteschlange hinzufügen) aus, um zwei Warteschlangen hinzuzufügen. Jetzt gibt es drei Warteschlangen: Queue 1 (Warteschlange 1), Queue 2 (Warteschlange 2) und Default Queue (Standardwarteschlange).
10. Geben Sie die Informationen für jede Warteschlange wie folgt ein:

    - Für Queue 1 (Warteschlange 1) geben Sie **`30`** für Memory (%) (Arbeitsspeicher (%)), **`2`** für Concurrency on main (Gleichzeitigkeit auf dem Haupt-Cluster) und **`test`** für Query groups (Abfragegruppen) ein. Übernehmen Sie für die anderen Einstellungen die eingestellten Standardwerte.
    - Für Queue 2 (Warteschlange 2) geben Sie **`40`** für Memory (%) (Arbeitsspeicher (%)), **`3`** für Concurrency on main (Gleichzeitigkeit auf dem Haupt-Cluster) und **`admin`** für User groups (Benutzergruppen) ein. Übernehmen Sie für die anderen Einstellungen die eingestellten Standardwerte.
    - Nehmen Sie unter Default queue (Standardwarteschlange) keine Änderungen vor. WLM weist der Standardwarteschlange nicht zugewiesenen Speicher zu.
11. Wählen Sie Save, um Ihre Einstellungen zu speichern.

Ordnen Sie anschließend die Parametergruppe mit der manuellen WLM-Konfiguration einem Cluster zu.

So ordnen Sie eine Parametergruppe einer manuellen WLM-Konfiguration einem Cluster zu:

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü die Option Clusters (Cluster) und dann erneut Clusters (Cluster) aus, um eine Liste Ihrer Cluster anzuzeigen.
3. Wählen Sie Ihren Cluster aus, z. B. `examplecluster`, um die Details zum Cluster anzuzeigen. Wählen Sie dann die Registerkarte Properties (Eigenschaften) aus, um die Cluster-Eigenschaften anzuzeigen.
4. Wählen Sie im Abschnitt Database configurations (Datenbankkonfigurationen) Edit (Bearbeiten), Edit parameter group (Parametergruppe bearbeiten) aus, um das Parametergruppenfenster zu öffnen.

5. Wählen Sie unter Parameter groups (Parametergruppen) die Parametergruppe **`wlmtutorial`** aus, die Sie zuvor erstellt haben.
6. Klicken Sie auf Save changes (Änderungen speichern), um die Parametergruppe zuzuordnen.

   Der Cluster wird mit der geänderten Parametergruppe modifiziert. Sie müssen jedoch den Cluster neu starten, damit die Änderungen auch auf die Datenbank angewendet werden können.
7. Wählen Sie Ihren Cluster aus und wählen Sie dann Reboot (Neu starten) für Actions (Aktionen) aus.

Nachdem der Cluster neu gestartet wurde, kehrt sein Status zu Available (Verfügbar) zurück.

Abschnitt 3: Weiterleiten von Abfragen zu Warteschlangen auf der Grundlage von Benutzergruppen und Abfragegruppen

Nun haben Sie Ihren Cluster einer neuen Parametergruppe zugeordnet und WLM konfiguriert. Führen Sie als Nächstes ein paar Abfragen aus, um zu sehen, wie Amazon Redshift Abfragen zur Bearbeitung in Warteschlangen weiterleitet.

Schritt 1: Anzeigen der Konfiguration der Abfragewarteschlange in der Datenbank

Prüfen Sie zuerst, ob die WLM-Konfiguration der Datenbank Ihren Erwartungen entspricht.

So zeigen Sie die Konfiguration der Abfragewarteschlange an

1. Öffnen Sie RSQL und führen Sie die folgende Abfrage aus. Die Abfrage verwendet die Ansicht WLM_QUEUE_STATE_VW, die Sie in erstellt haben Schritt 1: Erstellen der Ansicht WLM_QUEUE_STATE_VW. Wenn vor dem Neustart des Clusters bereits eine Sitzung mit der Datenbank verbunden war, müssen Sie die Verbindung erneut herstellen.

   ```
   select * from wlm_queue_state_vw;
   ```

   Nachfolgend sehen Sie ein Beispielergebnis.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	0	0
2	(user group: admin)	3	557	0	false	false	0	0	0
3	(querytype: any)	5	250	0	false	false	0	1	0

   Vergleichen Sie die Ergebnisse mit denen in Schritt 1: Erstellen der Ansicht WLM_QUEUE_STATE_VW. Beachten Sie, dass jetzt zwei weitere Warteschlangen vorhanden

sind. Warteschlange 1 ist jetzt die Warteschlange für die Testabfragegruppe, und Warteschlange 2 ist die Warteschlange für die Admin User-Gruppe.

Warteschlange 3 ist jetzt die Standardwarteschlange. Die letzte Warteschlange in der Liste ist immer die Standardwarteschlange. Dies ist die Warteschlange, zu der Abfragen standardmäßig weitergeleitet werden, wenn in einer Abfrage keine Benutzergruppe oder Abfragegruppe angegeben ist.

2. Führen Sie die folgende Abfrage aus, um zu bestätigen, dass Ihre Abfrage jetzt in Warteschlange 3 ausgeführt wird.

```
select * from wlm_query_state_vw;
```

Nachfolgend sehen Sie ein Beispielergebnis.

query	queue	slot_count	start_time	state	queue_time	exec_time
2144	3	1	2014-09-24 23:49:59	Executing	0	550430

Schritt 2: Ausführen einer Abfrage mit der Abfragegruppenwarteschlange

So führen Sie eine Abfrage mit der Abfragegruppenwarteschlange aus

1. Führen Sie die folgende Abfrage aus, um sie zur Abfragegruppe `test` weiterzuleiten.

```
set query_group to test;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Führen Sie vom anderen RSQL-Fenster aus die folgende Abfrage aus.

```
select * from wlm_query_state_vw;
```

Nachfolgend sehen Sie ein Beispielergebnis.

query	queue	slot_count	start_time	state	queue_time	exec_time
2168	1	1	2014-09-24 23:54:18	Executing	0	6343309
2170	3	1	2014-09-24 23:54:24	Executing	0	847

Die Abfrage wurde zur Testabfragegruppe geleitet; dies ist jetzt Warteschlange 1.

3. Wählen Sie in der Ansicht des Warteschlangenzustands alle aus.

```
select * from wlm_queue_state_vw;
```

Das Ergebnis sieht in etwa wie folgt aus.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	1	0
2	(user group: admin)	3	557	0	false	false	0	0	0
3	(querytype: any)	5	250	0	false	false	0	1	3

4. Setzen Sie jetzt die Abfragegruppe zurück, und führen Sie die lange Abfrage erneut aus:

```
reset query_group;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

5. Führen Sie die Abfragen gegen die Ansichten aus, um die Ergebnisse zu sehen.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Nachfolgend sehen Sie einige Beispielergebnisse.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	0	1
2	(user group: admin)	3	557	0	false	false	0	0	0
3	(querytype: any)	5	250	0	false	false	0	2	5

query	queue	slot_count	start_time	state	queue_time	exec_time
2186	3	1	2014-09-24 23:57:52	Executing	0	649
2184	3	1	2014-09-24 23:57:48	Executing	0	4137349

Das Ergebnis sollte sein, dass die Abfrage jetzt wieder in Warteschlange 3 ausgeführt wird.

Schritt 3: Erstellen eines Datenbankbenutzers und einer Benutzergruppe

Bevor Sie in dieser Warteschlange Abfragen ausführen können, müssen Sie die Benutzergruppe in der Datenbank erstellen und ihr einen Benutzer hinzufügen. Melden Sie sich dann bei RSQL mit den Anmeldeinformationen des neuen Benutzers an und führen Sie Abfragen aus. Um Datenbankbenutzer zu erstellen, müssen Sie Abfragen als Superuser (etwa als „admin user“) ausführen.

So erstellen Sie einen neuen Datenbankbenutzer und eine Benutzergruppe

1. Erstellen Sie in der Datenbank einen neuen Datenbankbenutzer mit der Bezeichnung `adminwlm`, indem Sie in einem RSQL-Fenster den folgenden Befehl ausführen.

```
create user adminwlm createuser password '123Admin';
```

2. Führen Sie dann die folgenden Befehl aus, um die neue Benutzergruppe zu erstellen und den neuen Benutzer `adminwlm` hinzuzufügen.

```
create group admin;
alter group admin add user adminwlm;
```

Schritt 4: Ausführen einer Abfrage mit der Benutzergruppenwarteschlange

Führen Sie als Nächstes eine Abfrage aus und leiten Sie sie zur Benutzergruppenwarteschlange weiter. Sie tun dies, wenn Sie Ihre Abfrage zu einer Warteschlange weiterleiten möchten, die für die Bearbeitung der Art der auszuführenden Abfrage konfiguriert ist.

So führen Sie eine Abfrage mit der Benutzergruppenwarteschlange aus

1. Führen Sie in RSQL-Fenster 2 die folgenden Abfragen aus, um zum Konto `adminwlm` zu wechseln und eine Abfrage als dieser Benutzer auszuführen.

```
set session authorization 'adminwlm';
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Führen Sie in RSQL-Fenster 1 die folgende Abfrage aus, um die Abfragewarteschlange zu sehen, zu der die Abfragen geleitet werden.

```
select * from wlm_query_state_vw;
select * from wlm_queue_state_vw;
```

Nachfolgend sehen Sie einige Beispielergebnisse.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	0	1
2	(user group: admin)	3	557	0	false	false	0	1	0
3	(querytype: any)	5	250	0	false	false	0	1	8

query	queue	slot_count	start_time	state	queue_time	exec_time
2202	2	1	2014-09-25 00:01:38	Executing	0	4885796
2204	3	1	2014-09-25 00:01:43	Executing	0	650

Die Warteschlange, in der diese Abfrage ausgeführt wurde, ist Warteschlange 2, die Warteschlange für `admin`-Benutzer. Immer wenn Sie als dieser Benutzer Abfragen ausführen, werden diese in Warteschlange 2 ausgeführt, sofern Sie nicht eine andere Abfragegruppe zur Verwendung angeben. Die ausgewählte Warteschlange ist abhängig von den Zuweisungsregeln für Warteschlangen. Weitere Informationen finden Sie unter WLM-Warteschlangenzuweisungsregeln.

3. Führen Sie jetzt von RSQL-Fenster 2 aus die folgende Abfrage aus.

```
set query_group to test;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

4. Führen Sie in RSQL-Fenster 1 die folgende Abfrage aus, um die Abfragewarteschlange zu sehen, zu der die Abfragen geleitet werden.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Nachfolgend sehen Sie einige Beispielergebnisse.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	1	1
2	(user group: admin)	3	557	0	false	false	0	0	1
3	(querytype: any)	5	250	0	false	false	0	1	10

query	queue	slot_count	start_time	state	queue_time	exec_time
2218	1	1	2014-09-25 00:04:30	Executing	0	4819666
2220	3	1	2014-09-25 00:04:35	Executing	0	685

5. Setzen Sie anschließend die Abfragegruppe zurück.

```
reset query_group;
```

Abschnitt 4: Verwenden von wlm_query_slot_count zur temporären Übergehung der Gleichzeitigkeitsstufe in einer Warteschlange

Gelegentlich benötigen Benutzer mehr Ressourcen für eine bestimmte Abfrage. Wenn dies der Fall ist, können Sie die Konfigurationseinstellung wlm_query_slot_count verwenden, um die Zuweisungsweise für Slots in einer Abfragewarteschlange vorübergehend zu übergehen. Slots sind Arbeitsspeicher- und CPU-Einheiten, die zur Verarbeitung von Abfragen verwendet werden. Sie können die Slotzahl übergehen, wenn Sie gelegentlich Abfragen ausführen müssen, die viele Ressourcen in dem Cluster beanspruchen, etwa bei einer VACUUM-Operation in einer Datenbank.

Sie werden möglicherweise feststellen, dass Benutzer oft für bestimmte Typen von Abfragen wlm_query_slot_count festlegen müssen. Ist dies der Fall, empfehlen wir, die WLM-Konfiguration anzupassen und Benutzern eine Warteschlange bereitzustellen, die die Anforderungen ihrer Abfragen besser erfüllt. Für weitere Informationen zum temporären Übergehen der Gleichzeitigkeitsstufe auf der Grundlage der Slotzahl vgl. wlm_query_slot_count.

Schritt 1: Übergehen der Gleichzeitigkeitsstufe mit wlm_query_slot_count

Für die Zwecke dieses Tutorials führen wir die gleiche lang andauernde SELECT-Abfrage aus. Wir führen diese als Benutzer `adminwlm` aus und verwenden wlm_query_slot_count zur Erhöhung der Zahl der für die Abfrage verfügbaren Slots.

So übergehen Sie die Gleichzeitigkeitsstufe mit wlm_query_slot_count

1. Erhöhen Sie den Grenzwert auf der Abfrage, um sicherzustellen, dass Sie ausreichend Zeit für die Abfrage der Ansicht WLM_QUERY_STATE_VW und die Anzeige eines Ergebnisses haben.

   ```
   set wlm_query_slot_count to 3;
   select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
   ```

2. Fragen Sie jetzt WLM_QUERY_STATE_VW als Administrator ab, um zu sehen, wie die Abfrage ausgeführt wird.

   ```
   select * from wlm_query_state_vw;
   ```

   Nachfolgend sehen Sie ein Beispielergebnis.

query	queue	slot_count	start_time	state	queue_time	exec_time
2240	2	3	2014-09-25 00:08:45	Executing	0	3731414
2242	3	1	2014-09-25 00:08:49	Executing	0	596

Beachten Sie, dass die Slotzahl für die Abfrage 3 ist. Diese Zahl bedeutet, dass die Abfrage alle drei Slots für die Verarbeitung verwendet, wobei alle Ressourcen in der Warteschlange dieser Abfrage zugewiesen werden.

3. Führen Sie jetzt die folgende Abfrage aus.

```
select * from WLM_QUEUE_STATE_VW;
```

Nachfolgend sehen Sie ein Beispielergebnis.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	0	4
2	(user group: admin)	3	557	0	false	false	0	1	3
3	(querytype: any)	5	250	0	false	false	0	1	25

Die Konfigurationseinstellung wlm_query_slot_count gilt nur für die aktuelle Sitzung. Wenn diese Sitzung abgelaufen ist, oder wenn ein anderer Benutzer eine Abfrage ausführt, wird die WLM-Konfiguration verwendet.

4. Setzen Sie die Slotzahl zurück, und führen Sie den Test erneut aus.

```
reset wlm_query_slot_count;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

Nachfolgend sehen Sie einige Beispielergebnisse.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	0	0	2
2	(user group: admin)	3	557	0	false	false	0	1	2
3	(querytype: any)	5	250	0	false	false	0	1	14

query	queue	slot_count	start_time	state	queue_time	exec_time
2260	2	1	2014-09-25 00:12:11	Executing	0	4042618
2262	3	1	2014-09-25 00:12:15	Executing	0	680

Schritt 2: Ausführen von Abfragen aus verschiedenen Sitzungen

Führen Sie dann Abfragen aus verschiedenen Sitzungen aus.

So führen Sie Abfragen aus verschiedenen Sitzungen aus

1. Führen Sie in den RSQL-Fenstern 1 und 2 die folgende Abfrage aus, um die Testabfragegruppe zu verwenden.

```
set query_group to test;
```

2. Führen Sie in RSQL-Fenster 1 die folgende lang andauernde Abfrage aus.

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

3. Da die lang andauernde Abfrage weiterhin in RSQL-Fenster 1 läuft, führen Sie die folgende Abfrage aus. Durch diese Befehle wird die Slot-Zahl erhöht, so dass alle Slots für die Warteschlange verwendet werden. Anschließend wird die länger andauernde Abfrage gestartet.

```
set wlm_query_slot_count to 2;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

4. Öffnen Sie ein drittes RSQL-Fenster und fragen Sie die Ansichten ab, um die Ergebnisse zu sehen.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Nachfolgend sehen Sie einige Beispielergebnisse.

queue	description	slots	mem	max_time	user_*	query_*	queued	executing	executed
0	(super user) and (query group: superuser)	1	357	0	false	false	0	0	0
1	(query group: test)	2	627	0	false	false	1	1	2
2	(user group: admin)	3	557	0	false	false	0	0	3
3	(querytype: any)	5	250	0	false	false	0	1	18

query	queue	slot_count	start_time	state	queue_time	exec_time
2286	1	2	2014-09-25 00:16:48	QueuedWaiting	3758950	0
2282	1	1	2014-09-25 00:16:33	Executing	0	19335850
2288	3	1	2014-09-25 00:16:52	Executing	0	666

Beachten Sie, dass die erste Abfrage einen der Slots verwendet, die der Warteschlange 1 für das Ausführen der Abfrage zugeordnet sind. Beachten Sie außerdem, dass es eine Abfrage gibt, die in der Warteschlange wartet (`queued` ist hierbei 1 und `state` ist `QueuedWaiting`). Nachdem die erste Abfrage abgeschlossen ist, beginnt die Ausführung der zweiten. Dies geschieht, da beide

Abfragen zur Abfragegruppe `test` geleitet werden und die zweite Abfrage auf eine ausreichende Zahl von Slots warten muss, bevor sie ausgeführt werden kann.

Abschnitt 5: Bereinigen Ihrer Ressourcen

Solange dieser ausgeführt wird, fallen Gebühren für Ihren Cluster an. Setzen Sie, wenn Sie dieses Tutorial abgeschlossen haben, Ihre Umgebung wieder auf den ursprünglichen Zustand zurück, indem Sie die folgenden Schritte in Suchen von zusätzlichen Ressourcen und Zurücksetzen der Umgebung im Handbuch Erste Schritte mit Amazon Redshift durchführen.

Weitere Informationen zu WLM finden Sie unter Implementierung von Workload Management.

Arbeiten mit Nebenläufigkeitsskalierung

Mittels der Funktion für Nebenläufigkeitsskalierung können Sie Tausende gleichzeitige Benutzer und Abfragen bei einer konsistent schnellen Abfrageleistung unterstützen. Wenn Sie Nebenläufigkeitsskalierung aktivieren, fügt Amazon Redshift automatisch zusätzliche Cluster-Kapazität hinzu, um einen Anstieg von Lese- und Schreibabfragen zu verarbeiten. Benutzern werden stets die jeweils aktuellen Daten angezeigt, unabhängig davon, ob die Abfragen auf dem Haupt- oder einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden.

Sie können verwalten, welche Abfragen an das Nebenläufigkeitsskalierungs-Cluster gesendet werden, indem Sie WLM-Warteschlangen konfigurieren. Wenn die Nebenläufigkeitsskalierung aktiviert ist, werden entsprechend qualifizierte Abfragen an das Nebenläufigkeitsskalierungs-Cluster übermittelt, anstatt in einer Warteschlange zu warten.

Die Nebenläufigkeitsskalierungs-Cluster werden Ihnen nur für die Zeit, in der sie aktiv Abfragen ausführen, in Rechnung gestellt. Weitere Informationen zu den Preisen, einschließlich anfallender Gebühren und Mindestgebühren, finden Sie unter Preise für die Skalierung von Parallelität.

Nebenläufigkeitsskalierungsfunktionen

Wenn Sie die Nebenläufigkeitsskalierung für eine WLM-Warteschlange aktivieren, kann sie für Lesevorgänge, z. B. Dashboard-Abfragen, verwendet werden. Sie kann auch für häufig verwendete Schreibvorgänge eingesetzt werden, wie zum Beispiel Anweisungen für die Datenaufnahme und -verarbeitung.

Nebenläufigkeitsskalierungsfunktionen für Schreibvorgänge

Die Nebenläufigkeitsskalierung unterstützt häufig verwendete Schreibvorgänge, wie ETL-Anweisungen (Extract, Transform, Load). Die Nebenläufigkeitsskalierung für Schreibvorgänge ist besonders nützlich, wenn Sie konsistente Reaktionszeiten erreichen möchten, wenn Ihr Cluster eine große Anzahl von Anforderungen empfängt. Sie verbessert den Durchsatz für Schreibvorgänge, die dieselben Ressourcen im Haupt-Cluster benötigen.

Die Parallelitätsskalierung unterstützt die Anweisungen COPY, INSERT, DELETE, UPDATE und CREATE TABLE AS (CTAS). Darüber hinaus unterstützt die Parallelitätsskalierung die Aktualisierung von materialisierten Ansichten für MVs, die keine Aggregationen verwenden. Andere Data Manipulation Language (DML)- und Data Definition Language (DDL)-Anweisungen werden nicht unterstützt. Wenn nicht unterstützte Schreibanweisungen, wie CREATE ohne TABLE AS, in einer expliziten Transaktion vor den unterstützten Schreibanweisungen enthalten sind, wird keine der Schreibanweisungen auf Clustern mit Parallelitätsskalierung ausgeführt.

Wenn Sie Guthaben für die Nebenläufigkeitsskalierung angesammelt haben, gilt das Guthaben sowohl für Schreib- als auch Lesevorgänge.

Einschränkungen bei der Nebenläufigkeitsskalierung

Bei der Verwendung der Nebenläufigkeitsskalierung in Amazon Redshift gelten folgende Einschränkungen:

- Sie unterstützt keine Abfragen für Tabellen, die überlappende Sortierschlüssel verwenden.
- Sie unterstützt keine Abfragen für temporäre Tabellen.
- Sie unterstützt keine Abfragen, die auf externe Ressourcen zugreifen, die durch restriktive Netzwerk- oder VPC-Konfigurationen (Virtual Private Cloud) geschützt sind.
- Sie unterstützt keine Abfragen, die benutzerdefinierte Python-Funktionen (UDFs) und Lambda UDFs enthalten.
- Sie unterstützt keine Abfragen, die auf Systemtabellen, PostgreSQL-Katalogtabellen oder Tabellen ohne Sicherungskopie zugreifen.
- Sie unterstützt keine COPY- oder UNLOAD-Abfragen, die auf eine externe Ressource zugreifen, wenn restriktive IAM-Richtlinienberechtigungen vorhanden sind. Dazu gehören Berechtigungen, die entweder auf die Ressource, wie einen Amazon S3 S3-Bucket oder eine DynamoDB-Tabelle, oder auf die Quelle angewendet werden. IAM-Quellen können Folgendes beinhalten:
  - `aws:sourceVpc`— Eine Quell-VPC.

- `aws:sourceVpce`— Ein Quell-VPC-Endpunkt.
- `aws:sourceIp`— Eine Quell-IP-Adresse.

In einigen Fällen müssen Sie möglicherweise Berechtigungen entfernen, die entweder die Ressource oder die Quelle einschränken, sodass COPY- und UNLOAD-Abfragen, die auf die Ressource zugreifen, an den Concurrency-Scaling-Cluster gesendet werden.

Weitere Informationen zu Ressourcenrichtlinien finden Sie unter Richtlinientypen im AWS Identity and Access Management Benutzerhandbuch und Steuern des Zugriffs von VPC-Endpunkten mit Bucket-Richtlinien.

- Die Amazon-Redshift-Nebenläufigkeitsskalierung für Schreibvorgänge wird für DDL-Vorgänge wie CREATE TABLE oder ALTER TABLE nicht unterstützt.
- Sie unterstützt nicht ANALYZE für den COPY-Befehl.
- Sie unterstützt keine Schreibvorgänge für eine Zieltabelle, in der DISTSTYLE auf ALL festgelegt ist.
- COPY wird aus den folgenden Dateiformaten nicht unterstützt:
  - Parquet
  - ORC
- Sie unterstützt keine Schreibvorgänge für Tabellen mit Identitätsspalten.
- Amazon Redshift unterstützt die Nebenläufigkeitsskalierung für Schreibvorgänge nur auf Amazon-Redshift-RA3-Knoten, insbesondere ra3.16xlarge, ra3.4xlarge und ra3.xlplus. Die Nebenläufigkeitsskalierung für Schreibvorgänge wird auf anderen Knotentypen nicht unterstützt.

AWS-Regionen für Parallelitätsskalierung

Die Parallelitätsskalierung ist in den folgenden AWS Regionen verfügbar:

- Region USA Ost (Nord-Virginia) (us-east-1)
- Region USA Ost (Ohio) (us-east-2)
- AWS GovCloud (US-Ost)
- Region USA West (Nordkalifornien) (us-west-1)
- Region USA West (Oregon) (us-west-2)
- Region Asien-Pazifik (Mumbai) (ap-south-1)
- Region Asien-Pazifik (Seoul) (ap-northeast-2)

- Region Asien-Pazifik (Singapur) (ap-southeast-1)
- Region Asien-Pazifik (Sydney) (ap-southeast-2)
- Region Asien-Pazifik (Tokio) (ap-northeast-1)
- Region Kanada (Zentral) (ca-central-1)
- Region Europa (Frankfurt) (eu-central-1)
- Region Europa (Irland) (eu-west-1)
- Region Europa (London) (eu-west-2)
- Region Europa (Paris) (eu-west-3)
- Region Europa (Stockholm) (eu-north-1)
- Region Südamerika (São Paulo) (sa-east-1)

Kandidaten für die Nebenläufigkeitsskalierung

Abfragen werden nur an den Nebenläufigkeitsskalierungs-Cluster weitergeleitet, wenn der Haupt-Cluster die folgenden Voraussetzungen erfüllt:

- EC2-VPC-Plattform
- Der Knotentyp muss dc2.8xlarge, dc2.large, ra3.xlplus, ra3.4xlarge oder ra3.16xlarge sein. Die Nebenläufigkeitsskalierung für Schreibvorgänge wird nur auf den folgenden Amazon-Redshift-RA3-Knoten unterstützt: ra3.16xlarge, ra3.4xlarge und ra3.xlplus.
- Maximal 32 Rechenknoten für Cluster mit den Knotentypen ra3.xlplus, ra3.4xlarge oder ra3.16xlarge. Darüber hinaus kann die Anzahl der Knoten des Haupt-Clusters nicht größer als 32 Knoten sein, als der Cluster ursprünglich erstellt wurde. Selbst wenn ein Cluster derzeit über 20 Knoten verfügt, aber ursprünglich mit 40 erstellt wurde, erfüllt er nicht die Anforderungen für die Parallelitätsskalierung. Umgekehrt erfüllt ein DC2-Cluster, wenn er derzeit 40 Knoten hat, ursprünglich aber mit 20 erstellt wurde, die Anforderungen für die Parallelitätsskalierung.
- Kein Einzelknoten-Cluster.

Konfigurieren von Nebenläufigkeitsskalierungswarteschlangen

Sie leiten Abfragen an Nebenläufigkeitsskalierungs-Cluster weiter, indem Sie eine Workload Manager(WLM)-Warteschlange als Nebenläufigkeitsskalierungswarteschlange einrichten. Zur Aktivierung der Nebenläufigkeitsskalierung für eine Warteschlange legen Sie für den Wert für Concurrency Scaling mode (Nebenläufigkeitsskalierungsmodus) auf auto (automatisch) fest.

Wenn die Anzahl der Abfragen, die an eine Nebenläufigkeitsskalierungswarteschlange weitergeleitet werden, die konfigurierte Nebenläufigkeit der Warteschlange überschreitet, werden berechtigte Abfragen an den Nebenläufigkeitsskalierungs-Cluster gesendet. Wenn Slots verfügbar sind, werden Abfragen auf dem Haupt-Cluster ausgeführt. Die Anzahl von Warteschlangen wird nur durch die Anzahl der pro Cluster zulässigen Abfragen begrenzt. Wie bei allen WLM-Warteschlangen leiten Sie Abfragen an eine Nebenläufigkeitsskalierungswarteschlange anhand von Benutzergruppen oder durch Beschriften von Abfragen mit Abfragegruppenbezeichnungen weiter. Sie können Abfragen auch weiterleiten, indem Sie definieren WLM-Abfrageüberwachungsregeln. Sie können zum Beispiel alle Abfragen, die länger als 5 Sekunden dauern, an eine Nebenläufigkeitsskalierungswarteschlange weiterleiten.

Die Standardanzahl von Clustern für Parallelitätsskalierung ist eins. Die Anzahl der Nebenläufigkeitsskalierungscluster, die verwendet werden können, wird von gesteuert max_concurrency_scaling_clusters.

Überwachen der Nebenläufigkeitsskalierung

Sie können feststellen, ob eine Abfrage auf dem Haupt-Cluster oder einem Nebenläufigkeitsskalierungs-Cluster ausgeführt wird. Navigieren Sie dazu in der Amazon-Redshift-Konsole zu Cluster und wählen Sie einen Cluster aus. Wählen Sie dann die Registerkarte Abfrageüberwachung und Gleichzeitigkeit von Workloads aus, um Informationen zu laufenden Abfragen und Abfragen in der Warteschlange anzuzeigen.

Wenn Sie Ausführungszeiten ermitteln möchten, fragen Sie die Tabelle STL_QUERY ab und filtern die Spalte `concurrency_scaling_status`. Die folgende Abfrage vergleicht die Warteschlangen- und die Ausführungszeit von Abfragen, die auf dem Nebenläufigkeitsskalierungs-Cluster und auf dem Haupt-Cluster ausgeführt wurden.

```
SELECT w.service_class AS queue
, CASE WHEN q.concurrency_scaling_status = 1 THEN 'concurrency scaling cluster' ELSE
 'main cluster' END as concurrency_scaling_status
, COUNT( * ) AS queries
, SUM( q.aborted ) AS aborted
, SUM( ROUND( total_queue_time::NUMERIC / 1000000,2) ) AS queue_secs
, SUM( ROUND( total_exec_time::NUMERIC / 1000000,2) ) AS exec_secs
FROM stl_query q
JOIN stl_wlm_query w
USING (userid,query)
WHERE q.userid > 1
AND q.starttime > '2019-01-04 16:38:00'
```

```
AND q.endtime < '2019-01-04 17:40:00'
GROUP BY 1,2
ORDER BY 1,2;
```

Passen Sie die Werte `starttime` und `endtime` an Ihre Anforderungen an.

Systemansichten der Nebenläufigkeitsskalierung

Es gibt eine Reihe von Systemansichten mit dem Präfix SVCS, die Details aus den Systemprotokolltabellen zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern enthalten.

Die folgenden Ansichten umfassen die gleichen Informationen wie die entsprechenden STL-oder SVL-Ansichten:

- SVCS_ALERT_EVENT_LOG
- SVCS_COMPILE
- SVCS_EXPLAIN
- SVCS_PLAN_INFO
- SVCS_QUERY_SUMMARY
- SVCS_STREAM_SEGS

Die folgenden Ansichten sind spezifische Ansichten für die Nebenläufigkeitsskalierung.

- SVCS_CONCURRENCY_SCALING_USAGE

Weitere Informationen zur Nebenläufigkeitsskalierung finden Sie in den folgenden Themen im Amazon-Redshift-Verwaltungshandbuch.

- Anzeigen von Nebenläufigkeitsskalierungsdaten
- Anzeige der Clusterleistung während der Abfrageausführung
- Anzeige von Abfragedetails

Arbeiten mit Short Query Acceleration

Wenn Sie Short Query Acceleration (SQA) verwenden, werden ausgewählte, kurze Abfragen gegenüber Abfragen mit einer höheren Dauer priorisiert. SQA führt kurze Abfragen an einer

dedizierten Stelle aus, sodass SQA-Abfragen nicht hinter längeren Abfragen in Warteschlangen eingereiht werden. SQA priorisiert nur Warteschlangen, die über eine kurze Zeit ausgeführt werden und sich in einer benutzerdefinierten Warteschlange befinden. Mit SQA werden kürzere Abfragen schneller ausgeführt und der Benutzer sieht schneller Ergebnisse.

Wenn Sie SQA aktivieren, können Sie dedizierte Workload Management (WLM)-Warteschlangen für kurze Abfragen reduzieren. Außerdem konkurrieren lange Abfragen nicht mit kurzen Abfragen um Slots in einer Warteschlange, daher können Sie Ihre WLM-Warteschlangen mit weniger Abfrage-Slots konfigurieren. Wenn Sie einen niedrigeren Nebenläufigkeitswert verwenden, wird bei den meisten Arbeitslasten der Abfragedurchsatz erhöht und die gesamte Systemleistung verbessert.

CREATE TABLE AS (CTAS)-Anweisungen und schreibgeschützte Abfragen wie SELECT-Anweisungen sind für SQA qualifiziert.

Amazon Redshift verwendet einen Machine-Learning-Algorithmus, um berechtigte Abfragen zu analysieren und die Ausführungsdauer der Abfrage abzuschätzen. Standardmäßig weist WLM dynamisch einen Wert für die maximale SQA-Laufzeit zu, basierend auf der Analyse des Cluster-Workloads. Alternativ können Sie einen festen Wert von 1–20 Sekunden angeben. Wenn die prognostizierte Laufzeit der Abfrage geringer ist als die definierte oder dynamisch zugewiesene maximale SQA-Laufzeit und die Abfrage in einer WLM-Warteschlange wartet, separiert SQA die Abfrage von den WLM-Warteschlangen und plant sie für die vorrangige Ausführung. Wenn eine Abfrage länger als die maximale SQA-Laufzeit ausgeführt wird, verschiebt WLM die Abfrage entsprechend den WLM-Warteschlangenzuweisungsregeln in die erste übereinstimmende WLM-Warteschlange. Mit der Zeit werden die Voraussagen besser, da SQA aus den Abfragemustern lernt.

SQA ist in der Standardparametergruppe und für alle neuen Parametergruppen standardmäßig aktiviert. Um SQA in der Amazon-Redshift-Konsole zu deaktivieren, bearbeiten Sie die WLM-Konfiguration für eine Parametergruppe und deaktivieren Enable short query acceleration (Short Query Acceleration aktivieren). Wir empfehlen, üblicherweise höchstens 15 WLM-Abfrage-Slots zu verwenden, um die Systemleistung zu optimieren. Weitere Informationen zum Ändern von WLM-Konfigurationen finden Sie unter Konfigurieren von Workload Management im Amazon-Redshift-Verwaltungshandbuch.

Maximale Ausführungszeit für kurze Abfragen

Wenn Sie SQA aktivieren, legt WLM die maximale Ausführungszeit für kurze Abfragen standardmäßig als „dynamisch“ fest. Sie sollten die dynamische Einstellung für die maximale SQA-Laufzeit beibehalten. Sie können die Standardeinstellung überschreiben, indem Sie einen festen Wert von 1–20 Sekunden angeben.

In einigen Fällen sollten Sie möglicherweise die Verwendung anderer Werte für die maximale SQA-Laufzeit in Betracht ziehen, um die Leistung des Systems zu verbessern. Analysieren Sie in solchen Fällen den Workload, um die maximale Ausführungsdauer der meisten kürzeren Abfragen zu bestimmen. Die folgende Abfrage gibt die maximale Laufzeit für Abfragen mit ungefähr der 70. Perzentile zurück.

```
select least(greatest(percentile_cont(0.7)
within group (order by total_exec_time / 1000000) + 2, 2), 20)
from stl_wlm_query
where userid >= 100
and final_state = 'Completed';
```

Nachdem Sie für Ihren Workload eine geeignete maximale Laufzeit festgelegt haben, müssen Sie diese nur ändern, wenn der Workload erheblich geändert wird.

SQA-Überwachung

Führen Sie die folgende Abfrage aus, um zu überprüfen, ob SQA aktiviert ist. Wenn für die Abfrage eine Zeile zurückgegeben wird, ist SQA aktiviert.

```
select * from stv_wlm_service_class_config
where service_class = 14;
```

Die folgende Abfrage zeigt die Anzahl der Abfragen, die die einzelnen Abfragewarteschlangen durchlaufen haben (Service-Klasse). Sie zeigt zudem die durchschnittliche Ausführungsdauer, die Anzahl der Abfragen mit einer Wartezeit ab der 90. Perzentile und die durchschnittliche Wartezeit. SQA-Abfragen in Serviceklasse 14.

```
select final_state, service_class, count(*), avg(total_exec_time),
percentile_cont(0.9) within group (order by total_queue_time), avg(total_queue_time)
from stl_wlm_query where userid >= 100 group by 1,2 order by 2,1;
```

Um herauszufinden, welche Abfragen von SQA abgeholt und erfolgreich abgeschlossen wurden, führen Sie die folgende Abfrage aus.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class = 14 and a.final_state = 'Completed'
```

```
order by b.query desc limit 5;
```

Um herauszufinden, welche Abfragen von SQA abgeholt wurden, aber einen Timeout hatten, führen Sie die folgende Abfrage aus.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class = 14 and a.final_state = 'Evicted'
order by b.query desc limit 5;
```

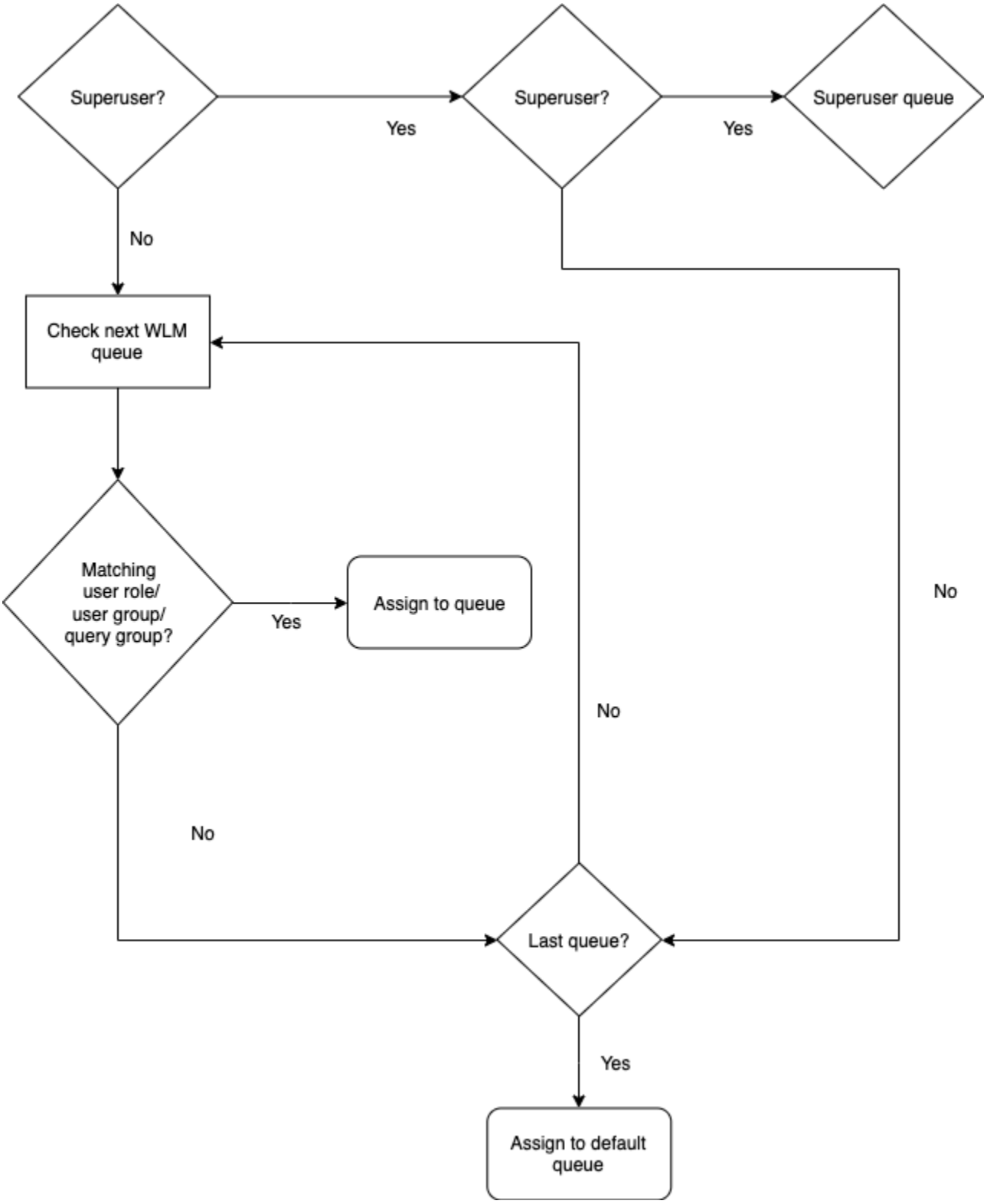
Weitere Informationen zu bereinigten Abfragen und allgemein zu regelbasierten Aktionen, die auf Abfragen angewendet werden können, finden Sie unter WLM-Abfrageüberwachungsregeln.

WLM-Warteschlangenzuweisungsregeln

Wenn ein Benutzer eine Abfrage ausführt, weist WLM die Abfrage auf der Grundlage der WLM-Warteschlangenzuweisungsregeln der ersten übereinstimmenden Warteschlange zu.

1. Wenn ein Benutzer als Superuser angemeldet ist und eine Abfrage in der mit „superuser“ bezeichneten Abfragegruppe ausführt, wird die Abfrage der Superuser-Warteschlange zugewiesen.
2. Wenn ein Benutzer Teil einer Rolle ist, zu einer aufgeführten Benutzergruppe gehört oder eine Abfrage in einer aufgeführten Abfragegruppe ausführt, wird die Abfrage der ersten passenden Warteschlange zugewiesen.
3. Wenn eine Abfrage keinem der Kriterien entspricht, wird sie der Standard-Warteschlange zugewiesen, die die letzte der in der WLM-Konfiguration definierten Warteschlangen ist.

Das folgende Diagramm zeigt, wie diese Regeln funktionieren.

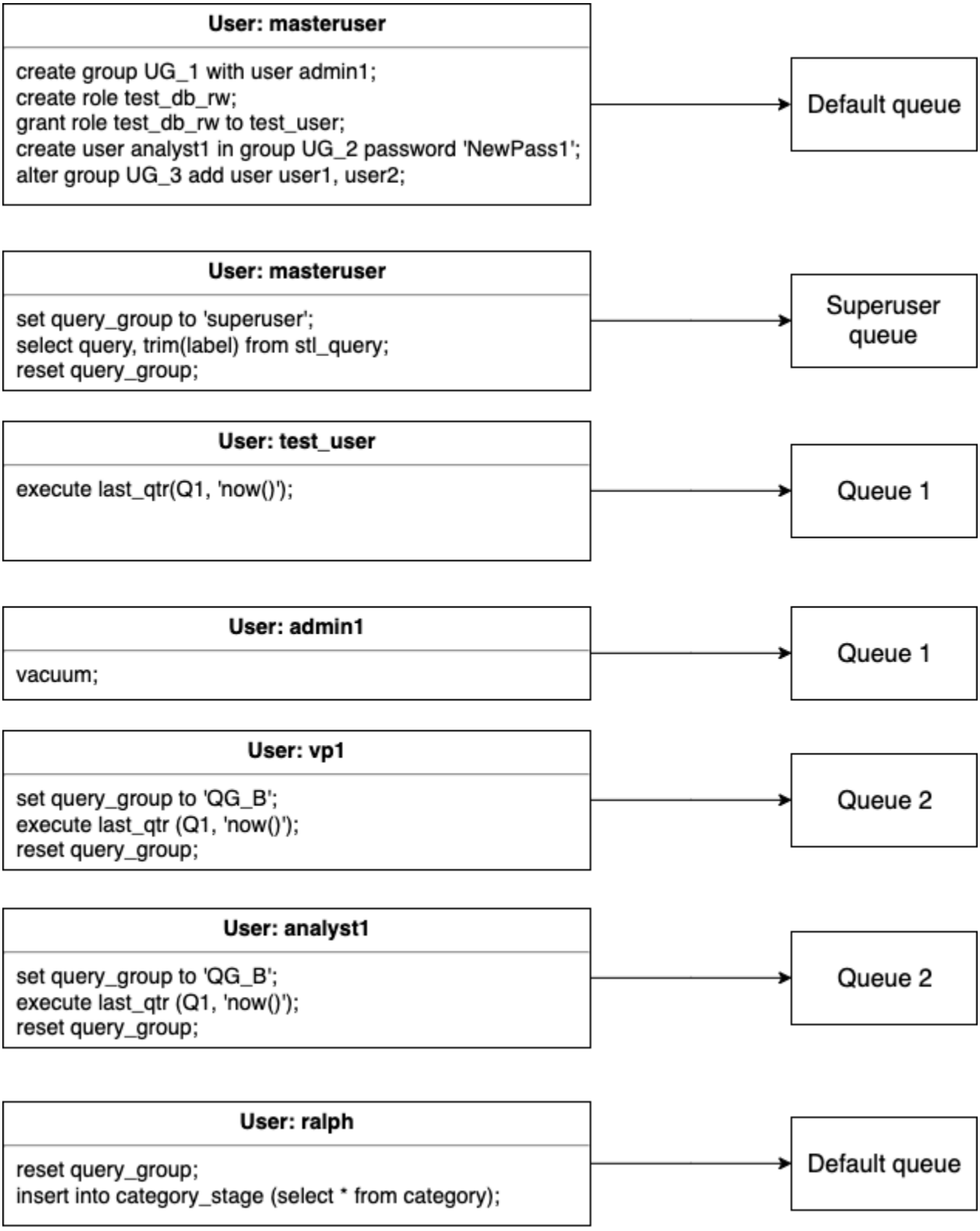

Superuser?
Yes
Superuser?
Yes
Superuser queue
No
Check next WLM queue
Matching user role/ user group/ query group?
Yes
Assign to queue
No
No
No
Last queue?
Yes
Assign to default queue

Beispiel für Warteschlangenzuweisungen

Die folgende Tabelle zeigt eine WLM-Konfiguration mit der Superuser-Warteschlange und vier benutzerdefinierten Warteschlangen.

Warteschlange	Gleichzeitigkeit	Benutzerrollen	User Groups (Benutzer gruppen)	Query Groups (Abfrageg ruppen)
Superuser	1			superuser
1	5	test_db_rw	UG_1	
2	5			QG_B
3	5		UG_2	QG_C
Standard	5			

Das folgende Beispiel zeigt, wie Abfragen basierend auf Benutzer- und Abfragegruppen den in der vorherigen Tabelle aufgelisteten Warteschlangen zugewiesen werden. Informationen zum Zuweisen von Abfragen zu Benutzer- und Abfragegruppen zur Laufzeit finden Sie unter Zuweisen von Abfragen zu Warteschlangen an späterer Stelle in diesem Abschnitt.



In diesem Beispiel nimmt WLM die folgenden Zuweisungen vor:

1. Der erste Satz von Anweisungen zeigt drei Möglichkeiten für die Zuweisung von Benutzern zu Benutzergruppen. Die Anweisungen werden von dem Benutzer `adminuser` ausgeführt, der kein Mitglied einer in einer WLM-Warteschlange aufgeführten Benutzergruppe ist. Es ist keine Abfragegruppe eingerichtet, die Anweisungen werden daher zur Standard-Warteschlange geleitet.
2. Der Benutzer `adminuser` ist ein Superuser, und die Abfragegruppe ist auf `'superuser'` gesetzt, die Abfrage wird daher der Superuser-Warteschlange zugewiesen.
3. Dem Benutzer `test_user` wurde die Rolle `test_db_rw` zugewiesen, die in Warteschlange 1 aufgeführt ist. Daher wird die Abfrage Warteschlange 1 zugewiesen.
4. Der Benutzer `admin1` ist ein Mitglied der in Warteschlange 1 aufgeführten Benutzergruppe, die Abfrage wird daher Warteschlange 1 zugewiesen.
5. Der Benutzer `vp1` ist kein Mitglied einer aufgeführten Benutzergruppe. Die Abfragegruppe ist auf `'QG_B'` gesetzt, die Abfrage wird daher Warteschlange 2 zugewiesen.
6. Der Benutzer `analyst1` ist ein Mitglied der in Warteschlange 3 aufgeführten Benutzergruppe, `'QG_B'` entspricht jedoch Warteschlange 2, die Abfrage wird daher Warteschlange 2 zugewiesen.
7. Der Benutzer `ralph` ist kein Mitglied einer aufgeführten Benutzergruppe, und die Abfragegruppe wurde zurückgesetzt, es gibt daher keine passende Warteschlange. Die Abfrage wird der Standard-Warteschlange zugewiesen.

Zuweisen von Abfragen zu Warteschlangen

Die folgenden Beispiele weisen Abfragen zu Warteschlangen auf der Grundlage von Benutzerrollen, Benutzergruppen und Abfragegruppen zu.

Zuweisen von Abfragen zu Warteschlangen auf der Grundlage von Benutzerrollen

Wenn ein Benutzer einer Rolle zugewiesen ist und diese Rolle an eine Warteschlange angehängt ist, werden von diesem Benutzer ausgeführte Abfragen dieser Warteschlange zugewiesen. Im folgenden Beispiel wird eine Benutzerrolle mit dem Namen `sales_rw` erstellt und der Benutzer `test_user` dieser Rolle zugewiesen.

```
create role sales_rw;
grant role sales_rw to test_user;
```

Sie können die Berechtigungen zweier Rollen auch kombinieren, indem Sie eine Rolle explizit einer anderen Rolle zuweisen. Wenn Sie einem Benutzer eine verschachtelte Rolle zuweisen, erhält der Benutzer Berechtigungen für beide Rollen.

```
create role sales_rw;
create role sales_ro;
grant role sales_ro to role sales_rw;
grant role sales_rw to test_user;
```

Um die Liste der Benutzer zu sehen, denen Rollen im Cluster zugewiesen wurden, fragen Sie die Tabelle SVV_USER_GRANTS ab. Um die Liste der Rollen zu sehen, denen Rollen im Cluster zugewiesen wurden, fragen Sie die Tabelle SVV_ROLE_GRANTS ab.

```
select * from svv_user_grants;
select * from svv_role_grants;
```

Zuweisen von Abfragen zu Warteschlangen auf der Grundlage von Benutzergruppen

Wenn der Name einer Benutzergruppe in der Definition einer Warteschlange aufgeführt ist, werden von Mitgliedern dieser Benutzergruppe ausgeführte Abfragen der entsprechenden Warteschlange zugewiesen. Das folgende Beispiel erstellt Benutzergruppen und fügt den Gruppen Benutzern mithilfe der SQL-Befehle CREATE USER, CREATE GROUP und ALTER GROUP hinzu.

```
create group admin_group with user admin246, admin135, sec555;
create user vp1234 in group ad_hoc_group password 'vpPass1234';
alter group admin_group add user analyst44, analyst45, analyst46;
```

Zuweisen einer Abfrage zu einer Abfragegruppe

Sie können eine Abfrage zur Laufzeit einer Warteschlange zuweisen, indem Sie die Abfrage der entsprechenden Abfragegruppe zuweisen. Verwenden Sie zum Starten einer Abfragegruppe den Befehl SET.

```
SET query_group TO group_label
```

Hier ist *group_label* die Beschriftung einer Abfragegruppe, die in der WLM-Konfiguration aufgeführt ist.

Alle Abfragen, die Sie nach dem Befehl `SET query_group` ausführen, werden als Mitglieder der angegebenen Abfragegruppe ausgeführt, bis Sie die Abfragegruppe zurücksetzen oder die aktuelle Anmeldungssitzung beenden. Informationen zum Einstellen und Zurücksetzen von Amazon-Redshift-Objekten finden Sie unter SET und RESET in der SQL-Befehlsreferenz.

Die von Ihnen angegebenen Abfragegruppenbezeichnungen müssen in der aktuellen WLM-Konfiguration enthalten sein. Andernfalls hat der Befehl SET query_group keine Auswirkungen auf Abfragewarteschlangen.

Die in der TO-Klausel definierte Beschriftung wird in den Abfrageprotokollen erfasst und kann daher für Fehlerbehebungszwecke verwendet werden. Für Informationen über den Konfigurationsparameter query_group vgl. query_group in der Konfigurationsreferenz.

Das folgende Beispiel führt zwei Abfragen als Teil der Abfragegruppe „Priorität“ aus. Anschließend wird die Abfragegruppe zurückgesetzt.

```
set query_group to 'priority';
select count(*)from stv_blocklist;
select query, elapsed, substring from svl_qlog order by query desc limit 5;
reset query_group;
```

Zuweisen von Abfragen zur Superuser-Warteschlange

Um eine Abfrage der Superuser-Warteschlange zuzuweisen, melden Sie sich bei Amazon Redshift als Superuser an und führen anschließend die Abfrage in der Superuser-Gruppe aus. Wenn Sie fertig sind, setzen Sie die Abfragegruppe zurück, damit nachfolgende Abfragen nicht in der Superuser-Warteschlange ausgeführt werden.

Im folgenden Beispiel werden zwei Befehle zur Ausführung in der Superuser-Warteschlange zugewiesen.

```
set query_group to 'superuser';

analyze;
vacuum;
reset query_group;
```

Um die Liste der Superusers anzuzeigen, fragen Sie die Systemkatalogtabelle PG_USER ab.

```
select * from pg_user where usesuper = 'true';
```

Dynamische und statische WLM-Konfigurationseigenschaften

Die WLM-Konfigurationseigenschaften sind dynamisch oder statisch. Sie können dynamische Eigenschaften ohne Neustart des Clusters auf die Datenbank anwenden. Statische Eigenschaften erfordern jedoch einen Neustart des Clusters, damit die Änderungen wirksam werden. Wenn Sie jedoch dynamische und statische Eigenschaften gleichzeitig ändern, müssen Sie das Cluster neu starten, damit alle Eigenschaftsänderungen wirksam werden. Dies gilt unabhängig davon, ob die geänderten Eigenschaften dynamisch oder statisch sind.

Während der Anwendung dynamischer Eigenschaften ist der Status des Clusters `modifying`. Das Umschalten zwischen automatischem und manuellem WLM ist eine statische Änderung und macht einen Cluster-Neustart erforderlich.

Die folgende Tabelle zeigt, welche WLM-Eigenschaften dynamisch bzw. statisch sind, wenn automatisches bzw. manuelles WLM verwendet werden.

WLM-Eigenschaft	Automatisches WLM	Manuelles WLM
Abfragegruppen	Dynamisch	Statisch
Abfragegruppenplatzhalter	Dynamisch	Statisch
Benutzergruppen	Dynamisch	Statisch
Benutzergruppenplatzhalter	Dynamisch	Statisch
Benutzerrollen	Dynamisch	Statisch
Platzhalter für die Benutzerrolle	Dynamisch	Statisch
Nebenläufigkeit auf dem Haupt-Cluster	Nicht zutreffend	Dynamisch
Nebenläufigkeitsskalierungsmodus	Dynamisch	Dynamisch
Aktivieren von Short Query Acceleration	Nicht zutreffend	Dynamisch

WLM-Eigenschaft	Automatisches WLM	Manuelles WLM
Maximale Ausführungszeit für kurze Abfragen	Dynamisch	Dynamisch
Prozentsatz des Arbeitsspeichers, der verwendet werden soll	Nicht zutreffend	Dynamisch
Timeout (Zeitüberschreitung)	Nicht zutreffend	Dynamisch
Priorität	Dynamisch	Nicht zutreffend
Hinzufügen oder Entfernen von Warteschlangen	Dynamisch	Statisch

Wenn Sie eine Abfrageüberwachungsregel (Query Monitoring Rule, QMR) ändern, erfolgt die Änderung automatisch, ohne dass der Cluster geändert werden muss.

> **Note**
>
> Wenn bei Verwendung von manuellem WLM der Timeout-Wert geändert wird, wird der neue Wert auf alle Abfragen angewendet, deren Ausführung nach Änderung des Werts beginnt. Bei Änderung der Nebenläufigkeit oder des Prozentsatzes des Arbeitsspeichers, der verwendet werden soll, wird Amazon Redshift dynamisch in die neue Konfiguration geändert. Somit sind derzeit laufende Abfragen von der Änderung nicht betroffen. Weitere Informationen finden Sie unter Dynamische WLM-Speicherzuweisung.

Themen

Dynamische WLM-Speicherzuweisung

In jeder Warteschlange erstellt WLM eine Anzahl von Abfrageslots, die der Nebenläufigkeitsstufe der Warteschlange entspricht. Die Menge des einem Abfrageslot zugewiesenen Speichers entspricht

dem Prozentsatz des der Warteschlange zugewiesenen Speichers, dividiert durch die Anzahl der Slots. Ändern Sie die Speicherzuweisung oder Nebenläufigkeit, verwaltet Amazon Redshift den Übergang zu der neuen WLM-Konfiguration dynamisch. Somit können aktive Abfragen abgeschlossen werden unter Verwendung der derzeit zugewiesenen Speichermenge. Gleichzeitig stellt Amazon Redshift sicher, dass der Gesamtspeicherverbrauch 100 Prozent des verfügbaren Speichers nicht überschreitet.

Der Workload Manager geht bei der Verwaltung des Übergangs wie folgt vor.

1. WLM berechnet die Speicherzuweisung für jeden neuen Abfrageslot neu.
2. Wenn ein Abfrageslot nicht aktiv von einer Abfrage verwendet wird, entfernt WLM den Slot und gibt so Speicher für neue Slots frei.
3. Wenn ein Abfrageslot aktiv verwendet wird, warten WLM auf den Abschluss der Abfrage.
4. Wenn aktive Abfragen abgeschlossen werden, werden leere Slots entfernt, und der entsprechende Speicherplatz wird freigegeben.
5. Wenn ausreichend Speicherplatz verfügbar ist, um einen oder mehrere Slots hinzuzufügen, werden neue Slots hinzugefügt.
6. Wenn alle zum Zeitpunkt der Änderung aktiven Abfragen abgeschlossen sind, entspricht die Zahl der Slots der neuen Nebenläufigkeitsstufe, und der Übergang zu der neuen WLM-Konfiguration ist abgeschlossen.

Tatsächlich verwenden Abfragen, die zum Zeitpunkt der Änderung ausgeführt werden, weiterhin die ursprüngliche Speicherzuweisung. Abfragen, die sich zum Zeitpunkt der Änderung in einer Warteschlange befinden, werden an neue Slots weitergeleitet, sobald diese verfügbar sind.

Wenn die dynamischen WLM-Eigenschaften während des Übergangsvorgangs geändert werden, beginnt WLM sofort mit dem Übergang zu der neuen Konfiguration, beginnend mit dem aktuellen Zustand. Um den Status des Übergangs anzuzeigen, fragen Sie die Systemtabelle STV_WLM_SERVICE_CLASS_CONFIG ab.

Beispiel für dynamische WLM-Eigenschaften

Angenommen, Ihr Cluster-WLM ist mit zwei Warteschlangen konfiguriert, die die folgenden dynamischen Eigenschaften verwenden.

Warteschlange	Gleichzeitigkeit	Zu verwendender Speicherprozentsatz
1	4	50 %
2	4	50 %

Nehmen Sie weiter an, in Ihrem Cluster sind 200 GB Speicher für die Abfrageverarbeitung verfügbar. (Diese Zahl ist willkürlich und dient hier lediglich Demonstrationszwecken.) Wie die folgende Gleichung zeigt, werden jedem Slot 25 GB zugewiesen.

```
(200 GB * 50% ) / 4 slots  = 25 GB
```

Dann ändern Sie Ihr WLM zur Verwendung der folgenden dynamischen Eigenschaften.

Warteschlange	Gleichzeitigkeit	Zu verwendender Speicherprozentsatz
1	3	75 %
2	4	25 %

Wie die folgende Gleichung zeigt, beträgt die neue Speicherzuweisung für jeden Slot in Warteschlange 1 50 GB.

```
(200 GB * 75% ) / 3 slots = 50 GB
```

Nehmen wir an, die Abfragen A1, A2, A3 und A4 werden ausgeführt, während die neue Konfiguration angewendet wird, und die Abfragen B1, B2, B3 und B4 befinden sich in einer Warteschlange. WLM konfiguriert die Abfrageslots dynamisch wie folgt neu.

Schritt	Laufende Abfragen	Aktuelle Slotzahl	Ziel-Slotzahl	Zugewiesener Speicher	Verfügbarer Speicher
1	A1, A2, A3, A4	4	0	100 GB	50 GB

Schritt	Laufende Abfragen	Aktuelle Slotzahl	Ziel-Slotzahl	Zugewiesener Speicher	Verfügbarer Speicher
2	A2, A3, A4	3	0	75 GB	75 GB
3	A3, A4	2	0	50 GB	100 GB
4	A3, A4, B1	2	1	100 GB	50 GB
5	A4, B1	1	1	75 GB	75 GB
6	A4, B1, B2	1	2	125 GB	25 GB
7	B1, B2	0	2	100 GB	50 GB
8	B1, B2, B3	0	3	150 GB	0 GB

1. WLM berechnet die Speicherzuweisung für jeden Abfrageslot neu. Ursprünglich waren Warteschlange 1 100 GB zugewiesen. Die neue Warteschlange hat eine Gesamtzuweisung von 150 GB, der neuen Warteschlange stehen also sofort 50 GB zur Verfügung. Warteschlange 1 verwendet jetzt vier Slots und die neue Nebenläufigkeitsstufe beträgt drei Slots. Es werden also keine neuen Slots hinzugefügt.
2. Wenn eine Abfrage abgeschlossen wird, wird der Slot entfernt, und 25 GB werden freigegeben. Abfrage 1 hat jetzt drei Slots und 75 GB verfügbaren Speicherplatz. Die neue Konfiguration benötigt 50 GB für jeden neuen Slot, die neue Nebenläufigkeitsstufe liegt aber bei drei Slots, es werden daher neue Slots hinzugefügt
3. Wenn eine zweite Abfrage abgeschlossen wird, wird der Slot entfernt, und 25 GB werden freigegeben. Abfrage 1 hat jetzt zwei Slots und 100 GB freien Speicherplatz.
4. Ein neuer Slot mit 50 GB freiem Speicherplatz wird hinzugefügt. Abfrage 1 hat jetzt drei Slots und 50 GB freien Speicherplatz. In einer Warteschlange befindliche Abfragen können jetzt zu dem neuen Slot geleitet werden.
5. Wenn eine dritte Abfrage abgeschlossen wird, wird der Slot entfernt, und 25 GB werden freigegeben. Abfrage 1 hat jetzt zwei Slots und 75 GB freien Speicherplatz.
6. Ein neuer Slot mit 50 GB freiem Speicherplatz wird hinzugefügt. Abfrage 1 hat jetzt drei Slots und 25 GB freien Speicherplatz. In einer Warteschlange befindliche Abfragen können jetzt zu dem neuen Slot geleitet werden.

7. Wenn die vierte Abfrage abgeschlossen wird, wird der Slot entfernt, und 25 GB werden freigegeben. Abfrage 1 hat jetzt zwei Slots und 50 GB freien Speicherplatz.
8. Ein neuer Slot mit 50 GB freiem Speicherplatz wird hinzugefügt. Abfrage q hat jetzt drei Slots mit jeweils 50 GB, und der gesamte verfügbare Speicherplatz wurde zugewiesen.

Der Übergang ist abgeschlossen, und alle Abfrageslots sind für Abfragen aus Warteschlangen verfügbar.

WLM-Abfrageüberwachungsregeln

In Amazon Redshift Workload Management (WLM) definieren Abfrageüberwachungsregeln auf Metriken basierende Leistungsgrenzen für WLM-Warteschlangen und geben an, welche Aktion durchgeführt werden soll, wenn eine Abfrage diese Grenzwerte überschreitet. So können Sie etwa für eine für kurze Abfragen dedizierte Warteschlange eine Regel erstellen, die Abfragen abbricht, die länger als 60 Sekunden ausgeführt werden. Zur Nachverfolgung schlecht gestalteter Abfragen können Sie eine weitere Regel verwenden, die Abfragen mit eingebetteten Schleifen protokolliert.

Sie definieren die Abfrageüberwachungsregeln im Rahmen Ihrer Workload Management (WLM)-Konfiguration. Sie können für jede Warteschlange bis zu 25 Regeln definieren. Dabei liegt die Grenze für alle Warteschlangen insgesamt bei 25 Regeln. Jede Regel enthält bis zu drei Bedingungen (oder Prädikate) sowie eine Aktion. Ein Prädikat besteht aus einer Metrik, einer Vergleichsbedingung (=, <, oder >) und einem Wert. Wenn alle Prädikate für eine Regel erfüllt sind, wird die Aktion der Regel ausgelöst. Mögliche Regelaktionen sind log, hop und abort, wie nachfolgend erläutert.

Die Regeln in einer Warteschlange gelten nur für Abfragen, die in dieser Warteschlange ausgeführt werden. Eine Regel ist unabhängig von anderen Regeln.

WLM evaluiert die Metriken alle 10 Sekunden. Wenn mehr als eine Regel im selben Zeitraum ausgelöst wird, initiiert WLM die schwerwiegendste Aktion – abort, dann hop, dann log. Wenn die Aktion hop oder abort ist, wird die Aktion protokolliert, und die Abfrage wird aus der Warteschlange entfernt. Wenn die Aktion log ist, wird die Abfrage weiterhin in der Warteschlange ausgeführt. WLM initiiert nur eine log-Aktion pro Abfrage und Regel. Wenn die Warteschlange weitere Regeln enthält, bleiben diese wirksam. Wenn die Aktion hop ist und die Abfrage zu einer anderen Warteschlange geleitet wird, gelten die Regeln für die neue Warteschlange. Weitere Informationen zur Abfrageüberwachung und zur Nachverfolgung von Aktionen, die bei bestimmten Abfragen ergriffen wurden, finden Sie in der Beispielsammlung unter Arbeiten mit Short Query Acceleration.

Wenn alle Prädikate einer Regel erfüllt sind, schreibt WLM eine Zeile in die Systemtabelle STL_WLM_RULE_ACTION. Zusätzlich zeichnet Amazon Redshift Abfragemetriken für aktuell ausgeführte Abfragen in auf STV_QUERY_METRICS. Metriken für abgeschlossene Abfragen werden in gespeichert STL_QUERY_METRICS.

Definition einer Abfrageüberwachungsregel

Sie erstellen Abfrageüberwachungsregeln als Teil Ihrer WLM-Konfiguration, die Sie im Rahmen der Parametergruppendefinition für Ihren Cluster definieren.

Sie können Regeln mithilfe von AWS Management Console oder programmgesteuert mithilfe von JSON erstellen.

> **Note**
>
> Wenn Sie Regeln auf programmatischen Wege erstellen, empfehlen wir nachdrücklich, die Konsole zu verwenden, um die JSON-Elemente zu erstellen, die Sie in der Parametergruppendefinition verwenden. Weitere Informationen finden Sie unter Erstellen oder Modifizieren einer Abfrageüberwachungsregel mit der Konsole und Konfiguration von Parameterwerten mit der AWS CLI im Amazon-Redshift-Verwaltungshandbuch.

Zur Definition einer Abfrageüberwachungsregel geben Sie die folgenden Elemente an:

- Einen Regelnamen – Regelnamen müssen innerhalb der WLM-Konfiguration eindeutig sein. Regelnamen können aus bis zu 32 alphanumerischen Zeichen oder Unterstrichen bestehen und dürfen keine Leerzeichen oder Anführungszeichen enthalten. Sie können bis zu 25 Regeln pro Warteschlange und insgesamt 25 Regeln für alle Warteschlangen festlegen.
- Ein oder mehrere Prädikat(e) – Sie können bis zu drei Prädikate pro Regel angeben. Wenn alle Prädikate für eine Regel erfüllt sind, wird die Aktion der Regel ausgelöst. Ein Prädikat wird durch einen Metriknamen, einen Operator (=, <, or >) und einen Wert definiert. Ein Beispiel ist `query_cpu_time > 100000`. Für eine Liste der Metriken und Beispiele für Werte bei unterschiedlichen Metriken vgl. Abfrageüberwachungsmetriken für Amazon Redshift wurden bereitgestellt unten in diesem Abschnitt.
- Eine Aktion – Wenn mehr als eine Regel ausgelöst wird, wählt WLM die Regel mit der schwerwiegendsten Aktion. Mögliche Aktionen, aufsteigend nach Schweregrad, sind:
  - Log – Aufzeichnung von Informationen zu der Abfrage in der Systemtabelle STL_WLM_RULE_ACTION. Verwenden Sie diese Aktion, wenn Sie lediglich einen

Protokolleintrag schreiben möchten. WLM erstellt höchstens einen Protokolleintrag pro Abfrage und Regel. Nach einer Log-Aktion bleiben andere Regeln wirksam, und WLM überwacht die Abfrage weiter.

- Hop (Nur bei manuellem WLM verfügbar) – Protokollieren der Aktion und Hopping der Abfrage zur nächsten übereinstimmenden Warteschlange. Wenn keine andere passende Warteschlange vorhanden ist, wird die Abfrage abgebrochen. QMR führt Hopping nur für CREATE TABLE AS (CTAS)-Anweisungen und schreibgeschützte Abfragen aus, beispielsweise SELECT-Anweisungen. Weitere Informationen finden Sie unter WLM-Abfragewarteschlangen-Hopping.
- Abort – Protokollieren der Aktion und Abbrechen der Abfrage. QMR stoppt keine COPY-Anweisungen und Wartungsoperationen wie ANALYZE und VACUUM.
- Ändern der Priorität (nur bei automatischem WLM verfügbar) – Ändern der Priorität einer Warteschlange.

Zur Begrenzung der Laufzeit von Abfragen empfehlen wir das Erstellen einer Abfrageüberwachungsregel anstelle eines WLM-Timeouts. Sie können beispielsweise `max_execution_time` auf 50.000 Millisekunden festlegen, wie im folgenden JSON-Snippet gezeigt.

```
"max_execution_time": 50000
```

Sie sollten jedoch stattdessen besser eine gleichwertige Abfrageüberwachungsregel definieren, die `query_execution_time` auf 50 Sekunden festlegt wie im folgenden JSON-Snippet gezeigt.

```
"rules":
[
    {
        "rule_name": "rule_query_execution",
        "predicate": [
            {
                "metric_name": "query_execution_time",
                "operator": ">",
                "value": 50
            }
        ],
        "action": "abort"
    }
]
```

Informationen zu den Schritten zum Erstellen oder Ändern einer Abfrageüberwachungsregel finden Sie unter Erstellen oder Modifizieren einer Abfrageüberwachungsregel mit der Konsole und Eigenschaften im Parameter wlm_json_configuration im Amazon-Redshift-Verwaltungshandbuch.

Weitere Informationen zu Abfrageüberwachungsregeln finden Sie in den folgenden Themen:

- Abfrageüberwachungsmetriken für Amazon Redshift wurden bereitgestellt
- Vorlagen für Abfrageüberwachungsregeln
- Erstellen einer Regel mit der Konsole
- Workload-Management-Konfiguration
- Systemtabellen und Ansichten für Abfrageüberwachungsregeln

Abfrageüberwachungsmetriken für Amazon Redshift wurden bereitgestellt

Die folgende Tabelle beschreibt die Metriken für Abfrageüberwachungsregeln. (Diese Metriken unterscheiden sich von den in den Systemtabellen STV_QUERY_METRICS und STL_QUERY_METRICS gespeicherten Metriken.)

Für eine Metrik wird der Leistungsschwellenwert auf Abfrage- oder auf Segmentebene verfolgt. Für weitere Informationen zu Segmenten und Schritten vgl. Workflow der Abfrageplanung und -ausführung.

> **Note**
>
> Der Parameter WLM-Timeout unterscheidet sich von Abfrageüberwachungsregeln.

Metrik	Name	Beschreibung
Abfrage-CPU-Zeit	`query_cpu_time`	Von der Abfrage verwendete CPU-Zeit, in Sekunden. `CPU time` unterscheidet sich von `Query execution time`. Gültige Werte liegen zwischen 0 und 999 999.
Gelesene Blöcke	`query_blocks_read`	Anzahl der von der Abfrage gelesenen 1 MB-Blöcke.

Metrik	Name	Beschreibung
		Gültige Werte liegen zwischen 0 und 1 048 575.
Anzahl der Scan-Zeilen	`scan_row_count`	Die Anzahl der Zeilen in einem Scan-Schritt. Die Anzahl der Zeilen ist die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter ausgegeben wurden. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.
Abfrageausführungszeit	`query_execution_time`	Verstrichene Ausführungszeit für eine Abfrage, in Sekunden. Die Ausführungszeit enthält nicht die in einer Warteschlange verbrachte Zeit. Gültige Werte liegen zwischen 0 und 86 399.
Zeit in Abfragewarteschlange	`query_queue_time`	Wartezeit in einer Warteschlange in Sekunden. Gültige Werte liegen zwischen 0 und 86 399.
CPU-Verwendung	`query_cpu_usage_percent`	Prozentsatz der von der Abfrage verwendeten CPU-Kapazität. Gültige Werte liegen zwischen 0 und 6 399.
Speicher zu Festplatte	`query_temp_blocks_to_disk`	Der temporäre Festplattenspeicherplatz, der zum Schreiben von Zwischenergebnissen verwendet wird, in 1 MB-Blöcken. Gültige Werte liegen zwischen 0 und 319 815 679.

Metrik	Name	Beschreibung
CPU-Verzerrung	`cpu_skew`	Das Verhältnis der maximalen CPU-Nutzung für einen Slice zur durchschnittlichen CPU-Nutzung für alle Slices. Diese Metrik wird auf Segmentebene definiert. Gültige Werte liegen zwischen 0 und 99.
I/O-Verzerrung	`io_skew`	Das Verhältnis der maximalen Zahl der gelesenen Blöcke (I/O) für einen Slice zur durchschnittlichen Zahl der gelesenen Blöcke für alle Slices. Diese Metrik wird auf Segmentebene definiert. Gültige Werte liegen zwischen 0 und 99.
Verbundene Zeilen	`join_row_count`	Die Anzahl der in einem Join-Schritt verarbeiteten Zeilen. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.
Zahl der Zeilen mit eingebetteten Loop-Joins.	`nested_loop_join_row_count`	Die Anzahl der Zeilen in einem eingebetteten Loop-Join. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.
Anzahl der Rückgabezeilen.	`return_row_count`	Die Anzahl der von der Abfrage ausgegebenen Zeilen. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.

Metrik	Name	Beschreibung
Segment-Ausführungszeit.	`segment_execution_time`	Verstrichene Ausführungszeit für ein einzelnes Segment, in Sekunden. Nehmen Sie zur Vermeidung oder Reduzierung von Samplingfehlern `segment_execution_time > 10` in Ihre Regeln auf. Gültige Werte liegen zwischen 0 und 86 388.
Anzahl der Spectrum-Scan-Zeilen	`spectrum_scan_row_count`	Die Anzahl der Datenzeilen in Amazon S3, die von einer Amazon-Redshift-Spectrum-Abfrage gescannt wurden. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.
Spektrum-Scan-Größe.	`spectrum_scan_size_mb`	Die Datengröße in Amazon S3, in MB, die von einer Amazon-Redshift-Spectrum-Abfrage gescannt wurde. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.
Abfragepriorität	`query_priority`	Die Priorität der Abfrage. Gültige Werte sind `HIGHEST`, `HIGH`, `NORMAL`, `LOW` und `LOWEST`. Beim Vergleich von `query_priority` mittels der Operatoren „größer als (>)“ und „kleiner als (<)“ ist `HIGHEST` größer als `HIGH`, `HIGH` größer als `NORMAL` usw.

Note

- Die Hop-Aktion wird mit dem Prädikat `query_queue_time` nicht unterstützt. Das heißt, Regeln, die zum Durchführen von Hopping definiert werden, wenn ein `query_queue_time`-Prädikat erfüllt wird, werden ignoriert.

- Kurze Segmentausführungszeiten können bei einigen Metriken zu Samplingfehlern führen, etwa bei `io_skew` und `query_cpu_usage_percent`. Nehmen Sie zur Vermeidung oder Reduzierung von Samplingfehlern die Segmentausführungszeit in Ihre Regeln auf. Ein guter Ausgangspunkt ist `segment_execution_time > 10`.

Die Ansicht SVL_QUERY_METRICS zeigt die Metriken für abgeschlossene Abfragen. Die Ansicht SVL_QUERY_METRICS_SUMMARY zeigt die maximalen Werte der Metriken für abgeschlossene Abfragen. Verwenden Sie die Werte in diesen Ansichten als Hilfe zur Bestimmung der Schwellenwerte für die Definition von Abfrageüberwachungsregeln.

Abfrageüberwachungsmetriken für Amazon Redshift Serverless

Die folgende Tabelle beschreibt die Metriken, die in Abfrageüberwachungsregeln für Amazon Redshift Serverless verwendet werden.

Metrik	Name	Beschreibung
Abfrage-CPU-Zeit	`max_query_cpu_time`	Von der Abfrage verwendete CPU-Zeit, in Sekunden. `CPU time` unterscheidet sich von `Query execution time`. Gültige Werte liegen zwischen 0 und 999 999.
Gelesene Blöcke	`max_query_blocks_read`	Anzahl der von der Abfrage gelesenen 1 MB-Blöcke. Gültige Werte liegen zwischen 0 und 1 048 575.
Anzahl der Scan-Zeilen	`max_scan_row_count`	Die Anzahl der Zeilen in einem Scan-Schritt. Die Anzahl der Zeilen ist die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter ausgegeben wurden.

Metrik	Name	Beschreibung
		Gültige Werte liegen zwischen 0 und 999 999 999 999 999.
Abfrageausführungszeit	max_query_execution_time	Verstrichene Ausführungszeit für eine Abfrage, in Sekunden. Die Ausführungszeit enthält nicht die in einer Warteschlange verbrachte Zeit. Wenn eine Abfrage die festgelegte Ausführungszeit überschreitet, stoppt Amazon Redshift Serverless die Abfrage. Gültige Werte liegen zwischen 0 und 86 399.
Zeit in Abfragewarteschlange	`max_query_queue_time`	Wartezeit in einer Warteschlange in Sekunden. Gültige Werte liegen zwischen 0 und 86 399.
CPU-Verwendung	`max_query_cpu_usage_percent`	Prozentsatz der von der Abfrage verwendeten CPU-Kapazität. Gültige Werte liegen zwischen 0 und 6 399.
Speicher zu Festplatte	`max_query_temp_blocks_to_disk`	Der temporäre Festplattenspeicherplatz, der zum Schreiben von Zwischenergebnissen verwendet wird, in 1 MB-Blöcken. Gültige Werte liegen zwischen 0 und 319 815 679.
Verbundene Zeilen	`max_join_row_count`	Die Anzahl der in einem Join-Schritt verarbeiteten Zeilen. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.

Metrik	Name	Beschreibung
Zahl der Zeilen mit eingebetteten Loop-Joins.	`max_nested_loop_join_row_count`	Die Anzahl der Zeilen in einem eingebetteten Loop-Join. Gültige Werte liegen zwischen 0 und 999 999 999 999 999.

> **Note**
>
> - Die Hop-Aktion wird mit dem Prädikat `max_query_queue_time` nicht unterstützt. Das heißt, Regeln, die zum Durchführen von Hopping definiert werden, wenn ein `max_query_queue_time`-Prädikat erfüllt wird, werden ignoriert.
> - Kurze Segmentausführungszeiten können bei einigen Metriken zu Samplingfehlern führen, etwa bei `max_io_skew` und `max_query_cpu_usage_percent`.

Vorlagen für Abfrageüberwachungsregeln

Beim Hinzufügen einer Regel mit der Amazon-Redshift-Konsole können Sie eine Regel aus einer vordefinierten Vorlage erstellen. Amazon Redshift erstellt eine neue Regel mit einem Satz von Prädikaten und füllt die Prädikate mit Standardwerten aus. Die Standardaktion ist log. Sie können die Prädikate und die Aktion an Ihren Anwendungsfall anpassen.

In der folgenden Tabelle werden die verfügbaren Vorlagen aufgeführt.

Name der Vorlage	Prädikate	Beschreibung
Nested loop join	`nested_loop_join_row_count > 100`	Ein eingebetteter Loop-Join kann auf ein unvollständiges Join-Prädikat hindeuten; diese führen oft zu sehr großen Ausgabesätzen (einem cartesischen Produkt). Verwenden Sie eine geringe Zahl von Zeilen, um eine mögliche Ausreißerabfrage frühzeitig zu finden.

Name der Vorlage	Prädikate	Beschreibung
Die Abfrage gibt eine sehr große Zahl von Zeilen aus.	`return_row_count > 1000000`	Wenn Sie eine Warteschlange für einfache und kurze Abfragen einrichten, können Sie eine Regel anwenden, die Abfragen mit einer großen Zahl ausgegebener Zeilen findet. Die Vorlage verwendet einen Standardwert von 1 Million Zeilen. Bei manchen Systemen kann 1 Million ein zu hoher Wert sein, während in anderen erst ein Wert von 1 Milliarde oder mehr als hoch gilt.
Join mit großer Zahl von Zeilen	`join_row_count > 1000000000`	Ein Join-Schritt mit einer großen Zahl von Zeilen kann darauf hindeuten, dass restriktivere Filter erforderlich sind. Die Vorlage verwendet einen Standardwert von 1 Milliarde Zeilen. Für eine (einmalige) Ad-Hoc-Warteschlange für schnelle und einfache Abfragen sollten Sie eine geringere Zahl verwenden.
Hohe Festplattennutzung beim Schreiben von Zwischenergebnissen	`query_temp_blocks_to_disk > 100000`	Wenn gleichzeitig ausgeführte Abfragen mehr als den verfügbaren System-RAM verwenden, schreibt die Abfrageausführungsengine Zwischenergebnisse auf die Festplatte. Typischerweise ist dies das Ergebnis einer fehlerhaften Abfrage; solche Abfragen verwenden normalerweise auch den meisten Festplattenspeicherplatz. Der akzeptable Schwellenwert für die Festplattennutzung variiert je nach Typ des Clusterknotens und der Anzahl der Knoten. Die Vorlage verwendet einen Standardwert von 100.000 Blöcken bzw. 100 GB. Für einen kleineren Cluster werden Sie wahrscheinlich einen geringeren Wert verwenden.

Name der Vorlage	Prädikate	Beschreibung
Lang dauernde Abfrage mit hoher I/O-Verzerrung	`segment_execution_time > 120` und `io_skew > 1.30`	I/O-Verzerrung tritt auf, wenn ein Knotenslice eine viel höhere I/O-Rate aufweist als die anderen Slices. Als Ausgangspunkt gilt eine Verzerrung von 1,30 (das 1,3-fache des Durchschnitts) als hoch. Eine hohe I/O-Verzerrung ist nicht immer ein Problem. in Verbindung mit langen Abfrageausführungszeiten kann sie jedoch auf ein Problem mit dem Verteilungsstil oder dem Sortierschlüssel hindeuten.

Systemtabellen und Ansichten für Abfrageüberwachungsregeln

Wenn alle Prädikate einer Regel erfüllt sind, schreibt WLM eine Zeile in die Systemtabelle STL_WLM_RULE_ACTION. Diese Zeile enthält Details zu der Abfrage, die die Regel ausgelöst hat, und die daraus resultierende Aktion.

Darüber hinaus zeichnet Amazon Redshift Abfragemetriken der folgenden Systemtabellen und Ansichten auf.

- Die Tabelle STV_QUERY_METRICS zeigt die Metriken für aktuell ausgeführte Abfragen an.
- Die Tabelle STL_QUERY_METRICS zeichnet die Metriken für abgeschlossene Abfragen auf.
- Die Ansicht SVL_QUERY_METRICS zeigt die Metriken für abgeschlossene Abfragen.
- Die Ansicht SVL_QUERY_METRICS_SUMMARY zeigt die maximalen Werte der Metriken für abgeschlossene Abfragen.

WLM-Systemtabellen und Ansichten

WLM konfiguriert Abfragewarteschlangen anhand von intern definierten WLM-Serviceklassen. Amazon Redshift erstellt mehrere interne Warteschlangen gemäß diesen Serviceklassen, zusammen mit den in der WLM-Konfiguration definierten Warteschlangen. Die Begriffe Warteschlange und Serviceklasse haben in den Systemtabellen häufig dieselbe Bedeutung. Die Superuser-Warteschlange verwendet Serviceklasse 5. Benutzerdefinierte Warteschlangen verwenden Service-Klasse 6 und darüber.

Sie können den Status von Abfragen, Warteschlangen und Service-Klassen mit WLM-spezifischen Systemtabellen anzeigen. Fragen Sie die folgenden Systemtabellen ab, um Folgendes zu tun:

- Anzeige, welche Abfragen verfolgt und welche Ressourcen vom Workload Manager zugewiesen werden.
- Anzeige, welcher Warteschlange eine Abfrage zugewiesen wurde.
- Anzeige des Status einer Abfrage, die derzeit vom Workload Manager verfolgt wird.

Tabellenname	Beschreibung
STL_WLM_ERROR	Enthält ein Protokoll WLM-bezogener Fehlerereignisse.
STL_WLM_QUERY	Führt Abfragen auf, die von WLM verfolgt werden.
STV_WLM_CLASSIFICATION_CONFIG	Zeigt die aktuellen Klassifizierungsregeln für WLM.
STV_WLM_QUERY_QUEUE_STATE	Zeichnet den aktuelles Status der Abfragewarteschlangen auf.
STV_WLM_QUERY_STATE	Bietet einen Snapshot des aktuellen Status der von WLM verfolgten Abfragen.
STV_WLM_QUERY_TASK_STATE	Enthält den derzeitigen Status der Abfrageaufgaben.
STV_WLM_SERVICE_CLASS_CONFIG	Zeichnet die Service-Klassen-Konfigurationen für WLM auf.
STV_WLM_SERVICE_CLASS_STATE	Enthält den derzeitigen Status der Service-Klassen.
STL_WLM_RULE_ACTION	Zeigt Einzelheiten zu Aktionen aufgrund von WLM-Abfra geüberwachungsregeln im Zusammenhang mit benutzerd efinierten Abfragen auf.
STV_WLM_QMR_CONFIG	Zeichnet die Konfiguration für WLM-Abfrageüberwac hungsregeln (QMR) auf.

Sie verwenden die Task-ID zur Verfolgung einer Abfrage in den Systemtabellen. Das folgende Beispiel zeigt, wie Sie die Task-ID der zuletzt übermittelten Benutzerabfrage abrufen:

```
select task from stl_wlm_query where exec_start_time =(select max(exec_start_time) from
 stl_wlm_query);

task
------
137
(1 row)
```

Das folgende Beispiel zeigt Abfragen an, die derzeit ausgeführt werden oder in verschiedenen Service-Klassen (Warteschlangen) warten. Diese Abfrage ist nützlich für die Verfolgung des allgemeinen gleichzeitigen Workloads für Amazon Redshift:

```
select * from stv_wlm_query_state order by query;


xid |task|query|service_| wlm_start_  |  state  |queue_ | exec_
    |    |     |class   | time        |         |time   | time
----+----+-----+--------+-------------+---------+-------+--------
2645| 84 | 98  | 3      | 2010-10-... |Returning|   0   | 3438369
2650| 85 | 100 | 3      | 2010-10-... |Waiting  |   0   | 1645879
2660| 87 | 101 | 2      | 2010-10-... |Executing|   0   | 916046
2661| 88 | 102 | 1      | 2010-10-... |Executing|   0   | 13291
(4 rows)
```

WLM-Serviceklassen-IDs

Die folgende Tabelle listet die den Serviceklassen zugeordneten IDs auf.

ID	Serviceklassen-ID.
1–4	Reserviert für die Systemverwendung.
5	Wird von der Superuser-Warteschlange verwendet.
6–13	Wird von den Warteschlangen des manuellen WLM verwendet, die in der WLM-Konfiguration definiert sind.

ID	Serviceklassen-ID.
14	Wird von Short Query Acceleration verwendet.
15	Reserviert für Wartungsaktivitäten, die von Amazon Redshift durchgeführt werden.
100–107	Wird von den automatischen WLM-Warteschlangen verwendet, wenn auto_wlm „true“ ist.

Verwalten der Datenbanksicherheit

Themen

Sie verwalten die Datenbanksicherheit, indem Sie steuern, welche Benutzer auf welche Datenbankobjekte zugreifen dürfen.

Der Zugriff auf Datenbankobjekte hängt von den Berechtigungen ab, die Sie Benutzern oder Gruppen gewähren. Die folgenden Leitlinien den fassen zusammen, wie Datenbanksicherheit funktioniert:

- Standardmäßig erhält nur der Besitzer eines Objekts Zugriffsberechtigungen.
- Amazon-Redshift-Datenbankbenutzer sind benannte Benutzer, die eine Verbindung zu der Datenbank herstellen können. Benutzern können Zugriffsberechtigungen gewährt werden. Dies kann explizit erfolgen, indem die betreffenden Berechtigungen dem Benutzerkonto direkt gewährt werden, oder implizit, indem die betreffenden Berechtigungen einer Gruppe gewährt werden, der das Benutzerkonto angehört.
- Eine Gruppe ist eine Menge von Benutzerkonten, denen zusammen Berechtigungen zugewiesen werden können, um so die Sicherheitswartung zu optimieren.
- Schemata sind Mengen von Datenbanktabellen und anderen Datenbankobjekten. Schemata sind mit Verzeichnissen eines Dateisystems vergleichbar, sie ermöglichen jedoch keine Verschachtelung. Benutzern kann der Zugriff auf ein einzelnes Schema oder auf mehrere Schemata gewährt werden.

Darüber hinaus verwendet Amazon Redshift die folgenden Funktionen, um Ihnen eine genauere Kontrolle darüber zu ermöglichen, welche Benutzer auf die jeweiligen Datenbankobjekte zugreifen können:

- Mit der rollenbasierten Zugriffskontrolle (RBAC) können Sie Rollen Berechtigungen zuweisen und dann auf Benutzer anwenden, sodass Sie die Berechtigungen für große Benutzergruppen kontrollieren können. Im Gegensatz zu Gruppen können Rollen Berechtigungen von anderen Rollen übernehmen.

  Mit der Sicherheit auf Zeilenebene (RLS) können Sie Richtlinien definieren, die den Zugriff auf Zeilen Ihrer Wahl einschränken, und diese Richtlinien dann auf Benutzer oder Gruppen anwenden.

  Die dynamische Datenmaskierung (DDM) schützt Ihre Daten zusätzlich, indem diese während der Abfragelaufzeit transformiert werden, sodass Sie Benutzern Zugriff auf Daten gewähren können, ohne vertrauliche Informationen offenzulegen.

Beispiele von Sicherheitsimplementierungen finden Sie unter Beispiel zur Steuerung des Zugriffs durch Benutzer und Gruppen.

Weitere Informationen zum Schutz Ihrer Daten finden Sie unter Sicherheit in Amazon Redshift im Amazon-Redshift-Verwaltungshandbuch.

Übersicht zur Sicherheit in Amazon Redshift

Die Datenbanksicherheit in Amazon Redshift unterscheidet sich von anderen Sicherheitsfunktionen in Amazon Redshift. Neben der Datenbanksicherheit, die in diesem Abschnitt ausführlich beschrieben wird, bietet Amazon Redshift die folgenden Funktionen zur Verwaltung der Sicherheit:

- Anmeldeinformationen — Der Zugriff auf Ihre Amazon Redshift AWS Management Console wird durch Ihre AWS Kontoberechtigungen gesteuert. Weitere Informationen finden Sie unter Anmeldeinformationen.
- Zugriffsverwaltung — Um den Zugriff auf bestimmte Amazon Redshift Redshift-Ressourcen zu kontrollieren, definieren Sie AWS Identity and Access Management (IAM-) Konten. Weitere Informationen finden Sie unter Steuern des Zugriffs auf Amazon-Redshift-Ressourcen.
- Clustersicherheitsgruppen – Um anderen Benutzern den eingehenden Zugriff auf einen Amazon-Redshift-Cluster zu gewähren, definieren Sie eine Clustersicherheitsgruppe und verknüpfen

sie mit einem Cluster. Weitere Informationen finden Sie unter Amazon-Redshift-Cluster-Sicherheitsgruppen.

- VPC – Sie können den Zugriff auf Ihren Cluster unter Verwendung einer virtuellen Netzwerkumgebung steuern, wenn Sie Ihren Cluster in einer Amazon Virtual Private Cloud (VPC) starten. Weitere Informationen finden Sie unter Verwalten von Clustern in einer Virtual Private Cloud (VPC).
- Clusterverschlüsselung – Sie können die Daten in allen von Benutzern erstellten Tabellen verschlüsseln, indem Sie beim Starten des Clusters die Clusterverschlüsselung aktivieren. Weitere Informationen finden Sie unter Amazon-Redshift-Cluster.
- SSL-Verbindungen – Sie können Sie eine SSL-Verschlüsselung (Secure Sockets Layer) verwenden, wenn Sie die Verbindung zwischen Ihrem SQL-Client und Ihrem Cluster verschlüsseln möchten. Weitere Informationen finden Sie unter Verbinden mit Ihrem Cluster über SSL.
- Ladedatenverschlüsselung – Sie können die Dateien mit den Tabellenladedaten beim Hochladen zu Amazon S3 serverseitig oder clientseitig verschlüsseln. Wenn Sie serverseitig verschlüsselte Daten laden, führt Amazon S3 die Entschlüsselung transparent durch. Wenn Sie clientseitig verschlüsselte Daten laden, werden die Daten beim Laden der Tabelle mit dem Amazon-Redshift-Befehl COPY entschlüsselt. Weitere Informationen finden Sie unter Hochladen verschlüsselter Daten in Amazon S3.
- Daten während der Übertragung — Um Ihre Daten bei der Übertragung innerhalb der AWS Cloud zu schützen, verwendet Amazon Redshift hardwarebeschleunigtes SSL für die Kommunikation mit Amazon S3 oder Amazon DynamoDB für KOPIER-, ENTLADEN-, Sicherungs- und Wiederherstellungsvorgänge.
- Zugriffssteuerung auf Spaltenebene – Um die Zugriffssteuerung auf Spaltenebene für Daten in Amazon Redshift zu nutzen, verwenden Sie Gewähren- und Widerrufen-Anweisungen auf Spaltenebene, ohne ansichtsbasierte Zugriffssteuerung implementieren oder ein anderes System verwenden zu müssen.
- Sicherheitskontrolle auf Zeilenebene – Um die Sicherheit von Daten in Amazon Redshift auf Zeilenebene zu kontrollieren, müssen Sie Richtlinien erstellen und Rollen oder Benutzern zuordnen, die den Zugriff auf die in der Richtlinie definierten Zeilen einschränken.

Standardberechtigungen für Datenbankbenutzer

Wenn Sie ein Datenbankobjekt erstellen, sind Sie dessen Besitzer. Standardmäßig sind nur Superuser und der Besitzer eines Objekts berechtigt, Abfragen für dieses Objekt durchzuführen, es zu bearbeiten oder Berechtigungen dafür zu erteilen. Wenn Sie möchten, dass ein Benutzer ein

Objekt verwenden darf, müssen Sie dem Benutzer (bzw. einer Gruppe, der der Benutzer angehört) die benötigten Berechtigungen erteilen. Datenbank-Superuser haben dieselben Berechtigungen wie die Besitzer von Datenbanken.

Amazon Redshift unterstützt die folgenden Berechtigungen: SELECT, INSERT, UPDATE, DELETE, REFERENCES, CREATE, TEMPORARY und USAGE. Für die verschiedenen Objekttypen kommen jeweils bestimmte Berechtigungen in Frage. Weitere Informationen zu den Berechtigungen für von Amazon Redshift unterstützte Datenbankobjekte finden Sie unter dem Befehl GRANT.

Nur der Eigentümer hat die Berechtigung zum Ändern oder Löschen eines Objekts.

Standardmäßig besitzen alle Benutzer CREATE- und USAGE-Berechtigungen für das Schema PUBLIC einer Datenbank. Um Benutzer daran zu hindern, Objekte im Schema PUBLIC einer Datenbank zu erstellen, verwenden Sie den Befehl REVOKE, um diese Berechtigung zu entfernen.

Um erteilte Berechtigungen zu widerrufen, verwenden Sie den Befehl REVOKE. Die Berechtigungen des Besitzers des Objekts (wie DROP, GRANT und REVOKE) sind implizit und können nicht direkt erteilt oder widerrufen werden. Die Besitzer von Objekten können auch ihre eigenen Berechtigungen zurückziehen. Dies kann beispielsweise nützlich sein, um eine Tabelle für alle Benutzer (sich selbst eingeschlossen) vor Änderungen zu schützen, indem sie schreibgeschützt gemacht wird. Superuser behalten unabhängig von GRANT- und REVOKE-Befehlen stets alle Berechtigungen.

Superuser

Datenbank-Superuser haben für alle Datenbanken dieselben Berechtigungen wie die Besitzer von Datenbanken.

Der Admin-Benutzer ist der Benutzer, den Sie beim Starten des Clusters erstellt haben, und wird als Superuser bezeichnet.

Sie müssen Superuser sein, um einen anderen Superuser zu erstellen.

Amazon-Redshift-Systemtabellen und Systemsichten sind entweder nur für Superuser sichtbar oder aber für alle Benutzer. Nur Superuser haben die Berechtigung, Abfragen über Systemtabellen und Systemsichten durchzuführen, die als „nur für Superuser sichtbar“ festgelegt sind. Weitere Informationen finden Sie unter Systemtabellen und Ansichten.

Superuser können alle Katalogtabellen anzeigen. Weitere Informationen finden Sie unter Systemkatalogtabellen.

Datenbank-Superuser umgehen alle Überprüfungen von Berechtigungen. Superuser behalten unabhängig von GRANT- und REVOKE-Befehlen stets alle Berechtigungen. Als Superuser sollten Sie vorsichtig vorgehen. Wir empfehlen, allgemeine Aufgaben unter Verwendung eines anderen Benutzerkontos und nicht als Superuser durchzuführen. Sie können eine Administratorrolle mit restriktiveren Berechtigungen erstellen. Weitere Informationen zum Erstellen von Rollen finden Sie unter Rollenbasierte Zugriffskontrolle (RBAC).

Um einen neuen Datenbank-Superuser zu erstellen, melden Sie sich an der Datenbank als ein Superuser an, führen den Befehl CREATE USER oder den Befehl ALTER USER mit der Berechtigung CREATEUSER aus.

```
CREATE USER adminuser CREATEUSER PASSWORD '1234Admin';
ALTER USER adminuser CREATEUSER;
```

Um einen Superuser zu erstellen, zu ändern oder zu löschen, verwenden Sie dieselben Befehle wie zur Verwaltung der Benutzer. Weitere Informationen finden Sie unter Erstellen, Modifizieren und Löschen von Benutzern.

Benutzer

Sie können Datenbankbenutzer mit den Amazon Redshift-SQL-Befehlen CREATE USER und ALTER USER erstellen und verwalten. Oder Sie können Ihren SQL-Client mit benutzerdefinierten Amazon-Redshift-JDBC- oder -ODBC-Treibern konfigurieren. Diese Treiber verwalten die Erstellung von Datenbankbenutzern und temporären Passwörtern als Teil der Datenbankanmeldung.

Die Treiber authentifizieren Datenbankbenutzer auf der Grundlage der (IAM AWS Identity and Access Management -) Authentifizierung. Wenn Sie Benutzeridentitäten bereits außerhalb von verwalten AWS, können Sie einen SAML 2.0-kompatiblen Identitätsanbieter (IdP) verwenden, um den Zugriff auf Amazon Redshift Redshift-Ressourcen zu verwalten. Sie verwenden eine IAM-Rolle, um Ihren IdP zu konfigurieren und Ihren Verbundbenutzern AWS zu ermöglichen, temporäre Datenbankanmeldedaten zu generieren und sich Amazon Redshift Redshift-Datenbanken anzumelden. Weitere Informationen finden Sie unter Verwenden der IAM-Authentifizierung zur Erstellung von Benutzeranmeldeinformationen für die Datenbank.

Amazon-Redshift-Benutzer können ausschließlich von Datenbank-Superusern erstellt und gelöscht werden. Benutzer werden bei der Amazon-Redshift-Anmeldung authentifiziert. Sie können Datenbanken und Datenbankobjekte (beispielsweise Tabellen) besitzen. Außerdem können sie für diese Objekte Benutzern, Gruppen und Schemata Berechtigungen erteilen, um zu steuern, wer auf

welches Objekt zugreifen darf. Benutzer mit CREATE DATABASE-Rechten dürfen Datenbanken erstellen und können für diese Datenbanken Berechtigungen erteilen. Superuser haben für alle Datenbanken dieselben Berechtigungen wie deren Besitzer.

Erstellen, Modifizieren und Löschen von Benutzern

Datenbankbenutzer gelten global im gesamten Data-Warehouse-Cluster (und nicht nur für einzelne Datenbanken).

- Verwenden Sie den Befehl CREATE USER, um einen neuen Benutzer zu erstellen.
- Verwenden Sie den Befehl CREATE USER mit der Option CREATEUSER, um einen Superuser zu erstellen.
- Mit dem Befehl DROP USER können Sie einen bestehenden Benutzer entfernen.
- Verwenden Sie den Befehl ALTER USER, um einen Benutzer zu ändern und zum Beispiel ein neues Passwort festzulegen.
- Führen Sie eine Abfrage wie folgt über der Katalogtabelle PG_USER aus, um eine Liste aller Benutzer anzuzeigen.

```
select * from pg_user;

  usename   | usesysid | usecreatedb | usesuper | usecatupd |  passwd  | valuntil |
 useconfig
------------+----------+-------------+----------+-----------+----------+----------
+-----------
 rdsdb      |        1 | t           | t        | t         | ******** |          |
 masteruser |      100 | t           | t        | f         | ******** |          |
 dwuser     |      101 | f           | f        | f         | ******** |          |
 simpleuser |      102 | f           | f        | f         | ******** |          |
 poweruser  |      103 | f           | t        | f         | ******** |          |
 dbuser     |      104 | t           | f        | f         | ******** |          |
(6 rows)
```

Gruppen

Gruppen sind eine Menge von Benutzern, die alle die Berechtigungen erhalten, die der Gruppe zugewiesen sind. Sie können Gruppen dazu verwenden, Berechtigungen zuzuweisen. Sie können beispielsweise verschiedene Gruppen für Vertrieb, Verwaltung und Support erstellen, und den Benutzern in diesen Gruppen den jeweils für ihre Aufgaben benötigten Datenzugriff erlauben. Sie

können dann Berechtigungen auf der Ebene von Gruppen erteilen und zurücknehmen, und diese Änderungen gelten dann für alle Benutzer der Gruppe, mit Ausnahme der Superuser.

Führen Sie eine Abfrage wie folgt über der Systemkatalogtabelle PG_GROUP aus, um eine Liste aller Benutzergruppen anzuzeigen:

```
select * from pg_group;
```

Wenn Sie beispielsweise alle Datenbankbenutzer nach Gruppen auflisten möchten, führen Sie den folgenden SQL-Befehl aus.

```
SELECT u.usesysid
,g.groname
,u.usename
FROM pg_user u
LEFT JOIN pg_group g ON u.usesysid = ANY (g.grolist)
```

Erstellen, Modifizieren und Löschen von Gruppen

Nur Superuser können Gruppen erstellen, ändern oder löschen.

Sie können folgende Aktionen ausführen:

- Verwenden Sie den Befehl CREATE GROUP, um eine neue Gruppe zu erstellen.
- Mit dem Befehl ALTER GROUP können Sie einer bestehenden Gruppe Benutzer hinzufügen oder Benutzer aus der Gruppe entfernen.
- Verwenden Sie den Befehl DROP GROUP, um eine Gruppe zu löschen. Mit diesem Befehl wird nur die Gruppe gelöscht, die Benutzer, die der Gruppe angehören, werden selbst nicht gelöscht.

Beispiel zur Steuerung des Zugriffs durch Benutzer und Gruppen

In diesem Beispiel werden Benutzergruppen und Benutzer erstellt und anschließend werden ihnen verschiedene Berechtigungen für eine Amazon-Redshift-Datenbank erteilt, die sich mit einem Webanwendungsclient verbindet. In diesem Beispiel werden drei Gruppen von Benutzern zugrundegelegt: normale Benutzer der Webanwendung, Hauptbenutzer der Webanwendung und Webentwickler.

1. Es werden die Gruppen erstellt, denen die verschiedenen Benutzer zugewiesen werden. Mit den folgenden Befehlen werden drei verschiedene Benutzergruppen erstellt:

```
create group webappusers;

create group webpowerusers;

create group webdevusers;
```

2. Es werden mehrere Datenbankbenutzer mit unterschiedlichen Berechtigungen erstellt und anschließend den Gruppen hinzugefügt.

   a. Es werden zwei Benutzer erstellt und der Gruppe WEBAPPUSERS hinzugefügt:

   ```
   create user webappuser1 password 'webAppuser1pass'
   in group webappusers;

   create user webappuser2 password 'webAppuser2pass'
   in group webappusers;
   ```

   b. Es wird ein Konto für einen Webentwickler erstellt und der Gruppe WEBDEVUSERS hinzugefügt:

   ```
   create user webdevuser1 password 'webDevuser2pass'
   in group webdevusers;
   ```

   c. Erstellen Sie einen Superuser. Dieser Benutzer hat die Berechtigung, andere Benutzer zu erstellen:

   ```
   create user webappadmin  password 'webAppadminpass1'
   createuser;
   ```

3. Es wird ein Schema erstellt, das mit den von der Webanwendung verwendeten Datenbanktabellen verknüpft wird. Anschließend wird den verschiedenen Benutzergruppen Zugriff auf dieses Schema gewährt:

   a. Das WEBAPP-Schema wird erstellt:

   ```
   create schema webapp;
   ```

   b. Der Gruppe WEBAPPUSERS wird die USAGE-Berechtigung gewährt:

   ```
   grant usage on schema webapp to group webappusers;
   ```

   c. Der Gruppe WEBPOWERUSERS wird die USAGE-Berechtigung gewährt:

```
grant usage on schema webapp to group webpowerusers;
```

d. Der Gruppe WEBDEVUSERS wird die ALL-Berechtigung gewährt:

```
grant all on schema webapp to group webdevusers;
```

Nun sind die benötigten Benutzer und Gruppen eingerichtet. Sie können die Benutzer und Gruppen jetzt ändern.

4. Mit dem folgenden Befehl wird beispielsweise der Parameter „search_path“ für den WEBAPPUSER1 geändert.

```
alter user webappuser1 set search_path to webapp, public;
```

Über SEARCH_PATH wird festgelegt, in welcher Reihenfolge die Schemata nach Datenbankobjekten wie Tabellen und Funktionen durchsucht werden, wenn ein Objekt nur über seinen einfachen Namen (ohne Angabe des Schemas) referenziert wird.

5. Sie können einer Gruppe auch Benutzer hinzufügen, nachdem sie erstellt wurde, beispielsweise WEBAPPUSER2 zur Gruppe WEBPOWERUSERS:

```
alter group webpowerusers add user webappuser2;
```

Schemata

Eine Datenbank enthält einen oder mehrere benannte Schemata. Jedes Schema in einer Datenbank enthält Tabellen und andere Arten von benannten Objekten. Standardmäßig haben alle Datenbanken ein einziges Schema, mit dem Namen PUBLIC. Sie können Schemata verwenden, um Datenbankobjekte unter einem gemeinsamen Namen zusammenzufassen. Schemata sind mit Verzeichnissen eines Dateisystems vergleichbar, sie ermöglichen jedoch keine Verschachtelung.

Wenn Sie in einer Datenbank mehrere Schemata haben, können in den verschiedenen Schemas ansonsten identische Datenbankenobjekte mit demselben Namen verwendet werden, ohne dass dies zu einem Namenskonflikt führt. Es können beispielsweise das Schema MY_SCHEMA und das Schema YOUR_SCHEMA beide eine Tabelle mit dem Namen MYTABLE enthalten. Wenn Benutzer über geeignete Berechtigungen verfügen, können sie auf Objekte in verschiedenen Schemas in der Datenbank zugreifen.

Standardmäßig werden neu erstellte Objekte in dem ersten Schema im Suchpfad der Datenbank erstellt. Weitere Informationen finden Sie unter Suchpfad weiter oben in diesem Abschnitt.

Schemata können wie folgt dazu beitragen, in einer Mehrfachbenutzerumgebung Probleme in Zusammenhang mit der Strukturierung und der Ausführung paralleler Anweisungen zu vermeiden:

- Schemata ermöglichen eine Zusammenarbeit von Entwicklern an derselben Datenbank, ohne dass Namenskonflikte auftreten.
- Sie ermöglichen die Zusammenfassung von Datenbankobjekten zu logischen Einheiten, die besser verwaltet werden können.
- Sie ermöglichen es Anwendungen, ihre Objekte in separaten Schemata abzulegen, ohne dass deren Namen zu einem Konflikt mit Objekten von anderen Anwendungen führen.

Erstellen, Modifizieren und Löschen von Schemata

Alle Benutzer können Schemata erstellen und eigene Schemata ändern und löschen.

Sie können folgende Aktionen ausführen:

- Verwenden Sie den Befehl CREATE SCHEMA, um ein neues Schema zu erstellen.
- Mit dem Befehl ALTER SCHEMA können Sie den Besitzer eines Schemas ändern.
- Um ein Schema und alle enthaltenen Objekte zu löschen, führen Sie den Befehl DROP SCHEMA aus.
- Um eine Tabelle in einem Schema zu erstellen, verwenden Sie bei der Erstellung der Tabelle die Syntax schema_name.tabellen_name.

Führen Sie eine Abfrage wie folgt über der Katalogtabelle PG_NAMESPACE aus, um eine Liste aller Schemata anzuzeigen:

```
select * from pg_namespace;
```

Führen Sie eine Abfrage über der Katalogtabelle PG_TABLE_DEF aus, um eine Liste aller Tabellen in einem Schema anzuzeigen. Beispiel: Die folgende Abfrage gibt eine Liste der Tabellen im Schema PG_CATALOG zurück.

```
select distinct(tablename) from pg_table_def
```

```
where schemaname = 'pg_catalog';
```

Suchpfad

Der Suchpfad wird im Parameter „search_path“ als kommagetrennte Liste von Schemanamen definiert. Über den Suchpfad wird festgelegt, in welcher Reihenfolge die Schemas durchsucht werden, wenn ein Objekt, beispielsweise eine Tabelle oder eine Funktion, mit dem einfachen Namen (also ohne Kennzeichnung des Schemas) referenziert wird.

Wenn ein Objekt ohne Angabe eines Zielschemas erstellt wird, wird das Objekt in dem ersten Schema in dem Suchpfad angelegt. Wenn in verschiedenen Schemata Objekte mit demselben Namen vorhanden sind, werden Objektaufrufe ohne Schemaangabe als Verweis auf das erste Schema in dem Suchpfad interpretiert, das ein Objekt mit dem angegebenen Namen enthält.

Um das Standardschema für die aktuelle Sitzung zu ändern, führen Sie den Befehl SET aus.

Weitere Informationen finden Sie in der search_path-Beschreibung in der Konfigurationsreferenz.

Schemabasierte Berechtigungen

Schemabasierte Berechtigungen werden vom Besitzer des Schemas festgelegt:

- Standardmäßig besitzen alle Benutzer CREATE- und USAGE-Berechtigungen für das Schema PUBLIC einer Datenbank. Um Benutzer daran zu hindern, Objekte im Schema PUBLIC einer Datenbank zu erstellen, verwenden Sie den Befehl REVOKE, um diese Berechtigung zu entfernen.
- Benutzer können auf Objekte in Schemas, die sie nicht besitzen, nur dann zugreifen, wenn der Besitzer des jeweiligen Objekts ihnen die USAGE-Berechtigung erteilt hat.
- Wenn Benutzern die CREATE-Berechtigung für ein Schema erteilt wird, das von einem anderen Benutzer erstellt wurde, sind diese Benutzer berechtigt, Objekte in diesem Schema zu erstellen.

Rollenbasierte Zugriffskontrolle (RBAC)

Durch die Verwendung der rollenbasierten Zugriffssteuerung (RBAC) zum Verwalten von Datenbankberechtigungen in Amazon Redshift können Sie die Verwaltung von Sicherheitsberechtigungen in Amazon Redshift vereinfachen. Sie können den Zugriff auf sensible Daten sichern, indem Sie steuern, welche Aktionen Benutzer auf einer breit gefassten oder differenzierten Ebene ausführen können. Sie können auch den Benutzerzugriff auf Aufgaben steuern, die normalerweise auf Superuser beschränkt sind. Indem Sie verschiedenen Rollen unterschiedliche

Berechtigungen zuweisen und sie verschiedenen Benutzern zuweisen, haben Sie eine genauere Kontrolle über den Benutzerzugriff.

Benutzer mit einer zugewiesenen Rolle können nur die Aufgaben ausführen, für die sie durch die ihnen zugewiesene Rolle autorisiert sind. Beispielsweise ist ein Benutzer, dem die Rolle mit den Berechtigungen CREATE TABLE und DROP TABLE zugewiesen ist, nur berechtigt, diese Aufgaben auszuführen. Sie können den Zugriff von Benutzern steuern, indem Sie verschiedenen Benutzern unterschiedliche Sicherheitsberechtigungen für den Zugriff auf Daten erteilen, die sie für ihre Arbeit benötigen.

RBAC wendet das Prinzip der geringsten Berechtigungen auf Benutzer basierend auf ihren Rollenanforderungen an, unabhängig von den beteiligten Objekttypen. Das Erteilen und Widerrufen von Berechtigungen erfolgt auf Rollenebene, ohne dass Berechtigungen für einzelne Datenbankobjekte aktualisiert werden müssen.

Mit RBAC können Sie Rollen mit Berechtigungen zum Ausführen von Befehlen erstellen, für die früher Superuser-Berechtigungen erforderlich waren. Benutzer können diese Befehle ausführen, sofern ihnen die Rolle zugewiesen wurde, die diese Berechtigungen enthält. Auf ähnliche Weise können Sie auch Rollen erstellen, um den Zugriff auf bestimmte Befehle zu beschränken, und die Rolle entweder Superusern oder Benutzern zuzuweisen, die mit der Rolle entsprechend autorisiert wurden.

Weitere Informationen zur Funktionsweise von Amazon Redshift RBAC finden Sie im folgenden Video: Introducing Role-based access control (RBAC) in Amazon Redshift.

Rollenhierarchie

Rollen sind Sammlungen von Berechtigungen, die Sie einem Benutzer oder einer anderen Rolle zuweisen können. Sie können einer Rolle System- oder Datenbankberechtigungen zuweisen. Ein Benutzer erbt Berechtigungen von einer ihm zugewiesenen Rolle.

In RBAC können Benutzer verschachtelte Rollen haben. Sie können Rollen sowohl Benutzern als auch Rollen zuweisen. Wenn Sie einem Benutzer eine Rolle zuweisen, erteilen Sie dem Benutzer damit alle Berechtigungen, die diese Rolle beinhaltet. Wenn Sie einem Benutzer eine Rolle r1 gewähren, autorisieren Sie den Benutzer mit den Berechtigungen von r1. Der Benutzer hat jetzt Berechtigungen von r1 und auch alle vorhandenen Berechtigungen, die er bereits besitzt.

Wenn Sie eine Rolle (r1) einer anderen Rolle (r2) zuweisen, erteilen Sie r2 alle Berechtigungen von r1. Wenn Sie r2 für eine andere Rolle (r3) gewähren, sind die Berechtigungen von r3 die Vereinigung

der Berechtigungen von r1 und r2. In der Rollenhierarchie erbt r2 die Berechtigungen von r1. Amazon Redshift propagiert Berechtigungen mit jeder Rollenautorisierung. Die Gewährung von r1 bis r2 und dann r2 bis r3 autorisiert r3 mit allen Berechtigungen der drei Rollen. Durch die Gewährung von r3 an einen Benutzer verfügt der Benutzer über alle Berechtigungen aus den drei Rollen.

Amazon Redshift lässt die Erstellung eines Rollenautorisierungszyklus nicht zu. Ein Rollenautorisierungszyklus findet statt, wenn eine verschachtelte Rolle wieder einer vorherigen Rolle in der Rollenhierarchie zugewiesen wird, z. B. wenn r3 wieder r1 zugewiesen wird. Weitere Informationen zum Erstellen von Rollen und Verwalten von Rollenzuweisungen finden Sie unter Verwalten von Rollen in RBAC.

Rollenzuweisung

Superuser und normale Benutzer mit der CREATE ROLE-Berechtigung können die CREATE ROLE-Anweisung verwenden, um Rollen zu erstellen. Superuser und Rollenadministratoren können die GRANT ROLE-Anweisung verwenden, um anderen eine Rolle zu gewähren. Sie können die REVOKE ROLE-Anweisung verwenden, um eine Rolle von anderen zu widerrufen, und die DROP ROLE-Anweisung, um Rollen zu löschen. Zu den Rollenadministratoren gehören Rollenbesitzer und Benutzer, denen die Rolle mit der Berechtigung ADMIN OPTION erteilt wurde.

Nur Superuser oder Rollenadministratoren können Rollen gewähren und widerrufen. Sie können eine oder mehrere Rollen einer oder mehreren Rollen oder Benutzern zuweisen oder die Zuweisung widerrufen. Verwenden Sie die Option WITH ADMIN OPTION in der GRANT ROLE-Anweisung, um allen Berechtigungsempfänger die Verwaltungsoptionen für alle gewährten Rollen bereitzustellen.

Amazon Redshift unterstützt verschiedene Kombinationen von Rollenzuweisungen, z. B. die Gewährung mehrerer Rollen oder das Zuweisen von Rollen zu mehreren Berechtigungsempfängern. WITH ADMIN OPTION gilt nur für Benutzer und nicht für Rollen. Verwenden Sie auf ähnliche Weise die Option WITH ADMIN OPTION in der REVOKE ROLE-Anweisung, um die Rolle und die Verwaltungsautorisierung des Berechtigungsempfängers zu widerrufen. Bei Verwendung mit der ADMIN OPTION wird nur die Verwaltungsautorisierung von der Rolle widerrufen.

Das folgende Beispiel widerruft die Verwaltungsautorisierung der Rolle `sample_role2` von `user2`.

```
REVOKE ADMIN OPTION FOR sample_role2 FROM user2;
```

Weitere Informationen zum Erstellen von Rollen und Verwalten von Rollenzuweisungen finden Sie unter Verwalten von Rollen in RBAC.

Systemdefinierte Amazon-Redshift-Rollen

Amazon Redshift bietet einige systemdefinierte Rollen, die mit bestimmten Berechtigungen definiert sind. Systemspezifische Rollen beginnen mit einem `sys:`-Präfix. Nur Benutzer mit entsprechendem Zugriff können systemdefinierte Rollen ändern oder benutzerdefinierte systemdefinierte Rollen erstellen. Sie können das `sys:`-Präfix nicht für eine benutzerdefinierte systemdefinierte Rolle verwenden.

In der folgenden Tabelle finden Sie eine Zusammenfassung der Rollen und deren Berechtigungen.

Rollenname	Beschreibung		
`sys:monitor`	Diese Rolle hat die Berechtigung, auf Katalog- oder Systemtabellen zuzugreifen.		
`sys:operator`	Diese Rolle hat die Berechtigung, auf Katalog- oder Systemtabellen zuzugreifen, Abfragen zu analysieren, zu bereinigen oder abzubrechen.		
`sys:dba`	Diese Rolle hat die Berechtigungen zum Erstellen von Schemas, zum Erstellen von Tabellen, zum Löschen von Schemas, zum Löschen von Tabellen und zum Kürzen von Tabellen. Sie besitzt die Berechtigungen, gespeicherte Verfahren zu erstellen oder zu ersetzen, Verfahren zu löschen, Funktionen zu erstellen oder zu ersetzen, externe Funktionen zu erstellen oder zu ersetzen, Ansichten zu erstellen und Ansichten zu löschen. Diese Rolle erbt auch alle		

Rollenname	Beschreibung		
	Berechtigungen von der sys:operator-Rolle.		
`sys:superuser`	Diese Rolle verfügt über alle unterstützten Systemberechtigungen, die in Systemberechtigungen für RBAC definiert sind.		
`sys:secadmin`	• Diese Rolle hat die Berechtigung, Benutzer zu erstellen, Benutzer zu ändern, Benutzer zu löschen, Rollen zu erstellen, Rollen zu löschen und Rollen zu gewähren. • Diese Rolle verfügt über Berechtigungen zum Ein- oder Ausschalten von RLS für eine Relation und über Berechtigungen zur Verwaltung von RLS- und DDM-Richtlinien (CREATE, DROP, ATTACH, DETACH und ALTER). Beachten Sie außerdem, dass dieser Rolle standardmäßig die Berechtigungen EXPLAIN RLS, IGNORE RLS und EXPLAIN MASKING gewährt werden. • Diese Rolle kann nur dann auf Benutzertabellen zugreifen, wenn die Berechtigung explizit für die Rolle erteilt wird.		

Systemdefinierte Rollen und Benutzer für die gemeinsame Nutzung von Daten

Amazon Redshift erstellt Rollen und Benutzer für den internen Gebrauch, die Datashares und Datashare-Verbrauchern entsprechen. Jeder interne Rollenname und Benutzername hat das reservierte Namespace-Präfix. `ds:` Sie haben das folgende Format:

Name	Beschreibung		
`ds:sharei`	Eine Systemrolle, die einer Datenfreigabe entspricht.		
`ds:sharei_consumei`	Ein Systembenutzer, der einem Datashare-Verbraucher entspricht.		

Für jeden Datashare wird eine Rolle für die gemeinsame Nutzung von Daten erstellt. Sie enthält alle Berechtigungen, die dem Datashare derzeit gewährt werden. Für jeden Nutzer eines Datashare wird ein Benutzer für die gemeinsame Nutzung von Daten erstellt. Ihm wird die Berechtigung für eine einzelne Rolle für die gemeinsame Nutzung von Daten erteilt. Für einen Nutzer, der mehreren Datenfreigaben hinzugefügt wird, wird für jeden Datenaustausch ein Benutzer für die gemeinsame Nutzung von Daten erstellt.

Diese Benutzer und Rollen sind erforderlich, damit die gemeinsame Nutzung von Daten ordnungsgemäß funktioniert. Sie können nicht geändert oder gelöscht werden und sie können nicht aufgerufen oder für Aufgaben verwendet werden, die von Kunden ausgeführt werden. Sie können sie getrost ignorieren. Weitere Informationen zur gemeinsamen Nutzung von Daten finden Sie unter Clusterübergreifende gemeinsame Nutzung von Daten in Amazon Redshift.

> **Note**
>
> Sie können das `ds:` Präfix nicht verwenden, um benutzerdefinierte Rollen oder Benutzer zu erstellen.

Systemberechtigungen für RBAC

Im Folgenden finden Sie eine Liste der Systemberechtigungen, die Sie einer Rolle gewähren oder entziehen können.

Befehl	Sie benötigen eine der folgenden Berechtigungen, um den Befehl ausführen zu können
CREATE ROLE	• Superuser. • Benutzer mit der Berechtigung CREATE ROLE.
DROP ROLE	• Superuser. • Rollenbesitzer, der entweder der Benutzer ist, der die Rolle erstellt hat, oder ein Benutzer, dem die Rolle mit dem Privileg WITH ADMIN OPTION gewährt wurde.
CREATE USER	• Superuser. • Benutzer mit der Berechtigung CREATE USER. Diese Benutzer können keine Superuser erstellen.
DROP USER	• Superuser. • Benutzer mit der Berechtigung DROP USER.
ALTER USER	• Superuser. • Benutzer mit der Berechtigung ALTER USER. Diese Benutzer können Benutzer nicht in Superuser ändern oder Superuser in Benutzer ändern. • Aktueller Benutzer, der sein eigenes Passwort ändern möchte.
CREATE SCHEMA	• Superuser. • Benutzer mit der Berechtigung CREATE SCHEMA.
DROP SCHEMA	• Superuser. • Benutzer mit der Berechtigung DROP SCHEMA. • Schemabesitzer.
ALTER DEFAULT PRIVILEGES	• Superuser. • Benutzer mit der Berechtigung ALTER DEFAULT PRIVILEGES. • Benutzer, die ihre eigenen Standardzugriffsberechtigungen ändern.

Befehl	Sie benötigen eine der folgenden Berechtigungen, um den Befehl ausführen zu können		
	• Benutzer legen Berechtigungen für Schemas fest, auf die sie Zugriffsberechtigungen haben.		
CREATE TABLE	• Superuser. • Benutzer mit der Berechtigung CREATE TABLE. • Benutzer mit der CREATE-Berechtigung für Schemas.		
DROP TABLE	• Superuser. • Benutzer mit der Berechtigung DROP TABLE. • Tabellenbesitzer mit der Berechtigung USAGE für das Schema.		
ALTER TABLE	• Superuser. • Benutzer mit der Berechtigung ALTER TABLE. • Tabellenbesitzer mit der Berechtigung USAGE für das Schema.		

CREATE OR REPLAC FUNCTIC	• Für CREATE FUNCTION:
	  • Superuser.
	  • Benutzer mit der Berechtigung CREATE OR REPLACE FUNCTION.
	  • Benutzer mit der Berechtigung USAGE für die Sprache.
	• Für REPLACE FUNCTION:
	  • Superuser.
	  • Benutzer mit der Berechtigung CREATE OR REPLACE FUNCTION.
	  • Funktionsbesitzer.		

CREATE OR REPLAC EXTERN FUNCTIC	• Superuser. • Benutzer mit der Berechtigung CREATE OR REPLACE EXTERNAL FUNCTION.		

Befehl	Sie benötigen eine der folgenden Berechtigungen, um den Befehl ausführen zu können		
DROP FUNCTIO	• Superuser. • Benutzer mit der Berechtigung DROP FUNCTION. • Funktionsbesitzer.		

CREATE OR REPLAC PROCED	• Für CREATE PROCEDURE:
	  • Superuser.
	  • Benutzer mit der Berechtigung CREATE OR REPLACE PROCEDURE.
	  • Benutzer mit der Berechtigung USAGE für die Sprache.
	• Für REPLACE PROCEDURE:
	  • Superuser.
	  • Benutzer mit der Berechtigung CREATE OR REPLACE PROCEDURE.
	  • Besitzer des Verfahrens.		

DROP PROCED	• Superuser. • Benutzer mit der Berechtigung DROP PROCEDURE. • Besitzer des Verfahrens.		

CREATE OR REPLAC VIEW	• Für CREATE VIEW:
	  • Superuser.
	  • Benutzer mit der Berechtigung CREATE OR REPLACE VIEW.
	  • Benutzer mit der CREATE-Berechtigung für Schemas.
	• Für REPLACE VIEW:
	  • Superuser.
	  • Benutzer mit der Berechtigung CREATE OR REPLACE VIEW.
	  • Besitzer der Ansicht.		

Befehl	Sie benötigen eine der folgenden Berechtigungen, um den Befehl ausführen zu können		
DROP VIEW	• Superuser. • Benutzer mit der Berechtigung DROP VIEW. • Besitzer der Ansicht.		
CREATE MODEL	• Superuser. • Benutzer mit der CREATE MODEL-Systemberechtigung, die in der Lage sein sollten, die Beziehung von CREATE MODEL zu lesen. • Benutzer mit der Berechtigung CREATE MODEL.		
DROP MODEL	• Superuser. • Benutzer mit der Berechtigung DROP MODEL. • Modellbesitzer. • Schemabesitzer.		
CREATE DATASH	• Superuser. • Benutzer mit der Berechtigung CREATE DATASHARE. • Datenbankbesitzer.		
ALTER DATASH	• Superuser. • Benutzer mit der Berechtigung ALTER DATASHARE. • Benutzer mit Berechtigung ALTER oder ALL für das Datashare. • Um bestimmte Objekte zu einem Datashare hinzuzufügen, müssen diese Benutzer die Berechtigung für die Objekte haben. Benutzer sollten auch Besitzer der Objekte sein oder die Berechtigungen SELECT, USAGE oder ALL für die Objekte haben.		
DROP DATASH	• Superuser. • Benutzer mit der Berechtigung DROP DATASHARE. • Datenbankbesitzer.		

Befehl	Sie benötigen eine der folgenden Berechtigungen, um den Befehl ausführen zu können		
CREATE LIBRARY	• Superuser. • Benutzer mit der Berechtigung CREATE LIBRARY oder mit der Berechtigung für die angegebene Sprache.		
DROP LIBRARY	• Superuser. • Benutzer mit der Berechtigung DROP LIBRARY. • Der Bibliotheksbesitzer.		
ANALYZE	• Superuser. • Benutzer mit der Berechtigung ANALYZE. • Besitzer der Beziehung. • Datenbankbesitzer, für den die Tabelle freigegeben wird.		
CANCEL	• Superuser bricht seine eigene Abfrage ab. • Superuser bricht die Abfrage eines Benutzers ab. • Benutzer mit der Berechtigung CANCEL, die die Abfrage eines Benutzers abbrechen. • Der Benutzer bricht seine eigene Abfrage ab.		
TRUNCATE TABLE	• Superuser. • Benutzer mit der Berechtigung TRUNCATE TABLE. • Tabellenbesitzer.		
VACUUM	• Superuser. • Benutzer mit der Berechtigung VACUUM. • Tabellenbesitzer. • Datenbankbesitzer, für den die Tabelle freigegeben wird.		
IGNORE RLS	• Superuser. • Benutzer innerhalb der `sys:secadmin` -Rolle.		

Befehl	Sie benötigen eine der folgenden Berechtigungen, um den Befehl ausführen zu können
EXPLAIN RLS	• Superuser. • Benutzer innerhalb der `sys:secadmin` -Rolle.
EXPLAIN MASKING	• Superuser. • Benutzer innerhalb der `sys:secadmin` -Rolle.

Datenbankobjektberechtigungen

Neben Systemberechtigungen enthält Amazon Redshift Datenbankobjektberechtigungen, die Zugriffsoptionen definieren. Hierzu gehören Optionen wie das Lesen von Daten in Tabellen und Ansichten, das Schreiben von Daten und das Erstellen und Entfernen von Tabellen. Weitere Informationen finden Sie unter GRANT-Befehl.

Mithilfe von RBAC können Sie Rollen Datenbankobjektberechtigungen zuweisen, ähnlich wie das bei Systemberechtigungen möglich ist. Anschließend können Sie Benutzern Rollen zuweisen, Benutzer mit Systemberechtigungen autorisieren und Benutzer mit Datenbankberechtigungen autorisieren.

ALTER DEFAULT PRIVILEGES für RBAC

Sie können die Anweisung ALTER DEFAULT PRIVILEGES verwenden, um den Standardsatz von Zugriffrechten zu definieren, die auf Objekte angewendet werden sollen, die vom angegebenen Benutzer in der Zukunft erstellt werden. Standardmäßig können Benutzer nur ihre eigenen Standardzugriffsrechte ändern. Mit RBAC können Sie die Standardzugriffsberechtigungen für Rollen festlegen. Weitere Informationen finden Sie unter ALTER DEFAULT PRIVILEGES-Befehl.

Mithilfe von RBAC können Sie Rollen Datenbankobjektberechtigungen zuweisen, ähnlich wie das bei Systemberechtigungen möglich ist. Anschließend können Sie Benutzern Rollen zuweisen und Benutzer mit Systemberechtigungen und/oder Datenbankberechtigungen autorisieren.

Überlegungen zur Rollennutzung in RBAC

Bei der Arbeit mit RBAC-Rollen sollten Sie Folgendes berücksichtigen:

- Amazon Redshift lässt keine Zyklen von Rollenberechtigungen zu. Sie können nicht r1 für r2 gewähren und dann r2 für r1 zuweisen.

- RBAC funktioniert sowohl für native Amazon-Redshift-Objekte als auch für Amazon Redshift Spectrum-Tabellen.
- Als Amazon-Redshift-Administrator können Sie RBAC aktivieren, indem Sie Ihren Cluster auf den neuesten Wartungspatch aktualisieren, um loszulegen.
- Nur Superuser und Benutzer mit der CREATE ROLE-Systemberechtigung können Rollen erstellen.
- Nur Superuser und Rollenadministratoren können Rollen ändern oder löschen.
- Ein Rollenname darf nicht mit einem Benutzernamen identisch sein.
- Ein Rollenname darf keine unzulässigen Zeichen wie „:/\ n“ enthalten.
- Ein Rollenname darf kein reserviertes Wort sein, beispielsweise PUBLIC.
- Der Rollenname kann nicht mit dem reservierten Präfix für Standardrollen beginnen: `sys:`.
- Sie können eine Rolle mit dem RESTRICT-Parameter nicht löschen, wenn sie einer anderen Rolle zugewiesen wird. Die Standardeinstellung ist RESTRICT. Amazon Redshift löst einen Fehler aus, wenn Sie versuchen, eine Rolle zu löschen, die eine andere Rolle geerbt hat.
- Benutzer, die keine Administratorberechtigungen für eine Rolle haben, können eine Rolle nicht zuweisen oder widerrufen.

Verwalten von Rollen in RBAC

Verwenden Sie die folgenden Befehle, um die unten aufgeführten Aktionen auszuführen:

- Um eine Rolle zu erstellen, verwenden Sie den Befehl CREATE ROLE.
- Um eine Rolle umzubenennen oder den Besitzer der Rolle zu ändern, verwenden Sie den Befehl ALTER ROLE.
- Verwenden Sie zum Löschen einer Rolle den Befehl DROP ROLE.
- Um einem Benutzer eine Rolle zuzuweisen, verwenden Sie den Befehl GRANT.
- Um eine Rolle von einem Benutzer zu widerrufen, verwenden Sie den Befehl REVOKE.
- Um Systemberechtigungen für eine Rolle zu erteilen, verwenden Sie den Befehl GRANT.
- Um Systemberechtigungen für eine Rolle zu entziehen, verwenden Sie den Befehl REVOKE.

Eine Liste der Rollen in Ihrem Cluster oder Ihrer Arbeitsgruppe finden Sie unter SVV_ROLES.

Tutorial: Rollen erstellen und Abfragen mit RBAC durchführen

Mit RBAC können Sie Rollen mit Berechtigungen zum Ausführen von Befehlen erstellen, für die früher Superuser-Berechtigungen erforderlich waren. Benutzer können diese Befehle ausführen, sofern ihnen die Rolle zugewiesen wurde, die diese Berechtigungen enthält.

In diesem Tutorial verwenden Sie die rollenbasierte Zugriffskontrolle (RBAC), um Berechtigungen in einer von Ihnen erstellten Datenbank zu verwalten. Anschließend stellen Sie eine Verbindung mit der Datenbank her und fragen die Datenbank von zwei verschiedenen Rollen aus ab, um die Funktionalität von RBAC zu testen.

Die beiden Rollen, die Sie erstellen und verwenden, um die Datenbank abzufragen, sind und. `sales_ro` `sales_rw` Sie erstellen die `sales_ro` Rolle und fragen Daten als Benutzer mit der `sales_ro` Rolle ab. Der `sales_ro` Benutzer kann nur den Befehl SELECT verwenden, nicht jedoch den Befehl UPDATE. Anschließend erstellen Sie die `sales_rw` Rolle und fragen Daten als Benutzer mit der `sales_rw` Rolle ab. Der `sales_rw` Benutzer kann den Befehl SELECT und den Befehl UPDATE verwenden.

Darüber hinaus können Sie Rollen erstellen, um den Zugriff auf bestimmte Befehle einzuschränken, und die Rolle entweder Superusern oder Benutzern zuweisen.

Aufgaben

- Voraussetzungen
- Schritt 1: Erstellen Sie einen Administratorbenutzer
- Schritt 2: Schemas einrichten
- Schritt 3: Erstellen Sie einen schreibgeschützten Benutzer
- Schritt 4: Fragen Sie die Daten als schreibgeschützter Benutzer ab
- Schritt 5: Erstellen Sie einen Benutzer mit Lese-/Schreibzugriff
- Schritt 6: Fragen Sie die Daten als Benutzer mit der geerbten Rolle „Nur Lesen“ ab
- Schritt 7: Erteilen Sie der Lese- und Schreibrolle Aktualisierungs- und Einfügeberechtigungen
- Schritt 8: Fragen Sie die Daten als Benutzer mit Lese-/Schreibzugriff ab
- Schritt 9: Analysieren und Vakuumieren von Tabellen in einer Datenbank als Administratorbenutzer
- Schritt 10: Kürzen Sie Tabellen als Benutzer mit Lese-/Schreibzugriff

Voraussetzungen

- Erstellen Sie einen Amazon Redshift Redshift-Cluster oder eine serverlose Arbeitsgruppe, die mit der TICKIT-Beispieldatenbank geladen ist. Informationen zum Erstellen einer serverlosen Arbeitsgruppe finden Sie unter Amazon Redshift Serverless. Informationen zum Erstellen eines Clusters finden Sie unter Erstellen eines Amazon Redshift Redshift-Beispielclusters. Weitere Informationen zur TICKIT-Beispieldatenbank finden Sie unter Beispieldatenbank.
- Haben Sie Zugriff auf einen Benutzer mit Superuser- oder Rollenadministratorberechtigungen. Nur Superuser oder Rollenadministratoren können Rollen gewähren oder entziehen. Weitere Informationen zu den für RBAC erforderlichen Berechtigungen finden Sie unter. Systemberechtigungen für RBAC
- Überprüfen Sie das Überlegungen zur Rollennutzung in RBAC.

Schritt 1: Erstellen Sie einen Administratorbenutzer

Um dieses Tutorial einzurichten, erstellen Sie eine Datenbankadministratorrolle und weisen sie in diesem Schritt einem Datenbankadministratorbenutzer zu. Sie müssen den Datenbankadministrator als Superuser oder Rollenadministrator erstellen.

Führen Sie alle Abfragen in Amazon Redshift https://docs.aws.amazon.com/redshift/latest/mgmt/query-editor-v2-using.html aus.

1. Verwenden Sie das folgende Beispiel, um die Administratorrolle db_admin zu erstellen.

```
CREATE ROLE db_admin;
```

2. Verwenden Sie das folgende Beispiel, um einen Datenbankbenutzer mit dem Namen dbadmin zu erstellen.

```
CREATE USER dbadmin PASSWORD 'Test12345';
```

3. Verwenden Sie das folgende Beispiel, um der Rolle db_admin die systemdefinierte Rolle mit dem Namen sys:dba zuzuweisen. Wenn dem Benutzer dbadmin die Rolle sys:dba zugewiesen wird, kann er Schemas und Tabellen erstellen. Weitere Informationen finden Sie unter Systemdefinierte Amazon-Redshift-Rollen.

Schritt 2: Schemas einrichten

In diesem Schritt stellen Sie als Datenbankadministrator eine Verbindung zu Ihrer Datenbank her. Anschließend erstellen Sie zwei Schemas und fügen ihnen Daten hinzu.

1. Stellen Sie mit dem Abfrage-Editor v2 als dbadmin-Benutzer eine Verbindung zur Dev-Datenbank her. Weitere Hinweise zum Herstellen einer Verbindung zu einer Datenbank finden Sie unter Arbeiten mit dem Abfrage-Editor v2.
2. Verwenden Sie das folgende Beispiel, um die Datenbankschemas für Vertrieb und Marketing zu erstellen.

```
CREATE SCHEMA sales;
CREATE SCHEMA marketing;
```

3. Verwenden Sie das folgende Beispiel, um Werte zu erstellen und in Tabellen im Vertriebsschema einzufügen.

```
CREATE TABLE sales.cat(
catid smallint,
catgroup varchar(10),
catname varchar(10),
catdesc varchar(50)
);
INSERT INTO sales.cat(SELECT * FROM category);

CREATE TABLE sales.dates(
dateid smallint,
caldate date,
day char(3),
week smallint,
month char(5),
qtr char(5),
year smallint,
holiday boolean
);
INSERT INTO sales.dates(SELECT * FROM date);

CREATE TABLE sales.events(
eventid integer,
venueid smallint,
catid smallint,
dateid smallint,
```

```
eventname varchar(200),
starttime timestamp
);
INSERT INTO sales.events(SELECT * FROM event);

 CREATE TABLE sales.sale(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp
);
INSERT INTO sales.sale(SELECT * FROM sales);
```

4. Verwenden Sie das folgende Beispiel, um Werte zu erstellen und in Tabellen im Marketingschema einzufügen.

```
CREATE TABLE marketing.cat(
catid smallint,
catgroup varchar(10),
catname varchar(10),
catdesc varchar(50)
);
INSERT INTO marketing.cat(SELECT * FROM category);

CREATE TABLE marketing.dates(
dateid smallint,
caldate date,
day char(3),
week smallint,
month char(5),
qtr char(5),
year smallint,
holiday boolean
);
INSERT INTO marketing.dates(SELECT * FROM date);

CREATE TABLE marketing.events(
```

```
eventid integer,
venueid smallint,
catid smallint,
dateid smallint,
eventname varchar(200),
starttime timestamp
);
INSERT INTO marketing.events(SELECT * FROM event);

CREATE TABLE marketing.sale(
marketingid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp
);
INSERT INTO marketing.sale(SELECT * FROM marketing);
```

Schritt 3: Erstellen Sie einen schreibgeschützten Benutzer

In diesem Schritt erstellen Sie eine schreibgeschützte Rolle und einen Salesanalyst-Benutzer für die schreibgeschützte Rolle. Der Vertriebsanalyst benötigt nur Lesezugriff auf die Tabellen im Vertriebsschema, um die ihm zugewiesene Aufgabe zu erfüllen, nämlich die Ereignisse zu finden, die zu den höchsten Provisionen geführt haben.

1. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um die Rolle sales_ro zu erstellen.

```
CREATE ROLE sales_ro;
```

3. Verwenden Sie das folgende Beispiel, um den Benutzer salesanalyst zu erstellen.

```
CREATE USER salesanalyst PASSWORD 'Test12345';
```

4. Verwenden Sie das folgende Beispiel, um der Rolle sales_ro die Verwendung und den Auswahlzugriff auf Objekte des Vertriebsschemas zu gewähren.

```
GRANT USAGE ON SCHEMA sales TO ROLE sales_ro;
GRANT SELECT ON ALL TABLES IN SCHEMA sales TO ROLE sales_ro;
```

5. Verwenden Sie das folgende Beispiel, um dem Benutzer salesanalyst die Rolle sales_ro zuzuweisen.

```
GRANT ROLE sales_ro TO salesanalyst;
```

Schritt 4: Fragen Sie die Daten als schreibgeschützter Benutzer ab

In diesem Schritt fragt der Salesanalyst-Benutzer Daten aus dem Vertriebsschema ab. Anschließend versucht der Salesanalyst-Benutzer, eine Tabelle zu aktualisieren und Tabellen im Marketingschema zu lesen.

1. Stellen Sie als salesanalyst-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um die 10 Verkäufe mit den höchsten Provisionen zu finden.

```
SET SEARCH_PATH TO sales;
SELECT DISTINCT events.dateid, sale.commission, cat.catname
FROM sale, events, dates, cat
WHERE events.dateid=dates.dateid AND events.dateid=sale.dateid AND events.catid =
 cat.catid
ORDER BY 2 DESC LIMIT 10;

+--------+------------+----------+
| dateid | commission | catname  |
+--------+------------+----------+
|   1880 |     1893.6 | Pop      |
|   1880 |     1893.6 | Opera    |
|   1880 |     1893.6 | Plays    |
|   1880 |     1893.6 | Musicals |
|   1861 |       1500 | Plays    |
|   2003 |       1500 | Pop      |
|   1861 |       1500 | Opera    |
|   2003 |       1500 | Plays    |
|   1861 |       1500 | Musicals |
|   1861 |       1500 | Pop      |
+--------+------------+----------+
```

3. Verwenden Sie das folgende Beispiel, um 10 Ereignisse aus der Ereignistabelle im Vertriebsschema auszuwählen.

```
SELECT * FROM sales.events LIMIT 10;

+---------+---------+-------+--------+--------------------+---------------------+
| eventid | venueid | catid | dateid |     eventname      |      starttime      |
+---------+---------+-------+--------+--------------------+---------------------+
|    4836 |      73 |     9 |   1871 | Soulfest           | 2008-02-14 19:30:00 |
|    5739 |      41 |     9 |   1871 | Fab Faux           | 2008-02-14 19:30:00 |
|     627 |     229 |     6 |   1872 | High Society       | 2008-02-15 14:00:00 |
|    2563 |     246 |     7 |   1872 | Hamlet             | 2008-02-15 20:00:00 |
|    7703 |      78 |     9 |   1872 | Feist              | 2008-02-15 14:00:00 |
|    7903 |      90 |     9 |   1872 | Little Big Town    | 2008-02-15 19:30:00 |
|    7925 |     101 |     9 |   1872 | Spoon              | 2008-02-15 19:00:00 |
|    8113 |      17 |     9 |   1872 | Santana            | 2008-02-15 15:00:00 |
|     463 |     303 |     8 |   1873 | Tristan und Isolde | 2008-02-16 19:00:00 |
|     613 |     236 |     6 |   1873 | Pal Joey           | 2008-02-16 15:00:00 |
+---------+---------+-------+--------+--------------------+---------------------+
```

4. Führen Sie das folgende Beispiel aus, um zu versuchen, den Eventnamen für Event-ID 1 zu aktualisieren. Dieses Beispiel führt zu dem Fehler „Zugriff verweigert“, da der Benutzer salesanalyst nur über SELECT-Berechtigungen für die Ereignistabelle im Vertriebsschema verfügt. Um die Ereignistabelle zu aktualisieren, müssen Sie der Rolle sales_ro die Berechtigungen UPDATE gewähren. Weitere Informationen zum Erteilen von Berechtigungen zum Aktualisieren einer Tabelle finden Sie im UPDATE-Parameter für. GRANT Weitere Hinweise zum Befehl UPDATE finden Sie unterUPDATE.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;

ERROR: permission denied for relation events
```

5. Verwenden Sie das folgende Beispiel, um zu versuchen, alle Ereignisse aus der Ereignistabelle im Marketingschema auszuwählen. Dieses Beispiel führt zu dem Fehler „Zugriff verweigert“, da der Benutzer salesanalyst nur über SELECT-Berechtigungen für die Ereignistabelle im Vertriebsschema verfügt. Um Daten aus der Ereignistabelle im Marketingschema auszuwählen, müssen Sie der Rolle sales_ro SELECT-Berechtigungen für die Ereignistabelle im Marketingschema gewähren.

```
SELECT * FROM marketing.events;

               ERROR: permission denied for schema marketing
```

Schritt 5: Erstellen Sie einen Benutzer mit Lese-/Schreibzugriff

In diesem Schritt erhält der Vertriebsingenieur, der für den Aufbau der ETL-Pipeline (Extrahieren, Transformieren und Laden) für die Datenverarbeitung im Vertriebsschema verantwortlich ist, nur Lesezugriff. Später erhält er jedoch Lese- und Schreibzugriff, um seine Aufgaben auszuführen.

1. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um die Rolle sales_rw im Vertriebsschema zu erstellen.

```
CREATE ROLE sales_rw;
```

3. Verwenden Sie das folgende Beispiel, um den Benutzer salesengineer zu erstellen.

```
CREATE USER salesengineer PASSWORD 'Test12345';
```

4. Verwenden Sie das folgende Beispiel, um der Rolle sales_rw die Verwendung zu gewähren und Zugriff auf Objekte des Vertriebsschemas auszuwählen, indem Sie ihr die Rolle sales_ro zuweisen. Weitere Informationen darüber, wie Rollen Berechtigungen in Amazon Redshift erben, finden Sie unter. Rollenhierarchie

```
GRANT ROLE sales_ro TO ROLE sales_rw;
```

5. Verwenden Sie das folgende Beispiel, um dem Benutzer salesengineer die Rolle sales_rw zuzuweisen.

```
GRANT ROLE sales_rw TO salesengineer;
```

Schritt 6: Fragen Sie die Daten als Benutzer mit der geerbten Schreibschutzrolle ab

In diesem Schritt versucht der SalesEngineer-Benutzer, die Ereignistabelle zu aktualisieren, bevor ihm Leseberechtigungen erteilt werden.

1. Stellen Sie als salesengineer-Benutzer eine Connect zur Datenbank her.

2. Der salesengineer-Benutzer kann erfolgreich Daten aus der Ereignistabelle des Vertriebsschemas lesen. Verwenden Sie das folgende Beispiel, um das Ereignis mit der Event-ID 1 aus der Ereignistabelle im Vertriebsschema auszuwählen.

```
SELECT * FROM sales.events where eventid=1;

+---------+---------+-------+--------+-----------------+---------------------+
| eventid | venueid | catid | dateid |    eventname    |      starttime      |
+---------+---------+-------+--------+-----------------+---------------------+
|       1 |     305 |     8 |   1851 | Gotterdammerung | 2008-01-25 14:30:00 |
+---------+---------+-------+--------+-----------------+---------------------+
```

3. Verwenden Sie das folgende Beispiel, um zu versuchen, alle Ereignisse aus der Ereignistabelle im Marketingschema auszuwählen. Der SalesEngineer-Benutzer hat keine Berechtigungen für Tabellen im Marketing-Schema, daher führt diese Abfrage zu dem Fehler „Zugriff verweigert“. Um Daten aus der Ereignistabelle im Marketingschema auszuwählen, müssen Sie der Rolle sales_rw SELECT-Berechtigungen für die Ereignistabelle im Marketingschema gewähren.

```
SELECT * FROM marketing.events;

ERROR: permission denied for schema marketing
```

4. Führen Sie das folgende Beispiel aus, um zu versuchen, den Eventnamen für Event-ID 1 zu aktualisieren. Dieses Beispiel führt zu dem Fehler „Zugriff verweigert“, da der Benutzer salesengineer nur über Auswahlberechtigungen für die Ereignistabelle im Vertriebsschema verfügt. Um die Ereignistabelle zu aktualisieren, müssen Sie der Rolle sales_rw die Berechtigungen UPDATE gewähren.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;

ERROR: permission denied for relation events
```

Schritt 7: Erteilen Sie der Lese- und Schreibrolle Aktualisierungs- und Einfügeberechtigungen

In diesem Schritt erteilen Sie der Rolle sales_rw Aktualisierungs- und Einfügeberechtigungen.

1. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um der Rolle sales_rw die Berechtigungen UPDATE, INSERT und DELETE zu erteilen.

```
GRANT UPDATE, INSERT, ON ALL TABLES IN SCHEMA sales TO role sales_rw;
```

Schritt 8: Fragen Sie die Daten als Benutzer mit Lese-/Schreibzugriff ab

In diesem Schritt aktualisiert der Vertriebsingenieur die Tabelle erfolgreich, nachdem seiner Rolle Einfüge- und Aktualisierungsberechtigungen erteilt wurden. Als Nächstes versucht der Vertriebsingenieur, die Ereignistabelle zu analysieren und zu leeren, was ihm jedoch nicht gelingt.

1. Stellen Sie als salesengineer-Benutzer eine Connect zur Datenbank her.
2. Führen Sie das folgende Beispiel aus, um den Eventnamen für Event-ID 1 zu aktualisieren.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;
```

3. Um die in der vorherigen Abfrage vorgenommene Änderung anzuzeigen, verwenden Sie das folgende Beispiel, um das Ereignis mit der Ereignis-ID 1 aus der Ereignistabelle im Vertriebsschema auszuwählen.

```
SELECT * FROM sales.events WHERE eventid=1;

+---------+---------+-------+--------+--------------+---------------------+
| eventid | venueid | catid | dateid |   eventname  |      starttime      |
+---------+---------+-------+--------+--------------+---------------------+
|       1 |     305 |     8 |   1851 | Comment event | 2008-01-25 14:30:00 |
+---------+---------+-------+--------+--------------+---------------------+
```

4. Verwenden Sie das folgende Beispiel, um die aktualisierte Ereignistabelle im Vertriebsschema zu analysieren. Dieses Beispiel führt zu dem Fehler „Zugriff verweigert“, da der Benutzer salesengineer nicht über die erforderlichen Berechtigungen verfügt und nicht der Besitzer der Ereignistabelle im Vertriebsschema ist. Um die Ereignistabelle zu analysieren, müssen Sie der Rolle sales_rw mithilfe des Befehls GRANT die Berechtigungen für ANALYZE erteilen. Weitere Hinweise zum Befehl ANALYZE finden Sie unter. ANALYZE

```
ANALYZE sales.events;
```

```
ERROR: skipping "events" --- only table or database owner can analyze
```

5. Verwenden Sie das folgende Beispiel, um die aktualisierte Ereignistabelle zu leeren. Dieses Beispiel führt zu dem Fehler „Zugriff verweigert“, da der Benutzer salesengineer nicht über die erforderlichen Berechtigungen verfügt und nicht der Besitzer der Ereignistabelle im Vertriebsschema ist. Um die Ereignistabelle zu leeren, müssen Sie VACUUM mithilfe des Befehls GRANT die Rollenberechtigungen sales_rw gewähren. Weitere Hinweise zum Befehl VACUUM finden Sie unter. VACUUM

```
VACUUM sales.events;

ERROR: skipping "events" --- only table or database owner can vacuum it
```

Schritt 9: Analysieren und Vakuumieren von Tabellen in einer Datenbank als Administratorbenutzer

In diesem Schritt analysiert der dbadmin-Benutzer alle Tabellen und entfernt sie. Der Benutzer hat Administratorrechte für diese Datenbank, sodass er diese Befehle ausführen kann.

1. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um die Ereignistabelle im Vertriebsschema zu analysieren.

```
ANALYZE sales.events;
```

3. Verwenden Sie das folgende Beispiel, um die Ereignistabelle im Vertriebsschema zu leeren.

```
VACUUM sales.events;
```

4. Verwenden Sie das folgende Beispiel, um die Ereignistabelle im Marketingschema zu analysieren.

```
ANALYZE marketing.events;
```

5. Verwenden Sie das folgende Beispiel, um die Ereignistabelle im Marketingschema zu leeren.

```
VACUUM marketing.events;
```

Schritt 10: Kürzen Sie Tabellen als Benutzer mit Lese-/Schreibzugriff

In diesem Schritt versucht der SalesEngineer-Benutzer, die Ereignistabelle im Vertriebsschema zu kürzen, was jedoch nur gelingt, wenn ihm der Benutzer dbadmin die Rechte zum Kürzen erteilt.

1. Stellen Sie als salesengineer-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um zu versuchen, alle Zeilen aus der Ereignistabelle im Vertriebsschema zu löschen. Dieses Beispiel führt zu einem Fehler, da der Benutzer salesengineer nicht über die erforderlichen Berechtigungen verfügt und nicht der Besitzer der Ereignistabelle im Vertriebsschema ist. Um die Ereignistabelle zu kürzen, müssen Sie der Rolle sales_rw mithilfe des Befehls GRANT TRUNCATE Berechtigungen erteilen. Weitere Informationen zur Verwendung des TRUNCATE-Befehls finden Sie unter TRUNCATE.

```
TRUNCATE sales.events;

ERROR: must be owner of relation events
```

3. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.
4. Verwenden Sie das folgende Beispiel, um der Rolle sales_rw die Rechte zum Kürzen von Tabellen zu gewähren.

```
GRANT TRUNCATE TABLE TO role sales_rw;
```

5. Stellen Sie mit dem Abfrage-Editor v2 als salesengineer-Benutzer eine Verbindung zur Datenbank her.
6. Verwenden Sie das folgende Beispiel, um die ersten 10 Ereignisse aus der Ereignistabelle im Vertriebsschema zu lesen.

```
SELECT * FROM sales.events ORDER BY eventid LIMIT 10;

+---------+---------+-------+--------+----------------------------
+---------------------+
| eventid | venueid | catid | dateid |          eventname          |      starttime
    |
+---------+---------+-------+--------+----------------------------
+---------------------+
|       1 |     305 |     8 |   1851 | Comment event               | 2008-01-25
 14:30:00 |
|       2 |     306 |     8 |   2114 | Boris Godunov               | 2008-10-15
 20:00:00 |
```

```
|       3 |     302 |     8 |   1935 | Salome                        | 2008-04-19
 14:30:00 |
|       4 |     309 |     8 |   2090 | La Cenerentola (Cinderella) | 2008-09-21
 14:30:00 |
|       5 |     302 |     8 |   1982 | Il Trovatore                  | 2008-06-05
 19:00:00 |
|       6 |     308 |     8 |   2109 | L Elisir d Amore              | 2008-10-10
 19:30:00 |
|       7 |     309 |     8 |   1891 | Doctor Atomic                 | 2008-03-06
 14:00:00 |
|       8 |     302 |     8 |   1832 | The Magic Flute               | 2008-01-06
 20:00:00 |
|       9 |     308 |     8 |   2087 | The Fly                       | 2008-09-18
 19:30:00 |
|      10 |     305 |     8 |   2079 | Rigoletto                     | 2008-09-10
 15:00:00 |
+---------+---------+-------+--------+-------------------------------
+---------------------+
```

7. Verwenden Sie das folgende Beispiel, um die Ereignistabelle im Vertriebsschema zu kürzen.

```
TRUNCATE sales.events;
```

8. Verwenden Sie das folgende Beispiel, um die Daten aus der aktualisierten Ereignistabelle im Vertriebsschema zu lesen.

```
SELECT * FROM sales.events ORDER BY eventid LIMIT 10;

+---------+---------+-------+--------+-------------------------------
+---------------------+
| eventid | venueid | catid | dateid |          eventname          |      starttime
    |
+---------+---------+-------+--------+-------------------------------
+---------------------+
```

Erstellen Sie Rollen mit Schreibschutz und Lese-/Schreibzugriff für das Marketingschema (optional)

In diesem Schritt erstellen Sie Rollen mit Schreibschutz und Lese-/Schreibzugriff für das Marketingschema.

1. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.

2. Verwenden Sie das folgende Beispiel, um Rollen mit Lese-/Schreibzugriff für das Marketingschema zu erstellen.

```
CREATE ROLE marketing_ro;

CREATE ROLE marketing_rw;

GRANT USAGE ON SCHEMA marketing TO ROLE marketing_ro, ROLE marketing_rw;

GRANT SELECT ON ALL TABLES IN SCHEMA marketing TO ROLE marketing_ro;

GRANT ROLE marketing_ro TO ROLE marketing_rw;

GRANT INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA marketing TO ROLE marketing_rw;

CREATE USER marketinganalyst PASSWORD 'Test12345';

CREATE USER marketingengineer PASSWORD 'Test12345';

GRANT ROLE marketing_ro TO marketinganalyst;

GRANT ROLE marketing_rw TO marketingengineer;
```

Systemfunktionen für RBAC (optional)

Amazon Redshift bietet zwei Funktionen zur Bereitstellung von Systeminformationen zur Benutzermitgliedschaft und Rollenmitgliedschaft in zusätzlichen Gruppen oder Rollen: role_is_member_of und user_is_member_of. Diese Funktionen stehen Superusern und regulären Benutzern zur Verfügung. Superuser können alle Rollenmitgliedschaften überprüfen. Reguläre Benutzer können nur die Mitgliedschaft von Rollen überprüfen, für die ihnen Zugriff gewährt wurde.

Um die Funktion role_is_member_of zu verwenden

1. Stellen Sie als salesengineer-Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um zu überprüfen, ob die Rolle sales_rw ein Mitglied der Rolle sales_ro ist.

```
SELECT role_is_member_of('sales_rw', 'sales_ro');

+-------------------+
```

```
| role_is_member_of |
+-------------------+
| true              |
+-------------------+
```

3. Verwenden Sie das folgende Beispiel, um zu überprüfen, ob die Rolle sales_ro ein Mitglied der Rolle sales_rw ist.

```
SELECT role_is_member_of('sales_ro', 'sales_rw');

+-------------------+
| role_is_member_of |
+-------------------+
| false             |
+-------------------+
```

Um die Funktion user_is_member_of zu verwenden

1. Stellen Sie als salesengineer-Benutzer eine Connect zur Datenbank her.
2. Im folgenden Beispiel wird versucht, die Benutzermitgliedschaft des Benutzers salesanalyst zu überprüfen. Diese Abfrage führt zu einem Fehler, da salesengineer keinen Zugriff auf salesanalyst hat. Um diesen Befehl erfolgreich auszuführen, stellen Sie als salesanalyst-Benutzer eine Verbindung mit der Datenbank her und verwenden Sie das Beispiel.

```
SELECT user_is_member_of('salesanalyst', 'sales_ro');

ERROR
```

3. Stellen Sie als Superuser eine Connect zur Datenbank her.
4. Verwenden Sie das folgende Beispiel, um die Mitgliedschaft des Benutzers salesanalyst zu überprüfen, wenn er als Superuser angemeldet ist.

```
SELECT user_is_member_of('salesanalyst', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| true              |
+-------------------+
```

5. Stellen Sie als dbadmin-Benutzer eine Connect zur Datenbank her.
6. Verwenden Sie das folgende Beispiel, um die Mitgliedschaft des Benutzers salesengineer zu überprüfen.

```
SELECT user_is_member_of('salesengineer', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| true              |
+-------------------+

SELECT user_is_member_of('salesengineer', 'marketing_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| false             |
+-------------------+

SELECT user_is_member_of('marketinganalyst', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| false             |
+-------------------+
```

Systemansichten für RBAC (optional)

Verwenden Sie die Systemansichten für Amazon Redshift, um die Rollen, die Rollenzuweisung zu Benutzern, die Rollenhierarchie und die Rechte für Datenbankobjekte über Rollen anzuzeigen. Diese Ansichten sind für Superuser und reguläre Benutzer verfügbar. Superuser können alle Rollendetails überprüfen. Reguläre Benutzer können nur die Details von Rollen überprüfen, für die ihnen Zugriff gewährt wurde.

1. Verwenden Sie das folgende Beispiel, um eine Liste der Benutzer anzuzeigen, denen explizit Rollen im Cluster gewährt wurden.

```
SELECT * FROM svv_user_grants;
```

2. Verwenden Sie das folgende Beispiel, um eine Liste der Rollen anzuzeigen, denen im Cluster explizit Rollen zugewiesen wurden.

```
SELECT * FROM svv_role_grants;
```

Die vollständige Liste der Systemansichten finden Sie unterSVV-Metadatenansichten.

Verwenden Sie Sicherheit auf Zeilenebene mit RBAC (optional)

Verwenden Sie Sicherheit auf Zeilenebene (RLS), um eine detaillierte Zugriffskontrolle für Ihre vertraulichen Daten zu erhalten. Weitere Informationen zu RLS finden Sie unter Sicherheit auf Zeilenebene.

In diesem Abschnitt erstellen Sie eine RLS-Richtlinie, die dem `salesengineer` Benutzer die Berechtigung gibt, nur Zeilen in der `cat` Tabelle anzuzeigen, die den `catdesc` Wert Major League Baseball haben. Anschließend fragen Sie die Datenbank als `salesengineer` Benutzer ab.

1. Stellen Sie als `salesengineer` Benutzer eine Connect zur Datenbank her.
2. Verwenden Sie das folgende Beispiel, um die ersten 5 Einträge in der `cat` Tabelle anzuzeigen.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

3. Stellen Sie als `dbadmin` Benutzer eine Connect zur Datenbank her.
4. Verwenden Sie das folgende Beispiel, um eine RLS-Richtlinie für die `catdesc` Spalte in der `cat` Tabelle zu erstellen.

```
CREATE RLS POLICY policy_mlb_engineer
```

```
WITH (catdesc VARCHAR(50))
USING (catdesc = 'Major League Baseball');
```

5. Verwenden Sie das folgende Beispiel, um die RLS-Richtlinie an die `sales_rw` Rolle anzuhängen.

```
ATTACH RLS POLICY policy_mlb_engineer ON sales.cat TO ROLE sales_rw;
```

6. Verwenden Sie das folgende Beispiel, um die Tabelle so zu ändern, dass RLS aktiviert wird.

```
ALTER TABLE sales.cat ROW LEVEL SECURITY ON;
```

7. Stellen Sie als `salesengineer` Benutzer eine Connect zur Datenbank her.
8. Verwenden Sie das folgende Beispiel, um zu versuchen, die ersten 5 Einträge in der `cat` Tabelle anzuzeigen. Beachten Sie, dass nur Einträge angezeigt werden, wenn die `catdesc` Spalte es ist`Major League Baseball`.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
|     1 | Sports   | MLB     | Major League Baseball |
+-------+----------+---------+----------------------+
```

9. Stellen Sie als `salesanalyst` Benutzer eine Connect zur Datenbank her.
10. Verwenden Sie das folgende Beispiel, um zu versuchen, die ersten 5 Einträge in der `cat` Tabelle anzuzeigen. Beachten Sie, dass keine Einträge angezeigt werden, da die Standardrichtlinie „Alle verweigern“ angewendet wird.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
```

11. Stellen Sie als `dbadmin` Benutzer eine Connect zur Datenbank her.
12. Verwenden Sie das folgende Beispiel, um der `sales_ro` Rolle die RLS-Berechtigung IGNORE zu erteilen. Dadurch erhält der `salesanalyst` Benutzer die Berechtigung, RLS-Richtlinien zu ignorieren, da er Mitglied der `sales_ro` Rolle ist.

```
GRANT IGNORE RLS TO ROLE sales_ro;
```

13. Stellen Sie als `salesanalyst` Benutzer eine Connect zur Datenbank her.
14. Verwenden Sie das folgende Beispiel, um die ersten 5 Einträge in der `cat` Tabelle anzuzeigen.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

15. Stellen Sie als `dbadmin` Benutzer eine Connect zur Datenbank her.
16. Verwenden Sie das folgende Beispiel, um der `sales_ro` Rolle die IGNORE-RLS-Berechtigung zu entziehen.

```
REVOKE IGNORE RLS FROM ROLE sales_ro;
```

17. Stellen Sie als `salesanalyst` Benutzer eine Connect zur Datenbank her.
18. Verwenden Sie das folgende Beispiel, um zu versuchen, die ersten 5 Einträge in der `cat` Tabelle anzuzeigen. Beachten Sie, dass keine Einträge angezeigt werden, da die Standardrichtlinie „Alle verweigern“ angewendet wird.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;
```

```
+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
```

19. Stellen Sie als `dbadmin` Benutzer eine Connect zur Datenbank her.
20. Verwenden Sie das folgende Beispiel, um die RLS-Richtlinie von der `cat` Tabelle zu trennen.

```
DETACH RLS POLICY policy_mlb_engineer ON cat FROM ROLE sales_rw;
```

21. Stellen Sie als `salesanalyst` Benutzer eine Connect zur Datenbank her.
22. Verwenden Sie das folgende Beispiel, um zu versuchen, die ersten 5 Einträge in der `cat` Tabelle anzuzeigen. Beachten Sie, dass keine Einträge angezeigt werden, da die Standardrichtlinie „Alle verweigern“ angewendet wird.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

23. Stellen Sie als `dbadmin` Benutzer eine Connect zur Datenbank her.
24. Verwenden Sie das folgende Beispiel, um die RLS-Richtlinie zu löschen.

```
DROP RLS POLICY policy_mlb_engineer;
```

25. Verwenden Sie das folgende Beispiel, um RLS zu entfernen.

```
ALTER TABLE cat ROW LEVEL SECURITY OFF;
```

Verwandte Themen

Weitere Informationen zu RBAC finden Sie in der folgenden Dokumentation:

- Rollenhierarchie
- Rollenzuweisung
- Datenbankobjektberechtigungen
- ALTER DEFAULT PRIVILEGES für RBAC

Sicherheit auf Zeilenebene

Mit Row-Level-Security (RLS) in Amazon Redshift können Sie eine granulare Zugriffssteuerung für Ihre sensiblen Daten vornehmen. Sie können entscheiden, welche Benutzer oder Rollen auf bestimmte Datensätze in Schemas oder Tabellen zugreifen können, basierend auf RLS-Richtlinien, die auf Datenbankobjektebene definiert sind. Zusätzlich zur Sicherheit auf Spaltenebene, bei der Sie Benutzern Berechtigungen für eine Teilmenge von Spalten erteilen können, verwenden Sie RLS-Richtlinien (Row-Level Security), um den Zugriff auf bestimmte Zeilen der sichtbaren Spalten weiter einzuschränken. Weitere Hinweise zur Sicherheit auf Spaltenebene finden Sie unter Verwendungshinweise für die Zugriffskontrolle auf Spaltenebene.

Wenn Sie RLS-Richtlinien (Row-Level Security) für Tabellen erzwingen, können Sie zurückgegebene Ergebnismengen einschränken, wenn Benutzer Abfragen ausführen.

Beim Erstellen von RLS-Richtlinien (Row-Level Security) können Sie Ausdrücke angeben, die bestimmen, ob Amazon Redshift vorhandene Zeilen in einer Tabelle in einer Abfrage zurückgibt. Durch das Erstellen von RLS-Richtlinien (Row-Level Security) zur Einschränkung des Zugriffs müssen Sie keine zusätzlichen Bedingungen in Ihren Abfragen hinzufügen oder externalisieren.

Beim Erstellen von RLS-Richtlinien (Row-Level Security) empfehlen wir, dass Sie einfache Richtlinien erstellen und komplexe Anweisungen in Richtlinien vermeiden. Verwenden Sie beim Definieren von RLS-Richtlinien (Row-Level Security) keine übermäßigen Tabellenverknüpfungen in der Richtliniendefinition, die auf Richtlinien basieren.

Wenn eine Richtlinie auf eine Nachschlagetabelle verweist, scannt Amazon Redshift diese weitere Tabelle zusätzlich zu der Tabelle, in der die Richtlinie enthalten ist. Es gibt Leistungsunterschiede zwischen derselben Abfrage für einen Benutzer mit angefügter RLS-Richtlinie und für einen Benutzer ohne angefügte Richtlinie.

Verwenden von RLS-Richtlinien (Row-Level Security) in SQL-Anweisungen

Bei der Verwendung von RLS-Richtlinien (Row-Level Security) in SQL-Anweisungen wendet Amazon Redshift die folgenden Regeln an:

- Amazon Redshift wendet RLS-Richtlinien (Row-Level Security) standardmäßig auf die SELECT-, UPDATE- und DELETE-Anweisungen an.
- Für SELECT und UNLOAD filtert Amazon Redshift Zeilen gemäß Ihrer definierten Richtlinie.
- Für UPDATE aktualisiert Amazon Redshift nur die Zeilen, die für Sie sichtbar sind. Wenn eine Richtlinie eine Teilmenge der Zeilen in einer Tabelle einschränkt, können Sie diese nicht aktualisieren.
- Für DELETE können Sie nur die Zeilen löschen, die für Sie sichtbar sind. Wenn eine Richtlinie eine Teilmenge der Zeilen in einer Tabelle einschränkt, können Sie sie nicht löschen. Für TRUNCATE können Sie die Tabelle trotzdem abschneiden.
- Bei CREATE TABLE LIKE erben Tabellen, die mit den LIKE-Optionen erstellt wurden, keine Berechtigungseinstellungen von der Quelltabelle. Ebenso erbt die Zieltabelle die RLS-Richtlinien (Row-Level Security) nicht von der Quelltabelle.

Kombinieren mehrerer Richtlinien pro Benutzer

RLS in Amazon Redshift unterstützt das Anfügen mehrerer Richtlinien pro Benutzer und Objekt. Wenn für einen Benutzer mehrere Richtlinien definiert sind, wendet Amazon Redshift alle Richtlinien mit der UND- oder ODER-Syntax an, abhängig von der Einstellung RLS CONJUNCTION TYPE für die Tabelle. Weitere Informationen zu Verbindungstypen finden Sie unter ALTER TABLE.

Ihnen können mehrere Richtlinien für eine Tabelle zugeordnet werden. Entweder sind Ihnen mehrere Richtlinien direkt zugeordnet, oder Sie gehören mehreren Rollen an, und den Rollen sind unterschiedliche Richtlinien zugeordnet.

Wenn mehrere Richtlinien den Zeilenzugriff in einer bestimmten Relation einschränken sollen, können Sie RLS CONJUNCTION TYPE der Relation auf AND setzen. Betrachten Sie das folgende Beispiel. Alice kann nur Sport-Ereignisse sehen, deren „Katzenname“ in der angegebenen Richtlinie „NBA“ ist.

```
-- Create an analyst role and grant it to a user named Alice.
CREATE ROLE analyst;
```

```
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
GRANT ROLE analyst TO alice;

-- Create an RLS policy that only lets the user see sports.
CREATE RLS POLICY policy_sports
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Sports');

-- Create an RLS policy that only lets the user see NBA.
CREATE RLS POLICY policy_nba
WITH (catname VARCHAR(10))
USING (catname = 'NBA');

-- Attach both to the analyst role.
ATTACH RLS POLICY policy_sports ON category TO ROLE analyst;
ATTACH RLS POLICY policy_nba ON category TO ROLE analyst;

-- Activate RLS on the category table with AND CONJUNCTION TYPE.
ALTER TABLE category ROW LEVEL SECURITY ON CONJUNCTION TYPE AND;

-- Change session to Alice.
SET SESSION AUTHORIZATION alice;

-- Select all from the category table.
SELECT catgroup, catname
FROM category;

 catgroup | catname
---------+---------
 Sports   | NBA
(1 row)
```

Wenn mehrere Richtlinien Benutzern ermöglichen sollen, mehr Zeilen in einer bestimmten Relation zu sehen, kann der Benutzer RLS CONJUNCTION TYPE der Relation auf OR setzen. Betrachten Sie das folgende Beispiel. Alice kann gemäß der angegebenen Richtlinie nur „Konzerte“ und „Sport“ sehen.

```
-- Create an analyst role and grant it to a user named Alice.
CREATE ROLE analyst;
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
GRANT ROLE analyst TO alice;

-- Create an RLS policy that only lets the user see concerts.
```

```
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');

-- Create an RLS policy that only lets the user see sports.
CREATE RLS POLICY policy_sports
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Sports');

-- Attach both to the analyst role.
ATTACH RLS POLICY policy_concerts ON category TO ROLE analyst;
ATTACH RLS POLICY policy_sports ON category TO ROLE analyst;

-- Activate RLS on the category table with OR CONJUNCTION TYPE.
ALTER TABLE category ROW LEVEL SECURITY ON CONJUNCTION TYPE OR;

-- Change session to Alice.
SET SESSION AUTHORIZATION alice;

-- Select all from the category table.
SELECT catgroup, count(*)
FROM category
GROUP BY catgroup ORDER BY catgroup;

 catgroup | count
---------+-------
 Concerts |  3
 Sports   |  5
(2 rows)
```

Besitz und Verwaltung von RLS-Richtlinien (Row-Level Security)

Als Superuser, Sicherheitsadministrator oder Benutzer mit der Rolle sys:secadmin können Sie alle RLS-Richtlinien (Row-Level Security) für Tabellen erstellen, ändern oder verwalten. Auf Objektebene können Sie die Sicherheit auf Zeilenebene ein- oder ausschalten, ohne die Schemadefinition für Tabellen zu ändern.

Um mit der Sicherheit auf Zeilenebene zu beginnen, können Sie die folgenden SQL-Anweisungen verwenden:

- Verwenden Sie die ALTER TABLE-Anweisung, um das RLS für eine Tabelle ein- oder auszuschalten. Weitere Informationen finden Sie unter ALTER TABLE.

- Verwenden Sie die CREATE RLS POLICY-Anweisung, um eine RLS-Richtlinie (Row-Level Security) für eine oder mehrere Tabellen zu erstellen, und geben Sie einen oder mehrere Benutzer oder Rollen in der Richtlinie an.

  Weitere Informationen finden Sie unter CREATE RLS POLICY.

- Verwenden Sie die Anweisung ALTER RLS POLICY, um die Richtlinie zu bearbeiten, z. B. die Richtliniendefinition zu ändern. Sie können dieselbe Richtlinie für mehrere Tabellen oder Ansichten verwenden.

  Weitere Informationen finden Sie unter ALTER RLS POLICY.

- Verwenden Sie die Anweisung ATTACH RLS POLICY, um eine Richtlinie an eine oder mehrere Relationen, einen oder mehrere Benutzer oder Rollen anzufügen.

  Weitere Informationen finden Sie unter ATTACH RLS POLICY .

- Verwenden Sie die Anweisung DETACH RLS POLICY, um eine Richtlinie von einer oder mehreren Beziehungen, von einem oder mehreren Benutzern oder von Rollen zu trennen.

  Weitere Informationen finden Sie unter DETACH RLS POLICY .

- Verwenden Sie die DROP RLS POLICY-Anweisung, um eine Richtlinie zu entfernen.

  Weitere Informationen finden Sie unter DROP RLS POLICY .

- Verwenden Sie die GRANT- und REVOKE-Anweisungen, um SELECT-Berechtigungen für RLS-Richtlinien(Row-Level Security) , die auf Nachschlagetabellen verweisen, explizit zu erteilen und zu widerrufen. Weitere Informationen finden Sie unter GRANT und REVOKE.

Um die erstellten Richtlinien zu überwachen, kann sys:secadmin die SVV_RLS_POLICY und SVV_RLS_ATTACHED_POLICY anzeigen.

Um RLS-geschützte Relationen aufzulisten, kann sys:secadmin SVV_RLS_RELATION anzeigen.

Um die Anwendung von RLS-Richtlinien (Row-Level Security) auf Abfragen zu verfolgen, die auf RLS-geschützte Relationen verweisen, kann ein Superuser, sys:operator oder jeder Benutzer mit der Systemberechtigung ACCESS SYSTEM TABLE SVV_RLS_APPLIED_POLICY anzeigen. Beachten Sie, dass sys:secadmin diese Berechtigungen standardmäßig nicht gewährt werden.

Um Tabellen mit zugeordneten RLS-Richtlinien (Row-Level Security) abzufragen, sie aber nicht anzuzeigen, können Sie jedem Benutzer die Berechtigung IGNORE RLS erteilen. Benutzern, die

Superuser oder sys:secadmin sind, wird automatisch die Berechtigung IGNORE RLS gewährt. Weitere Informationen finden Sie unter GRANT.

Um die RLS-Richtlinienfilter (Row-Level Security) einer Abfrage im EXPLAIN-Plan zur Fehlerbehebung bei RLS-bezogenen Abfragen zu erklären, können Sie jedem Benutzer die Berechtigung EXPLAIN RLS erteilen. Weitere Informationen finden Sie unter GRANT und EXPLAIN.

Richtlinienabhängige Objekte und Prinzipien

Um die Sicherheit für Anwendungen zu gewährleisten und zu verhindern, dass Richtlinienobjekte veraltet oder ungültig werden, lässt Amazon Redshift das Entfernen oder Ändern von Objekten, auf die von RLS-Richtlinien verwiesen wird, nicht zu.

Das folgende Beispiel veranschaulicht, wie die Schemaabhängigkeit nachverfolgt wird.

```
-- The CREATE and ATTACH policy statements for `policy_events` references some
-- target and lookup tables.
-- Target tables are tickit_event_redshift and target_schema.target_event_table.
-- Lookup table is tickit_sales_redshift.
-- Policy `policy_events` has following dependencies:
--   table tickit_sales_redshift column eventid, qtysold
--   table tickit_event_redshift column eventid
--   table target_event_table column eventid
--   schema public and target_schema
CREATE RLS POLICY policy_events
WITH (eventid INTEGER)
USING (
    eventid IN (SELECT eventid FROM tickit_sales_redshift WHERE qtysold <3)
);

ATTACH RLS POLICY policy_events ON tickit_event_redshift TO ROLE analyst;

ATTACH RLS POLICY policy_events ON target_schema.target_event_table TO ROLE consumer;
```

Im Folgenden werden Schema-Objektabhängigkeiten aufgelistet, die Amazon Redshift für RLS-Richtlinien (Row-Level Security) verfolgt.

- Beim Verfolgen der Schemaobjektabhängigkeit für die Zieltabelle folgt Amazon Redshift diesen Regeln:
  - Amazon Redshift trennt die Richtlinie von einer Relation, einem Benutzer, einer Rolle oder einer Öffentlichkeit, wenn Sie eine Zieltabelle entfernen.

  - Wenn Sie den Namen einer Zieltabelle umbenennen, hat dies keine Auswirkungen auf die zugeordneten Richtlinien.
  - Sie können die Spalten der Zieltabelle, auf die in der Richtliniendefinition verwiesen wird, nur entfernen, wenn Sie zuerst die Richtlinie entfernen oder trennen. Dies gilt auch, wenn die Option CASCADE angegeben ist. Sie können andere Spalten in der Zieltabelle entfernen.
  - Sie können die referenzierten Spalten der Zieltabelle nicht umbenennen. Um verwiesene Spalten umzubenennen, trennen Sie zuerst die Richtlinie. Dies gilt auch, wenn die Option CASCADE angegeben ist.
  - Sie können den Typ der referenzierten Spalte nicht ändern, selbst wenn Sie die CASCADE-Option angeben.
- Bei der Verfolgung der Schema-Objekt-Abhängigkeit für die Nachschlagetabelle folgt Amazon Redshift diesen Regeln:
  - Sie können eine Nachschlagetabelle nicht entfernen. Um eine Nachschlagetabelle zu entfernen, entfernen Sie zuerst die Richtlinie, in der auf die Nachschlagetabelle verwiesen wird.
  - Sie können eine Nachschlagetabelle nicht umbenennen. Um eine Nachschlagetabelle umzubenennen, entfernen Sie zunächst die Richtlinie, in der auf die Nachschlagetabelle verwiesen wird. Dies gilt auch, wenn die Option CASCADE angegeben ist.
  - Sie können die in der Richtliniendefinition verwendeten Spalten der Nachschlagetabelle nicht entfernen. Um die in der Richtliniendefinition verwendeten Nachschlagetabellenspalten zu entfernen, entfernen Sie zunächst die Richtlinie, in der die Nachschlagetabelle referenziert wird. Dies gilt auch, wenn die Option CASCADE in der ALTER TABLE DROP COLUMN-Anweisung angegeben ist. Sie können andere Spalten in der Nachschlagetabelle entfernen.
  - Sie können die referenzierten Spalten der Nachschlagetabelle nicht umbenennen. Um referenzierte Spalten umzubenennen, entfernen Sie zunächst die Richtlinie, in der auf die Nachschlagetabelle verwiesen wird. Dies gilt auch, wenn die Option CASCADE angegeben ist.
  - Sie können den Typ der referenzierten Spalte nicht ändern.
- Wenn ein Benutzer oder eine Rolle entfernt wird, trennt Amazon Redshift automatisch alle dem Benutzer oder der Rolle zugeordneten Richtlinien.
- Wenn Sie die CASCADE-Option in der DROP SCHEMA-Anweisung verwenden, entfernt Amazon Redshift auch die Relationen im Schema. Außerdem werden die Relationen in allen anderen Schemata entfernt, die von den Relationen des entfernten Schemas abhängig sind. Für eine Relation, die eine Nachschlagetabelle in einer Richtlinie ist, lässt Amazon Redshift die DROP SCHEMA DDL fehlschlagen. Für alle Relationen, die durch die DROP-SCHEMA-Anweisung entfernt werden, trennt Amazon Redshift alle Richtlinien, die diesen Relationen zugeordnet sind.

- Sie können eine Suchfunktion (eine Funktion, auf die innerhalb einer Richtliniendefinition verwiesen wird) nur entfernen, wenn Sie auch die Richtlinie entfernen. Dies gilt auch, wenn die Option CASCADE angegeben ist.
- Wenn eine Richtlinie an eine Tabelle angefügt wird, prüft Amazon Redshift, ob diese Tabelle eine Nachschlagetabelle in einer anderen Richtlinie ist. In diesem Fall lässt Amazon Redshift das Anhängen einer Richtlinie an diese Tabelle nicht zu.
- Beim Erstellen einer RLS-Richtlinie (Row-Level Security) prüft Amazon Redshift, ob diese Tabelle eine Zieltabelle für eine andere RLS-Richtlinie ist. In diesem Fall lässt Amazon Redshift das Erstellen einer Richtlinie für diese Tabelle nicht zu.

Überlegungen zur Verwendung von RLS-Richtlinien

Im Folgenden finden Sie Überlegungen für das Arbeiten mit RLS-Richtlinien (Row-Level Security):

- Amazon Redshift wendet RLS-Richtlinien (Row-Level Security) auf SELECT-, UPDATE- oder DELETE-Anweisungen an.
- Amazon Redshift wendet keine RLS-Richtlinien (Row-Level Security) auf die Anweisungen INSERT, COPY, ALTER TABLE APPEND an.
- Die Sicherheit auf Zeilenebene arbeitet mit der Sicherheit auf Spaltenebene zusammen, um Ihre Daten zu schützen.
- Wenn Ihr Amazon-Redshift-Cluster auf der neuesten allgemein verfügbaren Version war, die RLS (Row-Level Security) unterstützt, aber auf eine frühere Version heruntergestuft wurde, gibt Amazon Redshift einen Fehler zurück, wenn Sie eine Abfrage auf Basistabellen mit zugeordneten RLS-Richtlinien ausführen. Der sys:secadmin kann Benutzern, denen eingeschränkte Richtlinien gewährt wurden, den Zugriff entziehen, RLS für Tabellen deaktivieren und die Richtlinien entfernen.
- Wenn RLS für die Quellrelation aktiviert ist, unterstützt Amazon Redshift die Anweisung ALTER TABLE APPEND für Superuser, Benutzer, denen ausdrücklich das Systemberechtigung IGNORE RLS gewährt wurde, oder die Rolle sys:secadmin. In diesem Fall können Sie die Anweisung ALTER TABLE APPEND ausführen, um Zeilen an eine Zieltabelle anzufügen, indem Sie Daten aus einer vorhandenen Quelltabelle verschieben. Amazon Redshift verschiebt alle Tupel aus der Quellrelation in die Zielrelation. Der RLS-Status der Zielrelation hat keinen Einfluss auf die Anweisung ALTER TABLE APPEND.
- Um die Migration von anderen Data-Warehouse-Systemen zu erleichtern, können Sie benutzerdefinierte Sitzungskontextvariablen für eine Verbindung festlegen und abrufen, indem Sie den Variablennamen und -wert angeben.

Im folgenden Beispiel werden Sitzungskontextvariablen für eine Richtlinie für Sicherheit auf Zeilenebene (RLS) festgelegt.

```
-- Set a customized context variable.
SELECT set_config('app.category', 'Concerts', FALSE);

-- Create a RLS policy using current_setting() to get the value of a customized
 context variable.
CREATE RLS POLICY policy_categories
WITH (catgroup VARCHAR(10))
USING (catgroup = current_setting('app.category', FALSE));

-- Set correct roles and attach the policy on the target table to one or more roles.
ATTACH RLS POLICY policy_categories ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;
```

Einzelheiten zum Festlegen und Abrufen angepasster Sitzungskontextvariablen finden Sie unter SET, SET_CONFIG, ZEIGEN, CURRENT_SETTING, und RESET.

- Wenn Sie den Sitzungsbenutzer mit SET SESSION AUTHORIZATION zwischen DECLARE und FETCH oder zwischen nachfolgenden FETCH-Anweisungen ändern, wird der bereits vorbereitete Plan nicht aktualisiert, der auf den Benutzerrichtlinien zum Zeitpunkt von DECLARE basiert. Vermeiden Sie einen Wechsel des Sitzungsbenutzers, wenn Cursor mit RLS-geschützten Tabellen verwendet werden.
- Wenn die Basisobjekte innerhalb eines Ansichtsobjekts RLS-geschützt sind, werden Richtlinien, die dem die Abfrage ausführenden Benutzer angefügt sind, auf die entsprechenden Basisobjekte angewendet. Dies unterscheidet sich von den Berechtigungsprüfungen auf Objektebene, bei denen die Berechtigungen des Besitzers der Ansicht mit den Basisobjekten der Ansicht verglichen werden. Sie können die RLS-geschützten Relationen einer Abfrage in ihrer EXPLAIN-Planausgabe einsehen.
- Wenn in einer RLS-Richtlinie einer einem Benutzer angefügten Relation auf eine benutzerdefinierte Funktion (User-defined function, UDF) verwiesen wird, muss der Benutzer über die Berechtigung EXECUTE für die UDF verfügen, um die Relation abzufragen.
- Die Sicherheit auf Zeilenebene kann die Abfrageoptimierung einschränken. Wir empfehlen, die Abfrageleistung sorgfältig zu prüfen, bevor Sie RLS-geschützte Ansichten für große Datenmengen bereitstellen.

- Sicherheitsrichtlinien auf Zeilenebene, die auf Late-Binding-Ansichten angewendet werden, können in Verbundtabellen übertragen werden. Diese RLS-Richtlinien sind möglicherweise in den Protokollen der externen Verarbeitungs-Engine sichtbar.

Einschränkungen

Im Folgenden sind die Einschränkungen beim Arbeiten mit RLS-Richtlinien (Row-Level Security) aufgeführt:

- Amazon Redshift unterstützt SELECT-Anweisungen für bestimmte RLS-Richtlinien (Row-Level Security) mit Suchvorgängen, die komplexe Verknüpfungen aufweisen, unterstützt jedoch keine UPDATE- oder DELETE-Anweisungen. In Fällen mit UPDATE- oder DELETE-Anweisungen gibt Amazon Redshift den folgenden Fehler zurück:

  ```
  ERROR: One of the RLS policies on target relation is not supported in UPDATE/DELETE.
  ```

- Immer wenn in einer RLS-Richtlinie einer einem Benutzer angefügten Relation auf eine benutzerdefinierte Funktion (UDF) verwiesen wird, muss der Benutzer über die Berechtigung EXECUTE für die UDF verfügen, um die Relation abzufragen.
- Korrelierte Unterabfragen werden nicht unterstützt. Amazon Redshift gibt den folgenden Fehler zurück:

  ```
  ERROR: RLS policy could not be rewritten.
  ```

- RLS-Richtlinien können nicht an externe Tabellen und materialisierte Ansichten angefügt werden.
- Amazon Redshift unterstützt keine Datenfreigabe mit RLS. Wenn bei einer Relation RLS für Datashares nicht deaktiviert ist, schlägt die Abfrage im Konsumenten-Cluster mit dem folgenden Fehler fehl:

  ```
  RLS-protected relation "rls_protected_table" cannot be accessed via datasharing
   query.
  ```

- Bei datenbankübergreifenden Abfragen blockiert Amazon Redshift den Lesezugriff auf RLS-geschützte Relationen. Benutzer mit der Berechtigung IGNORE RLS können mithilfe von datenbankübergreifenden Abfragen auf die geschützte Relation zugreifen. Wenn ein Benutzer ohne die Berechtigung IGNORE RLS über eine datenbankübergreifende Abfrage auf die RLS-geschützte Relation zugreift, wird der folgende Fehler angezeigt:

```
RLS-protected relation "rls_protected_table" cannot be accessed via cross-database
 query.
```

- ALTER RLS POLICY unterstützt nur das Ändern einer RLS-Richtlinie mithilfe der USING (using_predicate_exp)-Klausel. Sie können eine RLS-Richtlinie nicht mit einer WITH-Klausel ändern, wenn Sie ALTER RLS POLICY ausführen.
- Sie können keine Beziehungen abfragen, für die die Sicherheit auf Zeilenebene aktiviert ist, wenn die Werte für eine der folgenden Konfigurationsoptionen nicht dem Standardwert der Sitzung entsprechen:
  - `enable_case_sensitive_super_attribute`
  - `enable_case_sensitive_identifier`
  - `downcase_delimited_identifier`

  Erwägen Sie, die Konfigurationsoptionen Ihrer Sitzung zurückzusetzen, wenn Sie versuchen, eine Beziehung mit aktivierter Sicherheit auf Zeilenebene abzufragen und die Meldung „Die RLS-geschützte Beziehung unterstützt keine Konfiguration auf Sitzungsebene, da die Berücksichtigung von Groß- und Kleinschreibung vom Standardwert abweicht“ angezeigt wird.
- Wenn Ihr bereitgestellter Cluster oder Serverless-Namespace über Sicherheitsrichtlinien auf Zeilenebene verfügt, werden die folgenden Befehle für normale Benutzer blockiert:

  ```
  ALTER <current_user> SET enable_case_sensitive_super_attribute/
  enable_case_sensitive_identifier/downcase_delimited_identifier
  ```

  Wenn Sie RLS-Richtlinien erstellen, empfehlen wir, die Einstellungen der Standardkonfigurationsoptionen für normale Benutzer so zu ändern, dass sie den Einstellungen der Sitzungskonfigurationsoptionen zum Zeitpunkt der Erstellung der Richtlinie entsprechen. Superuser und Benutzer mit der Berechtigung ALTER USER können dies mithilfe von Parametergruppeneinstellungen oder dem Befehl ALTER USER erreichen. Informationen zu Parametergruppen finden Sie unter Amazon-Redshift-Parametergruppen im Amazon-Redshift-Verwaltungshandbuch. Informationen zum Befehl ALTER USER finden Sie unter ALTER USER.
- Ansichten und Late-Binding-Ansichten mit Sicherheitsrichtlinien auf Zeilenebene können nicht von normalen Benutzern mit dem Befehl CREATE VIEW ersetzt werden. Um Ansichten oder LBVs durch RLS-Richtlinien zu ersetzen, trennen Sie zunächst alle mit diesen verknüpften RLS-Richtlinien, ersetzen Sie die Ansichten oder LBVs und fügen Sie die Richtlinien erneut an.

Superuser und Benutzer mit `sys:secadmin permission` können CREATE VIEW für Ansichten oder LBVs mit RLS-Richtlinien verwenden, ohne die Richtlinien zu trennen.

- Ansichten mit Sicherheitsrichtlinien auf Zeilenebene können nicht auf Systemtabellen und Systemansichten verweisen.
- Eine Late-Binding-Ansicht, auf die von einer regulären Ansicht verwiesen wird, kann nicht durch RLS geschützt werden.
- Auf RLS-geschützte Relationen und verschachtelte Daten aus Data Lakes kann nicht in derselben Abfrage zugegriffen werden.

Bewährte Methoden für RLS-Leistung

Im Folgenden finden Sie bewährte Methoden, um eine bessere Leistung von Amazon Redshift auf Tabellen zu gewährleisten, die durch RLS geschützt sind.

Sicherheit von Operatoren und Funktionen

Bei der Abfrage von RLS-geschützten Tabellen kann die Verwendung bestimmter Operatoren oder Funktionen zu Leistungseinbußen führen. Amazon Redshift klassifiziert Operatoren und Funktionen entweder als sicher oder unsicher für die Abfrage von RLS-geschützten Tabellen. Eine Funktion oder ein Operator wird als RLS-sicher eingestuft, wenn sie abhängig von den Eingaben keine beobachtbaren Nebenwirkungen hat. Insbesondere darf eine RLS-sichere Funktion oder ein Operator nicht einer der folgenden sein:

- Gibt einen Eingabewert oder jeden vom Eingabewert abhängigen Wert mit oder ohne Fehlermeldung aus.
- Schlägt fehl oder gibt Fehler zurück, die vom Eingabewert abhängen.

Zu den RLS-unsicheren Operatoren gehören:

- Arithmetische Operatoren — +, -, /, *, %.
- Textoperatoren – LIKE und SIMILAR TO.
- Cast-Operatoren.
- UDFs.

Verwenden Sie die folgende SELECT-Anweisung, um die Sicherheit von Operatoren und Funktionen zu überprüfen.

```
SELECT proname, proc_is_rls_safe(oid) FROM pg_proc;
```

Amazon Redshift schränkt die Reihenfolge der Auswertung von Benutzerprädikaten ein, die RLS-unsichere Operatoren und Funktionen enthalten, wenn Abfragen für RLS-geschützte Tabellen geplant werden. Abfragen, die auf RLS-unsichere Operatoren oder Funktionen verweisen, können beim Abfragen von RLS-geschützten Tabellen zu Leistungseinbußen führen. Die Leistung kann sich erheblich verschlechtern, wenn Amazon Redshift RLS-unsichere Prädikate nicht auf Basistabellen-Scans verschieben kann, um Sortierschlüssel zu nutzen.. Vermeiden Sie für eine bessere Leistung Abfragen mit RLS-unsicheren Prädikaten, die einen Sortierschlüssel nutzen. Um zu überprüfen, ob Amazon Redshift in der Lage ist, Operatoren und Funktionen weiterzugeben, können Sie EXPLAIN-Anweisungen in Kombination mit der Systemberechtigung EXPLAIN RLS verwenden.

Ergebnis-Zwischenspeicherung

Um die Laufzeit von Abfragen zu reduzieren und die Systemleistung zu verbessern, stellt Amazon Redshift die Ergebnisse bestimmter Abfragetypen in den Speicher auf dem Führungsknoten.

Amazon Redshift verwendet zwischengespeicherte Ergebnisse für eine neue Abfrage, die RLS-geschützte Tabellen scannt, wenn alle Bedingungen für ungeschützte Tabellen zutreffen und wenn alle der folgenden Bedingungen zutreffen:

- Die Tabellen oder Ansichten in der Richtlinie wurden nicht geändert.
- Die Richtlinie verwendet keine Funktion, die jedes Mal ausgewertet werden muss, wenn sie ausgeführt wird, wie z. B. GETDATE oder CURRENT_USER.

Um eine bessere Leistung zu erzielen, sollten Sie die Verwendung von Richtlinienprädikaten vermeiden, die die oben genannten Bedingungen nicht erfüllen.

Weitere Informationen zum Zwischenspeichern von Ergebnissen in Amazon Redshift finden Sie unter Ergebnis-Zwischenspeicherung .

Komplexe Richtlinien

Vermeiden Sie für eine bessere Leistung die Verwendung komplexer Richtlinien mit Unterabfragen, die mehrere Tabellen verknüpfen.

Erstellen, Anhängen, Trennen und Entfernen von RLS-Richtlinien (Row-Level Security)

Sie können folgende Aktionen ausführen:

- Um eine RLS-Richtlinie (Row-Level Security) zu erstellen, verwenden Sie den CREATE RLS POLICY-Befehl.
- Um eine RLS-Richtlinie (Row-Level Security) für eine Tabelle an einen oder mehrere Benutzer oder Rollen anzufügen, verwenden Sie den ATTACH RLS POLICY -Befehl.
- Um eine RLS-Richtlinie für eine Tabelle von einem oder mehreren Benutzern oder Rollen zu trennen, verwenden Sie den DETACH RLS POLICY -Befehl.
- Um eine RLS-Richtlinie (Row-Level Security) für alle Tabellen in allen Datenbanken zu entfernen, verwenden Sie den DROP RLS POLICY -Befehl.

Im Folgenden finden Sie ein end-to-end Beispiel, das veranschaulicht, wie ein Superuser einige Benutzer und Rollen erstellt. Anschließend erstellt ein Benutzer mit der Rolle secadmin RLS-Richtlinien (Row-Level Security), fügt sie hinzu, löst sie auf und entfernt sie. In diesem Beispiel wird die Musterdatenbank tickit verwendet. Weitere Informationen finden Sie unter Laden von Daten von Amazon S3 in Amazon Redshift im Amazon-Redshift-Handbuch für Erste Schritte.

```
-- Create users and roles referenced in the policy statements.
CREATE ROLE analyst;
CREATE ROLE consumer;
CREATE ROLE dbadmin;
CREATE ROLE auditor;
CREATE USER bob WITH PASSWORD 'Name_is_bob_1';
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
CREATE USER joe WITH PASSWORD 'Name_is_joe_1';
CREATE USER molly WITH PASSWORD 'Name_is_molly_1';
CREATE USER bruce WITH PASSWORD 'Name_is_bruce_1';
GRANT ROLE sys:secadmin TO bob;
GRANT ROLE analyst TO alice;
GRANT ROLE consumer TO joe;
GRANT ROLE dbadmin TO molly;
GRANT ROLE auditor TO bruce;
GRANT ALL ON TABLE tickit_category_redshift TO PUBLIC;
GRANT ALL ON TABLE tickit_sales_redshift TO PUBLIC;
GRANT ALL ON TABLE tickit_event_redshift TO PUBLIC;
```

```
-- Create table and schema referenced in the policy statements.
CREATE SCHEMA target_schema;
GRANT ALL ON SCHEMA target_schema TO PUBLIC;
CREATE TABLE target_schema.target_event_table (LIKE tickit_event_redshift);
GRANT ALL ON TABLE target_schema.target_event_table TO PUBLIC;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check the tuples visible to analyst alice.
-- Should contain all 3 categories.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');

SELECT poldb, polname, polalias, polatts, polqual, polenabled, polmodifiedby FROM
 svv_rls_policy WHERE poldb = CURRENT_DATABASE();

ATTACH RLS POLICY policy_concerts ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;

ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;

SELECT * FROM svv_rls_attached_policy;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check that tuples with only `Concert` category will be visible to analyst alice.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to consumer joe.
SET SESSION AUTHORIZATION joe;

-- Although the policy is attached to a different role, no tuples will be
```

```
-- visible to consumer joe because the default deny all policy is applied.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to dbadmin molly.
SET SESSION AUTHORIZATION molly;

-- Check that tuples with only `Concert` category will be visible to dbadmin molly.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Check that EXPLAIN output contains RLS SecureScan to prevent disclosure of
-- sensitive information such as RLS filters.
EXPLAIN SELECT catgroup, count(*) FROM tickit_category_redshift GROUP BY catgroup ORDER
 BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

-- Grant IGNORE RLS permission so that RLS policies do not get applicable to role
 dbadmin.
GRANT IGNORE RLS TO ROLE dbadmin;

-- Grant EXPLAIN RLS permission so that anyone in role auditor can view complete
 EXPLAIN output.
GRANT EXPLAIN RLS TO ROLE auditor;

-- Change session to dbadmin molly.
SET SESSION AUTHORIZATION molly;

-- Check that all tuples are visible to dbadmin molly because `IGNORE RLS` is granted
 to role dbadmin.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to auditor bruce.
SET SESSION AUTHORIZATION bruce;

-- Check explain plan is visible to auditor bruce because `EXPLAIN RLS` is granted to
 role auditor.
```

```
EXPLAIN SELECT catgroup, count(*) FROM tickit_category_redshift GROUP BY catgroup ORDER
 BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

DETACH RLS POLICY policy_concerts ON tickit_category_redshift FROM ROLE analyst, ROLE
 dbadmin;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check that no tuples are visible to analyst alice.
-- Although the policy is detached, no tuples will be visible to analyst alice
-- because of default deny all policy is applied if the table has RLS on.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

CREATE RLS POLICY policy_events
WITH (eventid INTEGER) AS ev
USING (
    ev.eventid IN (SELECT eventid FROM tickit_sales_redshift WHERE qtysold <3)
);

ATTACH RLS POLICY policy_events ON tickit_event_redshift TO ROLE analyst;
ATTACH RLS POLICY policy_events ON target_schema.target_event_table TO ROLE consumer;

RESET SESSION AUTHORIZATION;

-- Can not cannot alter type of dependent column.
ALTER TABLE target_schema.target_event_table ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_event_redshift ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_sales_redshift ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_sales_redshift ALTER COLUMN qtysold TYPE float;

-- Can not cannot rename dependent column.
ALTER TABLE target_schema.target_event_table RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_event_redshift RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_sales_redshift RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_sales_redshift RENAME COLUMN qtysold TO renamed_qtysold;
```

```
-- Can not drop dependent column.
ALTER TABLE target_schema.target_event_table DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_event_redshift DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_sales_redshift DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_sales_redshift DROP COLUMN qtysold CASCADE;

-- Can not drop lookup table.
DROP TABLE tickit_sales_redshift CASCADE;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

DROP RLS POLICY policy_concerts;
DROP RLS POLICY IF EXISTS policy_events;

ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY OFF;

RESET SESSION AUTHORIZATION;

-- Drop users and roles.
DROP USER bob;
DROP USER alice;
DROP USER joe;
DROP USER molly;
DROP USER bruce;
DROP ROLE analyst;
DROP ROLE consumer;
DROP ROLE auditor FORCE;
DROP ROLE dbadmin FORCE;
```

Sicherheit von Metadaten

Wie die Sicherheit von Amazon Redshift auf Zeilenebene bietet Ihnen die Metadatensicherheit eine genauere Kontrolle über Ihre Metadaten. Wenn die Metadatensicherheit für Ihren bereitgestellten Cluster oder Ihre Serverless-Arbeitsgruppe aktiviert ist, können Benutzer Metadaten für die Objekte sehen, für die sie Anzeigezugriff haben. Mithilfe der Metadatensicherheit können Sie die Sichtbarkeit nach Ihren Bedürfnissen trennen. Sie können beispielsweise ein einziges Data Warehouse verwenden, um Ihren gesamten Datenspeicher zu zentralisieren. Wenn Sie jedoch Daten für mehrere Sektoren speichern, kann die Verwaltung der Sicherheit schwierig werden. Wenn die Metadatensicherheit aktiviert ist, können Sie Ihre Sichtbarkeit konfigurieren. Benutzer eines Sektors

können ihre Objekte besser einsehen, während Sie den Anzeigezugriff auf Benutzer eines anderen Sektors einschränken. Die Metadatensicherheit unterstützt alle Objekttypen, wie z. B. Schemata, Tabellen, Ansichten, materialisierte Ansichten, gespeicherte Prozeduren, benutzerdefinierte Funktionen und Machine-Learning-Modelle.

Benutzer können Metadaten von Objekten unter den folgenden Umständen sehen:

- Wenn dem Benutzer Objektzugriff gewährt wurde.
- Wenn Objektzugriff einer Gruppe oder Rolle gewährt wurde, zu der der Benutzer gehört.
- Das Objekt ist öffentlich.
- Der Benutzer ist der Eigentümer des Datenbankobjekts.

Verwenden Sie den Befehl ALTER SYSTEM, um die Metadatensicherheit zu aktivieren.
Im Folgenden finden Sie die Syntax für die Verwendung des Befehls ALTER SYSTEM mit Metadatensicherheit.

```
ALTER SYSTEM SET metadata_security=[true|t|on|false|f|off];
```

Wenn Sie die Metadatensicherheit aktivieren, können alle Benutzer, die über die erforderlichen Berechtigungen verfügen, die relevanten Metadaten der Objekte sehen, auf die sie Zugriff haben. Wenn Sie möchten, dass nur bestimmte Benutzer die Metadatensicherheit sehen können, gewähren Sie einer Rolle die `ACCESS CATALOG`-Berechtigung und weisen Sie die Rolle dann dem Benutzer zu. Weitere Informationen zur Verwendung von Rollen zur besseren Kontrolle der Sicherheit finden Sie unter Rollenbasierte Zugriffskontrolle.

Das folgende Beispiel zeigt, wie Sie einer Rolle die `ACCESS CATALOG`-Berechtigung gewähren und die Rolle dann einem Benutzer zuweisen. Weitere Informationen zum Erteilen von Berechtigungen finden Sie unter dem GRANT-Befehl.

```
CREATE ROLE sample_metadata_viewer;

GRANT ACCESS CATALOG TO ROLE sample_metadata_viewer;

GRANT ROLE sample_metadata_viewer to salesadmin;
```

Wenn Sie es vorziehen, bereits definierte Rollen zu verwenden, verfügen die systemdefinierten Rollen `operator`, `secadmin`, `dba` und `superuser` alle über die erforderlichen Berechtigungen zum Anzeigen von Objektmetadaten. Standardmäßig können Superuser den vollständigen Katalog sehen.

```
GRANT ROLE operator to sample_user;
```

Wenn Sie Rollen zur Steuerung der Metadatensicherheit verwenden, haben Sie Zugriff auf alle Systemansichten und Funktionen, die mit der rollenbasierten Zugriffskontrolle einhergehen. Sie können beispielsweise die Ansicht SVV_ROLES abfragen, um alle Rollen zu sehen. Um zu sehen, ob ein Benutzer Mitglied einer Rolle oder Gruppe ist, verwenden Sie die Funktion USER_IS_MEMBER_OF. Eine vollständige Liste der SVV-Ansichten finden Sie unter SVV-Metadatenansichten. Eine Liste der Systeminformationsfunktionen finden Sie unter Systeminformationsfunktionen.

Dynamische Datenmaskierung

Übersicht

Mit der dynamischen Datenmaskierung (Dynamic Data Masking, DDM) in Amazon Redshift können Sie sensible Daten in Ihrem Data Warehouse schützen. Sie können festlegen, wie Amazon Redshift den Benutzern sensible Daten zum Zeitpunkt der Abfrage anzeigt, ohne diese in der Datenbank zu transformieren. Sie kontrollieren den Zugriff auf Daten mithilfe von Maskierungsrichtlinien, die benutzerdefinierte Verschleierungsregeln auf einen bestimmten Benutzer oder eine bestimmte Rolle anwenden. Auf diese Weise können Sie auf sich ändernde Datenschutzanforderungen reagieren, ohne die zugrunde liegenden Daten zu ändern oder SQL-Abfragen zu bearbeiten.

Richtlinien für die dynamische Datenmaskierung sorgen dafür, dass Daten, die einem bestimmten Format entsprechen, verborgen, verschleiert oder pseudonymisiert werden. Wenn ein Maskierungsausdruck an eine Tabelle angefügt ist, wird er auf eine oder mehrere Spalten der Tabelle angewendet. Sie können die Maskierungsrichtlinien weiter modifizieren, um sie nur auf bestimmte Benutzer oder auf benutzerdefinierte Rollen anzuwenden, die Sie mit Rollenbasierte Zugriffskontrolle (RBAC) erstellen. Darüber hinaus können Sie DDM auf Zellebene anwenden, indem Sie beim Erstellen Ihrer Maskierungsrichtlinie bedingte Spalten verwenden. Weitere Informationen zur bedingten Maskierung finden Sie unter Bedingte dynamische Datenmaskierung.

Sie können mehrere Maskierungsrichtlinien mit unterschiedlichen Verschleierungsgraden auf dieselbe Spalte in einer Tabelle anwenden und verschiedenen Rollen zuweisen. Um Konflikte zu vermeiden, wenn Sie verschiedene Rollen haben und unterschiedliche Richtlinien für eine Spalte gelten, können Sie Prioritäten für jede Anwendung festlegen. Auf diese Weise können Sie steuern, auf welche Daten ein bestimmter Benutzer oder eine bestimmte Rolle zugreifen kann. DDM-Richtlinien können Daten teilweise oder vollständig unkenntlich machen oder mithilfe von

benutzerdefinierten Funktionen, die in SQL, Python oder mit AWS Lambda geschrieben wurden, hashen. Durch das Maskieren von Daten mithilfe von Hashes können Sie Verknüpfungen auf diese Daten anwenden, ohne auf potenziell sensible Informationen zugreifen zu müssen.

nd-to-end Ein Beispiel

Das folgende end-to-end Beispiel zeigt, wie Sie Maskierungsrichtlinien erstellen und an eine Spalte anhängen können. Diese Richtlinien ermöglichen es Benutzern, auf eine Spalte zuzugreifen und unterschiedliche Werte zu sehen, je nachdem, welcher Grad der Verschleierung in den ihren Rollen zugeordneten Richtlinien festgelegt ist. Für die Durchführung dieses Beispiels müssen Sie Superuser sein oder über die Rolle `sys:secadmin` verfügen.

Erstellen einer Maskierungsrichtlinie

Erstellen Sie zunächst eine Tabelle und füllen Sie sie mit Kreditkartenwerten.

```
--create the table
CREATE TABLE credit_cards (
  customer_id INT,
  credit_card TEXT
);

--populate the table with sample values
INSERT INTO credit_cards
VALUES
  (100, '4532993817514842'),
  (100, '4716002041425888'),
  (102, '5243112427642649'),
  (102, '6011720771834675'),
  (102, '6011378662059710'),
  (103, '373611968625635')
;

--run GRANT to grant permission to use the SELECT statement on the table
GRANT SELECT ON credit_cards TO PUBLIC;

--create two users
CREATE USER regular_user WITH PASSWORD '1234Test!';

CREATE USER analytics_user WITH PASSWORD '1234Test!';

--create the analytics_role role and grant it to analytics_user
```

```
--regular_user does not have a role
CREATE ROLE analytics_role;

GRANT ROLE analytics_role TO analytics_user;
```

Erstellen Sie dann eine Maskierungsrichtlinie, die auf die Analyserolle angewendet werden soll.

```
--create a masking policy that fully masks the credit card number
CREATE MASKING POLICY mask_credit_card_full
WITH (credit_card VARCHAR(256))
USING ('000000XXXX0000'::TEXT);

--create a user-defined function that partially obfuscates credit card data
CREATE FUNCTION REDACT_CREDIT_CARD (credit_card TEXT)
RETURNS TEXT IMMUTABLE
AS $$
    import re
    regexp = re.compile("^([0-9]{6})[0-9]{5,6}([0-9]{4})")

    match = regexp.search(credit_card)
    if match != None:
        first = match.group(1)
        last = match.group(2)
    else:
        first = "000000"
        last = "0000"

    return "{}XXXXX{}".format(first, last)
$$ LANGUAGE plpythonu;

--create a masking policy that applies the REDACT_CREDIT_CARD function
CREATE MASKING POLICY mask_credit_card_partial
WITH (credit_card VARCHAR(256))
USING (REDACT_CREDIT_CARD(credit_card));

--confirm the masking policies using the associated system views
SELECT * FROM svv_masking_policy;

SELECT * FROM svv_attached_masking_policy;
```

Anfügen einer Maskierungsrichtlinie

Fügen Sie die Maskierungsrichtlinien der Kreditkartentabelle an.

```
--attach mask_credit_card_full to the credit card table as the default policy
--all users will see this masking policy unless a higher priority masking policy is
 attached to them or their role
ATTACH MASKING POLICY mask_credit_card_full
ON credit_cards(credit_card)
TO PUBLIC;

--attach mask_credit_card_partial to the analytics role
--users with the analytics role can see partial credit card information
ATTACH MASKING POLICY mask_credit_card_partial
ON credit_cards(credit_card)
TO ROLE analytics_role
PRIORITY 10;

--confirm the masking policies are applied to the table and role in the associated
 system view
SELECT * FROM svv_attached_masking_policy;

--confirm the full masking policy is in place for normal users by selecting from the
 credit card table as regular_user
SET SESSION AUTHORIZATION regular_user;

SELECT * FROM credit_cards;

--confirm the partial masking policy is in place for users with the analytics role by
 selecting from the credit card table as analytics_user
SET SESSION AUTHORIZATION analytics_user;

SELECT * FROM credit_cards;
```

Ändern einer Maskierungsrichtlinie

Der folgende Abschnitt veranschaulicht die Änderung einer Richtlinie für die dynamische Datenmaskierung.

```
--reset session authorization to the default
RESET SESSION AUTHORIZATION;

--alter the mask_credit_card_full policy
ALTER MASKING POLICY mask_credit_card_full
USING ('00000000000000'::TEXT);
```

```
--confirm the full masking policy is in place after altering the policy, and that
 results are altered from '000000XXXX0000' to '00000000000000'
SELECT * FROM credit_cards;
```

Trennen und Löschen einer Maskierungsrichtlinie

Im folgenden Abschnitt wird gezeigt, wie Sie Maskierungsrichtlinien trennen und löschen, indem Sie alle Richtlinien für die dynamische Datenmaskierung aus der Tabelle entfernen.

```
--reset session authorization to the default
RESET SESSION AUTHORIZATION;

--detach both masking policies from the credit_cards table
DETACH MASKING POLICY mask_credit_card_full
ON credit_cards(credit_card)
FROM PUBLIC;

DETACH MASKING POLICY mask_credit_card_partial
ON credit_cards(credit_card)
FROM ROLE analytics_role;

--drop both masking policies
DROP MASKING POLICY mask_credit_card_full;

DROP MASKING POLICY mask_credit_card_partial;
```

Hinweise zur dynamischen Datenmaskierung

Bei Verwendung der dynamischen Datenmaskierung ist Folgendes zu beachten:

- Beim Abfragen von Objekten, die aus Tabellen erstellt wurden, wie z. B. Ansichten, werden den Benutzern Ergebnisse auf der Grundlage ihrer eigenen Maskierungsrichtlinien angezeigt, nicht basierend auf den Richtlinien des Benutzers, der die Objekte erstellt hat. Einem Benutzer mit der Analystenrolle, der eine von einem Benutzer mit der Rolle „secadmin“ erstellte Ansicht abfragt, würden beispielsweise Ergebnisse mit Maskierungsrichtlinien angezeigt, die der Analystenrolle angefügt sind.
- Damit bei Verwendung des Befehls EXPLAIN keine sensiblen Maskierungsrichtlinienfilter offengelegt werden, können nur Benutzer mit der Berechtigung SYS_EXPLAIN_DDM Maskierungsrichtlinien sehen, die in EXPLAIN-Ausgaben angewendet wurden. Standardmäßig verfügen Benutzer nicht über die Berechtigung SYS_EXPLAIN_DDM.

Im Folgenden finden Sie die Syntax für die Erteilung von Berechtigungen für eine Rolle.

```
GRANT EXPLAIN MASKING TO ROLE rolename
```

Weitere Informationen zum Befehl EXPLAIN finden Sie unter EXPLAIN.

- Benutzer mit unterschiedlichen Rollen können je nach verwendeten Filter- oder Join-Bedingungen unterschiedliche Ergebnisse sehen. So schlägt beispielsweise die Ausführung eines Befehls SELECT für eine Tabelle mit einem bestimmten Spaltenwert fehl, wenn für den Benutzer, der den Befehl ausführt, eine Maskierungsrichtlinie angewendet wurde, die diese Spalte verschleiert.
- DDM-Richtlinien müssen vor allen Prädikatoperationen oder Projektionen angewendet werden. Die Maskierungsrichtlinien können Folgendes beinhalten:
  - Kostengünstige konstante Operationen wie die Umwandlung eines Werts in Null
  - Operationen zu moderaten Kosten wie HMAC-Hashing
  - Kostenintensive Operationen wie Aufrufe externer benutzerdefinierter Lambda-Funktionen

  Daher empfehlen wir, wenn möglich einfache Maskierungsausdrücke zu verwenden.
- Sie können DDM-Richtlinien für Rollen mit Sicherheitsrichtlinien auf Zeilenebene verwenden. Beachten Sie jedoch, dass RLS-Richtlinien vor DDM angewendet werden. Ein Ausdruck für die dynamische Datenmaskierung ist nicht in der Lage, eine Zeile zu lesen, die durch RLS geschützt wurde. Weitere Informationen zu RLS finden Sie unter Sicherheit auf Zeilenebene.
- Wenn Sie den Befehl COPY zum Kopieren aus Parquet in geschützte Zieltabellen verwenden, sollten Sie in der COPY-Anweisung explizit Spalten angeben. Weitere Informationen zur Zuordnung von Spalten mit COPY finden Sie unter Optionen für das Mapping von Spalten.
- DDM-Richtlinien können nicht an die folgenden Relationen angefügt werden:
  - Systemtabellen und Kataloge
  - Externe Tabellen
  - Datasharing-Tabellen
  - Materialisierte Ansichten
  - Datenbankübergreifende Relationen
  - Temporäre Tabellen
  - Korrelierte Abfragen
- DDM-Richtlinien können Suchtabellen enthalten. Suchtabellen können in der USING-Klausel enthalten sein. Die folgenden Relationstypen können nicht als Suchtabellen verwendet werden:

- Systemtabellen und Kataloge
- Externe Tabellen
- Datasharing-Tabellen
- Ansichten, materialisierte Ansichten und Late-Binding-Ansichten.
- Datenbankübergreifende Relationen
- Temporäre Tabellen
- Korrelierte Abfragen

Im Folgenden finden Sie ein Beispiel für das Anfügen einer Maskierungsrichtlinie an eine Suchtabelle.

```
--Create a masking policy referencing a lookup table
CREATE MASKING POLICY lookup_mask_credit_card WITH (credit_card TEXT) USING (
  CASE
    WHEN
      credit_card IN (SELECT credit_card_lookup FROM credit_cards_lookup)
    THEN '000000XXXX0000'
    ELSE REDACT_CREDIT_CARD(credit_card)
    END
  );

--Provides access to the lookup table via a policy attached to a role
GRANT SELECT ON TABLE credit_cards_lookup TO MASKING POLICY lookup_mask_credit_card;
```

- Sie können keine Maskierungsrichtlinie anfügen, die zu einer mit dem Typ und der Größe der Zielspalte nicht kompatiblen Ausgabe führen würde. So können Sie beispielsweise keine Maskierungsrichtlinie anfügen, die eine 12-stellige Zeichenfolge an eine VARCHAR (10)-Spalte ausgibt. Amazon Redshift unterstützt die folgenden Ausnahmen:
  - Eine Maskierungsrichtlinie mit dem Eingabetyp INTN kann einer Richtlinie mit der Größe INTM angefügt werden, solange M < N ist. Beispielsweise kann eine BIGINT (INT8)-Richtlinie an eine Smallint (INT4)-Spalte angefügt werden.
  - Eine Maskierungsrichtlinie mit dem Eingabetyp NUMERIC oder DECIMAL kann immer an eine FLOAT-Spalte angefügt werden.
- DDM-Richtlinien können nicht für die Freigabe von Daten verwendet werden. Wenn der Produzent der Datashare-Daten eine DDM-Richtlinie an eine Tabelle im Datashare anfügt, können Benutzer des Datenkonsumenten, die versuchen, die Tabelle abzufragen, nicht mehr auf die Tabelle

zugreifen. Tabellen mit angehängten DDM-Richtlinien können keinem Datashare hinzugefügt werden.

- Sie können keine Beziehungen abfragen, für über angefügte DDM-Richtlinien verfügen, wenn Ihre Werte für eine der folgenden Konfigurationsoptionen nicht dem Standardwert der Sitzung entsprechen:
  - `enable_case_sensitive_super_attribute`
  - `enable_case_sensitive_identifier`
  - `downcase_delimited_identifier`

  Erwägen Sie, die Konfigurationsoptionen Ihrer Sitzung zurückzusetzen, wenn Sie versuchen, eine Beziehung mit einer angefügten DDM-Richtlinie abzufragen und die Meldung „Die DDM-geschützte Beziehung unterstützt keine Konfiguration auf Sitzungsebene, da die Berücksichtigung von Groß- und Kleinschreibung vom Standardwert abweicht“ angezeigt wird.
- Wenn Ihr bereitgestellter Cluster oder Serverless-Namespace über Richtlinien zur dynamischen Datenmaskierung verfügt, werden die folgenden Befehle für normale Benutzer blockiert:

  ```
  ALTER <current_user> SET enable_case_sensitive_super_attribute/
  enable_case_sensitive_identifier/downcase_delimited_identifier
  ```

  Wenn Sie DDM-Richtlinien erstellen, empfehlen wir, die Einstellungen der Standardkonfigurationsoptionen für normale Benutzer so zu ändern, dass sie den Einstellungen der Sitzungskonfigurationsoptionen zum Zeitpunkt der Erstellung der Richtlinie entsprechen. Superuser und Benutzer mit der Berechtigung ALTER USER können dies mithilfe von Parametergruppeneinstellungen oder dem Befehl ALTER USER erreichen. Informationen zu Parametergruppen finden Sie unter Amazon-Redshift-Parametergruppen im Amazon-Redshift-Verwaltungshandbuch. Informationen zum Befehl ALTER USER finden Sie unter ALTER USER.
- Ansichten und Late-Binding-Ansichten mit DDM-Richtlinien auf Zeilenebene können nicht von normalen Benutzern mit dem Befehl CREATE VIEW ersetzt werden. Um Ansichten oder LBVs durch RLS-Richtlinien zu ersetzen, trennen Sie zunächst alle mit diesen verknüpften DDM-Richtlinien, ersetzen Sie die Ansichten oder LBVs und fügen Sie die Richtlinien erneut an. Superuser und Benutzer mit der `sys:secadmin`-Berechtigung können CREATE VIEW für Ansichten oder LBVs mit DDM-Richtlinien verwenden, ohne die Richtlinien zu trennen.
- Ansichten mit angefügten DDM-Richtlinien können nicht auf Systemtabellen und Ansichten verweisen. Late-Binding-Ansichten können auf Systemtabellen und Ansichten verweisen.

- Late-Binding-Ansichten mit angefügten DDM-Richtlinien können nicht auf verschachtelte Daten in Data Lakes verweisen, wie z. B. JSON-Dokumente.
- Late-Binding-Ansichten können keine DDM-Richtlinien angefügt werden, wenn die Late-Binding-Ansicht von einer Ansicht referenziert wird.
- DDM-Richtlinien, die an Late-Binding-Ansichten angefügt sind, werden anhand des Spaltennamens angefügt. Bei der Abfrage überprüft Amazon Redshift, ob alle Maskierungsrichtlinien, die an die Late-Binding-Ansicht angefügt sind, erfolgreich angewendet wurden und ob der Ausgabespaltentyp der Late-Binding-Ansicht mit den Typen in den angefügten Maskierungsrichtlinien übereinstimmt. Wenn die Überprüfung fehlschlägt, gibt Amazon Redshift einen Fehler für die Abfrage zurück.

Verwalten von Richtlinien für die dynamische Datenmaskierung

Sie können folgende Aktionen ausführen:

- Um eine DDM-Richtlinie zu erstellen, verwenden Sie den Befehl ERSTELLEN EINER MASKIERUNGSRICHTLINIE.

  Im Folgenden finden Sie ein Beispiel für die Erstellung einer Maskierungsrichtlinie mithilfe einer SHA-2-Hash-Funktion.

```
CREATE MASKING POLICY hash_credit
WITH (credit_card varchar(256))
USING (sha2(credit_card + 'testSalt', 256));
```

- Verwenden Sie den Befehl ALTER MASKING POLICY, um eine bestehende DDM-Richtlinie zu ändern.

  Im Folgenden finden Sie ein Beispiel für das Ändern einer vorhandenen Maskierungsrichtlinie.

```
ALTER MASKING POLICY hash_credit
USING (sha2(credit_card + 'otherTestSalt', 256));
```

- Um eine DDM-Richtlinie für eine Tabelle einem oder mehreren Benutzern oder Rollen anzufügen, verwenden Sie den Befehl ANFÜGEN EINER MASKIERUNGSRICHTLINIE.

  Im Folgenden finden Sie ein Beispiel für das Anfügen einer Maskierungsrichtlinie an ein Spalten-/Rollenpaar.

```
ATTACH MASKING POLICY hash_credit
```

```
ON credit_cards (credit_card)
TO ROLE science_role
PRIORITY 30;
```

Die PRIORITY-Klausel bestimmt, welche Maskierungsrichtlinie für eine Benutzersitzung gilt, wenn derselben Spalte mehrere Richtlinien angefügt sind. Wenn der Benutzer im vorherigen Beispiel beispielsweise derselben Kreditkartenspalte eine weitere Maskierungsrichtlinie mit einer Priorität von 20 angefügt hat, gilt die Richtlinie von science_role, da sie eine höhere Priorität von 30 hat.

- Um eine DDM-Richtlinie für eine Tabelle von einem oder mehreren Benutzern oder Rollen zu trennen, verwenden Sie den Befehl TRENNEN EINER MASKIERUNGSRICHTLINIE.

  Im Folgenden finden Sie ein Beispiel für das Trennen einer Maskierungsrichtlinie von einem Spalten-/Rollenpaar.

  ```
  DETACH MASKING POLICY hash_credit
  ON credit_cards(credit_card)
  FROM ROLE science_role;
  ```

- Um eine DDM-Richtlinie aus allen Datenbanken zu löschen, verwenden Sie den Befehl ENTFERNEN EINER MASKIERUNGSRICHTLINIE

  Im Folgenden finden Sie ein Beispiel für das Löschen einer Maskierungsrichtlinie aus allen Datenbanken.

  ```
  DROP MASKING POLICY hash_credit;
  ```

Hierarchie der Maskierungsrichtlinien

Beachten Sie beim Anfügen mehrerer Maskierungsrichtlinien die folgenden Überlegungen:

- Sie können mehrere Maskierungsrichtlinien an eine einzelne Spalte anfügen.
- Wenn mehrere Maskierungsrichtlinien für eine Abfrage anwendbar sind, gilt die Richtlinie mit der höchsten Priorität, die der jeweiligen Spalte anfügt ist. Betrachten Sie das folgende Beispiel.

  ```
  ATTACH MASKING POLICY partial_hash
  ON credit_cards(address, credit_card)
  TO ROLE analytics_role
  PRIORITY 20;
  ```

```
ATTACH MASKING POLICY full_hash
ON credit_cards(credit_card, ssn)
TO ROLE auditor_role
PRIORITY 30;

SELECT address, credit_card, ssn
FROM credit_cards;
```

Beim Ausführen der SELECT-Anweisung wird Benutzern mit Analyse- und Auditorrolle die Adressspalte mit angewendeter Maskierungsrichtlinie `partial_hash` angezeigt. Sie sehen die Spalten „Credit Card“ (Kreditkarte) und „SSN“, auf die die Maskierungsrichtlinie `full_hash` angewendet wurde, da die Richtline `full_hash` in der Spalte „Credit Card“ (Kreditkarte) die höhere Priorität hat.

- Wenn Sie beim Anfügen einer Maskierungsrichtlinie keine Priorität angeben, lautet die Standardpriorität 0.
- Sie können derselben Spalte nicht zwei Richtlinien mit derselben Priorität zuordnen.
- Sie können nicht derselben Kombination aus Benutzer und Spalte oder Rolle und Spalte zwei Richtlinien zuordnen.
- Wenn mehrere Maskierungsrichtlinien auf demselben SUPER-Pfad angewendet werden können, obwohl sie demselben Benutzer oder derselben Rolle zugeordnet sind, wird nur der Anhang mit der höchsten Priorität wirksam. Betrachten Sie die folgenden Beispiele:

  Das erste Beispiel zeigt zwei Maskierungsrichtlinien, die demselben Pfad zugeordnet sind, wobei die Richtlinie mit der höheren Priorität wirksam wird.

```
ATTACH MASKING POLICY hide_name
ON employees(col_person.name)
TO PUBLIC
PRIORITY 20;

ATTACH MASKING POLICY hide_last_name
ON employees(col_person.name.last)
TO PUBLIC
PRIORITY 30;

--Only the hide_last_name policy takes effect.
SELECT employees.col_person.name FROM employees;
```

Das zweite Beispiel zeigt zwei Maskierungsrichtlinien, die unterschiedlichen Pfaden im selben SUPER-Objekt zugeordnet sind, ohne dass es zu Konflikten zwischen den Richtlinien kommt. Beide Anhänge sind gleichzeitig gültig.

```
ATTACH MASKING POLICY hide_first_name
ON employees(col_person.name.first)
TO PUBLIC
PRIORITY 20;

ATTACH MASKING POLICY hide_last_name
ON employees(col_person.name.last)
TO PUBLIC
PRIORITY 20;

--Both col_person.name.first and col_person.name.last are masked.
SELECT employees.col_person.name FROM employees;
```

Um zu überprüfen, welche Maskierungsrichtlinie für eine bestimmte Kombination von Rolle/Benutzer und Spalte gilt, können Benutzer mit der Rolle `sys:secadmin` das Spalten-/Rollenpaar oder das Spalten-/Benutzerpaar in der SVV_ATTACHED_MASKING_POLICY-Systemansicht einsehen. Weitere Informationen finden Sie unter Systemansichten für dynamische Datenmaskierung.

Verwendung dynamischer Datenmaskierung mit Pfaden des Datentyps SUPER

Amazon Redshift unterstützt das Anfügen dynamischer Datenmaskierungsrichtlinien an Pfade von Spalten des Typs SUPER. Weitere Informationen zum SUPER-Datentyp finden Sie unter Erfassen und Abfragen von halbstrukturierten Daten in Amazon Redshift.

Beachten Sie beim Anfügen von Maskierungsrichtlinien an Pfade von Spalten des Typs SUPER Folgendes.

- Beim Anfügen einer Maskierungsrichtlinie an einen Pfad in einer Spalte muss diese Spalte mit dem Datentyp SUPER definiert werden. Sie können Maskierungsrichtlinien nur auf Skalarwerte im SUPER-Pfad anwenden. Sie können Maskierungsrichtlinien nicht auf komplexe Strukturen oder Arrays anwenden.
- Sie können unterschiedliche Maskierungsrichtlinien auf mehrere Skalarwerte in einer einzelnen SUPER-Spalte anwenden, sofern die SUPER-Pfade nicht miteinander in Konflikt stehen.

Beispielsweise stehen die SUPER-Pfade `a.b` in `a.b.c` Konflikt, weil sie sich auf demselben Pfad befinden, wobei `a.b` das übergeordnete Element von `a.b.c` ist. Die SUPER-Pfade `a.b.c` und `a.b.d` stehen nicht im Konflikt.

- Amazon Redshift kann erst überprüfen, ob die Pfade, denen eine Maskierungsrichtlinie angefügt wird, in den Daten existieren und den erwarteten Typ haben, wenn die Richtlinie zur Laufzeit der Benutzerabfrage angewendet wird. Wenn Sie beispielsweise eine Maskierungsrichtlinie anfügen, die TEXT-Werte mit einem SUPER-Pfad maskiert, der einen INT-Wert enthält, versucht Amazon Redshift, den Typ des Werts in dem Pfad umzuwandeln.

  In solchen Situationen hängt das Verhalten von Amazon Redshift zur Laufzeit von Ihren Konfigurationseinstellungen für die Abfrage von SUPER-Objekten ab. Standardmäßig befindet sich Amazon Redshift im Lax-Modus und löst fehlende Pfade und ungültige Umwandlungen wie beim angegebenen `NULL` für den SUPER-Pfad auf. Weitere Informationen zu den Konfigurationseinstellungen im Zusammenhang mit SUPER finden Sie unter SUPER-Konfigurationen.
- SUPER ist ein Typ ohne Schema, was bedeutet, dass Amazon Redshift die Existenz des Werts in einem bestimmten SUPER-Pfad nicht bestätigen kann. Wenn Sie eine Maskierungsrichtlinie an einen SUPER-Pfad anfügen, der nicht existiert, und Amazon Redshift sich im Lax-Modus befindet, löst Amazon Redshift den Pfad zu einem `NULL`-Wert auf. Wir empfehlen, das erwartete Format von SUPER-Objekten und die Wahrscheinlichkeit, dass sie unerwartete Attribute aufweisen, zu berücksichtigen, wenn Sie Maskierungsrichtlinien an Pfade von SUPER-Spalten anfügen. Wenn Sie vermuten, dass Ihre SUPER-Spalte ein unerwartetes Schema enthält, sollten Sie erwägen, Ihre Maskierungsrichtlinien direkt an die SUPER-Spalte anzufügen. Sie können die Informationsfunktionen vom Typ SUPER verwenden, um Attribute und Typen zu überprüfen und um mit `OBJECT_TRANSFORM` die Werte zu maskieren. Weitere Informationen zu SUPER-Informationsfunktionen finden Sie unter Funktionen für SUPER-Typinformationen.

Beispiele

Anfügen von Maskierungsrichtlinien an SUPER-Pfade

Im folgenden Beispiel werden mehrere Maskierungsrichtlinien an mehrere SUPER-Pfade in einer Spalte angefügt.

```
CREATE TABLE employees (
    col_person SUPER
);
```

```
INSERT INTO employees
VALUES
    (
        json_parse('
            {
                "name": {
                    "first": "John",
                    "last": "Doe"
                },
                "age": 25,
                "ssn": "111-22-3333",
                "company": "Company Inc."
            }
        ')
    ),
    (
        json_parse('
            {
                "name": {
                    "first": "Jane",
                    "last": "Appleseed"
                },
                "age": 34,
                "ssn": "444-55-7777",
                "company": "Organization Org."
            }
        ')
    )
;
GRANT ALL ON ALL TABLES IN SCHEMA "public" TO PUBLIC;

-- Create the masking policies.

-- This policy converts the given name to all uppercase letters.
CREATE MASKING POLICY mask_first_name
WITH(first_name TEXT)
USING ( UPPER(first_name) );

-- This policy replaces the given name with the fixed string 'XXXX'.
CREATE MASKING POLICY mask_last_name
WITH(last_name TEXT)
USING ( 'XXXX'::TEXT );

-- This policy rounds down the given age to the nearest 10.
```

```
CREATE MASKING POLICY mask_age
WITH(age INT)
USING ( (FLOOR(age::FLOAT / 10) * 10)::INT );

-- This policy converts the first five digits of the given SSN to 'XXX-XX'.
CREATE MASKING POLICY mask_ssn
WITH(ssn TEXT)
USING ( 'XXX-XX-'::TEXT || SUBSTRING(ssn::TEXT FROM 8 FOR 4) );

-- Attach the masking policies to the employees table.
ATTACH MASKING POLICY mask_first_name
ON employees(col_person.name.first)
TO PUBLIC;

ATTACH MASKING POLICY mask_last_name
ON employees(col_person.name.last)
TO PUBLIC;

ATTACH MASKING POLICY mask_age
ON employees(col_person.age)
TO PUBLIC;

ATTACH MASKING POLICY mask_ssn
ON employees(col_person.ssn)
TO PUBLIC;

-- Verify that your masking policies are attached.
SELECT
    policy_name,
    TABLE_NAME,
    priority,
    input_columns,
    output_columns
FROM
    svv_attached_masking_policy;

   policy_name   | table_name | priority |           input_columns           |
 output_columns
-----------------+------------+----------+-----------------------------------
+-----------------------------------
 mask_age        | employees  |        0 | ["col_person.\"age\""]            |
 ["col_person.\"age\""]
 mask_first_name | employees  |        0 | ["col_person.\"name\".\"first\""] |
 ["col_person.\"name\".\"first\""]
```

```
 mask_last_name  | employees  |        0 | ["col_person.\"name\".\"last\""]  |
 ["col_person.\"name\".\"last\""]
 mask_ssn        | employees  |        0 | ["col_person.\"ssn\""]            |
 ["col_person.\"ssn\""]
(4 rows)

-- Observe the masking policies taking effect.
SELECT col_person FROM employees ORDER BY col_person.age;

-- This result is formatted for ease of reading.
         col_person
--------------------------------
{
    "name": {
        "first": "JOHN",
        "last": "XXXX"
    },
    "age": 20,
    "ssn": "XXX-XX-3333",
    "company": "Company Inc."
}
{
    "name": {
        "first": "JANE",
        "last": "XXXX"
    },
    "age": 30,
    "ssn": "XXX-XX-7777",
    "company": "Organization Org."
}
```

Im Folgenden finden Sie einige Beispiele für ungültige Maskierungsrichtlinien-Anhänge an SUPER-Pfade.

```
-- This attachment fails because there is already a policy
-- with equal priority attached to employees.name.last, which is
-- on the same SUPER path as employees.name.
ATTACH MASKING POLICY mask_ssn
ON employees(col_person.name)
TO PUBLIC;
ERROR:  DDM policy "mask_last_name" is already attached on relation "employees" column
 "col_person."name"."last"" with same priority
```

```
-- Create a masking policy that masks DATETIME objects.
CREATE MASKING POLICY mask_date
WITH(INPUT DATETIME)
USING ( INPUT );

-- This attachment fails because SUPER type columns can't contain DATETIME objects.
ATTACH MASKING POLICY mask_date
ON employees(col_person.company)
TO PUBLIC;
ERROR:  cannot attach masking policy for output of type "timestamp without time zone"
 to column "col_person."company"" of type "super
```

Im Folgenden finden Sie ein Beispiel für das Anfügen einer Maskierungsrichtlinie an einen SUPER-Pfad, der nicht existiert. Standardmäßig löst Amazon Redshift den Pfad zu NULL auf.

```
ATTACH MASKING POLICY mask_first_name
ON employees(col_person.not_exists)
TO PUBLIC;

SELECT col_person FROM employees LIMIT 1;

-- This result is formatted for ease of reading.
         col_person
-----------------------------------
{
    "name": {
        "first": "JOHN",
        "last": "XXXX"
    },
    "age": 20,
    "ssn": "XXX-XX-3333",
    "company": "Company Inc.",
    "not_exists": null
}
```

Bedingte dynamische Datenmaskierung

Sie können Daten auf Zellenebene maskieren, indem Sie Maskierungsrichtlinien mit bedingten Ausdrücken im Maskierungsausdruck erstellen. So können Sie beispielsweise eine Maskierungsrichtlinie erstellen, die abhängig vom Wert einer anderen Spalte in dieser Zeile unterschiedliche Masken auf einen Wert anwendet.

Im Folgenden finden Sie ein Beispiel für die Verwendung der bedingten Datenmaskierung, um eine Maskierungsrichtlinie zu erstellen und anzufügen, die in Betrugsfälle verwickelte Kreditkartennummern teilweise unkenntlich, während alle anderen Kreditkartennummern vollständig ausgeblendet werden. Für die Durchführung dieses Beispiels müssen Sie Superuser sein oder über die Rolle `sys:secadmin` verfügen.

```
--Create an analyst role.
CREATE ROLE analyst;

--Create a credit card table. The table contains an is_fraud boolean column,
--which is TRUE if the credit card number in that row was involved in a fraudulent
 transaction.
CREATE TABLE credit_cards (id INT, is_fraud BOOLEAN, credit_card_number VARCHAR(16));

--Create a function that partially redacts credit card numbers.
CREATE FUNCTION REDACT_CREDIT_CARD (credit_card VARCHAR(16))
RETURNS VARCHAR(16) IMMUTABLE
AS $$
    import re
    regexp = re.compile("^([0-9]{6})[0-9]{5,6}([0-9]{4})")

    match = regexp.search(credit_card)
    if match != None:
        first = match.group(1)
        last = match.group(2)
    else:
        first = "000000"
        last = "0000"

    return "{}XXXXX{}".format(first, last)
$$ LANGUAGE plpythonu;

--Create a masking policy that partially redacts credit card numbers if the is_fraud
 value for that row is TRUE,
--and otherwise blanks out the credit card number completely.
CREATE MASKING POLICY card_number_conditional_mask
    WITH (fraudulent BOOLEAN, pan varchar(16))
    USING (CASE WHEN fraudulent THEN REDACT_CREDIT_CARD(pan)
                ELSE Null
           END);

--Attach the masking policy to the credit_cards/analyst table/role pair.
ATTACH MASKING POLICY card_number_conditional_mask ON credit_cards (credit_card_number)
```

```
USING (is_fraud, credit_card_number)
TO ROLE analyst PRIORITY 100;
```

Systemansichten für dynamische Datenmaskierung

Superuser, Benutzer mit der `sys:operator` Rolle und Benutzer mit der ACCESS SYSTEM TABLE-Berechtigung können auf die folgenden DDM-bezogenen Systemansichten zugreifen.

- SVV_MASKING_POLICY

  Verwenden Sie SVV_MASKING_POLICY, um alle Maskierungsrichtlinien anzuzeigen, die auf dem Cluster oder der Arbeitsgruppe erstellt wurden.
- SVV_ATTACHED_MASKING_POLICY

  Verwenden Sie SVV_ATTACHED_MASKING_POLICY, um alle Beziehungen und Benutzer oder Rollen mit angehängten Richtlinien an die aktuell verbundene Datenbank anzuzeigen.
- SYS_APPLIED_MASKING_POLICY_LOG

  Verwenden Sie SYS_APPLIED_MASKING_POLICY_LOG, um die Anwendung von Maskierungsrichtlinien auf Abfragen nachzuverfolgen, die auf DDM-geschützte Beziehungen verweisen.

Im Folgenden finden Sie einige Beispiele für Informationen, die Sie mithilfe von Systemansichten finden können.

```
--Select all policies associated with specific users, as opposed to roles
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE grantee_type = 'user';

--Select all policies attached to a specific user
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE grantee = 'target_grantee_name'
```

```
--Select all policies attached to a given table
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE table_name = 'target_table_name'
      AND schema_name = 'target_schema_name';

--Select the highest priority policy attachment for a given role
SELECT samp.policy_name,
       samp.priority,
       samp.grantee,
       smp.policy_expression
FROM svv_masking_policy AS smp
JOIN svv_attached_masking_policy AS samp
    ON samp.policy_name = smp.policy_name
WHERE
    samp.grantee_type = 'role' AND
    samp.policy_name = mask_get_policy_for_role_on_column(
        'target_schema_name',
        'target_table_name',
        'target_column_name',
        'target_role_name')
ORDER BY samp.priority desc
LIMIT 1;

--See which policy a specific user will see on a specific column in a given relation
SELECT samp.policy_name,
       samp.priority,
       samp.grantee,
       smp.policy_expression
FROM svv_masking_policy AS smp
JOIN svv_attached_masking_policy AS samp
    ON samp.policy_name = smp.policy_name
WHERE
    samp.grantee_type = 'role' AND
    samp.policy_name = mask_get_policy_for_user_on_column(
        'target_schema_name',
        'target_table_name',
        'target_column_name',
        'target_user_name')
ORDER BY samp.priority desc;
```

```
 --Select all policies attached to a given relation.
SELECT policy_name,
schema_name,
relation_name,
database_name
FROM sys_applied_masking_policy_log
WHERE relation_name = 'relation_name'
AND schema_name = 'schema_name';
```

Bereichsbeschränkte Berechtigungen

Mit bereichsbezogenen Berechtigungen können Sie einem Benutzer oder einer Rolle Berechtigungen für alle Objekte eines Typs innerhalb einer Datenbank oder eines Schemas gewähren. Benutzer und Rollen mit bereichsbezogenen Berechtigungen verfügen über die angegebenen Berechtigungen für alle aktuellen und future Objekte innerhalb der Datenbank oder des Schemas.

Weitere Informationen zum Anwenden bereichsbezogener Berechtigungen finden Sie unter GRANT und REVOKE.

Überlegungen zur Verwendung bereichsbezogener Berechtigungen

Beachten Sie bei der Verwendung bereichsbezogener Berechtigungen die folgenden Überlegungen:

- Sie können bereichsbezogene Berechtigungen verwenden, um einem bestimmten Benutzer oder einer bestimmten Rolle Berechtigungen für einen Datenbank- oder Schemabereich zu gewähren oder zu entziehen.
- Sie können Benutzergruppen keine bereichsbezogenen Berechtigungen gewähren.
- Durch das Gewähren oder Widerrufen bereichsbezogener Berechtigungen werden die Berechtigungen für alle derzeitigen und zukünftigen Objekte im Gültigkeitsbereich geändert.
- Bereichsbezogene Berechtigungen und Berechtigungen auf Objektebene funktionieren unabhängig voneinander. Beispielsweise behält ein Benutzer in den beiden folgenden Fällen die Berechtigungen für eine Tabelle bei.
  - Dem Benutzer wird die SELECT-Berechtigung für die Tabelle schema1.table1 und die SELECT-Bereichsberechtigung für schema1 erteilt. Dem Benutzer wurde dann die SELECT-Option für alle Tabellen im Schema schema1 entzogen. Der Benutzer behält SELECT auf schema1.table1 bei.

  - Dem Benutzer wird die SELECT-Berechtigung für die Tabelle schema1.table1 und die SELECT-Bereichsberechtigung für schema1 erteilt. Dem Benutzer wurde dann die SELECT-Option für schema1.table1 entzogen. Der Benutzer behält SELECT für schema1.table1 bei.
- Um bereichsbezogene Berechtigungen zu gewähren oder zu widerrufen, müssen Sie mindestens eines der folgenden Kriterien erfüllen:
  - Superuser.
  - Benutzer mit der Option „Gewähren“ für diese Berechtigung. Weitere Informationen zu Grant-Optionen finden Sie zum Parameter WITH GRANT OPTION unter. GRANT
- Bereichsbezogene Berechtigungen können nur Objekten für die verbundene Datenbank oder für Datenbanken, die aus einem Datashare importiert wurden, gewährt oder entzogen werden.
- Sie können bereichsbezogene Berechtigungen verwenden, um die Standardberechtigungen für eine Datenbank festzulegen, die aus einer Datenfreigabe erstellt wurde. Ein verbraucherseitiger Datashare-Benutzer, dem bereichsbezogene Berechtigungen für eine gemeinsam genutzte Datenbank gewährt werden, erhält diese Berechtigungen automatisch für jedes neue Objekt, das dem Datashare auf der Produzentenseite hinzugefügt wird.
- Produzenten können einem Datashare bereichsbezogene Berechtigungen für Objekte innerhalb eines Schemas gewähren. (Vorschau)

SQL-Referenz

Themen

Amazon-Redshift-SQL

Themen

Amazon Redshift baut auf dem Branchenstandard SQL auf, dem erweiterte Funktionen hinzugefügt wurden: für die Verwaltung sehr großer Datensätze und für die Unterstützung hochperformanter Analyse sowie für die Berichterstellung zu diesen Daten.

> **Note**
>
> Die maximal zulässige Größe für eine einzelne Amazon-Redshift-SQL-Anweisung ist 16 MB.

SQL-Funktionen, die auf dem Führungsknoten unterstützt werden

Einige Amazon-Redshift-Abfragen werden auf den Rechenknoten verteilt und ausgeführt. Andere Abfragen werden ausschließlich auf dem Führungsknoten ausgeführt.

Der Führungsknoten verteilt SQL auf den Datenverarbeitungsknoten, wenn eine Abfrage benutzererstellte Tabellen oder Systemtabellen (Tabellen mit einem STL- oder STV-Präfix und Systemansichten mit einem SVL- oder SVV-Präfix) referenziert. Eine Abfrage, die nur Katalogtabellen (Tabellen mit einem PG-Präfix wie PG_TABLE_DEF, die sich auf dem Führungsknoten befinden) oder keine Tabellen referenziert, wird ausschließlich auf dem Führungsknoten ausgeführt.

Einige SQL-Funktionen von Amazon Redshift werden nur auf dem Führungsknoten und nicht auf den Rechenknoten unterstützt. Eine Abfrage, die eine Führungsknotenfunktion verwendet, darf nur auf dem Führungsknoten und nicht auf den Rechenknoten ausgeführt werden. Andernfalls wird ein Fehler zurückgegeben.

Die Dokumentation für die einzelnen Funktionen, die ausschließlich auf dem Führungsknoten ausgeführt werden können, enthält einen Hinweis darauf, dass die Funktion einen Fehler zurückgibt, wenn sie benutzerdefinierte Tabellen oder Amazon-Redshift-Systemtabellen referenziert. Eine Liste von Funktionen, die ausschließlich auf dem Führungsknoten ausgeführt werden, finden Sie unter Exklusive Führungsknotenfunktionen.

Beispiele

Die folgenden Beispiele verwenden die TICKIT-Beispieldatenbank. Weitere Informationen zur Beispieldatenbank finden Sie unter. Beispieldatenbank

CURRENT_SCHEMA

Die Funktion CURRENT_SCHEMA ist eine exklusiven Führungsknotenfunktion. In diesem Beispiel referenziert die Abfrage keine Tabelle, d. h. sie wird ausschließlich auf dem Führungsknoten ausgeführt.

```
select current_schema();

current_schema
---------------
public
```

Im nächsten Beispiel referenziert die Abfrage eine Systemkatalogtabelle, d. h. sie wird ausschließlich auf dem Führungsknoten ausgeführt.

```
select * from pg_table_def
where schemaname = current_schema() limit 1;

 schemaname | tablename | column |   type   | encoding | distkey | sortkey | notnull
------------+-----------+--------+----------+----------+---------+---------+---------
 public     | category  | catid  | smallint | none     | t       |       1 | t
```

Im nächsten Beispiel referenziert die Abfrage eine Amazon-Redshift-Systemtabelle, die sich auf den Rechenknoten befindet, d. h. es wird ein Fehler zurückgegeben.

```
select current_schema(), userid from users;

INFO:  Function "current_schema()" not supported.
ERROR:  Specified types or functions (one per INFO message) not supported on Amazon
 Redshift tables.
```

SUBSTR

SUBSTR ist außerdem eine exklusive Führungsknotenfunktion. Im folgenden Beispiel wird die Abfrage ausschließlich auf dem Führungsknoten ausgeführt, da sie nicht auf eine Tabelle verweist.

```
SELECT SUBSTR('amazon', 5);

+--------+
| substr |
+--------+
| on     |
+--------+
```

Im folgenden Beispiel verweist die Abfrage auf eine Tabelle, die sich auf dem Datenverarbeitungsknoten befindet. Dies führt zu einem Fehler.

```
SELECT SUBSTR(catdesc, 1) FROM category LIMIT 1;

ERROR: SUBSTR() function is not supported (Hint: use SUBSTRING instead)
```

Verwenden Sie SUBSTRING, um die vorherige Abfrage erfolgreich auszuführen.

```
SELECT SUBSTRING(catdesc, 1) FROM category LIMIT 1;

+---------------------------------+
|            substring            |
+---------------------------------+
| National Basketball Association |
+---------------------------------+
```

FAKTORIELL ()

FACTORIAL () ist eine reine Leader-Node-Funktion. Im folgenden Beispiel wird die Abfrage ausschließlich auf dem Führungsknoten ausgeführt, da sie nicht auf eine Tabelle verweist.

```
SELECT FACTORIAL(5);

 factorial
-------------
 120
```

Im folgenden Beispiel verweist die Abfrage auf eine Tabelle, die sich auf dem Datenverarbeitungsknoten befindet. Dies führt bei der Ausführung mit dem Abfrage-Editor v2 zu einem Fehler.

```
create table t(a int);
insert into t values (5);
select factorial(a) from t;

ERROR: Specified types or functions (one per INFO message) not supported on Redshift
 tables.
Info: Function "factorial(bigint)" not supported.
```

Amazon Redshift und PostgreSQL

Themen

Amazon Redshift basiert auf PostgreSQL. Zwischen Amazon Redshift und PostgreSQL gibt es eine Reihe wichtiger Unterschiede, die Sie berücksichtigen müssen, wenn Sie Ihre Data-Warehouse-Anwendungen entwerfen und entwickeln.

Amazon Redshift wurde speziell für OLAP- (Online Analytic Processing) und BI-Anwendungen (Business Intelligence) entwickelt, die komplexe Abfragen für großen Datasätze erfordern. Da Amazon Redshift sehr unterschiedliche Anforderungen erfüllen soll, unterscheiden sich das spezialisierte Datenspeicher-Schema und die Abfrageausführungs-Engine erheblich von der Umsetzung in PostgreSQL. Zum Beispiel: Während OLTP-Anwendungen (Online Transaction Processing) Daten typischerweise in Zeilen speichern, speichert Amazon Redshift Daten in Spalten

und verwendet spezielle Datenkomprimierungskodierungen für optimale Speichernutzung und Speicherträger-I/O. Einige PostgreSQL-Features, die für kleinere OLTP-Verarbeitungen geeignet sind, etwa sekundäre Indizes oder effiziente Einzeilen-Datenmanipulationsoperationen, wurden ausgelassen, um die Leistung zu verbessern.

Detaillierte Informationen zu der Amazon-Redshift-Systemarchitektur für Data Warehouses finden Sie unter System- und Architekturübersicht.

PostgreSQL 9.x enthält einige Funktionen, die in Amazon Redshift nicht unterstützt werden. Darüber hinaus gibt es eine Reihe wichtiger Unterschiede zwischen Amazon-Redshift-SQL und PostgreSQL, die Sie kennen müssen. Dieser Abschnitt hebt die Unterschiede zwischen Amazon Redshift und PostgreSQL hervor und bietet eine Anleitung für die Entwicklung eines Data Warehouse, bei dem alle Vorteile der Umsetzung in Amazon-RedshiftSQL in vollem Umfang genutzt werden.

Amazon Redshift und PostgreSQL JDBC und ODBC

Da Amazon Redshift auf PostgreSQL aufbaut, haben wir bisher empfohlen, Version 8.4.703 des JDBC4-Postgresql-Treibers und Version 9.x des psqlODBC-Treibers zu verwenden. Wenn Sie diese Treiber derzeit verwenden, empfehlen wir, zu den neuen Amazon-Redshift-Treibern zu wechseln. Weitere Informationen über Treiber und das Konfigurieren von Verbindungen finden Sie unter JDBC- und ODBC-Treiber für Amazon Redshift im Amazon-Redshift-Verwaltungshandbuch.

Um clientseitige out-of-memory Fehler beim Abrufen großer Datensätze mit JDBC zu vermeiden, können Sie Ihrem Client ermöglichen, Daten stapelweise abzurufen, indem Sie den Parameter JDBC-Abrufgröße festlegen. Weitere Informationen finden Sie unter Festlegen des JDBC-Parameters für die Abrufgröße.

Der JDBC-maxRows-Parameter wird von Amazon Redshift nicht erkannt. Sie müssen stattdessen eine LIMIT-Klausel angeben, um den Ergebnissatz einzuschränken. Sie können auch eine OFFSET-Klausel verwenden, um zu einem bestimmten Startpunkt im Ergebnissatz zu springen.

Funktionen, die auf andere Weise umgesetzt sind

Viele Amazon-Redshift-SQL-Sprachelemente weisen andere Leistungsmerkmale auf und unterscheiden sich hinsichtlich der Syntax und Semantik erheblich von der entsprechenden Umsetzung in PostgreSQL.

> **Important**
>
> Beachten Sie, dass die Semantik von Elementen, die sowohl in Amazon Redshift als auch in PostgreSQL vorkommen, nicht zwingend identisch ist. Es wird dringend empfohlen, die entsprechenden Stellen im Amazon-Redshift-Entwicklerhandbuch SQL-Befehle nachzuschlagen, um die oft sehr subtilen Unterschiede zu verstehen.

Dies gilt insbesondere für den Befehl VACUUM, mit dem Tabellen bereinigt und neu organisiert werden. VACUUM funktioniert auf andere Weise und verwendet einen anderen Satz von Parametern als die PostgreSQL-Version. Weitere Informationen zur Verwendung von VACUUM in Amazon Redshift finden Sie unter Bereinigen von Tabellen.

Außerdem gibt es viele Unterschiede hinsichtlich der Datenbankverwaltung sowie der Administrationsfunktionen und -tools. Amazon Redshift bietet z. B. über eine Reihe von Systemtabellen und Ansichten, die Informationen zur Funktionsweise des Systems bereitstellen. Weitere Informationen finden Sie unter Systemtabellen und Ansichten.

Die folgende Liste enthält ein paar Beispiele für SQL-Merkmale, die in Amazon Redshift auf andere Weise umgesetzt wurden.

- CREATE TABLE

  In Amazon Redshift werden Tablespaces, die Tabellenpartitionierung und Vererbung sowie bestimmte Einschränkungen nicht unterstützt. Mit der Amazon-Redshift-Implementierung von CREATE TABLE können Sie die Algorithmen für die Sortierung und Verteilung so definieren, dass die parallele Verarbeitung optimiert wird.

  Amazon Redshift Spectrum unterstützt die Tabellenpartitionierung durch den Befehl CREATE EXTERNAL TABLE.

- ALTER TABLE

  Nur ein Teilbereich der ALTER COLUMN-Aktionen wird unterstützt.

  Mit ADD COLUMN können Sie in jeder ALTER TABLE-Anweisung nur eine Spalte hinzufügen.

- COPY

  Der Amazon-Redshift-Befehl COPY wurde stark angepasst, um das Laden von Daten aus Amazon-S3-Buckets und Amazon-DynamoDB-Tabellen zu ermöglichen und die automatische

Komprimierung zu erleichtern. Weitere Informationen finden Sie im Abschnitt Laden von Daten und in der COPY-Befehlsreferenz.

- VACUUM

  Alle Parameter für VACUUM sind anders umgesetzt. Die VACUUM-Standardoperation in PostgreSQL gewinnt z. B. einfach nur Festplattenplatz zurück und stellt ihn zur Wiederverwendung bereit. Die VACUUM-Standardoperation in Amazon Redshift ist jedoch VACUUM FULL; diese gewinnt Festplattenplatz zurück und sortiert alle Zeilen neu.
- Leerzeichen am Ende von VARCHAR-Werten werden beim Vergleich von Zeichenfolgen ignoriert. Weitere Informationen finden Sie unter Die Bedeutung von Leerzeichen am Ende.

Nicht unterstützte PostgreSQL-Funktionen

Diese PostgreSQL-Funktionen werden in Amazon Redshift nicht unterstützt.

> **Important**
>
> Beachten Sie, dass die Semantik von Elementen, die sowohl in Amazon Redshift als auch in PostgreSQL vorkommen, nicht zwingend identisch ist. Es wird dringend empfohlen, die entsprechenden Stellen im Amazon-Redshift-Entwicklerhandbuch SQL-Befehle nachzuschlagen, um die oft sehr subtilen Unterschiede zu verstehen.

- Das Abfragetool psql wird nicht unterstützt. Der Amazon-Redshift-RSQL-Client wird unterstützt.
- Tabellenpartitionierung (Partitionierung nach Intervall und Liste)
- Tablespaces
- Einschränkungen
  - Eindeutig
  - Fremdschlüssel
  - Primärschlüssel
  - Einschränkungen prüfen
  - Ausschluss-Einschränkungen

  Einschränkungen für eindeutige, Primär- und Fremdschlüssel sind zulässig, aber sie dienen lediglich Informationszwecken. Sie werden nicht vom System erzwungen, aber sie werden vom Abfrageplaner verwendet.

- Datenbankrollen
- Vererbung
- PostgreSQL-Systemspalten

  In Amazon-Redshift-SQL werden Systemspalten nicht implizit definiert. Die Namen von PostgreSQL-Systemspalten können jedoch nicht als Namen für benutzerdefinierte Spalten verwendet werden: `oid`, `tableoid`, `xmin`, `cmin`, `xmax`, `cmax` und `ctid`. Weitere Informationen finden Sie unter https://www.postgresql.org/docs/8.0/static/ddl-system-columns.html.
- Indizes
- NULLS-Klausel in Fensterfunktionen
- Sortierreihenfolgen

  Amazon Redshift unterstützt keine Gebietsschema-spezifischen oder benutzerdefinierten Sortierreihenfolgen. Siehe Sortierreihenfolgen.
- Wertausdrücke
  - Indizierte Ausdrücke
  - Array-Konstruktoren
  - Zeilen-Konstruktoren
- Auslöser
- Verwaltung externer Daten (SQL/MED)
- Tabellenfunktionen
- Als Konstanten-Tabellen verwendete VALUES-Liste
- Sequenzen
- Volltextsuche

Nicht unterstützte PostgreSQL-Datentypen

Wenn eine Abfrage versucht, einen nicht unterstützten Datentyp zu verwenden, einschließlich expliziter oder impliziter Umwandlungen, gibt sie einen Fehler aus. Manchen Abfragen, die nicht unterstützte Datentypen verwenden, werden jedoch auf dem Führungsknoten und nicht auf den Datenverarbeitungsknoten ausgeführt. Siehe SQL-Funktionen, die auf dem Führungsknoten unterstützt werden.

Eine Liste der unterstützten Datentypen finden Sie unter Datentypen.

Diese PostgreSQL-Datentypen werden in Amazon Redshift nicht unterstützt.

- Arrays
- BIT, BIT VARYING
- BYTEA
- Zusammengesetzte Typen
- Datum/Zeit
- Typen mit Aufzählungswerten
- Geometrische Typen
- HSTORE
- JSON
- Netzwerkadressen
- Numerische Typen
  - SERIAL, BIGSERIAL, SMALLSERIAL
  - MONEY
- Objekt-IDs
- Pseudo-Typen
- Intervall-Typen
- Sonderzeichen-Typen
  - „char“ – Ein interner Typ mit einem Byte (wobei der Datentyp namens „char“ in Anführungszeichen gesetzt ist).
  - name – Ein interner Typ für Objektnamen.

  Weitere Informationen zu diesen Typen finden Sie unter Sonderzeichen-Typen in der PostgreSQL-Dokumentation.
- Textsuche-Typen
- TXID_SNAPSHOT
- UUID
- XML

Nicht unterstützte PostgreSQL-Funktionen

Viel Funktionen, die nicht weggelassen wurden, haben eine andere Semantik oder werden anders verwendet. Manche der unterstützten Funktionen werden z. B. ausschließlich auf dem Führungsknoten ausgeführt. Außerdem geben manche der nicht unterstützten Funktionen keinen Fehler zurück, wenn sie auf dem Führungsknoten ausgeführt werden. Die Tatsache, dass diese Funktionen in manchen Fällen keinen Fehler zurückgeben, sollte nicht als Anzeichen dafür gewertet werden, dass die Funktion von Amazon Redshift unterstützt wird.

> **Important**
>
> Beachten Sie, dass die Semantik von Elementen, die sowohl in Amazon Redshift als auch in PostgreSQL vorkommen, nicht zwingend identisch ist. Es wird dringend empfohlen, die entsprechenden Stellen im Datenbankentwicklerhandbuch zu Amazon Redshift SQL-Befehle nachzuschlagen, um die oft sehr subtilen Unterschiede zu verstehen.

Weitere Informationen finden Sie unter SQL-Funktionen, die auf dem Führungsknoten unterstützt werden.

Diese PostgreSQL-Funktionen werden in Amazon Redshift nicht unterstützt.

- Abfragefunktionen für Zugriffsberechtigungen
- Funktionen für empfohlene Sperren
- Aggregationsfunktionen
  - STRING_AGG()
  - ARRAY_AGG()
  - EVERY()
  - XML_AGG()
  - CORR()
  - COVAR_POP()
  - COVAR_SAMP()
  - REGR_AVGX(), REGR_AVGY()
  - REGR_COUNT()
  - REGR_INTERCEPT()

  - REGR_R2()
  - REGR_SLOPE()
  - REGR_SXX(), REGR_SXY(), REGR_SYY()
- Array-Funktionen und -Operatoren
- Funktionen für die Sicherungssteuerung
- Funktionen für Kommentarinformationen
- Speicherortfunktionen für Datenbankobjekte
- Größenfunktionen für Datenbankobjekte
- Datums- und Zeitfunktionen sowie -operatoren
  - CLOCK_TIMESTAMP()
  - JUSTIFY_DAYS(), JUSTIFY_HOURS(), JUSTIFY_INTERVAL()
  - PG_SLEEP()
  - TRANSACTION_TIMESTAMP()
- Funktionen für die ENUM-Unterstützung
- Geometrische Funktionen und Operatoren
- Funktionen für den allgemeinen Dateizugriff
- IS DISTINCT FROM
- Funktionen und Operatoren für Netzwerkadressen
- Mathematische Funktionen
  - DIV()
  - SETSEED()
  - WIDTH_BUCKET()
- Datensatz-Rückgabefunktionen
  - GENERATE_SERIES()
  - GENERATE_SUBSCRIPTS()
- Intervallfunktionen und -operatoren
- Funktionen für die Wiederherstellungssteuerung
- Funktionen für Wiederherstellungsinformationen
- ROLLBACK TO SAVEPOINT-Funktion
- Abfragefunktionen für die Schema-Sichtbarkeit

- Server-Benachrichtigungsfunktionen
- Snapshot-Synchronisierungsfunktionen
- Sequenz-Bearbeitungsfunktionen
- Zeichenfolgefunktionen
  - BIT_LENGTH()
  - OVERLAY()
  - CONVERT(), CONVERT_FROM(), CONVERT_TO()
  - ENCODE()
  - FORMAT()
  - QUOTE_NULLABLE()
  - REGEXP_MATCHES()
  - REGEXP_SPLIT_TO_ARRAY()
  - REGEXP_SPLIT_TO_TABLE()
- Funktionen für Systemkataloginformationen
- Funktionen für Systeminformationen
  - CURRENT_CATALOG CURRENT_QUERY()
  - INET_CLIENT_ADDR()
  - INET_CLIENT_PORT()
  - INET_SERVER_ADDR() INET_SERVER_PORT()
  - PG_CONF_LOAD_TIME()
  - PG_IS_OTHER_TEMP_SCHEMA()
  - PG_LISTENING_CHANNELS()
  - PG_MY_TEMP_SCHEMA()
  - PG_POSTMASTER_START_TIME()
  - PG_TRIGGER_DEPTH()
  - SHOW VERSION()
- Funktionen und Operatoren für die Textsuche
- Funktionen für Transaktions-IDs und Snapshots
- Trigger-Funktionen
- XML-Funktionen

Verwenden von SQL

Themen

Die SQL-Sprache besteht aus Befehlen und Funktionen für die Arbeit mit Datenbanken und Datenbankobjekten. Außerdem definiert die Sprache Regeln für die Verwendung von Datentypen, Ausdrücken und Literalen.

Konventionen für die SQL-Referenz

In diesem Abschnitt werden die Konventionen erklärt, die für die Beschreibung der Syntax der SQL-Ausdrücke, -Befehle und -Funktionen im Abschnitt mit der SQL-Referenz verwendet werden.

Zeichen	Beschreibung
GROSSBUCHSTABEN	Wörter in Großbuchstaben sind Schlüsselwörter.
[ ]	Eckige Klammern bezeichnen optionale Argumente. Mehrere Argumente in eckigen Klammern zeigen an, dass Sie eine beliebige Anzahl der Argumente verwenden können. Argumente in eckigen Klammern, die jeweils in einer eigenen Zeile stehen, zeigen außerdem an, dass der Amazon-Redshift-Parser die Argumente in der Reihenfolge erwartet, in der sie in der Syntax aufgelistet sind. Ein Beispiel finden Sie unter SELECT.
{ }	Geschweifte Klammern zeigen an, dass Sie nur eines der Argumente verwenden können, die innerhalb der Klammern stehen.
|	Pipe-Zeichen zeigen an, dass Sie zwischen den Argumenten wählen können.
Kursivschrift	Wörter in Kursivschrift zeigen Platzhalter an. Sie müssen das kursiv formatier te Wort durch den entsprechenden Wert ersetzen.

Zeichen	Beschreibung
. . .	Auslassungspunkte zeigen an, dass Sie das Element davor wiederholen können.
'	Wörter in einfachen Anführungszeichen müssen zusammen mit den Anführungszeichen verwendet werden.

Grundelemente

Themen

- Namen und Kennungen
- Literale
- Null-Werte
- Datentypen
- Sortierreihenfolgen

In diesem Abschnitt werden die Regeln für die Arbeit mit Datenbankobjekt-Namen, Literalen, Null-Werten und Datentypen beschrieben.

Namen und Kennungen

Namen bezeichnen Datenbankobjekte, u. a. Tabellen und Spalten sowie Benutzer und Passwörter. Die Begriffe Name und Kennung können synonym verwendet werden. Es gibt zwei Arten von Kennungen, Standard-Kennungen und abgegrenzte Kennungen in Anführungszeichen. Kennungen dürfen nur aus druckbaren UTF-8-Zeichen bestehen. Bei ASCII-Zeichen in Standard-Kennungen und abgegrenzten Kennungen wird in der Datenbank nicht zwischen Groß- und Kleinschreibung unterschieden; Großbuchstaben werden wie Kleinbuchstaben behandelt. In Abfrageergebnissen werden Spaltennamen standardmäßig in Kleinbuchstaben zurückgegeben. Um Spaltennamen in Großbuchstaben zurückzugeben, legen Sie den describe_field_name_in_uppercase-Konfigurationsparameter auf **`true`** fest.

Standard-Kennungen

Standard-Kennungen in SQL sind einer Reihe von Regeln unterworfen und müssen:

- Mit einem alphabetischen ASCII-Zeichen mit einer Länge von einem Byte oder einem Unterstrich beginnen, oder mit einem UTF-8-Zeichen mit einer Länge von zwei bis vier Bytes
- Die nachfolgenden Zeichen können alphanumerische ASCII-Zeichen mit einer Länge von einem Byte, Unterstriche oder Dollar-Zeichen sein, oder UTF-8-Zeichen mit einer Länge von zwei bis vier Bytes
- Zwischen 1 und 127 Byte lang sein, ohne die Anführungszeichen für abgegrenzte Kennungen.
- Dürfen keine Anführungszeichen oder Leerzeichen enthalten
- Dürfen nicht ein SQL-Schlüsselwort sein

Abgegrenzte Kennungen

Abgegrenzte Kennungen (bzw. Kennungen in Anführungszeichen) beginnen und enden mit einem doppelten Anführungszeichen ("). Wenn Sie eine abgegrenzte Kennungen verwenden, müssen Sie die doppelten Anführungszeichen bei jeder Referenz auf das Objekt verwenden. Die Kennung kann alle druckbaren UTF-8-Standardzeichen enthalten, mit Ausnahme der Anführungszeichen. D. h. Sie können Spalten- oder Tabellennamen erstellen, die Zeichen enthalten, die sonst nicht zulässig wären, z. B. Leerzeichen oder das Prozentsymbol.

Bei ASCII-Zeichen in abgegrenzten Kennungen wird nicht zwischen Groß- und Kleinschreibung unterschieden; Großbuchstaben werden wie Kleinbuchstaben behandelt. Sie können ein doppeltes Anführungszeichen in einer Zeichenfolge verwenden, indem Sie ein weiteres Anführungszeichen voranstellen.

Bezeichner mit Groß-/Kleinschreibung

Bezeichner mit Groß- und Kleinschreibung (auch als Bezeichner mit gemischter Groß-/Kleinschreibung bekannt) können sowohl Groß- als auch Kleinbuchstaben enthalten.
Um Bezeichner mit Groß-/Kleinschreibung zu verwenden, können Sie die Konfiguration `enable_case_sensitive_identifier` auf `true` festlegen. Sie können diese Konfiguration für den Cluster oder für eine Sitzung festlegen. Weitere Informationen finden Sie unter Standardparameterwerte im Amazon-Redshift-Verwaltungshandbuch und unter enable_case_sensitive_identifier.

Systemspaltennamen

Die folgenden Namen von PostgreSQL-Systemspalten können nicht als Namen für benutzerdefinierte Spalten verwendet werden. Weitere Informationen finden Sie unter https://www.postgresql.org/docs/8.0/static/ddl-system-columns.html.

- `oid`
- `tableoid`
- `xmin`
- `cmin`
- `xmax`
- `cmax`
- `ctid`

Beispiele

Die folgende Tabelle enthält ein paar Beispiele für abgegrenzte Kennungen, die jeweils resultierende Ausgabe und eine Erklärung:

Syntax	Ergebnis	Erklärung
"Gruppe"	Gruppe	GROUP ist ein reserviertes Wort, d. h. es kann in einer Kennung nur mit doppelten Anführungszeichen verwendet werden.
"""WHERE"""	"where"	WHERE ist ebenfalls ein reserviertes Wort. Sie können Anführungszeichen in der Zeichenfolge verwenden, indem Sie jedem doppelte Anführungszeichen ein zusätzlic hes doppeltes Anführungszeichen als Escape-Zeichen voranstellen.
"This name"	this name	Falls Leerzeichen erhalten bleiben sollen, müssen doppelte Anführungszeichen verwendet werden.
"This ""IS IT"""	this "is it"	Den Anführungszeichen um IS IT muss jeweils ein eigenes Anführungszeichen vorangestellt werden, sodass sie als Bestandteil des Namens interpretiert werden.

So erstellen Sie eine Tabelle mit dem Namen „group” und mit einer Spalte mit dem Namen „this "is it"”:

```
create table "group" (
```

```
"This ""IS IT""" char(10));
```

Die folgenden Abfragen geben dasselbe Ergebnis zurück:

```
select "This ""IS IT"""
from "group";

this "is it"
--------------
(0 rows)
```

```
select "this ""is it"""
from "group";

this "is it"
--------------
(0 rows)
```

Die folgende vollständig qualifizierte `table.column`-Syntax gibt ebenfalls dasselbe Ergebnis zurück:

```
select "group"."this ""is it"""
from "group";

this "is it"
--------------
(0 rows)
```

Der folgende Befehl CREATE TABLE erstellt eine Tabelle mit einem Schrägstrich in einem Spaltennamen:

```
create table if not exists city_slash_id(
                  "city/id" integer not null,
                  state char(2) not null);
```

Literale

Ein Literal oder eine Konstante ist ein fester Datenwert, bestehend aus einer Zeichenfolge oder einer numerischen Konstante. Amazon Redshift unterstützt verschiedene Arten von Literalen, darunter:

- Numerische Literale für Ganzzahlen, Dezimalzahlen und Gleitkommazahlen. Weitere Informationen finden Sie unter Ganzzahl- und Gleitkommaliterale.

- Zeichenliterale, auch als Strings, Zeichenfolgen oder Zeichenkonstanten bezeichnet
- Datums-/Uhrzeitliterale und Intervalliterale, die in Datum-/Uhrzeit-Datentypen verwendet werden. Weitere Informationen erhalten Sie unter Datums-, Zeit- und Zeitstempelliterale und Datentypen und Literale für Intervalle.

Null-Werte

Wenn in einer Zeile eine Spalte fehlt, unbekannt oder ungültig ist, stellt sie einen Null-Wert dar bzw. enthält einen Null-Wert. Null-Werte können in allen Datentypfeldern ohne Primärschlüsseleinschränkung oder NOT NULL-Einschränkung auftreten. Ein Null-Wert ist nicht identisch mit dem Zahlenwert „Null” oder mit einer leeren Zeichenfolge.

Jeder arithmetische Ausdruck, der einen Null-Wert enthält, ergibt auch immer einen Null-Wert. Alle Operatoren mit Ausnahme der Verkettung Geben einen Null-Wert zurück, wenn ihnen ein Null-Wert-Argument oder -Operand übergeben wird.

Sie können diese auf Null-Werte überprüfen, indem Sie die Vergleichsbedingungen IS NULL und IS NOT NULL verwenden. Da Null-Werte das Fehlen von Daten bezeichnen, ist ein Null-Wert nie gleich oder ungleich einem anderen (Null-)Wert.

Datentypen

Themen

- Multibyte-Zeichen
- Numerische Typen
- Zeichentypen
- Datum-/Uhrzeittypen
- Typ BOOLEAN
- HLLSKETCH-Typ
- Typ SUPER
- Typ VARBYTE
- Kompatibilität von Typen und Umwandlung zwischen Typen

Jeder Wert, den Amazon Redshift speichert oder abruft, gehört zu einem Datentyp mit einem festen Satz von Eigenschaften. Datentypen werden deklariert, wenn Tabellen erstellt werden. Eine Datentyp beschränkt die Gruppe von Werten, den eine Spalte oder ein Argument enthalten kann.

In der folgenden Tabelle sind alle Datentypen aufgeführt, die Sie in Amazon-Redshift-Tabellen verwenden können.

Datentyp	Aliasnamen	Beschreibung
SMALLINT	INT2	2-Byte-Ganzzahl mit Vorzeichen
INTEGER	INT, INT4	4-Byte-Ganzzahl mit Vorzeichen
BIGINT	INT8	8-Byte-Ganzzahl mit Vorzeichen
DECIMAL	NUMERIC	Genauer Zahlenwert mit wählbarer Genauigkeit
REAL	FLOAT4	Gleitkommazahl mit einfacher Genauigkeit
DOUBLE PRECISION	FLOAT8, FLOAT	Gleitkommazahl mit doppelter Genauigkeit
CHAR	CHARACTER, NCHAR, BPCHAR	Zeichenfolge mit fester Länge
VARCHAR	CHARACTER VARYING, NVARCHAR, TEXT	Zeichenfolge mit variabler Länge und benutzerdefinierte m Grenzwert
DATUM		Kalenderdatum (Jahr, Monat, Tag)
TIME	TIME WITHOUT TIME ZONE	Uhrzeit
TIMETZ	TIME WITH TIME ZONE	Uhrzeit mit Zeitzone
TIMESTAMP (ZEITSTEMPEL)	TIMESTAMP WITHOUT TIME ZONE	Datum und Uhrzeit (ohne Zeitzone)

Datentyp	Aliasnamen	Beschreibung
TIMESTAMPTZ	TIMESTAMP WITH TIME ZONE	Datum und Uhrzeit (mit Zeitzone)
INTERVAL YEAR TO MONTH		Zeitdauer in der Reihenfolge von Jahr zu Monat
INTERVAL DAY TO SECOND		Zeitdauer in der Reihenfolge von Tag bis zur zweiten
BOOLEAN	BOOL	Logischer/Boolescher Wert (wahr/falsch)
HLLSKETCH		Typ, der für HyperLogLog Skizzen verwendet wird.
SUPER		Ein übergeordneter Datentyp, der alle skalaren Typen von Amazon Redshift umfasst, einschließlich komplexer Typen wie ARRAY und STRUCTS.
VARBYTE	VARBINARY, BINARY VARYING	Binärwert mit variabler Länge
GEOMETRY		Geodaten
GEOGRAPHY		Geodaten

Note

Weitere Informationen über nicht unterstützte Datentypen wie „char“ (dieser Datentyp steht in Anführungszeichen) finden Sie unter Nicht unterstützte PostgreSQL-Datentypen.

Multibyte-Zeichen

Der Datentyp VARCHAR unterstützt Multibyte-UTF-8-Zeichen mit einer Länge von bis zu vier Bytes. Zeichen mit einer Länge von fünf Bytes oder mehr werden nicht unterstützt. Sie berechnen die Größe einer VARCHAR-Spalte, die Multibyte-Zeichen enthält, indem Sie die Anzahl der Zeichen mit der Anzahl der Bytes pro Zeichen multiplizieren. Wenn eine Zeichenfolge z. B. vier chinesischen Zeichen enthält und jedes Zeichen drei Bytes lang ist, dann ist eine VARCHAR(12)-Spalte erforderlich, um die Zeichenfolge zu speichern.

Der Datentyp VARCHAR bietet keine Unterstützung für die folgenden ungültigen UTF-8-Codepunkte:

`0xD800 – 0xDFFF` (Bytesequenzen: `ED A0 80–ED BF BF`)

Der Datentyp CHAR bietet keine Unterstützung für Multibyte-Zeichen.

Numerische Typen

Themen

Numerische Datentypen sind Ganzzahlen, Dezimalzahlen und Gleitkommazahlen.

Ganzzahl-Typen

Verwenden Sie die Datentypen SMALLINT, INTEGER und BIGINT, um Ganzzahlen aus verschiedenen Bereichen zu speichern. Sie können für die einzelnen Typen keine Werte außerhalb des zulässigen Bereichs speichern.

Name	Speicher	Bereich
SMALLINT oder INT2	2 Bytes	-32768 bis +32767

Name	Speicher	Bereich
INTEGER, INT oder INT4	4 Bytes	-2147483648 bis +2147483647
BIGINT oder INT8	8 Bytes	-9223372036854775808 bis +9223372036854775807

Typ DECIMAL oder NUMERIC

Verwenden Sie den Datentyp DECIMAL oder NUMERIC, um Werte mit benutzerdefinierter Genauigkeit zu speichern. Die Schlüsselwörter DECIMAL und NUMERIC können synonym verwendet werden. In diesem Dokument wird der Begriff dezimal für diesen Datentyp bevorzugt. Der Begriff numerisch wird in der Regel als Oberbegriff für Ganzzahl-, Dezimalzahl- und Gleitkommazahl-Datentypen verwendet.

Speicher	Bereich
Variabel, bis zu 128 Bits für unkomprimierte DECIMAL-Typen	128-Bit-Ganzzahlen mit Vorzeichen und einer Genauigkeit von bis zu 38 Stellen

Definieren Sie eine DECIMAL-Spalte in einer Tabelle, indem Sie die Genauigkeit (precision) und die Dezimalstellen bzw. Nachkommastellen (scale) angeben:

```
decimal(precision, scale)
```

precision

Die Anzahl aller signifikanten Stellen im gesamten Wert: die Anzahl der Stellen auf beiden Seiten des Dezimaltrennzeichens. Die Zahl `48.2891` hat z. B. eine Genauigkeit von 6 und 4 Dezimalstellen. Wenn Sie nichts angeben, wird standardmäßig eine Genauigkeit von 18 verwendet. Die maximale Genauigkeit ist 38.

Wenn die Anzahl der Stellen auf der linken Seite des Dezimaltrennzeichens in einem Eingabewert die Genauigkeit der Spalte minus der Dezimalstellen überschreitet, kann der Wert nicht in die Spalte kopiert werden. Diese Regel gilt für alle Werte, die nicht innerhalb des Bereichs der

Spaltendefinition liegen. Der zulässige Wertebereich für eine numeric(5,2)-Spalte erstreckt sich z. B. von -999.99 bis 999.99.

scale

Die Anzahl aller Dezimalstellen im Nachkommabereich des Wertes bzw. die Anzahl der Stellen auf der rechten Seite des Dezimaltrennzeichens. Ganzzahlen haben keine Dezimalstellen. In einer Spaltenspezifikation muss der Wert für die Dezimalstellen kleiner oder gleich dem Wert für die Genauigkeit sein. Wenn Sie nichts angeben, werden standardmäßig 0 Dezimalstellen verwendet. Es sind maximal 37 Dezimalstellen zulässig.

Wenn ein Eingabewert, der in eine Tabelle geladen wird, mehr Dezimalstellen aufweist, als für die Spalte zulässig sind, wird der Wert auf die angegebene Dezimalstelle gerundet. Die Spalte PRICEPAID in der Tabelle SALES ist z. B. eine DECIMAL(8,2)-Spalte. Wenn ein DECIMAL(8,4)-Wert in die Spalte PRICEPAID eingefügt wird, wird der Wert auf 2 Dezimalstellen gerundet.

```
insert into sales
values (0, 8, 1, 1, 2000, 14, 5, 4323.8951, 11.00, null);

select pricepaid, salesid from sales where salesid=0;

pricepaid | salesid
-----------+---------
4323.90 |       0
(1 row)
```

Die Ergebnisse expliziter Umwandlungen von Werten, aus der Tabelle ausgewählt wurden, werden jedoch nicht gerundet.

> **Note**
>
> Der maximale positive Wert, der in eine DECIMAL(19,0)-Spalte eingefügt werden kann, ist 9223372036854775807 (2^{63} -1). Die maximale negative Wert ist -9223372036854775807. Wenn versucht wird, den Wert 9999999999999999999 (19 mal die Ziffer Neun) einzufügen, wird ein Überlauffehler verursacht. Unabhängig von der Position des Dezimaltrennzeichens ist die längste Zeichenkette, die Amazon Redshift als DECIMAL-Zahl darstellen kann 9223372036854775807. Der größte Wert, der in eine DECIMAL(19,18)-Spalte geladen werden kann, ist z. B. 9.223372036854775807.

Diese Regeln gelten, da DECIMAL-Werte mit einer Genauigkeit von 19 oder weniger signifikanten Stellen intern als 8-Byte-Ganzzahlen gespeichert werden, während DECIMAL-Werte mit einer Genauigkeit von 20 bis 38 signifikanten Stellen als 16-Byte-Ganzzahlen gespeichert werden.

Hinweise zur Verwendung von 128-Bit-DECIMAL- oder -NUMERIC-Spalten

Weisen Sie DECIMAL-Spalten nur dann maximale Genauigkeit zu, wenn Sie sicher sind, dass Ihre Anwendung diese Präzision erfordert. 128-Bit-Werte belegen doppelt so viel Speicherplatz wie 64-Bit-Werte und können zu langsameren Ausführungszeiten von Abfragen führen.

Gleitkommazahl-Typen

Verwenden Sie die Datentypen REAL oder DOUBLE PRECISION, um numerische Werte mit variabler Genauigkeit zu speichern. Diese Typen sind ungenaue Typen, d. h. manche Werte werden als Annährungen gespeichert, so dass bei der Speicherung und Rückgabe eines bestimmten Wertes leichte Abweichungen auftreten können. Wenn Sie auf genaue Speicherungen und Berechnungen zurückgreifen müssen (z. B. bei Geldbeträgen), verwenden Sie den Datentyp DECIMAL.

REAL steht für das Gleitkommaformat mit einfacher Genauigkeit gemäß dem IEEE-Standard 754 für binäre Gleitkommaarithmetik. Es hat eine Genauigkeit von etwa 6 Ziffern und einen Bereich von etwa 1E-37 bis 1E+37. Sie können diesen Datentyp auch als FLOAT4 angeben.

DOUBLE PRECISION steht für das Gleitkommaformat mit doppelter Genauigkeit gemäß dem IEEE-Standard 754 für binäre Gleitkommaarithmetik. Es hat eine Genauigkeit von etwa 15 Ziffern und einen Bereich von etwa 1E-307 bis 1E+308. Sie können diesen Datentyp auch als FLOAT oder FLOAT8 angeben.

Zusätzlich zu den gewöhnlichen numerischen Werten haben Gleitkommatypen mehrere spezielle Werte. Verwenden Sie einfache Anführungszeichen bei diesen Werten, wenn Sie sie in SQL verwenden:

- `NaN` – not-a-number
- `Infinity` – unendlich
- `-Infinity` – negativ unendlich

Um beispielsweise eine Tabellenspalte not-a-number `day_charge` einzufügen, `customer_activity` führen Sie den folgenden SQL-Befehl aus:

```
insert into customer_activity(day_charge) values('NaN');
```

Berechnungen mit numerischen Werten

In diesem Kontext bezieht sich der Begriff Berechnungen auf binäre mathematische Operationen: Addition, Subtraktion, Multiplikation und Division. In diesem Abschnitt werden die erwarteten Ausgabetypen dieser Operationen beschrieben sowie die spezielle Formel, die verwendet wird, um die Genauigkeit und die Dezimalstellen zu ermitteln, wenn DECIMAL-Datentypen involviert sind.

Wenn bei der Verarbeitung von Abfragen numerische Werte berechnet werden, kann es vorkommen, dass eine Berechnung nicht möglich ist und die Abfrage einen numerischen Überlauffehler zurückgibt. Außerdem können Fälle auftreten, in denen die Dezimalstellen berechneter Werte variieren bzw. nicht den Erwartungen entsprechen. Bei manchen Operationen ist es möglich, diese Probleme durch explizite Umwandlungen (Typerweiterung) oder Amazon-Redshift-Konfigurationsparameter zu umgehen.

Weitere Informationen zu den Ergebnissen ähnlicher Berechnungen mit SQL-Funktionen finden Sie unter Aggregationsfunktionen.

Ausgabetypen für Berechnungen

In der folgenden Tabelle werden die erwarteten Ausgabewerte für die Operationen Addition, Subtraktion, Multiplikation und Division unter Berücksichtigung der in Amazon Redshift unterstützten numerischen Datentypen aufgeführt. Die erste Spalte links in der Tabelle enthält dabei den ersten Operanden und die oberste Zeile den zweiten Operanden der Berechnung.

	INT2	INT4	INT8	DECIMAL	FLOAT4	FLOAT8
INT2	INT2	INT4	INT8	DECIMAL	FLOAT8	FLOAT8
INT4	INT4	INT4	INT8	DECIMAL	FLOAT8	FLOAT8
INT8	INT8	INT8	INT8	DECIMAL	FLOAT8	FLOAT8
DECIMAL	DECIMAL	DECIMAL	DECIMAL	DECIMAL	FLOAT8	FLOAT8
FLOAT4	FLOAT8	FLOAT8	FLOAT8	FLOAT8	FLOAT4	FLOAT8
FLOAT8	FLOAT8	FLOAT8	FLOAT8	FLOAT8	FLOAT8	FLOAT8

Genauigkeit und Dezimalstellen der berechneten DECIMAL-Ergebnisse

In der folgenden Tabelle werden die Regeln für die Berechnung der Genauigkeit und der Dezimalstellen zusammengefasst, wenn mathematische Operationen DECIMAL-Ergebnisse ausgeben. In dieser Tabelle stellen `p1` und `s1` die Genauigkeit und die Dezimalstellen des ersten Operanden einer Berechnung und `p2` und `s2` die Genauigkeit und die Dezimalstellen des zweiten Operanden dar. (Unabhängig von diesen Berechnungen ist die maximale Genauigkeit eines Ergebnisses 38 und der maximale Wert für die Dezimalstellen 38.)

Operation	Genauigkeit und Dezimalstellen in Ergebnissen
+ oder -	Skalieren = `max(s1,s2)` Genauigkeit = `max(p1-s1,p2-s2)+1+scale`
*	Skalieren = `s1+s2` Genauigkeit = `p1+p2+1`
/	Skalieren = `max(4,s1+p2-s2+1)` Genauigkeit = `p1-s1+ s2+scale`

Die Spalten PRICEPAID und COMMISSION in der Tabelle SALES sind z. B. DECIMAL(8,2)-Spalten. Wenn Sie PRICEPAID durch COMMISSION dividieren (oder umgekehrt), sieht die Formel wie folgt aus:

```
Precision = 8-2 + 2 + max(4,2+8-2+1)
= 6 + 2 + 9 = 17

Scale = max(4,2+8-2+1) = 9

Result = DECIMAL(17,9)
```

Die folgende Berechnung stellt die allgemeine Regel für die Berechnung der Genauigkeit und der Dezimalstellen in Ergebnissen von Operationen dar, die mit DECIMAL-Werten sowie mit Satzoperatoren wie UNION, INTERSECT und EXCEPT oder Funktionen wie COALESCE und DECODE durchgeführt werden:

```
Scale = max(s1,s2)
Precision = min(max(p1-s1,p2-s2)+scale,19)
```

Eine DEC1-Tabelle mit einer DECIMAL(7,2)-Spalte wird z. B. mit einer DEC2-Tabelle mit einer DECIMAL(15,3)-Spalte verbunden, um eine DEC3-Tabelle zu erstellen. Das Schema von DEC3 zeigt, dass die Spalte zu einer NUMERIC(15,3)-Spalte wird.

```
create table dec3 as select * from dec1 union select * from dec2;
```

Ergebnis

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'dec3';

column |     type      | encoding | distkey | sortkey
-------+---------------+----------+---------+---------
c1     | numeric(15,3) | none     | f       | 0
```

Im Beispiel oben wird die Formel wie folgt angewendet:

```
Precision = min(max(7-2,15-3) + max(2,3), 19)
= 12 + 3 = 15

Scale = max(2,3) = 3

Result = DECIMAL(15,3)
```

Hinweise für Divisionsoperationen

Bei Divisionsoperationen geben divide-by-zero Bedingungen Fehler zurück.

Für Dezimalstellen gilt ein Grenzwert von 100, nachdem die Genauigkeit und die Dezimalstellen berechnet wurden. Wenn im Ergebnis mehr als 100 Dezimalstellen berechnet wurden, wird das Ergebnis der Division wie folgt skaliert:

- Genauigkeit = `precision - (scale - max_scale)`
- Skalieren = `max_scale`

Wenn die berechnete Genauigkeit über dem maximalen Wert für die Genauigkeit (38) liegt, wird die Genauigkeit auf 38 reduziert, und für die Dezimalstellen wird die folgende Formel angewendet: `max((38 + scale - precision), min(4, 100))`

Überlaufbedingungen

Der Überlauf wird bei allen numerischen Berechnungen geprüft. DECIMAL-Daten mit einer Genauigkeit von 19 oder weniger werden als 64-Bit-Ganzzahlen gespeichert. DECIMAL-Daten mit einer Genauigkeit größer als 19 werden als 128-Bit-Ganzzahlen gespeichert. Die maximale Genauigkeit für alle DECIMAL-Werte beträgt 38, und es sind maximal 37 Dezimalstellen zulässig. Überlauffehler treten auf, wenn ein Wert diese Grenzwerte überschreitet; diese gelten sowohl für Zwischenergebnissätze als auch für Endergebnissätze:

- Ergebnisse von expliziten Umwandlungen in Überlauffehlern während der Laufzeit, wenn bestimmte Datenwerte nicht die Genauigkeit oder die Dezimalstellen aufweisen, die in der Umwandlungsfunktion angegeben sind. Sie können z. B. nicht alle Werte in der Spalte PRICEPAID in der Tabelle SALES (eine DECIMAL(8,2)-Spalte) umwandeln und ein DECIMAL(7,3)-Ergebnis zurückgeben:

  ```
  select pricepaid::decimal(7,3) from sales;
  ERROR:  Numeric data overflow (result precision)
  ```

  Dieser Fehler tritt auf, weil manche der größeren Werte in der Spalte PRICEPAID nicht umgewandelt werden können.

- Multiplikationsoperationen produzieren Ergebnisse, bei denen sich die Anzahl der Dezimalstellen aus der Summe der Dezimalstellen der einzelnen Operanden ergeben. Wenn beide Operanden z. B. 4 Dezimalstellen haben, hat das Ergebnis 8 Dezimalstellen, d. h. es bleiben nur 10 Stellen auf der linken Seite des Dezimaltrennzeichens übrig. Es kann daher relativ schnell passieren, dass Überlaufbedingungen bei der Multiplikation zweier großer Zahlen auftreten, die jeweils eine nicht unerheblich Anzahl von Dezimalstellen aufweisen.

  Der folgende Code führt beispielsweise zu einem Überlauf-Fehler:

  ```
  SELECT CAST(1 AS DECIMAL(38, 20)) * CAST(10 AS DECIMAL(38, 20));
  ERROR: 128 bit numeric data overflow (multiplication)
  ```

Sie können den Überlauf-Fehler umgehen, indem Sie Division statt Multiplikation benutzen. Verwenden Sie das folgende Beispiel, um durch 1 geteilt durch den ursprünglichen Divisor zu dividieren.

```
SELECT CAST(1 AS DECIMAL(38, 20)) / (1 / CAST(10 AS DECIMAL(38, 20)));
+----------+
| ?column? |
+----------+
| 10       |
+----------+
```

Numerische Berechnungen mit den Typen INTEGER und DECIMAL

Wenn einer der Operanden in einer Berechnung zum Datentyp INTEGER gehört und der andere ein DECIMAL-Operand ist, dann wird der INTEGER-Operand implizit in einen DECIMAL-Wert umgewandelt:

- INT2 (SMALLINT) wird in DECIMAL(5,0) umgewandelt.
- INT4 (INTEGER) wird in DECIMAL(10,0) umgewandelt.
- INT8 (BIGINT) wird in DECIMAL(19,0) umgewandelt.

Wenn Sie z. B. SALES.COMMISSION, eine DECIMAL(8,2)-Spalte, mit SALES.QTYSOLD, einer SMALLINT-Spalte multiplizieren, wird diese Berechnung umgewandelt in:

```
DECIMAL(8,2) * DECIMAL(5,0)
```

Ganzzahl- und Gleitkommaliterale

Literale oder Konstanten, die Zahlen darstellen, können Ganzzahlen oder Gleitkommazahlen sein.

Ganzzahlliterale

Eine Ganzzahlkonstante ist eine Folge der Ziffern 0-9 mit einem optionalen positiven (+) oder negativen (-) Vorzeichen (das den Ziffern vorangestellt wird).

Syntax

```
[ + | - ] digit ...
```

Beispiele

Zulässige Ganzzahlen sind z. B.:

```
23
-555
+17
```

Gleitkommaliterale

Gleitkommaliterale (auch als Dezimal-, numerische oder Bruchliterale bezeichnet) sind Folgen von Ziffern, in denen auch ein Dezimaltrennzeichen sowie optional ein Exponent (e) enthalten sein kann.

Syntax

```
[ + | - ] digit ... [ . ] [ digit ...]
[ e | E [ + | - ] digit ... ]
```

Argumente

e | E

e oder E zeigt an, dass die Zahl in wissenschaftlicher Notation angegeben wird.

Beispiele

Zulässige Gleitkommaliterale sind z. B.:

```
3.14159
-37.
2.0e19
-2E-19
```

Beispiele mit numerischen Typen

CREATE TABLE-Anweisung

Die folgende CREATE TABLE-Anweisung illustriert die Deklaration von verschiedenen numerischen Datentypen:

```
create table film (
```

```
film_id integer,
language_id smallint,
original_language_id smallint,
rental_duration smallint default 3,
rental_rate numeric(4,2) default 4.99,
length smallint,
replacement_cost real default 25.00);
```

Der Versuch, eine Ganzzahl einzufügen, die außerhalb des Bereichs liegt

Im folgenden Beispiel wird versucht, den Wert 33000 in eine SMALLINT-Spalte einzufügen.

```
insert into film(language_id) values(33000);
```

Der Bereich für SMALLINT liegt zwischen -32768 und +32767, d. h. Amazon Redshift gibt einen Fehler aus.

```
An error occurred when executing the SQL command:
insert into film(language_id) values(33000)

ERROR: smallint out of range [SQL State=22003]
```

Einfügen eines Dezimalwerts in eine Ganzzahlspalte

Im folgenden Beispiel wird ein Dezimalwert in eine INT-Spalte eingefügt.

```
insert into film(language_id) values(1.5);
```

Dieser Wert wird eingefügt, aber auf den ganzzahligen Wert 2 aufgerundet.

Erfolgreiches Einfügen einer Dezimalzahl, da die Dezimalstellen gerundet werden

Im folgenden Beispiel wird ein Dezimalwert mit einer höheren Genauigkeit als die Spalte eingefügt.

```
insert into film(rental_rate) values(35.512);
```

In diesem Fall wird der Wert `35.51` in die Spalte eingefügt.

Der Versuch, einen Dezimalwert einzufügen, die außerhalb des Bereichs liegt

In diesem Fall liegt der Wert `350.10` außerhalb des Bereichs. Die Anzahl der Stellen für Werte in DECIMAL-Spalten entspricht der Genauigkeit der Spalte minus ihrer Dezimalstellen (4 minus 2 für die

Spalte RENTAL_RATE). In anderen Worten: Der zulässige Wertebereich für eine DECIMAL(4,2)-Spalte erstreckt sich von -99.99 bis 99.99.

```
insert into film(rental_rate) values (350.10);
ERROR:  numeric field overflow
DETAIL:  The absolute value is greater than or equal to 10^2 for field with precision
 4, scale 2.
```

Einfügen von Werten mit variabler Genauigkeit in eine REAL-Spalte

Im folgenden Beispiel werden Werte mit variabler Genauigkeit in eine REAL-Spalte eingefügt.

```
insert into film(replacement_cost) values(1999999.99);

insert into film(replacement_cost) values(1999.99);

select replacement_cost from film;

+------------------+
| replacement_cost |
+------------------+
| 2000000          |
| 1999.99          |
+------------------+
```

Der Wert 1999999.99 wird in 2000000 konvertiert, um die Genauigkeit der Spaltenanforderung REAL zu erfüllen. Der Wert 1999.99 wird unverändert geladen.

Zeichentypen

Themen

- Speicherung und Bereiche
- CHAR oder CHARACTER
- VARCHAR oder CHARACTER VARYING
- NCHAR- und NVARCHAR-Typen
- TEXT- und BPCHAR-Typen
- Die Bedeutung von Leerzeichen am Ende
- Beispiele mit Zeichentypen

Zu den Zeichendatentypen gehören die Typen CHAR (character) und VARCHAR (character varying).

Speicherung und Bereiche

Die Datentypen CHAR und VARCHAR werden in Bezug auf ihre Bytes definiert, nicht über die Zeichen. Eine CHAR-Spalte kann nur Einzelbyte-Zeichen enthalten, d. h. eine CHAR(10)-Spalte kann eine Zeichenfolge mit einer maximalen Länge von 10 Bytes enthalten. Eine VARCHAR-Spalte kann Multibyte-Zeichen bis zu einer maximalen Länge von vier Bytes pro Zeichen enthalten. Eine VARCHAR(12)-Spalte kann z. B. 12 Einzelbyte-Zeichen, 6 Zeichen mit einer Länge von je 2 Bytes, 4 Zeichen mit einer Länge von je 3 Bytes oder 3 Zeichen mit einer Länge von je 4 Bytes enthalten.

Name	Speicher	Bereich (Breite der Spalte)
CHAR, CHARACTER oder NCHAR	Länge der Zeichenfolge einschließlich der Leerzeichen am Ende (falls vorhanden)	4096 Bytes
VARCHAR, CHARACTER VARYING oder NVARCHAR	4 Bytes + alle Bytes für Zeichen, wobei jedes Zeichen zwischen 1 und 4 Bytes lang ist.	65535 Bytes (64 K -1)
BPCHAR	Konvertiert in CHAR(256) mit fester Länge	256 Byte
TEXT	Konvertiert in VARCHAR(256)	260 Bytes

> **Note**
>
> Die CREATE TABLE-Syntax unterstützt das Schlüsselwort MAX für Zeichendatentypen. Beispiel:

```
create table test(col1 varchar(max));
```

Die MAX-Einstellung definiert die Breite einer Spalte als 4096 Bytes für CHAR oder 65535 Bytes für VARCHAR.

CHAR oder CHARACTER

Verwenden Sie eine CHAR- oder CHARACTER-Spalte, um Zeichenfolgen mit einer festen Länge zu speichern. Diese Zeichenfolgen werden mit Leerzeichen aufgefüllt, sodass eine CHAR(10)-Spalte immer 10 Bytes im Speicher belegt.

```
char(10)
```

Eine CHAR-Spalte ohne Längenangabe wird als CHAR(1)-Spalte umgesetzt.

VARCHAR oder CHARACTER VARYING

Verwenden Sie eine VARCHAR- oder CHARACTER VARYING-Spalte, um Zeichenfolgen mit einer variablen Länge und einem festen Grenzwert zu speichern. Diese Zeichenfolgen werden mit Leerzeichen aufgefüllt, d. h. eine VARCHAR(120)-Spalte besteht aus jeweils maximal 120 Einzelbyte-Zeichen, 60 Zeichen mit einer Länge von je 2 Bytes, 40 Zeichen mit einer Länge von je 3 Bytes oder 30 Zeichen mit einer Länge von je 4 Bytes.

```
varchar(120)
```

Wenn Sie den VARCHAR-Datentyp ohne Längenangabe in einer CREATE TABLE-Anweisung verwenden, ist die Standardlänge 256. Bei Verwendung in einem Ausdruck wird die Größe der Ausgabe anhand des Eingabeausdrucks (bis zu 65535) bestimmt.

NCHAR- und NVARCHAR-Typen

Sie können Spalten mit den Typen NCHAR und NVARCHAR erstellen (auch als NATIONAL CHARACTER- und NATIONAL CHARACTER VARYING-Typen bezeichnet). Diese Typen werden jeweils in CHAR- und VARCHAR-Typen konvertiert und in der angegebenen Anzahl von Bytes gespeichert.

Eine NCHAR-Spalte ohne Längenangabe wird in eine CHAR(1)-Spalte konvertiert.

Eine NVARCHAR-Spalte ohne Längenangabe wird in eine VARCHAR(256)-Spalte konvertiert.

TEXT- und BPCHAR-Typen

Sie können eine Amazon-Redshift-Tabelle mit einer TEXT-Spalte erstellen; diese wird jedoch in eine VARCHAR(256)-Spalte konvertiert, die Werte mit einer variablen Länge bis maximal 256 Zeichen akzeptiert.

Sie können eine Amazon-Redshift-Spalte mit dem Typ BPCHAR (blank-padded character) erstellen; diese wird von Amazon Redshift in eine CHAR(256)-Spalte mit fester Länge konvertiert.

Die Bedeutung von Leerzeichen am Ende

Die Datentypen CHAR und VARCHAR speichern Zeichenfolgen mit einer Länge von bis zu n Bytes. Jeder Versuch, eine längere Zeichenfolge in eine Spalten dieses Typs einzufügen, verursacht einen Fehler – nur wenn es sich bei den überschüssigen Zeichen um Leerzeichen handelt, wird die Zeichenfolge auf die maximal zulässige Länge gekürzt. Wenn die Zeichenfolge kürzer als die maximal zulässige Länge ist, werden CHAR-Werte mit Leerzeichen aufgefüllt; VARCHAR-Werte speichern die Zeichenfolge dagegen ohne Leerzeichen.

Leerzeichen am Ende von CHAR-Werten sind semantisch immer ohne Bedeutung. Sie werden beim Vergleich zweier CHAR-Werte ignoriert, werden bei LENGTH-Berechnungen nicht berücksichtigt und werden entfernt, wenn Sie einen CHAR-Wert in einen anderen Zeichenfolgetyp konvertieren.

Leerzeichen am Ende von VARCHAR- und CHAR- Werten werden beim Vergleich von Werten als semantisch ohne Bedeutung behandelt.

Längenberechnungen geben die Länge von VARCHAR-Zeichenfolgen einschließlich der Leerzeichen am Ende zurück. Leerzeichen am Ende werden im Fall von Zeichenfolgen mit fester Länge nicht zu der Länge gezählt.

Beispiele mit Zeichentypen

CREATE TABLE-Anweisung

Die folgende CREATE TABLE-Anweisung illustriert die Verwendung der Datentypen VARCHAR und CHAR:

```
create table address(
address_id integer,
```

```
address1 varchar(100),
address2 varchar(50),
district varchar(20),
city_name char(20),
state char(2),
postal_code char(5)
);
```

Die folgenden Beispiele verwenden diese Tabelle.

Leerzeichen am Ende von Zeichenfolgen mit variabler Länge

Da es sich bei ADDRESS1 um eine VARCHAR-Spalte handelt, sind die Leerzeichen am Ende der zweiten eingefügten Adresse semantisch ohne Bedeutung. In anderen Worten: Diese beiden eingefügten Adressen stimmen überein.

```
insert into address(address1) values('9516 Magnolia Boulevard');

insert into address(address1) values('9516 Magnolia Boulevard  ');
```

```
select count(*) from address
where address1='9516 Magnolia Boulevard';

count
-------
2
(1 row)
```

Wenn die Spalte ADDRESS1 eine CHAR-Spalte wäre und dieselben Werte eingefügt werden, würde die COUNT(*)-Abfrage die Zeichenfolgen als identisch erkennen und ausgeben 2.

Ergebnisse der LENGTH-Funktion

Die LENGTH-Funktion erkennt Leerzeichen am Ende von VARCHAR-Spalten:

```
select length(address1) from address;

length
--------
23
25
```

```
(2 rows)
```

Für den Wert `Augusta` in der Spalte CITY_NAME, einer CHAR-Spalte, würde immer eine Länge von 7 Zeichen ausgegeben werden, unabhängig von evtl. vorhandenen Leerzeichen am Ende der Eingabezeichenfolge.

Werte, die die Länge der Spalte überschreiten

Zeichenfolgen werden nie gekürzt, um der deklarierten Spaltenbreite zu entsprechen:

```
insert into address(city_name) values('City of South San Francisco');
ERROR: value too long for type character(20)
```

Sie können dieses Problem umgehen, indem Sie den Wert an die Spaltengröße anpassen:

```
insert into address(city_name)
values('City of South San Francisco'::char(20));
```

In diesem Fall würden die ersten 20 Zeichen der Zeichenfolge (`City of South San Fr`) in die Spalte geladen werden.

Datum-/Uhrzeittypen

Themen

- Speicherung und Bereiche
- DATUM
- TIME
- TIMETZ
- TIMESTAMP (ZEITSTEMPEL)
- TIMESTAMPTZ
- Beispiele mit Datum-/Uhrzeittypen
- Datums-, Zeit- und Zeitstempelliterale
- Datentypen und Literale für Intervalle

Zu den Datum-/Uhrzeittypen gehören DATE, TIME, TIMETZ, TIMESTAMP und TIMESTAMPTZ.

Speicherung und Bereiche

Name	Speicher	Bereich	Behebung
DATUM	4 Bytes	4713 v. Chr. bis 294276 n. Chr.	1 Tag
TIME	8 Bytes	00:00:00 bis 24:00:00	1 Mikrosekunde
TIMETZ	8 Bytes	00:00:00+1459 bis 00:00:00+1459	1 Mikrosekunde
TIMESTAMP	8 Bytes	4713 v. Chr. bis 294276 n. Chr.	1 Mikrosekunde
TIMESTAMPTZ	8 Bytes	4713 v. Chr. bis 294276 n. Chr.	1 Mikrosekunde

DATUM

Verwenden sie den Datentyp DATE, um einfache Kalenderdaten ohne Zeitstempel zu speichern.

TIME

TIME ist ein Alias von TIME WITHOUT TIME ZONE.

Verwenden Sie den Datentyp TIME, um die Uhrzeit zu speichern.

TIME-Spalten speichern Werte mit einer Genauigkeit von maximal sechs Stellen für Sekundenbruchteile.

TIME-Werte entsprechen standardmäßig der Zeitzone UTC (Coordinated Universal Time), sowohl in Benutzertabellen als auch in Amazon-Redshift-Systemtabellen.

TIMETZ

TIMETZ ist ein Alias von TIME WITH TIME ZONE.

Verwenden Sie den Datentyp TIMETZ, um die Uhrzeit mit einer Zeitzone zu speichern.

TIMETZ-Spalten speichern Werte mit einer Genauigkeit von maximal sechs Stellen für Sekundenbruchteile.

TIMETZ-Werte entsprechen standardmäßig der Zeitzone UTC, sowohl in Benutzertabellen als auch in Amazon-Redshift-Systemtabellen.

TIMESTAMP (ZEITSTEMPEL)

TIMESTAMP ist ein Alias von TIMESTAMP WITHOUT TIME ZONE.

Verwenden sie den Datentyp TIMESTAMP, um vollständige Zeitstempel zu speichern, die das Datum und die Uhrzeit umfassen.

TIMESTAMP-Spalten speichern Werte mit einer Genauigkeit von maximal sechs Stellen für Sekundenbruchteile.

Wenn Sie ein Datum oder ein Datum mit einem unvollständigen Zeitstempelwert in eine TIMESTAMP-Spalte einfügen, wird der Wert implizit in einen vollständigen Zeitstempelwert konvertiert. Dieser vollständige Zeitstempelwert hat Standardwerte (00) für fehlende Stunden, Minuten und Sekunden. Zeitzonenwerte in Eingabezeichenfolgen werden ignoriert.

TIMESTAMP-Werte entsprechen standardmäßig der Zeitzone UTC, sowohl in Benutzertabellen als auch in Amazon-Redshift-Systemtabellen.

TIMESTAMPTZ

TIMESTAMPTZ ist ein Alias von TIMESTAMP WITH TIME ZONE.

Verwenden sie den Datentyp TIMESTAMPTZ, um vollständige Zeitstempel einzugeben, die das Datum, die Uhrzeit und eine Zeitzone umfassen. Wenn in einem Eingabewert eine Zeitzone enthalten ist, verwendet Amazon Redshift die Zeitzone, um den Wert in UTC-Zeit zu konvertieren, und speichert den UTC-Wert.

Führen Sie den folgenden Befehl aus, um eine Liste der unterstützten Zeitzonennamen anzuzeigen.

```
select pg_timezone_names();
```

Führen Sie den folgenden Befehl aus, um eine Liste der unterstützten Zeitzonenabkürzungen anzuzeigen.

```
select pg_timezone_abbrevs();
```

Aktuelle Informationen zu Zeitzonen finden Sie in der IANA Time Zone Database.

In der folgenden Tabelle werden Beispiele zu Zeitzonenformaten aufgeführt.

Format	Beispiel
DD Mon HH:MI:SS YYYY TZ	17 Dec 07:37:16 1997 PST
mm/tt/jjjj hh:mi:ss.ss zz	12/17/1997 07:37:16.00 PST
mm/tt/jjjj hh:mi:ss.ss zz	12/17/1997 07:37:16.00 US/Pacific
yyyy-mm-dd hh:mi:ss+/-tz	1997-12-17 07:37:16-08
tt.mm.jjjj hh:mi:ss zz	17.12.1997 07:37:16.00 PST

TIMESTAMPTZ-Spalten speichern Werte mit einer Genauigkeit von maximal sechs Stellen für Sekundenbruchteile.

Wenn Sie ein Datum oder ein Datum mit einem unvollständigen Zeitstempelwert in eine TIMESTAMPTZ-Spalte einfügen, wird der Wert implizit in einen vollständigen Zeitstempelwert konvertiert. Dieser vollständige Zeitstempelwert hat Standardwerte (00) für fehlende Stunden, Minuten und Sekunden.

TIMESTAMPTZ-Werte in Benutzertabellen entsprechen der Zeitzone UTC.

Beispiele mit Datum-/Uhrzeittypen

Im Folgenden finden Sie Beispiele für die Arbeit mit Datum-/Uhrzeittypen, die von Amazon Redshift unterstützt werden.

Datumsbeispiele

Die folgenden Beispiele fügen Datumsangaben in verschiedenen Formaten ein und zeigen die Ausgabe an.

```
create table datetable (start_date date, end_date date);
```

```
insert into datetable values ('2008-06-01','2008-12-31');

insert into datetable values ('Jun 1,2008','20081231');
```

```
select * from datetable order by 1;
```

```
start_date |  end_date
-----------------------
2008-06-01 | 2008-12-31
2008-06-01 | 2008-12-31
```

Wenn Sie einen Zeitstempel in eine DATE-Spalte eingeben, wird die Uhrzeit ignoriert, und nur das Datum wird geladen.

Zeit-Beispiele

Die folgenden Beispiele fügen TIME- und TIMETZ-Werte in verschiedenen Formaten ein und zeigen die Ausgabe an.

```
create table timetable (start_time time, end_time timetz);
```

```
insert into timetable values ('19:11:19','20:41:19 UTC');
insert into timetable values ('191119', '204119 UTC');
```

```
select * from timetable order by 1;
start_time |  end_time
------------------------
 19:11:19  | 20:41:19+00
 19:11:19  | 20:41:19+00
```

Zeitstempelbeispiele

Wenn Sie ein Datum in eine TIMESTAMP- oder TIMESTAMPTZ-Spalte eingeben, wird für die Uhrzeit standardmäßig Mitternacht verwendet. Wenn Sie beispielsweise das Literal `20081231` eingeben, ist der gespeicherte Wert `2008-12-31 00:00:00`.

Wenn Sie die Zeitzone für die aktuelle Sitzung ändern möchten, verwenden Sie den Befehl SET, um den Konfigurationsparameter Zeitzone einzustellen.

Das folgende Beispiel fügt Zeitstempel in verschiedenen Formaten ein und zeigt die Ausgabe an.

```
create table tstamp(timeofday timestamp, timeofdaytz timestamptz);

insert into tstamp values('Jun 1,2008  09:59:59', 'Jun 1,2008 09:59:59 EST' );
insert into tstamp values('Dec 31,2008 18:20','Dec 31,2008 18:20');
insert into tstamp values('Jun 1,2008  09:59:59 EST', 'Jun 1,2008 09:59:59');
```

```
SELECT * FROM tstamp;

+---------------------+-----------------------+
|      timeofday      |      timeofdaytz      |
+---------------------+-----------------------+
| 2008-06-01 09:59:59 | 2008-06-01 14:59:59+00 |
| 2008-12-31 18:20:00 | 2008-12-31 18:20:00+00 |
| 2008-06-01 09:59:59 | 2008-06-01 09:59:59+00 |
+---------------------+-----------------------+
```

Datums-, Zeit- und Zeitstempelliterale

Im Folgenden finden Sie Regeln zum Arbeiten mit Datums-, Zeit- und Zeitstempelliteralen, die von Amazon Redshift unterstützt werden.

Datumsangaben

Die folgenden Eingabedaten sind allesamt gültige Beispiele für literale Datumswerte für den DATE-Datentyp, die Sie in Amazon Redshift Redshift-Tabellen laden können. Es wird davon ausgegangen, dass der standardmäßige `MDY DateStyle`-Modus aktiviert ist. Dieser Modus bedeutet, dass der Monatswert vor dem Tageswert steht, zum Beispiel in Zeichenfolgen wie `1999-01-08` und `01/02/00`.

> **Note**
>
> Datums- bzw. Zeitstempelliterale müssen in Anführungszeichen stehen, wenn Sie sie in eine Tabelle laden.

Eingegebenes Datum	Vollständiges Datum
January 8, 1999	January 8, 1999
1999-01-08	January 8, 1999
1/8/1999	January 8, 1999
01/02/00	January 2, 2000
2000-Jan-31	January 31, 2000

Eingegebenes Datum	Vollständiges Datum
Jan-31-2000	January 31, 2000
31-Jan-2000	January 31, 2000
20080215	February 15, 2008
080215	February 15, 2008
2008.366	December 31, 2008 (dreistellige Datumskomponente muss zwischen 001 und 366 liegen)

Times

Die folgenden Eingabezeiten sind allesamt gültige Beispiele für literale Zeitwerte für die Datentypen TIME und TIMETZ, die Sie in Amazon Redshift Redshift-Tabellen laden können.

Eingegebene Zeiten	Beschreibung (der Uhrzeitkomponente)
04:05:06.789	4.05 Uhr und 6,789 Sekunden
04:05:06	4.05 Uhr und 6 Sekunden
04:05	Genau 4.05 Uhr
040506	4.05 Uhr und 6 Sekunden
04:05 AM	Genau 4.05 Uhr, AM ist optional
04:05 PM	Genau 16.05 Uhr, Stundenwert muss kleiner als 12 sein
16:05	Genau 16.05 Uhr

Zeitstempel

Die folgenden Eingabezeitstempel sind allesamt gültige Beispiele für literale Zeitwerte für die Datentypen TIMESTAMP und TIMESTAMPTZ, die Sie in Amazon Redshift Redshift-Tabellen laden können. Alle gültigen Datumsliterale können mit den folgenden Uhrzeitliteralen kombiniert werden.

Eingegebene Zeitstempel (konkatenierte Datums- und Uhrzeitliterale)	Beschreibung (der Uhrzeitkomponente)
20080215 04:05:06.789	4.05 Uhr und 6,789 Sekunden
20080215 04:05:06	4.05 Uhr und 6 Sekunden
20080215 04:05	Genau 4.05 Uhr
20080215 040506	4.05 Uhr und 6 Sekunden
20080215 04:05 AM	Genau 4.05 Uhr, AM ist optional
20080215 04:05 PM	Genau 16.05 Uhr, Stundenwert muss kleiner als 12 sein
20080215 16:05	Genau 16.05 Uhr
20080215	Mitternacht (durch Standardwert)

Besondere Datums-/Uhrzeitwerte

Die folgenden besonderen Werte können als Datums-/Uhrzeitliterale und als Parameter für Datumsfunktionen verwendet werden. Sie müssen in einfachen Anführungszeichen (') angegeben werden und werden bei der Verarbeitung der Abfrage in reguläre Zeitstempelwerte umgewandelt.

Sonderwert	Beschreibung
now	Wird zu der Startzeit der aktuellen Transaktion ausgewertet und gibt einen Zeitstempel mit auf Mikrosekunden genauer Uhrzeitkomponente zurück.

Sonderwert	Beschreibung
today	Wird zu dem entsprechenden Datum ausgewertet und gibt einen Zeitstempel mit Nullen für die Uhrzeitkomponente zurück.
tomorrow	Wird zu dem entsprechenden Datum ausgewertet und gibt einen Zeitstempel mit Nullen für die Uhrzeitkomponente zurück.
yesterday	Wird zu dem entsprechenden Datum ausgewertet und gibt einen Zeitstempel mit Nullen für die Uhrzeitkomponente zurück.

Die folgenden Beispiele zeigen, wie now und today mit der DATEADD-Funktion zusammenarbeiten.

```
select dateadd(day,1,'today');

date_add
---------------------
2009-11-17 00:00:00
(1 row)

select dateadd(day,1,'now');

date_add
----------------------------
2009-11-17 10:45:32.021394
(1 row)
```

Datentypen und Literale für Intervalle

Sie können einen Intervalldatentyp verwenden, um Zeitdauern in Einheiten wie,,, seconds minutes hoursdays, months und zu speichern. years Intervalldatentypen und -literale können in Berechnungen von Datum und Uhrzeit verwendet werden, z. B. beim Hinzufügen von Intervallen zu Datums- und Zeitstempeln, beim Summieren von Intervallen und beim Subtrahieren eines Intervalls von einem Datum oder einem Zeitstempel. Intervallliterale können als Eingabewerte für Spalten vom Datentyp Intervall in einer Tabelle verwendet werden.

Syntax des Intervall-Datentyps

Um einen Intervalldatentyp zum Speichern einer Zeitdauer in Jahren und Monaten anzugeben:

```
INTERVAL year_to_month_qualifier
```

Um einen Intervalldatentyp zum Speichern einer Dauer in Tagen, Stunden, Minuten und Sekunden anzugeben:

```
INTERVAL day_to_second_qualifier [ (fractional_precision) ]
```

Syntax des Intervallliterals

Um ein Intervallliteral anzugeben, um eine Zeitdauer in Jahren und Monaten zu definieren:

```
INTERVAL quoted-string year_to_month_qualifier
```

Um ein Intervallliteral anzugeben, um eine Dauer in Tagen, Stunden, Minuten und Sekunden zu definieren:

```
INTERVAL quoted-string day_to_second_qualifier [ (fractional_precision) ]
```

Argumente

Zeichenfolge in Anführungszeichen

Gibt einen positiven oder negativen numerischen Wert an, der eine Menge und die Datums-/Uhrzeiteinheit als Eingabezeichenfolge angibt. Wenn die Zeichenfolge in Anführungszeichen nur eine Zahl enthält, bestimmt Amazon Redshift die Einheiten aus dem year_to_month_qualifier oder day_to_second_qualifier. Zum `'23' MONTH` Beispiel `1 year 11 months` `'-2' DAY` `-2 days 0 hours 0 minutes 0.0 seconds` steht `1 year 2 months` für `'1-2' MONTH` `'13 day 1 hour 1 minute 1.123 seconds' SECOND` `13 days 1 hour 1 minute 1.123 seconds`, repräsentiert, repräsentiert und repräsentiert. Weitere Hinweise zu den Ausgabeformaten eines Intervalls finden Sie unterIntervallstile.

year_to_month_qualifier

Gibt den Bereich des Intervalls an. Wenn Sie einen Qualifier verwenden und ein Intervall mit Zeiteinheiten erstellen, die kleiner als der Qualifier sind, schneidet Amazon Redshift die kleineren Teile des Intervalls ab und verwirft sie. Gültige Werte für year_to_month_qualifier sind:

- YEAR
- MONTH
- YEAR TO MONTH

day_to_second_qualifier

Gibt den Bereich des Intervalls an. Wenn Sie einen Qualifier verwenden und ein Intervall mit Zeiteinheiten erstellen, die kleiner als der Qualifier sind, schneidet Amazon Redshift die kleineren Teile des Intervalls ab und verwirft sie. Gültige Werte für day_to_second_qualifier sind:

- DAY
- HOUR
- MINUTE
- SECOND
- DAY TO HOUR
- DAY TO MINUTE
- DAY TO SECOND
- HOUR TO MINUTE
- HOUR TO SECOND
- MINUTE TO SECOND

Die Ausgabe des INTERVAL-Literals wird auf die kleinste angegebene INTERVAL-Komponente gekürzt. Wenn Sie beispielsweise einen MINUTE-Qualifier verwenden, verwirft Amazon Redshift die Zeiteinheiten, die kleiner als MINUTE sind.

```
select INTERVAL '1 day 1 hour 1 minute 1.123 seconds' MINUTE
```

Der resultierende Wert wird auf gekürzt. `'1 day 01:01:00'`

fractional_precision

Optionaler Parameter, der die Anzahl der im Intervall zulässigen Nachkommastellen angibt. Das Argument fractional_precision sollte nur angegeben werden, wenn Ihr Intervall SECOND enthält. `SECOND(3)`Erzeugt beispielsweise ein Intervall, das nur drei Nachkommastellen erlaubt, z. B. 1,234 Sekunden. Die maximale Anzahl von Nachkommastellen ist sechs.

Die Sitzungskonfiguration `interval_forbid_composite_literals` bestimmt, ob ein Fehler zurückgegeben wird, wenn ein Intervall mit den Teilen JAHR BIS MONAT und TAG BIS SEKUNDE angegeben wird. Weitere Informationen finden Sie unter interval_forbid_composite_literals.

Intervall-Arithmetik

Sie können Intervallwerte zusammen mit anderen Datetime-Werten verwenden, um arithmetische Operationen durchzuführen. In der folgenden Tabelle werden die verfügbaren Operationen und der Datentyp beschrieben, der sich aus den einzelnen Operationen ergibt. Wenn Sie beispielsweise ein `interval` zu einem hinzufügen, ist `date` das Ergebnis ein `date` Intervall von Jahr zu Monat und ein Zeitstempel, wenn es sich um ein Intervall von Tag bis Sekunde handelt.

		Datum	Zeitstempel	Intervall	Numerischer Wert
Interval	-	N/A	N/A	Intervall	N/A
	+	Datum	Datums-/Zeitstempel	Intervall	N/A
	*	–	–	N/A	Intervall
	/	N/A	–	N/A	Intervall
Date (Datum)	-	Numerischer Wert	Intervall	Datum/Zeitstempel	Datum
	+	N/A	–	–	N/A
Zeitstempel	-	Intervall	Intervall	Zeitstempel	Zeitstempel
	+	N/A	–	–	N/A

Intervallstile

Sie können den the section called “SET” SQL-Befehl verwenden, um das Ausgabeanzeigeformat Ihrer Intervallwerte zu ändern. Wenn Sie den Intervalldatentyp in SQL verwenden, wandeln Sie ihn in Text um, um den erwarteten Intervallstil zu sehen, `YEAR TO MONTH::text` z. B. Verfügbare Werte, um den `IntervalStyle` Wert FESTZULEGEN, sind:

- `postgres`— folgt dem PostgreSQL-Stil. Dies ist die Standardeinstellung.
- `postgres_verbose`— folgt dem ausführlichen PostgreSQL-Stil.
- `sql_standard`— folgt dem SQL-Standardstil für Intervallliterale.

Der folgende Befehl setzt den Intervallstil auf`sql_standard`.

```
SET IntervalStyle to 'sql_standard';
```

Postgres-Ausgabeformat

Das Folgende ist das Ausgabeformat für den `postgres` Intervallstil. Jeder numerische Wert kann negativ sein.

```
'<numeric> <unit> [, <numeric> <unit> ...]'
```

```
select INTERVAL '1-2' YEAR TO MONTH::text

varchar
---------------
1 year 2 mons
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
------------------
1 day 02:03:04.5678
```

postgres_verbose, Ausgabeformat

Die postgres_verbose-Syntax ähnelt der von Postgres, aber die Ausgaben von postgres_verbose enthalten auch die Zeiteinheit.

```
'[@] <numeric> <unit> [, <numeric> <unit> ...] [direction]'
```

```
select INTERVAL '1-2' YEAR TO MONTH::text

varchar
-----------------
```

```
@ 1 year 2 mons
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
--------------------------
@ 1 day 2 hours 3 mins 4.56 secs
```

sql_standard-Ausgabeformat

Die Werte für das Intervall von Jahr zu Monat werden wie folgt formatiert. Die Angabe eines negativen Vorzeichens vor dem Intervall bedeutet, dass das Intervall ein negativer Wert ist und für das gesamte Intervall gilt.

```
'[-]yy-mm'
```

Die Werte des Intervalls von Tag bis Sekunde werden wie folgt formatiert.

```
'[-]dd hh:mm:ss.ffffff'
```

```
SELECT INTERVAL '1-2' YEAR TO MONTH::text

varchar
-------
1-2
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
---------------
1 2:03:04.5678
```

Beispiele für den Datentyp „Intervall“

Die folgenden Beispiele zeigen, wie INTERVAL-Datentypen mit Tabellen verwendet werden.

```
create table sample_intervals (y2m interval month, h2m interval hour to minute);
insert into sample_intervals values (interval '20' month, interval '2 days
 1:1:1.123456' day to second);
```

```
select y2m::text, h2m::text from sample_intervals;


      y2m      |       h2m
---------------+-----------------
 1 year 8 mons | 2 days 01:01:00
```

```
update sample_intervals set y2m = interval '2' year where y2m = interval '1-8' year to
 month;
select * from sample_intervals;

   y2m   |       h2m
---------+-----------------
 2 years | 2 days 01:01:00
```

```
delete from sample_intervals where h2m = interval '2 1:1:0' day to second;
select * from sample_intervals;

 y2m | h2m
-----+-----
```

Beispiele für Intervallliterale

Die folgenden Beispiele werden ausgeführt, wobei der Intervallstil auf `postgres` gesetzt ist.

Das folgende Beispiel zeigt, wie ein INTERVAL-Literal für 1 Jahr erstellt wird.

```
select INTERVAL '1' YEAR

intervaly2m
---------------
1 years 0 mons
```

Wenn Sie eine Zeichenfolge in Anführungszeichen angeben, die den Qualifizierer überschreitet, werden die verbleibenden Zeiteinheiten aus dem Intervall gekürzt. Im folgenden Beispiel wird aus einem Intervall von 13 Monaten 1 Jahr und 1 Monat, aber der verbleibende Monat wird aufgrund des Qualifizierers YEAR weggelassen.

```
select INTERVAL '13 months' YEAR
```

```
intervaly2m
---------------
1 years 0 mons
```

Wenn Sie einen Qualifizierer verwenden, der unter Ihrer Intervallzeichenfolge liegt, werden übrig gebliebene Einheiten eingeschlossen.

```
select INTERVAL '13 months' MONTH

intervaly2m
---------------
1 years 1 mons
```

Wenn Sie eine Genauigkeit in Ihrem Intervall angeben, wird die Anzahl der Nachkommastellen auf die angegebene Genauigkeit gekürzt.

```
select INTERVAL '1.234567' SECOND (3)

intervald2s
--------------------------------
0 days 0 hours 0 mins 1.235 secs
```

Wenn Sie keine Genauigkeit angeben, verwendet Amazon Redshift die maximale Genauigkeit von 6.

```
select INTERVAL '1.23456789' SECOND

intervald2s
---------------------------------
0 days 0 hours 0 mins 1.234567 secs
```

Das folgende Beispiel zeigt, wie ein Bereichsintervall erstellt wird.

```
select INTERVAL '2:2' MINUTE TO SECOND

intervald2s
------------------------------
0 days 0 hours 2 mins 2.0 secs
```

Qualifikatoren bestimmen die Einheiten, die Sie angeben. Obwohl im folgenden Beispiel beispielsweise dieselbe Zeichenfolge in Anführungszeichen von '2:2' wie im vorherigen Beispiel

verwendet wird, erkennt Amazon Redshift aufgrund des Qualifizierers, dass es unterschiedliche Zeiteinheiten verwendet.

```
select INTERVAL '2:2' HOUR TO MINUTE

intervald2s
-----------------------------
0 days 2 hours 2 mins 0.0 secs
```

Abkürzungen und Pluralformen der einzelnen Einheiten werden ebenfalls unterstützt. Beispielsweise sind `5s5 second`, und äquivalente `5 seconds` Intervalle. Unterstützte Einheiten sind Jahre, Monate, Stunden, Minuten und Sekunden.

```
select INTERVAL '5s' SECOND

intervald2s
-----------------------------
0 days 0 hours 0 mins 5.0 secs
```

```
select INTERVAL '5 HOURS' HOUR

intervald2s
-----------------------------
0 days 5 hours 0 mins 0.0 secs
```

```
select INTERVAL '5 h' HOUR

intervald2s
-----------------------------
0 days 5 hours 0 mins 0.0 secs
```

Beispiele für Intervallliterale ohne Qualifier-Syntax

> **Note**
>
> Die folgenden Beispiele veranschaulichen die Verwendung eines Intervallliterals ohne einen `YEAR TO MONTH` oder -Bezeichner. `DAY TO SECOND` Hinweise zur Verwendung des empfohlenen Intervallliterals mit einem Qualifier finden Sie unter. Datentypen und Literale für Intervalle

Mit Intervallliteralen können Zeiträume angegeben werden, beispielsweise `12 hours` oder `6 months`. Die Intervallliterale können in Bedingungen und Berechnungen verwendet werden, die Datums-/Uhrzeitausdrücke enthalten.

Ein Intervallliteral wird als Kombination des Schlüsselworts INTERVAL mit einer numerischen Menge und einem unterstützten Datumsteil ausgedrückt, zum Beispiel oder. `INTERVAL '7 days'` `INTERVAL '59 minutes'` Sie können Mengenangaben und Einheiten kombinieren und auf diese Weise das Intervall präzisieren. Beispiel: `INTERVAL '7 days, 3 hours, 59 minutes'`. Die Einheiten können abgekürzt und in ihren Pluralformen verwendet werden. Beispiele: `5 s`, `5 second` und `5 seconds` drücken dasselbe Intervall aus.

Wenn keine Datumskomponente angegeben wird, gibt der Intervallwert Sekunden an. Die Mengenangabe kann auch ein Dezimalwert sein. Beispiel: .: `0.5 days`).

Die folgenden Beispiele stellen eine Reihe von Berechnungen mit verschiedenen Intervallwerten dar.

Im Folgenden wird dem angegebenen Datum 1 Sekunde hinzugefügt.

```
select caldate + interval '1 second' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 00:00:01
(1 row)
```

Im Folgenden wird dem angegebenen Datum 1 Minute hinzugefügt.

```
select caldate + interval '1 minute' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 00:01:00
(1 row)
```

Im Folgenden werden dem angegebenen Datum 3 Stunden und 35 Minuten hinzugefügt.

```
select caldate + interval '3 hours, 35 minutes' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 03:35:00
```

```
(1 row)
```

Im Folgenden werden dem angegebenen Datum 52 Wochen hinzugefügt.

```
select caldate + interval '52 weeks' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2009-12-30 00:00:00
(1 row)
```

Im Folgenden werden dem angegebenen Datum 1 Woche, 1 Stunde, 1 Minute und 1 Sekunde hinzugefügt.

```
select caldate + interval '1w, 1h, 1m, 1s' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2009-01-07 01:01:01
(1 row)
```

Im Folgenden werden dem angegebenen Datum 12 Stunden (ein halber Tag) hinzugefügt.

```
select caldate + interval '0.5 days' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 12:00:00
(1 row)
```

Im Folgenden werden 4 Monate vom 15. Februar 2023 abgezogen und das Ergebnis ist der 15. Oktober 2022.

```
select date '2023-02-15' - interval '4 months';

?column?
---------------------
2022-10-15 00:00:00
```

Im Folgenden werden 4 Monate vom 31. März 2023 abgezogen und das Ergebnis ist der 30. November 2022. Die Berechnung berücksichtigt die Anzahl der Tage in einem Monat.

```
select date '2023-03-31' - interval '4 months';

?column?
---------------------
2022-11-30 00:00:00
```

Typ BOOLEAN

Verwenden Sie den Typ BOOLEAN, um die Wahrheitswerte „wahr“ bzw. „falsch“ in einer Einzelbytespalte zu speichern. Die folgende Tabelle enthält Beschreibungen der drei möglichen Zustände der Booleschen Werte und deren entsprechenden Literale. Unabhängig von der eingegebenen Zeichenfolge werden Werte in Booleschen Spalten mit „t“ für den Wahrheitswert „wahr“ und „f“ für den Wahrheitswert „falsch“ gespeichert und angezeigt.

Status	Zulässige Literalwerte	Speicher
Wahr	`TRUE 't' 'true' 'y' 'yes' '1'`	1 Byte
Falsch	`FALSE 'f' 'false' 'n' 'no' '0'`	1 Byte
Unbekannt	`NULL`	1 Byte

Sie können einen IS-Vergleich nur verwenden, um einen booleschen Wert als Prädikat in der WHERE-Klausel zu prüfen. Sie können den IS-Vergleich nicht mit einem booleschen Wert in der SELECT-Liste verwenden.

Beispiele

Sie können eine Spalte vom Typ BOOLEAN verwenden, um in einer Tabelle CUSTOMER den Status der Kunden als „aktiv“ bzw. „inaktiv“ zu speichern.

```
create table customer(
custid int,
```

```
active_flag boolean default true);
```

```
insert into customer values(100, default);
```

```
select * from customer;
custid | active_flag
-------+-------------
   100 | t
```

Wenn in der CREATE TABLE-Anweisung kein Standardwert (`true` oder `false`) festgelegt wird, wird als Wert stets „Null“ eingefügt.

In diesem Beispiel werden mit einer Abfrage aus der Tabelle USERS diejenigen Benutzer ausgewählt, die Sport mögen, nicht aber Theaterveranstaltungen:

```
select firstname, lastname, likesports, liketheatre
from users
where likesports is true and liketheatre is false
order by userid limit 10;

firstname |  lastname  | likesports | liketheatre
----------+------------+------------+-------------
Lars      | Ratliff    | t          | f
Mufutau   | Watkins    | t          | f
Scarlett  | Mayer      | t          | f
Shafira   | Glenn      | t          | f
Winifred  | Cherry     | t          | f
Chase     | Lamb       | t          | f
Liberty   | Ellison    | t          | f
Aladdin   | Haney      | t          | f
Tashya    | Michael    | t          | f
Lucian    | Montgomery | t          | f
(10 rows)
```

Im folgenden Beispiel werden diejenigen Benutzer aus der Tabelle USERS ausgewählt, von denen nicht bekannt ist, ob sie Rockmusik mögen.

```
select firstname, lastname, likerock
from users
where likerock is unknown
order by userid limit 10;
```

```
firstname | lastname | likerock
----------+----------+----------
Rafael    | Taylor   |
Vladimir  | Humphrey |
Barry     | Roy      |
Tamekah   | Juarez   |
Mufutau   | Watkins  |
Naida     | Calderon |
Anika     | Huff     |
Bruce     | Beck     |
Mallory   | Farrell  |
Scarlett  | Mayer    |
(10 rows)
```

Im folgenden Beispiel wird ein Fehler zurückgegeben, weil ein IS-Vergleich in der SELECT-Liste verwendet wird.

```
select firstname, lastname, likerock is true as "check"
from users
order by userid limit 10;

[Amazon](500310) Invalid operation: Not implemented
```

Das folgende Beispiel wird erfolgreich ausgeführt, weil ein Gleich-Vergleich (=) in der SELECT-Liste anstelle des IS-Vergleichs verwendet wird.

```
select firstname, lastname, likerock = true as "check"
from users
order by userid limit 10;

firstname | lastname  | check
----------+-----------+------
Rafael    | Taylor    |
Vladimir  | Humphrey  |
Lars      | Ratliff   | true
Barry     | Roy       |
Reagan    | Hodge     | true
Victor    | Hernandez | true
Tamekah   | Juarez    |
Colton    | Roy       | false
Mufutau   | Watkins   |
```

```
Naida     | Calderon  |
```

HLLSKETCH-Typ

Verwenden Sie den HLLSKETCH-Datentyp für Skizzen. HyperLogLog Amazon Redshift unterstützt HyperLogLog Skizzendarstellungen, die entweder dünn oder dicht sind. Skizzen sind zunächst rudimentär und werden ausführlicher, wenn das ausführliche Format effizienter ist, um den verwendeten Speicherbedarf zu minimieren.

Amazon Redshift wandelt beim Importieren, Exportieren oder Drucken von HyperLogLog Skizzen im folgenden JSON-Format automatisch eine dünne Skizze um.

```
{"logm":15,"sparse":{"indices":[4878,9559,14523],"values":[1,2,1]}}
```

Amazon Redshift verwendet eine Zeichenkettendarstellung in einem Base64-Format, um eine dichte HyperLogLog Skizze darzustellen.

Amazon Redshift verwendet die folgende Zeichenkettendarstellung in einem Base64-Format, um eine dichte HyperLogLog Skizze darzustellen.

```
"ABAABA..."
```

Die maximale Größe eines HLLSKETCH-Objekts beträgt 24 580 Byte, wenn es in RAW-Komprimierung verwendet wird.

Typ SUPER

Verwenden Sie den SUPER-Datentyp, um halbstrukturierte Daten oder Dokumente als Werte zu speichern.

Halbstrukturierte Daten entsprechen nicht der starren und tabellarischen Struktur des relationalen Datenmodells, das in SQL-Datenbanken verwendet wird. Sie enthalten Tags, die unterschiedliche Entitäten innerhalb der Daten referenzieren. Sie können komplexe Werte wie Arrays, verschachtelte Strukturen und andere komplexe Strukturen enthalten, die Serialisierungsformaten wie JSON zugeordnet sind. Der SUPER-Datentyp ist ein Satz von schemalosen Array- und Strukturwerten, die alle anderen skalaren Typen von Amazon Redshift umfassen.

Der SUPER-Datentyp unterstützt bis zu 16 MB an Daten für ein einzelnes SUPER-Objekt. Weitere Informationen zum SUPER-Datentyp, einschließlich Beispiele für dessen Implementierung in einer Tabelle, finden Sie unter Erfassen und Abfragen von halbstrukturierten Daten in Amazon Redshift.

SUPER-Objekte, die größer als 1 MB sind, können nur aus den folgenden Dateiformaten aufgenommen werden:

- Parquet
- JSON
- TEXT
- CSV

Der SUPER-Datentyp hat folgende Eigenschaften:

- Ein Skalarwert von Amazon Redshift:
  - Ein Nullwert
  - Ein boolescher Wert
  - Eine Zahl, wie z. B. smallint, integer, bigint, decimal oder floating point (z. B. float4 oder float8)
  - Ein Zeichenfolgenwert, z. B. varchar oder char
- Ein komplexer Wert:
  - Ein Array von Werten, einschließlich skalarer oder komplexer
  - Eine Struktur, auch bekannt als Tupel oder Objekt, die eine Zuordnung von Attributnamen und -werten (skalar oder komplex) darstellt

Jeder der beiden Typen komplexer Werte enthält eigene Skalare oder komplexe Werte ohne Einschränkungen für die Regelmäßigkeit.

Der SUPER-Datentyp unterstützt die Persistenz von halbstrukturierten Daten in einer schemalosen Form. Obwohl sich das hierarchische Datenmodell ändern kann, können die alten Datenversionen in derselben SUPER-Spalte nebeneinander existieren.

Amazon Redshift verwendet PartiQL, um die Navigation in Arrays und Strukturen zu ermöglichen. Amazon Redshift verwendet die PartiQL-Syntax auch, um über SUPER-Arrays zu iterieren. Weitere Informationen finden Sie unter Navigation und Aufheben der Verschachtelung von Abfragen.

Amazon Redshift verwendet die dynamische Typisierung, um schemalose SUPER-Daten zu verarbeiten, ohne dass die Datentypen deklariert werden müssen, bevor Sie sie in Ihrer Abfrage verwenden. Weitere Informationen finden Sie unter Dynamische Typisierung.

Sie können dynamische Datenmaskierungsrichtlinien auf `scalar`-Werte in den Pfaden von Spalten des Typs SUPER anwenden. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung. Informationen zur Verwendung der dynamischen Datenmaskierung mit dem SUPER-Datentyp finden Sie unter Verwendung dynamischer Datenmaskierung mit Pfaden des Datentyps SUPER.

Typ VARBYTE

Verwenden Sie eine VARBYTE-, VARBINARY- oder BINARY VARYING-Spalte, um einen Binärwert mit variabler Länge mit einem festen Grenzwert zu speichern.

```
varbyte [ (n) ]
```

Die maximale Anzahl von Byte (n) kann zwischen 1 und 16.777.216 liegen. Der Standardwert ist 64 000.

Hier sind einige Beispiele, in denen es empfehlenswert sein kann, einen VARBYTE-Datentyp zu verwenden:

- Verknüpfen von Tabellen in VARBYTE-Spalten.
- Erstellen materialisierter Ansichten, die VARBYTE-Spalten enthalten. Die inkrementelle Aktualisierung materialisierter Ansichten, die VARBYTE-Spalten enthalten, wird unterstützt. Andere Aggregationsfunktionen als COUNT, MIN und MAX sowie GROUP BY bei VARBYTE-Spalten unterstützen jedoch keine inkrementelle Aktualisierung.

Um sicherzustellen, dass alle Bytes druckbare Zeichen sind, verwendet Amazon Redshift das Hexadezimalformat zum Drucken von VARBYTE-Werten. Das folgende SQL konvertiert beispielsweise die Hexadezimalzeichenfolge 6162 in einen Binärwert. Der zurückgegebene Wert ist zwar ein Binärwert, aber die Ergebnisse werden als Hexadezimalzahl 6162 ausgedruckt.

```
select from_hex('6162');

 from_hex
----------
 6162
```

Amazon Redshift unterstützt die Umwandlung von VARBYTE in die folgenden Datentypen und umgekehrt:

- CHAR
- VARCHAR
- SMALLINT
- INTEGER
- BIGINT

Beim Umwandeln mit CHAR und VARCHAR wird das UTF-8-Format verwendet. Weitere Informationen zum UTF-8-Format finden Sie unter TO_VARBYTE. Beim Umwandeln von SMALLINT, INTEGER und BIGINT wird die Anzahl der Bytes des ursprünglichen Datentyps beibehalten. Das sind zwei Bytes bei SMALLINT, vier Bytes bei INTEGER und acht Bytes bei BIGINT.

Mit der folgenden SQL-Anweisung wird eine VARCHAR-Zeichenfolge in VARBYTE umgewandelt. Der zurückgegebene Wert ist zwar ein Binärwert, aber die Ergebnisse werden als Hexadezimalzahl 616263 ausgedruckt.

```
select 'abc'::varbyte;

 varbyte
---------
 616263
```

Mit der folgenden SQL-Anweisung wird ein CHAR-Wert in einer Spalte in VARBYTE umgewandelt. In diesem Beispiel wird eine Tabelle mit einer CHAR(10)-Spalte (c) erstellt und es werden Zeichenwerte eingefügt, die kürzer als die Länge von 10 sind. Die resultierende Umwandlung füllt das Ergebnis mit Leerzeichen (hex'20') auf die definierte Spaltengröße auf. Der zurückgegebene Wert ist zwar ein Binärwert, aber die Ergebnisse werden als Hexadezimalzahl ausgedruckt.

```
create table t (c char(10));
insert into t values ('aa'), ('abc');
select c::varbyte from t;
          c
----------------------
 6161202020202020202020
 6162632020202020202020
```

Mit der folgenden SQL-Anweisung wird eine SMALLINT-Zeichenfolge in VARBYTE umgewandelt. Der zurückgegebene Wert ist zwar ein Binärwert, aber die Ergebnisse werden als Hexadezimalzahl `0005` ausgedruckt, die zwei Bytes oder vier Hexadezimalzeichen entspricht.

```
select 5::smallint::varbyte;

 varbyte
---------
 0005
```

Mit der folgenden SQL-Anweisung wird INTEGER in VARBYTE umgewandelt. Der zurückgegebene Wert ist zwar ein Binärwert, aber die Ergebnisse werden als Hexadezimalzahl `00000005` ausgedruckt, die vier Bytes oder acht Hexadezimalzeichen entspricht.

```
select 5::int::varbyte;

 varbyte
----------
 00000005
```

Mit der folgenden SQL-Anweisung wird BIGINT in VARBYTE umgewandelt. Der zurückgegebene Wert ist zwar ein Binärwert, aber die Ergebnisse werden als Hexadezimalzahl `0000000000000005` ausgedruckt, die acht Bytes oder 16 Hexadezimalzeichen entspricht.

```
select 5::bigint::varbyte;

     varbyte
------------------
 0000000000000005
```

Zu den Amazon-Redshift-Funktionen, die den VARBYTE-Datentyp unterstützen, gehören:

- VARBYTE-Operatoren
- CONCAT
- LEN
- Die Funktion LENGTH
- OCTET_LENGTH
- Die Funktion SUBSTRING

- FROM_HEX
- TO_HEX
- FROM_VARBYTE
- TO_VARBYTE
- GETBIT
- Laden einer Spalte des Datentyps VARBYTE
- Entladen einer Spalte des Datentyps VARBYTE

Einschränkungen bei der Verwendung des VARBYTE-Datentyps mit Amazon Redshift

Folgende Einschränkungen gelten bei der Verwendung des VARBYTE-Datentyps mit Amazon Redshift:

- Amazon Redshift Spectrum unterstützt den Datentyp VARBYTE nur für Parquet- und ORC-Dateien.
- Der Abfrage-Editor von Amazon Redshift und der Abfrage-Editor v2 von Amazon Redshift unterstützen den VARBYTE-Datentyp noch nicht vollständig. Verwenden Sie deshalb einen anderen SQL-Client, wenn Sie mit VARBYTE-Ausdrücken arbeiten.

  Wenn Sie dennoch den Abfrage-Editor verwenden möchten, gibt es einen Workaround: Sofern die Größe der Daten unter 64 KB liegt und der Inhalt gültiges UTF-8 ist, können Sie die VARBYTE-Werte beispielsweise in VARCHAR umwandeln:

  ```
  select to_varbyte('6162', 'hex')::varchar;
  ```

- Sie können VARBYTE-Datentypen nicht mit benutzerdefinierten Python- oder Lambda-Funktionen (UDFs) verwenden.
- Sie können aus einer VARBYTE-Spalte keine HLLSKETCH-Spalte erstellen oder für eine VARBYTE-Spalte APPROXIMATE COUNT DISTINCT verwenden.
- VARBYTE-Werte, die größer als 1 MB sind, können nur aus den folgenden Dateiformaten aufgenommen werden:
  - Parquet
  - Text
  - Durch Kommas getrennte Werte (CSV)

Kompatibilität von Typen und Umwandlung zwischen Typen

Im Folgenden finden Sie eine Erläuterung der Funktionsweise der Regeln zur Umwandlung von Typen und zur Kompatibilität zwischen Datentypen in Amazon Redshift.

Kompatibilität

Es gibt verschiedene Datenbankoperationen, bei denen die Datentypen passend gemacht und den Literalwerten und Konstanten Datentypen zugewiesen werden. Hierzu gehören die folgenden:

- DML- (Data Manipulation Language-)Operationen über Tabellen
- UNION-, INTERSECT- und EXCEPT-Abfragen
- CASE-Ausdrücke
- Auswertung von Prädikaten wie LIKE oder IN
- Auswertung von SQL-Funktionen, bei denen Vergleiche durchgeführt oder Daten extrahiert werden
- Vergleiche mit mathematischen Operatoren

Die Ergebnisse dieser Operationen hängen von den Regeln zur Umwandlung von Typen und der Kompatibilität zwischen Datentypen ab. Kompatibilität bedeutet, dass ein one-to-one Abgleich eines bestimmten Werts und eines bestimmten Datentyps nicht immer erforderlich ist. Da bestimmte Datentypen untereinander kompatibel sind, ist eine implizite Umwandlung (englisch coercion) möglich. Weitere Informationen finden Sie unter Arten von impliziter Umwandlung. Wenn Datentypen inkompatibel sind, können Sie manchmal einen Wert in einen anderen Datentyp umwandeln, indem Sie eine explizite Typumwandlungsfunktion verwenden.

Allgemeine Regeln zur Kompatibilität und zur Umwandlung

Beachten Sie die folgenden Regeln zur Kompatibilität und zur Typumwandlung:

- Datentypen aus derselben Kategorie sind i. d. R. miteinander kompatibel und können implizit ineinander konvertiert werden. Ein Beispiel hierfür sind numerische Datentypen.

  Sie können beispielsweise mit einer impliziten Umwandlung einen Dezimalwert in eine Spalte mit Ganzzahlen einfügen. Dabei werden Dezimalwerte auf eine Ganzzahl gerundet. Sie können auch einen Zahlenwert wie `2008` aus einem Datum extrahieren und den Wert in eine ganzzahlige Spalte einfügen.
- Numerische Datentypen erzwingen Überlaufbedingungen, die auftreten, wenn Sie versuchen, out-of-range Werte einzufügen. Beispielsweise passt ein Dezimalwert mit einer Genauigkeit von

5 Stellen nicht in eine Dezimalspalte mit einer Genauigkeit von 4 Stellen. Eine Ganzzahl bzw. der ganzzahlige Anteil einer Dezimalzahl wird nie abgeschnitten. Hingegen kann können die Nachkommastellen von Dezimalzahlen bei Bedarf auf- oder abgerundet werden. Die Ergebnisse expliziter Umwandlungen von Werten, aus der Tabelle ausgewählt wurden, werden jedoch nicht gerundet.

- Unterschiedliche Arten von Zeichenfolgen sind miteinander kompatibel. Zeichenfolgen in VARCHAR-Spalten, die Einzelbytedaten enthalten und Zeichenfolgen in CHAR-Spalten können miteinander verglichen und implizit konvertiert werden. VARCHAR-Zeichenfolgen mit Multibytedaten können nicht mit CHAR-Spalten verglichen werden. Sie können eine Zeichenfolge in ein Datum, eine Uhrzeit, einen Zeitstempel oder einen Zahlenwert umwandeln, wenn die Zeichenfolge einen gültigen Literalwert darstellt. Leerzeichen am Anfang und am Ende der Zeichenfolge werden dabei ignoriert. Umgekehrt können Sie auch ein Datum, eine Uhrzeit, einen Zeitstempel oder einen Zahlenwert in eine Zeichenfolge mit fester oder variabler Länge konvertieren.

  > **Note**
  >
  > Wenn Sie eine Zeichenfolge in einen numerischen Typ umwandeln möchten, muss die Zeichenfolge die Zeichendarstellung einer Zahl sein. Sie können beispielsweise die Zeichenfolgen `'1.0'` und `'5.9'` in Dezimalwerte konvertieren, `'ABC'` hingegen in keinen numerischen Typ.

- Wenn Sie DECIMAL-Werte mit Zeichenfolgen vergleichen, versucht Amazon Redshift, die Zeichenfolge in einen DECIMAL-Wert umzuwandeln. Wenn Sie alle anderen numerischen Werte mit Zeichenfolgen vergleichen, werden die numerischen Werte in Zeichenfolgen konvertiert. Um eine Konvertierung in der Gegenrichtung zu erreichen (beispielsweise Zeichenfolgen in Ganzzahlen oder DECIMAL-Werte in Zahlenfolgen umzuwandeln), müssen Sie eine explizite Funktion wie beispielsweise CAST verwenden.
- Wenn Sie einen 64-Bit-Wert vom Typ DECIMAL oder NUMERIC in einen Typ mit einer höheren Genauigkeit umwandeln möchten, müssen Sie eine explizite Funktion verwenden, beispielsweise CAST oder CONVERT.
- Wenn Sie DATE oder TIMESTAMP in TIMESTAMPTZ konvertieren oder TIME in TIMETZ konvertieren, wird die Zeitzone auf die Zeitzone der aktuellen Sitzung festgelegt. Standardmäßig ist als Zeitzone für Sitzungen UTC festgelegt. Weitere Informationen zum Festlegen der Zeitzone für Sitzungen finden Sie unter Zeitzone.

- Analog dazu wird auch TIMESTAMPTZ auf der Grundlage der Zeitzone der aktuellen Sitzung in DATE, TIME oder TIMESTAMP konvertiert. Standardmäßig ist als Zeitzone für Sitzungen UTC festgelegt. Die Zeitzoneninformationen werden nach der Konvertierung wieder entfernt.
- Zeichenfolgen, die einen Zeitstempel mit angegebener Zeitzone darstellen, werden unter Verwendung der Zeitzone der aktuellen Sitzung in TIMESTAMPTZ konvertiert. Diese ist standardmäßig UTC. Zeichenfolgen, die eine Uhrzeit mit angegebener Zeitzone darstellen, werden unter Verwendung der Zeitzone der aktuellen Sitzung in TIMETZ konvertiert. Diese ist standardmäßig UTC.

Arten von impliziter Umwandlung

Es gibt zwei Arten von impliziten Typumwandlungen:

- Implizite Umwandlung bei der Zuweisung, beispielsweise bei der Zuweisung von Werten in INSERT- oder UPDATE-Befehlen.
- Implizite Umwandlung in Ausdrücken, beispielsweise bei Vergleichen in der WHERE-Klausel.

In der folgenden Tabelle sind die Datentypen aufgelistet, die eine implizite Konvertierung in Zuweisungen oder Ausdrücken ermöglichen. Sie können diese Konvertierungen auch mit expliziten Umwandlungsfunktionen durchführen.

Von Typ	Zu Typ
BIGINT (INT8)	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRECISION (FLOAT8)
	INTEGER (INT, INT4)
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR

Von Typ	Zu Typ
CHAR	VARCHAR
DATUM	CHAR
	VARCHAR
	TIMESTAMP
	TIMESTAMPTZ
DECIMAL (NUMERIC)	BIGINT (INT8)
	CHAR
	DOUBLE PRECISION (FLOAT8)
	INTEGER (INT, INT4)
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR
DOUBLE PRECISION (FLOAT8)	BIGINT (INT8)
	CHAR
	DECIMAL (NUMERIC)
	INTEGER (INT, INT4)
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR
INTEGER (INT, INT4)	BIGINT (INT8)

Von Typ	Zu Typ
	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRECISION (FLOAT8)
	REAL (FLOAT4)
	SMALLINT (INT2)
	VARCHAR
REAL (FLOAT4)	BIGINT (INT8)
	CHAR
	DECIMAL (NUMERIC)
	INTEGER (INT, INT4)
	SMALLINT (INT2)
	VARCHAR
SMALLINT (INT2)	BIGINT (INT8)
	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRECISION (FLOAT8)
	INTEGER (INT, INT4)
	REAL (FLOAT4)

Von Typ	Zu Typ
	VARCHAR
TIMESTAMP	CHAR
	DATUM
	VARCHAR
	TIMESTAMPTZ
	TIME
TIMESTAMPTZ	CHAR
	DATUM
	VARCHAR
	TIMESTAMP
	TIMETZ
TIME	VARCHAR
	TIMETZ
	INTERVAL DAY TO SECOND
TIMETZ	VARCHAR
	TIME
GEOMETRY	GEOGRAPHY
GEOGRAPHY	GEOMETRY

> **Note**
>
> Bei impliziten Typumwandlungen zwischen TIMESTAMPTZ, TIMESTAMP, DATE, TIME, TIMETZ oder Zeichenfolgen wird die Zeitzone der aktuellen Sitzung verwendet. Weitere Informationen zum Festlegen der aktuellen Zeitzone finden Sie unter Zeitzone.
> Die Datentypen GEOMETRY und GEOGRAPHY können nicht implizit in einen anderen Datentyp umgewandelt werden (außer untereinander). Weitere Informationen finden Sie unter CAST-Funktion.
> Der Datentyp VARBYTE kann nicht implizit in einen anderen Datentyp umgewandelt werden. Weitere Informationen finden Sie unter CAST-Funktion.

Verwenden der dynamischen Typisierung für den SUPER-Datentyp

Amazon Redshift verwendet die dynamische Typisierung, um schemalose SUPER-Daten zu verarbeiten, ohne dass die Datentypen deklariert werden müssen, bevor Sie sie in Ihrer Abfrage verwenden. Bei der dynamischen Typisierung werden die Ergebnisse der Navigation in SUPER-Datenspalten verwendet, ohne sie explizit in Amazon-Redshift-Typen umwandeln zu müssen. Weitere Informationen zur Verwendung der dynamischen Typisierung für SUPER Datentypen finden Sie unter Dynamische Typisierung.

Sie können SUPER-Werte in andere Datentypen umwandeln und umgekehrt, mit einigen Ausnahmen. Weitere Informationen finden Sie unter Einschränkungen.

Sortierreihenfolgen

Amazon Redshift unterstützt keine Gebietsschema-spezifischen oder benutzerdefinierten Sortierreihenfolgen. Generell gilt, dass das Ergebnis von Prädikaten in einem beliebigem Kontext verfälscht werden kann, wenn keine Gebietsschema-spezifischen Regeln für das Sortieren und Vergleichen von Datenwerten verfügbar sind. ORDER BY-Ausdrücke und Funktionen wie MIN, MAX und RANK geben ihre Ergebnisse auf der Basis der binären UTF8-Reihenfolge der Daten zurück. Diese Sortierung berücksichtigt keine gebietsschemaspezifischen Zeichen.

Ausdrücke

Themen

- Einfache Ausdrücke
- Zusammengesetzte Ausdrücke

- Ausdruckslisten
- Skalare Unterabfragen
- Funktionsausdrücke

Ein Ausdruck ist eine Kombination von Werten, Operatoren und Funktionen, die zu einem Wert ausgewertet werden können. Der Datentyp eines Ausdrucks wird normalerweise durch seine Komponenten festgelegt.

Einfache Ausdrücke

Die folgenden Elemente sind alle einfache Ausdrücke:

- Konstanten und Literalwerte
- Spaltennamen und Spaltenverweise
- Skalare Funktionen
- Aggregationsfunktionen (Mengenfunktionen)
- Fensterfunktionen
- Skalare Unterabfragen

Einige Beispiele für den einfache Ausdrücke:

```
5+12
dateid
sales.qtysold * 100
sqrt (4)
max (qtysold)
(select max (qtysold) from sales)
```

Zusammengesetzte Ausdrücke

Zusammengesetzte Ausdrücke sind Folgen von einfachen Ausdrücken, die durch arithmetische Operatoren miteinander verbunden sind. Einfache Ausdrücke in zusammengesetzten Ausdrücken müssen einen Zahlenwert zurückgeben.

Syntax

```
expression
operator
```

```
expression | (compound_expression)
```

Argumente

expression

Ein einfacher Ausdruck, der zu einem Wert ausgewertet wird.

operator

Ein zusammengesetzter arithmetischer Ausdruck kann unter Verwendung der folgenden Operatoren, unter Berücksichtigung der unten angegeben Vorrang-Reihenfolge konstruiert werden:

- (): Klammern zur Steuerung der Auswertungsabfolge
- + -: Positives und negatives Vorzeichen bzw. einstelliger Operator
- ^ , |/ , ||/ : Potenzierung, Quadratwurzel, Kubikwurzel
- * , / , % : Multiplikation, Division und Modulo
- @: Absoluter Wert (Betrag)
- + , - : Addition und Subtraktion
- & , |, #, ~, <<, >> : AND, OR, NOT, bitweise Links- und Rechtsverschiebung
- ||: Verkettung

(compound_expression)

Zusammengesetzte Ausdrücke können mit Klammern verschachtelt werden.

Beispiele

Einige Beispiele für zusammengesetzte Ausdrücke finden Sie im Folgenden.

```
('SMITH' || 'JONES')
sum(x) / y
sqrt(256) * avg(column)
rank() over (order by qtysold) / 100
(select (pricepaid - commission) from sales where dateid = 1882) * (qtysold)
```

Einige Funktionen können auch in andere Funktionen verschachtelt werden. Es können beispielsweise Skalarfunktionen in andere Skalarfunktionen verschachtelt werden. Das folgende Beispiel gibt die Summe der Beträge aus einer Zahlenmenge zurück:

```
sum(abs(qtysold))
```

Fensterfunktionen können nicht als Argumente in Aggregationsfunktionen oder anderen Fensterfunktionen verwendet werden. Der folgende Ausdruck gibt einen Fehler zurück:

```
avg(rank() over (order by qtysold))
```

Fensterfunktionen können verschachtelte Aggregationsfunktionen enthalten. Der folgende Ausdruck summiert Wertmengen auf und sortiert sie:

```
rank() over (order by sum(qtysold))
```

Ausdruckslisten

Eine Ausdrucksliste ist eine Kombination von Ausdrücken, die in Element- und Vergleichsbedingungen (WHERE-Klauseln) und in GROUP BY-Klauseln vorkommen können.

Syntax

```
expression , expression , ... | (expression, expression, ...)
```

Argumente

expression

Ein einfacher Ausdruck, der zu einem Wert ausgewertet wird. Eine Ausdrucksliste kann eine oder mehrere kommagetrennte Ausdrücke oder eine oder mehrere Mengen von kommagetrennte Ausdrücken enthalten. Bei mehreren Ausdrucksmengen müssen alle Mengen dieselbe Anzahl an Ausdrücken enthalten und durch Klammern untergliedert sein. Die Anzahl der Ausdrücke in jeder Menge muss mit der Anzahl der Ausdrücke vor dem Operator in der Bedingung übereinstimmen.

Beispiele

Im Folgenden sehen Sie einige Beispiele für Ausdruckslisten in Bedingungen.

```
(1, 5, 10)
('THESE', 'ARE', 'STRINGS')
(('one', 'two', 'three'), ('blue', 'yellow', 'green'))
```

Die Anzahl der Ausdrücke in jeder Menge in der Bedingung muss mit der ersten Komponente der Anweisung übereinstimmen:

```
select * from venue
where (venuecity, venuestate) in (('Miami', 'FL'), ('Tampa', 'FL'))
order by venueid;

venueid |        venuename        | venuecity | venuestate | venueseats
---------+-------------------------+-----------+------------+------------
28 | American Airlines Arena | Miami     | FL         |          0
54 | St. Pete Times Forum    | Tampa     | FL         |          0
91 | Raymond James Stadium   | Tampa     | FL         |      65647
(3 rows)
```

Skalare Unterabfragen

Eine skalare Unterabfrage ist eine reguläre SELECT-Abfrage, die in Klammern eingefasst ist, und die genau einen Wert zurückgibt, nämlich eine Zeile mit einer Spalte. Wenn die Abfrage ausgeführt wird, kann der Rückgabewert in der äußeren Abfrage verwendet werden. Wenn die Unterabfrage keine Zeilen zurückgibt, ist der Wert der Unterabfrage „Null“. Wenn mehr als eine Zeile zurückgegeben wird, gibt Amazon Redshift einen Fehler zurück. Die Unterabfrage kann auf Variablen aus der übergeordneten Abfrage verweisen, die beim einem Aufruf der Unterabfrage wie eine Konstante behandelt werden.

Sie können skalare Unterabfragen in den meisten Anweisungen verwenden, die einen Ausdruck als Argument zu sich nehmen. In den folgenden Fällen sind skalare Unterabfragen hingegen keine gültigen Ausdrücke:

- Als Standardwerte für Ausdrücke
- In GROUP BY- und HAVING-Klauseln

Beispiel

Die folgende Unterabfrage berechnet den durchschnittlichen Preis pro Verkauf über das gesamte Jahr 2008. Anschließend verwendet die äußere Abfrage diesen Wert in der Ausgabe, um einen Vergleich des durchschnittlichen Preises für die Verkäufe pro Quartal durchzuführen:

```
select qtr, avg(pricepaid) as avg_saleprice_per_qtr,
(select avg(pricepaid)
from sales join date on sales.dateid=date.dateid
```

```
where year = 2008) as avg_saleprice_yearly
from sales join date on sales.dateid=date.dateid
where year = 2008
group by qtr
order by qtr;
qtr  | avg_saleprice_per_qtr | avg_saleprice_yearly
-------+----------------------+----------------------
1     |               647.64 |               642.28
2     |               646.86 |               642.28
3     |               636.79 |               642.28
4     |               638.26 |               642.28
(4 rows)
```

Funktionsausdrücke

Syntax

Alle integrierten Funktionen können als Ausdruck verwendet werden. Die Syntax für Funktionsaufrufe ist der Name einer Funktion, gefolgt von der Liste der Funktionsargumente in Klammern.

```
function ( [expression [, expression...]] )
```

Argumente

Funktion

Alle integrierten Funktionen Beispielfunktionen finden Sie unter SQL-Funktionsreferenz.

expression

Beliebige Ausdrücke, allerdings entsprechend den Datentypen und der Anzahl der Argumente, die die Funktion erwartet.

Beispiele

```
abs (variable)
select avg (qtysold + 3) from sales;
select dateadd (day,30,caldate) as plus30days from date;
```

Bedingungen

Themen

Eine Bedingung ist eine Anweisung, die aus einem oder mehreren Ausdrücken und logischen Operatoren besteht, und die zu einem der Werte „wahr“, „falsch“ oder „unbekannt“ ausgewertet werden kann. Bedingungen werden manchmal auch als Prädikate bezeichnet.

> **Note**
>
> Bei Vergleichen von Zeichenfolgen und bei LIKE-Patternmatches wird die Groß-/Kleinschreibung berücksichtigt. Zum Beispiel entsprechen sich 'A' und 'a' nicht. Wenn Sie beim Patternmatching die Groß-/Kleinschreibung nicht berücksichtigen möchten, werden Sie statt LIKE das Prädikat ILIKE.

Syntax

```
comparison_condition
| logical_condition
| range_condition
| pattern_matching_condition
| null_condition
| EXISTS_condition
| IN_condition
```

Vergleichsbedingung

Vergleichsbedingungen machen eine Aussage bezüglich der logischen Beziehungen zwischen zwei Werten. Alle Vergleichsbedingungen sind binäre Operatoren mit einem Booleschen Rückgabewert. Amazon Redshift unterstützt die in der folgenden Tabelle beschriebenen Vergleichsoperatoren:

Operator	Syntax	Beschreibung
<	`a < b`	Der Wert von a ist kleiner als der Wert von b.
>	`a > b`	Der Wert von a ist größer als der Wert von b.
<=	`a <= b`	Der Wert von a ist kleiner als oder gleich dem Wert von b.
>=	`a >= b`	Der Wert von a ist größer als oder gleich dem Wert von b.
=	`a = b`	Der Wert von a ist gleich dem Wert von b.
<> oder !=	`a <> b or a != b`	Der Wert von a ist nicht gleich dem Wert von b.
ANY | SOME	`a = ANY(subquery)`	Der Wert von a entspricht einem der in der Unterabfrage „subquery“ zurückgegebenen Werte.
ALL	`a <> ALL or != ALL (subquery))`	Der Wert von a entspricht keinem der in der Unterabfrage „subquery“ zurückgegebenen Werte.
IS TRUE | FALSE | UNKNOWN	`a IS TRUE`	Der Wert von a ist der Boolesche Wert „wahr“ (TRUE).

Nutzungshinweise

= ANY | SOME

Die Schlüsselwörter ANY und SOME sind synonym mit der IN-Bedingung. Sie geben „wahr“ zurück, wenn mindestens einer der von einer Unterabfrage zurückgegebenen Werte „wahr“ ist. Amazon Redshift unterstützt nur die Bedingung = (gleich) für ANY und SOME. Ungleichheitsbedingungen werden nicht unterstützt.

> **Note**
>
> Das Prädikat ALL wird nicht unterstützt.

<> ALL

Das Schlüsselwort ALL ist synonym mit NOT IN (vgl. IN-Bedingung-Bedingung) und gibt „wahr“ zurück, wenn der Ausdruck nicht in den Ergebnissen der Unterabfrage enthalten ist. Amazon Redshift unterstützt nur die <> oder != (nicht gleich)-Bedingung für ALL. Andere Vergleichsbedingungen werden nicht unterstützt.

IS TRUE/FALSE/UNKNOWN

Werte ungleich 0 werden zu TRUE ausgewertet, 0 zu FALSE, und „Null“ zu UNKNOWN. Siehe Datentyp Typ BOOLEAN.

Beispiele

Einige einfache Beispiele für Vergleichsbedingungen:

```
a = 5
a < b
min(x) >= 5
qtysold = any (select qtysold from sales where dateid = 1882
```

Die folgende Abfrage gibt aus der Tabelle VENUE die Veranstaltungen mit über 10.000 Plätzen zurück:

```
select venueid, venuename, venueseats from venue
where venueseats > 10000
order by venueseats desc;

venueid |           venuename            | venueseats
---------+--------------------------------+------------
83 | FedExField                    |      91704
 6 | New York Giants Stadium       |      80242
79 | Arrowhead Stadium             |      79451
78 | INVESCO Field                 |      76125
69 | Dolphin Stadium               |      74916
67 | Ralph Wilson Stadium          |      73967
76 | Jacksonville Municipal Stadium |      73800
89 | Bank of America Stadium       |      73298
72 | Cleveland Browns Stadium      |      73200
86 | Lambeau Field                 |      72922
...
```

```
(57 rows)
```

In diesem Beispiel werden diejenigen Benutzer (USERID) aus der Tabelle USERS ausgewählt, die Rockmusik schätzen:

```
select userid from users where likerock = 't' order by 1 limit 5;

userid
--------
3
5
6
13
16
(5 rows)
```

In diesem Beispiel werden diejenigen Benutzer(USERID) aus der Tabelle USERS ausgewählt, von denen nicht bekannt ist, ob sie Rockmusik schätzen:

```
select firstname, lastname, likerock
from users
where likerock is unknown
order by userid limit 10;

firstname | lastname | likerock
----------+----------+----------
Rafael    | Taylor   |
Vladimir  | Humphrey |
Barry     | Roy      |
Tamekah   | Juarez   |
Mufutau   | Watkins  |
Naida     | Calderon |
Anika     | Huff     |
Bruce     | Beck     |
Mallory   | Farrell  |
Scarlett  | Mayer    |
(10 rows
```

Beispiele mit einer TIME-Spalte

Die folgende Beispieltabelle TIME_TEST enthält eine Spalte TIME_VAL (Typ TIME) mit drei eingefügten Werten.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

Im folgenden Beispiel werden die Stunden aus jedem timetz_val extrahiert.

```
select time_val from time_test where time_val < '3:00';
   time_val
---------------
 00:00:00.5550
 00:58:00
```

Im folgenden Beispiel werden zwei Zeitliterale verglichen.

```
select time '18:25:33.123456' = time '18:25:33.123456';
 ?column?
----------
 t
```

Beispiele mit einer TIMETZ-Spalte

Die folgende Beispieltabelle TIMETZ_TEST enthält eine Spalte TIMETZ_VAL (Typ TIMETZ) mit drei eingefügten Werten.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

Im folgenden Beispiel werden nur die TIMETZ-Werte ausgewählt, die kleiner als sind `3:00:00 UTC`. Der Vergleich erfolgt nach der Umwandlung des Wertes in UTC.

```
select timetz_val from timetz_test where timetz_val < '3:00:00 UTC';
```

```
   timetz_val
---------------
 00:00:00.5550+00
```

Im folgenden Beispiel werden zwei TIMETZ-Literale verglichen. Beim Vergleich wird die Zeitzone ignoriert.

```
select time '18:25:33.123456 PST' < time '19:25:33.123456 EST';

 ?column?
----------
 t
```

Logische Bedingungen

Logische Bedingungen führen die Ergebnisse zweier Bedingungen zu einem Ergebnis zusammen. Alle logischen Bedingungen sind binäre Operatoren mit einem Booleschen Rückgabewert.

Syntax

```
expression
{ AND | OR }
expression
NOT expression
```

Bei logischen Bedingungen wird eine dreiwertige Boolesche Logik verwendet, bei der der Wert „Null“ als „unbekannt“ interpretiert wird. Die folgende Tabelle beschreibt die Ergebnisse von logischen Bedingungen, wobei `E1` und `E2` Ausdrücke sind:

E1	E2	E1 AND E2	E1 OR E2	NOT E2
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
TRUE	UNKNOWN	UNKNOWN	TRUE	UNKNOWN
FALSE	TRUE	FALSE	TRUE	

E1	E2	E1 AND E2	E1 OR E2	NOT E2
FALSE	FALSE	FALSE	FALSE	
FALSE	UNKNOWN	FALSE	UNKNOWN	
UNKNOWN	TRUE	UNKNOWN	TRUE	
UNKNOWN	FALSE	FALSE	UNKNOWN	
UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN	

Der Operator NOT wird vor AND ausgewertet, und AND vor OR. Diese Auswertungsreihenfolge kann durch Klammerung außer Kraft gesetzt werden.

Beispiele

In dem folgenden Beispiel werden USERID und USERNAME aus der Tabelle USERS zurückgegeben, die sowohl Las Vegas als auch Sport mögen:

```
select userid, username from users
where likevegas = 1 and likesports = 1
order by userid;

userid | username
--------+----------
1 | JSG99FHE
67 | TWU10MZT
87 | DUF19VXU
92 | HYP36WEQ
109 | FPL38HZK
120 | DMJ24GUZ
123 | QZR22XGQ
130 | ZQC82ALK
133 | LBN45WCH
144 | UCX04JKN
165 | TEY68OEB
169 | AYQ83HGO
184 | TVX65AZX
...
(2128 rows)
```

In dem nächsten Beispiel werden USERID und USERNAME aus der Tabelle USERS zurückgegeben, die Las Vegas oder Sport mögen: Diese Abfrage gibt alle Ergebnisse des vorangehenden Beispiels, zuzüglich der Benutzer, die Las Vegas mögen, zuzüglich der Benutzer, die Sport mögen.

```
select userid, username from users
where likevegas = 1 or likesports = 1
order by userid;

userid | username
--------+----------
1 | JSG99FHE
2 | PGL08LJI
3 | IFT66TXU
5 | AEB55QTM
6 | NDQ15VBM
9 | MSD36KVR
10 | WKW41AIW
13 | QTF33MCG
15 | OWU78MTR
16 | ZMG93CDD
22 | RHT62AGI
27 | KOY02CVE
29 | HUH27PKK
...
(18968 rows)
```

In der folgenden Abfrage wird die Bedingung OR in Klammern gesetzt, um alle Veranstaltungen zu suchen, die in New York oder in Kalifornien stattfinden, und bei denen Macbeth gegeben wird:

```
select distinct venuename, venuecity
from venue join event on venue.venueid=event.venueid
where (venuestate = 'NY' or venuestate = 'CA') and eventname='Macbeth'
order by 2,1;

venuename                |   venuecity
-----------------------------------------+---------------
Geffen Playhouse                         | Los Angeles
Greek Theatre                            | Los Angeles
Royce Hall                               | Los Angeles
American Airlines Theatre                | New York City
August Wilson Theatre                    | New York City
Belasco Theatre                          | New York City
```

```
Bernard B. Jacobs Theatre              | New York City
...
```

Wenn die Klammerung in dem vorangehenden Beispiel entfernt wird, ändert sich die der bei der Auswertung ermittelte Wert und damit das Ergebnis der Abfrage.

In dem folgenden Beispiel wird der Operator NOT verwendet.

```
select * from category
where not catid=1
order by 1;

catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
2 | Sports   | NHL       | National Hockey League
3 | Sports   | NFL       | National Football League
4 | Sports   | NBA       | National Basketball Association
5 | Sports   | MLS       | Major League Soccer
...
```

Im folgenden Beispiel wird eine NOT-Bedingung verwendet, gefolgt von einer AND-Bedingung:

```
select * from category
where (not catid=1) and catgroup='Sports'
order by catid;

catid | catgroup | catname |             catdesc
-------+----------+---------+---------------------------------
2 | Sports   | NHL     | National Hockey League
3 | Sports   | NFL     | National Football League
4 | Sports   | NBA     | National Basketball Association
5 | Sports   | MLS     | Major League Soccer
(4 rows)
```

Patternmatching-Bedingungen

Themen

- LIKE
- SIMILAR TO
- POSIX-Operatoren

Patternmatching-Operatoren durchsuchen eine Zeichenfolge nach einem Muster, das in dem Bedingungsausdruck übergeben wird, und geben „wahr“ oder „falsch“ zurück, je nachdem, ob eine Übereinstimmung gefunden wird. Amazon Redshift verwendet drei Methoden für Patternmatching:

- LIKE-Ausdrücke

  Der LIKE-Operator vergleicht einen Zeichenfolgenausdruck (beispielsweise einen Spaltennamen) mit einem Muster, in dem die Platzhalterzeichen % (Prozentzeichen) und _ (Unterstrich) verwendet werden können. Beim LIKE-Patternmatching wird jeweils die gesamte Zeichenfolge durchsucht. Bei LIKE erfolgt diese Suche unter Berücksichtigung der Groß-/Kleinschreibung, bei ILIKE wird die Groß-/Kleinschreibung ignoriert.
- Reguläre Ausdrücke mit SIMILAR TO

  Der Operator SIMILAR TO führt das Patternmatching unter Verwendung von regulären Ausdrücken entsprechend dem Standardformat für SQL durch. Als Metazeichen werden dabei beiden Zeichen unterstützt, die auch der Operator LIKE unterstützt. Bei SIMILAR TO erfolgt die Suche über die gesamte Zeichenfolge und unter Berücksichtigung der Groß-/Kleinschreibung.
- Reguläre Ausdrücke entsprechend der POSIX-Spezifikation

  Reguläre Ausdrücke entsprechend der POSIX-Spezifikation ermöglichen eine feinere Modellierung von Mustern als die Operatoren LIKE und SIMILAR TO. Reguläre Ausdrücke entsprechend der POSIX-Spezifikation können auf beliebige Teilzeichenfolgen angewendet werden, die Suche erfolgt unter Berücksichtigung der Groß-Kleinschreibung.

Patternmatching mit regulären Ausdrücken unter Verwendung der Operatoren SIMILAR TO und der POSIX-Operatoren erfordert einen hohen Rechenaufwand. Wir empfehlen, nach Möglichkeit LIKE zu verwenden, insbesondere, wenn eine große Anzahl an Zeilen verarbeitet werden muss. Ein Beispiel: Die folgenden Abfragen sind funktional identisch, aber die Abfrage mit LIKE wird 7-mal schneller ausgeführt als die Abfrage mit einem regulären Ausdruck:

```
select count(*) from event where eventname SIMILAR TO '%(Ring|Die)%';
select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die%';
```

LIKE

Der LIKE-Operator vergleicht einen Zeichenfolgenausdruck (beispielsweise einen Spaltennamen) mit einem Muster, in dem die Platzhalterzeichen % (Prozentzeichen) und _ (Unterstrich) verwendet werden können. Beim LIKE-Patternmatching wird jeweils die gesamte Zeichenfolge durchsucht. Um

für ein Muster anzugeben, dass es an einer beliebigen Stelle innerhalb der Zeichenfolge auftreten kann, muss es in Prozentzeichen eingeschlossen werden.

Bei LIKE wird die Groß-/Kleinschreibung berücksichtigt, bei ILIKE nicht.

Syntax

```
expression [ NOT ] LIKE | ILIKE pattern [ ESCAPE 'escape_char' ]
```

Argumente

expression

Ein gültiger UTF8-Zeichenfolgenausdruck (beispielsweise ein Spaltenname).

LIKE | ILIKE

Bei LIKE wird beim Patternmatching die Groß-/Kleinschreibung berücksichtigt. Bei ILIKE wird beim Patternmatching die Groß-/Kleinschreibung nicht berücksichtigt, wenn die Zeichenfolge aus UTF-8 (ASCII)-Zeichen besteht. Zur Durchführung eines Patternmatchingvorgangs ohne Berücksichtigung der Groß-/Kleinschreibung verwenden Sie die LOWER-Funktion für expression und pattern mit einer LIKE-Bedingung.

Im Gegensatz zu Vergleichsprädikaten wie = und <> ignorieren LIKE- und ILIKE-Prädikate nicht implizit nachfolgende Leerzeichen. Um nachfolgende Leerzeichen zu ignorieren, verwenden Sie RTRIM, oder konvertieren Sie eine CHAR-Spalte explizit zu VARCHAR.

Der Operator ~~ entspricht LIKE und ~~* entspricht ILIKE. Die Operatoren !~~ und !~~* entsprechen außerdem NOT LIKE und NOT ILIKE.

pattern

Ein gültiger UTF8-Zeichenfolgenausdruck mit dem Muster für das Patternmatching.

escape_char

Ein Zeichenfolgenausdruck zur Kennzeichnung von Metazeichen im Muster als Literal. Dies ist standardmäßig die Zeichenfolge \\ (doppelter umgekehrter Schrägstrich).

Wenn das Muster pattern keine Metazeichen enthält, ist wird das Muster als die Zeichenfolge selbst interpretiert. In diesem Fall liefert LIKE dasselbe Ergebnis wie der Gleichheitsoperator.

Die Zeichenfolgenausdrücke können vom Datentyp CHAR oder VARCHAR sein. Wenn unterschiedliche Datentypen verwendet werden, konvertiert Amazon Redshift pattern in den Datentyp des Ausdrucks expression.

LIKE unterstützt die folgenden Metazeichen in Mustern:

Operator	Beschreibung
%	Entspricht einer Folge von 0 oder mehr Zeichen.
_	Entspricht einem beliebigen Zeichen.

Beispiele

In der folgenden Tabelle werden Beispiele für Patternmatching mit LIKE dargestellt.

Ausdruck	Rückgabewert
`'abc' LIKE 'abc'`	Wahr
`'abc' LIKE 'a%'`	True
`'abc' LIKE '_B_'`	False
`'abc' ILIKE '_B_'`	True
`'abc' LIKE 'c%'`	Falsch

Das folgende Beispiel listet alle Städte auf, die mit „E“ beginnen:

```
select distinct city from users
where city like 'E%' order by city;
city
--------------
East Hartford
East Lansing
East Rutherford
East St. Louis
Easthampton
```

```
Easton
Eatontown
Eau Claire
...
```

Das folgende Beispiel listet alle Benutzer auf, deren Nachname „ten“ enthält:

```
select distinct lastname from users
where lastname like '%ten%' order by lastname;
lastname
-------------
Christensen
Wooten
...
```

Im folgenden Beispiel wird veranschaulicht, wie Sie mehrere Muster zuordnen.

```
select distinct lastname from tickit.users
where lastname like 'Chris%' or lastname like '%Wooten' order by lastname;
lastname
-------------
Christensen
Christian
Wooten
...
```

Das folgende Beispiel listet alle Städte auf, deren dritter und vierter Buchstabe die Folge „ea“ ist:
In dem Befehl wird ILIKE verwendet, um zu zeigen, dass bei ILIKE die Groß-/Kleinschreibung nicht berücksichtigt wird:

```
select distinct city from users where city ilike '__EA%' order by city;
city
-------------
Brea
Clearwater
Great Falls
Ocean City
Olean
Wheaton
(6 rows)
```

Im folgenden Beispiel wird die Standard-Escape-Zeichenfolge (\\) verwendet, um nach Zeichenfolgen zu suchen, die „start_“ (den Text `start` gefolgt von einem Unterstrich _) enthalten:

```
select tablename, "column" from pg_table_def
where "column" like '%start\\_%'
limit 5;

     tablename     |    column
-------------------+---------------
 stl_s3client      | start_time
 stl_tr_conflict   | xact_start_ts
 stl_undone        | undo_start_ts
 stl_unload_log    | start_time
 stl_vacuum_detail | start_row
(5 rows)
```

Im folgenden Beispiel wird als Escape-Zeichenfolge ^ (das Caret-Zeichen) verwendet, und dann wird nach Zeichenfolgen gesucht, die „start_“ (den Text `start` gefolgt von einem Unterstrich _) enthalten:

```
select tablename, "column" from pg_table_def
where "column" like '%start^_%' escape '^'
limit 5;

     tablename     |    column
-------------------+---------------
 stl_s3client      | start_time
 stl_tr_conflict   | xact_start_ts
 stl_undone        | undo_start_ts
 stl_unload_log    | start_time
 stl_vacuum_detail | start_row
(5 rows)
```

Im folgenden Beispiel wird der Operator ~~* verwendet, um eine Suche ohne Berücksichtigung der Groß- und Kleinschreibung (ILIKE) nach Städten durchzuführen, die mit „Ag“ beginnen.

```
select distinct city from users where city ~~* 'Ag%' order by city;

city
------------
Agat
Agawam
Agoura Hills
```

```
Aguadilla
```

SIMILAR TO

Der SIMILAR TO-Operator vergleicht einen Zeichenfolgenausdruck (beispielsweise einen Spaltennamen) mit einem regulären Ausdruck entsprechend dem Standardformat für SQL. Ein regulärer Ausdruck entsprechend dem Standardformat für SQL kann verschiedene Metazeichen zum Patternmatching enthalten, darunter die beiden Zeichen, die auch der Operator LIKE unterstützt.

Der Operator SIMILAR TO gibt „wahr“ genau dann zurück, wenn das Muster der gesamten Zeichenfolge entspricht. Bei regulären Ausdrücken entsprechend dem POSIX-Standard hingegen kann das Muster einer beliebigen Teilzeichenfolge entsprechen.

Bei SIMILAR TO wird beim Patternmatching die Groß-/Kleinschreibung berücksichtigt.

> **Note**
>
> Patternmatching mit regulären Ausdrücken unter Verwendung von SIMILAR TO erfordert einen hohen Rechenaufwand. Wir empfehlen, nach Möglichkeit LIKE zu verwenden, insbesondere, wenn eine große Anzahl an Zeilen verarbeitet werden muss. Ein Beispiel: Die folgenden Abfragen sind funktional identisch, aber die Abfrage mit LIKE wird 7-mal schneller ausgeführt als die Abfrage mit einem regulären Ausdruck:
>
> ```
> select count(*) from event where eventname SIMILAR TO '%(Ring|Die)%';
> select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die
> %';
> ```

Syntax

```
expression [ NOT ] SIMILAR TO pattern [ ESCAPE 'escape_char' ]
```

Argumente

expression

Ein gültiger UTF8-Zeichenfolgenausdruck (beispielsweise ein Spaltenname).

SIMILAR TO

Bei SIMILAR TO erfolgt die Suche über die gesamte Zeichenfolge in dem Ausdruck expression und unter Berücksichtigung der Groß-/Kleinschreibung.

pattern

Ein gültiger UTF8-Zeichenfolgenausdruck, der einen regulären Ausdruck entsprechend dem SQL-Standard darstellt.

escape_char

Ein Zeichenfolgenausdruck zur Kennzeichnung von Metazeichen im Muster als Literal. Dies ist standardmäßig die Zeichenfolge \\ (doppelter umgekehrter Schrägstrich).

Wenn das Muster pattern keine Metazeichen enthält, wird das Muster als die Zeichenfolge selbst interpretiert.

Die Zeichenfolgenausdrücke können vom Datentyp CHAR oder VARCHAR sein. Wenn unterschiedliche Datentypen verwendet werden, konvertiert Amazon Redshift pattern in den Datentyp des Ausdrucks expression.

SIMILAR TO unterstützt die folgenden Metazeichen in Mustern:

Operator	Beschreibung
`%`	Entspricht einer Folge von 0 oder mehr Zeichen.
`_`	Entspricht einem beliebigen Zeichen.
`|`	Alternativen (eines der beiden angegebenen Teilmuster).
`*`	Wiederholt das vorangehende Element 0 oder mehr Male.
`+`	Wiederholt das vorangehende Element 1 oder mehr Male.
`?`	Wiederholt das vorangehende Element 0 oder 1 Mal.
`{m}`	Wiederholt das vorangehende Element genau m Male.
`{m,}`	Wiederholt das vorangehende Element m oder mehr Male.

Operator	Beschreibung
`{m,n}`	Wiederholt das vorangehende Element m bis n Male.
`()`	Klammern fassen Elemente zu einem logischen Element zusammen.
`[...]`	Ausdrücke in eckigen Klammern geben eine Zeichenklasse an, wie bei regulären Ausdrücken entsprechend dem POSIX-Standard.

Beispiele

In der folgenden Tabelle werden Beispiele für Patternmatching mit SIMILAR TO dargestellt.

Ausdruck	Rückgabewert
`'abc' SIMILAR TO 'abc'`	Wahr
`'abc' SIMILAR TO '_b_'`	True
`'abc' SIMILAR TO '_A_'`	False
`'abc' SIMILAR TO '%(b|d)%'`	True
`'abc' SIMILAR TO '(b|c)%'`	False
`'AbcAbcdefgefg12efgefg12' SIMILAR TO '((Ab)?c)+d((efg)+(12))+'`	True
`'aaaaaab11111xy' SIMILAR TO 'a{6}_[0-9]{5}(x|y){2}'`	True
`'$0.87' SIMILAR TO '$[0-9]+(.[0-9][0-9])?'`	Wahr

Das folgende Beispiel listet Städte auf, deren Name ein „E“ oder ein „H“ enthält:

```
SELECT DISTINCT city FROM users
WHERE city SIMILAR TO '%E%|%H%' ORDER BY city LIMIT 5;
```

```
      city
-----------------
 Agoura Hills
 Auburn Hills
 Benton Harbor
 Beverly Hills
 Chicago Heights
```

In dem folgenden Beispiel wird die Standard-Escape-Zeichenfolge (\\) verwendet, um nach Zeichenfolgen zu suchen, die den Unterstrich (_) enthalten.

```
SELECT tablename, "column" FROM pg_table_def
WHERE "column" SIMILAR TO '%start\\_%'
ORDER BY tablename, "column" LIMIT 5;

        tablename         |       column
--------------------------+---------------------
 stcs_abort_idle          | idle_start_time
 stcs_abort_idle          | txn_start_time
 stcs_analyze_compression | start_time
 stcs_auto_worker_levels  | start_level
 stcs_auto_worker_levels  | start_wlm_occupancy
```

In dem folgenden Beispiel wird als Escape-Zeichenfolge ^ verwendet, und dann wird nach Zeichenfolgen gesucht, die den Unterstrich (_) enthalten.

```
SELECT tablename, "column" FROM pg_table_def
WHERE "column" SIMILAR TO '%start^_%' ESCAPE '^'
ORDER BY tablename, "column" LIMIT 5;

        tablename         |       column
--------------------------+---------------------
 stcs_abort_idle          | idle_start_time
 stcs_abort_idle          | txn_start_time
 stcs_analyze_compression | start_time
 stcs_auto_worker_levels  | start_level
 stcs_auto_worker_levels  | start_wlm_occupancy
```

POSIX-Operatoren

Ein regulärer POSIX-Ausdruck ist eine Zeichenfolge, die ein Übereinstimmungsmuster angibt. Eine Zeichenfolge entspricht einem regulären Ausdruck, wenn sie Teil der regulären Menge ist, die durch den regulären Ausdruck beschrieben wird.

Reguläre Ausdrücke entsprechend der POSIX-Spezifikation ermöglichen eine feinere Modellierung von Mustern als die Operatoren LIKE und SIMILAR TO. Bei regulären Ausdrücken entsprechend dem POSIX-Standard kann das Muster einer beliebigen Teilzeichenfolge entsprechen. Der Operator SIMILAR TO hingegen gibt „wahr“ genau dann zurück, wenn das Muster der gesamten Zeichenfolge entspricht.

> **Note**
>
> Patternmatching mit regulären Ausdrücken unter Verwendung von POSIX-Operatoren erfordert einen hohen Rechenaufwand. Wir empfehlen, nach Möglichkeit LIKE zu verwenden, insbesondere, wenn eine große Anzahl an Zeilen verarbeitet werden muss. Ein Beispiel: Die folgenden Abfragen sind funktional identisch, aber die Abfrage mit LIKE wird 7-mal schneller ausgeführt als die Abfrage mit einem regulären Ausdruck:
>
> ```
> select count(*) from event where eventname ~ '.*(Ring|Die).*';
> select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die
> %';
> ```

Syntax

```
expression [ ! ] ~ pattern
```

Argumente

expression

Ein gültiger UTF8-Zeichenfolgenausdruck (beispielsweise ein Spaltenname).

!

Negationsoperator. Entspricht nicht dem regulären Ausdruck.

~

Führt eine Suche über beliebige Teilzeichenfolgen in dem Ausdruck expression und unter Berücksichtigung der Groß-/Kleinschreibung durch.

> **Note**
>
> Der Operator ~~ ist ein Synonym für LIKE.

pattern

Ein Zeichenfolgenliteral, das ein Muster für reguläre Ausdrücke darstellt.

Wenn das Muster pattern keine Platzhalterzeichen enthält, wird die Zeichenfolge selbst als Muster verwendet.

Um nach Zeichenfolgen zu suchen, die Metazeichen, beispielsweise `. * | ?` usw. enthalten, müssen Sie dieses Zeichen mit zwei umgekehrten Schrägstrichen (`\\`) als Literal kennzeichnen. Im Gegensatz zu `SIMILAR TO` und `LIKE` unterstützen reguläre Ausdrücke entsprechend dem POSIX-Standard keine benutzerdefinierten Escape-Zeichenfolgen.

Die Zeichenfolgenausdrücke können vom Datentyp CHAR oder VARCHAR sein. Wenn unterschiedliche Datentypen verwendet werden, konvertiert Amazon Redshift pattern in den Datentyp des Ausdrucks expression.

Alle Zeichenfolgenausdrücke können vom Datentyp CHAR oder VARCHAR sein. Wenn in den Ausdrücken unterschiedliche Datentypen verwendet werden, konvertiert Amazon Redshift sie in den Datentyp des Ausdrucks expression.

POSIX unterstützt die folgenden Metazeichen in Mustern:

POSIX	Beschreibung
.	Entspricht einem beliebigen Zeichen.
*	Findet 0 oder mehr Vorkommen.
+	Findet 1 oder mehr Vorkommen.

POSIX	Beschreibung
?	Findet 0 oder 1 Vorkommen.
|	Gibt alternative Übereinstimmungen an. Beispiel: `E | H` bedeutet E oder H.
^	Entspricht dem beginning-of-line Zeichen.
$	Entspricht dem end-of-line Zeichen.
$	Findet die Position am Ende der ganzen Zeichenfolge.
[]	In eckigen Klammern steht eine Musterliste, aus der ein Element übereinst immen muss. Ein der Liste vorangestelltes Caret-Zeichen (^) steht für eine Musterausschlussliste, die auf alle Zeichenfolgen außer den Ausdrücken in der Liste passt.
( )	Klammern fassen Elemente zu einem logischen Element zusammen.
{m}	Wiederholt das vorangehende Element genau m Male.
{m,}	Wiederholt das vorangehende Element m oder mehr Male.
{m,n}	Wiederholt das vorangehende Element m bis n Male.
[: :]	Passt auf alle Zeichen in einer POSIX-Zeichenklasse. In den folgenden Zeichenklassen unterstützt Amazon Redshift nur ASCII-Zeichen: `[:alnum:]` , `[:alpha:]` , `[:lower:]` , `[:upper:]`

Amazon Redshift unterstützt die folgenden POSIX-Zeichenklassen:

Zeichenklasse	Beschreibung
[[:alnum:]]	Alle alphanumerischen ASCII-Zeichen
[[:alpha:]]	Alle alphabetischen ASCII-Zeichen
[[:blank:]]	Alle Whitespace-Zeichen

Zeichenklasse	Beschreibung
`[[:cntrl:]]`	Alle Steuerzeichen (nicht druckbar)
`[[:digit:]]`	Alle Ziffern
`[[:lower:]]`	Alle ASCII-Kleinbuchstaben
`[[:punct:]]`	Alle Interpunktionszeichen
`[[:space:]]`	Alle Leerzeichen (nicht druckbar)
`[[:upper:]]`	Alle ASCII-Großbuchstaben
`[[:xdigit:]]`	Alle gültigen Hexadezimalzeichen

Amazon Redshift unterstützt in regulären Ausdrücken die folgenden Operatoren, die an Perl angelehnt sind. Wenn Sie den Operator nicht als Metazeichen verwenden möchten, müssen Sie ihn mit zwei umgekehrten Schrägstrichen () als Literal kennzeichnen. (‘\\’).

Operator	Beschreibung	Entsprechender Zeichenklassenausdruck
`\\d`	Eine Ziffer	`[[:digit:]]`
`\\D`	Ein Zeichen, das keine Ziffer ist	`[^[:digit:]]`
`\\w`	Ein Zeichen, das Bestandteil eines Worts sein kann	`[[:word:]]`
`\\W`	Ein Zeichen, das kein Bestandteil eines Worts sein kann	`[^[:word:]]`
`\\s`	Ein Whitespace-Zeichen	`[[:space:]]`
`\\S`	Ein Zeichen, das keinen Whitespace darstellt	`[^[:space:]]`
`\\b`	Ein Grenzwort	

Beispiele

In der folgenden Tabelle werden Beispiele für Patternmatching unter Verwendung von POSIX-Operatoren dargestellt:

Ausdruck	Rückgabewert
`'abc' ~ 'abc'`	Wahr
`'abc' ~ 'a'`	True
`'abc' ~ 'A'`	False
`'abc' ~ '.*(b|d).*'`	True
`'abc' ~ '(b|c).*'`	True
`'AbcAbcdefgefg12efgefg12' ~ '((Ab)?c)+d((efg)+(12))+'`	True
`'aaaaaab11111xy' ~ 'a{6}.[1]{5}(x|y){2}'`	True
`'$0.87' ~ '\\$[0-9]+(\\.[0-9][0-9])?'`	True
`'ab c' ~ '[[:space:]]'`	True
`'ab c' ~ '\\s'`	True
`' ' ~ '\\S'`	Falsch

Das folgende Beispiel listet Städte auf, deren Name ein `E` oder ein `H` enthält:

```
SELECT DISTINCT city FROM users
WHERE city ~ '.*E.*|.*H.*' ORDER BY city LIMIT 5;

      city
-----------------
 Agoura Hills
 Auburn Hills
```

```
 Benton Harbor
 Beverly Hills
 Chicago Heights
```

Das folgende Beispiel listet Städte auf, deren Name weder E noch H enthält:

```
SELECT DISTINCT city FROM users WHERE city !~ '.*E.*|.*H.*' ORDER BY city LIMIT 5;

      city
-----------------
 Aberdeen
 Abilene
 Ada
 Agat
 Agawam
```

In dem folgenden Beispiel wird die Escape-Zeichenfolge (\\) verwendet, um nach Zeichenfolgen zu suchen, die den Punkt (.) enthalten.

```
SELECT venuename FROM venue
WHERE venuename ~ '.*\\..*'
ORDER BY venueid;

          venuename
------------------------------
 St. Pete Times Forum
 Jobing.com Arena
 Hubert H. Humphrey Metrodome
 U.S. Cellular Field
 Superpages.com Center
 E.J. Nutter Center
 Bernard B. Jacobs Theatre
 St. James Theatre
```

BETWEEN-Bereichsbedingung

Eine BETWEEN-Bedingung überprüft, ob Ausdrücke Elemente aus einem Bereich von Werten enthalten, der über die Schlüsselwörter BETWEEN und AND angegeben wird.

Syntax

```
expression [ NOT ] BETWEEN expression AND expression
```

Der Datentyp der Ausdrücke kann ein numerischer, ein Zeichen- oder ein Datum/Uhrzeit-Typ sein, die Typen müssen jedoch untereinander kompatibel sein. Der angegebene Bereich versteht sich inklusive der angegebenen Werte.

Beispiele

Im ersten Beispiel werden die Transaktionen, bei denen 2, 3, oder 4 Tickets verkauft wurden, gezählt:

```
select count(*) from sales
where qtysold between 2 and 4;

count
--------
104021
(1 row)
```

Bei der Bereichsbedingung werden die Anfangs- und Endwerte mitgezählt (inklusiver Bereich).

```
select min(dateid), max(dateid) from sales
where dateid between 1900 and 1910;

min  | max
-----+-----
1900 | 1910
```

Bei einer Bereichsbedingung muss der erste Wert stets der kleinere und der zweite der größere sein. In dem folgenden Beispiel werden immer 0 Zeilen zurückgegeben, weil die Werte in dem Bedingungsausdruck vertauscht wurden:

```
select count(*) from sales
where qtysold between 4 and 2;

count
-------
0
(1 row)
```

Wenn die Bedingung mit NOT negiert wird, werden nicht 0, sondern alle Zeilen gezählt:

```
select count(*) from sales
where qtysold not between 4 and 2;

count
--------
172456
(1 row)
```

Die folgende Abfrage gibt eine Liste der Events mit 20.000 bis 50.000 Plätzen zurück:

```
select venueid, venuename, venueseats from venue
where venueseats between 20000 and 50000
order by venueseats desc;

venueid |       venuename               | venueseats
---------+-------------------------------+------------
116 | Busch Stadium                 |      49660
106 | Rangers BallPark in Arlington |      49115
96 | Oriole Park at Camden Yards   |      48876
...
(22 rows)
```

Das folgende Beispiel zeigt die Verwendung von BETWEEN für Datumswerte:

```
select salesid, qtysold, pricepaid, commission, saletime
from sales
where eventid between 1000 and 2000
   and saletime between '2008-01-01' and '2008-01-03'
order by saletime asc;

salesid | qtysold | pricepaid | commission |   saletime
--------+---------+-----------+------------+---------------
  65082 |       4 |       472 |       70.8 | 1/1/2008 06:06
 110917 |       1 |       337 |      50.55 | 1/1/2008 07:05
 112103 |       1 |       241 |      36.15 | 1/2/2008 03:15
 137882 |       3 |      1473 |     220.95 | 1/2/2008 05:18
  40331 |       2 |        58 |        8.7 | 1/2/2008 05:57
 110918 |       3 |      1011 |     151.65 | 1/2/2008 07:17
  96274 |       1 |       104 |       15.6 | 1/2/2008 07:18
 150499 |       3 |       135 |      20.25 | 1/2/2008 07:20
  68413 |       2 |       158 |       23.7 | 1/2/2008 08:12
```

Beachten Sie, dass sich der BETWEEN-Bereich zwar inklusive der angegebenen Werte versteht, die Datumsangaben jedoch standardmäßig einen Zeitwert von 00:00:00 haben. Die einzige gültige Zeile für 3. Januar bei der Beispielabfrage wäre eine Zeile mit der Saletime (Verkaufszeit) `1/3/2008 00:00:00`.

„Null“-Bedingung

Bei der „Null“-Bedingung wird eine Überprüfung auf „Null“-Werte durchgeführt, wenn ein Wert fehlt oder unbekannt ist.

Syntax

```
expression IS [ NOT ] NULL
```

Argumente

expression

Ein Ausdruck, beispielsweise eine Spalte.

IS NULL

Gibt „wahr“ zurück, wenn der Wert des Ausdrucks „Null“ ist, und „falsch“, wenn der Ausdruck einen Wert hat.

IS NOT NULL

Gibt „falsch“ zurück, wenn der Wert des Ausdrucks „Null“ ist, und „wahr“, wenn der Ausdruck einen Wert hat.

Beispiel

Dieses Beispiel gibt an, wie oft die Tabelle SALES im Feld QTYSOLD „Null“ enthält:

```
select count(*) from sales
where qtysold is null;
count
-------
0
(1 row)
```

EXISTS-Bedingung

Die EXISTS-Bedingung überprüft, ob eine Unterabfrage Zeilen zurückgibt, und gibt „wahr“ zurück, wenn die Unterabfrage mindestens eine Zeile zurückgibt. Bei Voranstellung von NOT wird gibt die Bedingung „wahr“ zurück, wenn die Unterabfrage 0 Zeilen zurückgibt.

Syntax

```
[ NOT ] EXISTS (table_subquery)
```

Argumente

EXISTS

Ist „wahr“, wenn die Unterabfrage table_subquery wenigstens eine Zeile zurückgibt.

NOT EXISTS

Ist „wahr“, wenn die Unterabfrage table_subquery keine Zeilen zurückgibt.

table_subquery

Eine Unterabfrage, die zu einer Tabelle mit einer oder mehreren Spalten und einer oder mehreren Zeilen ausgewertet wird.

Beispiel

In diesem Beispiel werden nacheinander die Identifier für jedes Datum aufgelistet, an dem ein Verkauf stattgefunden hat:

```
select dateid from date
where exists (
select 1 from sales
where date.dateid = sales.dateid
)
order by dateid;

dateid
--------
1827
1828
1829
```

```
...
```

IN-Bedingung

Eine IN-Bedingung überprüft, ob ein Wert Element aus einer Menge von Werten oder aus einer Unterabfrage ist.

Syntax

```
expression [ NOT ] IN (expr_list | table_subquery)
```

Argumente

expression

Ein numerischer, Zeichen- oder Datum/Uhrzeit-Ausdruck, der anhand der Ausdrucksliste expr_list oder der Unterabfrage table_subquery ausgewertet wird, und der mit dem Datentyp der Liste bzw. Abfrage kompatibel sein muss.

expr_list

Ein oder mehrere kommagetrennte Ausdrücke oder ein oder mehrere Mengen von kommagetrennten Ausdrücken, als Klammerausdruck.

table_subquery

Eine Unterabfrage, die zu einer Tabelle mit einer oder mehreren Zeilen ausgewertet wird, aber höchstens eine Spalte in ihrer SELECT-Liste enthält.

IN | NOT IN

In gibt „wahr“ zurück, wenn der Ausdruck Element der Ausdrucksliste oder der Abfrage ist. NOT IN gibt „wahr“ zurück, wenn der Ausdruck darin nicht enthalten ist. IN und NOT IN geben NULL und keine Zeilen zurück, wenn der Ausdruck expression zu „Null“ ausgewertet wird, oder wenn in der Ausdrucksliste expr_list bzw. der Unterabfrage table_subquery keine übereinstimmenden Werte gefunden wurden und mindestens eine der verglichenen Zeilen als Ergebnis „Null“ zurückgegeben hat.

Beispiele

Die folgenden Bedingungen sind nur für die aufgelisteten Werte wahr:

```
qtysold in (2, 4, 5)
date.day in ('Mon', 'Tues')
date.month not in ('Oct', 'Nov', 'Dec')
```

Optimierung bei großen IN-Listen

Um die Abfrageleistung zu optimieren, werden IN-Listen mit mehr als 10 Werten intern als Zahlenarray ausgewertet. IN-Listen mit weniger Werten werden als Reihe von OR-Prädikaten ausgewertet. Diese Optimierung wird für die Datentypen SMALLINT, INTEGER, BIGINT, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP und TIMESTAMPTZ unterstützt.

Den Effekt dieser Optimierung verdeutlicht die Ausgabe, wenn ein EXPLAIN über der Abfrage ausgeführt wird. Beispiel:

```
explain select * from sales
QUERY PLAN
------------------------------------------------------------------------
XN Seq Scan on sales  (cost=0.00..6035.96 rows=86228 width=53)
Filter: (salesid = ANY ('{1,2,3,4,5,6,7,8,9,10,11}'::integer[]))
(2 rows)
```

SQL-Befehle

Die SQL-Sprache besteht aus Befehlen, die Sie zum Erstellen und Bearbeiten von Datenbankobjekten, zum Ausführen von Abfragen, zum Laden von Tabellen und zum Ändern von Daten in Tabellen verwenden.

Amazon Redshift basiert auf PostgreSQL. Zwischen Amazon Redshift und PostgreSQL gibt es eine Reihe wichtiger Unterschiede, die Sie berücksichtigen müssen, wenn Sie Ihre Data-Warehouse-Anwendungen entwerfen und entwickeln. Weitere Informationen zu den Unterschieden zwischen Amazon-Redshift-SQL und PostgreSQL finden Sie unter Amazon Redshift und PostgreSQL.

> **Note**
>
> Die maximal zulässige Größe für eine einzelne SQL-Anweisung ist 16 MB.

Themen

- ABORT
- ALTER DATABASE
- ALTER DATASHARE
- ALTER DEFAULT PRIVILEGES
- ALTER EXTERNAL VIEW (Vorschau)
- ALTER FUNCTION
- ALTER GROUP
- ALTER IDENTITY PROVIDER
- ALTER MASKING POLICY
- ALTER MATERIALIZED VIEW
- ALTER RLS POLICY
- ALTER ROLE
- ALTER PROCEDURE
- ALTER SCHEMA
- ALTER SYSTEM
- ALTER TABLE
- ALTER TABLE APPEND
- ALTER USER
- ANALYZE
- ANALYZE COMPRESSION
- ANFÜGEN EINER MASKIERUNGSRICHTLINIE
- ATTACH RLS POLICY
- BEGIN
- CALL
- CANCEL
- CLOSE
- COMMENT
- COMMIT
- COPY

- CREATE DATABASE
- DATASHARE ERSTELLEN
- CREATE EXTERNAL FUNCTION
- CREATE EXTERNAL SCHEMA
- CREATE EXTERNAL TABLE
- CREATE EXTERNAL VIEW (Vorschau)
- CREATE FUNCTION
- CREATE GROUP
- CREATE IDENTITY PROVIDER
- CREATE LIBRARY
- ERSTELLEN EINER MASKIERUNGSRICHTLINIE
- CREATE MATERIALIZED VIEW
- CREATE MODEL
- CREATE PROCEDURE
- CREATE RLS POLICY
- CREATE ROLE
- CREATE SCHEMA
- CREATE TABLE
- CREATE TABLE AS
- CREATE USER
- CREATE VIEW
- DEALLOCATE
- DECLARE
- DELETE
- DESC DATASHARE
- DESC IDENTITY PROVIDER
- TRENNEN EINER MASKIERUNGSRICHTLINIE
- DETACH RLS POLICY
- DROP DATABASE

- DROP DATASHARE
- DROP EXTERNAL VIEW (Vorschau)
- DROP FUNCTION
- DROP GROUP
- DROP IDENTITY PROVIDER
- DROP LIBRARY
- ENTFERNEN EINER MASKIERUNGSRICHTLINIE
- DROP MODEL
- DROP MATERIALIZED VIEW
- DROP PROCEDURE
- DROP RLS POLICY
- DROP ROLE
- DROP SCHEMA
- DROP TABLE
- DROP USER
- DROP VIEW
- END
- EXECUTE
- EXPLAIN
- FETCH
- GRANT
- INSERT
- INSERT (externe Tabelle)
- LOCK
- MERGE
- PREPARE
- REFRESH MATERIALIZED VIEW
- RESET
- REVOKE

- ROLLBACK
- SELECT
- SELECT INTO
- SET
- SET SESSION AUTHORIZATION
- SET SESSION CHARACTERISTICS
- ZEIGEN
- SHOW_COLUMNS
- SHOW EXTERNAL TABLE
- SHOW DATABASES
- SHOW MODEL
- SHOW DATASHARES
- SHOW PROCEDURE
- SHOW SCHEMAS
- SHOW TABLE
- SHOW TABLES
- SHOW VIEW
- START TRANSACTION
- TRUNCATE
- UNLOAD
- UPDATE
- VACUUM

ABORT

Bricht die zurzeit ausgeführte Transaktion ab und verwirft alle Aktualisierungen, die durch diese Transaktion ausgeführt wurden. ABORT wirkt sich nicht auf bereits abgeschlossene Transaktionen aus.

Dieser Befehl hat die gleiche Funktion wie der Befehl ROLLBACK. Weitere Informationen finden Sie unter ROLLBACK.

Syntax

```
ABORT [ WORK | TRANSACTION ]
```

Parameter

WORK

Optionales Schlüsselwort.

TRANSACTION

Optionales Schlüsselwort; WORK und TRANSACTION sind Synonyme.

Beispiel

Im folgenden Beispiel werden eine Tabelle erstellt und eine Transaktion gestartet, bei der Daten in die Tabelle eingefügt werden. Anschließend wird mit dem Befehl ABORT die Dateieinfügung zurückgenommen, um die Tabelle leer zu lassen.

Mit dem folgenden Befehl wird eine Beispieltabelle namens MOVIE_GROSS erstellt:

```
create table movie_gross( name varchar(30), gross bigint );
```

Mit dem nächsten Satz von Befehlen wird eine Transaktion gestartet, die zwei Datenzeilen in die Tabelle einfügt:

```
begin;

insert into movie_gross values ( 'Raiders of the Lost Ark', 23400000);

insert into movie_gross values ( 'Star Wars', 10000000 );
```

Als Nächstes werden mit dem folgenden Befehl die Daten aus der Tabelle ausgewählt, um zu zeigen, dass sie erfolgreich eingefügt wurden:

```
select * from movie_gross;
```

Die Befehlsausgabe zeigt, dass beide Zeilen erfolgreich eingefügt wurden:

```
         name           |  gross
------------------------+----------
Raiders of the Lost Ark | 23400000
Star Wars               | 10000000
(2 rows)
```

Mit diesem Befehl werden nun die Datenänderungen auf den Zeitpunkt zurückgesetzt, an dem die Transaktion gestartet wurde:

```
abort;
```

Wenn nun Daten aus der Tabelle ausgewählt werden, wird eine leere Tabelle gezeigt:

```
select * from movie_gross;

 name | gross
------+-------
(0 rows)
```

ALTER DATABASE

Ändert die Attribute einer Datenbank.

Erforderliche Berechtigungen

Um ALTER DATABASE zu benutzen, ist eine der folgenden Berechtigungen erforderlich.

- Superuser
- Benutzer mit der Berechtigung ALTER DATABASE.
- Datenbankbesitzer

Syntax

```
ALTER DATABASE database_name
{ RENAME TO new_name
| OWNER TO new_owner
| CONNECTION LIMIT { limit | UNLIMITED }
| COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }
| ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT }
```

```
| INTEGRATION REFRESH {{ ALL | INERROR } TABLES [IN SCHEMA schema [, ...]] |
 TABLE schema.table [, ...]}
}
```

Parameter

database_name

Der Name der Datenbank, die geändert werden soll. In der Regel ändern Sie eine Datenbank, mit der Sie zurzeit nicht verbunden sind. In jedem Fall werden die Änderungen nur in folgenden Sitzungen wirksam. Sie können den Besitzer der aktuellen Datenbank ändern, sie jedoch nicht umbenennen:

```
alter database tickit rename to newtickit;
ERROR:  current database may not be renamed
```

RENAME TO

Benennt die angegebene Datenbank um. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen. Sie können die Datenbanken dev, padb_harvest, template0 oder template1 oder sys:internal-Datenbanken nicht umbenennen. Nur der Datenbankbesitzer oder ein superuser (p. 925) können eine Datenbank umbenennen. Besitzer, die keine Superuser sind, benötigen ebenfalls das CREATEDB-Recht.

new_name

Neuer Datenbankname.

OWNER TO

Ändert den Besitzer der angegebenen Datenbank. Sie können den Besitzer der aktuellen Datenbank oder einer anderen Datenbank ändern. Nur ein Superuser kann den Besitzer ändern.

new_owner

Neuer Datenbankbesitzer. Der neue Besitzer muss ein vorhandener Datenbankbenutzer mit Schreibberechtigungen sein. Weitere Informationen zu Benutzerrechten finden Sie in GRANT.

CONNECTION LIMIT { Limit | UNLIMITED }

Redshift unterstützt das Schreiben von verschachteltem JSON-Code, wenn das Abfrageergebnis SUPER-Spalten enthält. Das Limit wird für Superuser nicht durchgesetzt. Mithilfe des Schlüsselworts UNLIMITED können Sie die maximale Zahl gleichzeitiger Verbindungen festlegen.

Möglicherweise gilt auch ein Limit für die Zahl der Verbindungen für die einzelnen Benutzer. Weitere Informationen finden Sie unter CREATE USER. Der Standardwert ist UNLIMITED. Um die aktuellen Verbindungen anzuzeigen, führen Sie eine Abfrage für die Systemansicht STV_SESSIONS aus.

> **Note**
>
> Wenn sowohl für Benutzer- als auch für Datenbankverbindungen Limits gelten, muss ein ungenutzter Verbindungsplatz verfügbar sein, der sich innerhalb beider Grenzen befindet, wenn ein Benutzer versucht, eine Verbindung herzustellen.

COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }

Eine Klausel, die angibt, ob bei der Suche oder dem Vergleich von Zeichenfolgen zwischen Groß- und Kleinschreibung unterschieden wird oder nicht.

Sie können die Unterscheidung zwischen Groß- und Kleinschreibung in der aktuellen, leeren Datenbank ändern.

Sie müssen über die Berechtigung für die aktuelle Datenbank verfügen, um die Unterscheidung zwischen Groß-/Kleinschreibung zu ändern. Superuser oder Datenbankbesitzer mit dem CREATE DATABASE-Recht können auch die Unterscheidung zwischen Groß-/Kleinschreibung in der Datenbank ändern.

ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT }

Eine Klausel, die die verwendete Isolationsstufe bei Abfragen für eine Datenbank angibt.

- SERIALIZABLE-Isolation (serialisierbare Isolation) – bietet volle Serialisierbarkeit für gleichzeitige Transaktionen. Weitere Informationen finden Sie unter Serialisierbare Isolierung.
- SNAPSHOT-Isolation – bietet eine Isolationsstufe mit Schutz vor Aktualisierungs- und Löschkonflikten

Weitere Hinweise zu Isolationsstufen finden Sie unter CREATE DATABASE.

Beachten Sie die folgenden Elemente, wenn Sie die Isolationsstufe einer Datenbank ändern:

- Sie müssen über Superuser-Rechte oder über die Berechtigung CREATE DATABASE (Datenbank erstellen) für die aktuelle Datenbank verfügen, um die Isolationsstufe für die Datenbank zu ändern.
- Sie können die Isolationsstufe der `dev`-Datenbank nicht ändern.

- Sie können die Isolationsstufe innerhalb eines Transaktionsblocks nicht ändern.
- Der Befehl zum Ändern der Isolationsstufe schlägt fehl, wenn andere Benutzer mit der Datenbank verbunden sind.
- Mit dem Befehl zum Ändern der Isolationsstufe können die Isolationsstufen-Einstellungen der aktuellen Sitzung geändert werden.

INTEGRATION REFRESH {{ ALL | INERROR } TABLES [IN SCHEMA schema [, ...]] | TABLE schema.table [, ...]}

Eine Klausel, die angibt, ob Amazon Redshift alle Tabellen oder Tabellen mit Fehlern im angegebenen Schema oder der Tabelle aktualisiert. Die Aktualisierung löst aus, dass die Tabellen im angegebenen Schema oder der angegebenen Tabelle vollständig aus der Quelldatenbank repliziert werden.

Weitere Informationen finden Sie unter Arbeiten mit Null-ETL-Integrationen im Amazon-Redshift-Verwaltungshandbuch. Weitere Informationen zu Integrationszuständen finden Sie unter SVV_INTEGRATION_TABLE_STATE und SVV_INTEGRATION.

Nutzungshinweise

ALTER DATABASE-Befehle gelten für folgende, nicht für aktuelle Sitzungen. Sie müssen die Verbindung mit der geänderten Datenbank erneut herstellen, um die Auswirkungen der Änderung anzuzeigen.

Beispiele

Im folgenden Beispiel wird die Datenbank mit dem Namen TICKIT_SANDBOX in TICKIT_TEST umbenannt:

```
alter database tickit_sandbox rename to tickit_test;
```

Im folgenden Beispiel wird der Besitzer der TICKIT-Datenbank (der aktuellen Datenbank) in DWUSER umbenannt:

```
alter database tickit owner to dwuser;
```

Im folgenden Beispiel wird die Unterscheidung zwischen Groß-/Kleinschreibung in der Datenbank sampledb geändert:

```
ALTER DATABASE sampledb COLLATE CASE_INSENSITIVE;
```

Im folgenden Beispiel wird eine Datenbank namens **sampledb** mit der Isolationsstufe SNAPSHOT geändert.

```
ALTER DATABASE sampledb ISOLATION LEVEL SNAPSHOT;
```

Im folgenden Beispiel werden die Tabellen **sample_table1** und **sample_table2** in der Datenbank **sample_integration_db** in Ihrer Null-ETL-Integration aktualisiert.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH TABLES sample_table1,
 sample_table2;
```

Im folgenden Beispiel werden alle synchronisierten und fehlgeschlagenen Tabellen in Ihrer Null-ETL-Integration aktualisiert.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH ALL tables;
```

Im folgenden Beispiel werden alle Tabellen aktualisiert, die sich im `ErrorState` im Schema **sample_schema** befinden.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH INERROR TABLES in SCHEMA
 sample_schema;
```

ALTER DATASHARE

Ändert die Definition eines Datashares. Sie können mit ALTER DATASHARE Objekte hinzufügen oder entfernen. Sie können nur ein Datashare in der aktuellen Datenbank ändern. Fügen Sie Objekte aus der zugehörigen Datenbank einem Datashare hinzu oder entfernen Sie sie daraus. Der Besitzer des Datashares mit den erforderlichen Berechtigungen für die hinzuzufügenden oder zu entfernenden Datashare-Objekte kann das Datashare ändern.

Erforderliche Berechtigungen

Für ALTER DATASHARE sind folgende Berechtigungen erforderlich:

- Superuser.

- Benutzer mit der Berechtigung ALTER DATASHARE.
- Benutzer mit der Berechtigung ALTER oder ALL für das Datashare.
- Um bestimmte Objekte zu einem Datashare hinzuzufügen, müssen diese Benutzer die Berechtigung für die Objekte haben. Für diesen Fall sollten Benutzer auch Besitzer der Objekte sein oder SELECT-, USAGE- oder ALL-Berechtigungen für die Objekte haben.

Syntax

Die folgende Syntax veranschaulicht, wie Objekte zum Datashare hinzugefügt oder von diesem entfernt werden.

```
ALTER DATASHARE datashare_name { ADD | REMOVE } {
TABLE schema.table [, ...]
| SCHEMA schema [, ...]
| FUNCTION schema.sql_udf (argtype,...) [, ...]
| ALL TABLES IN SCHEMA schema [, ...]
| ALL FUNCTIONS IN SCHEMA schema [, ...] }
```

Die folgende Syntax veranschaulicht, wie die Eigenschaften eines Datashares konfiguriert werden.

```
ALTER DATASHARE datashare_name {
[ SET PUBLICACCESSIBLE [=] TRUE | FALSE ]
[ SET INCLUDENEW [=] TRUE | FALSE FOR SCHEMA schema ] }
```

Parameter

datashare_name

Der Name der zu ändernden Datashares.

ADD | REMOVE

Eine Klausel, die angibt, ob dem Datashare Objekte hinzugefügt oder daraus entfernt werden sollen.

TABLE schema.table [, ...]

Der Name der Tabelle oder Ansicht im angegebenen Schema, die dem Datashare hinzugefügt werden soll.

SCHEMA schema [, ...]

Der Name des Schemas, das dem Datashare hinzugefügt werden soll.

FUNCTION schema.sql_udf (argtype,...) [, ...]

Der Name der benutzerdefinierten SQL-Funktion mit Argumenttypen, die dem Datashare hinzugefügt werden soll.

ALL TABLES IN SCHEMA schema [, ...]

Eine Klausel, die angibt, ob alle Tabellen und Ansichten im angegebenen Schema zum Datashare hinzugefügt werden sollen.

ALL FUNCTIONS IN SCHEMA schema [, ...] }

Eine Klausel, die das Hinzufügen aller Funktionen im angegebenen Schema zum Datashare festlegt.

[SET PUBLICACCESSIBLE [=] TRUE | FALSE]

Eine Klausel, die angibt, ob ein Datashare für öffentlich zugängliche Cluster freigegeben werden kann.

[SET INCLUDENEW [=] TRUE | FALSE FOR SCHEMA schema]

Eine Klausel, die angibt, ob zukünftige Tabellen, Ansichten oder benutzerdefinierte SQL-Funktionen (UDFs), die in dem angegebenen Schema erstellt wurden, dem Datashare hinzugefügt werden sollen. Aktuelle Tabellen, Ansichten oder SQL UDFs im angegebenen Schema werden dem Datashare nicht hinzugefügt. Nur Superuser können diese Eigenschaft für jedes Datashare-Schema-Paar ändern. Standardmäßig lautet die INCLUDENEW-Klausel „false“.

Nutzungshinweise für ALTER DATASHARE

- Die folgenden Benutzer können ein Datashare ändern:
  - Ein Superuser
  - Der Besitzer des Datashares
  - Benutzer mit ALTER- oder ALL-Privilegien für das Datashare
- Um bestimmte Objekte zu einem Datashare hinzuzufügen, müssen Benutzer die richtigen Berechtigungen für die Objekte haben. Benutzer sollten auch Besitzer der Objekte sein oder SELECT, USAGE- oder ALL-Berechtigungen für die Objekte haben.

- Sie können Schemata, Tabellen, reguläre Ansichten, spätbindende Ansichten, materialisierte Ansichten und benutzerdefinierte SQL-Funktionen (UDFs) freigeben. Fügen Sie einem Datashare zuerst ein Schema hinzu, bevor Sie Objekte im Schema hinzufügen.

  Wenn Sie ein Schema hinzufügen, fügt Amazon Redshift nicht alle untergeordneten Objekte hinzu. Sie müssen sie explizit hinzufügen.
- Wir empfehlen, dass Sie AWS Data Exchange Datashares mit aktivierter Einstellung für öffentlich zugänglich erstellen.
- Im Allgemeinen empfehlen wir, eine AWS Data Exchange Datenfreigabe nicht mit der ALTER DATASHARE-Anweisung zu ändern, um den öffentlichen Zugriff zu deaktivieren. Wenn Sie dies tun, verlieren die Benutzer, AWS-Konten die Zugriff auf die Datenfreigabe haben, den Zugriff, wenn ihre Cluster öffentlich zugänglich sind. Diese Art der Änderung kann außerdem zu einer Verletzung der Datenproduktbedingungen in AWS Data Exchange führen. Eine Ausnahme dieser Empfehlung wird im Folgenden erklärt.

  Das folgende Beispiel zeigt einen Fehler, wenn ein AWS Data Exchange Datashare mit deaktivierter Einstellung erstellt wird.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
ERROR:  Alter of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value 'c670ba4db22f4b'
```

  Damit das Ändern einer AWS Data Exchange Datenfreigabe zur Deaktivierung der öffentlich zugänglichen Einstellung möglich ist, legen Sie die folgende Variable fest und führen Sie die ALTER DATASHARE-Anweisung erneut aus.

```
SET datashare_break_glass_session_var to 'c670ba4db22f4b';
```

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

  In diesem Fall generiert Amazon Redshift einen zufälligen Einmalwert zur Festlegung der Sitzungsvariable, um ALTER DATASHARE SET PUBLICACCESSIBLE FALSE für ein AWS Data Exchange -Datashare zu erlauben.

Beispiele

Im folgenden Beispiel wird das Schema zum Datashare hinzugefügt. `public` `salesshare`

```
ALTER DATASHARE salesshare ADD SCHEMA public;
```

Im folgenden Beispiel wird die Tabelle `public.tickit_sales_redshift` zum Datashare `salesshare` hinzugefügt.

```
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
```

Im folgenden Beispiel werden alle Tabellen zum Datashare `salesshare` hinzugefügt.

```
ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
```

Im folgenden Beispiel wird die Tabelle `public.tickit_sales_redshift` vom Datashare `salesshare` entfernt.

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

ALTER DEFAULT PRIVILEGES

Definiert den Standardsatz von Zugriffsberechtigungen, die auf Objekte angewendet werden sollen, die in future vom angegebenen Benutzer erstellt werden. Standardmäßig können Benutzer nur ihre eigenen Standardzugriffsrechte ändern. Nur ein Superuser kann Standardberechtigungen für andere Benutzer angeben.

Sie können Standardrechte auf Rollen, Benutzer oder Benutzergruppen anwenden. Sie können Standardberechtigungen global für alle Objekte festlegen, die in der aktuellen Datenbank erstellt wurden, oder für Objekte, die nur in den angegebenen Schemas erstellt wurden.

Standardberechtigungen gelten nur für neue Objekte. Durch das Ausführen von ALTER DEFAULT PRIVILEGES werden die Berechtigungen für bestehende Objekte nicht geändert. Informationen zum Erteilen von Berechtigungen für alle aktuellen und future Objekte, die von einem beliebigen Benutzer in einer Datenbank oder einem Schema erstellt wurden, finden Sie unter Bereichsberechtigungen.

Um Informationen zu den Standardrechten für Datenbankbenutzer anzuzeigen, führen Sie eine Abfrage für die Systemkatalogtabelle PG_DEFAULT_ACL aus.

Weitere Informationen zu Rechten finden Sie in GRANT.

Erforderliche Berechtigungen

Im Folgenden sind die erforderlichen Berechtigungen für ALTER DEFAULT PRIVILEGES aufgeführt:

- Superuser
- Benutzer mit der Berechtigung ALTER DEFAULT PRIVILEGES
- Benutzer, die ihre eigenen Standardzugriffsberechtigungen ändern
- Benutzer, die Berechtigungen für Schemas festlegen, auf die sie Zugriffsberechtigungen haben

Syntax

```
ALTER DEFAULT PRIVILEGES
    [ FOR USER target_user [, ...] ]
    [ IN SCHEMA schema_name [, ...] ]
    grant_or_revoke_clause

where grant_or_revoke_clause is one of:

GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | TRUNCATE } [,...] |
 ALL [ PRIVILEGES ] }
 ON TABLES
 TO { user_name [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 TO { user_name [ WITH GRANT OPTION ] |  ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 TO { user_name [ WITH GRANT OPTION ] |  ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

REVOKE [ GRANT OPTION FOR ] { { SELECT | INSERT | UPDATE | DELETE | REFERENCES |
 TRUNCATE } [,...] | ALL [ PRIVILEGES ] }
 ON TABLES
 FROM user_name [, ...] [ RESTRICT ]

REVOKE  { { SELECT | INSERT | UPDATE | DELETE | REFERENCES | TRUNCATE } [,...] | ALL
 [ PRIVILEGES ] }
 ON TABLES
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ] { EXECUTE | ALL [ PRIVILEGES ] }
```

```
 ON FUNCTIONS
 FROM user_name [, ...] [ RESTRICT ]

REVOKE { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ] { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 FROM user_name [, ...] [ RESTRICT ]

REVOKE { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]
```

Parameter

FOR USER target_user

Optional. Der Name des Benutzers, für den Standardrechte definiert sind. Nur Superuser können Standardrechte für andere Benutzer ändern. Der Standardwert ist der aktuelle Benutzer.

IN SCHEMA schema_name

Optional. Wenn eine IN SCHEMA-Klausel angezeigt wird, gelten die angegebenen Standardrechte für neue Objekte, die im angegebenen schema_name erstellt werden. In diesem Fall müssen der Benutzer oder die Benutzergruppe, die das Ziel von ALTER DEFAULT PRIVILEGES ist, das CREATE-Recht für das angegebene Schema besitzen. Standardrechte, die für ein Schema spezifisch sind, werden vorhandenen globalen Standardrechten hinzugefügt. Standardmäßig werden Standardrechte global auf die gesamte Datenbank angewendet.

GRANT

Der Satz von Rechten, die den angegebenen Benutzern oder Gruppen für alle neuen Tabellen und Ansichten, Funktionen oder gespeicherten Prozeduren gewährt werden sollen, die vom angegebenen Benutzer erstellt wurden. Sie können mit der GRANT-Klausel die gleichen Rechte und Optionen wie mit dem Befehl GRANT festlegen.

WITH GRANT OPTION

Eine Klausel, die angibt, dass der Benutzer, der die Rechte erhält, anderen Benutzern die gleichen Rechte gewähren kann. Sie können einer Gruppe oder PUBLIC keine Rechte WITH GRANT OPTION gewähren.

TO user_name | ROLE role_name | GROUP group_name

Der Name des Benutzers, der Rolle oder der Benutzergruppe, auf die die angegebenen Standardrechte angewendet werden.

REVOKE

Der Satz von Rechten, der den angegebenen Benutzern oder Gruppen für alle neuen Tabellen, Funktionen oder gespeicherten Prozeduren entzogen werden soll, die vom angegebenen Benutzer erstellt werden. Sie können mit der REVOKE-Klausel die gleichen Rechte und Optionen wie mit dem Befehl REVOKE festlegen.

GRANT OPTION FOR

Eine Klausel, die nur die Option, anderen Benutzern ein spezifisches Recht zu gewähren, und nicht das Recht selbst widerruft. Sie können GRANT OPTION nicht für eine Gruppe oder PUBLIC widerrufen.

FROM user_name | ROLE role_name | GROUP group_name

Der Name des Benutzers, der Rolle oder der Benutzergruppe, für die die angegebenen Standardrechte standardmäßig widerrufen werden.

RESTRICT

Die Option RESTRICT widerruft nur die Berechtigungen, die der Benutzer direkt gewährt hat. Dies ist die Standardeinstellung.

Beispiele

Angenommen, Sie möchten jedem Benutzer in der Benutzergruppe `report_readers` erlauben, alle vom Benutzer erstellten Tabellen und Ansichten anzuzeigen`report_admin`. In diesem Fall führen Sie den folgenden Befehl als Superuser aus.

```
alter default privileges for user report_admin grant select on tables to group
 report_readers;
```

Im folgenden Beispiel gewährt der erste Befehl SELECT-Rechte für alle neuen Tabellen und Ansichten, die Sie erstellen.

```
alter default privileges grant select on tables to public;
```

Im folgenden Beispiel wird der Benutzergruppe `sales_admin` das INSERT-Recht für alle neuen Tabellen und Ansichten gewährt, die Sie im Schema `sales` erstellen.

```
alter default privileges in schema sales grant insert on tables to group sales_admin;
```

Im folgenden Beispiel wird der Befehl ALTER DEFAULT PRIVILEGES im vorherigen Beispiel zurückgenommen.

```
alter default privileges in schema sales revoke insert on tables from group
 sales_admin;
```

Standardmäßig besitzt die Benutzergruppe PUBLIC die Ausführungsberechtigung für alle neuen, von Benutzern definierten Funktionen. Um die `public`-Ausführungsberechtigungen für Ihre neuen Funktionen zu widerrufen und dann die Ausführungsberechtigung nur der Benutzergruppe `dev_test` zu gewähren, führen Sie die folgenden Befehle aus.

```
alter default privileges revoke execute on functions from public;
alter default privileges grant execute on functions to group dev_test;
```

ALTER EXTERNAL VIEW (Vorschau)

Hierbei handelt es sich um die vorab veröffentlichte Dokumentation im Datenkatalog für Amazon Redshift, derzeit in der Vorschauversion. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter Beta- und Vorschauversionen in den AWS -Servicebedingungen.

Sie können einen Amazon-Redshift-Cluster in der Vorschau erstellen, um neue Funktionen von Amazon Redshift zu testen. Sie haben nicht die Möglichkeit, diese Funktionen in der Produktion zu verwenden oder Ihren Vorschau-Cluster in einen Produktionscluster oder einen Cluster auf einem anderen Pfad zu verschieben. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Erstellen eines Clusters in der Vorschau

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.

2. Wählen Sie im Navigationsmenü Provisioned clusters dashboard (Dashboard für bereitgestellte Cluster) und dann Clusters (Cluster) aus. Die aktuellen Cluster für Ihr Konto AWS-Region sind aufgeführt. Eine Teilmenge der Eigenschaften jedes Clusters wird in den Spalten der Liste angezeigt.
3. Auf der Seite mit der Clusterliste (Clusters) wird ein Banner angezeigt, das die Vorschau vorstellt. Wählen Sie die Schaltfläche Create preview cluster (Vorschau-Cluster erstellen) aus, um die Seite zum Erstellen von Clustern zu öffnen.
4. Geben Sie Eigenschaften für Ihren Cluster ein. Wählen Sie den Vorschaupfad (Preview track) aus, der die zu testenden Funktionen enthält. Wir empfehlen, einen Namen für den Cluster zu verwenden, der darauf hinweist, dass sich dieser auf einem Vorschaupfad befindet. Wählen Sie Optionen für Ihren Cluster, einschließlich Optionen mit der Bezeichnung -preview (Vorschau), für die zu testenden Funktionen. Allgemeine Informationen zum Erstellen von Clustern finden Sie unter Erstellen eines Clusters im Amazon-Redshift-Verwaltungshandbuch.
5. Wählen Sie Vorschau-Cluster erstellen aus, um einen Cluster in der Vorschau zu erstellen.

   > **Note**
   >
   > Der `preview_2023`-Track ist der neueste verfügbare Vorschau-Track. Dieser Track unterstützt nur die Erstellung von Clustern mit RA3-Knotentypen. Der Knotentyp DC2 und alle älteren Knotentypen werden nicht unterstützt.

6. Wenn Ihr Vorschau-Cluster verfügbar ist, verwenden Sie Ihren SQL-Client, um Daten zu laden und abzufragen.

Das Vorschau-Feature für den Datenkatalog ist nur in den folgenden Regionen verfügbar.

- USA Ost (Ohio): (us-east-2)
- USA Ost (Nord-Virginia): (us-east-1)
- USA West (Nordkalifornien) (us-west-1)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Sie können außerdem eine Vorschau-Arbeitsgruppe erstellen, um Datenkatalog-Ansichten zu testen. Sie können diese Features nicht in der Produktion verwenden und Ihre Vorschau-Arbeitsgruppe auch

nicht in eine andere Arbeitsgruppe verschieben. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter „Beta- und Vorschauversionen“ in den AWS -Servicebedingungen. Eine Anleitung zur Erstellung einer Vorschau-Arbeitsgruppe finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe.

Verwenden Sie den Befehl ALTER EXTERNAL VIEW, um Ihre externe Ansicht zu aktualisieren. Je nachdem, welche Parameter Sie verwenden, können andere SQL-Engines wie Amazon Athena und Amazon EMR Spark, die ebenfalls auf diese Ansicht verweisen können, betroffen sein. Weitere Informationen zu Datenkatalog-Ansichten finden Sie unter Erstellen von Datenkatalog-Ansichten (Vorschau).

Syntax

```
ALTER EXTERNAL VIEW schema_name.view_name
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
[FORCE] { AS (query_definition) | REMOVE DEFINITION }
```

Parameter

schema_name.view_name

Das Schema, das an Ihre AWS Glue Datenbank angehängt ist, gefolgt vom Namen der Ansicht.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

Die Notation des Schemas, das beim Ändern der Ansicht verwendet werden soll. Sie können angeben AWS Glue Data Catalog, ob Sie eine von Ihnen erstellte Glue-Datenbank oder ein von Ihnen erstelltes externes Schema verwenden möchten. Weitere Informationen finden Sie unter CREATE DATABASE und CREATE EXTERNAL SCHEMA.

FORCE

Gibt an, ob die Definition der Ansicht aktualisiert werden AWS Lake Formation soll, auch wenn die in der Tabelle referenzierten Objekte nicht mit anderen SQL-Engines übereinstimmen. Wenn Lake Formation die Ansicht aktualisiert, gilt die Ansicht für die anderen SQL-Engines als veraltet, bis auch diese Engines aktualisiert werden.

AS query_definition

Die Definition der SQL-Abfrage, die Amazon Redshift ausführt, um die Ansicht zu ändern.

REMOVE DEFINITION

Ob die Ansichten gelöscht und neu erstellt werden sollen. Ansichten müssen gelöscht und neu erstellt werden, um sie als PROTECTED zu kennzeichnen.

Beispiele

Im folgenden Beispiel wird eine Datenkatalog-Ansicht mit dem Namen sample_schema.glue_data_catalog_view geändert.

```
ALTER EXTERNAL VIEW sample_schema.glue_data_catalog_view
FORCE
REMOVE DEFINITION
```

ALTER FUNCTION

Benennt eine Funktion um oder ändert den Besitzer. Sowohl der Funktionsname als auch die Datentypen sind erforderlich. Nur der Besitzer oder ein Superuser kann eine Funktion umbenennen. Nur ein Superuser kann den Besitzer einer Funktion ändern.

Syntax

```
ALTER FUNCTION function_name ( { [ py_arg_name py_arg_data_type | sql_arg_data_type }
 [ , ... ] ] )
    RENAME TO new_name
```

```
ALTER FUNCTION function_name ( { [ py_arg_name py_arg_data_type | sql_arg_data_type }
 [ , ... ] ] )
    OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
```

Parameter

function_name

Der Name der Funktion, die geändert werden soll. Geben Sie entweder den Namen der Funktion im aktuellen Suchpfad an, oder verwenden Sie das Format, `schema_name.function_name` um ein bestimmtes Schema zu verwenden.

py_arg_name py_arg_data_type | sql_arg_data_type

Optional. Eine Liste von Eingabeargumentnamen und Datentypen für die benutzerdefinierte Python-Funktion oder eine Liste von Eingabeargumentdatentypen für die benutzerdefinierte SQL-Funktion.

new_name

Ein neuer Name für die benutzerdefinierte Funktion.

new_owner | CURRENT_USER | SESSION_USER

Ein neuer Besitzer für die benutzerdefinierte Funktion.

Beispiele

Im folgenden Beispiel wird der Name einer Funktion von `first_quarter_revenue` in geändert. `quarterly_revenue`

```
ALTER FUNCTION first_quarter_revenue(bigint, numeric, int)
         RENAME TO quarterly_revenue;
```

Im folgenden Beispiel wird der Besitzer der `quarterly_revenue` Funktion in geändert`etl_user`.

```
ALTER FUNCTION quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

ALTER GROUP

Ändert eine Benutzergruppe. Mit diesem Befehl können Sie der Gruppe Benutzer hinzufügen, Benutzer aus der Gruppe entfernen oder die Gruppe umbenennen.

Syntax

```
ALTER GROUP group_name
{
ADD USER username [, ... ] |
DROP USER username [, ... ] |
RENAME TO new_name
}
```

Parameter

group_name

Der Name der Benutzergruppe, die geändert werden soll.

ADD

Fügt einer Benutzergruppe einen Benutzer hinzu.

DROP

Entfernt einen Benutzer aus einer Benutzergruppe.

username (Benutzername

Der Name des Benutzers, der der Gruppe hinzugefügt oder aus der Gruppe entfernt werden soll.

RENAME TO

Benennt die Benutzergruppe um. Gruppennamen, die mit zwei Unterstrichen beginnen, sind für die interne Verwendung durch Amazon Redshift reserviert. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

new_name

Der neue Name der Benutzergruppe.

Beispiele

Im folgenden Beispiel wird der Gruppe ADMIN_GROUP ein Benutzer mit dem Namen DWUSER hinzugefügt.

```
ALTER GROUP admin_group
ADD USER dwuser;
```

Im folgenden Beispiel wird die Gruppe ADMIN_GROUP in ADMINISTRATORS umbenannt.

```
ALTER GROUP admin_group
RENAME TO administrators;
```

Im folgenden Beispiel werden der Gruppe ADMIN_GROUP zwei Benutzer hinzugefügt.

```
ALTER GROUP admin_group
```

```
ADD USER u1, u2;
```

Im folgenden Beispiel werden aus der Gruppe ADMIN_GROUP zwei Benutzer entfernt.

```
ALTER GROUP admin_group
DROP USER u1, u2;
```

ALTER IDENTITY PROVIDER

Ändert einen Identitätsanbieter, um neue Parameter und Werte zuzuweisen. Wenn Sie diesen Befehl ausführen, werden alle zuvor festgelegten Parameterwerte gelöscht, bevor die neuen Werte zugewiesen werden. Nur Superuser können Identitätsanbieter ändern.

Syntax

```
ALTER IDENTITY PROVIDER identity_provider_name
[PARAMETERS parameter_string]
[NAMESPACE namespace]
[IAM_ROLE iam_role]
[DISABLE | ENABLE]
```

Parameter

identity_provider_name

Name des neuen Identitätsanbieters. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

parameter_string

Eine Zeichenfolge mit einem ordnungsgemäß formatierten JSON-Objekt, das die für den spezifischen Identitätsanbieter erforderlichen Parameter und Werte enthält.

Namespace

Der Organisations-Namespace.

iam_role

Die IAM-Rolle, die Berechtigungen für die Verbindung zum IAM Identity Center bereitstellt. Dieser Parameter ist nur anwendbar, wenn der Identitätsanbietertyp ist. AWSIDC

DEAKTIVIEREN oder AKTIVIEREN

Schaltet einen Identitätsanbieter ein oder aus. Die Standardeinstellung ist ENABLE

Beispiele

Im folgenden Beispiel wird ein Identitätsanbieter mit dem Namen oauth_standard geändert. Dies gilt insbesondere dann, wenn Microsoft Azure AD der Identitätsanbieter ist.

```
ALTER IDENTITY PROVIDER oauth_standard
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

Das folgende Beispiel zeigt, wie der Namespace für den Identitätsanbieter festgelegt wird. Dies kann für Microsoft Azure AD gelten, wenn es einer Anweisung wie im vorherigen Beispiel folgt, oder für einen anderen Identitätsanbieter. Dies kann auch für den Fall gelten, dass Sie einen vorhandenen von Amazon Redshift bereitgestellten Cluster oder eine Amazon Redshift Serverless-Arbeitsgruppe mit IAM Identity Center verbinden, wenn Sie eine Verbindung über eine verwaltete Anwendung eingerichtet haben.

```
ALTER IDENTITY PROVIDER "my-redshift-idc-application"
NAMESPACE 'MYCO';
```

Das folgende Beispiel legt die IAM-Rolle fest und funktioniert im Anwendungsfall für die Konfiguration der Redshift-Integration mit IAM Identity Center.

```
ALTER IDENTITY PROVIDER "my-redshift-idc-application"
IAM_ROLE 'arn:aws:iam::123456789012:role/myadministratorrole';
```

Weitere Informationen zum Einrichten einer Verbindung zu IAM Identity Center von Redshift aus finden Sie unter Redshift mit IAM Identity Center Connect, um Benutzern ein Single-Sign-On-Erlebnis zu bieten.

Deaktivieren eines Identitätsanbieters

Die folgende Beispielanweisung zeigt, wie ein Identitätsanbieter deaktiviert wird. Wenn er deaktiviert ist, können sich Verbundbenutzer des Identitätsanbieters erst wieder beim Cluster anmelden, wenn er wieder aktiviert ist.

```
ALTER IDENTITY PROVIDER "redshift-idc-app" DISABLE;
```

ALTER MASKING POLICY

Ändert eine vorhandene Richtlinie für die dynamische Datenmaskierung. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Maskierungsrichtlinie ändern.

Syntax

```
ALTER MASKING POLICY policy_name
   USING (masking_expression);
```

Parameter

policy_name

Der Name der Maskierungsrichtlinie. Dies muss der Name einer Maskierungsrichtlinie sein, die bereits in der Datenbank vorhanden ist.

masking_expression

Der SQL-Ausdruck, der zur Transformation der Zielspalten verwendet wird. Er kann mithilfe von Datenmanipulationsfunktionen wie z. B. Funktionen zur Zeichenkettenmanipulation oder in Verbindung mit benutzerdefinierten Funktionen geschrieben werden, die in SQL, Python oder mit AWS Lambda geschrieben wurden.

Der Ausdruck muss mit den Eingabespalten und Datentypen des ursprünglichen Ausdrucks übereinstimmen. Wenn die Eingabespalten der ursprünglichen Maskierungsrichtlinie beispielsweise `sample_1 FLOAT` und `sample_2 VARCHAR(10)` sind, können Sie die Maskierungsrichtlinie nicht so ändern, dass sie eine dritte Spalte oder FLOAT und BOOLEAN verwendet. Wenn Sie eine Konstante als Maskierungsausdruck verwenden, müssen Sie sie explizit in einen dem Eingabetyp entsprechenden Typ umwandeln.

Sie benötigen USAGE-Berechtigung für alle benutzerdefinierten Funktionen, die Sie im Maskierungsausdruck verwenden.

ALTER MATERIALIZED VIEW

Aktiviert das automatische Aktualisieren einer materialisierten Ansicht.

Syntax

```
ALTER MATERIALIZED VIEW mv_name
[ AUTO REFRESH { YES | NO } ]
[ ROW LEVEL SECURITY { ON | OFF } [ CONJUNCTION TYPE { AND | OR } ] [FOR DATASHARES] ];
```

Parameter

mv_name

Der Name der materialisierten Ansicht, der geändert werden soll.

AUTO REFRESH { YES | NO }

Eine Klausel, die das automatische Aktualisieren einer materialisierten Ansicht aktiviert oder deaktiviert. Weitere Informationen zur automatischen Aktualisierung von materialisierten Ansichten finden Sie unter Aktualisieren einer materialisierten Ansicht.

ROW LEVEL SECURITY { ON | OFF } [CONJUNCTION TYPE { AND | OR }] [FOR DATASHARES]

Eine Klausel, die die Sicherheit auf Zeilenebene für eine Relation aktiviert oder deaktiviert.

Wenn die Sicherheit auf Zeilenebene für eine Relation aktiviert ist, können Sie nur die Zeilen lesen, für die Ihnen die Richtlinie auf Zeilenebene Zugriff gewährt. Wenn es keine Richtlinie gibt, die Ihnen Zugriff auf die Relation gewährt, können Sie keine Zeilen aus der Relation sehen. Nur Superuser und Benutzer oder Rollen, denen die `sys:secadmin`-Rolle zugewiesen ist, können die Klausel ROW LEVEL SECURITY festlegen. Weitere Informationen finden Sie unter Sicherheit auf Zeilenebene.

- [CONJUNCTION TYPE { AND | OR }]

  Eine Klausel, mit der Sie den Verbindungstyp einer Sicherheitsrichtlinie auf Zeilenebene für eine Relation auswählen können. Wenn einer Relation mehrere Sicherheitsrichtlinien auf Zeilenebene zugeordnet sind, können Sie die Richtlinien mit der AND- oder OR-Klausel

kombinieren. Standardmäßig kombiniert Amazon Redshift RLS-Richtlinien mit der AND-Klausel. Superuser, Benutzer oder Rollen, denen diese `sys:secadmin`-Rolle zugewiesen wurde, können diese Klausel verwenden, um den Verbindungstyp der Sicherheitsrichtlinie auf Zeilenebene für eine Relation zu definieren. Weitere Informationen finden Sie unter Kombinieren mehrerer Richtlinien pro Benutzer.

- FOR DATASHARES

  Eine Klausel, die festlegt, ob auf eine RLS-geschützte Relation über Datashares zugegriffen werden kann. Standardmäßig kann auf eine RLS-geschützte Relation nicht über ein Datashare zugegriffen werden. Ein ALTER MATERIALIZED VIEW ROW LEVEL SECURITY-Befehl, der mit dieser Klausel ausgeführt wird, wirkt sich nur auf die Zugänglichkeitseigenschaft des Datashares der Relation aus. Die Eigenschaft ROW LEVEL SECURITY wird nicht geändert.

  Wenn Sie eine RLS-geschützte Relation über Datashares zugänglich machen, bietet die Relation in der verbraucherseitigen, gemeinsam genutzten Datenbank keine Sicherheit auf Zeilenebene. Die Relation behält ihre RLS-Eigenschaft auf der Produzentenseite.

Beispiele

Im folgenden Beispiel wird die materialisierte Ansicht `tickets_mv` automatisch aktualisiert.

```
ALTER MATERIALIZED VIEW tickets_mv AUTO REFRESH YES
```

Beispiele für DISTSTYLE und SORTKEY

Die Beispiele in diesem Thema zeigen, wie Sie DISTSTYLE- und SORTKEY-Änderungen mithilfe von ALTER MATERIALIZED VIEW vornehmen.

Die folgenden Beispielabfragen zeigen, wie Sie eine DISTSTYLE KEY DISTKEY-Spalte anhand einer Beispiel-Basistabelle ändern können:

```
CREATE TABLE base_inventory(
inv_date_sk int4 not null,
inv_item_sk int4 not null,
inv_warehouse_sk int4 not null,
inv_quantity_on_hand int4
);

INSERT INTO base_inventory VALUES(1,1,1,1);
```

```
CREATE MATERIALIZED VIEW inventory DISTSTYLE EVEN
as SELECT * FROM base_inventory;
SELECT "table", DISTSTYLE FROM svv_table_info WHERE "table" = 'inventory';

ALTER MATERIALIZED VIEW inventory ALTER DISTSTYLE KEY DISTKEY inv_warehouse_sk;
SELECT "table", DISTSTYLE FROM svv_table_info where "table" = 'inventory';

ALTER MATERIALIZED VIEW inventory ALTER DISTKEY inv_item_sk;
SELECT "table", diststyle from svv_table_info where "table" = 'inventory';
```

Ändern Sie eine materialisierte Ansicht in DISTSTYLE ALL:

```
CREATE TABLE base_inventory(
inv_date_sk int4 not null,
inv_item_sk int4 not null,
inv_warehouse_sk int4 not null,
inv_quantity_on_hand int4
);

INSERT INTO base_inventory values(1,1,1,1);

CREATE MATERIALIZED VIEW inventory DISTSTYLE EVEN
as SELECT * FROM base_inventory;

SELECT "table", DISTSTYLE FROM svv_table_info WHERE "table" = 'inventory';
```

Die folgenden Befehle zeigen SORTKEY-Beispiele für ALTER MATERIALIZED VIEW unter Verwendung einer Beispiel-Basistabelle:

```
CREATE MATERIALIZED VIEW base_inventory (c0 int, c1 int);

CREATE MATERIALIZED VIEW inventory
interleaved sortkey(c0, c1)
as SELECT * FROM base_inventory;

SELECT "table", sortkey1 FROM svv_table_info WHERE "table" = 'inventory';

ALTER MATERIALIZED VIEW t1 alter sortkey(c0, c1);
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

ALTER MATERIALIZED VIEW t1 alter sortkey none;
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';
```

```
ALTER MATERIALIZED VIEW t1 alter sortkey(c0);
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';
```

ALTER RLS POLICY

Ändern Sie eine bestehende Sicherheitsrichtlinie auf Zeilenebene in einer Tabelle.

Superuser und Benutzer oder Rollen, denen die `sys:secadmin`-Rolle zugewiesen ist, können eine Richtlinie ändern.

Syntax

```
ALTER RLS POLICY policy_name
USING ( using_predicate_exp );
```

Parameter

policy_name

Der Name der -Richtlinie.

MITHILFE VON (using_predicate_exp)

Gibt einen Filter an, der auf die WHERE-Klausel der Abfrage angewendet wird. Amazon Redshift wendet ein Richtlinienprädikat vor den Benutzerprädikaten auf Abfrageebene an. Beispielsweise schränkt **`current_user = 'joe' and price > 10`** ein, dass Joe nur Datensätze mit einem Preis von mehr als 10 USD sehen kann.

Der Ausdruck hat Zugriff auf die Variablen, die in der WITH-Klausel der Anweisung CREATE RLS POLICY deklariert sind, mit der die Richtlinie mit dem Namen policy_name erstellt wurde.

Beispiele

Im folgenden Beispiel wird eine RLS-Richtlinie geändert.

```
-- First create an RLS policy that limits access to rows where catgroup is 'concerts'.
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'concerts');
```

```
-- Then, alter the RLS policy to only show rows where catgroup is 'piano concerts'.
ALTER RLS POLICY policy_concerts
USING (catgroup = 'piano concerts');
```

ALTER ROLE

Benennt eine Rolle um oder ändert den Besitzer. Eine Liste der systemdefinierten Amazon-Redshift-Rollen finden Sie unter the section called “Systemdefinierte Amazon-Redshift-Rollen”.

Erforderliche Berechtigungen

Für ALTER ROLE sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung ALTER ROLE

Syntax

```
ALTER ROLE role [ WITH ]
  { { RENAME TO role } | { OWNER TO user_name } }[, ...]
  [ EXTERNALID TO external_id ]
```

Parameter

Rolle

Der Name der zu ändernden Rolle.

RENAME TO

Ein neuer Name für die Rolle.

OWNER TO benutzer_name

Ein neuer Besitzer für die Rolle.

EXTERNALID TO external_id

Eine neue externe ID für die Rolle, die einem Identitätsanbieter zugeordnet ist. Weitere Informationen finden Sie unter Nativer Identitätsanbieter(IdP)-Verbund für Amazon Redshift.

Beispiele

Das folgende Beispiel ändert den Namen einer Rolle von `sample_role1` in `sample_role2`.

```
ALTER ROLE sample_role1 WITH RENAME TO sample_role2;
```

Das folgende Beispiel ändert den Besitzer der Rolle.

```
ALTER ROLE sample_role1 WITH OWNER TO user1
```

Die Syntax von ALTER ROLE ähnelt der von ALTER PROCEDURE.

```
ALTER PROCEDURE first_quarter_revenue(bigint, numeric) RENAME TO quarterly_revenue;
```

Das folgende Beispiel ändert den Besitzer einer Prozedur in `etl_user`.

```
ALTER PROCEDURE quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

Im folgenden Beispiel wird eine Rolle `sample_role1` mit einer neuen externen ID aktualisiert, die einem Identitätsanbieter zugeordnet ist.

```
ALTER ROLE sample_role1 EXTERNALID TO "XYZ456";
```

ALTER PROCEDURE

Benennt eine Prozedur um oder ändert den Besitzer. Erforderlich sind sowohl der Name der Prozedur als auch die Datentypen bzw. Signatur. Eine Prozedur kann nur vom Besitzer oder von einem Superuser umbenannt werden. Nur ein Superuser kann den Besitzer einer Prozedur ändern.

Syntax

```
ALTER PROCEDURE sp_name [ ( [ [ argname ] [ argmode ] argtype [, ...] ] ) ]
    RENAME TO new_name
```

```
ALTER PROCEDURE sp_name [ ( [ [ argname ] [ argmode ] argtype [, ...] ] ) ]
    OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
```

Parameter

sp_name

Der Name der Prozedur, die geändert werden soll. Geben Sie einfach den Namen der Prozedur im aktuellen Suchpfad an oder verwenden Sie das Format `schema_name.sp_procedure_name`, um ein spezifisches Schema zu verwenden.

[argname] [argmode] argtype

Eine Liste von Argumentnamen, Argumentmodi und Datentypen. Nur die Eingabedatentypen sind erforderlich. Diese werden zur Identifizierung der gespeicherten Prozedur verwendet. Alternativ können Sie auch die komplette Signatur angeben, die zum Erstellen der Prozedur verwendet wurde, einschließlich der Eingabe- und Ausgabeparameter mit ihren Modi.

new_name

Ein neuer Name für die gespeicherte Prozedur.

new_owner | CURRENT_USER | SESSION_USER

Ein neuer Besitzer für die gespeicherte Prozedur.

Beispiele

Das folgende Beispiel ändert den Namen einer Prozedur von `first_quarter_revenue` in `quarterly_revenue`.

```
ALTER PROCEDURE first_quarter_revenue(volume INOUT bigint, at_price IN numeric,
 result OUT int) RENAME TO quarterly_revenue;
```

Dieses Beispiel entspricht Folgendem.

```
ALTER PROCEDURE first_quarter_revenue(bigint, numeric) RENAME TO quarterly_revenue;
```

Das folgende Beispiel ändert den Besitzer einer Prozedur in `etl_user`.

```
ALTER PROCEDURE quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

ALTER SCHEMA

Ändert die Definition eines vorhandenen Schemas. Mit diesem Befehl können Sie ein Schema umbenennen oder den Besitzer eines Schemas ändern. Sie können beispielsweise ein vorhandenes Schema umbenennen, um eine Sicherungskopie dieses Schemas zu behalten, wenn Sie planen, eine neue Version dieses Schemas zu erstellen. Weitere Informationen zu Schemata finden Sie unter CREATE SCHEMA.

Um die konfigurierten Schemakontingente anzuzeigen, siehe SVV_SCHEMA_QUOTA_STATE.

Um die Datensätze anzuzeigen, bei denen die Schemakontingente überschritten wurden, siehe STL_SCHEMA_QUOTA_VIOLATIONS.

Erforderliche Berechtigungen

Für ALTER SCHEMA sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung ALTER SCHEMA
- Schemabesitzer

Beachten Sie beim Ändern eines Schemanamens, dass Objekte, die den alten Namen verwenden, wie z. B. gespeicherte Prozeduren oder materialisierte Ansichten, aktualisiert werden müssen, um den neuen Namen zu verwenden.

Syntax

```
ALTER SCHEMA schema_name
{
RENAME TO new_name |
OWNER TO new_owner |
QUOTA { quota [MB | GB | TB] | UNLIMITED }
}
```

Parameter

schema_name

Der Name des Datenbankschemas, das geändert werden soll.

RENAME TO

Eine Klausel, die das Schema umbenennt.

new_name

Der neue Name des Schemas. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

OWNER TO

Eine Klausel, die den Besitzer des Schemas ändert.

new_owner

Der neue Besitzer des Schemas.

QUOTA

Die maximale Menge an Speicherplatz, die das angegebene Schema verwenden kann. Dieser Platz ist die Gesamtgröße aller Tabellen unter dem angegebenen Schema. Amazon Redshift konvertiert den ausgewählten Wert in Megabyte. Gigabyte ist die Standardmaßeinheit, wenn Sie keinen Wert angeben.

Weitere Informationen zur Konfiguration von Schemakontingenten finden Sie unter CREATE SCHEMA.

Beispiele

Im folgenden Beispiel wird das Schema SALES in US_SALES umbenannt.

```
alter schema sales
rename to us_sales;
```

Im folgenden Beispiel erhält der Benutzer DWUSER den Besitz des Schemas US_SALES.

```
alter schema us_sales
owner to dwuser;
```

Im folgenden Beispiel wird das Kontingent zu 300 GB geändert und das Kontingent entfernt.

```
alter schema us_sales QUOTA 300 GB;
```

```
alter schema us_sales QUOTA UNLIMITED;
```

ALTER SYSTEM

Ändert eine Konfigurationsoption auf Systemebene für den Amazon-Redshift-Cluster oder die Redshift-Serverless-Arbeitsgruppe.

Erforderliche Berechtigungen

Einer der folgenden Benutzertypen kann den Befehl ALTER SYSTEM ausführen:

- Superuser
- Admin-Benutzer

Syntax

```
ALTER SYSTEM SET system-level-configuration = {true| t | on | false | f | off}
```

Parameter

Konfiguration auf Systemebene

Eine Konfiguration auf Systemebene. Gültiger Wert: `data_catalog_auto_mount` und `metadata_security`.

{true| t | on | false | f | off}

Ein Wert zum Aktivieren oder Deaktivieren der Konfiguration auf Systemebene. `true`, `t` oder `on` gibt an, die Konfiguration zu aktivieren. `false`, `f` oder `off` gibt an, die Konfiguration zu deaktivieren.

Nutzungshinweise

Für einen bereitgestellten Cluster werden Änderungen an `data_catalog_auto_mount` beim nächsten Neustart des Clusters wirksam. Weitere Informationen finden Sie unter Neustart eines Cluster im Amazon-Redshift-Verwaltungshandbuch.

Bei einer Serverless-Arbeitsgruppe werden die Änderungen an `data_catalog_auto_mount` nicht sofort wirksam.

Beispiele

Im folgenden Beispiel wird das automatische Mounting von AWS Glue Data Catalog aktiviert.

```
ALTER SYSTEM SET data_catalog_auto_mount = true;
```

Im folgenden Beispiel wird die Metadatensicherheit aktiviert.

```
ALTER SYSTEM SET metadata_security = true;
```

Einen Standard-Identitäts-Namespace einrichten

Dieses Beispiel bezieht sich speziell auf die Arbeit mit einem Identitätsanbieter. Sie können Redshift mit IAM Identity Center und einem Identitätsanbieter integrieren, um das Identitätsmanagement für Redshift und andere Dienste zu zentralisieren. AWS

Das folgende Beispiel zeigt, wie der Standard-Identitätsnamespace für das System festgelegt wird. Wenn Sie dies anschließend tun, ist es einfacher, die Anweisungen GRANT und CREATE auszuführen, da Sie den Namespace nicht als Präfix für jede Identität angeben müssen.

```
ALTER SYSTEM SET default_identity_namespace = 'MYCO';
```

Nachdem Sie den Befehl ausgeführt haben, können Sie Anweisungen wie die folgenden ausführen:

```
GRANT SELECT ON TABLE mytable TO alice;

GRANT UPDATE ON TABLE mytable TO salesrole;

CREATE USER bob password 'md50c983d1a624280812631c5389e60d48c';
```

Die Festlegung des standardmäßigen Identitätsnamespaces hat zur Folge, dass er nicht für jede Identität als Präfix erforderlich ist. In diesem Beispiel `alice` wird ersetzt `MYCO:alice` durch. Dies geschieht mit jeder mitgelieferten Identität. Weitere Informationen zur Verwendung eines Identitätsanbieters mit Redshift finden Sie unter Connect von Redshift mit IAM Identity Center, um Benutzern eine Single-Sign-On-Erfahrung zu bieten.

Weitere Informationen zu Einstellungen, die sich auf die Redshift-Konfiguration mit IAM Identity Center beziehen, finden Sie unter und. SET ALTER IDENTITY PROVIDER

ALTER TABLE

Dieser Befehl ändert die Definition einer Amazon-Redshift-Tabelle oder einer externen Amazon-Redshift-Spectrum-Tabelle. Dieser Befehl aktualisiert die von CREATE TABLE oder CREATE EXTERNAL TABLE festgelegten Werte und Eigenschaften.

Sie können ALTER TABLE nicht für eine externe Tabelle innerhalb eines Transaktionsblocks (BEGIN ... END) ausführen. Weitere Informationen zu Transaktionen finden Sie unter Serialisierbare Isolierung.

ALTER TABLE sperrt die Tabelle für Lese- und Schreibvorgänge, bis die die Operation ALTER TABLE umschließende Transaktion abgeschlossen ist, es sei denn, in der Dokumentation ist ausdrücklich angegeben, dass Sie Daten abfragen oder andere Operationen an der Tabelle ausführen können, während sie geändert wird.

Erforderliche Berechtigungen

Der Benutzer, der eine Tabelle ändert, benötigt die entsprechenden Berechtigungen, damit der Befehl erfolgreich ausgeführt werden kann. Je nach verwendetem Befehl ALTER TABLE ist eine der folgenden Berechtigungen erforderlich.

- Superuser
- Benutzer mit der Berechtigung ALTER TABLE
- Tabellenbesitzer mit dem USAGE-Recht für das Schema

Syntax

```
ALTER TABLE table_name
{
ADD table_constraint
| DROP CONSTRAINT constraint_name [ RESTRICT | CASCADE ]
| OWNER TO new_owner
| RENAME TO new_name
| RENAME COLUMN column_name TO new_name
| ALTER COLUMN column_name TYPE updated_varchar_data_type_size
| ALTER COLUMN column_name ENCODE new_encode_type
| ALTER COLUMN column_name ENCODE encode_type,
| ALTER COLUMN column_name ENCODE encode_type, .....;
| ALTER DISTKEY column_name
| ALTER DISTSTYLE ALL
```

```
| ALTER DISTSTYLE EVEN
| ALTER DISTSTYLE KEY DISTKEY column_name
| ALTER DISTSTYLE AUTO
| ALTER [COMPOUND] SORTKEY ( column_name [,...] )
| ALTER SORTKEY AUTO
| ALTER SORTKEY NONE
| ALTER ENCODE AUTO
| ADD [ COLUMN ] column_name column_type
  [ DEFAULT default_expr ]
  [ ENCODE encoding ]
  [ NOT NULL | NULL ]
  [ COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE } ] |
| DROP [ COLUMN ] column_name [ RESTRICT | CASCADE ]
| ROW LEVEL SECURITY { ON | OFF } [ CONJUNCTION TYPE { AND | OR } ] [ FOR DATASHARES ]}

where table_constraint is:

[ CONSTRAINT constraint_name ]
{ UNIQUE ( column_name [, ... ] )
| PRIMARY KEY ( column_name [, ... ] )
| FOREIGN KEY (column_name [, ... ] )
   REFERENCES  reftable [ ( refcolumn ) ]}

The following options apply only to external tables:

SET LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }
| SET FILE FORMAT format |
| SET TABLE PROPERTIES ('property_name'='property_value')
| PARTITION ( partition_column=partition_value [, ...] )
  SET LOCATION { 's3://bucket/folder' |'s3://bucket/manifest_file' }
| ADD [IF NOT EXISTS]
    PARTITION ( partition_column=partition_value [, ...] ) LOCATION
 { 's3://bucket/folder' |'s3://bucket/manifest_file' }
    [, ... ]
| DROP PARTITION ( partition_column=partition_value [, ...] )
```

Um die Zeit für die Ausführung des Befehls ALTER TABLE zu verkürzen, können Sie einige Klauseln des Befehls ALTER TABLE kombinieren.

Amazon Redshift unterstützt die folgenden Kombinationen der ALTER TABLE-Klauseln:

```
ALTER TABLE tablename ALTER SORTKEY (column_list), ALTER DISTKEY column_Id;
ALTER TABLE tablename ALTER DISTKEY column_Id, ALTER SORTKEY (column_list);
```

```
ALTER TABLE tablename ALTER SORTKEY (column_list), ALTER DISTSTYLE ALL;
ALTER TABLE tablename ALTER DISTSTYLE ALL, ALTER SORTKEY (column_list);
```

Parameter

table_name

Der Name der Tabelle, die geändert werden soll. Geben Sie einfach den Namen der Tabelle an oder verwenden Sie das Format schema_name.table_name, um ein spezifisches Schema zu verwenden. Externe Tabellen müssen durch einen externen Schemanamen qualifiziert werden. Sie können auch einen Anzeigenamen angeben, wenn Sie die Anweisung ALTER TABLE verwenden, um eine Ansicht umzubenennen oder ihren Eigentümer zu ändern. Die maximale Länge des Tabellennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Sie können UTF-8-Multibyte-Zeichen bis zu einer Länge von vier Bytes verwenden. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

ADD table_constraint

Eine Klausel, die der Tabelle die angegebene Einschränkung hinzufügt. Beschreibungen der gültigen Werte für table_constraint finden Sie unter CREATE TABLE.

> **Note**
>
> Sie können einer löschbaren Spalte keine Einschränkung „primary-key“ hinzufügen. Wenn die Spalte ursprünglich mit der Einschränkung NOT NULL erstellt wurde, können Sie die Einschränkung „primary-key“ hinzufügen.

DROP CONSTRAINT constraint_name

Eine Klausel, die die angegebene Einschränkung aus der Tabelle entfernt. Um eine Einschränkung zu entfernen, geben Sie den Namen der Einschränkung an, nicht den Typ der Einschränkung. Um Namen von Tabelleneinschränkungen anzuzeigen, führen Sie die folgende Abfrage aus.

```
select constraint_name, constraint_type
from information_schema.table_constraints;
```

RESTRICT

Eine Klausel, die nur die angegebene Einschränkung entfernt. RESTRICT ist eine Option für DROP CONSTRAINT. RESTRICT kann nicht mit CASCADE verwendet werden.

CASCADE

Eine Klausel, die die angegebene Einschränkung und alles, was von dieser Einschränkung abhängig ist, entfernt. CASCADE ist eine Option für DROP CONSTRAINT. CASCADE kann nicht mit RESTRICT verwendet werden.

OWNER TO new_owner

Eine Klausel, die den Besitzer der Tabelle (oder Ansicht) in den Wert new_owner ändert.

RENAME TO new_name

Eine Klausel, die die Tabelle (oder Ansicht) in den Wert ändert, der in new_name angegeben ist. Die maximale Länge des Tabellennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten.

Sie können den Namen einer permanenten Tabelle nicht in einen Namen umbenennen, der mit „#“ beginnt. Ein Tabellenname, der mit „#“ beginnt, zeigt eine temporäre Tabelle an.

Sie können eine externe Tabelle nicht umbenennen.

ALTER COLUMN column_name TYPE updated_varchar_data_type_size

Eine Klausel, die die Größe einer Spalte ändert, die als VARCHAR-Datentyp definiert ist. Diese Klausel unterstützt nur eine Größenänderung für einen VARCHAR-Datentyp. Erwägen Sie die folgenden Einschränkungen:

- Spalten mit den Komprimierungskodierungen BYTEDICT, RUNLENGTH, TEXT255 oder TEXT32K können nicht geändert werden.
- Sie können die Größe nicht auf weniger als die maximale Größe vorhandener Daten reduzieren.
- Spalten mit Standardwerten können nicht geändert werden.
- Spalten mit UNIQUE, PRIMARY KEY oder FOREIGN KEY können nicht geändert werden.
- Spalten innerhalb eines Transaktionsblocks (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

ALTER COLUMN column_name ENCODE new_encode_type

Eine Klausel, die die Komprimierungskodierung einer Spalte ändert. Wenn Sie die Komprimierungskodierung für eine Spalte angeben, wird die Tabelle nicht mehr auf ENCODE

AUTO festgelegt. Weitere Informationen zur Komprimierungskodierung finden Sie unter Arbeiten mit Spaltenkomprimierung.

Wenn Sie die Komprimierungskodierung für eine Spalte ändern, bleibt die Tabelle zur Abfrage verfügbar.

Erwägen Sie die folgenden Einschränkungen:

- Sie können eine Spalte nicht in dieselbe Kodierung ändern, die derzeit für die Spalte definiert ist.
- Sie können die Kodierung für eine Spalte in einer Tabelle mit einem überlappenden Sortierschlüssel nicht ändern.

ALTER COLUMN column_name ENCODE encode_type, ALTER COLUMN column_name ENCODE encode_type, ...;

Eine Klausel, die die Komprimierungskodierung mehrerer Spalten in einem einzigen Befehl ändert. Weitere Informationen zur Komprimierungskodierung finden Sie unter Arbeiten mit Spaltenkomprimierung.

Wenn Sie die Komprimierungskodierung für eine Spalte ändern, bleibt die Tabelle zur Abfrage verfügbar.

Erwägen Sie die folgenden Einschränkungen:

- Sie können eine Spalte nicht mehrmals mit einem einzigen Befehl in denselben oder einen anderen Kodierungstyp ändern.
- Sie können eine Spalte nicht in dieselbe Kodierung ändern, die derzeit für die Spalte definiert ist.
- Sie können die Kodierung für eine Spalte in einer Tabelle mit einem überlappenden Sortierschlüssel nicht ändern.

ALTER DISTSTYLE ALL

Eine Klausel, die den vorhandenen Verteilungsstil einer Tabelle in ändert ALL. Berücksichtigen Sie dabei Folgendes:

- ALTER DISTSTYLE, ALTER SORTKEY und VACUUM können nicht gleichzeitig für dieselbe Tabelle ausgeführt werden.
  - Wenn VACUUM gerade ausgeführt wird, wird beim Ausführen von ALTER DISTSTYLE ALL ein Fehler zurückgegeben.

    - Wenn ALTER DISTSTYLE ALL ausgeführt wird, startet eine Hintergrundbereinigung nicht für eine Tabelle.
- Der Befehl ALTER DISTSTYLE ALL wird für Tabellen mit verschachtelten Sortierschlüsseln und temporäre Tabellen nicht unterstützt.
- Wenn der Verteilungsstil zuvor als AUTO definiert war, dann ist die Tabelle kein Kandidat mehr für die automatische Tabellenoptimierung.

Weitere Informationen zu DISTSTYLE ALL finden Sie unter CREATE TABLE.

ALTER DISTSTYLE EVEN

Eine Klausel, die den vorhandenen Verteilungsstil einer Tabelle in ändert `EVEN`. Berücksichtigen Sie dabei Folgendes:

- ALTER DISTSYTLE, ALTER SORTKEY und VACUUM können nicht gleichzeitig für dieselbe Tabelle ausgeführt werden.
    - Wenn VACUUM gerade ausgeführt wird, wird beim Ausführen von ALTER DISTSTYLE EVEN ein Fehler zurückgegeben.
    - Wenn ALTER DISTSTYLE EVEN ausgeführt wird, startet eine Hintergrundbereinigung nicht für eine Tabelle.
- Der Befehl ALTER DISTSTYLE EVEN wird für Tabellen mit verschachtelten Sortierschlüsseln und temporäre Tabellen nicht unterstützt.
- Wenn der Verteilungsstil zuvor als AUTO definiert war, dann ist die Tabelle kein Kandidat mehr für die automatische Tabellenoptimierung.

Weitere Informationen zu DISTSTYLE EVEN finden Sie unter CREATE TABLE.

ALTER DISTKEY column_name oder ALTER DISTSTYLE KEY DISTKEY column_name

Eine Klausel, die die Spalte ändert, die als Verteilungsschlüssel einer Tabelle verwendet wird. Berücksichtigen Sie dabei Folgendes:

- VACUUM und ALTER DISTKEY können nicht gleichzeitig für dieselbe Tabelle ausgeführt werden.
    - Wenn VACUUM bereits ausgeführt wird, gibt ALTER DISTKEY einen Fehler zurück.
    - Wenn ALTER DISTKEY ausgeführt wird, startet die Hintergrundbereinigung nicht für die Tabelle.
    - Wenn ALTER DISTKEY ausgeführt wird, dann gibt die Vordergrundbereinigung einen Fehler zurück.

- Sie können nur jeweils einen ALTER DISTKEY-Befehl für eine Tabelle ausführen.
- Der Befehl ALTER DISTKEY wird für Tabellen mit verschachtelten Sortierschlüsseln nicht unterstützt.
- Wenn der Verteilungsstil zuvor als AUTO definiert war, dann ist die Tabelle kein Kandidat mehr für die automatische Tabellenoptimierung.

Wenn Sie DISTSTYLE KEY angeben, werden die Daten nach den Werten in der DISTKEY-Spalte verteilt. Weitere Informationen zu DISTSTYLE finden Sie unter CREATE TABLE.

ALTER DISTSTYLE AUTO

Eine Klausel, die den vorhandenen Verteilungsstil einer Tabelle in AUTO ändert.

Wenn Sie einen Verteilungsstil auf AUTO ändern, wird der Verteilungsstil der Tabelle folgendermaßen festgelegt:

- Eine kleine Tabelle mit DISTSTYLE ALL wird in AUTO(ALL) umgewandelt.
- Eine kleine Tabelle mit DISTSTYLE EVEN wird in AUTO(ALL) umgewandelt.
- Eine kleine Tabelle mit DISTSTYLE KEY wird in AUTO(ALL) umgewandelt.
- Eine große Tabelle mit DISTSTYLE ALL wird in AUTO(EVEN) umgewandelt.
- Eine große Tabelle mit DISTSTYLE EVEN wird in AUTO(EVEN) umgewandelt.
- Eine große Tabelle mit DISTSTYLE KEY wird in AUTO(KEY) umgewandelt und der DISTKEY bleibt erhalten. In diesem Fall nimmt Amazon Redshift keine Änderungen an der Tabelle vor.

Wenn Amazon Redshift feststellt, dass ein neuer Verteilungsstil oder Schlüssel die Leistung von Abfragen verbessert, dann kann Amazon Redshift den Verteilungsstil oder Schlüssel Ihrer Tabelle in Zukunft ändern. Beispielsweise könnte Amazon Redshift eine Tabelle mit dem DISTSTYLE-Wert AUTO(KEY) in AUTO(EVEN) konvertieren oder umgekehrt. Weitere Informationen zum Verhalten bei Änderungen von Verteilungsschlüsseln, einschließlich Datenumverteilung und Sperren, finden Sie unter Empfehlungen für Amazon Redshift Advisor.

Weitere Informationen zu DISTSTYLE AUTO finden Sie unter CREATE TABLE.

Um den Verteilungsstil einer Tabelle anzuzeigen, fragen Sie die Systemkatalogansicht SVV_TABLE_INFO ab. Weitere Informationen finden Sie unter SVV_TABLE_INFO.
Um die Empfehlungen von Amazon Redshift Advisor für Tabellen anzuzeigen, fragen Sie die Systemkatalogansicht SVV_ALTER_TABLE_RECOMMENDATIONS ab. Weitere Informationen finden Sie unter SVV_ALTER_TABLE_RECOMMENDATIONS.
Um die von Amazon Redshift durchgeführten Aktionen anzuzeigen, fragen Sie die

Systemkatalogansicht SVL_AUTO_WORKER_ACTION ab. Weitere Informationen finden Sie unter SVL_AUTO_WORKER_ACTION.

ALTER [COMPOUND] SORTKEY (column_name [,...])

Eine Klausel, die den für eine Tabelle verwendeten Sortierschlüssel ändert oder hinzufügt.

Wenn Sie einen Sortierschlüssel ändern, kann sich die Kompressionskodierung von Spalten im neuen oder ursprünglichen Sortierschlüssel ändern. Wenn für die Tabelle keine Kodierung explizit definiert ist, weist Amazon Redshift automatisch Komprimierungskodierungen wie folgt zu:

- Spalten, die als Sortierschlüssel definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als die Datentypen BOOLEAN, REAL oder DOUBLE PRECISION definiert sind, wird die RAW-Kodierung zugewiesen.
- Spalten, die als SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP oder TIMESTAMPTZ definiert sind, wird die AZ64-Komprimierung zugewiesen.
- Spalten, die als CHAR oder VARCHAR definiert sind, wird LZO-Komprimierung zugewiesen.

Berücksichtigen Sie dabei Folgendes:

- Sie können maximal 400 Spalten für einen Sortierschlüssel pro Tabelle definieren.
- Sie können einen überlappenden Sortierschlüssel in einen zusammengesetzten Sortierschlüssel oder in keinen Sortierschlüssel ändern. Es ist jedoch nicht möglich, einen zusammengesetzten Sortierschlüssel in einen überlappenden Sortierschlüssel zu ändern.
- Wenn der Sortierschlüssel zuvor als AUTO definiert war, dann ist die Tabelle kein Kandidat mehr für die automatische Tabellenoptimierung.
- Amazon Redshift empfiehlt die Verwendung von RAW-Kodierung (keine Komprimierung) für Spalten, die als Sortierschlüssel definiert sind. Wenn Sie eine Spalte ändern, um sie als Sortierschlüssel zu wählen, wird die Komprimierung der Spalte auf RAW-Komprimierung (keine Komprimierung) geändert. Dadurch kann sich der Speicherbedarf der Tabelle erhöhen. Wie stark die Tabellengröße zunimmt, hängt von der jeweiligen Tabellendefinition und dem Tabelleninhalt ab. Weitere Informationen zur Komprimierung finden Sie unter . Kompressionskodierungen

Beim Laden von Daten in eine Tabelle werden die Daten in der Reihenfolge des Sortierschlüssels geladen. Wenn Sie den Sortierschlüssel ändern, ordnet Amazon Redshift die Daten neu an. Weitere Informationen zu SORTKEY finden Sie im Abschnitt CREATE TABLE.

ALTER SORTKEY AUTO

Eine Klausel, die den Sortierschlüssel der Zieltabelle auf AUTO ändert oder hinzufügt.

Wenn Sie einen Sortierschlüssel auf AUTO ändern, behält Amazon Redshift den bestehenden Sortierschlüssel der Tabelle bei.

Wenn Amazon Redshift feststellt, dass ein neuer Sortierschlüssel die Leistung von Abfragen verbessert, dann kann Amazon Redshift den Sortierschlüssel Ihrer Tabelle in Zukunft ändern.

Weitere Informationen zu SORTKEY AUTO finden Sie im Abschnitt CREATE TABLE.

Um den Sortierschlüssel einer Tabelle anzuzeigen, fragen Sie die Systemkatalogansicht SVV_TABLE_INFO ab. Weitere Informationen finden Sie unter SVV_TABLE_INFO.
Um die Empfehlungen von Amazon Redshift Advisor für Tabellen anzuzeigen, fragen Sie die Systemkatalogansicht SVV_ALTER_TABLE_RECOMMENDATIONS ab. Weitere Informationen finden Sie unter SVV_ALTER_TABLE_RECOMMENDATIONS.
Um die von Amazon Redshift durchgeführten Aktionen anzuzeigen, fragen Sie die Systemkatalogansicht SVL_AUTO_WORKER_ACTION ab. Weitere Informationen finden Sie unter SVL_AUTO_WORKER_ACTION.

ALTER SORTKEY NONE

Eine Klausel, die den Sortierschlüssel der Zieltabelle entfernt.

Wenn der Sortierschlüssel zuvor als AUTO definiert war, dann ist die Tabelle kein Kandidat mehr für die automatische Tabellenoptimierung.

ALTER ENCODE AUTO

Eine Klausel, die den Kodierungstyp der Spalten in der Zieltabelle auf AUTO ändert. Wenn Sie die Kodierung auf AUTO ändern, behält Amazon Redshift den bestehenden Kodierungstyp der Spalten in der Tabelle bei. Wenn Amazon Redshift dann feststellt, dass ein neuer Kodierungstyp die Abfrageleistung verbessern kann, kann Amazon Redshift den Kodierungstyp der Tabellenspalten ändern.

Wenn Sie eine oder mehrere Spalten ändern, um eine Kodierung anzugeben, passt Amazon Redshift die Kodierung nicht mehr automatisch für alle Spalten in der Tabelle an. Die Spalten behalten die aktuellen Kodierungseinstellungen bei.

Die folgenden Aktionen haben keinen Einfluss auf die Einstellung ENCODE AUTO für die Tabelle:

- Umbenennen der Tabelle.
- Ändern der DISTSTYLE- oder SORTKEY-Einstellung für die Tabelle.
- Hinzufügen oder Löschen einer Spalte mit einer ENCODE-Einstellung.

- Verwendung der Option COMPUPDATE des Befehls COPY. Weitere Informationen finden Sie unter Datenladeoperationen.

Um die Kodierung einer Tabelle anzuzeigen, fragen Sie die Systemkatalogansicht SVV_TABLE_INFO ab. Weitere Informationen finden Sie unter SVV_TABLE_INFO.

RENAME COLUMN column_name TO new_name

Eine Klausel, die eine Spalte in den Wert ändert, der in new_name angegeben ist. Die maximale Länge des Spaltennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

ADD [COLUMN] column_name

Eine Klausel, die der Tabelle eine Spalte mit dem angegebenen Namen hinzufügt. Sie können in jeder Anweisung ALTER TABLE nur eine Spalte hinzufügen.

Sie können keine Spalte hinzufügen, die der Verteilungsschlüssel (DISTKEY) oder ein Sortierschlüssel (SORTKEY) der Tabelle ist.

Sie können den Befehl ALTER TABLE ADD COLUMN nicht verwenden, um die folgenden Tabellen- und Spaltenattribute zu ändern:

- UNIQUE
- PRIMARY KEY
- REFERENCES (Fremdschlüssel)
- IDENTITY oder GENERATED BY DEFAULT AS IDENTITY

Die maximale Länge des Spaltennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Die maximale Anzahl der Spalten, die Sie in einer einzelnen Tabelle definieren können, ist 1.600.

Beim Hinzufügen von Spalten zu einer externen Tabelle gelten die folgenden Beschränkungen:

- Sie können keine Spalten zu einer externen Tabelle mit den Spaltenbeschränkungen DEFAULT, ENCODE; NOT NULL oder NULL hinzufügen.
- Sie können keine Spalten zu einer externen Tabelle hinzufügen, die mit dem AVRO-Dateiformat definiert ist.
- Falls Pseudospalten aktiviert sind, können Sie höchstens 1.598 Spalten in einer einzelnen externen Tabelle definieren. Falls Pseudospalten nicht aktiviert sind, können Sie höchstens 1.600 Spalten in einer einzelnen Tabelle definieren.

Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.

column_type

Der Datentyp der Spalte, die hinzugefügt wird. Im Fall der Spalten CHAR und VARCHAR können Sie das Schlüsselwort MAX verwenden, statt eine maximale Länge zu deklarieren. MAX legt die maximale Länge auf 4.096 Bytes für CHAR oder 65.535 Bytes für VARCHAR fest. Die maximale Größe eines GEOMETRY-Objekts beträgt 1.048.447 Byte..

Informationen zu den Datentypen, die Amazon Redshift unterstützt, finden Sie unter Datentypen.

DEFAULT default_expr

Eine Klausel, die der Spalte einen Standarddatenwert zuweist. Der Datentyp von default_expr muss dem Datentyp der Spalte entsprechen. Der Wert DEFAULT muss ein variablenloser Ausdruck sein. Unterabfragen, Querreferenzen auf andere Spalten in der aktuellen Tabelle und benutzerdefinierte Funktionen sind nicht zulässig.

Der default_expr wird in jeder INSERT-Operation verwendet, die keinen Wert für die Spalte angibt. Wenn kein Standardwert angegeben ist, ist der Standardwert für die Spalte null.

Wenn eine COPY-Operation auf ein Null-Feld in einer Spalte mit dem Wert DEFAULT und der Einschränkung NOT NULL trifft, fügt der Befehl COPY den Wert von default_expr ein.

DEFAULT wird für externe Tabellen nicht unterstützt.

ENCODE encoding

Die Kompressionskodierung für eine Spalte. Standardmäßig verwaltet Amazon Redshift die Komprimierungskodierung für alle Spalten in einer Tabelle automatisch, wenn Sie keine Komprimierungskodierung für eine Spalte in der Tabelle angeben oder wenn Sie die Option ENCODE AUTO für die Tabelle verwenden.

Wenn Sie die Komprimierungskodierung für eine beliebige Spalte in der Tabelle angeben oder nicht die Option ENCODE AUTO für die Tabelle angeben, weist Amazon Redshift den Spalten, für die Sie keine Komprimierungskodierung angeben, automatisch die Komprimierungskodierung wie folgt zu:

- Allen Spalten in temporären Tabellen wird standardmäßig die RAW-Kompression zugewiesen.
- Spalten, die als Sortierschlüssel definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als Datentyp BOOLEAN, REAL, DOUBLE PRECISION, GEOMETRY oder GEOGRAPHY definiert sind, wird die RAW-Kompression zugewiesen.

- Spalten, die als SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP oder TIMESTAMPTZ definiert sind, wird die AZ64-Komprimierung zugewiesen.
- Spalten, die als CHAR, VARCHAR oder VARBYTE definiert sind, wird LZO-Komprimierung zugewiesen.

> **Note**
>
> Wenn Sie nicht möchten, dass eine Spalte komprimiert wird, geben Sie die RAW-Kodierung explizit an.

Die folgenden compression encodings (p. 70) werden unterstützt:

- AZ64
- BYTEDICT
- DELTA
- DELTA32K
- LZO
- MOSTLY8
- MOSTLY16
- MOSTLY32
- RAW (keine Kompression)
- RUNLENGTH
- TEXT255
- TEXT32K
- ZSTD

ENCODE wird für externe Tabellen nicht unterstützt.

NOT NULL | NULL

NOT NULL gibt an, dass die Spalte keine Null-Werte enthalten darf. Der Standardwert NULL gibt an, dass die Spalte Null-Werte akzeptiert.

NOT NULL und NULL werden für externe Tabellen nicht unterstützt.

COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }

Eine Klausel, die angibt, ob bei der Suche oder Vergleichen in der Spalte zwischen Groß- und Kleinschreibung unterschieden wird (CASE_SENSITIVE) oder nicht (CASE_INSENSITIVE). Der Standardwert entspricht der aktuellen Konfiguration für die Beachtung der Groß-/Kleinschreibung in der Datenbank.

Verwenden Sie den folgenden Befehl, um die Sortierinformationen der Datenbank zu ermitteln:

```
SELECT db_collation();

db_collation
----------------
 case_sensitive
(1 row)
```

DROP [COLUMN] column_name

Der Name der Spalte, die aus der Tabelle gelöscht werden soll.

Die letzte Tabellenspalte kann nicht gelöscht werden. Eine Tabelle muss mindestens eine Spalte enthalten.

Sie können keine Spalte entfernen, die der Verteilungsschlüssel (DISTKEY) oder ein Sortierschlüssel (SORTKEY) der Tabelle ist. Das Standardverhalten für DROP COLUMN ist RESTRICT, wenn die Spalte abhängige Objekte enthält, wie eine Ansicht, einen Primärschlüssel, einen Fremdschlüssel oder die Einschränkung UNIQUE.

Beim Hineinziehen aus einer externen Tabelle gelten die folgenden Beschränkungen:

- Spalten, die als Partition verwendet werden, können nicht aus einer externen Tabelle entfernt werden.
- Sie können keine Spalten aus einer externen Tabelle entfernen, die mit dem AVRO-Dateiformat definiert ist.
- RESTRICT und CASCADE werden für externe Tabellen ignoriert.
- Sie können die Spalten der Richtlinientabelle, auf die innerhalb der Richtliniendefinition verwiesen wird, nicht löschen, es sei denn, Sie löschen oder trennen die Richtlinie. Dies gilt auch, wenn die Option CASCADE angegeben ist. Sie können andere Spalten in der Richtlinientabelle löschen.

Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.

RESTRICT

Wenn RESTRICT mit DROP COLUMN verwendet wird, bedeutet das, dass die Spalte, die entfernt werden soll, in den folgenden Fällen nicht entfernt wird:

- Wenn eine definierte Ansicht die Spalte referenziert, die entfernt werden soll
- Wenn ein Fremdschlüssel die Spalte referenziert
- Wenn die Spalte Teil eines mehrteiligen Schlüssels ist

RESTRICT kann nicht mit CASCADE verwendet werden.

RESTRICT und CASCADE werden für externe Tabellen ignoriert.

CASCADE

Bei Verwendung mit DROP COLUMN wird die angegebene Spalte und alles, was von dieser Spalte abhängig ist, entfernt. CASCADE kann nicht mit RESTRICT verwendet werden.

RESTRICT und CASCADE werden für externe Tabellen ignoriert.

Die folgenden Optionen gelten nur für externe Tabellen.

SET LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }

Der Pfad zum Amazon S3-Ordner enthält die Datendateien oder eine Manifestdatei, die eine Liste der Amazon S3-Objektpfade enthält. Die Buckets müssen sich in derselben AWS -Region wie der Amazon-Redshift-Cluster befinden. Eine Liste der unterstützten AWS Regionen finden Sie unter. Überlegungen zu Amazon Redshift Spectrum Weitere Informationen zur Verwendung einer Manifestdatei finden Sie unter LOCATION in der CREATE EXTERNAL TABLE-Referenz Parameter.

SET FILE FORMAT format

Das Dateiformat für externe Datendateien.

Gültige Formate sind folgende:

- AVRO
- PARQUET
- RCFILE
- SEQUENCEFILE
- TEXTFILE

SET TABLE PROPERTIES ('property_name'='property_value')

Eine Klausel, die die Tabellendefinition für Tabelleneigenschaften für eine externe Tabelle festlegt.

> **Note**
>
> Bei Tabelleneigenschaften muss die Groß-/Kleinschreibung beachtet werden.

'numRows'='row_count'

Eine Eigenschaft, die den Wert numRows für die Tabellendefinition festlegt. Um die Statistiken einer externen Tabelle explizit zu aktualisieren, legen Sie mit der Eigenschaft numRows die Größe der Tabelle fest. Amazon Redshift analysiert keine externen Tabellen, um die Tabellenstatistiken zu generieren, die der Abfrageoptimierer verwendet, um einen Abfrageplan zu erstellen. Wenn für eine externe Tabelle keine Tabellenstatistiken festgelegt sind, generiert Amazon Redshift einen Abfrageausführungsplan. Dieser Plan basiert auf der Annahme, dass externe Tabellen die größeren Tabellen und lokale Tabellen die kleineren Tabellen sind.

'skip.header.line.count'='line_count'

Eine Eigenschaft, die die Anzahl der Reihen festgelegt, die am Anfang jeder Quelldatei übersprungen wird.

PARTITION (partition_column=partition_value [, ...] SET LOCATION { 's3://bucket/folder' | 's3://bucket/manifest_file' }

Eine Klausel, die einen neuen Speicherort für eine oder mehrere Partitionsspalten festlegt.

ADD [IF NOT EXISTS] PARTITION (partition_column=partition_value [, ...]) LOCATION { 's3://bucket/folder' | 's3://bucket/manifest_file' } [, ...]

Eine Klausel, die eine oder mehrere Partitionen hinzufügt. Sie können mehrere PARTITION-Klauseln mit einer einzelnen ALTER TABLE … ADD-Anweisung angeben.

> **Note**
>
> Wenn Sie den AWS Glue Katalog verwenden, können Sie mit einer einzigen ALTER TABLE-Anweisung bis zu 100 Partitionen hinzufügen.

Die IF NOT EXISTS-Klausel gibt an, dass der Befehl keine Änderungen ausführen soll, wenn die angegebene Partition bereits vorhanden ist. Darüber hinaus gibt sie an, dass der Befehl eine Meldung zurückgeben sollte, dass die Partition vorhanden ist, anstatt mit einem Fehler beendet zu werden. Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn ALTER TABLE versucht, eine Partition hinzuzufügen, die bereits vorhanden ist.

DROP PARTITION (partition_column=partition_value [, ...])

Eine Klausel, die die angegebene Partition entfernt. Durch die Entfernung einer Partition werden nur die Metadaten der externen Tabelle geändert. Die Daten auf Amazon S3 sind nicht betroffen.

ROW LEVEL SECURITY { ON | OFF } [CONJUNCTION TYPE { AND | OR }] [FOR DATASHARES]

Eine Klausel, die die Sicherheit auf Zeilenebene für eine Relation aktiviert oder deaktiviert.

Wenn die Sicherheit auf Zeilenebene für eine Relation aktiviert ist, können Sie nur die Zeilen lesen, für die Ihnen die Richtlinie auf Zeilenebene Zugriff gewährt. Wenn es keine Richtlinie gibt, die Ihnen Zugriff auf die Relation gewährt, können Sie keine Zeilen aus der Relation sehen. Nur Superuser und Benutzer oder Rollen, denen die `sys:secadmin`-Rolle zugewiesen ist, können die Klausel ROW LEVEL SECURITY festlegen. Weitere Informationen finden Sie unter Sicherheit auf Zeilenebene.

- [CONJUNCTION TYPE { AND | OR }]

  Eine Klausel, mit der Sie den Verbindungstyp einer Sicherheitsrichtlinie auf Zeilenebene für eine Relation auswählen können. Wenn einer Relation mehrere Sicherheitsrichtlinien auf Zeilenebene zugeordnet sind, können Sie die Richtlinien mit der AND- oder OR-Klausel kombinieren. Standardmäßig kombiniert Amazon Redshift RLS-Richtlinien mit der AND-Klausel. Superuser, Benutzer oder Rollen, denen diese `sys:secadmin`-Rolle zugewiesen wurde, können diese Klausel verwenden, um den Verbindungstyp der Sicherheitsrichtlinie auf Zeilenebene für eine Relation zu definieren. Weitere Informationen finden Sie unter Kombinieren mehrerer Richtlinien pro Benutzer.

- FOR DATASHARES

  Eine Klausel, die festlegt, ob auf eine RLS-geschützte Relation über Datashares zugegriffen werden kann. Standardmäßig kann auf eine RLS-geschützte Relation nicht über ein Datashare zugegriffen werden. Ein ALTER TABLE ROW LEVEL SECURITY-Befehl, der mit dieser Klausel ausgeführt wird, wirkt sich nur auf die Zugänglichkeitseigenschaft des Datashares aus. Die Eigenschaft ROW LEVEL SECURITY wird nicht geändert.

Wenn Sie eine RLS-geschützte Relation über Datashares zugänglich machen, bietet die Relation in der verbraucherseitigen, gemeinsam genutzten Datenbank keine Sicherheit auf Zeilenebene. Die Relation behält ihre RLS-Eigenschaft auf der Produzentenseite.

Beispiele

In den folgenden Abschnitten finden Sie Beispiele für die Verwendung des Befehls ALTER TABLE.

- Beispiele für ALTER TABLE
- Beispiele für ALTER EXTERNAL TABLE
- Beispiele für ALTER TABLE ADD und DROP COLUMN

Beispiele für ALTER TABLE

Das folgende Beispiele zeigt die grundsätzliche Nutzung des Befehls ALTER TABLE.

Umbenennen einer Tabelle oder Ansicht

Mit dem folgenden Befehl wird die Tabelle USERS in USERS_BKUP umbenannt:

```
alter table users
rename to users_bkup;
```

Sie können diese Art von Befehl auch verwenden, um eine Ansicht umzubenennen.

Ändern des Besitzers einer Tabelle oder Ansicht

Mit dem folgenden Befehl wird der Besitzer der Tabelle VENUE in den Benutzer DWUSER geändert:

```
alter table venue
owner to dwuser;
```

Die folgenden Befehle erstellen eine Ansicht und ändern anschließend deren Besitzer:

```
create view vdate as select * from date;
alter table vdate owner to vuser;
```

Umbenennen einer Spalte

Mit dem folgenden Befehl wird die Spalte VENUESEATS in der Tabelle VENUE in VENUESIZE geändert:

```
alter table venue
rename column venueseats to venuesize;
```

Entfernen einer Tabelleneinschränkung

Um eine Tabelleneinschränkung zu entfernen, wie beispielsweise einen Primärschlüssel, einen Fremdschlüssel oder eine spezifische Einschränkung, suchen Sie zunächst den internen Namen der Einschränkung. Geben Sie anschließend den Namen der Einschränkung im Befehl ALTER TABLE ein. Im folgenden Beispiel werden die Einschränkungen für die Tabelle CATEGORY gesucht und anschließend der Primärschlüssel mit dem Namen entfernt `category_pkey`.

```
select constraint_name, constraint_type
from information_schema.table_constraints
where constraint_schema ='public'
and table_name = 'category';

constraint_name | constraint_type
----------------+----------------
category_pkey   | PRIMARY KEY

alter table category
drop constraint category_pkey;
```

Ändern einer VARCHAR-Spalte

Um Speicherplatz zu sparen, können Sie eine Tabelle zunächst mit VARCHAR-Spalten mit der Mindestgröße definieren, die für Ihre aktuellen Datenanforderungen erforderlich ist. Später können Sie längere Zeichenfolgen unterstützen, indem Sie die Spaltengröße in der Tabelle ändern.

Im folgenden Beispiel wird die Größe der EVENTNAME-Spalte auf VARCHAR(300) erhöht.

```
alter table event alter column eventname type varchar(300);
```

Ändern der Komprimierungskodierung für eine Spalte

Sie können die Komprimierungskodierung einer Spalte ändern. Nachstehend finden Sie eine Reihe von Beispielen, die diesen Ansatz veranschaulichen. Die Tabellendefinition für diese Beispielen lautet wie folgt.

```
create table t1(c0 int encode lzo, c1 bigint encode zstd, c2 varchar(16) encode lzo, c3
 varchar(32) encode zstd);
```

Die folgende Anweisung ändert die Komprimierungskodierung für Spalte c0 von LZO-Kodierung in AZ64-Kodierung.

```
alter table t1 alter column c0 encode az64;
```

Die folgende Anweisung ändert die Komprimierungskodierung für Spalte c1 von Zstandard-Kodierung in AZ64-Kodierung.

```
alter table t1 alter column c1 encode az64;
```

Die folgende Anweisung ändert die Komprimierungskodierung für Spalte c2 von LZO-Kodierung in Byte-Verzeichnis-Kodierung.

```
alter table t1 alter column c2 encode bytedict;
```

Die folgende Anweisung ändert die Komprimierungskodierung für Spalte c3 von Zstandard-Kodierung in Runlength-Kodierung.

```
alter table t1 alter column c3 encode runlength;
```

Ändern einer DISTSTYLE KEY DISTKEY-Spalte

Die folgenden Beispiele illustrieren das Ändern von DISTSTYLE und DISTKEY in einer Tabelle.

Erstellen Sie eine Tabelle mit der Verteilung EVEN. Die Ansicht SVV_TABLE_INFO zeigt für DISTSTYLE den Wert EVEN.

```
create table inventory(
  inv_date_sk int4 not null ,
```

```
  inv_item_sk int4 not null ,
  inv_warehouse_sk int4 not null ,
  inv_quantity_on_hand int4
) diststyle even;

Insert into inventory values(1,1,1,1);

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+---------------
 inventory |     EVEN
```

Ändern Sie die Tabelle DISTKEY in `inv_warehouse_sk`. Die Ansicht SVV_TABLE_INFO zeigt die Spalte `inv_warehouse_sk` als resultierenden Verteilerschlüssel an.

```
alter table inventory alter diststyle key distkey inv_warehouse_sk;

select "table", "diststyle" from svv_table_info;

   table   |       diststyle
-----------+-----------------------
 inventory | KEY(inv_warehouse_sk)
```

Ändern Sie die Tabelle DISTKEY in `inv_item_sk`. Die Ansicht SVV_TABLE_INFO zeigt die Spalte `inv_item_sk` als resultierenden Verteilerschlüssel an.

```
alter table inventory alter distkey inv_item_sk;

select "table", "diststyle" from svv_table_info;

   table   |       diststyle
-----------+-----------------------
 inventory | KEY(inv_item_sk)
```

Ändern einer Tabelle in DISTSTYLE ALL

Die folgenden Beispiele zeigen, wie Sie eine Tabelle in DISTSTYLE ALL ändern.

Erstellen Sie eine Tabelle mit der Verteilung EVEN. Die Ansicht SVV_TABLE_INFO zeigt für DISTSTYLE den Wert EVEN.

```
create table inventory(
  inv_date_sk int4 not null ,
  inv_item_sk int4 not null ,
  inv_warehouse_sk int4 not null ,
  inv_quantity_on_hand int4
) diststyle even;

Insert into inventory values(1,1,1,1);

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+----------------
 inventory |     EVEN
```

Ändern Sie die Tabelle DISTSTYLE in ALL. Die Ansicht SVV_TABLE_INFO zeigt die geänderte DISTSYTLE an.

```
alter table inventory alter diststyle all;

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+----------------
 inventory |     ALL
```

Ändern des SORTKEY einer Tabelle

Sie können festlegen, ob eine Tabelle einen zusammengesetzten Sortierschlüssel oder keinen Sortierschlüssel haben soll.

In der folgenden Tabellendefinition ist die Tabelle `t1` mit einem überlappenden Sortierschlüssel definiert.

```
create table t1 (c0 int, c1 int) interleaved sortkey(c0, c1);
```

Der folgende Befehle ändert die Tabelle so, dass aus dem überlappenden Sortierschlüssel ein zusammengesetzter Sortierschlüssel wird.

```
alter table t1 alter sortkey(c0, c1);
```

Der folgende Befehl ändert die Tabelle so, dass der überlappende Sortierschlüssel entfernt wird.

```
alter table t1 alter sortkey none;
```

In der folgenden Tabellendefinition ist die Tabelle `t1` mit der Spalte `c0` als Sortierschlüssel definiert.

```
create table t1 (c0 int, c1 int) sortkey(c0);
```

Der folgende Befehle ändert die Tabelle `t1` in einen zusammengesetzten Sortierschlüssel.

```
alter table t1 alter sortkey(c0, c1);
```

Ändern einer Tabelle in ENCODE AUTO

Das folgende Beispiel zeigt, wie Sie eine Tabelle in ENCODE AUTO ändern.

Die Tabellendefinition für dieses Beispiel lautet wie folgt. Spalte `c0` ist mit dem Kodierungstyp AZ64 definiert, und Spalte `c1` ist mit dem Kodierungstyp LZO definiert.

```
create table t1(c0 int encode AZ64, c1 varchar encode LZO);
```

Die folgende Anweisung ändert die Kodierung dieser Tabelle auf AUTO.

```
alter table t1 alter encode auto;
```

Das folgende Beispiel zeigt, wie Sie eine Tabelle ändern, um die Einstellung ENCODE AUTO zu entfernen.

Die Tabellendefinition für dieses Beispiel lautet wie folgt. Die Tabellenspalten werden ohne Kodierung definiert. In diesem Fall ist die Kodierung standardmäßig auf ENCODE AUTO eingestellt.

```
create table t2(c0 int, c1 varchar);
```

Die folgende Anweisung ändert die Kodierung für Spalte c0 dieser Tabelle in LZO. Die Tabellenkodierung ist nicht mehr auf ENCODE AUTO eingestellt.

```
alter table t2 alter column c0 encode lzo;;
```

Ändern einer Kontrolle der Sicherheit auf Zeilenebene

Der folgende Befehl schaltet RLS für die Tabelle aus:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY OFF;
```

Der folgende Befehl aktiviert RLS für die Tabelle:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
```

Der folgende Befehl aktiviert RLS für die Tabelle und macht es über Datashares zugänglich:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY FOR DATASHARES OFF;
```

Der folgende Befehl aktiviert RLS für die Tabelle und macht es über Datashares unzugänglich:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY FOR DATASHARES ON;
```

Der folgende Befehl aktiviert RLS und setzt den RLS-Verbindungstyp für die Tabelle auf OR:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON CONJUNCTION TYPE OR;
```

Der folgende Befehl aktiviert RLS und setzt den RLS-Verbindungstyp für die Tabelle auf AND:

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON CONJUNCTION TYPE AND;
```

Beispiele für ALTER EXTERNAL TABLE

Die folgenden Beispiele verwenden einen Amazon S3 S3-Bucket in der Region USA Ost (Nord-Virginia) (`us-east-1`) AWS-Region und die in Beispiele für CREATE TABLE erstellten Beispieltabellen. Weitere Informationen zur Verwendung von Partitionen mit externen Tabellen finden Sie unterPartitionierung externer Redshift-Spectrum-Tabellen.

Im folgenden Beispiel wird die Tabelleneigenschaft numRows für die externe Tabelle SPECTRUM.SALES auf 170.000 Zeilen festgelegt.

```
alter table spectrum.sales
```

```
set table properties ('numRows'='170000');
```

Im folgenden Beispiel wird der Speicherort der externen Tabelle SPECTRUM.SALES geändert.

```
alter table spectrum.sales
set location 's3://redshift-downloads/tickit/spectrum/sales/';
```

Im folgenden Beispiel wird das Format der externen Tabelle SPECTRUM.SALES in Parquet geändert.

```
alter table spectrum.sales
set file format parquet;
```

Im folgenden Beispiel wird der externen Tabelle SPECTRUM.SALES_PART eine Partition hinzugefügt.

```
alter table spectrum.sales_part
add if not exists partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/';
```

Im folgenden Beispiel werden der Tabelle SPECTRUM.SALES_PART drei Partitionen hinzugefügt.

```
alter table spectrum.sales_part add if not exists
partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/'
partition(saledate='2008-02-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/'
partition(saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/';
```

Im folgenden Beispiel wird die externe Tabelle SPECTRUM.SALES_PART dahingehend geändert, dass die Partition entfernt wird `saledate='2008-01-01'`.

```
alter table spectrum.sales_part
drop partition(saledate='2008-01-01');
```

Im folgenden Beispiel wird ein neuer Amazon S3-Pfad für die Partition mit festgelegt `saledate='2008-01-01'`.

```
alter table spectrum.sales_part
```

```
partition(saledate='2008-01-01')
set location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-01-01/';
```

Das folgende Beispiel ändert den Namen `sales_date` in `transaction_date`.

```
alter table spectrum.sales rename column sales_date to transaction_date;
```

Im folgenden Beispiel wird das Spalten-Mapping auf die Positionszuweisung zu einer externen Tabelle gesetzt, die das ORC (Optimized Row Columnar) Format verwendet.

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='position');
```

Im folgenden Beispiel wird das Spalten-Mapping auf die Namenszuweisung zu einer externen Tabelle gesetzt, die das ORC (Optimized Row Columnar) Format verwendet.

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='name');
```

Beispiele für ALTER TABLE ADD und DROP COLUMN

In den folgenden Beispielen wird gezeigt, wie ALTER TABLE verwendet wird, um eine einfache Tabellenspalte hinzuzufügen und dann zu entfernen und eine Spalte mit einem abhängigen Objekt zu entfernen.

Hinzufügen (ADD) und Entfernen (DROP) einer einfachen Spalte

Im folgenden Beispiel wird der Tabelle USERS die eigenständige Spalte FEEDBACK_SCORE hinzugefügt. Diese Spalte enthält einfach eine Ganzzahl und der Standardwert für diese Spalte ist NULL (kein Feedbackergebnis).

Führen Sie zunächst eine Abfrage für die Katalogtabelle PG_TABLE_DEF aus, um das Schema der Tabelle USERS anzuzeigen:

```
column        | type                    | encoding | distkey | sortkey
--------------+-------------------------+----------+---------+--------
userid        | integer                 | delta    | true    |       1
username      | character(8)            | lzo      | false   |       0
firstname     | character varying(30)   | text32k  | false   |       0
lastname      | character varying(30)   | text32k  | false   |       0
```

```
city          | character varying(30)  | text32k  | false   |       0
state         | character(2)           | bytedict | false   |       0
email         | character varying(100) | lzo      | false   |       0
phone         | character(14)          | lzo      | false   |       0
likesports    | boolean                | none     | false   |       0
liketheatre   | boolean                | none     | false   |       0
likeconcerts  | boolean                | none     | false   |       0
likejazz      | boolean                | none     | false   |       0
likeclassical | boolean                | none     | false   |       0
likeopera     | boolean                | none     | false   |       0
likerock      | boolean                | none     | false   |       0
likevegas     | boolean                | none     | false   |       0
likebroadway  | boolean                | none     | false   |       0
likemusicals  | boolean                | none     | false   |       0
```

Fügen Sie nun die Spalte feedback_score hinzu:

```
alter table users
add column feedback_score int
default NULL;
```

Wählen Sie die Spalte FEEDBACK_SCORE aus USERS aus, um zu überprüfen, ob sie hinzugefügt wurde:

```
select feedback_score from users limit 5;

feedback_score
----------------
NULL
NULL
NULL
NULL
NULL
```

Entfernen Sie die Spalte, um die ursprüngliche DDL wiederherzustellen:

```
alter table users drop column feedback_score;
```

Entfernen einer Spalte mit einem abhängigen Objekt

In diesem Beispiel wird eine Spalte entfernt, die ein abhängiges Objekt enthält. Daher wird das abhängige Objekt ebenfalls entfernt.

Fügen Sie zunächst der Tabelle USERS erneut die Spalte FEEDBACK_SCORE hinzu:

```
alter table users
add column feedback_score int
default NULL;
```

Erstellen Sie als Nächstes eine Ansicht aus der Tabelle USERS mit dem Namen USERS_VIEW:

```
create view users_view as select * from users;
```

Versuchen Sie nun, die Spalte FEEDBACK_SCORE aus der Spalte USERS zu entfernen. Diese DROP-Anweisung verwendet das Standardverhalten (RESTRICT):

```
alter table users drop column feedback_score;
```

Amazon Redshift zeigt die Fehlermeldung an, dass die Spalte nicht entfernt werden kann, da ein anderes Objekt von ihr abhängig ist.

Versuchen Sie erneut, die Spalte FEEDBACK_SCORE zu entfernen. Geben Sie dieses Mal CASCADE an, um alle abhängigen Objekte zu entfernen:

```
alter table users
drop column feedback_score cascade;
```

ALTER TABLE APPEND

Fügt einer Zieltabelle Zeilen hinzu, indem Daten aus einer vorhandenen Quelltabelle verschoben werden. Daten in der Quelltabelle werden zu entsprechenden Spalten in der Zieltabelle verschoben. Die Reihenfolge der Spalten spielt keine Rolle. Nachdem die Daten der Zieltabelle erfolgreich angefügt wurden, ist die Quelltabelle leer. ALTER TABLE APPEND ist in der Regel sehr viel schneller als die vergleichbaren Operationen CREATE TABLE AS oder INSERT, da die Daten verschoben und nicht dupliziert werden.

> **Note**
>
> ALTER TABLE APPEND verschiebt Datenblöcke zwischen der Quelltabelle und der Zieltabelle. Um die Leistung zu verbessern, komprimiert ALTER TABLE APPEND als Teil der append-Operation keinen Speicher. Dies hat zur Folge, dass die Nutzung von Arbeitsspeicher

vorübergehend zunimmt. Um den Speicherplatz wieder zurückzugewinnen, führen Sie eine VACUUM-Operation durch.

Spalten mit den gleichen Namen müssen identische Spaltenattribute besitzen. Wenn weder die Quelltabelle noch die Zieltabelle Spalten enthält, die in der jeweils anderen Tabelle nicht vorhanden sind, verwenden Sie die Parameter IGNOREEXTRA oder FILLTARGET, um anzugeben, wie zusätzliche Spalten verwaltet werden sollen.

Sie können keine Identitätsspalten anfügen. Wenn beide Tabellen eine Identitätsspalte enthalten, ist der Befehl nicht erfolgreich. Wenn nur eine Tabelle eine Identitätsspalte enthält, verwenden Sie die Parameter FILLTARGET oder IGNOREEXTRA. Weitere Informationen finden Sie unter Nutzungshinweise für ALTER TABLE APPEND.

Sie können eine GENERATED BY DEFAULT AS IDENTITY-Spalte anhängen. Sie können als GENERATED BY DEFAULT AS IDENTITY-Spalten mit von Ihnen angegebenen Werten aktualisieren. Weitere Informationen finden Sie unter Nutzungshinweise für ALTER TABLE APPEND.

Bei der Zieltabelle muss es sich um eine permanente Tabelle handeln. Die Quelle kann jedoch eine permanente Tabelle oder eine materialisierte Ansicht sein, die für die Streaming-Erfassung konfiguriert ist. Beide Objekte müssen den gleichen Verteilungsstil und Verteilungsschlüssel verwenden, wenn einer definiert wurde. Wenn die Objekte sortiert sind, müssen beide Objekte den gleichen Sortierstil verwenden und die gleichen Spalten als Sortierschlüssel definieren.

Der Befehl ALTER TABLE APPEND führt automatisch vor Abschluss der Operation einen Commit aus. Es kann kein Rollback ausgeführt werden. Sie können ALTER TABLE APPEND nicht innerhalb eines Transaktionsblocks (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Erforderliche Berechtigungen

Je nach verwendetem Befehl ALTER TABLE APPEND ist eine der folgenden Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Systemberechtigung ALTER TABLE
- Benutzer mit Berechtigungen DELETE und SELECT für die Quelltabelle und der Berechtigung INSERT für die Zieltabelle

Syntax

```
ALTER TABLE target_table_name APPEND FROM [ source_table_name
 | source_materialized_view_name ]
[ IGNOREEXTRA | FILLTARGET ]
```

Das Anhängen aus einer materialisierten Ansicht funktioniert nur, wenn Ihre materialisierte Ansicht für Streaming-Erfassung konfiguriert ist.

Parameter

target_table_name

Der Name der Tabelle, an die die Zeilen angefügt werden. Geben Sie einfach den Namen der Tabelle an, oder verwenden Sie das Format schema_name.table_name, um ein spezifisches Schema zu verwenden. Bei der Zieltabelle muss es sich um eine vorhandene permanente Tabelle handeln.

FROM source_table_name

Der Name der Tabelle, die die Zeilen bereitstellt, die angefügt werden sollen. Geben Sie einfach den Namen der Tabelle an, oder verwenden Sie das Format schema_name.table_name, um ein spezifisches Schema zu verwenden. Bei der Quelltabelle muss es sich um eine vorhandene permanente Tabelle handeln.

FROM source_materialized_view_name

Der Name der materialisierten Ansicht, die die Zeilen bereitstellt, die angefügt werden sollen. Das Anhängen aus einer materialisierten Ansicht funktioniert nur, wenn Ihre materialisierte Ansicht für Streaming-Erfassung konfiguriert ist. Die materialisierte Quellansicht muss bereits vorhanden sein.

IGNOREEXTRA

Ein Schlüsselwort, das angibt, dass Daten in den zusätzlichen Spalten verworfen werden sollen, wenn die Quelltabelle Spalten enthält, die nicht in der Zieltabelle vorhanden sind. Sie können IGNOREEXTRA nicht mit FILLTARGET verwenden.

FILLTARGET

Ein Schlüsselwort, das angibt, dass die Spalten mit dem Spaltenwert DEFAULT, wenn definiert, oder NULL ausgefüllt werden sollen, wenn die Zieltabelle Spalten enthält, die nicht in der Quelltabelle vorhanden sind. Sie können IGNOREEXTRA nicht mit FILLTARGET verwenden.

Nutzungshinweise für ALTER TABLE APPEND

ALTER TABLE APPEND verschiebt nur identische Spalten aus der Quelltabelle in die Zieltabelle. Die Reihenfolge der Spalten spielt keine Rolle.

Wenn weder die Quelltabelle noch die Zieltabelle zusätzliche Spalten enthält, verwenden Sie FILLTARGET oder IGNOREEXTRA entsprechend den folgenden Regeln:

- Wenn die Quelltabelle Spalten enthält, die in der Zieltabelle nicht vorhanden sind, verwenden Sie IGNOREEXTRA. Durch diesen Befehl werden die zusätzlichen Spalten in der Quelltabelle ignoriert.
- Wenn die Zieltabelle Spalten enthält, die in der Quelltabelle nicht vorhanden sind, verwenden Sie FILLTARGET. Der Befehl füllt die zusätzlichen Spalten in der Zieltabelle entweder mit dem Standardwert für die Spalte oder dem Wert IDENTITY aus, wenn definiert, oder NULL.
- Wenn sowohl die Quelltabelle als auch die Zieltabelle zusätzliche Spalten enthalten, ist der Befehl nicht erfolgreich. Sie können nicht FILLTARGET und IGNOREEXTRA gleichzeitig verwenden.

Wenn in den beiden Tabellen eine Spalte mit dem gleichen Namen, jedoch unterschiedlichen Attributen vorhanden ist, ist der Befehl nicht erfolgreich. Gleich benannten Spalten müssen die folgenden Attribute gemeinsam sein:

- Datentyp
- Spaltengröße
- Kompressionskodierung
- Nicht null
- Sortierstil
- Sortierschlüsselspalten
- Verteilungsstil
- Verteilungschlüsselspalten

Sie können keine Identitätsspalten anfügen. Wenn sowohl die Quelltabelle als auch die Zieltabelle Identitätsspalten enthalten, ist der Befehl nicht erfolgreich. Wenn nur die Quelltabelle eine Identitätsspalte enthält, verwenden Sie den Parameter IGNOREXTRA, sodass die Identitätsspalte ignoriert wird. Wenn nur die Zieltabelle eine Identitätsspalte enthält, verwenden Sie den Parameter FILLTARGET, sodass die Identitätsspalte entsprechend der IDENTITY-Klausel ausgefüllt wird, die für die Tabelle definiert ist. Weitere Informationen finden Sie unter DEFAULT.

Sie können eine Standard-Identity-Spalte mit der Anweisung ALTER TABLE APPEND anhängen. Weitere Informationen finden Sie unter CREATE TABLE.

Beispiele für ALTER TABLE APPEND

Nehmen Sie an, Ihre Organisation pflegt eine Tabelle namens SALES_MONTHLY, um aktuelle Verkaufstransaktionen zu erfassen. Sie möchten jeden Monat Daten aus der Transaktionstabelle in die Tabelle SALES verschieben.

Sie können die Befehle INSERT INTO und TRUNCATE verwenden, um die Aufgabe auszuführen.

```
insert into sales (select * from sales_monthly);
truncate sales_monthly;
```

Sie können diese Operation jedoch sehr viel effizienter ausführen, indem Sie den Befehl ALTER TABLE APPEND verwenden.

Führen Sie zunächst eine Abfrage für die Systemkatalogtabelle PG_TABLE_DEF aus, um zu überprüfen, ob beide Tabellen gleiche Spalten mit identischen Spaltenattributen enthalten.

```
select trim(tablename) as table, "column", trim(type) as type,
encoding, distkey, sortkey, "notnull"
from pg_table_def where tablename like 'sales%';

table      | column     | type                        | encoding | distkey | sortkey |
 notnull
-----------+------------+-----------------------------+----------+---------+---------
+--------
sales      | salesid    | integer                     | lzo      | false   |       0 |
 true
sales      | listid     | integer                     | none     | true    |       1 |
 true
sales      | sellerid   | integer                     | none     | false   |       2 |
 true
sales      | buyerid    | integer                     | lzo      | false   |       0 |
 true
sales      | eventid    | integer                     | mostly16 | false   |       0 |
 true
sales      | dateid     | smallint                    | lzo      | false   |       0 |
 true
sales      | qtysold    | smallint                    | mostly8  | false   |       0 |
 true
```

```
sales      | pricepaid  | numeric(8,2)                | delta32k | false   |       0 |
 false
sales      | commission | numeric(8,2)                | delta32k | false   |       0 |
 false
sales      | saletime   | timestamp without time zone | lzo      | false   |       0 |
 false
salesmonth | salesid    | integer                     | lzo      | false   |       0 |
 true
salesmonth | listid     | integer                     | none     | true    |       1 |
 true
salesmonth | sellerid   | integer                     | none     | false   |       2 |
 true
salesmonth | buyerid    | integer                     | lzo      | false   |       0 |
 true
salesmonth | eventid    | integer                     | mostly16 | false   |       0 |
 true
salesmonth | dateid     | smallint                    | lzo      | false   |       0 |
 true
salesmonth | qtysold    | smallint                    | mostly8  | false   |       0 |
 true
salesmonth | pricepaid  | numeric(8,2)                | delta32k | false   |       0 |
 false
salesmonth | commission | numeric(8,2)                | delta32k | false   |       0 |
 false
salesmonth | saletime   | timestamp without time zone | lzo      | false   |       0 |
 false
```

Betrachten Sie als Nächstes die Größe jeder Tabelle.

```
select count(*) from sales_monthly;
 count
-------
  2000
(1 row)

select count(*) from sales;
 count
-------
 412,214
(1 row)
```

Führen Sie nun den folgenden ALTER TABLE APPEND-Befehl aus.

```
alter table sales append from sales_monthly;
```

Betrachten Sie erneut die Größe jeder Tabelle. Die Spalte SALES_MONTHLY hat nun 0 Zeilen und die Spalte SALES ist um 2000 Zeilen größer geworden.

```
select count(*) from sales_monthly;
 count
-------
     0
(1 row)

select count(*) from sales;
 count
-------
 414214
(1 row)
```

Wenn die Quelltabelle mehr Spalten als die Zieltabelle enthält, geben Sie den Parameter IGNOREEXTRA an. Im folgenden Beispiel wird der Parameter IGNOREEXTRA verwendet, um beim Anfügen an die Tabelle SALES zusätzliche Spalten in der Tabelle SALES_LISTING zu ignorieren.

```
alter table sales append from sales_listing ignoreextra;
```

Wenn die Zieltabelle mehr Spalten als die Quelltabelle enthält, geben Sie den Parameter FILLTARGET an. Im folgenden Beispiel wird der Parameter FILLTARGET verwendet, um Spalten in der Tabelle SALES_REPORT auszufüllen, die in der Tabelle SALES_MONTH nicht vorhanden sind.

```
alter table sales_report append from sales_month filltarget;
```

Das folgende Beispiel zeigt ein Beispiel für die Verwendung von ALTER TABLE APPEND mit einer materialisierten Ansicht als Quelle.

```
ALTER TABLE target_tbl APPEND FROM my_streaming_materialized_view;
```

Die hier verwendeten Namen der Tabelle und der materialisierten Ansicht sind Beispiele. Das Anhängen aus einer materialisierten Ansicht funktioniert nur, wenn Ihre materialisierte Ansicht für Streaming-Erfassung konfiguriert ist. Es werden alle Datensätze in der materialisierten Quellansicht in eine Zieltabelle mit demselben Schema wie die materialisierte Ansicht verschoben und die

materialisierte Ansicht bleibt unverändert. Dies ist dasselbe Verhalten wie bei einer Tabelle als Datenquelle.

ALTER USER

Ändert einen Datenbankbenutzer.

Erforderliche Berechtigungen

Für ALTER USER sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung ALTER USER
- Aktueller Benutzer, der sein eigenes Passwort ändern möchte

Syntax

```
ALTER USER username [ WITH ] option [, ... ]

where option is

CREATEDB | NOCREATEDB
| CREATEUSER | NOCREATEUSER
| SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }
| PASSWORD { 'password' | 'md5hash' | DISABLE }
[ VALID UNTIL 'expiration_date' ]
| RENAME TO new_name |
| CONNECTION LIMIT { limit | UNLIMITED }
| SESSION TIMEOUT limit | RESET SESSION TIMEOUT
| SET parameter { TO | = } { value | DEFAULT }
| RESET parameter
| EXTERNALID external_id
```

Parameter

username (Benutzername

Name des Benutzers.

WITH

Optionales Schlüsselwort.

CREATEDB | NOCREATEDB

Mithilfe der Option CREATEDB kann der Benutzer neue Datenbanken erstellen. Der Standardwert ist NOCREATEDB.

CREATEUSER | NOCREATEUSER

Die Option CREATEUSER erstellt einen Superuser mit allen Datenbankrechten einschließlich CREATE USER. Der Standardwert ist NOCREATEUSER. Weitere Informationen finden Sie unter superuser.

SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }

Eine Klausel, über die die Zugriffsebene eines Benutzers auf die Amazon-Redshift-Systemtabellen und -Ansichten festgelegt wird.

Reguläre Benutzer mit der SYSLOG ACCESS RESTRICTED-Berechtigung können nur die Zeilen sehen, die von diesem Benutzer in für den Benutzer sichtbaren Systemtabellen und -ansichten generiert wurden. RESTRICTED ist der Standardwert.

Normale Benutzer mit der Berechtigung SYSLOG ACCESS UNRESTRICTED können alle Zeilen in für den Benutzer sichtbaren Systemtabellen und -ansichten sehen, einschließlich der Zeilen, die von einem anderen Benutzer generiert wurden. Über die Option UNRESTRICTED erhält ein Benutzer keinen Zugriff auf für Superuser sichtbare Tabellen. Nur Superuser können auf solche Tabellen zugreifen.

> **Note**
>
> Wenn Sie einem Benutzer uneingeschränkten Zugriff auf Systemtabellen gewähren, sieht der Benutzer auch Daten, die von anderen Benutzern generiert wurden. STL_QUERY und STL_QUERYTEXT enthalten beispielsweise den vollständigen Text von INSERT-, UPDATE- und DELETE-Anweisungen, die möglicherweise sensible von Benutzern generierte Daten enthalten.

Alle Zeilen in SVV_TRANSACTIONS sind für alle Benutzer sichtbar.

Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

PASSWORD { 'password' | 'md5hash' | DISABLE }

Legt das Benutzerpasswort fest.

Standardmäßig können Benutzer ihre eigenen Passwörter ändern, es sei denn, das Passwort ist deaktiviert. Legen Sie zum Deaktivieren des Passworts eines Benutzers DISABLE fest. Wenn das Passwort eines Benutzers deaktiviert ist, wird das Passwort aus dem System gelöscht und der Benutzer kann sich nur mit temporären AWS Identity and Access Management (IAM) Benutzeranmeldedaten anmelden. Weitere Informationen finden Sie unter Verwenden der IAM-Authentifizierung zur Erstellung von Benutzeranmeldeinformationen für die Datenbank. Nur ein Superuser kann Passwörter aktivieren oder deaktivieren. Sie können das Passwort eines Superusers nicht deaktivieren. Führen Sie zum Aktivieren eines Passworts ALTER USER aus und legen Sie ein Passwort fest.

Weitere Informationen zur Verwendung des PASSWORT-Parameters finden Sie unter CREATE USER.

VALID UNTIL 'expiration_date'

Gibt an, dass das Passwort ein Ablaufdatum hat. Verwenden Sie den Wert `'infinity'`, um ein Ablaufdatum zu vermeiden. Der gültige Datentyp für diesen Parameter ist timestamp.

Nur Superuser können diesen Parameter verwenden.

RENAME TO

Benennt den Benutzer um.

new_name

Der neue Name des Benutzers. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

> **Important**
>
> Wenn Sie einen Benutzer umbenennen, müssen Sie auch das Passwort des Benutzers ändern. Das neue Passwort muss sich nicht vom vorherigen Passwort unterscheiden. Der Benutzername wird als Teil der Passwortverschlüsselung verwendet. Daher wird das Passwort gelöscht, wen ein Benutzer umbenannt wird. Der Benutzer kann sich erst wieder anmelden, wenn das Passwort zurückgesetzt wurde. Beispiel:
>
> ```
> alter user newuser password 'EXAMPLENewPassword11';
> ```

CONNECTION LIMIT { Limit | UNLIMITED }

Die maximale Zahl von Datenbankverbindungen, die der Benutzer gleichzeitig geöffnet haben darf. Das Limit wird für Superuser nicht durchgesetzt. Mithilfe des Schlüsselworts UNLIMITED können Sie die maximale Zahl gleichzeitiger Verbindungen festlegen. Möglicherweise gilt auch ein Limit für die Zahl der Verbindungen für die einzelnen Datenbanken. Weitere Informationen finden Sie unter CREATE DATABASE. Der Standardwert ist UNLIMITED. Um die aktuellen Verbindungen anzuzeigen, führen Sie eine Abfrage für die Systemansicht STV_SESSIONS aus.

> **Note**
>
> Wenn sowohl für Benutzer- als auch für Datenbankverbindungen Limits gelten, muss ein ungenutzter Verbindungsplatz verfügbar sein, der sich innerhalb beider Grenzen befindet, wenn ein Benutzer versucht, eine Verbindung herzustellen.

SESSION TIMEOUT limit | RESET SESSION TIMEOUT

Die maximale Zeit in Sekunden, die eine Sitzung inaktiv oder untätig bleibt. Der Bereich liegt zwischen 60 Sekunden (einer Minute) und 1.728.000 Sekunden (20 Tagen). Wenn für den Benutzer kein Sitzungstimeout eingestellt ist, gilt die Clustereinstellung. Weitere Informationen finden Sie unter Kontingente und Limits in Amazon Redshift im Verwaltungshandbuch zu Amazon Redshift.

Wenn Sie das Sitzungstimeout festlegen, wird es nur auf neue Sitzungen angewendet.

Um Informationen über aktive Benutzersitzungen, einschließlich der Startzeit, des Benutzernamens und des Sitzungstimeouts anzuzeigen, fragen Sie die STV_SESSIONS-Systemansicht ab. Um Informationen über den Verlauf von Benutzersitzungen anzuzeigen, fragen Sie die STL_SESSIONS-Ansicht an. Um Informationen über Datenbankbenutzer, einschließlich Sitzungstimeouts, abzurufen, fragen Sie die SVL_USER_INFO-Ansicht ab.

SET

Legt für alle Sitzungen, die vom angegeben Benutzer ausgeführt werden, einen Konfigurationsparameter auf einen neuen Standardwert fest.

RESET

Setzt für den angegeben Benutzer einen Konfigurationsparameter auf den ursprünglichen Standardwert zurück.

Parameter

Der Name des Parameters, der festgelegt oder zurückgesetzt werden soll.

Wert

Neuer Wert des Parameters.

DEFAULT

Legt für alle Sitzungen, die vom angegeben Benutzer ausgeführt werden, den Konfigurationsparameter auf den Standardwert fest.

EXTERNALID external_id

Der Bezeichner für den Benutzer, der einem Identitätsanbieter zugeordnet ist. Der Benutzer muss sein Passwort deaktiviert haben. Weitere Informationen finden Sie unter Nativer Identitätsanbieter(IdP)-Verbund für Amazon Redshift.

Nutzungshinweise

- Versuch, rdsdb zu ändern – Sie können den Benutzer `rdsdb` nicht ändern.
- Ein unbekanntes Passwort erstellen — Wenn Sie die AWS Identity and Access Management (IAM-) Authentifizierung verwenden, um Datenbank-Benutzeranmeldeinformationen zu erstellen, möchten Sie möglicherweise einen Superuser erstellen, der sich nur mit temporären Anmeldeinformationen anmelden kann. Sie können das Passwort eines Superusers nicht deaktivieren, aber ein unbekanntes Passwort mithilfe einer willkürlich generierten MD5-Hash-Zeichenfolge erstellen.

```
alter user iam_superuser password 'md5123456789012345678012345678901 2';
```

- Festlegen von search_path – Wenn Sie den Parameter search_path mit dem Befehl ALTER USER festlegen, wirkt sich die Änderung bei der nächsten Anmeldung des angegebenen Benutzers aus. Wenn Sie den „search_path“-Wert für den aktuellen Benutzer und die aktuelle Sitzung ändern möchten, verwenden Sie den Befehl SET.
- Festlegen der Zeitzone – Wenn Sie SET TIMEZONE mit dem Befehl ALTER USER verwenden, wirkt sich die Änderung bei der nächsten Anmeldung des angegebenen Benutzers aus.
- Arbeiten mit dynamischer Datenmaskierung und Sicherheitsrichtlinien auf Zeilenebene: Wenn Ihr bereitgestellter Cluster oder Serverless-Namespace über dynamische Datenmaskierungs- oder

Sicherheitsrichtlinien auf Zeilenebene verfügt, werden die folgenden Befehle für normale Benutzer blockiert:

```
ALTER <current_user> SET enable_case_sensitive_super_attribute/
enable_case_sensitive_identifier/downcase_delimited_identifier
```

Nur Superuser und Benutzer mit der ALTER USER-Berechtigung können diese Konfigurationsoptionen festlegen. Weitere Informationen zur Sicherheit auf Zeilenebene finden Sie unter Sicherheit auf Zeilenebene. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

Beispiele

Im folgenden Beispiel erhält der Benutzer ADMIN das Recht, Datenbanken zu erstellen:

```
alter user admin createdb;
```

Im folgenden Beispiel werden das Passwort des Benutzers ADMIN auf `adminPass9` festgelegt und Ablaufdatum und Ablaufzeit für das Passwort festgelegt:

```
alter user admin password 'adminPass9'
valid until '2017-12-31 23:59';
```

Im folgenden Beispiel wird der Benutzer ADMIN in SYSADMIN umbenannt:

```
alter user admin rename to sysadmin;
```

Im folgenden Beispiel wird das Timeout für eine Leerlaufsitzung für einen Benutzer auf 300 Sekunden aktualisiert.

```
ALTER USER dbuser SESSION TIMEOUT 300;
```

Setzt das Timeout für Leerlaufsitzungen des Benutzers zurück. Wenn Sie es zurücksetzen, gilt die Clustereinstellung. Sie müssen Datenbank-Superuser sein, um diesen Befehl auszuführen. Weitere Informationen finden Sie unter Kontingente und Limits in Amazon Redshift im Verwaltungshandbuch zu Amazon Redshift.

```
ALTER USER dbuser RESET SESSION TIMEOUT;
```

Im folgenden Beispiel wird die externe ID für einen Benutzer namens bob aktualisiert. Der Namespace lautet myco_aad. Wenn der Namespace keinem registrierten Identitätsanbieter zugeordnet ist, führt dies zu einem Fehler.

```
ALTER USER myco_aad:bob EXTERNALID "ABC123" PASSWORD DISABLE;
```

Im folgenden Beispiel wird die Zeitzone für alle von einem bestimmten Datenbankbenutzer ausgeführten Sitzungen eingerichtet. Die Zeitzone wird für aufeinander folgende Sitzungen, nicht für die aktuelle Sitzung geändert.

```
ALTER USER odie SET TIMEZONE TO 'Europe/Zurich';
```

Im folgenden Beispiel wird die maximale Anzahl von Datenbankverbindungen festgelegt, die der Benutzer bob geöffnet haben darf.

```
ALTER USER bob CONNECTION LIMIT 10;
```

ANALYZE

Aktualisiert Tabellenstatistiken zur Verwendung durch den Abfrageplaner.

Erforderliche Berechtigungen

Für ANALYZE sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung ANALYZE
- Besitzer der Beziehung
- Datenbankbesitzer, für den die Tabelle freigegeben wird.

Syntax

```
ANALYZE [ VERBOSE ]
[ [ table_name [ ( column_name [, ...] ) ] ]
[ PREDICATE COLUMNS | ALL  COLUMNS ]
```

Parameter

VERBOSE

Eine Klausel, die Nachrichten zum Fortschritt in Bezug auf die Operation ANALYZE zurückgibt. Diese Option ist nützlich, wenn Sie keine Tabelle angeben.

table_name

Sie können spezifische Tabellen analysieren, einschließlich temporärer Tabellen. Sie können die Tabelle mittels ihres Schemanamens qualifizieren. Optional können Sie einen table_name angeben, um eine einzelne Tabelle zu analysieren. Sie können nicht mehr als einen table_name mit einer einzelnen Anweisung ANALYZE table_name angeben. Wenn Sie keinen table_name-Wert angeben, werden alle Tabellen in der aktuell verbundenen Datenbank analysiert, einschließlich der persistenten Tabellen im Systemkatalog. Amazon Redshift überspringt die Analyse einer Tabelle, wenn der Prozentsatz der Zeilen, die seit dem letzten ANALYZE-Vorgang geändert wurden, unter dem Analyseschwellenwert liegt. Weitere Informationen finden Sie unter Analyse-Schwellenwert.

Sie müssen Amazon-Redshift-Systemtabellen (STL- und STV-Tabellen) nicht analysieren.

column_name

Wenn Sie einen table_name angeben, können Sie auch eine oder mehrere Spalten in der Tabelle angeben (als durch Komma getrennte Liste in Klammern). Wenn eine Spaltenliste angegeben ist, werden nur die aufgelisteten Spalten analysiert.

PREDICATE COLUMNS | ALL COLUMNS

Klausel, die angibt, ob ANALYZE nur Prädikatspalten umfassen sollte. Geben Sie PREDICATE COLUMNS an, um nur Spalten zu analysieren, die in vorherigen Abfragen als Prädikate verwendet wurden oder wahrscheinlich als Prädikate verwendet werden. Geben Sie ALL COLUMNS an, um alle Spalten zu analysieren. Der Standardwert ist ALL COLUMNS.

Eine Spalte wird nur dann in den Satz der Prädikatspalten aufgenommen, wenn Folgendes zutrifft:

- Die Spalte wurde in einer Abfrage als Teil eines Filters, einer Join-Bedingung oder einer Gruppe durch Klausel verwendet.
- Die Spalte ist ein Verteilungsschlüssel.
- Die Spalte ist Teil eines Sortierschlüssels.

Wenn keine Spalten als Prädikatspalten markiert sind, beispielsweise weil für die Tabelle noch keine Abfrage ausgeführt wurde, werden alle Spalten analysiert, auch wenn PREDICATE

COLUMNS angegeben ist. Weitere Informationen zu Prädikatspalten finden Sie unter Analysieren von Tabellen.

Nutzungshinweise

Amazon Redshift führt automatisch ANALYZE für Tabellen aus, die Sie mit den folgenden Befehlen erstellen:

- CREATE TABLE AS
- CREATE TEMP TABLE AS
- SELECT INTO

Sie können keine externen Tabellen analysieren.

Sie müssen den Befehl ANALYZE nicht für diese Tabellen ausführen, wenn sie erstellt werden. Wenn Sie sie ändern, sollten Sie sie jedoch genauso wie andere Tabellen analysieren.

Analyse-Schwellenwert

Um die Verarbeitungszeit zu reduzieren und die allgemeine Systemleistung zu verbessern, überspringt Amazon Redshift ANALYZE für eine Tabelle, wenn der Prozentsatz der Zeilen, die seit der letzten Ausführung des Befehls ANALYZE geändert wurden, niedriger als der durch den Parameter analyze_threshold_percent angegebene Analyseschwellenwert ist. Der Standardwert für `analyze_threshold_percent` ist 10. Um `analyze_threshold_percent` für die aktuelle Sitzung zu ändern, führen Sie den Befehl SET aus. Im folgenden Beispiel wird `analyze_threshold_percent` in 20 Prozent geändert.

```
set analyze_threshold_percent to 20;
```

Um Tabellen zu analysieren, wenn nur eine kleine Zahl von Zeilen geändert wurde, legen Sie `analyze_threshold_percent` auf eine beliebig kleine Zahl fest. Wenn Sie beispielsweise `analyze_threshold_percent` auf 0,01 festlegen, wird eine Tabelle mit 100.000.000 Zeilen nicht übersprungen, wenn mindestens 10.000 Zeilen geändert wurden.

```
set analyze_threshold_percent to 0.01;
```

Wenn ANALYZE eine Tabelle überspringt, weil der Analyseschwellenwert nicht erreicht wird, gibt Amazon Redshift die folgende Meldung zurück.

```
ANALYZE SKIP
```

Um alle Tabellen zu analysieren, auch wenn keine Zeilen geändert wurden, legen Sie `analyze_threshold_percent` auf 0 fest.

Um die Ergebnisse von ANALYZE-Vorgängen anzuzeigen, führen Sie für die Systemtabelle STL_ANALYZE eine Abfrage aus.

Weitere Informationen zum Analysieren von Tabellen finden Sie unter Analysieren von Tabellen.

Beispiele

Analysiert alle Tabellen in der Datenbank TICKIT und gibt Fortschrittinformationen zurück.

```
analyze verbose;
```

Analysiert nur die Tabelle LISTING.

```
analyze listing;
```

Analysiert die Spalten VENUEID und VENUENAME in der Tabelle VENUE.

```
analyze venue(venueid, venuename);
```

Analysiert nur Prädikatspalten in der Tabelle VENUE.

```
analyze venue predicate columns;
```

ANALYZE COMPRESSION

Führt Kompressionsanalysen aus und erstellt einen Bericht mit der vorgeschlagenen Spaltenkodierung für die analysierten Tabellen. Für jede Spalte enthält der Bericht eine Schätzung der potenziellen Reduzierung des Festplattenspeichers im Vergleich zur RAW-Kodierung.

Syntax

```
ANALYZE COMPRESSION
[ [ table_name ]
[ ( column_name [, ...] ) ] ]
[COMPROWS numrows]
```

Parameter

table_name

Sie können die Kompression für spezifische Tabellen analysieren, einschließlich temporärer Tabellen. Sie können die Tabelle mittels ihres Schemanamens qualifizieren. Optional können Sie einen table_name angeben, um eine einzelne Tabelle zu analysieren. Wenn Sie keinen table_name angeben, werden alle Tabellen in der aktuell verbundenen Datenbank analysiert. Sie können nicht mehr als einen table_name mit einer einzelnen Anweisung ANALYZE COMPRESSION angeben.

column_name

Wenn Sie einen table_name angeben, können Sie auch eine oder mehrere Spalten in der Tabelle angeben (als durch Komma getrennte Liste in Klammern).

COMPROWS

Die Anzahl der Zeilen, die beispielhaft für die Kompressionanalyse verwendet werden sollen. Die Analyse wird für Zeilen aus jedem Daten-Slice ausgeführt. Wenn Sie beispielsweise COMPROWS 1000000 (1.000.000) angeben und das System insgesamt 4 Slices enthält, werden nicht mehr als 250.000 Zeilen pro Slice gelesen und analysiert. Wenn COMPROWS nicht angegeben wird, werden standardmäßig 100.000 Zeilen pro Slice beispielhaft analysiert. Werte für COMPROWS, die niedriger als der Standardwert von 100.000 Zeilen pro Slice sind, werden automatisch auf den Standardwert aktualisiert. Die Komprimierungsanalyse gibt jedoch keine Empfehlungen aus, wenn die Menge der Daten in einer Tabelle nicht als aussagekräftige Grundlage für die Analyse ausreicht. Wenn der Wert für COMPROWS größer als die Anzahl der Zeilen in der Tabelle ist, wird der Befehl ANALYZE COMPRESSION dennoch fortgesetzt und die Kompressionsanalyse wird anhand aller verfügbaren Zeilen ausgeführt.

numrows

Die Anzahl der Zeilen, die beispielhaft für die Kompressionanalyse verwendet werden sollen. Der akzeptierte Bereich für numrows ist eine Zahl zwischen 1000 und 1000000000 (1.000.000.000).

Nutzungshinweise

ANALYZE COMPRESSION bewirkt eine exklusive Tabellensperrung, die gleichzeitig ausgeführte Lese- und Schreibvorgänge für die Tabelle verhindert. Führen Sie den Befehl ANALYZE COMPRESSION nur aus, wenn die Tabelle nicht verwendet wird.

Führen Sie ANALYZE COMPRESSION aus, um basierend auf einer Beispielauswahl der Tabelleninhalte Empfehlungen zu Kodierungsschemata für Spalten zu erhalten. ANALYZE COMPRESSION ist ein Empfehlungstool und ändert die Spaltenkodierung der Tabelle nicht. Sie können die vorgeschlagene Kodierung anwenden, indem Sie die Tabelle neu erstellen oder eine neue Tabelle mit demselben Schema erstellen. Die Neuerstellung einer nicht komprimierten Tabelle mit geeigneten Kodierungsschemas kann den Platzbedarf auf der Festplatte erheblich reduzieren. Dieser Ansatz spart Speicherplatz und verbessert die Abfrageleistung für I/O-gebundene Workloads.

ANALYZE COMPRESSION überspringt die eigentliche Analysephase und gibt direkt den ursprünglichen Kodierungstyp für jede Spalte zurück, die als SORTKEY bezeichnet ist. Dies geschieht, weil bereichseingeschränkte Scans möglicherweise geringe Leistung zeigen, wenn SORTKEY-Spalten viel stärker komprimiert werden als andere Spalten.

Beispiele

Das folgende Beispiel zeigt die Codierung und die geschätzte prozentuale Reduzierung für die Spalten nur in der Tabelle LISTING:

```
analyze compression listing;

  Table  |     Column     | Encoding | Est_reduction_pct
---------+----------------+----------+-------------------
 listing | listid         | az64     | 40.96
 listing | sellerid       | az64     | 46.92
 listing | eventid        | az64     | 53.37
 listing | dateid         | raw      | 0.00
 listing | numtickets     | az64     | 65.66
 listing | priceperticket | az64     | 72.94
 listing | totalprice     | az64     | 68.05
 listing | listtime       | az64     | 49.74
```

Das folgende Beispiel analysiert die Spalten QTYSOLD, COMMISSION und SALETIME in der Tabelle SALES.

```
analyze compression sales(qtysold, commission, saletime);

 Table |   Column   | Encoding | Est_reduction_pct
-------+------------+----------+-------------------
 sales | salesid    | N/A      | 0.00
 sales | listid     | N/A      | 0.00
 sales | sellerid   | N/A      | 0.00
```

```
 sales | buyerid    | N/A      | 0.00
 sales | eventid    | N/A      | 0.00
 sales | dateid     | N/A      | 0.00
 sales | qtysold    | az64     | 83.06
 sales | pricepaid  | N/A      | 0.00
 sales | commission | az64     | 71.85
 sales | saletime   | az64     | 49.63
```

ANFÜGEN EINER MASKIERUNGSRICHTLINIE

Fügt eine vorhandene Richtlinie für die dynamische Datenmaskierung an eine Spalte an. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Maskierungsrichtlinie anfügen.

Syntax

```
ATTACH MASKING POLICY policy_name
   ON { relation_name }
   ( {output_columns_names | output_path} ) [ USING ( {input_column_names | input_path
 )} ]
   TO { user_name | ROLE role_name | PUBLIC }
   [ PRIORITY priority ];
```

Parameter

policy_name

Der Name der anzufügenden Maskierungsrichtlinie.

relation_name

Der Name der Relation, an die die Maskierungsrichtlinie angefügt werden soll.

output_column_names

Die Namen der Spalten, für die die Maskierungsrichtlinie gelten soll.

output_paths

Der vollständige Pfad des SUPER-Objekts, für das die Maskierungsrichtlinie gelten soll, einschließlich des Spaltennamens. Beispielsweise könnte für eine Relation mit einer Spalte vom Typ SUPER mit dem Namen `person` der output_path `person.name.first_name` sein.

input_column_names

Die Namen der Spalten, die die Maskierungsrichtlinie als Eingabe verwenden wird. Dieser Parameter ist optional. Wenn nicht angegeben, verwendet die Maskierungsrichtlinie output_column_names als Eingaben.

input_paths

Der vollständige Pfad des SUPER-Objekts, das die Maskierungsrichtlinie als Eingabe verwenden wird. Dieser Parameter ist optional. Wenn nicht angegeben, verwendet die Maskierungsrichtlinie output_path als Eingaben.

user_name

Der Name des Benutzers, dem die Maskierungsrichtlinie angefügt werden soll. Sie können nicht derselben Kombination aus Benutzer und Spalte oder Rolle und Spalte zwei Richtlinien zuordnen. Sie können eine Richtlinie einem Benutzer und eine weitere Richtlinie der Rolle des Benutzers anfügen. In diesem Fall gilt die Richtlinie mit der höheren Priorität.

In einem einzelnen Befehl ATTACH MASKING POLICY können Sie nur entweder user_name, role_name oder PUBLIC festlegen.

role_name

Der Name der Rolle, der die Maskierungsrichtlinie angefügt werden soll. Sie können nicht demselben Spalten-/Rollenpaar zwei Richtlinien anfügen. Sie können eine Richtlinie einem Benutzer und eine weitere Richtlinie der Rolle des Benutzers anfügen. In diesem Fall gilt die Richtlinie mit der höheren Priorität.

In einem einzelnen Befehl ATTACH MASKING POLICY können Sie nur entweder user_name, role_name oder PUBLIC festlegen.

PUBLIC

Fügt die Maskierungsrichtlinie allen Benutzern an, die auf die Tabelle zugreifen. Sie müssen anderen Maskierungsrichtlinien, die bestimmten Spalten-/Benutzer- oder Spalten-/Rollenpaaren angefügt sind, eine höhere Priorität als der PUBLIC-Richtlinie einräumen, damit sie angewendet werden.

In einem einzelnen Befehl ATTACH MASKING POLICY können Sie nur entweder user_name, role_name oder PUBLIC festlegen.

priority

Die Priorität der Maskierungsrichtlinie. Wenn mehrere Maskierungsrichtlinien für die Abfrage eines bestimmten Benutzers gelten, gilt die Richtlinie mit der höchsten Priorität.

Sie können derselben Spalte nicht zwei verschiedene Richtlinien mit derselben Priorität zuordnen, auch wenn die beiden Richtlinien unterschiedlichen Benutzern oder Rollen zugeordnet sind. Sie können dieselbe Richtlinie mehrmals demselben Satz von Tabellen-, Ausgabespalten-, Eingabespalten- und Prioritätsparametern zuordnen, sofern der Benutzer oder die Rolle, an die die Richtlinie angefügt wird, jedes Mal ein(e) andere(r) ist.

Sie können auf eine Spalte keine Richtlinie anwenden, die dieselbe Priorität wie eine andere dieser Spalte angefügte Richtlinie hat, auch wenn sie für unterschiedliche Rollen bestimmt ist. Dies ist ein optionales Feld. Wenn Sie keine Priorität angeben, wird die Maskierungsrichtlinie standardmäßig mit einer Priorität von 0 angefügt.

ATTACH RLS POLICY

Weisen Sie einem oder mehreren Benutzern oder Rollen eine RLS-Richtlinie für eine Tabelle zu.

Superuser und Benutzer oder Rollen, die die `sys:secadmin`-Rolle haben, können eine Richtlinie anfügen.

Syntax

```
ATTACH RLS POLICY policy_name ON [TABLE] table_name [, ...]
TO { user_name | ROLE role_name | PUBLIC } [, ...]
```

Parameter

policy_name

Der Name der -Richtlinie.

AUF [TABLE] table_name [, ...]

Die Relation, an die die Sicherheitsrichtlinie auf Zeilenebene angefügt ist.

AN { user_name | ROLE role_name | PUBLIC} [, ...]

Gibt an, ob die Richtlinie einem oder mehreren angegebenen Benutzern oder Rollen zugeordnet ist.

Nutzungshinweise

Beachten Sie bei der Arbeit mit der Anweisung ATTACH RLS POLICY Folgendes:

- Die angefügte Tabelle sollte alle Spalten enthalten, die in der WITH-Klausel der Anweisung zur Richtlinienerstellung aufgeführt sind.
- Amazon Redshift RLS unterstützt das Anfügen von RLS-Richtlinien an die folgenden Objekte nicht:
  - Katalogtabellen
  - Datenbankübergreifende Relationen
  - Externe Tabellen
  - Materialisierte Ansichten
  - Temporäre Tabellen
  - Suchtabellen
- Sie können keine RLS-Richtlinie an Superuser oder an Benutzer mit der `sys:secadmin`-Berechtigung anfügen.

Beispiele

Im folgenden Beispiel wird eine Richtlinie für eine Tabelle an eine Rolle angefügt.

```
ATTACH RLS POLICY policy_concerts ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;
```

BEGIN

Startet eine Transaktion. Synonym mit START TRANSACTION.

Eine Transaktion ist eine einzelne, logische Arbeitseinheit, unabhängig davon, ob sie aus einem einzigen oder aus mehreren Befehlen besteht. Im Allgemeinen werden alle Befehle in einer Transaktion für einen Snapshot der Datenbank ausgeführt, dessen Startzeit durch den Wert bestimmt wird, der für den Systemkonfigurationsparameter `transaction_snapshot_begin` festgelegt ist.

Standardmäßig werden einzelne Amazon-Redshift-Operationen (Abfragen, DDL-Anweisungen, Lasten) der Datenbank automatisch übergeben. Wenn Sie ein Commit für eine Operation aussetzen möchten, bis die anschließende Aufgabe abgeschlossen ist, müssen Sie eine Transaktion mit der BEGIN-Anweisung öffnen, die erforderlichen Befehle ausführen und dann die Transaktion mit der Anweisung COMMIT oder END schließen. Wenn notwendig, können Sie die Anweisung ROLLBACK

verwenden, um eine Transaktion abzubrechen, die gerade ausgeführt wird. Eine Ausnahme von diesem Verhalten stellt der Befehl TRUNCATE dar, der einen Commit für die Transaktion ausführt, in der er ausgeführt wird, und für den kein Rollback möglich ist.

Syntax

```
BEGIN [ WORK | TRANSACTION ] [ ISOLATION LEVEL option ] [ READ WRITE | READ ONLY ]

START TRANSACTION [ ISOLATION LEVEL option ] [ READ WRITE | READ ONLY ]

Where option is

SERIALIZABLE
| READ UNCOMMITTED
| READ COMMITTED
| REPEATABLE READ

Note: READ UNCOMMITTED, READ COMMITTED, and REPEATABLE READ have no
operational impact and map to SERIALIZABLE in Amazon Redshift. You can see database
 isolation levels on your cluster
by querying the stv_db_isolation_level table.
```

Parameter

WORK

Optionales Schlüsselwort.

TRANSACTION

Optionales Schlüsselwort; WORK und TRANSACTION sind Synonyme.

ISOLATION LEVEL SERIALIZABLE

Die serialisierbare Isolierung wird standardmäßig unterstützt. Daher bleibt das Verhalten der Transaktion stets gleich, unabhängig davon, ob diese Syntax in der Anweisung enthalten ist oder nicht. Weitere Informationen finden Sie unter Verwalten gleichzeitiger Schreiboperationen. Es werden keine anderen Isolierungsstufen unterstützt.

> **Note**
>
> Der SQL-Standard definiert vier Stufen der Transaktionsisolierung, um folgende Ereignisse zu verhindern: nicht korrekte Lesevorgänge (wenn eine Transaktion Daten

liest, die von einer gleichzeitigen Transaktion geschrieben werden, für die kein Commit ausgeführt wurde), nicht wiederholbare Lesevorgänge (wenn eine Transaktion Daten erneut liest, die sie zuvor bereits gelesen hat, und feststellt, dass die Daten durch eine andere Transaktion geändert wurden, für die seit dem ersten Lesevorgang ein Commit ausgeführt wurde) und Phantomlesevorgänge (wenn eine Transaktion eine Anfrage erneut ausführt, einen Satz von Zeilen zurückgibt, die eine Suchbedingung erfüllen, und dann feststellt, dass der Satz von Zeilen aufgrund einer anderen Transaktion, für die vor Kurzem ein Commit ausgeführt wurde, geändert wurde):

- Lesen von Daten, für die kein Commit ausgeführt wurde: Nicht korrekte Lesungen, nicht wiederholbare Lesungen und Phantomlesungen sind möglich.
- Lesen von Daten, für die ein Commit ausgeführt wurde: Nicht wiederholbare Lesungen und Phantomlesungen sind möglich.
- Wiederholbare Lesungen: Phantomlesungen sind möglich.
- Serialisierbar: Verhindert nicht korrekte Lesungen, nicht wiederholbare Lesungen und Phantomlesungen.

Obwohl Sie jede der vier Isolierungsstufen für Transaktionen verwenden können, verarbeitet Amazon Redshift alle Isolierungsstufen als serialisierbar.

READ WRITE

Erteilt der Transaktion Lese- und Schreibberechtigungen.

READ ONLY

Erteilt der Transaktion Leseberechtigungen.

Beispiele

Im folgenden Beispiel wird ein serialisierbarer Transaktionsblock gestartet:

```
begin;
```

Im folgenden Beispiel wird der Transaktionsblock auf der serialisierbaren Isolierungsstufe und mit Lese- und Schreibberechtigungen gestartet:

```
begin read write;
```

CALL

Führt eine gespeicherte Prozedur aus. Der CALL-Befehl muss den Namen der Prozedur und die Werte der Eingabeargumente enthalten. Zum Aufrufen einer gespeicherten Prozedur verwenden Sie die CALL-Anweisung.

> **Note**
>
> CALL kann nicht Teil einer regulären Abfrage sein.

Syntax

```
CALL sp_name ( [ argument ] [, ...] )
```

Parameter

sp_name

Der Name der Prozedur, die ausgeführt werden soll.

argument

Der Wert des Eingabeargumenten. Dieser Parameter kann auch ein Funktionsname sein, wie z. B. `pg_last_query_id()`. Abfragen können nicht als CALL-Argumente verwendet werden.

Nutzungshinweise

In Amazon Redshift gespeicherte Prozeduren unterstützen verschachtelte und rekursive Aufrufe wie im Nachfolgenden beschrieben. Stellen Sie außerdem sicher, dass Ihre Treiberunterstützung up-to-date, ebenfalls im Folgenden beschrieben, unterstützt wird.

Themen

- Verschachtelte Aufrufe
- Treiberunterstützung

Verschachtelte Aufrufe

In Amazon Redshift gespeicherte Prozeduren unterstützen verschachtelte und rekursive Aufrufe. Die zulässige Höchstanzahl an Verschachtelungsebenen beträgt 16. Verschachtelte Aufrufe können Geschäftslogik in kleineren Prozeduren verkapseln, die von mehreren Aufrufern geteilt werden können.

Wenn Sie eine verschachtelte Prozedur aufrufen, die über Ausgabeparameter verfügt, muss die innere Prozedur INOUT-Argumente definieren. In diesem Fall wird die innere Prozedur in einer nicht konstanten Variable übergeben. OUT-Argumente sind nicht zulässig. Dies ist darauf zurückzuführen, dass eine Variable zum Aufnehmen der Ausgabe des inneren Aufrufs erforderlich ist.

Die Beziehung zwischen inneren und äußeren Prozeduren wird in der Spalte `from_sp_call` von SVL_STORED_PROC_CALL protokolliert.

Das folgende Beispiel zeigt die Übergabe von Variablen an einen verschachtelten Prozeduraufruf über INOUT-Argumente.

```
CREATE OR REPLACE PROCEDURE inner_proc(INOUT a int, b int, INOUT c int) LANGUAGE
 plpgsql
AS $$
BEGIN
  a := b * a;
  c := b * c;
END;
$$;

CREATE OR REPLACE PROCEDURE outer_proc(multiplier int) LANGUAGE plpgsql
AS $$
DECLARE
  x int := 3;
  y int := 4;
BEGIN
  DROP TABLE IF EXISTS test_tbl;
  CREATE TEMP TABLE test_tbl(a int, b varchar(256));
  CALL inner_proc(x, multiplier, y);
  insert into test_tbl values (x, y::varchar);
END;
$$;

CALL outer_proc(5);
```

```
SELECT * from test_tbl;
 a  | b
----+----
 15 | 20
(1 row)
```

Treiberunterstützung

Wir empfehlen die Aktualisierung Ihrer Java Database Connectivity (JDBC)- und Open Database Connectivity (ODBC)-Treiber auf die neueste Version, die Unterstützung für in Amazon Redshift gespeicherte Prozeduren bietet.

Wenn Ihr Clienttool Treiber-API-Operationen verwendet, die die CALL-Anweisungen an den Server übergeben, können Sie unter Umständen Ihren vorhandenen Treiber verwenden. Ausgabeparameter werden (sofern vorhanden) als Ergebnissatz mit einer Zeile zurückgegeben.

Die neusten Versionen der Amazon Redshift JDBC- und ODBC-Treiber bieten Metadatenunterstützung für die Erkennung gespeicherter Prozeduren. Außerdem bieten sie `CallableStatement`-Unterstützung für individuelle Java-Anwendungen. Weitere Informationen zu Treibern finden Sie unter Herstellen von Verbindungen mit Amazon-Redshift-Clustern mithilfe von SQL-Client-Tools im Amazon-Redshift-Verwaltungshandbuch.

Die folgenden Beispiele zeigen, wie Sie unterschiedliche API-Operationen des JDBC-Treibers für Aufrufe gespeicherter Prozeduren verwenden können.

```
void statement_example(Connection conn) throws SQLException {
  statement.execute("CALL sp_statement_example(1)");
}

void prepared_statement_example(Connection conn) throws SQLException {
  String sql = "CALL sp_prepared_statement_example(42, 84)";
  PreparedStatement pstmt = conn.prepareStatement(sql);
  pstmt.execute();
}

void callable_statement_example(Connection conn) throws SQLException {
  CallableStatement cstmt = conn.prepareCall("CALL sp_create_out_in(?,?)");
  cstmt.registerOutParameter(1, java.sql.Types.INTEGER);
  cstmt.setInt(2, 42);
  cstmt.executeQuery();
  Integer out_value = cstmt.getInt(1);
```

```
}
```

Beispiele

Das folgende Beispiel ruft den Prozedurnamen auf `test_spl`.

```
call test_sp1(3,'book');
INFO:  Table "tmp_tbl" does not exist and will be skipped
INFO:  min_val = 3, f2 = book
```

Das folgende Beispiel ruft den Prozedurnamen auf `test_spl2`.

```
call test_sp2(2,'2019');

         f2          | column2
---------------------+---------
 2019+2019+2019+2019 | 2
(1 row)
```

CANCEL

Bricht eine Datenbankabfrage ab, die zurzeit ausgeführt wird.

Der Befehl CANCEL benötigt die Prozess- oder Sitzungs-ID der laufenden Abfrage und zeigt eine Bestätigungsmeldung an, um zu überprüfen, ob die Abfrage abgebrochen wurde.

Erforderliche Berechtigungen

Für CANCEL sind folgende Berechtigungen erforderlich:

- Superuser bricht seine eigene Abfrage ab
- Superuser bricht die Abfrage eines Benutzers ab
- Benutzer mit der Berechtigung CANCEL bricht die Abfrage eines Benutzers ab
- Benutzer storniert seine eigene Abfrage

Syntax

```
CANCEL process_id [ 'message' ]
```

Parameter

process_id

Um eine Abfrage abzubrechen, die in einem Amazon Redshift Redshift-Cluster ausgeführt wird, verwenden Sie die `pid` (Prozess-ID) vonSTV_RECENTS, die der Abfrage entspricht, die Sie stornieren möchten.

Um eine Abfrage abzubrechen, die in einer Amazon Redshift Serverless-Arbeitsgruppe ausgeführt wird, verwenden Sie das `session_id` Formular, SYS_QUERY_HISTORY das der Abfrage entspricht, die Sie stornieren möchten.

'message'

Eine optionale Bestätigungsmeldung, die angezeigt wird, wenn der Abbruch der Abfrage abgeschlossen ist. Wenn Sie keine Meldung angeben, zeigt Amazon Redshift die Standardmeldung als Verifizierung an. Sie müssen die Meldung in einfache Anführungszeichen setzen.

Nutzungshinweise

Sie können eine Abfrage nicht stornieren, indem Sie eine Abfrage-ID angeben. Sie müssen die Prozess-ID (PID) oder die Sitzungs-ID der Abfrage angeben. Sie können nur Aufträge abbrechen, die zurzeit von Ihrem Benutzer ausgeführt werden. Superuser können alle Abfragen abbrechen.

Wenn Abfragen in mehreren Sitzungen dieselbe Tabelle sperren, können Sie die Funktion PG_TERMINATE_BACKEND verwenden, um eine der Sitzungen zu beenden. Dabei werden alle Transaktionen, die zurzeit in der beendeten Sitzung ausgeführt werden, gezwungen, alle Sperren aufzuheben und für die Transaktionen ein Rollback auszuführen. Führen Sie eine Abfrage für die Systemtabelle STV_LOCKS aus, um die zurzeit vorhandenen Sperren anzuzeigen.

Im Anschluss an bestimmte interne Ereignisse startet Amazon Redshift möglicherweise eine aktive Sitzung neu und weist eine neue PID zu. Wenn die PID geändert wurde, erhalten Sie möglicherweise die folgende Fehlermeldung.

```
Session <PID> does not exist. The session PID might have changed. Check the
 stl_restarted_sessions system table for details.
```

Um die neue PID zu suchen, führen Sie eine Abfrage für die Systemtabelle STL_RESTARTED_SESSIONS aus und filtern nach der Tabelle `oldpid`.

```
select oldpid, newpid from stl_restarted_sessions where oldpid = 1234;
```

Beispiele

Um eine aktuell ausgeführte Abfrage in einem Amazon Redshift Redshift-Cluster abzubrechen, rufen Sie zunächst die Prozess-ID für die Abfrage ab, die Sie abbrechen möchten. Um die Prozess-IDs für alle zurzeit ausgeführten Abfragen zu ermitteln, geben Sie den folgenden Befehl ein:

```
select pid, starttime, duration,
trim(user_name) as user,
trim (query) as querytxt
from stv_recents
where status = 'Running';

pid |         starttime          | duration |   user   |    querytxt
-----+----------------------------+----------+----------+-----------------
802 | 2008-10-14 09:19:03.550885 |      132 | dwuser | select
venuename from venue where venuestate='FL', where venuecity not in
('Miami' , 'Orlando');
834 | 2008-10-14 08:33:49.473585 |  1250414 | dwuser | select *
from listing;
964 | 2008-10-14 08:30:43.290527 |   326179 | dwuser | select
sellerid from sales where qtysold in (8, 10);
```

Überprüfen Sie den Text der Abfrage, um festzustellen, welche Prozess-ID (POD) der Abfrage entspricht, die Sie abbrechen möchten.

Geben Sie den folgenden Befehl ein, um PID 802 zu verwenden und diese Abfrage abzubrechen:

```
cancel 802;
```

Die Sitzung, in der die Abfrage ausgeführt wurde, zeigt die folgende Meldung an:

```
ERROR:  Query (168) cancelled on user's request
```

168 ist die Abfrage-ID (nicht die Prozess-ID, die für den Abbruch der Abfrage verwendet wurde).

Alternativ können Sie eine benutzerdefinierte Bestätigungsmeldung angeben, die statt der Standardmeldung angezeigt wird. Um eine benutzerdefinierte Meldung anzugeben, setzen Sie die Meldung am Ende des Befehls CANCEL (ABBRECHEN) in Anführungszeichen:

```
cancel 802 'Long-running query';
```

Die Sitzung, in der die Abfrage ausgeführt wurde, zeigt die folgende Meldung an:

```
ERROR:  Long-running query
```

CLOSE

(Optional) Schließt alle freien Ressourcen, die mit einem offenen Cursor verbunden sind. COMMIT, END und ROLLBACK schließen den Cursor automatisch, so dass es nicht erforderlich ist, den CLOSE-Befehl zu verwenden, um den Cursor explizit zu schließen.

Weitere Informationen finden Sie unter DECLARE, FETCH.

Syntax

```
CLOSE cursor
```

Parameter

cursor

Der Name des Cursors, der geschlossen werden soll.

Beispiel für CLOSE

Mit den folgenden Befehlen werden der Cursor geschlossen und ein Commit ausgeführt, wodurch die Transaktion beendet wird:

```
close movie_cursor;
commit;
```

COMMENT

Erstellt oder ändert einen Kommentar zu einem Datenbankobjekt.

Syntax

```
COMMENT ON
```

```
{
TABLE object_name |
COLUMN object_name.column_name |
CONSTRAINT constraint_name ON table_name |
DATABASE object_name |
VIEW object_name
}
IS 'text' | NULL
```

Parameter

object_name

Der Name des Datenbankobjekts, das kommentiert wird. Sie können den folgenden Objekten Kommentare hinzufügen:

- TABLE
- COLUMN (akzeptiert auch column_name).
- CONSTRAINT (akzeptiert auch constraint_name und table_name).
- DATABASE
- VIEW
- SCHEMA

IS 'text' | NULL

Der Kommentartext, den Sie für das angegebene Objekt hinzufügen oder ersetzen möchten. Die Text-Zeichenfolge hat den Datentyp TEXT. Sie müssen den Kommentar in einfache Anführungszeichen einschließen. Setzen Sie den Wert auf NULL, um den Kommentartext zu entfernen.

column_name

Der Name der Spalte, die kommentiert wird. Parameter von COLUMN. Folgt einer Tabelle, die in angegeben ist `object_name`.

constraint_name

Der Name der Einschränkung, die kommentiert wird. Parameter von CONSTRAINT.

table_name

Der Name einer Tabelle, die die Einschränkung enthält. Parameter von CONSTRAINT.

Nutzungshinweise

Sie müssen der Besitzer eines Datenbankobjekts sein, um einen Kommentar hinzufügen oder aktualisieren zu können.

Kommentare zu Datenbanken können nur auf die aktuelle Datenbank angewendet werden. Es wird eine Warnmeldung angezeigt, wenn Sie versuchen, eine andere Datenbank zu kommentieren. Dieselbe Warnung wird für Kommentare zu Datenbanken angezeigt, die nicht vorhanden sind.

Kommentare zu externen Tabellen, externen Spalten und Spalten von Late Binding-Ansichten werden nicht unterstützt.

Beispiele

Im folgenden Beispiel wird der SALES-Tabelle ein neuer Kommentar hinzugefügt.

```
COMMENT ON TABLE sales IS 'This table stores tickets sales data';
```

Im folgenden Beispiel wird der Kommentar zur SALES-Tabelle angezeigt.

```
select obj_description('public.sales'::regclass);

obj_description
------------------------------------
This table stores tickets sales data
```

Im folgenden Beispiel wird aus der SALES-Tabelle ein Kommentar entfernt.

```
COMMENT ON TABLE sales IS NULL;
```

Im folgenden Beispiel wird der EVENTID-Spalte der SALES-Tabelle ein neuer Kommentar hinzugefügt.

```
COMMENT ON COLUMN sales.eventid IS 'Foreign-key reference to the EVENT table.';
```

Im folgenden Beispiel wird der Kommentar zur EVENTID-Spalte (Spalte 5) der SALES-Tabelle angezeigt.

```
select col_description( 'public.sales'::regclass, 5::integer );
```

```
col_description
----------------------------------------
Foreign-key reference to the EVENT table.
```

Im folgenden Beispiel wird der Tabelle EVENT ein beschreibender Kommentar hinzugefügt.

```
comment on table event is 'Contains listings of individual events.';
```

Zum Anzeigen von Kommentaren führen Sie eine Abfrage für den PG_DESCRIPTION-Systemkatalog aus. Das folgende Beispiel gibt die Beschreibung für die EVENT-Tabelle zurück.

```
select * from pg_catalog.pg_description
where objoid =
(select oid from pg_class where relname = 'event'
and relnamespace =
(select oid from pg_catalog.pg_namespace where nspname = 'public') );

objoid | classoid | objsubid | description
-------+----------+----------+----------------------------------------
116658 |     1259 |        0 | Contains listings of individual events.
```

COMMIT

Führt einen Commit der aktuellen Transaktion an die Datenbank aus. Durch diesen Befehl werden die Datenbankupdates durch die Transaktion permanent.

Syntax

```
COMMIT [ WORK | TRANSACTION ]
```

Parameter

WORK

Optionales Schlüsselwort. Dieses Schlüsselwort wird in einer gespeicherten Prozedur nicht unterstützt.

TRANSACTION

Optionales Schlüsselwort. WORK und TRANSACTION sind Synonyme. Beide werden in einer gespeicherten Prozedur nicht unterstützt.

Informationen zur Verwendung von COMMIT innerhalb einer gespeicherten Prozedur finden Sie unter Verwalten von Transaktionen.

Beispiele

In jedem der folgenden Beispiele wird ein Commit der aktuellen Transaktion an die Datenbank ausgeführt:

```
commit;
```

```
commit work;
```

```
commit transaction;
```

COPY

Lädt Daten aus Datendateien oder aus einer Amazon-DynamoDB-Tabelle in eine Tabelle. Die Dateien können sich in einem Amazon-Simple-Storage-Service(Amazon S3)-Bucket, einem Amazon-EMR-Cluster oder auf einem Remote-Host befinden, auf den über eine Secure-Shell(SSH)-Verbindung zugegriffen wird.

> **Note**
>
> Externe Tabellen von Amazon Redshift Spectrum sind schreibgeschützt. Sie können keinen COPY-Vorgang zu einer externen Tabelle ausführen.

Der COPY-Befehl fügt die Eingabedaten als zusätzliche Zeilen an die Tabelle an.

Die maximale Größe einer einzelnen Eingabezeile aus einer beliebigen Quelle beträgt 4 MB.

Themen

- Erforderliche Berechtigungen
- COPY-Syntax
- Erforderliche Parameter
- Optionale Parameter
- Nutzungshinweise und zusätzliche Ressourcen für den Befehl COPY

- Befehlsbeispiele
- COPY JOB (Vorschau)
- COPY-Parameterreferenz
- Nutzungshinweise
- Beispiele für COPY

Erforderliche Berechtigungen

Um den COPY-Befehl verwenden zu können, benötigen Sie das INSERT-Recht für die Amazon-Redshift-Tabelle.

COPY-Syntax

```
COPY table-name
[ column-list ]
FROM data_source
authorization
[ [ FORMAT ] [ AS ] data_format ]
[ parameter [ argument ] [, ... ] ]
```

Sie können COPY-Operationen mit nur drei Parametern ausführen: Tabellenname, Datenquelle und Autorisierung für den Zugriff auf die Daten.

Amazon Redshift erweitert die Funktionalität des COPY-Befehls, damit Sie Daten in verschiedenen Datenformaten aus mehreren Datenquellen laden, den Zugriff auf das Laden von Daten steuern, Datentransformierungen verwalten und die Ladeoperation verwalten können.

In den folgenden Abschnitten werden die erforderlichen Parameter für den COPY-Befehl vorgestellt und die optionalen Parameter nach Funktion gruppiert. Dazu werden die einzelnen Parameter beschrieben und es wird erläutert, wie die verschiedenen Optionen zusammenwirken. Sie können auch anhand der alphabetischen Liste der Parameter direkt zur Parameterbeschreibung wechseln.

Erforderliche Parameter

Der COPY-Befehl erfordert drei Elemente:

- Table Name
- Data Source
- Authorization

Der einfachste COPY-Befehl verwendet das folgende Format.

```
COPY table-name
FROM data-source
authorization;
```

Im folgenden Beispiel wird eine Tabelle namens CATDEMO erstellt. Anschließend wird die Tabelle mit Stichprobendaten aus einer Datendatei in Amazon S3 namens geladen `category_pipe.txt`.

```
create table catdemo(catid smallint, catgroup varchar(10), catname varchar(10), catdesc
 varchar(50));
```

Im folgenden Beispiel ist die Datenquelle für den COPY-Befehl eine Datendatei namens `category_pipe.txt` im Ordner `tickit` eines Amazon-S3-Buckets namens `redshift-downloads`. Der COPY-Befehl ist autorisiert, über eine AWS Identity and Access Management (IAM-) Rolle auf den Amazon S3 S3-Bucket zuzugreifen. Wenn Ihr Cluster bereits eine IAM-Rolle mit Berechtigung für den Zugriff auf Amazon S3 besitzt, können Sie den Amazon-Ressourcennamen (ARN) Ihrer Rolle im folgenden COPY-Befehl einsetzen und diesen ausführen.

```
copy catdemo
from 's3://redshift-downloads/tickit/category_pipe.txt'
iam_role 'arn:aws:iam::<aws-account-id>:role/<role-name>'
region 'us-east-1';
```

Vollständige Anweisungen zur Verwendung von COPY-Befehlen zum Laden von Beispieldaten, einschließlich Anweisungen zum Laden von Daten aus anderen AWS Regionen, finden Sie unter Load Sample Data from Amazon S3 im Amazon Redshift Getting Started Guide.

table-name

Der Name der Zieltabelle für den COPY-Befehl. Die Tabelle muss in der Datenbank bereits vorhanden sein. Die Tabelle kann temporär oder persistent sein. Der COPY-Befehl fügt die neuen Eingabedaten den vorhandenen Zeilen in der Tabelle an.

FROM data-source

Der Speicherort der Quelldaten, die in die Zieltabelle geladen werden sollen. Bei manchen Datenquellen kann eine Manifestdatei angegeben werden.

Das am häufigsten verwendete Daten-Repository sind Amazon-S3-Buckets. Außerdem können Sie Daten aus Datendateien laden, die sich in einem Amazon-EMR-Cluster, einer Amazon-EC2-

Instance oder auf einem Remote-Host befinden, auf den Ihr Cluster über eine SSH-Verbindung zugreifen kann. Sie können Daten auch direkt aus einer DynamoDB-Tabelle laden.

- COPY aus Amazon S3
- COPY aus Amazon EMR
- COPY von Remote-Hosts (SSH)
- COPY aus Amazon DynamoDB

Autorisierung

Eine Klausel, die die Methode angibt, die Ihr Cluster für die Authentifizierung und Autorisierung für den Zugriff auf andere AWS Ressourcen verwendet. Der COPY-Befehl benötigt eine Autorisierung, um auf Daten in einer anderen AWS Ressource zuzugreifen, einschließlich in Amazon S3, Amazon EMR, Amazon DynamoDB und Amazon EC2. Sie können diese Autorisierung bereitstellen, indem Sie eine IAM-Rolle referenzieren, die Ihrem Cluster angefügt ist, oder indem Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel für einen IAM-Benutzer bereitstellen.

- Autorisierungsparameter
- Rollenbasierte Zugriffskontrolle
- Schlüsselbasierte Zugriffssteuerung

Optionale Parameter

Sie können optional angeben, wie COPY den Spalten in der Zieltabelle Felddaten zuweist, Quelldatenattribute definieren, damit der COPY-Befehl die Quelldaten korrekt lesen und analysieren kann, und festlegen, welche Operation der COPY-Befehl während des Ladevorgangs ausführt.

- Optionen für das Mapping von Spalten
- Datenformatparameter
- Datenkonvertierungsparameter
- Datenladeoperationen

Mapping von Spalten

Standardmäßig fügt COPY Feldwerte in derselben Reihenfolge in die Spalten der Zieltabelle ein, die die Felder in den Datendateien haben. Wenn die standardmäßige Spaltenreihenfolge nicht

funktionieren wird, können Sie eine Spaltenliste angeben oder JSONPath-Ausdrücke verwenden, um den Zielspalten Quelldatenfelder zuzuweisen.

- Column List
- JSONPaths File

Datenformatparameter

Sie können Daten aus Textdateien in einem Format mit fester Breite, in einem Format mit Trennzeichen, in einem durch Komma getrennten Format (CSV) oder im JSON-Format laden. Sie können Daten auch aus Avro-Dateien laden.

Standardmäßig geht der COPY-Befehl davon aus, dass sich die Quelldaten in UTF-8-Textdateien mit Trennzeichen befinden. Das Standardtrennzeichen ist der senkrechte Strich (|). Wenn sich die Quelldaten in einem anderen Format befinden, verwenden Sie die folgenden Parameter, um das Datenformat anzugeben.

- FORMAT
- CSV
- DELIMITER
- FIXEDWIDTH
- SHAPEFILE
- AVRO
- JSON
- ENCRYPTED
- BZIP2
- GZIP
- LZOP
- PARQUET
- ORC
- ZSTD

Datenkonvertierungsparameter

Wenn COPY die Tabelle lädt, versucht der Befehl implizit, die Zeichenfolgen in den Quelldaten in den Datentyp der Zielspalte zu konvertieren. Wenn Sie eine Konvertierung angeben müssen, die sich vom Standardverhalten unterscheidet, oder wenn die Standardkonvertierung zu Fehlern führt, können Sie Datenkonvertierungen verwalten, indem Sie die folgenden Parameter angeben.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- BLANKSASNULL
- DATEFORMAT
- EMPTYASNULL
- ENCODING
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL AS
- REMOVEQUOTES
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

Datenladeoperationen

Verwalten Sie das Standardverhalten der Ladeoperation, um Fehler zu beheben oder die Ladezeiten zu reduzieren, indem Sie die folgenden Parameter angeben.

- COMPROWS
- COMPUPDATE
- IGNOREALLERRORS

- MAXERROR
- NOLOAD
- STATUPDATE

Nutzungshinweise und zusätzliche Ressourcen für den Befehl COPY

Weitere Informationen zur Verwendung des COPY-Befehls finden Sie in den folgenden Themen:

- Nutzungshinweise
- Tutorial: So laden Sie Daten aus Amazon S3
- Bewährte Methoden für Amazon Redshift zum Laden von Daten
- Verwenden eines COPY-Befehls zum Laden von Daten
  - So laden Sie Daten aus Amazon S3
  - So laden Sie Daten aus Amazon EMR:
  - Laden von Daten aus Remote-Hosts
  - Laden von Daten aus einer Amazon-DynamoDB-Tabelle
- Fehlerbehebung bei Datenladevorgängen

Befehlsbeispiele

Weitere Beispiele, die zeigen, wie aus verschiedenen Quellen, in unterschiedlichen Formaten und mit unterschiedlichen COPY-Optionen KOPIERT wird, finden Sie unter Beispiele für COPY.

COPY JOB (Vorschau)

Dies ist eine Vorabveröffentlichungsdokumentation für Autocopy (SQL COPY JOB), die sich in der Vorabversion befindet. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur in Test- und nicht in Produktionsumgebungen zu verwenden. Die öffentliche Vorversion endet am 31. Juli 2024. Vorschau-Cluster werden zwei Wochen nach dem Ende der Vorschauversion automatisch entfernt. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Weitere Informationen zur Verwendung dieses Befehls in der Vorschau finden Sie unter Kontinuierliche Dateierfassung von Amazon S3 (Vorschau).

Verwaltet COPY-Befehle, die Daten in eine Tabelle laden. Der Befehl COPY JOB ist eine Erweiterung des COPY-Befehls, der das Laden von Daten aus Amazon-S3-Buckets automatisiert. Wenn Sie einen COPY JOB erstellen, erkennt es Amazon Redshift, wenn neue Amazon-S3-Dateien in einem bestimmten Pfad erstellt werden, und lädt diese dann automatisch, ohne dass Sie eine Maßnahme ergreifen müssen. Beim Laden der Daten werden dieselben Parameter wie im ursprünglichen COPY-Befehl verwendet. Amazon Redshift verfolgt die geladenen Dateien, um sicherzustellen, dass sie nur einmal geladen werden.

> **Note**
>
> Informationen zum COPY-Befehl, seiner Syntax, seinen Parametern und Berechtigungen, finden Sie unter COPY.

Erforderliche Berechtigung

Um den COPY-Befehl von COPY JOB auszuführen, müssen Sie über die Berechtigung INSERT für die zu ladende Tabelle verfügen.

Die mit dem COPY-Befehl angegebene IAM-Rolle muss über die Berechtigung zum Zugriff auf die zu ladenden Daten verfügen. Weitere Informationen finden Sie unter IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY.

Syntax

Erstellen eines Kopierauftrags. Die Parameter des COPY-Befehls werden zusammen mit dem Kopierauftrag gespeichert.

```
COPY copy-command JOB CREATE job-name
[AUTO ON | OFF]
```

Ändern der Konfiguration eines Kopierauftrags.

```
COPY JOB ALTER job-name
[AUTO ON | OFF]
```

Ausführen eines Kopierauftrags. Es werden die gespeicherten Parameter des COPY-Befehls verwendet.

```
COPY JOB RUN job-name
```

Auflisten aller Kopieraufträge.

```
COPY JOB LIST
```

Anzeigen der Details zu dem Kopierauftrag.

```
COPY JOB SHOW job-name
```

Löschen eines Kopierauftrags.

```
COPY JOB DROP job-name
```

Parameter

copy-command

Ein COPY-Befehl, der Daten aus Amazon S3 in Amazon Redshift lädt. Die Klausel enthält COPY-Parameter, die den Amazon-S3-Bucket, die Zieltabelle, die IAM-Rolle und andere Parameter definieren, die beim Laden von Daten verwendet werden. Es werden alle Parameter eines COPY-Befehls zum Laden von Amazon-S3-Daten unterstützt, mit folgenden Ausnahmen:

- Der COPY JOB nimmt keine bereits vorhandenen Dateien in dem Ordner auf, auf den der COPY-Befehl verweist. Nur Dateien, die nach dem Erstellungszeitstempel von COPY JOB erstellt wurden, werden aufgenommen.
- Sie können einen COPY-Befehl nicht mit den Optionen MAXERROR oder IGNOREALLERRORS angeben.
- Sie können keine Manifestdatei angeben. COPY JOB erfordert einen festgelegten Amazon-S3-Speicherort, um diesen auf neu erstellte Dateien überwachen zu können.
- Sie können einen COPY-Befehl nicht mit Autorisierungstypen wie Zugriffsschlüsseln und geheimen Schlüsseln angeben. Es werden nur COPY-Befehle unterstützt, die den Parameter `IAM_ROLE` für die Autorisierung verwenden. Weitere Informationen finden Sie unter Autorisierungsparameter.
- Die dem Cluster zugeordnete standardmäßige IAM-Rolle wird von COPY JOB nicht unterstützt. Sie müssen die `IAM_ROLE` im COPY-Befehl angeben.

Weitere Informationen finden Sie unter COPY aus Amazon S3.

job-name

Der Name des Auftrags, der verwendet wird, um auf den COPY-Auftrag zu verweisen.

[AUTO ON / OFF]

Klausel, die angibt, ob Amazon-S3-Daten automatisch in Amazon-Redshift-Tabellen geladen werden.

- Bei Angabe von `ON` überwacht Amazon Redshift den Amazon-S3-Quellpfad auf neu erstellte Dateien. Falls welche gefunden werden, wird ein COPY-Befehl mit den COPY-Parametern aus der Auftragsdefinition ausgeführt. Dies ist die Standardeinstellung.
- Bei Angabe von `OFF` führt Amazon Redshift den COPY JOB nicht automatisch aus.

Nutzungshinweise

Die Optionen des COPY-Befehls werden erst zur Laufzeit validiert. Eine ungültige `IAM_ROLE` oder eine Amazon-S3-Datenquelle führt beispielsweise zu Laufzeitfehlern, wenn COPY JOB gestartet wird.

Wenn der Cluster angehalten ist, werden COPY JOBS nicht ausgeführt.

Informationen zur Abfrage von geladenen COPY-Befehlsdateien sowie zu Ladefehlern finden Sie unter STL_LOAD_COMMITS, STL_LOAD_ERRORS, STL_LOADERROR_DETAIL. Weitere Informationen finden Sie unter Überprüfung, ob die Daten korrekt geladen wurden.

Beispiele

Im folgenden Beispiel wird demonstriert, wie Sie einen COPY JOB erstellen, um Daten aus einem Amazon-S3-Bucket zu laden.

```
COPY public.target_table
FROM 's3://mybucket-bucket/staging-folder'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
JOB CREATE my_copy_job_name
AUTO ON;
```

COPY-Parameterreferenz

COPY hat viele Parameter, die in vielen Situationen verwendet werden können. Es werden jedoch nicht alle Parameter in jeder Situation unterstützt. Beispielsweise gibt es eine begrenzte Anzahl

unterstützter Parameter, um aus ORC- oder PARQUET-Dateien zu laden. Weitere Informationen finden Sie unter COPY aus spaltenbasierten Datenformaten.

Themen

- Datenquellen
- Autorisierungsparameter
- Optionen für das Mapping von Spalten
- Datenformatparameter
- Dateikomprimierungsparameter
- Datenkonvertierungsparameter
- Datenladeoperationen
- Alphabetische Liste der Parameter

Datenquellen

Sie können Daten aus Textdateien in einem Amazon-S3-Bucket, in einem Amazon-EMR-Cluster oder auf einem Remote-Host laden, auf den Ihr Cluster über eine SSH-Verbindung zugreifen kann. Sie können Daten auch direkt aus einer DynamoDB-Tabelle laden.

Die maximale Größe einer einzelnen Eingabezeile aus einer beliebigen Quelle beträgt 4 MB.

Um Daten aus einer Tabelle in einen Satz von Dateien in einem Amazon-S3-Bucket zu exportieren, verwenden Sie den Befehl UNLOAD.

Themen

- COPY aus Amazon S3
- COPY aus Amazon EMR
- COPY von Remote-Hosts (SSH)
- COPY aus Amazon DynamoDB

COPY aus Amazon S3

Um Daten aus Dateien zu laden, die sich in einzelnen oder mehreren S3 Buckets befinden, verwenden Sie die FROM-Klausel. Diese Klausel gibt an, wie COPY die Dateien in Amazon S3 sucht. Sie können den Objektpfad zu den Datendateien als Teil der FROM-Klausel bereitstellen oder den

Speicherort einer Manifestdatei angeben, die eine Liste von Amazon-S3-Objektpfaden enthält. COPY aus Amazon S3 verwendet eine HTTPS-Verbindung. Stellen Sie sicher, dass die S3-IP-Bereiche zu Ihrer Zulassungsliste hinzugefügt werden. Weitere Informationen zu den erforderlichen S3-IP-Bereichen finden Sie unter Netzwerkisolierung.

> **Important**
>
> Wenn sich die Amazon S3 S3-Buckets, die die Datendateien enthalten, nicht in derselben AWS Region wie Ihr Cluster befinden, müssen Sie den REGION Parameter verwenden, um die Region anzugeben, in der sich die Daten befinden.

Themen

- Syntax
- Beispiele
- Optionale Parameter
- Nicht unterstützte Parameter

Syntax

```
FROM { 's3://objectpath' | 's3://manifest_file' }
authorization
| MANIFEST
| ENCRYPTED
| REGION [AS] 'aws-region'
| optional-parameters
```

Beispiele

Im folgenden Beispiel wird ein Objektpfad verwendet, um Daten aus Amazon S3 zu laden.

```
copy customer
from 's3://mybucket/customer'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Im folgenden Beispiel wird eine Manifestdatei verwendet, um Daten aus Amazon S3 zu laden.

```
copy customer
```

```
from 's3://mybucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Parameter

FROM

Die Quelle der Daten, die geladen werden sollen. Weitere Informationen zur Kodierung der Amazon-S3-Datei finden Sie unter Datenkonvertierungsparameter.

's3://copy_from_s3_objectpath'

Gibt den Pfad zu den Amazon-S3-Objekten an, die die Daten enthalten, z. B. `'s3://mybucket/custdata.txt'`. Der Parameter s3://copy_from_s3_objectpath kann eine einzelne Datei, einen Satz von Objekten oder Ordner mit denselben Schlüsselpräfixen referenzieren. Beispielsweise ist der Name `custdata.txt` ein Schlüsselpräfix, der sich auf eine Anzahl physischer Dateien bezieht: `custdata.txt`, `custdata.txt.1`, `custdata.txt.2`, `custdata.txt.bak` usw. Das Schlüsselpräfix kann auch eine Anzahl von Ordnern referenzieren. Beispielsweise bezieht sich `'s3://mybucket/custfolder'` auf die Ordner `custfolder`, `custfolder_1`, `custfolder_2` usw. Wenn ein Schlüsselpräfix mehrere Ordner referenziert, werden alle Dateien in den Ordnern geladen. Wenn ein Schlüsselpräfix mit einer Datei und einem Ordner übereinstimmt, beispielsweise `custfolder.log`, versucht COPY, auch die Datei zu laden. Wenn ein Schlüsselpräfix dazu führen könnte, dass COPY versucht, nicht erwünschte Dateien zu laden, verwenden Sie eine Manifestdatei. Weitere Informationen finden Sie unter copy_from_s3_manifest_file.

> **Important**
>
> Wenn sich der S3-Bucket, der die Datendateien enthält, nicht in derselben AWS Region wie Ihr Cluster befindet, müssen Sie den REGION Parameter verwenden, um die Region anzugeben, in der sich die Daten befinden.

Weitere Informationen finden Sie unter So laden Sie Daten aus Amazon S3.

's3://copy_from_s3_manifest_file'

Gibt den Amazon-S3-Objektschlüssel für eine Manifestdatei an, die die Datendateien auflistet, die geladen werden sollen. Das Argument 's3://copy_from_s3_manifest_file' muss explizit

auf eine einzelne Datei verweisen, z. B. `'s3://mybucket/manifest.txt'`. Es darf kein Schlüsselpräfix referenzieren.

Das Manifest ist eine Textdatei im JSON-Format, die die URL jeder Datei auflistet, die aus Amazon S3 geladen werden soll. Die URL enthält den Bucket-Namen und den vollständigen Objektpfad für die Datei. Die im Manifest angegebenen Dateien können sich in verschiedenen Buckets befinden, aber alle Buckets müssen sich in derselben AWS Region wie der Amazon Redshift Redshift-Cluster befinden. Wenn eine Datei zweimal aufgelistet wird, wird die Datei zweimal geladen. Im folgenden Beispiel wird der JSON-Code für ein Manifest gezeigt, das drei Dateien lädt.

```
{
  "entries": [
    {"url":"s3://mybucket-alpha/custdata.1","mandatory":true},
    {"url":"s3://mybucket-alpha/custdata.2","mandatory":true},
    {"url":"s3://mybucket-beta/custdata.1","mandatory":false}
  ]
}
```

Die doppelten Anführungszeichen sind erforderlich. Es muss sich um normale Anführungszeichen (0x22) handeln. Es dürfen keine schrägen oder „smarten“ Anführungszeichen sein. Jeder Eintrag im Manifest kann optional ein `mandatory`-Flag enthalten. Wenn `mandatory` auf `true` gesetzt ist, wird COPY beendet, wenn der Befehl die Datei für diesen Eintrag nicht findet. Andernfalls wird COPY fortgesetzt. Der Standardwert für den `mandatory` beträgt `false`.

Beim Laden aus Datendateien im ORC- oder Parquet-Format ist ein `meta`-Feld erforderlich, wie im folgenden Beispiel gezeigt.

```
{
   "entries":[
      {
         "url":"s3://mybucket-alpha/orc/2013-10-04-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      },
      {
         "url":"s3://mybucket-beta/orc/2013-10-05-custdata",
         "mandatory":true,
         "meta":{
```

```
            "content_length":99
         }
      }
   ]
}
```

Die Manifestdatei darf nicht verschlüsselt oder komprimiert sein, auch dann nicht, wenn die Optionen ENCRYPTED, GZIP, LZOP, BZIP2 oder ZSTD angegeben sind. COPY gibt einen Fehler zurück, wenn die angegebene Manifestdatei nicht gefunden wird oder die Manifestdatei nicht korrekt formatiert ist.

Wenn ein Manifestdatei verwendet wird, muss der Parameter MANIFEST für den COPY-Befehl angegeben werden. Wenn der Parameter MANIFEST nicht angegeben ist, nimmt COPY an, dass die für FROM angegebene Datei eine Datendatei ist.

Weitere Informationen finden Sie unter So laden Sie Daten aus Amazon S3.

Autorisierung

Der COPY-Befehl benötigt eine Autorisierung für den Zugriff auf Daten in anderen AWS -Ressourcen, einschließlich Amazon S3, Amazon EMR, Amazon DynamoDB und Amazon EC2. Sie können diese Autorisierung bereitstellen, indem Sie auf eine AWS Identity and Access Management (IAM-) Rolle verweisen, die Ihrem Cluster zugeordnet ist (rollenbasierte Zugriffskontrolle), oder indem Sie die Zugangsdaten für einen Benutzer angeben (schlüsselbasierte Zugriffskontrolle). Um Sicherheit und Flexibilität zu verbessern, wird die Verwendung der IAM-rollenbasierten Zugriffssteuerung empfohlen. Weitere Informationen finden Sie unter Autorisierungsparameter.

MANIFEST

Gibt an, dass ein Manifest verwendet wird, um die Datendateien aufzulisten, die aus Amazon S3 geladen werden sollen. Wenn der Parameter MANIFEST verwendet wird, lädt COPY Daten aus den in dem Manifest aufgelisteten Dateien, das durch 's3://copy_from_s3_manifest_file' referenziert wird. Wenn die Manifestdatei nicht gefunden wird oder nicht korrekt formatiert ist, schlägt COPY fehl. Weitere Informationen finden Sie unter Verwenden eines Manifests für die Angabe von Datendateien.

ENCRYPTED

Eine Klausel, die angibt, dass die Eingabedateien in Amazon S3 mittels clientseitiger Verschlüsselung mit vom Kunden verwalteten Schlüsseln verschlüsselt sind. Weitere Informationen finden Sie unter Laden verschlüsselter Datendateien aus Amazon S3. Geben

Sie ENCRYPTED nicht an, wenn die Eingabedateien mittels serverseitiger Amazon S3-Verschlüsselung (SSE-KMS or SSE-S3) verschlüsselt sind. COPY liest serverseitig verschlüsselte Dateien automatisch.

Wenn Sie den Parameter ENCRYPTED angeben, müssen Sie auch den Parameter MASTER_SYMMETRIC_KEY angeben oder den Wert für **`master_symmetric_key`** in die Zeichenfolge CREDENTIALS einfügen.

Wenn die verschlüsselten Dateien ein komprimiertes Format aufweisen, fügen Sie den Parameter GZIP, LZOP, BZIP2 oder ZSTD hinzu.

Manifestdateien und JSONPaths-Dateien dürfen nicht verschlüsselt sein, auch dann nicht, wenn die Option ENCRYPTED angegeben ist.

MASTER_SYMMETRIC_KEY 'root_key'

Der symmetrische Root-Schlüssel, der für die Verschlüsselung von Datendateien in Amazon S3 verwendet wurde. Wenn MASTER_SYMMETRIC_KEY angegeben ist, muss auch der Parameter ENCRYPTED angegeben werden. MASTER_SYMMETRIC_KEY kann nicht mit dem Parameter CREDENTIALS verwendet werden. Weitere Informationen finden Sie unter Laden verschlüsselter Datendateien aus Amazon S3.

Wenn die verschlüsselten Dateien ein komprimiertes Format aufweisen, fügen Sie den Parameter GZIP, LZOP, BZIP2 oder ZSTD hinzu.

REGION [AS] 'aws-region'

Gibt die AWS Region an, in der sich die Quelldaten befinden. REGION ist für COPY aus einem Amazon-S3-Bucket oder einer DynamoDB-Tabelle erforderlich, wenn sich die AWS -Ressource, die die Daten enthält, nicht in derselben Region wie der Amazon-Redshift-Cluster befindet.

Der Wert für aws_region muss einer Region entsprechen, die in der Tabelle Amazon-Redshift-Regionen und -Endpunkte aufgeführt ist.

Wenn der Parameter REGION angegeben ist, müssen sich alle Ressourcen in der angegebenen Region befinden, einschließlich einer Manifestdatei oder mehrerer Amazon-S3-Buckets.

> **Note**
>
> Für die Übertragung von Daten zwischen Regionen werden dem Amazon-S3-Bucket oder der DynamoDB-Tabelle, die die Daten enthält, zusätzliche Gebühren berechnet. Weitere Informationen zu den Preisen finden Sie unter Ausgehende Datenübertragung

von Amazon S3 in eine andere AWS Region auf der Amazon S3 S3-Preisseite und Ausgehende Datenübertragung auf der Preisseite von Amazon DynamoDB.

Standardmäßig nimmt COPY an, dass sich die Daten in derselben Region wie der Amazon-Redshift-Cluster befinden.

Optionale Parameter

Sie können für COPY aus Amazon S3 optional die folgenden Parameter angeben:

- Optionen für das Mapping von Spalten
- Datenformatparameter
- Datenkonvertierungsparameter
- Datenladeoperationen

Nicht unterstützte Parameter

Die folgenden Parameter können Sie für COPY aus Amazon S3 nicht verwenden:

- SSH
- READRATIO

COPY aus Amazon EMR

Sie können den COPY-Befehl verwenden, um Daten parallel aus einem Amazon-EMR-Cluster zu laden, das für das Schreiben von Textdateien zum Hadoop Distributed File System (HDFS) in Form von Dateien mit fester Breite, von durch Zeichen getrennten Dateien, von CSV-Dateien, von JSON-Dateien oder von Avro-Dateien konfiguriert wurde.

Themen

- Syntax
- Beispiel
- Parameter
- Unterstützte Parameter
- Nicht unterstützte Parameter

Syntax

```
FROM 'emr://emr_cluster_id/hdfs_filepath'
authorization
[ optional_parameters ]
```

Beispiel

Im folgenden Beispiel werden Daten aus einem Amazon-EMR-Cluster geladen.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/part-*'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Parameter

FROM

Die Quelle der Daten, die geladen werden sollen.

'emr://emr_cluster_id/hdfs_file_path'

Der eindeutige Bezeichner für den Amazon-EMR-Cluster und den HDFS-Dateipfad, der die Datendateien für den COPY-Befehl referenziert. Die HDFS-Datendateinamen dürfen nicht die Platzhalterzeichen Sternchen (*) und Fragezeichen (?) enthalten.

> **Note**
>
> Der Amazon-EMR-Cluster muss weiter ausgeführt werden, bis die COPY-Operation abgeschlossen ist. Wenn eine der HDFS-Datendateien vor Abschluss der COPY-Operation geändert oder gelöscht wird, kann dies zu unerwarteten Ergebnissen führen. Es ist auch möglich, dass die COPY-Operation fehlschlägt.

Sie können die Platzhalterzeichen Sternchen (*) und Fragezeichen (?) als Teil des hdfs_file_path-Arguments verwenden, um mehrere zu ladende Dateien anzugeben. Beispielsweise identifiziert `'emr://j-SAMPLE2B500FC/myoutput/part*'` die Dateien `part-0000`, `part-0001` usw. Wenn der Dateipfad keine Platzhalterzeichen enthält, wird er als Zeichenfolgeliteral behandelt. Wenn Sie nur einen Ordnernamen angeben, versucht COPY, alle Dateien im Ordner zu laden.

> **Important**
>
> Wenn Sie Platzhalterzeichen oder nur den Ordnernamen verwenden, müssen Sie überprüfen, ob unerwünschte Dateien geladen werden. Einige Prozesse schreiben beispielsweise eine Protokolldatei in den Ausgabeordner.

Weitere Informationen finden Sie unter So laden Sie Daten aus Amazon EMR:.

Autorisierung

Der COPY-Befehl benötigt eine Autorisierung für den Zugriff auf Daten in anderen AWS -Ressourcen, einschließlich Amazon S3, Amazon EMR, Amazon DynamoDB und Amazon EC2. Sie können diese Autorisierung erteilen, indem Sie auf eine AWS Identity and Access Management (IAM-) Rolle verweisen, die Ihrem Cluster zugeordnet ist (rollenbasierte Zugriffskontrolle), oder indem Sie die Zugangsdaten für einen Benutzer angeben (schlüsselbasierte Zugriffskontrolle). Um Sicherheit und Flexibilität zu verbessern, wird die Verwendung der IAM-rollenbasierten Zugriffssteuerung empfohlen. Weitere Informationen finden Sie unter Autorisierungsparameter.

Unterstützte Parameter

Sie können für COPY aus Amazon EMR optional die folgenden Parameter angeben:

- Optionen für das Mapping von Spalten
- Datenformatparameter
- Datenkonvertierungsparameter
- Datenladeoperationen

Nicht unterstützte Parameter

Die folgenden Parameter können Sie für COPY aus Amazon EMR nicht verwenden:

- ENCRYPTED
- MANIFEST
- REGION
- READRATIO

- SSH

COPY von Remote-Hosts (SSH)

Sie können den Befehl COPY verwenden, um Daten parallel Daten aus einem oder mehreren Remote-Hosts wie Amazon-Elastic-Compute-Cloud(Amazon EC2)-Instances oder anderen Computern zu laden. COPY stellt über Secure Shell (SSH) eine Verbindung zu den Remote-Hosts her und führt Befehle auf den Remote-Hosts aus, um Textausgaben zu generieren. Beim Remote-Host kann es sich um eine EC2-Linux-Instance oder einen anderen Unix- oder Linux-Computer handeln, der für die Annahme von SSH-Verbindungen konfiguriert wurde. Amazon Redshift kann eine Verbindung zu mehreren Hosts herstellen und für jeden Host mehrere SSH-Verbindungen öffnen. Amazon Redshift sendet über jede Verbindung einen eindeutigen Befehl, um die Textausgabe an die Standardausgabe des Hosts zu generieren. Amazon Redshift liest diese dann wie eine Textdatei.

Verwenden Sie die FROM-Klausel, um den Amazon S3-Objektschlüssel für die Manifestdatei anzugeben, die die Informationen bereitstellt, die COPY zum Öffnen von SSH-Verbindungen und zum Ausführen der Remote-Befehle verwendet.

Themen

> **Important**
>
> Wenn sich der S3 Bucket, der die Manifestdatei enthält, nicht in derselben AWS -Region wie der Cluster befindet, müssen Sie den Parameter REGION verwenden, um die Region anzugeben, in der sich der Bucket befindet.

Syntax

```
FROM 's3://'ssh_manifest_file' }
authorization
SSH
```

```
| optional-parameters
```

Beispiele

Im folgenden Beispiel wird eine Manifestdatei verwendet, um Daten über SSH aus einem Remote-Host zu laden.

```
copy sales
from 's3://mybucket/ssh_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
ssh;
```

Parameter

FROM

Die Quelle der Daten, die geladen werden sollen.

's3://copy_from_ssh_manifest_file'

Der COPY-Befehl kann über SSH Verbindungen zu mehreren Hosts herstellen und für jeden Host mehrere SSH-Verbindungen erstellen. COPY führt über jede Hostverbindung einen Befehl aus und lädt anschließend die Ausgabe der Befehle parallel in die Tabelle. Das Argument s3://copy_from_ssh_manifest_file gibt den Amazon-S3-Objektschlüssel für die Manifestdatei an, die die Informationen bereitstellt, die COPY zum Öffnen von SSH-Verbindungen und zum Ausführen der Remote-Befehle verwendet.

Das Argument s3://copy_from_ssh_manifest_file muss explizit eine einzelne Datei referenzieren; es darf sich nicht um ein Schlüsselpräfix handeln. Im Folgenden sehen Sie ein Beispiel:

```
's3://mybucket/ssh_manifest.txt'
```

Die Manifestdatei ist eine Textdatei im JSON-Format, die Amazon Redshift zum Herstellen der Verbindung zum Host verwendet. Die Manifestdatei gibt die Endpunkte des SSH-Hosts und die Befehle an, die auf den Hosts ausgeführt werden, um Daten an Amazon Redshift zurückzugeben. Optional können Sie den öffentlichen Schlüssel des Hosts, den Anmeldebenutzernamen und ein obligatorisches Flag für die einzelnen Einträge einschließen. Im folgenden Beispiel wird eine Manifestdatei gezeigt, die zwei SSH-Verbindungen erstellt:

```
{
```

```
    "entries": [
     {"endpoint":"<ssh_endpoint_or_IP>",
           "command": "<remote_command>",
           "mandatory":true,
           "publickey": "<public_key>",
           "username": "<host_user_name>"},
     {"endpoint":"<ssh_endpoint_or_IP>",
           "command": "<remote_command>",
           "mandatory":true,
           "publickey": "<public_key>",
           "username": "<host_user_name>"}
     ]
}
```

Die Manifestdatei enthält für jede SSH-Verbindung jeweils ein `"entries"`-Konstrukt. Es kann mehrere Verbindungen zu einem einzelnen Host oder mehrere Verbindungen zu mehreren Hosts geben. Doppelte Anführungszeichen sind sowohl für Feldnamen als auch für Werte erforderlich. Es muss sich um normale Anführungszeichen (0x22) handeln. Es dürfen keine schrägen oder „smarten“ Anführungszeichen sein. Der einzige Wert, der keine doppelten Anführungszeichen benötigt, ist der boolesche Wert `true` oder `false` für das Feld `"mandatory"`.

Die folgende Liste beschreibt die Felder in der Manifestdatei.

endpoint

Die URL- oder IP-Adresse des Hosts, z. B. `"ec2-111-222-333.compute-1.amazonaws.com"` oder `"198.51.100.0"`.

command

Der Befehl, der durch den Host ausgeführt werden soll, um eine Textausgabe oder eine Binärausgabe im gzip-, lzop-, bzip2- oder zstd-Format zu generieren. Bei diesem Befehl kann es sich um jeden Befehl handeln, zu dessen Ausführung der Benutzer host_user_name berechtigt ist. Beim Befehl kann es sich um einen einfachen Befehl zum Drucken einer Datei oder um die Abfrage einer Datenbank oder das Starten eines Skripts handeln. Die Ausgabe (Textdatei, binäre gzip-Datei, binäre lzop-Datei oder binäre bzip2-Datei) muss ein Format aufweisen, das der Amazon-Redshift-Befehl COPY verarbeiten kann. Weitere Informationen finden Sie unter Vorbereiten der Eingabedaten.

publickey

(Optional) Der öffentliche Schlüssel des Hosts. Wenn angegeben, verwendet Amazon Redshift den öffentlichen Schlüssel, um den Host zu identifizieren. Wenn der öffentliche Schlüssel

nicht angegeben ist, versucht Amazon Redshift nicht, den Host zu identifizieren. Wenn beispielsweise der öffentliche Schlüssel des Remote-Hosts `ssh-rsa AbcCbaxxx…Example root@amazon.com` ist, geben Sie den folgenden Text in das Feld für den öffentlichen Schlüssel ein: `"AbcCbaxxx…Example"`.

mandatory

(Optional) Eine Klausel, die anzeigt, ob der COPY-Befehl fehlschlagen soll, wenn der Verbindungsversuch fehlschlägt. Der Standardwert ist `false`. Wenn Amazon Redshift nicht mindestens eine Verbindung herstellt, schlägt der COPY-Befehl fehl.

username

(Optional) Der Benutzername, der für die Anmeldung am Hostsystem und die Ausführung des Remotebefehls verwendet wird. Der Benutzeranmeldename muss mit dem Anmeldenamen identisch sein, der zum Hinzufügen des öffentlichen Schlüssels des Amazon-Redshift-Clusters zur Datei des Hosts mit den autorisierten Schlüsseln verwendet wurde. Der Standardbenutzername ist `redshift`.

Weitere Informationen zum Erstellen einer Manifestdatei finden Sie unter Prozess für das Laden von Daten.

Um eine COPY-Operation aus einem Remote-Host auszuführen, muss für den COPY-Befehl der Parameter SSH angegeben werden. Wenn der Parameter SSH nicht angegeben ist, nimmt COPY an, dass die für FROM angegebene Datei eine Datendatei ist und schlägt fehl.

Wenn Sie die automatische Komprimierung verwenden, führt der COPY-Befehl zwei Datenleseoperationen aus. Das bedeutet, dass der Remotebefehl zweimal ausgeführt wird. Die erste Leseoperation dient dazu, eine Datenstichprobe zur Kompressionsanalyse bereitzustellen. In der zweiten Leseoperation werden die Daten tatsächlich geladen. Wenn die zweimalige Ausführung des Remotebefehls Probleme verursachen könnte, sollten Sie die automatische Kompression deaktivieren. Um die automatische Kompression zu deaktivieren, führen Sie den COPY-Befehl aus, wobei Sie den Parameter COMPUPDATE auf OFF festlegen. Weitere Informationen finden Sie unter Laden von Tabellen mit automatischer Kompression.

Details zur Verwendung der COPY-Operation aus SSH finden Sie unter Laden von Daten aus Remote-Hosts.

Autorisierung

Der COPY-Befehl benötigt eine Autorisierung für den Zugriff auf Daten in anderen AWS -Ressourcen, einschließlich Amazon S3, Amazon EMR, Amazon DynamoDB und Amazon

EC2. Sie können diese Autorisierung erteilen, indem Sie auf eine AWS Identity and Access Management (IAM-) Rolle verweisen, die Ihrem Cluster zugeordnet ist (rollenbasierte Zugriffskontrolle), oder indem Sie die Zugangsdaten für einen Benutzer angeben (schlüsselbasierte Zugriffskontrolle). Um Sicherheit und Flexibilität zu verbessern, wird die Verwendung der IAM-rollenbasierten Zugriffssteuerung empfohlen. Weitere Informationen finden Sie unter Autorisierungsparameter.

SSH

Eine Klausel, die angibt, dass die Daten über das SSH-Protokoll aus einem Remote-Host geladen werden sollen. Wenn Sie SSH angeben, müssen Sie auch unter Verwendung des Arguments s3://copy_from_ssh_manifest_file eine Manifestdatei angeben.

> **Note**
>
> Wenn Sie SSH verwenden, um von einem Host mit einer privaten IP-Adresse in einem Remote-VPC zu kopieren, muss für den VPC erweitertes VPC-Routing aktiviert sein. Für weitere Informationen zu erweitertem VPC-Routing vgl. Amazon Redshift Enhanced VPC Routing.

Optionale Parameter

Sie können für COPY aus SSH optional die folgenden Parameter angeben:

- Optionen für das Mapping von Spalten
- Datenformatparameter
- Datenkonvertierungsparameter
- Datenladeoperationen

Nicht unterstützte Parameter

Die folgenden Parameter können Sie für COPY aus SSH nicht verwenden:

- ENCRYPTED
- MANIFEST
- READRATIO

COPY aus Amazon DynamoDB

Um Daten aus einer vorhandenen DynamoDB-Tabelle zu laden, verwenden Sie die FROM-Klausel, um den Namen der DynamoDB-Tabelle anzugeben.

Themen

> **Important**
>
> Wenn sich die DynamoDB-Tabelle nicht in derselben Region wie Ihr Amazon-Redshift-Cluster befindet, müssen Sie den Parameter REGION verwenden, um die Region anzugeben, in der sich die Daten befinden.

Syntax

```
FROM 'dynamodb://table-name'
authorization
READRATIO ratio
| REGION [AS] 'aws_region'
| optional-parameters
```

Beispiele

Im folgenden Beispiel werden Daten aus einer DynamoDB-Tabelle geladen.

```
copy favoritemovies from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Parameter

FROM

Die Quelle der Daten, die geladen werden sollen.

'dynamodb://table-name'

Der Name der DynamoDB-Tabelle, die die Daten enthält, beispielsweise `'dynamodb://ProductCatalog'`. Details dazu, wie DynamoDB-Attribute den Amazon-Redshift-Spalten zugewiesen werden, finden Sie unter Laden von Daten aus einer Amazon-DynamoDB-Tabelle.

Ein DynamoDB-Tabellenname ist für ein AWS Konto eindeutig, das durch die AWS Zugangsdaten identifiziert wird.

Autorisierung

Der COPY-Befehl benötigt eine Autorisierung für den Zugriff auf Daten in anderen AWS -Ressourcen, einschließlich Amazon S3, Amazon EMR, DynamoDB und Amazon EC2. Sie können diese Autorisierung gewähren, indem Sie auf eine AWS Identity and Access Management (IAM-) Rolle verweisen, die Ihrem Cluster zugeordnet ist (rollenbasierte Zugriffskontrolle), oder indem Sie die Zugangsdaten für einen Benutzer angeben (schlüsselbasierte Zugriffskontrolle). Um Sicherheit und Flexibilität zu verbessern, wird die Verwendung der IAM-rollenbasierten Zugriffssteuerung empfohlen. Weitere Informationen finden Sie unter Autorisierungsparameter.

READRATIO [AS] ratio

Der Prozentsatz des für die DynamoDB-Tabelle bereitgestellten Durchsatzes, der für das Laden der Daten verwendet werden soll. READRATIO ist für COPY aus DynamoDB erforderlich. Der Parameter kann nicht für COPY aus Amazon S3 verwendet werden. Es wird nachdrücklich empfohlen, das Verhältnis auf einen Wert festzulegen, der kleiner als der durchschnittliche Wert für nicht genutzten bereitgestellten Durchsatz ist. Gültige Werte sind Ganzzahlen von 1 bis 200.

> **Important**
>
> Wenn Sie READRATIO auf 100 oder höher festlegen, kann Amazon Redshift den gesamten Durchsatz nutzen, der für die DynamoDB-Tabelle bereitgestellt wurde. Dies führt zu einer erheblich schlechteren Leistung für Leseoperationen, die für dieselbe Tabelle gleichzeitig mit der COPY-Sitzung ausgeführt werden. Der Schreibdatenverkehr ist nicht betroffen. Werte über 100 sind zulässig, um Fehler im Zusammenhang mit seltenen Szenarien zu beheben, wenn Amazon Redshift den für die Tabelle bereitgestellten Durchsatz nicht erreicht. Wenn Sie kontinuierlich Daten aus DynamoDB in Amazon Redshift laden, sollten Sie Ihre DynamoDB-Tabellen als Zeitreihe organisieren, um den Live-Datenverkehr von der COPY-Operation zu trennen.

Optionale Parameter

Sie können für COPY aus Amazon DynamoDB optional die folgenden Parameter angeben:

- Optionen für das Mapping von Spalten
- Folgende Datenkonvertierungsparameter werden unterstützt:
  - ACCEPTANYDATE
  - BLANKSASNULL
  - DATEFORMAT
  - EMPTYASNULL
  - ROUNDEC
  - TIMEFORMAT
  - TRIMBLANKS
  - TRUNCATECOLUMNS
- Datenladeoperationen

Nicht unterstützte Parameter

Die folgenden Parameter können Sie für COPY aus DynamoDB nicht verwenden:

- Alle Datenformatparameter
- ESCAPE
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL
- REMOVEQUOTES
- ACCEPTINVCHARS
- MANIFEST
- ENCRYPTED

Autorisierungsparameter

Der COPY-Befehl benötigt eine Autorisierung, um auf Daten in einer anderen AWS Ressource zuzugreifen, einschließlich in Amazon S3, Amazon EMR, Amazon DynamoDB und Amazon EC2. Sie gewähren diese Autorisierung durch die Referenzierung einer IAM-Rolle (AWS Identity and Access Management), die mit Ihrem Cluster verbunden ist (rollenbasierte Zugriffskontrolle). Sie können Ihre Ladedaten auf Amazon S3 verschlüsseln.

In den folgenden Themen finden Sie weitere Details und Beispiele für Authentifizierungsoptionen:

- IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY
- Rollenbasierte Zugriffskontrolle
- Schlüsselbasierte Zugriffssteuerung

Verwenden Sie eines der folgenden Verfahren, um eine Autorisierung für den COPY-Befehl bereitzustellen:

- IAM_ROLE Parameter
- ACCESS_KEY_ID and SECRET_ACCESS_KEY parameters
- CREDENTIALS-Klausel

IAM_ROLE { default | 'arn:aws:iam::*<AWS-Konto-id>*:role/*<role-name>*' }

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der COPY-Befehl ausgeführt wird.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Wenn Sie IAM_ROLE angeben, können Sie ACCESS_KEY_ID und SECRET_ACCESS_KEY, SESSION_TOKEN oder CREDENTIALS nicht verwenden.

Im Folgenden wird die Syntax für den Parameter IAM_ROLE gezeigt.

```
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>' }
```

Weitere Informationen finden Sie unter Rollenbasierte Zugriffskontrolle.

ACCESS_KEY_ID 'SECRET_ACCESS_KEY' geheimer Zugriffsschlüssel 'access-key-id

Diese Autorisierungsmethode wird nicht empfohlen.

> **Note**
>
> Es wird nach nachdrücklich empfohlen, eine rollenbasierte Authentifizierung durch Angabe des Parameters IAM_ROLE zu verwenden, statt die Zugriffsanmeldeinformationen in Textform bereitzustellen. Weitere Informationen finden Sie unter Rollenbasierte Zugriffskontrolle.

SESSION_TOKEN 'temporary-token'

Das Sitzungstoken, das für temporäre Zugriffsanmeldeinformationen verwendet wird. Wenn SESSION_TOKEN angegeben wird, müssen Sie auch ACCESS_KEY_ID und SECRET_ACCESS_KEY verwenden, um temporäre Zugriffsschlüssel-Anmeldeinformationen bereitzustellen. Wenn Sie SESSION_TOKEN angeben, können Sie IAM_ROLE oder CREDENTIALS nicht verwenden. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch.

> **Note**
>
> Es wird nach nachdrücklich empfohlen, eine rollenbasierte Authentifizierung zu verwenden, statt temporäre Sicherheitsanmeldeinformationen zu erstellen. Wenn Sie eine Autorisierung unter Verwendung einer IAM-Rolle ausführen, erstellt Amazon Redshift automatisch für jede Sitzung temporäre Benutzeranmeldeinformationen. Weitere Informationen finden Sie unter Rollenbasierte Zugriffskontrolle.

Im Folgenden wird die Syntax für den Parameter SESSION_TOKEN mit den Parametern ACCESS_KEY_ID und SECRET_ACCESS_KEY gezeigt.

```
ACCESS_KEY_ID '<access-key-id>'
SECRET_ACCESS_KEY '<secret-access-key>'
SESSION_TOKEN '<temporary-token>';
```

Wenn Sie SESSION_TOKEN angeben, können Sie CREDENTIALS oder IAM_ROLE nicht verwenden.

[WITH] CREDENTIALS [AS] 'credentials-args'

Eine Klausel, die angibt, mit welcher Methode Ihr Cluster auf andere Ressourcen zugreift, die Daten- oder Manifestdateien enthalten. AWS Sie können den Parameter CREDENTIALS nicht mit IAM_ROLE oder ACCESS_KEY_ID und SECRET_ACCESS_KEY verwenden.

> **Note**
>
> Um die Flexibilität zu verbessern, wird die Verwendung des Parameters IAM_ROLE anstelle des Parameters CREDENTIALS empfohlen.

Wenn der Parameter ENCRYPTED verwendet wird, stellt die Zeichenfolge credentials-args auch den Verschlüsselungsschlüssel bereit.

Die Zeichenfolge credentials-args unterscheidet zwischen Groß- und Kleinschreibung und darf keine Leerzeichen enthalten.

Die Schlüsselwörter WITH und AS sind optional und werden ignoriert.

Sie können entweder role-based access control oder key-based access control angeben. In beiden Fällen müssen die IAM-Rolle oder der Benutzer die nötigen Berechtigungen besitzen, um auf die angegebenen AWS -Ressourcen zuzugreifen. Weitere Informationen finden Sie unter IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY.

> **Note**
>
> Um Ihre AWS Anmeldeinformationen und vertrauliche Daten zu schützen, empfehlen wir dringend, die rollenbasierte Zugriffskontrolle zu verwenden.

Um die rollenbasierte Zugriffssteuerung anzugeben, stellen Sie die Zeichenfolge credentials-args im folgenden Format bereit.

```
'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Um temporäre Token-Anmeldeinformationen zu verwenden, müssen Sie die temporäre Zugriffschlüssel-ID, den temporären geheimen Zugriffschlüssel und das temporäre Token bereitstellen. Die credentials-args-Zeichenfolge hat das folgende Format.

```
CREDENTIALS
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-
secret-access-key>;token=<temporary-token>'
```

Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen.

Wenn der Parameter ENCRYPTED verwendet wird, hat die Zeichenfolge credentials-args das folgende Format, wobei*<root-key>* der Wert des Root-Schlüssels ist, der verwendet wurde, um die Dateien zu verschlüsseln.

```
CREDENTIALS
'<credentials-args>;master_symmetric_key=<root-key>'
```

Der folgende COPY-Befehl verwendet beispielsweise die rollenbasierte Zugriffssteuerung mit einem Verschlüsselungsschlüssel.

```
copy customer from 's3://mybucket/mydata'
credentials
'aws_iam_role=arn:aws:iam::<account-id>:role/<role-name>;master_symmetric_key=<root-
key>'
```

Der folgende COPY-Befehl zeigt die rollenbasierte Zugriffssteuerung mit einem Verschlüsselungsschlüssel.

```
copy customer from 's3://mybucket/mydata'
credentials
'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-
name>;master_symmetric_key=<root-key>'
```

Optionen für das Mapping von Spalten

Standardmäßig fügt COPY Werte in derselben Reihenfolge in die Spalten der Zieltabelle ein, die die Felder in den Datendateien haben. Wenn die standardmäßige Spaltenreihenfolge nicht funktionieren wird, können Sie eine Spaltenliste angeben oder JSONPath-Ausdrücke verwenden, um den Zielspalten Quelldatenfelder zuzuweisen.

- Column List
- JSONPaths File

Spaltenliste

Sie können eine durch Komma getrennte Liste von Spaltennamen angeben, um Quelldatenfelder in spezifische Zielspalten zu laden. Die Spalten können in der COPY-Anweisung eine beliebige Reihenfolge aufweisen. Wenn sie jedoch aus Flat-Files geladen werden, beispielsweise in einem Amazon-S3-Bucket, muss ihre Reihenfolge mit der Reihenfolge der Quelldaten übereinstimmen.

Beim Laden aus einer Amazon-DynamoDB-Tabelle spielt die Reihenfolge keine Rolle. Der COPY-Befehl gleicht Attributnamen in den Elementen, die aus der DynamoDB-Tabelle abgerufen wurden, mit Spaltennamen in der Amazon-Redshift-Tabelle ab. Weitere Informationen finden Sie unter Laden von Daten aus einer Amazon-DynamoDB-Tabelle

Das Format einer Spaltenliste ist wie folgt.

```
COPY tablename (column1 [,column2, ...])
```

Wenn eine Spalte in der Zieltabelle aus der Spaltenliste ausgelassen wird, lädt COPY den DEFAULT-Ausdruck der Zielspalte.

Wenn die Zielspalte keinen Standardwert besitzt, versucht COPY, NULL zu laden.

Wenn COPY versucht, einer Spalte NULL zuzuweisen, die als NOT NULL definiert ist, schlägt der COPY-Befehl fehl.

Wenn eine IDENTITY-Spalte in der Spaltenliste enthalten ist, muss auch EXPLICIT_IDS angegeben werden. Wenn keine IDENTITY-Spalte angegeben ist, kann EXPLICIT_IDS nicht angegeben werden. Wenn keine Spaltenliste angegeben ist, verhält sich der Befehl, als ob eine vollständige, reihenfolgegerechte Spaltenliste angegeben ist, wobei IDENTITY-Spalten ausgelassen werden, wenn EXPLICIT_IDS nicht angegeben wurde.

Wenn eine Spalte mit GENERATED BY DEFAULT AS IDENTITY definiert ist, kann sie kopiert werden. Die Werte werden mit den von Ihnen angegebenen Werten generiert oder aktualisiert. Die Option EXPLICIT_IDS ist nicht erforderlich. COPY aktualisiert nicht die Identity High Watermark. Weitere Informationen finden Sie unter GENERATED BY DEFAULT AS IDENTITY.

JSONPaths-Datei

Beim Laden von Daten aus Datendateien im JSON- oder Avro-Format weist COPY die Datenelemente in den JSON- oder Avro-Quelldaten automatisch den Spalten in der Zieltabelle zu.

Dies geschieht durch den Abgleich von Feldnamen im Avro-Schema mit den Spaltennamen in der Zieltabelle oder Spaltenliste.

In einigen Fällen werden die Spaltennamen und Feldnamen nicht übereinstimmen oder Sie müssen Zuweisungen tiefer in der Datenhierarchie vornehmen. Um JSON- oder Avro-Datenelemente explizit Spalten zuzuweisen, können Sie eine JSONPaths-Datei verwenden.

Weitere Informationen finden Sie unter JSONPaths-Datei.

Datenformatparameter

Standardmäßig geht der COPY-Befehl davon aus, dass es sich bei den Quelldaten um UTF-8-Text mit Trennzeichen handelt. Das Standardtrennzeichen ist der senkrechte Strich (|). Wenn sich die Quelldaten in einem anderen Format befinden, verwenden Sie die folgenden Parameter, um das Datenformat anzugeben:

- FORMAT
- CSV
- DELIMITER
- FIXEDWIDTH
- SHAPEFILE
- AVRO
- JSON
- PARQUET
- ORC

Neben den Standarddatenformaten unterstützt COPY die folgenden spaltenbasierten Datenformate für COPY aus Amazon S3:

- ORC
- PARQUET

COPY aus einem spaltenbasierten Format wird mit bestimmten Einschränkungen unterstützt. Weitere Informationen finden Sie unter COPY aus spaltenbasierten Datenformaten.

Datenformatparameter

FORMAT [AS]

(Optional) Identifiziert Datenformatschlüsselwörter. Die FORMAT-Argumente werden im Folgenden beschrieben.

CSV [QUOTE [AS] 'quote_character']

Ermöglicht die Verwendung des CSV-Formats in den Eingabedaten. Um Trennzeichen, Zeichen für neue Zeilen und Zeilenumbrüche automatisch mit Escape-Zeichen zu markieren, schließen Sie das Feld in das durch den Parameter QUOTE angegebene Zeichen ein. Das Standardanführungszeichen ist das doppelte Anführungszeichen ("). Wird das Anführungszeichen innerhalb des Feldes verwendet, verwenden Sie ein weiteres Anführungszeichen als Escape-Zeichen. Wenn beispielsweise das Anführungszeichen ein doppeltes Anführungszeichen ist und die Zeichenfolge `A "quoted" word` eingefügt werden soll, sollte die Eingabedatei die Zeichenfolge `"A ""quoted"" word"` enthalten. Wenn der CSV-Parameter verwendet wird, ist das Standardtrennzeichen ein Komma (,). Sie können ein anderes Trennzeichen angeben, indem Sie den Parameter DELIMITER verwenden.

Wenn ein Feld in Anführungszeichen eingeschlossen wird, werden Leerzeichen zwischen Trennzeichen und Anführungszeichen ignoriert. Wenn das Trennzeichen ein Leerzeichen ist, beispielsweise ein Tabulatorzeichen, wird das Trennzeichen nicht als Leerzeichen behandelt.

Das CSV-Format kann nicht mit FIXEDWIDTH, REMOVEQUOTES oder ESCAPE verwendet werden.

QUOTE [AS] 'quote_character'

Optional. Gibt das Zeichen an, das als Anführungszeichen verwendet werden soll, wenn der CSV-Parameter verwendet wird. Der Standardwert ist das doppelte Anführungszeichen ("). Wenn Sie den Parameter QUOTE verwenden, um ein anderes Anführungszeichen als ein doppeltes Anführungszeichen zu definieren, müssen Sie doppelte Anführungszeichen innerhalb des Felds nicht mit Escape-Zeichen markieren. Der Parameter QUOTE kann nur mit dem Parameter CSV verwendet werden. Das Schlüsselwort AS ist optional.

DELIMITER [AS] ['delimiter_char']

Gibt das einzelne ASCII-Zeichen an, das verwendet wird, um Felder in der Eingabedatei zu trennen, beispielsweise ein Pipe-Zeichen (|), ein Komma (,) oder ein Tabulatorzeichen (\t). ASCII-Zeichen, die nicht gedruckt werden, werden unterstützt. ASCII-Zeichen können auch in oktaler Form dargestellt werden, und zwar im Format '\ddd', wobei 'd' eine Oktalziffer (0–7) ist. Das

Standardtrennzeichen ist das Pipe-Zeichen (|), wenn nicht der Parameter CSV verwendet wird. In diesem Fall ist das Standardtrennzeichen ein Komma (,). Das Schlüsselwort AS ist optional. DELIMITER kann nicht mit FIXEDWIDTH verwendet werden.

FIXEDWIDTH 'fixedwidth_spec'

Lädt die Daten aus einer Datei, in der jede Spalte eine feste Breite hat, statt durch ein Trennzeichen abgetrennt zu werden. fixedwidth_spec ist eine Zeichenfolge, die eine benutzerdefinierte Spaltenbezeichnung und eine benutzerdefinierte Spaltenbreite angibt. Die Spaltenbezeichnung kann eine Textzeichenfolge oder eine Ganzzahl sein, abhängig davon, wofür sich der Benutzer entscheidet. Spaltenbezeichnung und Spaltenname sind nicht aufeinander bezogen. Die Reihenfolge der Paare aus Bezeichnung/Breite muss exakt mit der Reihenfolge der Tabellenspalten übereinstimmen. FIXEDWIDTH kann nicht mit CSV oder DELIMITER verwendet werden. In Amazon Redshift wird die Länge der CHAR- und VARCHAR-Spalten in Bytes ausgedrückt. Achten Sie daher darauf, dass die von Ihnen angegebene Spaltenbreite beim Vorbereiten der Datei auf das Laden die binäre Länge von Multibyte-Zeichen berücksichtigt. Weitere Informationen finden Sie unter Zeichentypen.

Das Format für fixedwidth_spec wird im Folgenden gezeigt:

```
'colLabel1:colWidth1,colLabel:colWidth2, ...'
```

SHAPEFILE [SIMPLIFY [AUTO] ['Toleranz']]

Ermöglicht die Verwendung des SHAPEFILE-Formats in den Eingabedaten. Standardmäßig ist die erste Spalte des Shapefile entweder eine `GEOMETRY`- oder `IDENTITY`-Spalte. Alle nachfolgenden Spalten folgen der im Shapefile angegebenen Reihenfolge.

SHAPEFILE kann nicht mit FIXEDWIDTH, REMOVEQUOTES oder ESCAPE verwendet werden.

Um `GEOGRAPHY`-Objekte mit `COPY FROM SHAPEFILE` zu verwenden, erfassen Sie zuerst in eine `GEOMETRY`-Spalte und wandeln dann die Objekte in `GEOGRAPHY`-Objekte um.

SIMPLIFY [Toleranz]

(Optional) Vereinfacht alle Geometrien während der Erfassung mithilfe des Ramer-Douglas-Peucker-Algorithmus und der angegebenen Toleranz.

SIMPLIFY AUTO [Toleranz]

(Optional) Vereinfacht nur Geometrien, die größer als die maximale Geometriegröße sind. Diese Vereinfachung verwendet den Ramer-Douglas-Peucker-Algorithmus und

die automatisch berechnete Toleranz, wenn diese die angegebene Toleranz nicht überschreitet. Der Algorithmus berechnet die Größe zum Speichern von Objekten innerhalb der angegebenen Toleranz. Der Toleranz-Wert ist optional.

Beispiele zum Laden von Shapefiles finden Sie unter Laden eines Shapefile in Amazon Redshift.

AVRO [AS] 'avro_option'

Gibt an, dass die Quelldaten im Avro-Format vorliegen.

Das Avro-Format wird durch folgende Services und Protokolle für den COPY-Befehl unterstützt:

- Amazon S3
- Amazon EMR
- Remote-Hosts (SSH)

Avro wird für COPY aus DynamoDB nicht unterstützt.

Avro ist ein Protokoll für die Datenserialisierung. Eine Avro-Quelldatei enthält ein Schema, das die Struktur der Daten definiert. Der Avro-Schematyp muss sein `record`. COPY akzeptiert Avro-Dateien, die unter Verwendung des standardmäßigen nicht komprimierten Codec sowie der Komprimierungs-Codecs `deflate` und `snappy` erstellt wurden. Weitere Informationen zu Avro finden Sie unter Apache Avro.

Gültige Werte für avro_option sind folgende:

- `'auto'`
- `'auto ignorecase'`
- `'s3://jsonpaths_file'`

Der Standardwert ist `'auto'`.

COPY weist die Datenelemente in den Avro-Quelldaten automatisch den Spalten in der Zieltabelle zu. Dies geschieht durch den Abgleich von Feldnamen im Avro-Schema mit den Spaltennamen in der Zieltabelle. Mit `'auto'` wird beim Abgleich die Groß-/Kleinschreibung beachtet, mit `'auto ignorecase'` nicht.

Spaltennamen in Amazon-Redshift-Tabellen verwenden stets Kleinbuchstaben. Daher müssen übereinstimmende Feldnamen ebenfalls Kleinbuchstaben verwenden, wenn Sie die Option `'auto'` verwenden. Wenn die Feldnamen nicht alle Kleinbuchstaben sind, können Sie die Option `'auto ignorecase'` verwenden. Mit dem Standardargument `'auto'` erkennt COPY nur die erste Ebene von Feldern bzw. die äußeren Felder in der Struktur.

Um den Avro-Feldnamen explizit Spaltennamen zuzuordnen, können Sie verwenden JSONPaths-Datei.

Standardmäßig versucht COPY, alle Spalten in der Zieltabelle mit Avro-Feldnamen abzugleichen. Um einen Subsatz der Spalten zu laden, können Sie optional eine Spaltenliste angeben. Wenn eine Spalte in der Zieltabelle aus der Spaltenliste ausgelassen wird, lädt COPY den DEFAULT-Ausdruck der Zielspalte. Wenn die Zielspalte keinen Standardwert besitzt, versucht COPY, NULL zu laden. Wenn eine Spalte in der Spaltenliste enthalten ist und COPY in den Avro-Daten kein übereinstimmendes Feld findet, versucht COPY, NULL in die Spalte zu laden.

Wenn COPY versucht, einer Spalte NULL zuzuweisen, die als NOT NULL definiert ist, schlägt der COPY-Befehl fehl.

Avro-Schema

Eine Avro-Quelldatendatei enthält ein Schema, das die Struktur der Daten definiert. COPY liest das Schema, das Teil der Avro-Quelldatendatei ist, um Datenelemente zu Zieltabellenspalten zuzuweisen. Die folgende Abbildung zeigt ein Avro-Schema.

```
{
    "name": "person",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "guid", "type": "string"},
        {"name": "name", "type": "string"},
        {"name": "address", "type": "string"}]
}
```

Das Avro-Schema wird unter Verwendung des JSON-Formats definiert. Das JSON-Objekt auf der obersten Ebene enthält drei Name/Wert-Paare mit den Namen bzw. Schlüsseln `"name"`, `"type"` und `"fields"`.

Der `"fields"`-Schlüssel wird mit einem Array von Objekten verknüpft, die den Namen und Datentyp der einzelnen Felder in der Datenstruktur definieren. Standardmäßig gleicht COPY die Feldnamen mit Spaltennamen ab. Spaltennamen verwenden stets Kleinbuchstaben. Daher müssen übereinstimmende Feldnamen ebenfalls Kleinbuchstaben verwenden, außer Sie geben die Option `‘auto ignorecase’` an. Feldnamen, die mit keiner Spalte übereinstimmen, werden ignoriert. Die Reihenfolge spielt keine Rolle. Im vorherigen Beispiel führt COPY einen Abgleich mit den Spaltennamen `id`, `guid`, `name` und `address` aus.

Bei Verwendung des Standardarguments `'auto'` gleicht COPY nur Objekte auf der ersten Ebene mit Spalten ab. Um einen Abgleich mit tieferen Ebenen im Schema auszuführen oder wenn Feldnamen und Spaltennamen nicht übereinstimmen, verwenden Sie eine JSONPaths-Datei, um das Mapping zu definieren. Weitere Informationen finden Sie unter JSONPaths-Datei.

Wenn es sich bei dem Wert, der mit einem Schlüssel verknüpft ist, um einen komplexen Avro-Datentyp wie Byte, Array, Datensatz, Zuweisung oder Link handelt, lädt COPY den Wert als Zeichenfolge. Hier ist die Zeichenfolge die JSON-Darstellung der Daten. COPY lädt Avro-Aufzählungsdatentypen als Zeichenfolgen, wobei der Inhalt der Name des Typs ist. Ein Beispiel finden Sie unter COPY von JSON-Format.

Die maximale Größe des Avro-Dateiheaders, der das Schema und die Dateimetadaten enthält, beträgt 1 MB.

Die maximale Größe eines einzelnen Avro-Datenblocks beträgt 4 MB. Dies ist etwas anderes als die maximale Zeilengröße. Wenn die maximale Größe eines einzelnen Avro-Datenblocks überschritten wird, schlägt der COPY-Befehl fehl. Dies ist auch dann der Fall, wenn die entsprechende Zeilengröße unter der Grenze von 4 MB für Zeilengrößen liegt.

Bei der Berechnung der Zeilengröße zählt Amazon Redshift intern Pipe-Zeichen (|) zweimal. Wenn Ihre Eingabedaten eine sehr große Zahl von Pipe-Zeichen enthalten, ist es möglich, dass die Zeilengröße 4 MB überschreitet, auch wenn der Datenblock kleiner als 4 MB ist.

JSON [AS] 'json_option'

Die Quelldaten weisen das JSON-Format auf.

Das JSON-Format wird durch folgende Services und Protokolle für den COPY-Befehl unterstützt:

- Amazon S3
- COPY aus Amazon EMR
- COPY aus SSH

JSON wird für COPY aus DynamoDB nicht unterstützt.

Gültige Werte für json_option sind folgende:

- `'auto'`
- `'auto ignorecase'`
- `'s3://jsonpaths_file'`

- `'noshred'`

Der Standardwert ist `'auto'`. Amazon Redshift zerlegt die Attribute von JSON-Strukturen beim Laden eines JSON-Dokuments nicht in mehrere Spalten.

Standardmäßig versucht COPY, alle Spalten in der Zieltabelle mit JSON-Feldnamenschlüsseln abzugleichen. Um einen Subsatz der Spalten zu laden, können Sie optional eine Spaltenliste angeben. Wenn die JSON-Feldnamenschlüssel nicht nur Kleinbuchstaben verwenden, können Sie `'auto ignorecase'` oder JSONPaths-Datei verwenden, um Spaltennamen explizit zu JSON-Feldnamenschlüsseln zuzuweisen.

Wenn eine Spalte in der Zieltabelle aus der Spaltenliste ausgelassen wird, lädt COPY den DEFAULT-Ausdruck der Zielspalte. Wenn die Zielspalte keinen Standardwert besitzt, versucht COPY, NULL zu laden. Wenn eine Spalte in der Spaltenliste enthalten ist und COPY in den JSON-Daten kein übereinstimmendes Feld findet, versucht COPY, NULL in die Spalte zu laden.

Wenn COPY versucht, einer Spalte NULL zuzuweisen, die als NOT NULL definiert ist, schlägt der COPY-Befehl fehl.

COPY weist die Datenelemente in den JSON-Quelldaten den Spalten in der Zieltabelle zu. Dies geschieht durch den Abgleich der Objektschlüssel oder -namen in den Name-Wert-Paaren der Quelle mit den Namen der Spalten in der Zieltabelle.

Im Folgenden finden Sie nähere Angaben zu den einzelnen json_option-Werten:

'auto'

Bei dieser Option wird zwischen Groß- und Kleinschreibung unterschieden. Spaltennamen in Amazon-Redshift-Tabellen verwenden stets Kleinbuchstaben. Daher müssen übereinstimmende JSON-Feldnamen ebenfalls Kleinbuchstaben verwenden, wenn Sie die Option `'auto'` verwenden.

'auto ignorecase'

Bei dieser Option wird nicht zwischen Groß- und Kleinschreibung unterschieden. Spaltennamen in Amazon-Redshift-Tabellen verwenden stets Kleinbuchstaben. Wenn Sie die Option `'auto ignorecase'` verwenden, können entsprechende JSON-Feldnamen Kleinbuchstaben, Großbuchstaben oder beides verwenden.

's3://jsonpaths_file'

Mit dieser Option verwendet COPY die benannte JSONPaths-Datei, um die Datenelemente in den JSON-Quelldaten den Spalten in der Zieltabelle zuzuweisen. Das *s3://*

jsonpaths_file-Argument muss ein Amazon S3-Objektschlüssel sein, der explizit auf eine einzelne Datei verweist. Ein Beispiel ist `'s3://mybucket/jsonpaths.txt'`. Das Argument darf kein Schlüsselpräfix sein. Weitere Informationen zur Verwendung einer JSONPaths-Datei finden Sie unter the section called “JSONPaths-Datei”.

In einigen Fällen hat die von `jsonpaths_file` angegebene Datei dasselbe Präfix wie der von `copy_from_s3_objectpath` angegebene Pfad für die Datendateien. In diesem Fall liest COPY die JSONPaths-Datei als Datendatei und gibt Fehler zurück. Angenommen, Ihre Datendateien verwenden den Objektpfad `s3://mybucket/my_data.json` und Ihre JSONPaths-Datei ist `s3://mybucket/my_data.jsonpaths`. In diesem Fall versucht COPY, `my_data.jsonpaths` als Datendatei zu laden.

'noshred'

Mit dieser Option zerlegt Amazon Redshift die Attribute von JSON-Strukturen beim Laden eines JSON-Dokuments nicht in mehrere Spalten.

JSON-Datendatei

Die JSON-Datendatei enthält einen Satz von Objekten oder einen Satz von Arrays. COPY lädt jedes JSON-Objekt oder -Array in eine einzige Zeile in der Zieltabelle. Jedes Objekt oder Array, das einer Zeile entspricht, muss eine eigenständige Struktur auf Stammebene besitzen, d. h., es darf kein Mitglied einer anderen JSON-Struktur sein.

Ein JSON-Objekt beginnt und endet mit geschweiften Klammern ({}) und enthält eine nicht geordnete Sammlung von Name/Wert-Paaren. Name und Wert in einem Paar werden durch einen Doppelpunkt getrennt und die Paare werden durch Kommas getrennt. Standardmäßig muss der Objektschlüssel bzw. Name in den Name-Wert-Paaren mit dem Namen der entsprechenden Spalte in der Tabelle übereinstimmen. Spaltennamen in Amazon-Redshift-Tabellen verwenden stets Kleinbuchstaben. Daher müssen übereinstimmende JSON-Feldnamenschlüssel ebenfalls Kleinbuchstaben verwenden. Wenn Ihre Spaltennamen und JSON-Schlüssel nicht übereinstimmen, verwenden Sie the section called “JSONPaths-Datei”, um Spalten explizit Schlüsseln zuzuweisen.

Die Reihenfolge in einem JSON-Objekt spielt keine Rolle. Namen, die mit keiner Spalte übereinstimmen, werden ignoriert. Im Folgenden wird die Struktur eines einfachen JSON-Objekts gezeigt.

```
{
  "column1": "value1",
```

```
  "column2": value2,
  "notacolumn" : "ignore this value"
}
```

Ein JSON-Array beginnt und endet mit eckigen Klammern ([]) und enthält eine geordnete Sammlung von Werten, getrennt durch Kommas. Wenn Ihre Datendateien Arrays verwenden, müssen Sie eine JSONPaths-Datei angeben, um die Werte mit Spalten abzugleichen. Im Folgenden wird die Struktur eines einfachen JSON-Arrays gezeigt.

```
["value1", value2]
```

JSON muss wohl geformt sein. Die Objekte oder Arrays dürfen beispielsweise nicht durch Kommas oder andere Zeichen getrennt werden, sondern nur durch Leerzeichen. Zeichenfolgen müssen in doppelte Anführungszeichen eingeschlossen werden. Bei den Anführungszeichen muss es sich um normale Anführungszeichen (0x22) handeln. Es dürfen keine schrägen oder „smarten“ Anführungszeichen sein.

Die maximale Größe eines einzelnen JSON-Objekts oder -Arrays einschließlich geschweifter oder eckiger Klammern beträgt 4 MB. Dies ist etwas anderes als die maximale Zeilengröße. Wenn die maximale Größe eines einzelnen JSON-Objekts oder Arrays überschritten wird, schlägt der COPY-Befehl fehl. Dies ist auch dann der Fall, wenn die entsprechende Zeilengröße unter der Grenze von 4 MB für Zeilengrößen liegt.

Bei der Berechnung der Zeilengröße zählt Amazon Redshift intern Pipe-Zeichen (|) zweimal. Wenn Ihre Eingabedaten eine sehr große Zahl von Pipe-Zeichen enthalten, ist es möglich, dass die Zeilengröße 4 MB überschreitet, auch wenn die Objektgröße weniger als 4 MB beträgt.

COPY lädt `\n` als Zeichen für neue Zeilen und `\t` als Tabulatorzeichen. Um einen Backslash zu laden, muss ein Backslash als Escape-Zeichen verwendet werden (`\\`).

COPY durchsucht die angegebene JSON-Quelle nach einem wohl geformten, gültigen JSON-Objekt oder -Array. Wenn COPY vor dem Auffinden einer verwendbaren JSON-Struktur oder zwischen gültigen JSON-Objekten oder -Arrays auf Zeichen stößt, die keine Leerzeichen sind, gibt COPY für jede Instance einen Fehler zurück. Diese Fehler werden auf die MAXERROR-Fehlerzahl angerechnet. Wenn die Fehlerzahl gleich oder größer als MAXERROR ist, schlägt COPY fehl.

Für jeden Fehler zeichnet Amazon Redshift eine Zeile in der Systemtabelle STL_LOAD_ERRORS auf. Die Spalte LINE_NUMBER zeichnet die letzte Zeile des JSON-Objekts auf, das den Fehler verursacht hat.

Wenn IGNOREHEADER angegeben ist, ignoriert COPY die angegebene Zahl von Zeilen in den JSON-Daten. Zeichen für neue Zeilen in den JSON-Daten werden stets für IGNOREHEADER-Berechnungen berücksichtigt.

COPY lädt leere Zeichenfolgen standardmäßig als leere Felder. Wenn EMPTYASNULL angegeben ist, lädt COPY leere Zeichenfolgen für CHAR- und VARCHAR-Felder als NULL. Leere Zeichenfolgen für andere Datentypen, beispielsweise INT, werden stets mit NULL geladen.

Die folgenden Optionen werden für JSON nicht unterstützt:

- CSV
- DELIMITER
- ESCAPE
- FILLRECORD
- FIXEDWIDTH
- IGNOREBLANKLINES
- NULL AS
- READRATIO
- REMOVEQUOTES

Weitere Informationen finden Sie unter COPY von JSON-Format. Weitere Informationen zu JSON-Datenstrukturen finden Sie unter www.json.org.

JSONPaths-Datei

Wenn Sie Daten aus Quelldaten im JSON-Format oder aus Avro-Quelldaten laden, weist COPY standardmäßig die Datenelemente auf der ersten Ebene in den Quelldaten den Spalten in der Zieltabelle zu. Dazu wird jeder Name bzw. Objektschlüssel in einem Name-Wert-Paar mit dem Namen einer Spalte in der Zieltabelle abgeglichen.

Wenn Ihre Spaltennamen und Objektschlüssel nicht übereinstimmen oder wenn Sie Zuweisungen auf tieferen Ebenen in der Datenhierarchie durchführen möchten, können Sie eine JSONPaths-Datei verwenden, um JSON- oder Avro-Datenelemente explizit Spalten zuzuweisen. Die JSONPaths-Datei weist JSON-Datenelemente zu Spalten zu, indem die Spaltenreihenfolge in der Zieltabelle oder Spaltenliste abgeglichen wird.

Die JSONPaths-Datei darf nur ein einzelnes JSON-Objekt enthalten (kein Array). Das JSON-Objekt ist ein Name-Wert-Paar. Der Objektschlüssel (der Name im Name-Wert-Paar) muss

"jsonpaths" sein. Der Wert im Name-Wert-Paar ist ein Array aus JSONPath-Ausdrücken. Jeder JSONPath-Ausdruck referenziert ein einzelnes Element in der JSON-Datenhierarchie oder im Avro-Schema, ähnlich wie ein XPath-Ausdruck Elemente in einem XML-Dokument referenziert. Weitere Informationen finden Sie unter JSONPath-Ausdrücke.

Um eine JSONPaths-Datei zu verwenden, fügen Sie dem Befehl COPY das Schlüsselwort JSON oder AVRO hinzu. Geben Sie den Namen des S3 Buckets und den Objektpfad der JSONPaths-Datei in folgendem Format an.

```
COPY tablename
FROM 'data_source'
CREDENTIALS 'credentials-args'
FORMAT AS { AVRO | JSON } 's3://jsonpaths_file';
```

Der s3://jsonpaths_file-Wert muss ein Amazon S3-Objektschlüssel sein, der explizit auf eine einzelne Datei verweist, z. B. 's3://mybucket/jsonpaths.txt'. Es darf kein Schlüsselpräfix sein.

Bei Ladevorgängen aus Amazon S3 hat die von jsonpaths_file angegebene Datei in einigen Fällen dasselbe Präfix wie der von copy_from_s3_objectpath angegebene Pfad für die Datendateien. In diesem Fall liest COPY die JSONPaths-Datei als Datendatei und gibt Fehler zurück. Angenommen, Ihre Datendateien verwenden den Objektpfad s3://mybucket/my_data.json und Ihre JSONPaths-Datei ist s3://mybucket/my_data.jsonpaths. In diesem Fall versucht COPY, my_data.jsonpaths als Datendatei zu laden.

Wenn der Schlüsselname eine andere Zeichenfolge als "jsonpaths" ist, gibt der COPY-Befehl keinen Fehler zurück, sondern ignoriert jsonpaths_file und verwendet stattdessen das Argument 'auto'.

Wenn eine der folgenden Bedingungen zutrifft, schlägt der COPY-Befehl fehl:

- Der JSON-Code ist schlecht geformt.
- Es gibt mehr als ein JSON-Objekt.
- Außerhalb des Objekts sind Zeichen (außer Leerzeichen) vorhanden.
- Ein Array-Element ist eine leere Zeichenfolge oder keine Zeichenfolge.

MAXERROR gilt für die JSONPaths-Datei nicht.

Die JSONPaths-Datei darf nicht verschlüsselt sein, auch dann nicht, wenn die Option ENCRYPTED angegeben ist.

Weitere Informationen finden Sie unter COPY von JSON-Format.

JSONPath-Ausdrücke

Die JSONPaths-Datei verwendet JSONPaths-Ausdrücke, um Datenfelder zu Zielspalten zuzuweisen. Jeder JSONPath-Ausdruck entspricht einer einzelnen Spalte in der Amazon-Redshift-Zieltabelle. Die Reihenfolge der JSONPath-Array-Elemente muss der Reihenfolge der Spalten in der Zieltabelle oder Spaltenliste entsprechen, wenn eine Spaltenliste verwendet wird.

Doppelte Anführungszeichen sind sowohl für Feldnamen als auch für Werte erforderlich. Es muss sich um normale Anführungszeichen (0x22) handeln. Es dürfen keine schrägen oder „smarten“ Anführungszeichen sein.

Wenn ein Objektelement, das von einem JSONPath-Ausdruck referenziert wird, in den JSON-Daten nicht gefunden wird, versucht COPY, einen NULL-Wert zu laden. Wenn das referenzierte Objekt schlecht geformt ist, gibt COPY einen Ladefehler zurück.

Wenn ein Array-Element, das von einem JSONPath-Ausdruck referenziert wird, in den JSON- oder Avro-Daten nicht gefunden wird, schlägt COPY mit dem folgenden Fehler fehl: `Invalid JSONPath format: Not an array or index out of range.` Entfernen Sie alle Array-Elemente aus dem JSONPaths-Ausdruck, die in den Quelldaten nicht vorhanden sind, und überprüfen Sie, ob die Arrays in den Quelldaten wohl geformt sind.

Die JSONPath-Ausdrücke können eine Klammer- oder eine Punktnotierung verwenden. Sie können die Notierungen jedoch nicht mischen. Im folgenden Beispiel werden JSONPath-Ausdrücke gezeigt, die eine Klammernotierung verwenden.

```
{
    "jsonpaths": [
        "$['venuename']",
        "$['venuecity']",
        "$['venuestate']",
        "$['venueseats']"
    ]
}
```

Im folgenden Beispiel werden JSONPath-Ausdrücke gezeigt, die eine Punktnotierung verwenden.

```
{
    "jsonpaths": [
        "$.venuename",
        "$.venuecity",
        "$.venuestate",
        "$.venueseats"
    ]
}
```

Im Kontext der COPY-Syntax von Amazon Redshift muss ein JSONPath-Ausdruck den expliziten Pfad zu einem einzelnen Namenselement in einer hierarchischen JSON- oder Avro-Datenstruktur angeben. Amazon Redshift unterstützt keine JSONPath-Elemente, wie z. B. Platzhalterzeichen oder Filterausdrücke, die zu einem mehrdeutigen Pfad oder mehreren Namenselementen führen könnten.

Weitere Informationen finden Sie unter COPY von JSON-Format.

Verwenden von JSONPaths mit Avro-Daten

Im folgenden Beispiel wird ein Avro-Schema mit mehreren Ebenen gezeigt.

```
{
    "name": "person",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "guid", "type": "string"},
        {"name": "isActive", "type": "boolean"},
        {"name": "age", "type": "int"},
        {"name": "name", "type": "string"},
        {"name": "address", "type": "string"},
        {"name": "latitude", "type": "double"},
        {"name": "longitude", "type": "double"},
        {
            "name": "tags",
            "type": {
                        "type" : "array",
                        "name" : "inner_tags",
                        "items" : "string"
                    }
        },
        {
            "name": "friends",
```

```
            "type": {
                        "type" : "array",
                        "name" : "inner_friends",
                        "items" : {
                                    "name" : "friends_record",
                                    "type" : "record",
                                    "fields" : [
                                                {"name" : "id", "type" : "int"},
                                                {"name" : "name", "type" : "string"}
                                               ]
                                   }
                    }
        },
        {"name": "randomArrayItem", "type": "string"}
    ]
}
```

Das folgende Beispiel zeigt eine JsonPaths-Datei, die AvroPath Ausdrücke verwendet, um auf das vorherige Schema zu verweisen.

```
{
    "jsonpaths": [
        "$.id",
        "$.guid",
        "$.address",
        "$.friends[0].id"
    ]
}
```

Das JSONPaths-Beispiel enthält die folgenden Elemente:

jsonpaths

Der Name des JSON-Objekts, das die Ausdrücke enthält. AvroPath

[…]

Eckige Klammern schließen das JSON-Array ein, das die Pfadelemente enthält.

$

Das Dollarzeichen bezieht sich auf das Stammelement im Avro-Schema, das das `"fields"`-Array ist.

"$.id",

Das Ziel des AvroPath Ausdrucks. In diesem Fall ist das Ziel das Element im `"fields"`-Array mit dem Namen `"id"`. Die Ausdrücke werden durch Kommas getrennt.

"$.friends[0].id"

Eckige Klammern zeigen einen Array-Index an. JSONPath-Ausdrücke verwenden eine nullbasierte Indizierung. Daher referenziert dieser Ausdruck das erste Element im `"friends"`-Array mit dem Namen `"id"`.

Die Avro-Schemasyntax erfordert die Verwendung von internen Feldern, um die Struktur der Datensatz- und Array-Datentypen zu definieren. Die inneren Felder werden von den AvroPath Ausdrücken ignoriert. Beispielsweise definiert das Feld `"friends"` ein Array namens `"inner_friends"`, das wiederum einen Datensatz namens `"friends_record"` definiert. Der AvroPath Ausdruck, der auf das Feld verweist, `"id"` kann die zusätzlichen Felder ignorieren, um direkt auf das Zielfeld zu verweisen. Die folgenden AvroPath Ausdrücke verweisen auf die beiden Felder, die zum `"friends"` Array gehören.

```
"$.friends[0].id"
"$.friends[0].name"
```

Spaltendatenformat-Parameter

Neben den Standarddatenformaten unterstützt COPY die folgenden spaltenbasierten Datenformate für COPY aus Amazon S3. COPY aus einem Spaltenformat wird mit bestimmten Einschränkungen unterstützt. Weitere Informationen finden Sie unter COPY aus spaltenbasierten Datenformaten.

ORC

Lädt die Daten aus einer Datei im Optimized Row Columnar (ORC)-Format.

PARQUET

Lädt die Daten aus einer Datei im Parquet-Format.

Dateikomprimierungsparameter

Sie können Daten aus komprimierten Datendateien laden, indem Sie die folgenden Parameter angeben.

Dateikomprimierungsparameter

BZIP2

Ein Wert, der angibt, dass die Eingabedatei oder die Eingabedateien das komprimierte bzip2-Format (.bz2-Dateien) aufweist/aufweisen. Die COPY-Operation liest jede komprimierte Datei und entkomprimiert die Daten während des Ladens.

GZIP

Ein Wert, der angibt, dass die Eingabedatei oder die Eingabedateien das komprimierte gzip-Format (.gz-Dateien) aufweist/aufweisen. Die COPY-Operation liest jede komprimierte Datei und entkomprimiert die Daten während des Ladens.

LZOP

Ein Wert, der angibt, dass die Eingabedatei oder die Eingabedateien das komprimierte lzop-Format (.lzo-Dateien) aufweist/aufweisen. Die COPY-Operation liest jede komprimierte Datei und entkomprimiert die Daten während des Ladens.

> **Note**
>
> COPY unterstützt keine Dateien, die mittels der lzop-Option --filter komprimiert wurden.

ZSTD

Ein Wert, der angibt, dass die Eingabedatei oder die Eingabedateien das komprimierte Zstandard-Format (.zst-Dateien) aufweist/aufweisen. Die COPY-Operation liest jede komprimierte Datei und entkomprimiert die Daten während des Ladens. Weitere Informationen finden Sie unter ZSTD.

> **Note**
>
> ZSTD wird nur für COPY aus Amazon S3 unterstützt.

Datenkonvertierungsparameter

Wenn COPY die Tabelle lädt, versucht der Befehl implizit, die Zeichenfolgen in den Quelldaten in den Datentyp der Zielspalte zu konvertieren. Wenn Sie eine Konvertierung angeben müssen, die sich vom Standardverhalten unterscheidet, oder wenn die Standardkonvertierung zu Fehlern führt,

können Sie Datenkonvertierungen verwalten, indem Sie die folgenden Parameter angeben. Weitere Informationen zur Syntax dieser Parameter finden Sie unter COPY-Syntax.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- BLANKSASNULL
- DATEFORMAT
- EMPTYASNULL
- ENCODING
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL AS
- REMOVEQUOTES
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

Datenkonvertierungsparameter

ACCEPTANYDATE

Ermöglicht das Laden jedes Datumsformats einschließlich ungültiger Formate wie `00/00/00 00:00:00`, ohne dass ein Fehler generiert wird. Dieser Parameter gilt nur für die Spalten TIMESTAMP und DATE. Verwenden Sie ACCEPTANYDATE stets mit dem Parameter DATEFORMAT. Wenn das Datumsformat für die Daten nicht der Spezifikation DATEFORMAT entspricht, fügt Amazon Redshift einen NULL-Wert in das betreffende Feld ein.

ACCEPTINVCHARS [AS] ['replacement_char']

Ermöglicht das Laden von Daten in VARCHAR-Spalten, auch wenn die Daten ungültige UTF-8-Zeichen enthalten. Wenn ACCEPTINVCHARS angegeben ist, ersetzt COPY jedes ungültige

UTF-8-Zeichen durch eine Zeichenfolge mit gleicher Länge, die aus dem Zeichen besteht, das von replacement_char angegeben wird. Wenn das Ersetzungszeichen beispielsweise ^ ist, wird ein ungültiges Zeichen mit drei Bytes durch ^^^ ersetzt.

Das Ersetzungszeichen kann aus jedem ASCII-Zeichen außer NULL bestehen. Der Standardwert ist das Fragezeichen (?). Informationen zu ungültigen UTF-8-Zeichen finden Sie unter Fehler beim Laden von Multibyte-Zeichen.

COPY gibt die Anzahl der Zeilen mit ungültigen UTF-8-Zeichen zurück und fügt der Systemtabelle STL_REPLACEMENTS einen Eintrag für jede betroffene Zeile hinzu (bis zu maximal 100 Zeilen pro Knoten-Slice). Zusätzliche ungültige UTF-8-Zeichen werden ebenfalls ersetzt. Diese Ersetzungsereignisse werden jedoch nicht aufgezeichnet.

Wenn ACCEPTINVCHARS nicht angegeben ist, gibt COPY einen Fehler zurück, wenn der Befehl auf ein ungültiges UTF-8-Zeichen trifft.

ACCEPTINVCHARS ist nur für VARCHAR-Spalten gültig.

BLANKSASNULL

Lädt leere Felder, die nur aus Leerzeichen bestehen, als NULL. Diese Option gilt nur für CHAR- und VARCHAR-Spalten. Leere Felder für andere Datentypen, beispielsweise INT, werden stets mit NULL geladen. Beispielsweise wird eine Zeichenfolge, die drei aufeinanderfolgende Leerzeichen (und keine anderen Zeichen) enthält, als NULL geladen. Das Standardverhalten (ohne Angabe dieser Option) besteht im Laden der Leerzeichen wie vorhanden.

DATEFORMAT [AS] {'dateformat_string' | 'auto' }

Wenn kein DATEFORMAT angegeben ist, ist das Standardformat `'YYYY-MM-DD'`. Ein alternatives gültiges Format ist beispielsweise `'MM-DD-YYYY'`.

Wenn der COPY-Befehl das Format Ihrer Datums- oder Zeitwerte nicht erkennt oder Ihre Datums- oder Zeitwerte unterschiedliche Formate verwenden, verwenden Sie das Argument `'auto'` mit dem Parameter DATEFORMAT oder TIMEFORMAT. Das Argument `'auto'` erkennt verschiedene Formate, die bei der Verwendung einer DATEFORMAT- und TIMEFORMAT-Zeichenfolge nicht unterstützt werden. Das Schlüsselwort `'auto'` unterscheidet zwischen Groß- und Kleinschreibung. Weitere Informationen finden Sie unter Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT.

Das Datumsformat kann Zeitinformationen (Stunde, Minuten, Sekunden) enthalten. Diese Informationen werden jedoch ignoriert. Das Schlüsselwort AS ist optional. Weitere Informationen finden Sie unter DATEFORMAT- und TIMEFORMAT-Zeichenfolgen.

EMPTYASNULL

Zeigt an, dass Amazon Redshift leere CHAR- und VARCHAR-Felder als NULL laden soll. Leere Felder für andere Datentypen, beispielsweise INT, werden stets mit NULL geladen. Leere Felder treten auf, wenn Daten zwei aufeinanderfolgende Trennzeichen enthalten, ohne dass sich zwischen den Trennzeichen Zeichen befinden. EMPTYASNULL und NULL AS '' (leere Zeichenfolge) führen zum selben Verhalten.

ENCODING [AS] file_encoding

Gibt den Kodierungstyp der Daten an, die geladen werden. Der COPY-Befehl konvertiert während des Ladens die Daten aus der angegebenen Kodierung in UTF-8.

Gültige Werte für file_encoding sind folgende:

- `UTF8`
- `UTF16`
- `UTF16LE`
- `UTF16BE`

Der Standardwert ist `UTF8`.

Quelldateinamen müssen UTF-8-Kodierung verwenden.

Die folgenden Dateien müssen UTF-8-Kodierung verwenden, auch wenn für die Daten, die geladen werden, eine andere Kodierung angegeben ist:

- Manifestdateien
- JSONPaths-Dateien

Die mit den folgenden Parametern bereitgestellten Argumentzeichenfolgen müssen UTF-8 verwenden:

- FIXEDWIDTH 'fixedwidth_spec'
- ACCEPTINVCHARS 'replacement_char'
- DATEFORMAT 'dateformat_string'
- TIMEFORMAT 'timeformat_string'
- NULL AS 'null_string'

Datendateien mit fester Breite müssen UTF-8-Kodierung verwenden. Die Feldbreiten basieren auf der Anzahl der Zeichen, nicht der Bytes.

Alle geladenen Daten müssen die angegebene Kodierung verwenden. Wenn COPY auf eine andere Kodierung trifft, werden die Datei übersprungen und ein Fehler zurückgegeben.

Wenn Sie UTF16 angeben, müssen Ihre Daten eine Byte-Reihenfolgenmarkierung (Byte Order Mark, BOM) besitzen. Wenn Sie wissen, ob Ihre UTF-16-Daten little-endian (LE) oder big-endian (BE) sind, können Sie unabhängig vom Vorhandensein einer BOM UTF16LE oder UTF16BE verwenden.

ESCAPE

Wenn dieser Parameter angegeben ist, wird das Backslash-Zeichen (\) in Eingabedaten als Escape-Zeichen behandelt. Das Zeichen, das dem Backslash-Zeichen folgt, wird als Teil des aktuellen Spaltenwerts in die Tabelle geladen, auch wenn es sich um ein Zeichen handelt, das normalerweise einem speziellen Zweck dient. Sie können diesen Parameter beispielsweise verwenden, um das Trennzeichen, ein Fragezeichen, ein eingebettetes Zeichen für neue Zeilen oder das Escape-Zeichen selbst mit einer Escape-Markierung zu markieren, wenn eines dieser Zeichen legitimer Teil eines Spaltenwerts ist.

Wenn Sie den Parameter ESCAPE in Kombination mit dem Parameter REMOVEQUOTES angeben, können Sie Anführungszeichen (' oder ") mit Escape-Zeichen markieren und beibehalten, die ansonsten möglicherweise entfernt würden. Die standardmäßige Null-Zeichenfolge \N funktioniert wie vorhanden, kann jedoch in den Eingabedaten ebenfalls mit einem Escape-Zeichen markiert werden: \\N. Solange sie keine alternative Null-Zeichenfolge mit dem Parameter NULL AS angeben, führen \N und \\N zu denselben Ergebnissen.

> **Note**
>
> Das Steuerzeichen 0x00 (NUL) kann nicht mit einem Escape-Zeichen markiert werden und sollte aus den Eingabedaten entfernt werden oder konvertiert werden. Dieses Zeichen wird als Markierung für das Ende von Datensätzen (End of Record, EOR) verwendet, was dazu führt, dass der Rest des Datensatzes abgeschnitten wird.

Sie können den Parameter ESCAPE nicht bei FIXEDWIDTH-Ladevorgängen verwenden. Darüber hinaus können Sie das Escape-Zeichen selbst nicht angeben; das Escape Zeichen ist stets das Backslash-Zeichen. Außerdem müssen Sie sicherstellen, dass die Eingabedaten das Escape-Zeichen an den richtigen Stellen enthalten.

Im Folgenden finden Sie einige Beispiele für Eingabedaten und die entsprechenden geladenen Daten, wenn der Parameter ESCAPE angegeben ist. Das Ergebnis für Zeile 4 geht davon aus,

dass der Parameter REMOVEQUOTES ebenfalls angegeben ist. Die Eingabedaten bestehen aus zwei Feldern, die durch das Pipe-Zeichen getrennt sind:

```
1|The quick brown fox\[newline]
jumped over the lazy dog.
2| A\\B\\C
3| A \| B \| C
4| 'A Midsummer Night\'s Dream'
```

Die Daten, die in Spalte 2 geladen wurden, sehen wie folgt aus:

```
The quick brown fox
jumped over the lazy dog.
A\B\C
A|B|C
A Midsummer Night's Dream
```

> **Note**
>
> Die Anwendung des Escape-Zeichens auf die Eingabedaten, die geladen werden sollen, liegt in der Verantwortung der Benutzer. Eine Ausnahme für diese Anforderung liegt dann vor, wenn Sie Daten erneut laden, die zuvor mit dem Parameter ESCAPE entladen wurden. In diesem Fall enthalten die Daten die notwendigen Escape-Zeichen bereits.

Der Parameter ESCAPE interpretiert keine Oktal-, Hex-, Unicode- oder andere Escape-Sequenznotierungen. Wenn Ihre Quelldaten beispielsweise den oktalen Zeilen-Feed-Wert (`\012`) enthalten und Sie versuchen, diese Daten mit dem Parameter ESCAPE zu laden, lädt Amazon Redshift den Wert `012` in die Tabelle und interpretiert diesen Wert nicht als Zeilen-Feed, der mit einem Escape-Zeichen markiert wurde.

Um in Daten, die aus Microsoft Windows-Plattformen stammen, das Zeichen für neue Zeilen mit einem Escape-Zeichen zu markieren, müssen Sie möglicherweise zwei Escape-Zeichen verwenden: ein Zeichen für die Zeilenumschaltung und ein Zeichen für den Zeilen-Feed. Alternativ können Sie die Zeilenumschaltungen vor dem Laden der Datei entfernen (beispielsweise durch Verwendung des Hilfsprogramms dos2unix).

EXPLICIT_IDS

Verwenden Sie EXPLICIT_IDS für Tabellen, die IDENTITY-Spalten besitzen, wenn Sie die automatisch generierten Werte durch explizite Werte aus den Quelldatendateien für die Tabellen überschreiben möchten. Wenn der Befehl eine Spaltenliste enthält, muss diese Liste die IDENTITY-Spalten enthalten, um diesen Parameter verwenden zu können. Das Datenformat für EXPLICIT_IDS-Werte muss mit dem IDENTITY-Format übereinstimmen, das von der Definition CREATE TABLE angegeben wird.

Nachdem Sie einen COPY-Befehl mit der EXPLICIT_IDS-Option für eine Tabelle ausgeführt haben, überprüft Amazon Redshift die Eindeutigkeit von IDENTITY-Spalten in der Tabelle nicht.

Wenn eine Spalte mit GENERATED BY DEFAULT AS IDENTITY definiert ist, kann sie kopiert werden. Die Werte werden mit den von Ihnen angegebenen Werten generiert oder aktualisiert. Die Option EXPLICIT_IDS ist nicht erforderlich. COPY aktualisiert nicht die Identity High Watermark.

Ein Beispiel für einen COPY-Befehl mit EXPLICIT_IDS finden Sie unter Laden von VENUE mit expliziten Werte für eine IDENTITY-Spalte.

FILLRECORD

Ermöglicht das Laden von Datendateien, wenn am Ende einiger Datensätze angrenzende Spalten fehlen. Die fehlenden Spalten werden als NULL-Werte geladen. Wenn die fehlende Spalte eine VARCHAR-Spalte ist und es sich um ein Text- oder CSV-Format handelt, werden Zeichenfolgen mit null Länge geladen (anstelle von NULL-Werten). Um NULL-Werte von Text- oder CSV-Formaten in VARCHAR-Spalten zu laden, müssen Sie das EMPTYASNULL-Stichwort festlegen. Die NULL-Ersetzung funktioniert nur, wenn die Spaltendefinition NULL-Werte zulässt.

Wenn die Tabellendefinition beispielsweise vier nullwertfähige CHAR-Spalten enthält und ein Datensatz die Werte `apple, orange, banana, mango` enthält, könnte der COPY-Befehl einen Datensatz laden und ausfüllen, der nur die Werte `apple, orange` enthält. Die fehlenden CHAR-Werte würden als NULL-Werte geladen.

IGNOREBLANKLINES

Ignoriert leere Zeilen, die nur einen Zeilen-Feed in einer Datendatei enthalten, und versucht nicht, sie zu laden.

IGNOREHEADER [AS] number_rows

Behandelt die angegebene number_rows als Dateiheader und lädt sie nicht. Sie verwenden IGNOREHEADER, um in einem parallelen Ladevorgang Dateiheader in allen Dateien zu überspringen.

NULL AS 'null_string'

Lädt Felder, die mit null_string als NULL übereinstimmen, wobei null_string jede Zeichenfolge sein kann. Wenn Ihre Daten einen Null-Terminator enthalten, auch als NUL (UTF-8 0000) oder Binär-Null (0x000) bezeichnet, behandelt COPY diesen als ein beliebiges anderes Zeichen. Zum Beispiel wird ein Datensatz, der '1' || NUL || '2' enthält, als Zeichenkette der Länge 3 Bytes kopiert. Wenn ein Feld nur NUL enthält, können Sie NULL AS verwenden, um den Nullterminator durch NULL zu ersetzen, indem Sie `'\0'` oder `'\000'` angeben, z. B. `NULL AS '\0'` oder `NULL AS '\000'`. Wenn ein Feld eine Zeichenfolge enthält, die mit NUL endet, und NULL AS angegeben ist, wird die Zeichenfolge mit NUL am Ende eingefügt. Verwenden Sie für den null_string-Wert nicht '\n' (Zeilenumbruch). Amazon Redshift reserviert '\n' für die Verwendung als Zeilentrennzeichen. Der Standardwert für null_string ist `'\N'`.

> **Note**
>
> Wenn Sie versuchen, Null-Werte in eine Spalte zu laden, die als NOT NULL definiert ist, schlägt der Befehl COPY fehl.

REMOVEQUOTES

Entfernt in den eingehenden Daten Anführungszeichen, die Zeichenfolgen umschließen. Alle Zeichen innerhalb der Anführungszeichen einschließlich Trennzeichen bleiben bewahrt. Wenn eine Zeichenfolge mit einem einfachen oder doppelten Anführungszeichen beginnt, es jedoch kein entsprechendes schließendes Anführungszeichen gibt, lädt der COPY-Befehl diese Zeile nicht und gibt einen Fehler zurück. In der folgenden Tabelle werden einige einfache Beispiele für Zeichenfolgen mit Anführungszeichen und die entsprechenden geladenen Werte gezeigt.

Eingabezeichenfolge	Geladener Wert mit Option REMOVEQUOTES
"The delimiter is a pipe (|) character"	Das Trennzeichen ist ein Pipe-Zeichen (|)

Eingabezeichenfolge	Geladener Wert mit Option REMOVEQUOTES
'Schwarz'	Schwarz
"Weiß"	Weiß
Blau'	Blau'
Blau'	Wert nicht geladen: Fehlerbedingung
"Blue	Wert nicht geladen: Fehlerbedingung
' 'Black' ' '	' 'Schwarz' '
' '	<Leerzeichen>

ROUNDEC

Rundet numerische Werte auf, wenn die Skala des Eingabewerts größer als die Skala der Spalte ist. Standardmäßig schneidet COPY Werte ab, wenn notwendig, um sie an die Spaltenskala anzupassen. Wenn beispielsweise der Wert `20.259` in eine DECIMAL(8,2)-Spalte geladen wird, schneidet COPY den Wert standardmäßig auf `20.25` ab. Wenn ROUNDEC angegeben ist, rundet COPY den Wert auf auf `20.26`. Der INSERT-Befehl rundet Werte stets, wenn notwendig, um sie an die Spaltenskala anzupassen. Daher weist ein COPY-Befehl mit dem Parameter ROUNDEC dasselbe Verhalten wie ein INSERT-Befehl auf.

TIMEFORMAT [AS] {'timeformat_string' | 'auto' | 'epochsecs' | 'epochmillisecs' }

Gibt das Zeitformat an. Wenn TIMEFORMAT nicht angegeben wird, ist das Standardformat `YYYY-MM-DD HH:MI:SS` für TIMESTAMP-Spalten oder `YYYY-MM-DD HH:MI:SSOF` für TIMESTAMPTZ-Spalten, wobei `OF` der Unterschied zur Coordinated Universal Time (UTC) ist. Sie können in der Zeitformatzeichenfolge keinen Zeitzonenspezifikator verwenden. Um TIMESTAMPTZ-Daten zu laden, die sich in einem anderen Format als dem Standardformat befinden, geben Sie 'auto' an. Weitere Informationen finden Sie unter Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT. Weitere Informationen zu timeformat_string finden Sie unter DATEFORMAT- und TIMEFORMAT-Zeichenfolgen.

Das Argument `'auto'` erkennt verschiedene Formate, die bei der Verwendung einer DATEFORMAT- und TIMEFORMAT-Zeichenfolge nicht unterstützt werden. Wenn der COPY-

Befehl das Format Ihrer Datums- oder Zeitwerte nicht erkennt oder Ihre Datums- oder Zeitwerte unterschiedliche verwenden, verwenden Sie das Argument `'auto'` mit dem Parameter DATEFORMAT oder TIMEFORMAT. Weitere Informationen finden Sie unter Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT.

Wenn Ihre Quelldaten als Epochenzeit dargestellt werden, d. h. als Zahl der Sekunden oder Millisekunden seit dem 1. Januar 1970, 00:00:00 UTC, geben Sie `'epochsecs'` oder `'epochmillisecs'` an.

Die Schlüsselwörter `'auto'`, `'epochsecs'` und `'epochmillisecs'` unterscheiden zwischen Groß- und Kleinschreibung.

Das Schlüsselwort AS ist optional.

TRIMBLANKS

Entfernt Leerzeichen am Ende von VARCHAR-Zeichenfolgen. Dieser Parameter gilt nur für die Spalten mit dem Datentyp VARCHAR.

TRUNCATECOLUMNS

Schneidet Daten in Spalten auf die entsprechende Zahl von Zeichen ab, sodass sie der Spaltenspezifikation entsprechen. Gilt nur für Spalten mit dem Datentyp VARCHAR und CHAR und Zeilen mit einer Größe von 4 MB oder weniger.

Datenladeoperationen

Verwalten Sie das Standardverhalten der Ladeoperation, um Fehler zu beheben oder die Ladezeiten zu reduzieren, indem Sie die folgenden Parameter angeben.

- COMPROWS
- COMPUPDATE
- IGNOREALLERRORS
- MAXERROR
- NOLOAD
- STATUPDATE

Parameter

COMPROWS numrows

Gibt die Anzahl der Zeilen an, die als Stichprobengröße für Kompressionsanalysen verwendet werden soll. Die Analyse wird für Zeilen aus jedem Daten-Slice ausgeführt. Wenn Sie beispielsweise `COMPROWS 1000000` (1.000.000) angeben und das System insgesamt vier Slices enthält, werden nicht mehr als 250.000 Zeilen pro Slice gelesen und analysiert.

Wenn COMPROWS nicht angegeben wird, werden standardmäßig 100.000 Zeilen pro Slice als Stichprobe analysiert. Werte für COMPROWS, die niedriger als der Standardwert von 100.000 Zeilen pro Slice sind, werden automatisch auf den Standardwert aktualisiert. Es findet jedoch keine automatische Kompression statt, wenn die Menge der geladenen Daten nicht für eine relevante Stichrobe reicht.

Wenn der Wert für COMPROWS größer als die Anzahl der Zeilen in der Eingabedatei ist, wird der Befehl COPY dennoch fortgesetzt und die Kompressionsanalyse wird für alle verfügbaren Zeilen ausgeführt. Der akzeptierte Bereich für dieses Argument ist eine Zahl zwischen 1000 und 2147483647 (2,147,483,647).

COMPUPDATE [PRESET | { ON | TRUE } | { OFF | FALSE }],

Steuert, ob während einer COPY-Operation automatisch komprimierende Codierungen angewendet werden.

Während COMPUPDATE den Wert PRESET hat, wählt der COPY-Befehl die Komprimierungscodierung für jede Spalte, wenn die Zieltabelle leer ist. Dies gilt auch, wenn die Spalten bereits eine andere Codierung als RAW aufweisen. Aktuell angegebene Spaltenkodierungen können ersetzt werden. Die Codierung jeder Spalte basiert auf dem Datentyp der Spalte. Es werden keine Stichproben der Daten genommen. Amazon Redshift weist die Komprimierungskodierung automatisch wie folgt zu:

- Spalten, die als Sortierschlüssel definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als die Datentypen BOOLEAN, REAL oder DOUBLE PRECISION definiert sind, wird die RAW-Kodierung zugewiesen.
- Spalten, die als SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP oder TIMESTAMPTZ definiert sind, wird die AZ64-Komprimierung zugewiesen.
- Spalten, die als CHAR oder VARCHAR definiert sind, wird LZO-Komprimierung zugewiesen.

Wenn COMPUPDATE ausgelassen wird, wählt der COPY-Befehl die Komprimierungskodierung für die einzelnen Spalten nur dann aus, wenn die Zieltabelle leer ist und Sie keine Kodierung

(außer RAW) für die Spalten angegeben haben. Die Kodierung einer jeden Spalte wird durch Amazon Redshift bestimmt. Es werden keine Stichproben der Daten genommen.

Wenn COMPUDATE auf ON (oder TRUE) festgelegt ist oder COMPUPDATE ohne Option angegeben wird, wendet der COPY-Befehl die automatische Komprimierung an, wenn die Tabelle leer ist, auch wenn die Tabellenspalten bereits andere Kodierungen als RAW aufweisen. Aktuell angegebene Spaltenkodierungen können ersetzt werden. Die Kodierung einer jeden Spalte basiert auf der Analyse von Stichprobendaten. Weitere Informationen finden Sie unter Laden von Tabellen mit automatischer Kompression.

Wenn COMPUPDATE auf OFF (oder FALSE eingestellt ist), wird die automatische Kompression deaktiviert. Spaltenkodierungen werden nicht geändert.

Informationen zur Systemtabelle für die Analyse der Komprimierung finden Sie unter STL_ANALYZE_COMPRESSION.

IGNOREALLERRORS

Sie können diese Option angeben, wenn alle Fehler, die während des Ladevorgangs auftreten, ignoriert werden sollen.

Sie können die Option IGNOREALLERRORS nicht angeben, wenn Sie die Option MAXERROR verwenden. Sie können die Option IGNOREALLERRORS nicht für spaltenbasierte Formate wie ORC und Parquet angeben.

MAXERROR [AS] error_count

Wenn der Ladevorgang error_count Fehler oder mehr zurückgibt, schlägt der Ladevorgang fehl. Wenn der Ladevorgang weniger Fehler zurückgibt, wird er fortgesetzt und gibt eine INFO-Meldung zurück, die die Anzahl der Zeilen angibt, die nicht geladen werden konnten. Sie verwenden diesen Parameter, um die Fortsetzung von Ladevorgängen zuzulassen, wenn bestimmte Zeilen aufgrund von Formatierungsfehlern oder aufgrund anderer Inkonsistenzen in den Daten nicht in die Tabelle geladen werden können.

Legen Sie diesen Wert auf `0` oder `1` fest, wenn der Ladevorgang fehlschlagen soll, sobald der erste Fehler auftritt. Das Schlüsselwort AS ist optional. Der Standardwert für MAXERROR ist `0` und das Limit ist `100000`.

Aufgrund der parallelen Struktur von Amazon Redshift kann die tatsächliche Zahl der gemeldeten Fehler größer als der für MAXERROR angegebene Wert sein. Wenn ein Knoten im Amazon-Redshift-Cluster feststellt, dass MAXERROR ausgeführt wurde, melden alle Knoten alle Fehler, die sie festgestellt haben.

NOLOAD

Prüft die Gültigkeit der Datendatei, ohne die Daten tatsächlich zu laden. Verwenden Sie den Parameter NOLOAD, um zu prüfen, ob die Datendatei ohne Fehler geladen wird, bevor der tatsächliche Datenladevorgang ausgeführt wird. COPY mit angegebenem Parameter NOLOAD ist sehr viel schneller als das Laden der Daten, da die Dateien lediglich analysiert werden.

STATUPDATE [{ ON | TRUE } | { OFF | FALSE }]

Regelt die automatische Berechnung und die Aktualisierung der Optimierungsstatistik am Ende eines erfolgreichen COPY-Befehls. Wenn der Parameter STATUPDATE nicht verwendet wird, werden die Statistiken automatisch aktualisiert, wenn die Tabelle anfangs leer ist.

Wenn durch das Hinzufügen von Daten zu einer nicht leeren Tabelle die Größe der Tabelle erheblich verändert wird, sollten Sie die Statistik aktualisieren, indem Sie entweder einen ANALYZE-Befehl ausführen oder das Argument STATUPDATE ON verwenden.

Bei Verwendung des Arguments STATUPDATE ON (oder TRUE) werden die Statistiken automatisch aktualisiert, unabhängig davon, ob die Tabelle anfangs leer ist. Wenn STATUPDATE verwendet wird, muss der aktuelle Benutzer entweder der Tabellenbesitzer oder ein Superuser sein. Wenn STATUPDATE nicht angegeben ist, ist nur die INSERT-Berechtigung erforderlich.

Bei Angabe von STATUPDATE OFF (oder FALSE) werden die Statistiken niemals aktualisiert.

Weitere Informationen finden Sie unter Analysieren von Tabellen.

Alphabetische Liste der Parameter

In der folgenden List werden Links zur Beschreibung der einzelnen Parameter des COPY-Befehls in alphabetischer Reihenfolge bereitgestellt.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- ACCESS_KEY_ID and SECRET_ACCESS_KEY
- AVRO
- BLANKSASNULL
- BZIP2
- COMPROWS

- COMPUPDATE
- CREDENTIALS
- CSV
- DATEFORMAT
- DELIMITER
- EMPTYASNULL
- ENCODING
- ENCRYPTED
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- FIXEDWIDTH
- FORMAT
- FROM
- GZIP
- IAM_ROLE
- IGNOREALLERRORS
- IGNOREBLANKLINES
- IGNOREHEADER
- JSON
- LZOP
- MANIFEST
- MASTER_SYMMETRIC_KEY
- MAXERROR
- NOLOAD
- NULL AS
- READRATIO
- REGION
- REMOVEQUOTES

- ROUNDEC
- SESSION_TOKEN
- SHAPEFILE
- SSH
- STATUPDATE
- TIMEFORMAT
- SESSION_TOKEN
- TRIMBLANKS
- TRUNCATECOLUMNS
- ZSTD

Nutzungshinweise

Themen

- Berechtigungen für den Zugriff auf andere AWS -Ressourcen
- Verwenden von COPY mit Amazon-S3-Zugriffspunkt-Aliasen
- Laden von Multibyte-Daten aus Amazon S3
- Laden einer Spalte des Datentyps GEOMETRY oder GEOGRAPHY
- Laden des Datentyps HLLSKETCH
- Laden einer Spalte des Datentyps VARBYTE
- Fehler beim Lesen mehrerer Dateien
- COPY von JSON-Format
- COPY aus spaltenbasierten Datenformaten
- DATEFORMAT- und TIMEFORMAT-Zeichenfolgen
- Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT

Berechtigungen für den Zugriff auf andere AWS -Ressourcen

Um Daten zwischen Ihrem Cluster und einer anderen AWS Ressource wie Amazon S3, Amazon DynamoDB, Amazon EMR oder Amazon EC2 zu verschieben, muss Ihr Cluster über die Berechtigung verfügen, auf die Ressource zuzugreifen und die erforderlichen Aktionen auszuführen.

Um beispielsweise Daten aus Amazon S3 zu laden, muss COPY über LIST-Zugriff auf den Bucket und GET-Zugriff auf die Bucket-Objekte verfügen. Weitere Informationen zu den mindestens erforderlichen Berechtigungen finden Sie unter IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY.

Um die Autorisierung für den Zugriff auf die Ressource zu erhalten, muss Ihr Cluster authentifiziert werden. Sie können eine der beiden folgenden Authentifizierungsmethoden verwenden:

- Rollenbasierte Zugriffskontrolle— Für die rollenbasierte Zugriffskontrolle geben Sie eine AWS Identity and Access Management (IAM-) Rolle an, die Ihr Cluster für die Authentifizierung und Autorisierung verwendet. Um Ihre AWS Anmeldeinformationen und vertraulichen Daten zu schützen, empfehlen wir dringend, die rollenbasierte Authentifizierung zu verwenden.
- Schlüsselbasierte Zugriffssteuerung— Für die schlüsselbasierte Zugriffskontrolle geben Sie die Zugangsdaten (AWS Zugriffsschlüssel-ID und geheimer Zugriffsschlüssel) für einen Benutzer als Klartext an.

Rollenbasierte Zugriffskontrolle

Mit einer rollenbasierten Zugriffssteuerung übernimmt Ihr Cluster vorübergehend in Ihrem Namen eine IAM-Rolle. Anschließend kann Ihr Cluster auf der Basis der Autorisierungen, die der Rolle gewährt wurden, auf die erforderlichen AWS -Ressourcen zugreifen.

Das Erstellen einer IAM-Rolle ähnelt insofern dem Erteilen von Berechtigungen für einen Benutzer, als es sich um eine AWS -Identität mit Berechtigungsrichtlinien handelt, die festlegen, welche Aktionen die Identität in AWS ausführen kann und welche nicht. Eine Rolle ist jedoch nicht einem einzigen Benutzer zugeordnet, sondern kann von allen Entitäten angenommen werden, die diese Rolle benötigen. Einer Rolle sind auch keine Anmeldeinformationen (Passwort oder Zugriffsschlüssel) zugeordnet. Wenn eine Rolle mit einem Cluster verknüpft ist, werden die Zugriffsschlüssel stattdessen dynamisch erstellt und dem Cluster bereitgestellt.

Wir empfehlen die Verwendung einer rollenbasierten Zugriffskontrolle, da sie zusätzlich zum Schutz Ihrer Anmeldeinformationen eine sicherere und detailliertere Steuerung des Zugriffs auf AWS Ressourcen und vertrauliche Benutzerdaten ermöglicht. AWS

Die rollenbasierte Authentifizierung bietet folgende Vorteile:

- Sie können AWS Standard-IAM-Tools verwenden, um eine IAM-Rolle zu definieren und die Rolle mehreren Clustern zuzuordnen. Wenn Sie die Zugriffsrichtlinie für eine Rolle ändern, werden die Änderungen automatisch auf alle Cluster angewendet, die diese Rolle verwenden.

- Sie können detaillierte IAM-Richtlinien definieren, die bestimmten Clustern und Datenbankbenutzern Berechtigungen für den Zugriff auf bestimmte Ressourcen und Aktionen gewähren. AWS
- Ihr Cluster erhält temporäre Sitzungsanmeldeinformationen zur Laufzeit. Die Anmeldeinformationen werden wie benötigt aktualisiert, bis die Operation abgeschlossen ist. Wenn Sie schlüsselbasierte temporäre Anmeldeinformationen verwenden, schlägt die Operation fehl, wenn die temporären Anmeldeinformationen ablaufen, bevor die Operation abgeschlossen ist.
- Die Zugriffsschlüssel-ID und die ID des geheimen Zugriffsschlüssels werden nicht übertragen oder im SQL-Code gespeichert.

Um die rollenbasierte Zugriffssteuerung zu verwenden, müssen Sie zunächst unter Verwendung des Amazon-Redshift-Servicerollentyps eine IAM-Rolle erstellen und die Rolle anschließend Ihrem Cluster hinzufügen. Die Rolle muss mindestens die in aufgelisteten Berechtigungen besitzen IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY. Schritte zum Erstellen einer IAM-Rolle und zum Anhängen dieser Rolle an Ihren Cluster finden Sie unter Autorisieren von Amazon Redshift, in Ihrem Namen auf andere AWS Services zuzugreifen im Amazon Redshift Management Guide.

Sie können über die Amazon-Redshift-Managementkonsole, die CLI oder eine API einem Cluster eine Rolle hinzufügen oder die Rollen anzeigen, die mit einem Cluster verknüpft sind. Weitere Informationen finden Sie unter Verknüpfen einer IAM-Rolle mit einem Cluster im Amazon-Redshift-Verwaltungshandbuch.

Beim Erstellen einer IAM-Rolle gibt IAM einen Amazon-Ressourcennamen (ARN) für die Rolle zurück. Um eine IAM-Rolle anzugeben, geben Sie für den ARN der Rolle entweder den Parameter IAM_ROLE oder den Parameter CREDENTIALS an.

Angenommen, die folgende Rolle ist dem Cluster angefügt.

```
"IamRoleArn": "arn:aws:iam::0123456789012:role/MyRedshiftRole"
```

Im folgenden Beispiel für den COPY-Befehl wird der Parameter IAM_ROLE mit dem ARN im vorherigen Beispiel verwendet, um Authentifizierung und Zugriff auf Amazon S3 bereitzustellen.

```
copy customer from 's3://mybucket/mydata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Im folgenden Beispiel für den COPY-Befehl wird der Parameter CREDENTIALS verwendet, um die IAM-Rolle anzugeben.

```
copy customer from 's3://mybucket/mydata'
credentials
'aws_iam_role=arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Darüber hinaus kann ein Superuser Datenbankbenutzern und -gruppen das Privileg ASSUMEROLE gewähren, um einer Rolle Zugriff auf COPY-Operationen zu ermöglichen. Weitere Informationen finden Sie unter GRANT.

Schlüsselbasierte Zugriffssteuerung

Bei der schlüsselbasierten Zugriffskontrolle geben Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel für einen IAM-Benutzer an, der berechtigt ist, auf die Ressourcen zuzugreifen, die die Daten enthalten. AWS Sie können entweder die Parameter ACCESS_KEY_ID and SECRET_ACCESS_KEY zusammen verwenden oder den Parameter CREDENTIALS verwenden.

> **Note**
>
> Es wird nachdrücklich empfohlen, eine IAM-Rolle für die Authentifizierung zu verwenden, statt eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel im Textformat bereitzustellen. Wenn Sie sich für die schlüsselbasierte Zugriffskontrolle entscheiden, verwenden Sie niemals Ihre AWS Kontoanmeldeinformationen (Root). Erstellen Sie stets einen IAM-Benutzer und geben Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel für diesen Benutzer an. Schritte zum Erstellen eines IAM-Benutzers finden Sie unter Erstellen eines IAM-Benutzers in Ihrem AWS -Konto.

Um die Authentifizierung mittels ACCESS_KEY_ID und SECRET_ACCESS_KEY zu ermöglichen, ersetzen Sie *<access-key-id>* und *<secret-access-key>* durch die Zugriffsschlüssel-ID und den vollständigen geheimen Zugriffsschlüssel eines autorisierten Benutzers für die Parameter ACCESS_KEY_ID und SECRET_ACCESS_KEY, wie im Folgenden gezeigt.

```
ACCESS_KEY_ID '<access-key-id>'
SECRET_ACCESS_KEY '<secret-access-key>';
```

Um die Authentifizierung mittels des Parameters CREDENTIALS zu ermöglichen, ersetzen Sie *<access-key-id>* und *<secret-access-key>* durch die Zugriffsschlüssel-ID und den vollständigen geheimen Zugriffsschlüssel eines autorisierten Benutzers für die Parameter ACCESS_KEY_ID und SECRET_ACCESS_KEY, wie im Folgenden gezeigt.

```
CREDENTIALS
'aws_access_key_id=<access-key-id>;aws_secret_access_key=<secret-access-key>';
```

Der IAM-Benutzer muss mindestens die in aufgelisteten Berechtigungen besitzen IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY.

Temporäre Sicherheitsanmeldeinformationen

Wenn Sie die schlüsselbasierte Zugriffssteuerung verwenden, können Sie den Zugriff weiter einschränken, den Benutzer auf Ihre Daten haben, indem Sie temporäre Sicherheitsanmeldeinformationen verwenden. Die rollenbasierte Authentifizierung verwendet automatisch temporäre Anmeldeinformationen.

> **Note**
>
> Es wird nachdrücklich empfohlen, die role-based access control zu verwenden, statt temporäre Anmeldeinformationen zu erstellen und Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel in Form von reinem Text bereitzustellen. Die rollenbasierte Zugriffssteuerung verwendet automatisch temporäre Anmeldeinformationen.

Temporäre Sicherheitsanmeldeinformationen bieten erweiterte Sicherheit, da sie nur kurzlebig sind und nach ihrem Ablauf nicht erneut verwendet werden können. Die Zugriffsschlüssel-ID und der geheime Zugriffsschlüssel, die mit dem Token generiert werden, können nicht ohne das Token verwendet werden. Ein Benutzer, der diese temporären Sicherheitsanmeldeinformationen besitzt, kann auf Ihre Ressourcen nur solange zugreifen, bis die Anmeldeinformationen ablaufen.

Um Benutzern temporären Zugriff auf Ihre Ressourcen zu gewähren, rufen Sie AWS Security Token Service (AWS STS) API-Operationen auf. Die AWS STS API-Operationen geben temporäre Sicherheitsanmeldedaten zurück, die aus einem Sicherheitstoken, einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel bestehen. Sie geben die temporären Sicherheitsanmeldeinformationen an die Benutzer aus, die temporären Zugriff auf Ihre Ressourcen benötigen. Bei diesen Benutzern kann es sich um vorhandene IAM-Benutzer oder um Benutzer außerhalb von AWS handeln. Weitere Informationen zum Erstellen temporärer Sicherheitsanmeldeinformationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen im IAM-Benutzerhandbuch.

Sie können die ACCESS_KEY_ID and SECRET_ACCESS_KEY-Parameter zusammen mit dem Parameter SESSION_TOKEN oder dem Parameter CREDENTIALS verwenden. Sie müssen auch

die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel bereitstellen, die mit dem Token bereitgestellt wurden.

Zur Authentifizierung der Verwendung von ACCESS_KEY_ID, SECRET_ACCESS_KEY, und SESSION_TOKEN ersetzen Sie *<temporary-access-key-id>*, *<temporary-secret-access-key>* und *<temporary-token>* wie nachfolgend gezeigt.

```
ACCESS_KEY_ID '<temporary-access-key-id>'
SECRET_ACCESS_KEY '<temporary-secret-access-key>'
SESSION_TOKEN '<temporary-token>';
```

Um die Authentifizierung mittels CREDENTIALS durchzuführen, fügen Sie `session_token=`*<temporary-token>* in die Anmeldezeichenfolge ein wie im Folgenden gezeigt.

```
CREDENTIALS
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-secret-access-key>;session_token=<temporary-token>';
```

Im folgenden Beispiel wird ein COPY-Befehl mit temporären Sicherheitsanmeldeinformationen gezeigt.

```
copy table-name
from 's3://objectpath'
access_key_id '<temporary-access-key-id>'
secret_access_key '<temporary-secret-access-key>'
session_token '<temporary-token>';
```

Im folgenden Beispiel wird die Tabelle LISTING unter Verwendung temporärer Anmeldeinformationen und Dateiverschlüsselung geladen.

```
copy listing
from 's3://mybucket/data/listings_pipe.txt'
access_key_id '<temporary-access-key-id>'
secret_access_key '<temporary-secret-access-key>'
session_token '<temporary-token>'
master_symmetric_key '<root-key>'
encrypted;
```

Im folgenden Beispiel wird die Tabelle LISTING unter Verwendung des Parameters CREDENTIALS mit temporären Anmeldeinformationen und Dateiverschlüsselung geladen.

```
copy listing
from 's3://mybucket/data/listings_pipe.txt'
credentials
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-secret-access-key>;session_token=<temporary-token>;master_symmetric_key=<root-key>'
encrypted;
```

> **Important**
>
> Die temporären Sicherheitsanmeldeinformationen müssen für die gesamte Dauer der COPY- oder UNLOAD-Operation gültig sein. Wenn die temporären Sicherheitsanmeldeinformationen während der Operation ablaufen, schlägt der Befehl fehl und für die Transaktion wird ein Rollback ausgeführt. Wenn die temporären Sicherheitsanmeldeinformationen beispielsweise nach 15 Minuten ablaufen und die COPY-Operation eine Stunde benötigt, schlägt die Operation fehl, bevor sie abgeschlossen ist. Wenn Sie den rollenbasierten Zugriff verwenden, werden die temporären Sicherheitsanmeldeinformationen automatisch aktualisiert, bis die Operation abgeschlossen ist.

IAM-Berechtigungen für COPY, UNLOAD und CREATE LIBRARY

Die IAM-Rolle oder der Benutzer, die bzw. der durch den Parameter CREDENTIALS referenziert wird, muss mindestens die folgenden Berechtigungen besitzen:

- Für COPY aus Amazon S3 die LIST-Berechtigung für Amazon-S3-Buckets und die GET-Berechtigung für Amazon-S3-Objekte, die geladen werden, sowie die Manifestdatei, falls eine solche verwendet wird.
- Für COPY aus Amazon S3, Amazon EMR und Remote-Hosts (SSH) mit Daten im JSON-Format, LIST-Berechtigung und GET-Berechtigung für die JSONPaths-Datei in Amazon S3, wenn verwendet.
- Für COPY aus DynamoDB, SCAN- und DESCRIBE-Berechtigung für die DynamoDB-Tabelle, die geladen wird.
- Für COPY aus einem Amazon-EMR-Cluster die Berechtigung für die `ListInstances`-Aktion auf dem Amazon EMR-Cluster.
- Für UNLOAD zu Amazon S3, GET-, LIST- und PUT-Berechtigungen für den Amazon-S3-Bucket, in den die Datendateien entladen werden.

- Für CREATE LIBRARY aus Amazon S3 die LIST-Berechtigung für Amazon-S3-Buckets und die GET-Berechtigung für die zu importierenden Amazon-S3-Objekte.

> **Note**
>
> Wenn Sie bei Ausführung eines COPY-, UNLOAD- oder CREATE LIBRARY-Befehls die Fehlermeldung `S3ServiceException: Access Denied` erhalten, besitzt Ihr Cluster nicht die korrekten Zugriffsberechtigungen für Amazon S3.

Sie können IAM-Berechtigungen verwalten, indem Sie einer IAM-Rolle, die Ihrem Cluster angefügt ist, Ihrem Benutzer oder der Gruppe, zu der Ihr Benutzer gehört, eine IAM-Richtlinie anfügen. Beispielsweise gewährt die verwaltete `AmazonS3ReadOnlyAccess`-Richtlinie LIST- und GET-Berechtigungen in Bezug auf Amazon-S3-Ressourcen. Weitere Informationen zu IAM-Richtlinien finden Sie unter Verwalten von IAM-Richtlinien im IAM-Benutzerhandbuch.

Verwenden von COPY mit Amazon-S3-Zugriffspunkt-Aliasen

COPY unterstützt Amazon-S3-Zugriffspunkt-Aliase. Weitere Informationen finden Sie unter Verwenden eines Alias im Bucket-Stil für Ihren Zugriffspunkt im Amazon-Simple-Storage-Service-Benutzerhandbuch.

Laden von Multibyte-Daten aus Amazon S3

Wenn Ihre Daten Multibyte-Zeichen enthalten, die andere als ASCII-Zeichen verwenden (beispielsweise chinesische oder kyrillische Zeichen), müssen Sie die Daten in VARCHAR-Spalten laden. Der VARCHAR-Datentyp unterstützt UTF-8-Zeichen mit vier Bytes. Der CHAR-Datentyp unterstützt jedoch nur ASCII-Zeichen mit einem Byte. Sie können keine Zeichen mit fünf Bytes oder mehr in Amazon-Redshift-Tabellen laden. Weitere Informationen finden Sie unter Multibyte-Zeichen.

Laden einer Spalte des Datentyps GEOMETRY oder GEOGRAPHY

COPY in `GEOMETRY`- oder `GEOGRAPHY`-Spalten funktioniert nur aus Daten in einer zeichengetrennten Textdatei, z. B. einer CSV-Datei. Die Daten müssen in hexadezimaler Form des Well-Known-Binary-Formats (WKB oder EWKB) oder des Well-Known-Text-Formats (WKT oder EWKT) vorliegen und der maximalen Größe einer einzelnen Eingabezeile im COPY-Befehl angepasst sein. Weitere Informationen finden Sie unter COPY.

Weitere Informationen zum Laden aus einem Shapefile finden Sie unter Laden eines Shapefile in Amazon Redshift.

Weitere Informationen zum GEOMETRY- oder GEOGRAPHY-Datentyp finden Sie unter Abfrage von Geodaten in Amazon Redshift.

Laden des Datentyps HLLSKETCH

Sie können HLL-Skizzen nur im Sparse- oder Dense-Format kopieren, das von Amazon Redshift unterstützt wird. Um den Befehl COPY für HyperLogLog Skizzen zu verwenden, verwenden Sie das Base64-Format für dichte HyperLogLog Skizzen und das JSON-Format für Skizzen mit geringer Dichte. HyperLogLog Weitere Informationen finden Sie unter HyperLogLog Funktionen.

Das folgende Beispiel importiert Daten aus einer CSV-Datei in eine Tabelle mit CREATE TABLE und COPY. Im Beispiel wird zunächst die Tabelle t1 mit CREATE TABLE erstellt.

```
CREATE TABLE t1 (sketch hllsketch, a bigint);
```

Dann werden mit COPY Daten aus einer CSV-Datei in die Tabelle importiert t1.

```
COPY t1 FROM s3://DOC-EXAMPLE-BUCKET/unload/' IAM_ROLE
 'arn:aws:iam::0123456789012:role/MyRedshiftRole' NULL AS 'null' CSV;
```

Laden einer Spalte des Datentyps VARBYTE

Sie können Daten aus einer Datei im CSV-, Parquet- und ORC-Format laden. Bei Verwendung des CSV-Formats werden die Daten aus einer Datei in hexadezimaler Darstellung der VARBYTE-Daten geladen. Sie können keine VARBYTE-Daten mit der FIXEDWIDTH-Option laden. Die COPY-Optionen ADDQUOTES und REMOVEQUOTES werden nicht unterstützt. VARBYTE-Spalten können nicht als Partitionsspalten verwendet werden.

Fehler beim Lesen mehrerer Dateien

Der COPY-Befehl ist atomisch und transaktional. Mit anderen Worten, der gesamte Prozess wird als einzelne Transaktion behandelt, auch wenn der COPY-Befehl Daten aus mehreren Dateien liest. Wenn COPY beim Lesen einer Datei auf einen Fehler trifft, wird der Vorgang automatisch wiederholt, bis die Prozesszeit abläuft (siehe statement_timeout) oder wenn über einen längeren Zeitraum (zwischen 15 und 30 Minuten) keine Daten aus Amazon S3 heruntergeladen werden können. Dabei wird sichergestellt, dass jede Datei nur einmal geladen wird. Wenn der COPY-Befehl fehlschlägt, wird die gesamte Transaktion abgebrochen und es wird ein Rollback für alle Änderungen ausgeführt. Weitere Informationen zur Handhabung von Fehlern beim Laden finden Sie unter Fehlerbehebung bei Datenladevorgängen.

Wenn ein COPY-Befehl erfolgreich gestartet wurde, schlägt er nicht fehl, wenn die Sitzung beendet wird, beispielsweise, wenn die Verbindung des Clients getrennt wird. Wenn sich der COPY-Befehl jedoch innerhalb eines BEGIN … END-Transaktionsblocks befindet, der nicht abgeschlossen wird, weil die Sitzung beendet wird, wird für die gesamte Transaktion einschließlich der COPY-Operation ein Rollback ausgeführt. Weitere Informationen Transaktionen finden Sie unter BEGIN.

COPY von JSON-Format

Die JSON-Datenstruktur besteht aus einem Satz von Objekten oder Arrays. Ein JSON-Objekt beginnt und endet mit geschweiften Klammern und enthält eine nicht geordnete Sammlung von Name-Wert-Paaren. Jeder Name und jeder Wert werden durch einen Doppelpunkt getrennt und die Paare werden durch Kommas getrennt. Der Name ist eine Zeichenfolge in doppelten Anführungszeichen. Bei den Anführungszeichen muss es sich um normale Anführungszeichen (0x22) handeln. Es dürfen keine schrägen oder „smarte“ doppelte Anführungszeichen sein.

Ein JSON-Array beginnt und endet mit eckigen Klammern und enthält eine geordnete Sammlung von Werten, getrennt durch Kommas. Ein Wert kann eine Zeichenfolge in doppelten Anführungszeichen, eine Zahl, ein boolescher Wert (wahr oder falsch), null, ein JSON-Objekt oder ein Array sein.

JSON-Objekte und -Arrays können verschachtelt sein, was eine hierarchische Datenstruktur ermöglicht. Im folgenden Beispiel wird eine JSON-Datenstruktur mit zwei gültigen Objekten gezeigt.

```
{
    "id": 1006410,
    "title": "Amazon Redshift Database Developer Guide"
}
{
    "id": 100540,
    "name": "Amazon Simple Storage Service User Guide"
}
```

Im Folgenden werden dieselben Daten als zwei JSON-Arrays gezeigt.

```
[
    1006410,
    "Amazon Redshift Database Developer Guide"
]
[
    100540,
    "Amazon Simple Storage Service User Guide"
```

```
]
```

COPY-Optionen für JSON

Sie können die folgenden Optionen angeben, wenn Sie COPY mit Daten im JSON-Format verwenden:

- `'auto'` – COPY lädt automatisch Felder aus der JSON-Datei.
- `'auto ignorecase'` – COPY lädt automatisch Felder aus der JSON-Datei, die Groß-/Kleinschreibung von Feldnamen wird ignoriert.
- `s3://jsonpaths_file` – COPY verwendet eine JSONPaths-Datei, um die JSON-Quelldaten zu parsen. Eine JSONPaths-Datei ist eine Textdatei, die ein einzelnes JSON-Objekt mit dem Namen `"jsonpaths"` enthält, gepaart mit einem Array aus JSONPath-Ausdrücken. Wenn der Name eine andere Zeichenfolge als `"jsonpaths"` ist, verwendet COPY das Argument `'auto'` und nicht die JSONPaths-Datei.

Beispiele, die zeigen, wie Daten unter Verwendung von `'auto'`, `'auto ignorecase'` oder einer JSONPaths-Datei bzw. unter Verwendung von JSON-Objekten oder -Arrays geladen werden, finden Sie unter Beispiele für die COPY-Operation aus JSON.

JSONPath-Option

In der COPY-Syntax von Amazon Redshift gibt ein JSONPath-Ausdruck den expliziten Pfad zu einem einzelnen Namenselement in einer hierarchischen JSON-Datenstruktur an, wobei entweder die Klammer- oder die Punktnotation verwendet wird. Amazon Redshift unterstützt keine JSONPath-Elemente, wie z. B. Platzhalterzeichen oder Filterausdrücke, die zu einem mehrdeutigen Pfad oder mehreren Namenselementen führen könnten. Daher kann Amazon Redshift keine komplexen Datenstrukturen mit mehreren Ebenen analysieren.

Im folgenden Beispiel wird eine JSONPaths-Datei mit JSONPath-Ausdrücken gezeigt, die eine Klammernotierung verwenden. Das Dollarzeichen ($) stellt die Stammebenenstruktur dar.

```
{
    "jsonpaths": [
       "$['id']",
       "$['store']['book']['title']",
 "$['location'][0]"
    ]
}
```

Im vorherigen Beispiel referenziert `$['location'][0]` das erste Element in einem Array. JSON verwendet eine nullbasierte Array-Indizierung. Bei Array-Indizes muss es sich um positive Ganzzahlen handeln (größer als oder gleich null).

Im folgenden Beispiel wird die vorherige JSONPaths-Datei unter Verwendung einer Punktnotierung gezeigt.

```
{
    "jsonpaths": [
       "$.id",
       "$.store.book.title",
 "$.location[0]"
    ]
}
```

Sie können Klammer- und Punktnotierung im `jsonpaths`-Array nicht mischen. Eckige Klammern können sowohl in der Klammer- als auch in der Punktnotierung verwendet werden, um ein Array-Element zu referenzieren.

Bei Verwendung der Punktnotierung dürfen die JSONPath-Ausdrücke folgende Zeichen nicht enthalten:

- Einfache gerade Anführungszeichen (')
- Punkt (.)
- Eckige Klammern ([]), es sei denn, sie werden verwendet, um ein Array-Element zu referenzieren

Wenn es sich bei dem Wert in dem Name-Wert-Paar, das von einem JSONPath-Ausdruck referenziert wird, um ein Objekt oder ein Array handelt, wird das gesamte Objekt oder Array als eine Zeichenfolge geladen, einschließlich geschweifter oder eckiger Klammern. Angenommen, Ihre JSON-Daten enthalten das folgende Objekt.

```
{
    "id": 0,
    "guid": "84512477-fa49-456b-b407-581d0d851c3c",
    "isActive": true,
    "tags": [
        "nisi",
        "culpa",
        "ad",
```

```
        "amet",
        "voluptate",
        "reprehenderit",
        "veniam"
    ],
    "friends": [
        {
            "id": 0,
            "name": "Martha Rivera"
        },
        {
            "id": 1,
            "name": "Renaldo"
        }
    ]
}
```

Anschließend gibt der JSONPath-Ausdruck `$['tags']` den folgenden Wert zurück.

```
"["nisi","culpa","ad","amet","voluptate","reprehenderit","veniam"]"
```

Anschließend gibt der JSONPath-Ausdruck `$['friends'][1]` den folgenden Wert zurück.

```
"{"id": 1,"name": "Renaldo"}"
```

Jeder JSONPath-Ausdruck im `jsonpaths`-Array entspricht einer einzelnen Spalte in der Amazon-Redshift-Zieltabelle. Die Reihenfolge der `jsonpaths`-Array-Elemente muss der Reihenfolge der Spalten in der Zieltabelle oder Spaltenliste entsprechen, wenn eine Spaltenliste verwendet wird.

Beispiele, die zeigen, wie Daten unter Verwendung des Arguments `'auto'` oder einer JSONPaths-Datei bzw. unter Verwendung von JSON-Objekten oder Arrays geladen werden, finden Sie unter Beispiele für die COPY-Operation aus JSON.

Informationen zum Kopieren mehrerer JSON-Dateien finden Sie unter Verwenden eines Manifests für die Angabe von Datendateien.

Escape-Zeichen in JSON

COPY lädt `\n` als Zeichen für neue Zeilen und `\t` als Tabulatorzeichen. Um einen Backslash zu laden, muss ein Backslash als Escape-Zeichen verwendet werden (`\\`).

Angenommen, es gibt die folgenden JSON-Daten in einer Datei namens `escape.json` im Bucket `s3://mybucket/json/`.

```
{
  "backslash": "This is a backslash: \\",
  "newline": "This sentence\n is on two lines.",
  "tab": "This sentence \t contains a tab."
}
```

Führen Sie die folgenden Befehle aus, um die Tabelle ESCAPES zu erstellen und JSON zu laden.

```
create table escapes (backslash varchar(25), newline varchar(35), tab varchar(35));

copy escapes from 's3://mybucket/json/escape.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as json 'auto';
```

Führen Sie eine Abfrage für die Tabelle ESCAPES aus, um die Ergebnisse anzuzeigen.

```
select * from escapes;

       backslash        |      newline      |               tab
------------------------+-------------------+----------------------------------
 This is a backslash: \ | This sentence     | This sentence    contains a tab.
                        :  is on two lines.
(1 row)
```

Verlust der numerischen Präzision

Wenn Sie Zahlen aus Datendateien im JSON-Format in eine Spalte laden, die als numerischer Datentyp formatiert ist, geht möglicherweise die Präzision verloren. Einige Gleitpunktwerte werden in Rechnersystemen nicht genau dargestellt. Dies hat zur Folge, dass aus einer JSON-Datei kopierte Daten nicht wie von Ihnen erwartet gerundet werden. Um einen Verlust der Präzision zu vermeiden, raten wir zur Verwendung einer der folgenden Alternativen:

- Stellen Sie die Zahl als eile Zeichenfolge dar, deren Wert in doppelte Anführungszeichen gesetzt wird.
- Verwenden Sie ROUNDEC, um die Zahl zu runden, anstatt sie zu kürzen.
- Verwenden Sie anstelle von JSON- oder Avro-Dateien CSV-, durch Zeichen begrenzte oder Textdateien fester Breite.

COPY aus spaltenbasierten Datenformaten

COPY kann Daten aus Amazon S3 in folgenden spaltenbasierten Formaten laden:

- ORC
- Parquet

Beispiele für die Verwendung von COPY aus spaltenbasierten Datenformaten finden Sie unter Beispiele für COPY.

COPY unterstützt spaltenweise formatierte Daten unter Berücksichtigung der folgenden Aspekte:

- Der Amazon S3 S3-Bucket muss sich in derselben AWS Region wie die Amazon Redshift Redshift-Datenbank befinden.
- Um über einen VPC-Endpunkt auf Ihre Amazon-S3-Daten zugreifen zu können, richten Sie den Zugriff mithilfe von IAM-Richtlinien und IAM-Rollen ein, wie unter Verwendung von Amazon Redshift Spectrum mit Enhanced VPC-Routing im Amazon-Redshift-Verwaltungshandbuch beschrieben.
- COPY wendet Kompressionskodierungen nicht automatisch an.
- Es werden nur die folgenden COPY-Parameter unterstützt:
  - ACCEPTINVCHARS beim Kopieren aus einer ORC- oder Parquet-Datei.
  - FILLRECORD
  - FROM
  - IAM_ROLE
  - CREDENTIALS
  - STATUPDATE
  - MANIFEST
  - EXPLICIT_IDS
- Tritt beim Laden ein Fehler auf, schlägt der COPY-Befehl fehl. ACCEPTANYDATE und MAXERROR werden für spaltenbasierte Datentypen nicht unterstützt.
- Fehlermeldungen werden an den SQL-Client gesendet. Einige Fehler werden in STL_LOAD_ERRORS und STL_ERROR protokolliert.
- COPY fügt Werte in derselben Reihenfolge in die Spalten der Zieltabelle ein, in der die Spalten in den spaltenbasierten Datendateien vorkommen. Die Anzahl der Spalten in der Zieltabelle und die Anzahl der Spalten in der Datendatei müssen übereinstimmen.

- Wenn die Datei, die Sie für die COPY-Operation angeben, eine der folgenden Erweiterungen besitzt, werden die Daten dekomprimiert, ohne dass Parameter hinzugefügt werden müssen:
  - `.gz`
  - `.snappy`
  - `.bz2`
- COPY aus den Dateiformaten Parquet und ORC verwendet Redshift Spectrum und den Bucket-Zugriff. Um COPY für diese Formate zu verwenden, stellen Sie sicher, dass es keine IAM-Richtlinien gibt, die die Verwendung von vorsignierten Amazon S3 S3-URLs blockieren. Die von Amazon Redshift generierten vorsignierten URLs sind 1 Stunde lang gültig, sodass Amazon Redshift genügend Zeit hat, um alle Dateien aus dem Amazon S3 S3-Bucket zu laden. Für jede mit COPY gescannte Datei aus spaltenförmigen Datenformaten wird eine eindeutige vorsignierte URL generiert. Achten Sie bei Bucket-Richtlinien, die eine `s3:signatureAge` Aktion beinhalten, darauf, den Wert auf mindestens 3.600.000 Millisekunden festzulegen. Weitere Informationen finden Sie unter Verwenden von Amazon Redshift Spectrum mit Enhanced VPC Routing.

DATEFORMAT- und TIMEFORMAT-Zeichenfolgen

Der COPY-Befehl verwendet die Optionen DATEFORMAT und TIMEFORMAT zur Analyse von Datums- und Zeitwerten in Ihren Quelldaten. DATEFORMAT und TIMEFORMAT sind formatierte Zeichenfolgen, die dem Format der Datums- und Zeitwerte Ihrer Quelldaten entsprechen müssen. Ein COPY-Befehl, der Quelldaten mit dem Datumswert `Jan-01-1999` lädt, muss beispielsweise die folgende DATEFORMAT-Zeichenfolge enthalten:

```
COPY ...
            DATEFORMAT AS 'MON-DD-YYYY'
```

Weitere Informationen zur Verwaltung von COPY-Datenkonvertierungen finden Sie unter Datenkonvertierungsparameter.

Die DATEFORMAT- oder TIMEFORMAT-Zeichenfolgen können Datums-/Uhrzeittrennzeichen (wie „`-`“, „`/`“ oder „`:`“) sowie die Datumsteil– und Zeitteilformate in der folgenden Tabelle enthalten.

> **Note**
>
> Wenn Sie das Format Ihrer Datums- oder Zeitwerte nicht mit den folgenden Datumsteilen und Zeitteilen abgleichen können oder wenn Ihre Datums- und Zeitwerte unterschiedliche Formate aufweisen, verwenden Sie das Argument `'auto'` mit dem Parameter

DATEFORMAT oder TIMEFORMAT. Das Argument `'auto'` erkennt verschiedene Formate, die bei der Verwendung einer DATEFORMAT- oder TIMEFORMAT-Zeichenfolge nicht unterstützt werden. Weitere Informationen finden Sie unter Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT.

Datumsteil oder Zeitteil	Bedeutung
YY	Jahr ohne Jahrhundert
YYYY	Jahr mit Jahrhundert
MM	Monat als Zahl
MON	Monat als Name (abgekürzt oder vollständig)
DD	Tag des Monats als Zahl
HH oder HH24	Stunde (24-Stunden-Uhr) **Note** In DATETIME-Formatzeichenfolgen für SQL-Funktionen ist HH dasselbe wie HH12. In den DATEFORMAT- und TIMEFORMAT-Zeichenfolgen für COPY ist HH jedoch dasselbe wie HH24.
HH12	Stunde (12-Stunden-Uhr)
MI	Minuten
SS	Sekunden
AM oder PM	Meridiananzeige (für die 12-Stunden-Uhr)

Das Standarddatumsformat ist YYYY-MM-DD. Das Standardzeitstempelformat ohne Zeitzone (TIMESTAMP) ist YYYY-MM-DD HH:MI:SS. Das Standardformat für Zeitstempel mit Zeitzone (TIMESTAMPTZ) ist YYYY-MM-TT HH:MI:SSOF, wobei OF der Unterschied (Offset) zu UTC ist (zum Beispiel -8:00. Sie können keinen Zeitzonenspezifizierer (TZ, tz oder OF) im timeformat_string angeben. Das Sekundenfeld (SS) unterstützt auch Sekundenbruchteile bis zu einer Mikrosekunde. Um TIMESTAMPTZ-Daten zu laden, die sich in einem anderen Format als dem Standardformat befinden, geben Sie 'auto' an.

Im Folgenden finden Sie einige Beispiele für Datumsangaben oder Uhrzeiten, die Sie in Ihren Quelldaten finden können, sowie die entsprechenden DATEFORMAT- oder TIMEFORMAT-Zeichenfolgen.

Beispiel für das Datum oder die Uhrzeit in den Quelldaten	Syntax von DATEFORMAT oder TIMEFORMAT
03/31/2003	DATEFORMAT „MM/DD/YYYY“
31. März 2003	DATEFORMAT „MON DD, YYYY“
03.31.2003 18:45:05 03.31.2003 18:45:05.123456	TIMEFORMAT „MM.DD.YYYY HH:MI:SS“

Beispiel

Ein Beispiel für die Verwendung von TIMEFORMAT finden Sie unter Laden eines Zeit- oder Datumsstempels.

Verwenden der automatischen Erkennung bei DATEFORMAT und TIMEFORMAT

Wenn Sie `'auto'` als Argument für den Parameter DATEFORMAT oder TIMEFORMAT angeben, erkennt Amazon Redshift automatisch das Datums- oder Zeitformat in Ihren Quelldaten und konvertiert es. Es folgt ein Beispiel.

```
copy favoritemovies from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
dateformat 'auto';
```

Wenn COPY mit dem Argument `'auto'` für DATEFORMAT und TIMEFORMAT verwendet wird, erkennt COPY die Datums- und Zeitformate, die in der Tabelle in DATEFORMAT- und TIMEFORMAT-Zeichenfolgen aufgelistet sind. Zusätzlich erkennt das Argument `'auto'` die folgenden Formate, die bei Verwendung einer DATEFORMAT- und TIMEFORMAT-Zeichenfolge nicht unterstützt werden.

Format	Beispiel für eine gültige Eingabezeichenfolge
ISO 8601	2019-02-11T05:09:12.195Z
Julianisch	J2451187
BC	Jan-08-95 BC
YYYYMMDD HHMISS	19960108 040809
YYMMDD HHMISS	960108 040809
YYYY.DDD	1996.008
YYYY-MM-DD HH:MI:SS.SSS	1996-01-08 04:05:06.789
DD Mon HH:MI:SS YYYY TZ	17 Dec 07:37:16 1997 PST
MM/DD/YYYY HH:MI:SS.SS TZ	12/17/1997 07:37:16.00 PST
YYYY-MM-DD HH:MI:SS+/-TZ	1997-12-17 07:37:16-08
DD.MM.YYYY HH:MI:SS TZ	12.17.1997 07:37:16.00 PST

Die automatische Erkennung unterstützt keine Epochensekunden und Epochenmillisekunden.

Um herauszufinden, ob ein Datums- oder Zeitstempelwert automatisch konvertiert wird, verwenden Sie eine CAST-Funktion, um zu versuchen, die Zeichenfolge in einen Datums- oder Zeitstempelwert

zu konvertieren. Mit den folgenden Befehlen wird beispielsweise der Zeitstempelwert getestet `'J2345678 04:05:06.789'`:

```
create table formattest (test char(21));
insert into formattest values('J2345678 04:05:06.789');
select test, cast(test as timestamp) as timestamp, cast(test as date) as date from
 formattest;

        test          |      timestamp      | date
----------------------+---------------------+------------
J2345678 04:05:06.789   1710-02-23 04:05:06 1710-02-23
```

Wenn die Quelldaten für eine DATE-Spalte Zeitinformationen enthalten, wird die Zeitkomponente abgeschnitten. Wenn die Quelldaten für eine TIMESTAMP-Spalte Zeitinformationen auslassen, wird für die Zeitkomponente 00:00:00 verwendet.

Beispiele für COPY

> **Note**
>
> In den folgenden Beispielen werden aus Gründen der Lesbarkeit Zeilenumbrüche verwendet. Verwenden Sie in Ihrer Zeichenfolge credentials-args keine Zeilenumbrüche oder Leerzeichen.

Themen

- Laden von TIME aus einer GZIP-Datei mit Pipe-Zeichen als Trennzeichen
- Laden eines Zeit- oder Datumsstempels
- Laden von Daten aus einer Datei mit Standardwerten
- COPY-Operation für Daten mit der Option ESCAPE
- Beispiele für die COPY-Operation aus JSON
- Beispiele für die Kopie aus Avro
- Vorbereiten von Dateien auf die COPY-Operation mit der Option ESCAPE
- Laden eines Shapefile in Amazon Redshift
- COPY-Befehl mit der Option NOLOAD

Laden von FAVORITEMOVIES aus einer DynamoDB-Tabelle

Die AWS SDKs enthalten ein einfaches Beispiel für die Erstellung einer DynamoDB-Tabelle namens Movies. (Informationen zu diesem Beispiel finden Sie unter Erste Schritte mit DynamoDB.) Im folgenden Beispiel wird die Amazon-Redshift-Tabelle MOVIES mit Daten aus der DynamoDB-Tabelle geladen. Die Amazon-Redshift-Tabelle muss in der Datenbank bereits vorhanden sein.

```
copy favoritemovies from 'dynamodb://Movies'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Laden von LISTING aus einem Amazon-S3-Bucket

Im folgenden Beispiel wird LISTING aus einem Amazon-S3-Bucket geladen. Der COPY-Befehl lädt alle Dateien im Ordner `/data/listing/`.

```
copy listing
from 's3://mybucket/data/listing/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Laden von LISTING aus einem Amazon-EMR-Cluster

Im folgenden Beispiel wird die Tabelle SALES mit durch Tabulatoren getrennten Daten aus lzop-komprimierten Dateien in einem Amazon-EMR-Cluster geladen. COPY lädt jede Datei im Ordner `myoutput/`, die mit `part-` beginnt.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/part-*'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '\t' lzop;
```

Im folgenden Beispiel wird die Tabelle SALES mit JSON-formatierten Daten in einem Amazon-EMR-Cluster geladen. COPY lädt jede Datei im Ordner `myoutput/json/`.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/json/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
JSON 's3://mybucket/jsonpaths.txt';
```

Verwenden eines Manifests für die Angabe von Datendateien

Sie können ein Manifest verwenden, um sicherzustellen, dass der COPY-Befehl alle erforderlichen Dateien aus Amazon S3 und nur diese lädt. Sie können ein Manifest auch verwenden, wenn Sie mehrere Dateien aus verschiedenen Buckets laden müssen oder Dateien, die nicht dasselbe Präfix besitzen.

Angenommen, Sie müssen die folgenden drei Dateien laden: `custdata1.txt`, `custdata2.txt` und `custdata3.txt`. Sie könnten den folgenden Befehl verwenden, um alle Dateien in `mybucket` zu laden, die mit `custdata` beginnen, indem Sie ein Präfix angeben:

```
copy category
from 's3://mybucket/custdata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Wenn aufgrund eines Fehlers nur zwei der Dateien vorhanden sind, lädt COPY nur diese beiden Dateien und wird erfolgreich abgeschlossen. Dies führt zu einem unvollständigen Datenladevorgang. Wenn der Bucket auch eine nicht erwünschte Datei enthält, die zufällig dasselbe Präfix verwendet, beispielsweise eine Datei namens `custdata.backup`, lädt COPY diese Datei ebenfalls. Dies führt dazu, dass nicht erwünschte Daten geladen werden.

Um sicherzustellen, dass alle erforderlichen Dateien geladen werden, und zu verhindern, dass nicht erwünschte Dateien geladen werden, können Sie eine Manifestdatei verwenden. Die Manifestdatei ist eine JSON formatierte Textdatei, die die Dateien auflistet, die durch den COPY-Befehl verarbeitet werden sollen. Beispielsweise lädt das folgende Manifest die drei Dateien aus dem vorherigen Beispiel.

```
{
   "entries":[
```

```
      {
         "url":"s3://mybucket/custdata.1",
         "mandatory":true
      },
      {
         "url":"s3://mybucket/custdata.2",
         "mandatory":true
      },
      {
         "url":"s3://mybucket/custdata.3",
         "mandatory":true
      }
   ]
}
```

Das optionale `mandatory`-Flag gibt an, ob COPY beendet werden soll, wenn die Datei nicht vorhanden ist. Der Standardwert ist `false`. Unabhängig von obligatorischen Einstellungen wird COPY beendet, wenn keine Dateien gefunden werden. In diesem Beispiel gibt COPY einen Fehler zurück, wenn eine der Dateien nicht gefunden wird. Nicht erwünschte Dateien, die möglicherweise geladen werden, wenn Sie nur ein Schlüsselpräfix angeben, beispielsweise `custdata.backup`, werden ignoriert, da sie nicht im Manifest aufgelistet werden.

Beim Laden aus Datendateien im ORC- oder Parquet-Format ist ein `meta`-Feld erforderlich, wie im folgenden Beispiel gezeigt.

```
{
   "entries":[
      {
         "url":"s3://mybucket-alpha/orc/2013-10-04-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      },
      {
         "url":"s3://mybucket-beta/orc/2013-10-05-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      }
   ]
```

```
}
```

Im folgenden Beispiel wird ein Manifest namens verwendet `cust.manifest`.

```
copy customer
from 's3://mybucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as orc
manifest;
```

Sie können ein Manifest verwenden, um Dateien aus verschiedenen Buckets hochzuladen oder Dateien, die nicht das gleiche Präfix verwenden. Das folgende Beispiel zeigt das JSON-Format, um Daten aus Dateien zu laden, deren Namen mit einem Datumsstempel beginnen.

```
{
  "entries": [
    {"url":"s3://mybucket/2013-10-04-custdata.txt","mandatory":true},
    {"url":"s3://mybucket/2013-10-05-custdata.txt","mandatory":true},
    {"url":"s3://mybucket/2013-10-06-custdata.txt","mandatory":true},
    {"url":"s3://mybucket/2013-10-07-custdata.txt","mandatory":true}
  ]
}
```

Das Manifest kann Dateien auflisten, die sich in verschiedenen Buckets befinden, sofern sich die Buckets in derselben AWS Region wie der Cluster befinden.

```
{
  "entries": [
    {"url":"s3://mybucket-alpha/custdata1.txt","mandatory":false},
    {"url":"s3://mybucket-beta/custdata1.txt","mandatory":false},
    {"url":"s3://mybucket-beta/custdata2.txt","mandatory":false}
  ]
}
```

Laden von LISTING aus einer Datei mit Pipe-Zeichen als Trennzeichen (Standardtrennzeichen)

Das folgende Beispiel stellt einen sehr einfachen Fall dar, in dem keine Optionen angegeben sind und die Eingabedatei das Standardtrennzeichen enthält, ein Pipe-Zeichen (|).

```
copy listing
from 's3://mybucket/data/listings_pipe.txt'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Laden von LISTING unter Verwendung von Spaltendaten im Parquet-Format

Im folgenden Beispiel werden Daten aus einem Ordner in einem von Amazon S3 benannten Parquet geladen.

```
copy listing
from 's3://mybucket/data/listings/parquet/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as parquet;
```

Laden von LISTING unter Verwendung von Spaltendaten im ORC-Format

Im folgenden Beispiel werden Daten aus einem Ordner in Amazon S3 mit dem Namen `orc` geladen.

```
copy listing
from 's3://mybucket/data/listings/orc/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as orc;
```

Laden von EVENT mit Optionen

Im folgenden Beispiel werden durch Pipe-Zeichen getrennte Daten in die Tabelle EVENT geladen und anschließend die folgenden Regeln angewendet:

- Wenn Zeichenfolgen durch Paare von Angebotszeichen eingeschlossen werden, werden sie entfernt.
- Sowohl leere Zeichenfolgen als auch Zeichenfolgen, die Leerzeichen enthalten, werden als NULL-Werte geladen.
- Der Ladevorgang schlägt fehl, wenn mehr als 5 Fehler zurückgegeben werden.
- Zeitstempelwerte müssen das angegebene Format einhalten. Ein gültiger Zeitstempel ist beispielsweise `2008-09-26 05:43:12`.

```
copy event
from 's3://mybucket/data/allevents_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
removequotes
emptyasnull
blanksasnull
```

```
maxerror 5
delimiter '|'
timeformat 'YYYY-MM-DD HH:MI:SS';
```

Laden von VENUE aus einer Datendatei mit festen Spaltenbreiten

```
copy venue
from 's3://mybucket/data/venue_fw.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth 'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6';
```

Im vorherigen Beispiel wird angenommen, dass die Datendatei auf dieselbe Weise wie die gezeigten Beispieldateien formatiert ist. Im folgenden Beispiel dienen Leerzeichen als Platzhalter, sodass alle Spalten dieselbe Breite wie in der Spezifikation angegeben haben:

```
1  Toyota Park              Bridgeview  IL0
2  Columbus Crew Stadium    Columbus    OH0
3  RFK Stadium              Washington  DC0
4  CommunityAmerica BallparkKansas City KS0
5  Gillette Stadium         Foxborough  MA68756
```

Laden von CATEGORY aus einer CSV-Datei

Angenommen, Sie möchten CATEGORY mit den Werten laden, die in der folgenden Tabelle gezeigt werden.

catid	catgroup	catname	catdesc
12	Shows	Musicals	Musiktheater
13	Shows	Schauspiel	Theater, das kein Musiktheater ist
14	Shows	Oper	Alle Opern, sowohl leichte Opern als auch Rockopern
15	Konzerte	Klassisch	Alle Symphoniekonzerte, Konzerte und Chorkonzerte

Im folgenden Beispiel werden die Inhalte einer Textdatei gezeigt, deren Feldwerte durch Kommas getrennt sind.

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,All "non-musical" theatre
14,Shows,Opera,All opera, light, and "rock" opera
15,Concerts,Classical,All symphony, concerto, and choir concerts
```

Wenn Sie die Datei mit dem Parameter DELIMITER laden, um Eingabedaten mit Kommas als Trennzeichen anzugeben, schlägt der COPY-Befehl fehl, da einige Eingabefelder Kommas enthalten. Sie können dieses Problem beheben, indem Sie den Parameter CSV verwenden und die Felder, die Kommas enthalten, in Anführungszeichen einschließen. Wenn ein Anführungszeichen innerhalb einer Zeichenfolge vorkommt, die durch Anführungszeichen eingeschlossen wird, müssen Sie dieses mit einer Escape-Markierung versehen, indem Sie das Anführungszeichen verdoppeln. Das Standardanführungszeichen ist ein doppeltes Anführungszeichen. Daher müssen Sie jedes doppelte Anführungszeichen mit einem zusätzlichen doppelten Anführungszeichen als Escape-Zeichen verwenden. Ihre neue Eingabedatei sieht ungefähr wie folgt aus.

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,"All ""non-musical"" theatre"
14,Shows,Opera,"All opera, light, and ""rock"" opera"
15,Concerts,Classical,"All symphony, concerto, and choir concerts"
```

Angenommen, der Dateiname ist `category_csv.txt`, dann können Sie die Datei mittels des folgenden COPY-Befehls laden:

```
copy category
from 's3://mybucket/data/category_csv.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
csv;
```

Um die Notwendigkeit zu vermeiden, die doppelten Anführungszeichen in Ihrer Eingabe durch Escape-Zeichen zu markieren, können Sie ein anderes Anführungszeichen angeben, indem Sie den Parameter QUOTE AS verwenden. Beispielsweise verwendet die folgende Version von `category_csv.txt` '%' als Anführungszeichen:

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,%All "non-musical" theatre%
14,Shows,Opera,%All opera, light, and "rock" opera%
15,Concerts,Classical,%All symphony, concerto, and choir concerts%
```

Der folgende COPY-Befehl verwendet QUOTE AS, um zu laden `category_csv.txt`:

```
copy category
from 's3://mybucket/data/category_csv.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
csv quote as '%';
```

Laden von VENUE mit expliziten Werte für eine IDENTITY-Spalte

Im folgenden Beispiel wird angenommen, dass bei Erstellung der Tabelle VENUE mindestens eine Spalte (beispielsweise die Spalte `venueid`) als IDENTITY-Spalte angegeben wurde. Dieser Befehl überschreibt das IDENTITY-Standardverhalten, bei dem Werte für eine IDENTITY-Spalte automatisch generiert werden, und lädt stattdessen die expliziten Werte aus der Datei venue.txt. Amazon Redshift überprüft nicht, ob doppelte IDENTITY-Werte in die Tabelle geladen werden, wenn die Option EXLICIT_IDS verwendet wird.

```
copy venue
from 's3://mybucket/data/venue.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
explicit_ids;
```

Laden von TIME aus einer GZIP-Datei mit Pipe-Zeichen als Trennzeichen

Im folgenden Beispiel wird die Tabelle TIME aus einer GZIP-Datei mit Pipe-Zeichen als Trennzeichen geladen:

```
copy time
from 's3://mybucket/data/timerows.gz'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
gzip
delimiter '|';
```

Laden eines Zeit- oder Datumsstempels

Im folgenden Beispiel werden Daten mit formatierten Zeitstempeln geladen.

> **Note**
>
> Das TIMEFORMAT von `HH:MI:SS` kann auch Bruchteile von Sekunden jenseits von `SS` bis zu einer Detailtiefe von Mikrosekunden unterstützen. Die in diesem Beispiel verwendete Datei `time.txt` enthält eine einzige Zeile, `2009-01-12 14:15:57.119568`.

```
copy timestamp1
from 's3://mybucket/data/time.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
timeformat 'YYYY-MM-DD HH:MI:SS';
```

Das Ergebnis dieser COPY-Operation ist wie folgt:

```
select * from timestamp1;
c1
----------------------------
2009-01-12 14:15:57.119568
(1 row)
```

Laden von Daten aus einer Datei mit Standardwerten

Im folgenden Beispiel wird eine Variante der Tabelle VENUE in der Datenbank TICKIT verwendet. Betrachten Sie die Tabelle VENUE_NEW, die mit der folgenden Anweisung definiert wird:

```
create table venue_new(
venueid smallint not null,
venuename varchar(100) not null,
venuecity varchar(30),
venuestate char(2),
venueseats integer not null default '1000');
```

Betrachten Sie die Datendatei venue_noseats.txt, die keine Werte für die Spalte VENUESEATS enthält, wie im folgenden Beispiel gezeigt:

```
1|Toyota Park|Bridgeview|IL|
2|Columbus Crew Stadium|Columbus|OH|
3|RFK Stadium|Washington|DC|
4|CommunityAmerica Ballpark|Kansas City|KS|
5|Gillette Stadium|Foxborough|MA|
6|New York Giants Stadium|East Rutherford|NJ|
7|BMO Field|Toronto|ON|
8|The Home Depot Center|Carson|CA|
9|Dick's Sporting Goods Park|Commerce City|CO|
10|Pizza Hut Park|Frisco|TX|
```

Die folgende COPY-Anweisung lädt die Tabelle erfolgreich aus der Datei und wendet den Standardwert (1000) auf die ausgelassene Spalte an:

```
copy venue_new(venueid, venuename, venuecity, venuestate)
from 's3://mybucket/data/venue_noseats.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Betrachten Sie nun die geladene Tabelle:

```
select * from venue_new order by venueid;
venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
1 | Toyota Park                | Bridgeview      | IL         |       1000
2 | Columbus Crew Stadium      | Columbus        | OH         |       1000
3 | RFK Stadium                | Washington      | DC         |       1000
4 | CommunityAmerica Ballpark  | Kansas City     | KS         |       1000
5 | Gillette Stadium           | Foxborough      | MA         |       1000
6 | New York Giants Stadium    | East Rutherford | NJ         |       1000
7 | BMO Field                  | Toronto         | ON         |       1000
8 | The Home Depot Center      | Carson          | CA         |       1000
9 | Dick's Sporting Goods Park | Commerce City   | CO         |       1000
10 | Pizza Hut Park             | Frisco          | TX         |       1000
(10 rows)
```

Im folgenden Beispiel wird zusätzlich zur Annahme, dass in der Datei keine VENUESEATS-Daten enthalten sind, auch angenommen, dass keine VENUENAME-Daten enthalten sind:

```
1||Bridgeview|IL|
2||Columbus|OH|
3||Washington|DC|
4||Kansas City|KS|
5||Foxborough|MA|
6||East Rutherford|NJ|
7||Toronto|ON|
8||Carson|CA|
9||Commerce City|CO|
10||Frisco|TX|
```

Bei Verwendung derselben Tabellendefinition schlägt die COPY-Anweisung fehl, da für VENUENAME kein DEFAULT-Wert angegeben wurde und VENUENAME eine NOT NULL-Spalte ist:

```
copy venue(venueid, venuecity, venuestate)
from 's3://mybucket/data/venue_pipe.txt'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Betrachten Sie nun eine Variante der Tabelle VENUE, die eine IDENTITY-Spalte verwendet:

```
create table venue_identity(
venueid int identity(1,1),
venuename varchar(100) not null,
venuecity varchar(30),
venuestate char(2),
venueseats integer not null default '1000');
```

Nehmen Sie wie im vorherigen Beispiel an, dass es für die Spalte VENUESEATS keine entsprechenden Werte in der Quelldatei gibt. Die folgende COPY-Anweisung lädt die Tabelle erfolgreich einschließlich der vordefinierten IDENTITY-Datenwerte, anstatt diese Werte automatisch zu generieren:

```
copy venue(venueid, venuename, venuecity, venuestate)
from 's3://mybucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' explicit_ids;
```

Diese Anweisung schlägt fehl, da sie die Spalte IDENTITY nicht enthält (VENUEID fehlt in der Spaltenliste), aber den Parameter EXPLICIT_IDS einschließt:

```
copy venue(venuename, venuecity, venuestate)
from 's3://mybucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' explicit_ids;
```

Diese Anweisung schlägt fehl, da sie den Parameter EXPLICIT_IDS nicht enthält:

```
copy venue(venueid, venuename, venuecity, venuestate)
from 's3://mybucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

COPY-Operation für Daten mit der Option ESCAPE

Im folgenden Beispiel wird gezeigt, wie Zeichen geladen werden, die mit dem Trennzeichen übereinstimmen (in diesem Fall dem Pipe-Zeichen). Stellen Sie sicher, dass in der Eingabedatei alle

Pipe-Zeichen (|), die Sie laden möchten, mit dem Backslash-Zeichen (\) als Escape-Zeichen markiert sind. Laden Sie die Datei anschließend unter Verwendung des Parameters ESCAPE.

```
$ more redshiftinfo.txt
1|public\|event\|dwuser
2|public\|sales\|dwuser

create table redshiftinfo(infoid int,tableinfo varchar(50));

copy redshiftinfo from 's3://mybucket/data/redshiftinfo.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' escape;

select * from redshiftinfo order by 1;
infoid |       tableinfo
-------+--------------------
1      | public|event|dwuser
2      | public|sales|dwuser
(2 rows)
```

Ohne den Parameter ESCAPE schlägt dieser COPY-Befehl mit dem Fehler `Extra column(s) found` fehl.

> **Important**
>
> Wenn Sie Ihre Daten mittels einer COPY-Operation mit dem Parameter ESCAPE laden, müssen Sie den Parameter ESCAPE auch für Ihren UNLOAD-Befehl angeben, um die reziproke Ausgabedatei zu generieren. Wenn Sie UNLOAD unter Verwendung des Parameters ESCAPE ausführen, müssen Sie ESCAPE verwenden, um eine COPY-Operation für diese Daten auszuführen.

Beispiele für die COPY-Operation aus JSON

In den folgenden Beispielen wird die Tabelle CATEGORY mit den folgenden Daten geladen.

CATID	CATGROUP	CATNAME	CATDESC
1	Sport	MLB	Major League Baseball

CATID	CATGROUP	CATNAME	CATDESC
2	Sport	NHL	National Hockey League
3	Sport	NFL	National Football League
4	Sport	NBA	National Basketball Association
5	Konzerte	Klassisch	Alle Symphoniekonzerte, Konzerte und Chorkonzerte

Themen

- Laden von JSON-Daten unter Verwendung der Option „auto“
- Laden von JSON-Daten unter Verwendung der Option „auto ignorecase“
- Laden von JSON-Daten unter Verwendung einer JSONPaths-Datei
- Laden von JSON-Arrays unter Verwendung einer JSONPaths-Datei

Laden von JSON-Daten unter Verwendung der Option „auto“

Um JSON-Daten unter Verwendung der Option `'auto'` zu laden, müssen die JSON-Daten aus einem Satz von Objekten bestehen. Die Schlüsselnamen müssen mit den Spaltennamen übereinstimmen. In diesem Fall spielt die Reihenfolge jedoch keine Rolle. Im folgenden werden die Inhalte einer Datei namens gezeigt `category_object_auto.json`.

```
{
    "catdesc": "Major League Baseball",
    "catid": 1,
    "catgroup": "Sports",
    "catname": "MLB"
}
{
    "catgroup": "Sports",
    "catid": 2,
    "catname": "NHL",
    "catdesc": "National Hockey League"
}{
    "catid": 3,
    "catname": "NFL",
    "catgroup": "Sports",
```

```
    "catdesc": "National Football League"
}
{
    "bogus": "Bogus Sports LLC",
    "catid": 4,
    "catgroup": "Sports",
    "catname": "NBA",
    "catdesc": "National Basketball Association"
}
{
    "catid": 5,
    "catgroup": "Shows",
    "catname": "Musicals",
    "catdesc": "All symphony, concerto, and choir concerts"
}
```

Um Daten aus der JSON-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
from 's3://mybucket/category_object_auto.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 'auto';
```

Laden von JSON-Daten unter Verwendung der Option „auto ignorecase“

Um JSON-Daten unter Verwendung der Option `'auto ignorecase'` zu laden, müssen die JSON-Daten aus einem Satz von Objekten bestehen. Die Groß- und Kleinschreibung der Schlüsselnamen muss nicht mit den Spaltennamen übereinstimmen und die Reihenfolge spielt keine Rolle. Im folgenden werden die Inhalte einer Datei namens gezeigt `category_object_auto-ignorecase.json`.

```
{
    "CatDesc": "Major League Baseball",
    "CatID": 1,
    "CatGroup": "Sports",
    "CatName": "MLB"
}
{
    "CatGroup": "Sports",
    "CatID": 2,
    "CatName": "NHL",
```

```
    "CatDesc": "National Hockey League"
}{
    "CatID": 3,
    "CatName": "NFL",
    "CatGroup": "Sports",
    "CatDesc": "National Football League"
}
{
    "bogus": "Bogus Sports LLC",
    "CatID": 4,
    "CatGroup": "Sports",
    "CatName": "NBA",
    "CatDesc": "National Basketball Association"
}
{
    "CatID": 5,
    "CatGroup": "Shows",
    "CatName": "Musicals",
    "CatDesc": "All symphony, concerto, and choir concerts"
}
```

Um Daten aus der JSON-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
from 's3://mybucket/category_object_auto ignorecase.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 'auto ignorecase';
```

Laden von JSON-Daten unter Verwendung einer JSONPaths-Datei

Wenn die JSON-Datenobjekte den Spaltennamen nicht direkt entsprechen, können Sie eine JSONPaths-Datei verwenden, um die JSON-Elemente zu Spalten zuzuweisen. Die Reihenfolge der JSON-Quelldaten spielt keine Rolle. Die Reihenfolge der JSONPaths-Dateiausdrücke muss jedoch mit der Spaltenreihenfolge übereinstimmen. Angenommen, Sie verwenden die folgende Datendatei mit dem Namen `category_object_paths.json`.

```
{
    "one": 1,
    "two": "Sports",
    "three": "MLB",
    "four": "Major League Baseball"
```

```
}
{
    "three": "NHL",
    "four": "National Hockey League",
    "one": 2,
    "two": "Sports"
}
{
    "two": "Sports",
    "three": "NFL",
    "one": 3,
    "four": "National Football League"
}
{
    "one": 4,
    "two": "Sports",
    "three": "NBA",
    "four": "National Basketball Association"
}
{
    "one": 6,
    "two": "Shows",
    "three": "Musicals",
    "four": "All symphony, concerto, and choir concerts"
}
```

Die folgende JSONPaths-Datei namens `category_jsonpath.json` weist die Quelldaten zu den Tabellenspalten zu.

```
{
    "jsonpaths": [
        "$['one']",
        "$['two']",
        "$['three']",
        "$['four']"
    ]
}
```

Um Daten aus der JSON-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
```

```
from 's3://mybucket/category_object_paths.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 's3://mybucket/category_jsonpath.json';
```

Laden von JSON-Arrays unter Verwendung einer JSONPaths-Datei

Um Daten aus JSON-Datendateien zu laden, die aus einem Satz von Arrays bestehen, müssen Sie eine JSONPaths-Datei verwenden, um die Array-Elemente zu Spalten zuzuweisen. Angenommen, Sie verwenden die folgende Datendatei mit dem Namen `category_array_data.json`.

```
[1,"Sports","MLB","Major League Baseball"]
[2,"Sports","NHL","National Hockey League"]
[3,"Sports","NFL","National Football League"]
[4,"Sports","NBA","National Basketball Association"]
[5,"Concerts","Classical","All symphony, concerto, and choir concerts"]
```

Die folgende JSONPaths-Datei namens `category_array_jsonpath.json` weist die Quelldaten zu den Tabellenspalten zu.

```
{
    "jsonpaths": [
        "$[0]",
        "$[1]",
        "$[2]",
        "$[3]"
    ]
}
```

Um Daten aus der JSON-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
from 's3://mybucket/category_array_data.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 's3://mybucket/category_array_jsonpath.json';
```

Beispiele für die Kopie aus Avro

In den folgenden Beispielen wird die Tabelle CATEGORY mit den folgenden Daten geladen.

CATID	CATGROUP	CATNAME	CATDESC
1	Sport	MLB	Major League Baseball
2	Sport	NHL	National Hockey League
3	Sport	NFL	National Football League
4	Sport	NBA	National Basketball Association
5	Konzerte	Klassisch	Alle Symphoniekonzerte, Konzerte und Chorkonzerte

Themen

Laden von Avro-Daten unter Verwendung der Option „auto“

Um Daten aus Avro-Datendateien unter Verwendung des Arguments `'auto'` zu laden, müssen die Feldnamen im Avro-Schema mit den Spaltennamen übereinstimmen. Bei Verwendung des Arguments `'auto'` spielt die Reihenfolge keine Rolle. Im folgenden wird das Schema für eine Datei namens gezeigt `category_auto.avro`.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "catid", "type": "int"},
        {"name": "catdesc", "type": "string"},
        {"name": "catname", "type": "string"},
        {"name": "catgroup", "type": "string"},
}
```

Die Daten in einer Avro-Datei liegen im binären Format vor. Sie können daher nicht von Menschen gelesen werden. Im folgenden Beispiel wird eine JSON-Darstellung der Daten in der `category_auto.avro`-Datei gezeigt.

```
{
   "catid": 1,
   "catdesc": "Major League Baseball",
   "catname": "MLB",
   "catgroup": "Sports"
}
{
   "catid": 2,
   "catdesc": "National Hockey League",
   "catname": "NHL",
   "catgroup": "Sports"
}
{
   "catid": 3,
   "catdesc": "National Basketball Association",
   "catname": "NBA",
   "catgroup": "Sports"
}
{
   "catid": 4,
   "catdesc": "All symphony, concerto, and choir concerts",
   "catname": "Classical",
   "catgroup": "Concerts"
}
```

Um Daten aus der Avro-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
from 's3://mybucket/category_auto.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as avro 'auto';
```

Laden von Avro-Daten unter Verwendung der Option „auto ignorecase“

Um Daten aus Avro-Datendateien unter Verwendung des Arguments `'auto ignorecase'` zu laden, muss die Groß-/Kleinschreibung der Feldnamen im Avro-Schema nicht mit den Spaltennamen übereinstimmen. Bei Verwendung des Arguments `'auto ignorecase'` spielt die Reihenfolge keine Rolle. Im folgenden wird das Schema für eine Datei namens gezeigt `category_auto-ignorecase.avro`.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "CatID", "type": "int"},
        {"name": "CatDesc", "type": "string"},
        {"name": "CatName", "type": "string"},
        {"name": "CatGroup", "type": "string"},
}
```

Die Daten in einer Avro-Datei liegen im binären Format vor. Sie können daher nicht von Menschen gelesen werden. Im folgenden Beispiel wird eine JSON-Darstellung der Daten in der `category_auto-ignorecase.avro`-Datei gezeigt.

```
{
   "CatID": 1,
   "CatDesc": "Major League Baseball",
   "CatName": "MLB",
   "CatGroup": "Sports"
}
{
   "CatID": 2,
   "CatDesc": "National Hockey League",
   "CatName": "NHL",
   "CatGroup": "Sports"
}
{
   "CatID": 3,
   "CatDesc": "National Basketball Association",
   "CatName": "NBA",
   "CatGroup": "Sports"
}
{
   "CatID": 4,
   "CatDesc": "All symphony, concerto, and choir concerts",
   "CatName": "Classical",
   "CatGroup": "Concerts"
}
```

Um Daten aus der Avro-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
from 's3://mybucket/category_auto-ignorecase.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as avro 'auto ignorecase';
```

Laden von Avro-Daten unter Verwendung einer JSONPaths-Datei

Wenn die Feldnamen im Avro-Schema den Spaltennamen nicht direkt entsprechen, können Sie eine JSONPaths-Datei verwenden, um die Schema-Elemente zu Spalten zuzuweisen. Die Reihenfolge der JSONPaths-Dateiausdrücke muss mit der Reihenfolge der Spalten übereinstimmen.

Angenommen, Sie verwenden eine Datendatei namens `category_paths.avro`, die dieselben Daten wie im vorherigen Beispiel enthält, jedoch mit dem folgenden Schema.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "desc", "type": "string"},
        {"name": "name", "type": "string"},
        {"name": "group", "type": "string"},
        {"name": "region", "type": "string"}
     ]
}
```

Die folgende JSONPaths-Datei namens `category_path.avropath` weist die Quelldaten zu den Tabellenspalten zu.

```
{
    "jsonpaths": [
        "$['id']",
        "$['group']",
        "$['name']",
        "$['desc']"
    ]
}
```

Um Daten aus der Avro-Datei im vorherigen Beispiel zu laden, führen Sie den folgenden COPY-Befehl aus.

```
copy category
from 's3://mybucket/category_object_paths.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format avro 's3://mybucket/category_path.avropath ';
```

Vorbereiten von Dateien auf die COPY-Operation mit der Option ESCAPE

Im folgenden Beispiel wird beschrieben, wie Sie Daten vorbereiten können, damit Zeichen für neue Zeilen mit Escape-Zeichen markiert werden, bevor die Daten unter Verwendung des COPY-Befehls mit dem Parameter ESCAPE in eine Amazon-Redshift-Tabelle geladen werden. Wenn die Zeichen für neue Zeilen in den Daten nicht mit Escape-Zeichen markiert werden, gibt Amazon Redshift Ladefehler zurück, sobald Sie den COPY-Befehl ausführen, da die Zeichen für neue Zeilen normalerweise als Datensatztrennzeichen verwendet werden.

Betrachten Sie beispielsweise eine Datei oder eine Spalte in einer externen Tabelle, die Sie in eine Amazon-Redshift-Tabelle kopieren möchten. Wenn die Datei oder Spalte XML-formatierte Inhalte oder ähnliche Daten enthält, müssen Sie sicherstellen, dass alle Zeichen für neue Zeilen (\n), die Teil des Inhalts sind, mit dem Backslash-Zeichen (\) als Escape-Zeichen markiert werden.

Eine Datei oder Tabelle, die eingebettete Zeichen für Zeilenumbrüche enthält, bietet ein vergleichsweise einfaches Muster für den Vergleich. Die eingebetteten Zeichen für neue Zeilen folgen wahrscheinlich meistens einem >-Zeichen, wobei sich dazwischen möglicherweise einige Leerstellen (`' '` oder Tabulatorzeichen) befinden, wie Sie im folgenden Beispiel für eine Textdatei namens `nlTest1.txt` sehen können.

```
$ cat nlTest1.txt
<xml start>
<newline characters provide>
<line breaks at the end of each>
<line in content>
</xml>|1000
<xml>
</xml>|2000
```

Im folgenden Beispiel können Sie ein Textverarbeitungsprogramm ausführen, um die Quelldatei vorab zu bearbeiten und an den nötigen Stellen Escape-Zeichen einzufügen. (Das Zeichen | soll als Trennzeichen verwendet werden, um Spaltendaten beim Kopieren in eine Amazon-Redshift-Tabelle zu trennen).

```
$ sed -e ':a;N;$!ba;s/>[[:space:]]*\n/>\\\n/g' nlTest1.txt > nlTest2.txt
```

Ähnlich können Sie Perl verwenden, um eine vergleichbare Operation auszuführen:

```
cat nlTest1.txt | perl -p -e 's/>\s*\n/>\\\n/g' > nlTest2.txt
```

Um die Daten aus der Datei `nlTest2.txt` in Amazon Redshift laden zu können, wurde in eine Tabelle mit zwei Spalten erstellt. Die erste Spalte c1 ist eine Zeichenspalte, die XML-formatierte Inhalte aus der Datei `nlTest2.txt` aufnimmt. Die zweite Spalte c2 enthält Ganzzahlwerte, die aus derselben Datei geladen wurden.

Nach Ausführung des Befehls `sed` können Sie Daten unter Verwendung des Parameters ESCAPE korrekt aus der Datei `nlTest2.txt` in eine Amazon-Redshift-Tabelle laden.

> **Note**
>
> Wenn Sie den COPY-Befehl zusammen mit dem Parameter ESCAPE verwenden, wird eine Reihe von Sonderzeichen mit dem Escape-Zeichen markiert, zu denen auch das Backslash-Zeichen gehört (einschließlich des Zeichens für neue Zeilen).

```
copy t2 from 's3://mybucket/data/nlTest2.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
escape
delimiter as '|';

select * from t2 order by 2;

c1           |  c2
-------------+------
<xml start>
<newline characters provide>
<line breaks at the end of each>
<line in content>
</xml>
| 1000
<xml>
</xml>       | 2000
(2 rows)
```

Sie können Datendateien, die aus externen Datenbanken exportiert wurden, auf ähnliche Weise vorbereiten. Im Fall einer Oracle-Datenbank können Sie beispielsweise die Funktion REPLACE auf alle betroffenen Spalten in einer Tabelle anwenden, die Sie zu Amazon Redshift kopieren möchten.

```
SELECT c1, REPLACE(c2, \n',\\n' ) as c2 from my_table_with_xml
```

Zusätzlich stellen zahlreiche Tools für den Export sowie das Extrahieren, das Transformieren und das Laden (Extract, Transform, Load, ETL) von Datenbanken, die routinemäßig große Datenmengen verarbeiten, Optionen für die Angabe von Escape- und Trennzeichen bereit.

Laden eines Shapefile in Amazon Redshift

In den folgenden Beispielen wird gezeigt, wie Sie ein Esri-Shapefile mit COPY laden. Weitere Informationen zum Laden von Shapefiles finden Sie unter Laden eines Shapefile in Amazon Redshift.

Laden eines Shapefile

Die folgenden Schritte zeigen, wie Sie mit dem Befehl COPY OpenStreetMap Daten aus Amazon S3 aufnehmen. In diesem Beispiel wird davon ausgegangen, dass das norwegische Shapefile-Archiv von der Download-Site von Geofabrik in einen privaten Amazon S3 S3-Bucket in Ihrer Region hochgeladen wurde. AWS Die Dateien `.shp`, `.shx` und `.dbf` müssen dasselbe Amazon-S3-Präfix und denselben Dateinamen haben.

Erfassung von Daten ohne Vereinfachung

Mit den folgenden Befehlen werden Tabellen erstellt und Daten erfasst, die ohne Vereinfachung in die maximale Geometriegröße passen. Öffnen Sie `gis_osm_natural_free_1.shp` in Ihrer bevorzugten GIS-Software und überprüfen Sie die Spalten in diesem Layer. Standardmäßig werden entweder IDENTITY- oder GEOMETRY-Spalten zuerst angezeigt. Wenn eine GEOMETRY-Spalte zuerst angezeigt wird, können Sie die Tabelle wie folgt erstellen.

```
CREATE TABLE norway_natural (
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Wenn eine IDENTITY-Spalte an erster Stelle steht, können Sie die Tabelle wie folgt erstellen.

```
CREATE TABLE norway_natural_with_id (
```

```
   fid INT IDENTITY(1,1),
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Jetzt können Sie die Daten mit COPY erfassen.

```
COPY norway_natural FROM 's3://bucket_name/shapefiles/norway/
gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO: Load into table 'norway_natural' completed, 83891 record(s) loaded successfully
```

Oder Sie können die Daten wie folgt erfassen.

```
COPY norway_natural_with_id FROM 's3://bucket_name/shapefiles/norway/
gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO: Load into table 'norway_natural_with_id' completed, 83891 record(s) loaded
 successfully.
```

Erfassung von Daten mit Vereinfachung

Mit den folgenden Befehlen wird eine Tabelle erstellt und es wird versucht, Daten zu erfassen, die ohne Vereinfachung nicht in die maximale Geometriegröße passen. Untersuchen Sie das `gis_osm_water_a_free_1.shp`-Shapefile und erstellen Sie die entsprechende Tabelle wie folgt.

```
CREATE TABLE norway_water (
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Wenn der Befehl COPY ausgeführt wird, führt dies zu einem Fehler.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
```

```
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
ERROR:  Load into table 'norway_water' failed.  Check 'stl_load_errors' system table
 for details.
```

Eine Abfrage von STL_LOAD_ERRORS zeigt an, dass die Geometrie zu groß ist.

```
SELECT line_number, btrim(colname), btrim(err_reason) FROM stl_load_errors WHERE query
 = pg_last_copy_id();
 line_number |    btrim     |                                  btrim
-------------+--------------
+-------------------------------------------------------------------------
     1184705 | wkb_geometry | Geometry size: 1513736 is larger than maximum supported
 size: 1048447
```

Zur Vereinfachung der Geometrien wird der Parameter SIMPLIFY AUTO zum COPY-Befehl hinzugefügt.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
SIMPLIFY AUTO
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';

INFO:  Load into table 'norway_water' completed, 1989196 record(s) loaded successfully.
```

Um die Zeilen und Geometrien anzuzeigen, die vereinfacht wurden, fragen Sie ab SVL_SPATIAL_SIMPLIFY.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+----------------------
    20 |     1184704 |                -1 |      1513736 | t          |    1008808 |
 1.276386653895e-05
    20 |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

Die Verwendung von SIMPLIFY AUTO max_tolerance mit einer Toleranz, die niedriger ist als die automatisch berechnete, führt wahrscheinlich zu einem Einlesefehler. In diesem Fall sollten Sie MAXERROR verwenden, um Fehler zu ignorieren.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
SIMPLIFY AUTO 1.1E-05
MAXERROR 2
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';

INFO:  Load into table 'norway_water' completed, 1989195 record(s) loaded successfully.
INFO:  Load into table 'norway_water' completed, 1 record(s) could not be loaded.
 Check 'stl_load_errors' system table for details.
```

Fragen Sie SVL_SPATIAL_SIMPLIFY erneut ab, um den Datensatz zu identifizieren, den COPY nicht laden konnte.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+-----------------
    29 |     1184704 |           1.1e-05 |      1513736 | f          |          0 |
            0
    29 |     1664115 |           1.1e-05 |      1233456 | t          |     794432 |
     1.1e-05
```

In diesem Beispiel konnte der erste Datensatz nicht eingefügt werden, sodass die Spalte `simplified` falsch angezeigt wird. Der zweite Datensatz wurde innerhalb der vorgegebenen Toleranz geladen. Die endgültige Größe ist jedoch größer als bei Verwendung der automatisch berechneten Toleranz ohne Angabe der maximalen Toleranz.

Laden aus einem komprimierten Shapefile

Die COPY-Funktion von Amazon Redshift unterstützt die Erfassung von Daten aus einem komprimierten Shapefile. Alle Shapefile-Komponenten müssen dasselbe Amazon-S3-Präfix und dasselbe Komprimierungssuffix aufweisen. Nehmen wir an, Sie möchten die Daten aus dem vorherigen Beispiel laden. In diesem Fall müssen sich die Dateien `gis_osm_water_a_free_1.shp.gz`, `gis_osm_water_a_free_1.dbf.gz` und `gis_osm_water_a_free_1.shx.gz` das gleiche Amazon-S3-Verzeichnis teilen. Der COPY-Befehl erfordert die Option GZIP, und in der FROM-Klausel muss die richtige komprimierte Datei angegeben werden, wie im Folgenden gezeigt.

```
COPY norway_natural FROM 's3://bucket_name/shapefiles/norway/compressed/
gis_osm_natural_free_1.shp.gz'
FORMAT SHAPEFILE
GZIP
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO:  Load into table 'norway_natural' completed, 83891 record(s) loaded successfully.
```

Laden von Daten in eine Tabelle mit einer anderen Spaltenreihenfolge

Wenn Sie eine Tabelle haben, die nicht GEOMETRY als erste Spalte hat, können Sie die Spalten-Mapping verwenden, um die Spalten der Zieltabelle zuzuordnen. Erstellen Sie z. B. eine Tabelle mit osm_id als erste Spalte.

```
CREATE TABLE norway_natural_order (
   osm_id BIGINT,
   wkb_geometry GEOMETRY,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Laden Sie dann ein Shapefile unter Verwendung des Spalten-Mappings.

```
COPY norway_natural_order(wkb_geometry, osm_id, code, fclass, name)
FROM 's3://bucket_name/shapefiles/norway/gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO:  Load into table 'norway_natural_order' completed, 83891 record(s) loaded
 successfully.
```

Laden von Daten in eine Tabelle mit einer Geografiespalte

Wenn Sie eine Tabelle mit einer GEOGRAPHY-Spalte haben, erfassen Sie zuerst in eine GEOMETRY-Spalte und wandeln dann die Objekte in GEOGRAPHY-Objekte um. Zum Beispiel: Nachdem Sie Ihr Shapefile in eine GEOMETRY-Spalte kopiert haben, ändern Sie die Tabelle, um eine Spalte mit dem Datentyp GEOGRAPHY hinzuzufügen.

```
ALTER TABLE norway_natural ADD COLUMN wkb_geography GEOGRAPHY;
```

Dann konvertieren Sie die Geometrien in Geografien.

```
UPDATE norway_natural SET wkb_geography = wkb_geometry::geography;
```

Optional können Sie auch die GEOMETRY-Spalte entfernen.

```
ALTER TABLE norway_natural DROP COLUMN wkb_geometry;
```

COPY-Befehl mit der Option NOLOAD

Verwenden Sie die Option NOLOAD mit dem COPY-Befehl, um Datendateien zu validieren, bevor sie tatsächlich geladen werden. Amazon Redshift analysiert die Eingabedatei und zeigt alle auftretenden Fehler an. Das folgende Beispiel verwendet die Option NOLOAD und es werden tatsächlich keine Zeilen in die Tabelle geladen.

```
COPY public.zipcode1
FROM 's3://mybucket/mydata/zipcode.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
NOLOAD
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/myRedshiftRole';

Warnings:
Load into table 'zipcode1' completed, 0 record(s) loaded successfully.
```

CREATE DATABASE

Erstellt eine neue Datenbank.

Sie müssen ein Superuser sein oder über die CREATEDB-Berechtigung verfügen, um eine Datenbank zu erstellen. Um eine Datenbank zu erstellen, die einer Zero-ETL-Integration zugeordnet ist, müssen Sie ein Superuser sein oder sowohl über CREATEDB- als auch über CREATEUSER-Rechte verfügen.

CREATE DATABASE kann nicht innerhalb eines Transaktionsblocks (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Syntax

```
CREATE DATABASE database_name
```

```
[ { [ WITH ]
    [ OWNER [=] db_owner ]
    [ CONNECTION LIMIT { limit | UNLIMITED } ]
    [ COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE } ]
    [ ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT } ]
  }
  | { [ WITH PERMISSIONS ] FROM DATASHARE datashare_name ] OF [ ACCOUNT account_id ]
 NAMESPACE namespace_guid }
  | { FROM { { ARN '<arn>' } { WITH DATA CATALOG SCHEMA '<schema>' | WITH NO DATA
 CATALOG SCHEMA } }
             | { INTEGRATION '<integration_id>'} }
  | { IAM_ROLE  {default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' } }
```

Parameter

database_name

Name der neuen Datenbank. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

WITH

Optionales Schlüsselwort.

OWNER

Gibt einen Datenbankbesitzer an.

=

Optionales Zeichen.

db_owner

Benutzername des Datenbankbesitzers.

CONNECTION LIMIT { Limit | UNLIMITED }

Die maximale Zahl von Datenbankverbindungen, die Benutzer gleichzeitig geöffnet haben dürfen. Das Limit wird für Superuser nicht durchgesetzt. Mithilfe des Schlüsselworts UNLIMITED können Sie die maximale Zahl gleichzeitiger Verbindungen festlegen. Möglicherweise gilt auch ein Limit für die Zahl der Verbindungen für die einzelnen Benutzer. Weitere Informationen finden Sie unter CREATE USER. Der Standardwert ist UNLIMITED. Um die aktuellen Verbindungen anzuzeigen, führen Sie eine Abfrage für die Systemansicht STV_SESSIONS aus.

> **Note**
>
> Wenn sowohl für Benutzer- als auch für Datenbankverbindungen Limits gelten, muss ein ungenutzter Verbindungsplatz verfügbar sein, der sich innerhalb beider Grenzen befindet, wenn ein Benutzer versucht, eine Verbindung herzustellen.

COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }

Eine Klausel, die angibt, ob bei der Suche oder dem Vergleich von Zeichenfolgen zwischen Groß- und Kleinschreibung unterschieden wird (CASE_SENSITIVE) oder nicht (CASE_INSENSITIVE). Der Standardwert ist CASE_SENSITIVE.

ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT }

Eine Klausel, die die verwendete Isolationsstufe bei Abfragen für eine Datenbank angibt.

- SERIALIZABLE Isolation — Bietet vollständige Serialisierbarkeit für gleichzeitige Transaktionen. Weitere Informationen finden Sie unter Serialisierbare Isolierung.
- SNAPSHOT-Isolierung — Bietet eine Isolationsstufe mit Schutz vor Aktualisierungs- und Löschkonflikten. Dies ist die Standardeinstellung für eine Datenbank, die in einem bereitgestellten Cluster oder serverlosen Namespace erstellt wurde.

Sie können wie folgt anzeigen, welches Gleichzeitigkeitsmodell Ihre Datenbank ausführt:

- Fragen Sie die Katalogansicht STV_DB_ISOLATION_LEVEL ab. Weitere Informationen finden Sie unter STV_DB_ISOLATION_LEVEL.

  ```
  SELECT * FROM stv_db_isolation_level;
  ```

- Fragen Sie die Ansicht PG_DATABASE_INFO ab.

  ```
  SELECT datname, datconfig FROM pg_database_info;
  ```

  Die Isolationsstufe für jede Datenbank wird neben dem Schlüssel `concurrency_model` angezeigt. Der Wert 1 steht für SNAPSHOT. Der Wert 2 steht für SERIALIZABLE (Serialisierbar).

In Amazon-Redshift-Datenbanken stellen sowohl die SERIALIZABLE-Isolation (serialisierbare Isolation) als auch die SNAPSHOT-Isolation Arten serialisierbarer Isolationsstufen dar. Dies bedeutet, dass ungültige Lesevorgänge, nicht wiederholbare Lesevorgänge und

Phantomlesevorgänge gemäß dem SQL-Standard verhindert werden. Beide Isolationsstufen stellen sicher, dass eine Transaktion mit einem Snapshot der Daten arbeitet, wie er zu Beginn der Transaktion existiert, und dass keine andere Transaktion diesen Snapshot ändern kann. Die SNAPSHOT-Isolation bietet jedoch keine vollständige Serialisierbarkeit, da Schreibverzerrungen (Write Skew) bei Einfügungen und Aktualisierungen für verschiedene Tabellenzeilen nicht verhindert werden.

Das folgende Szenario veranschaulicht Aktualisierungen von Schreibverzerrungen (Write Skew) unter Verwendung der SNAPSHOT-Isolationsstufe. Eine Tabelle namens `Numbers` enthält eine Spalte namens `digits` mit den Werten `0` und `1`. Die UPDATE-Anweisungen der Benutzer überschneiden sich nicht mit den Anweisungen des jeweils anderen Benutzers. Allerdings sind die Werte `0` und `1` vertauscht. Die SQL, die sie ausführen, folgt dieser Zeitleiste und kommt zu diesen Ergebnissen:

Zeit	Aktion von Benutze	Aktion von Benutzer 2
1	BEGIN;	
2		BEGIN;

3	SELECT * FROM Number
	`digits`
	`-`
	`------`
	`0`
	`1`	

4		SELECT * FROM Numbers;
		`digits`
		`------`

Zeit	Aktion von Benutze	Aktion von Benutzer 2
		0 1
5	UPDATE Number SET digits=0 WHERE digits=1;	
6	SELECT * FROM Number digits - ----- 0 0	
7	COMMIT	
8		Update Numbers SET digits=1 WHERE digits=0;
9		SELECT * FROM Numbers; digits ------ 1 1
10		COMMIT;

Zeit	Aktion von Benutze	Aktion von Benutzer 2

11	SELECT * FROM Number
	`digits`
	`-`
	`-----`
	`1`
	`0`	

12		SELECT * FROM Numbers;
		`digits`
		`------`
		`1`
		`0`

Wenn dasselbe Szenario mit serialisierbarer Isolation ausgeführt wird, beendet Amazon Redshift Benutzer 2 aufgrund einer serialisierbaren Verletzung und gibt den Fehler `1023` zurück. Weitere Informationen finden Sie unter Beheben von Fehlern für die serialisierbare Isolierung. In diesem Fall kann nur Benutzer 1 einen erfolgreichen Commit ausführen. Nicht alle Workloads setzen eine serialisierbare Isolation voraus. Es genügt dann eine Snapshot-Isolation als Zielisolationsstufe für Ihre Datenbank.

VON ARN '<ARN>'

Der AWS Glue Datenbank-ARN, der zum Erstellen der Datenbank verwendet werden soll.

{DATENKATALOGSCHEMA <schema>" | OHNE DATENKATALOGSCHEMA}

> **Note**
>
> Dieser Parameter ist nur anwendbar, wenn Ihr CREATE DATABASE-Befehl auch den FROM ARN-Parameter verwendet.

Gibt an, ob die Datenbank unter Verwendung eines Schemas zur Unterstützung des Zugriffs in AWS Glue Data Catalog erstellt werden soll.

AUS DER INTEGRATION '<integration_id>'

Gibt an, ob die Datenbank mit einem Zero-ETL-Integrationsbezeichner erstellt werden soll. Sie können die `integration_id` aus der SVV_INTEGRATION-Systemansicht abrufen. Ein Beispiel finden Sie unter Erstellen Sie Datenbanken, um die Ergebnisse von Zero-ETL-Integrationen zu erhalten. Weitere Informationen zum Erstellen von Datenbanken mit Zero-ETL-Integrationen finden Sie unter Erstellen von Zieldatenbanken in Amazon Redshift im Amazon Redshift Management Guide.

IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::*<AWS-Konto-id>*:role/*<role-name>*' }

> **Note**
>
> Dieser Parameter ist nur anwendbar, wenn Ihr CREATE DATABASE-Befehl auch den FROM ARN-Parameter verwendet.

Wenn Sie beim Ausführen des CREATE DATABASE-Befehls eine IAM-Rolle angeben, die dem Cluster zugeordnet ist, verwendet Amazon Redshift die Anmeldeinformationen der Rolle, wenn Sie Abfragen in der Datenbank ausführen.

Die Angabe des Schlüsselworts `default` bedeutet, die IAM-Rolle zu verwenden, die als Standard festgelegt und mit dem Cluster verknüpft ist.

Verwenden Sie `'SESSION'`, wenn Sie über eine Verbundidentität eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellen und über das mit diesem Befehl erstellte externe Schema auf die Tabellen zugreifen. Ein Beispiel zur Verwendung einer Verbundidentität finden Sie unter Verwenden einer Verbundidentität zur Verwaltung des Amazon-Redshift-Zugriffs auf lokale Ressourcen und externe Amazon-Redshift-Spectrum-Tabellen. Darin wird erläutert, wie Sie eine Verbundidentität konfigurieren.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Die IAM-Rolle muss mindestens die Berechtigung besitzen, eine LIST-Operation für den Amazon-S3-Bucket auszuführen, auf den zugegriffen werden soll, und eine GET-Operation für die Amazon-S3-Objekte, die der Bucket enthält. Weitere Informationen zur Verwendung von IAM_ROLE beim Erstellen einer Datenbank mithilfe von AWS Glue Data Catalog For Datashares finden Sie unter Working with Lake Formation-managed datashares as a consumer.

Nachfolgend ist die Syntax für die IAM_ROLE-Parameterzeichenfolge für einen einzelnen ARN aufgeführt.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Sie können Rollen miteinander verketten. Auf diese Weise kann der Cluster eine andere IAM-Rolle annehmen, die möglicherweise zu einem anderen Konto gehört. Es können bis zu 10 Rollen miteinander verkettet werden. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.

Fügen Sie dieser IAM-Rolle eine IAM-Berechtigungsrichtlinie ähnlich der folgenden an:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Informationen zu den Schritten für das Erstellen einer IAM-Rolle zur Verwendung mit der Verbundabfrage finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.

> **Note**
>
> Fügen Sie keine Leerzeichen in die Liste der verketteten Rollen ein.

Nachfolgend finden Sie die Syntax für die Verkettung von drei Rollen.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

Syntax für die Verwendung von CREATE DATABASE mit einem Datashare

Die folgende Syntax beschreibt den Befehl CREATE DATABASE, der verwendet wird, um Datenbanken aus einem Datashare für die gemeinsame Nutzung von Daten innerhalb desselben Kontos zu erstellen. AWS

```
CREATE DATABASE database_name
[ [ WITH PERMISSIONS ] FROM DATASHARE datashare_name ] OF [ ACCOUNT account_id ]
 NAMESPACE namespace_guid
```

Die folgende Syntax beschreibt den Befehl CREATE DATABASE, der verwendet wird, um Datenbanken aus einem Datashare für die gemeinsame Nutzung von Daten zwischen Konten zu erstellen. AWS

```
CREATE DATABASE database_name
[ [ WITH PERMISSIONS ] FROM DATASHARE datashare_name ] OF ACCOUNT account_id
 NAMESPACE namespace_guid
```

Parameter für die Verwendung von CREATE DATABASE mit einem Datashare

VOM DATASHARE

Ein Schlüsselwort, das angibt, wo sich das Datashare befindet.

datashare_name

Der Name des Datashares, auf dem die Verbraucherdatenbank erstellt wird.

WITH PERMISSIONS

Gibt an, dass für die aus dem Datashare erstellte Datenbank für den Zugriff auf einzelne Datenbankobjekte Berechtigungen auf Objektebene erforderlich sind. Ohne diese Klausel haben Benutzer oder Rollen, denen die USAGE-Berechtigung für die Datenbank erteilt wurde, automatisch Zugriff auf alle Datenbankobjekte in der Datenbank.

NAMESPACE namespace_guid

Ein Wert, der den Produzenten-Namespace angibt, zu dem das Datashare gehört.

ACCOUNT account_id

Ein Wert, der das Produzenten-Konto angibt, zu dem das Datashare gehört.

Verwendungshinweise für CREATE DATABASE in Verbindung mit Datashares

Wenn Sie als Datenbank-Superuser CREATE DATABASE verwenden, um Datenbanken aus Datenfreigaben innerhalb des AWS Kontos zu erstellen, geben Sie die Option NAMESPACE an. Die Option ACCOUNT ist optional. Wenn Sie CREATE DATABASE verwenden, um Datenbanken aus AWS kontenübergreifenden Datenfreigaben zu erstellen, geben Sie sowohl ACCOUNT als auch NAMESPACE vom Producer an.

Sie können nur eine Konsumentendatenbank für ein Datashare in einem Konsumenten-Cluster erstellen. Sie können nicht mehrere Konsumentendatenbanken erstellen, die auf dasselbe Datashare verweisen.

DATENBANK ERSTELLEN von AWS Glue Data Catalog

Um eine Datenbank mit einem AWS Glue Datenbank-ARN zu erstellen, geben Sie den ARN in Ihrem CREATE DATABASE-Befehl an.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA;
```

Optional können Sie auch einen Wert im IAM_ROLE-Parameter eingeben. Weitere Informationen zu Parametern und zulässigen Werten finden Sie unter Parameter.

Die folgenden Beispiele veranschaulichen, wie Sie mithilfe einer IAM-Rolle eine Datenbank aus einem ARN erstellen.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA
 IAM_ROLE <iam-role-arn>
```

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA
 IAM_ROLE default;
```

Sie können eine Datenbank auch mithilfe eines DATA CATALOG SCHEMA erstellen.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH DATA CATALOG SCHEMA
 <sample_schema> IAM_ROLE default;
```

Erstellen Sie Datenbanken, um die Ergebnisse von Zero-ETL-Integrationen zu erhalten

Um eine Datenbank mit einer Zero-ETL-Integrationsidentität zu erstellen, geben Sie das `integration_id` in Ihrem CREATE DATABASE-Befehl an.

```
CREATE DATABASE destination_db_name FROM INTEGRATION 'integration_id';
```

Rufen Sie beispielsweise zunächst die Integrations-IDs aus SVV_INTEGRATION ab;

```
SELECT integration_id FROM SVV_INTEGRATION;
```

Verwenden Sie dann eine der abgerufenen Integrations-IDs, um die Datenbank zu erstellen, die Zero-ETL-Integrationen empfängt.

```
CREATE DATABASE sampledb FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111';
```

Limits für CREATE DATABASE

Amazon Redshift setzt diese Limits für Datenbanken durch:

- Pro Cluster sind höchstens 60 benutzerdefinierte Datenbanken zulässig.
- Datenbanknamen dürfen höchstens 127 Bytes enthalten.

- Ein Datenbankname darf kein reserviertes Wort sein.

Datenbanksortierung

Bei der Sortierung handelt es sich um einen Regelsatz, der definiert, wie das Datenbankmodul die Zeichentypdaten in SQL vergleicht und sortiert. Die Sortierung ohne Berücksichtigung der Groß-/Kleinschreibung ist die am häufigsten verwendete Methode. Amazon Redshift verwendet eine groß-/kleinschreibungsneutrale Sortierung, um die Migration von anderen Data-Warehouse-Systemen zu erleichtern. Amazon Redshift unterstützt nativ die groß-/kleinschreibungsneutrale Sortierung und verwendet weiterhin zentrale Optimierungsmethoden wie Verteilungsschlüssel, Sortierschlüssel oder bereichsbeschränkte Scans.

Die COLLATE-Klausel legt die Standardsortierung für alle CHAR- und VARCHAR-Spalten in der Datenbank fest. Wenn CASE_INSENSITIVE angegeben ist, verwenden alle CHAR- oder VARCHAR-Spalten eine groß-/kleinschreibungsneutrale Sortierung. Weitere Informationen zur Sortierung finden Sie unter Sortierreihenfolgen.

Daten, die in Spalten ohne Berücksichtigung der Groß-/Kleinschreibung eingefügt oder übernommen werden, behalten ihre ursprüngliche Groß-/Kleinschreibung bei. Bei allen vergleichsbasierten Zeichenfolgenoperationen, einschließlich Sortieren und Gruppieren, wird die Groß- und Kleinschreibung nicht berücksichtigt. Mustervergleichsoperationen wie LIKE-Prädikate, „ähnlich wie“ und Funktionen für reguläre Ausdrücke sind ebenfalls unabhängig von der Groß- und Kleinschreibung.

Die folgenden SQL-Operationen unterstützen die geltende Sortiersemantik:

- Vergleichsoperatoren: **=**, **<>**, **<**, **<=**, **>**, **>=**.
- LIKE-Operator
- ORDER BY-Klauseln
- GROUP BY-Klauseln
- Aggregatfunktionen, die einen String-Vergleich verwenden, wie MIN und MAX und LISTAGG
- Fensterfunktionen, wie PARTITION BY-Klauseln und ORDER BY-Klauseln
- Skalare Funktionen greatest() und least(), STRPOS(), REGEXP_COUNT(), REGEXP_REPLACE(), REGEXP_INSTR(), REGEXP_SUBSTR()
- Distinct-Klausel
- UNION, INTERSECT und EXCEPT

- IN LIST

Bei externen Abfragen, einschließlich Verbundabfragen in Amazon Redshift Spectrum und Aurora PostgreSQL, ist die Sortierung der VARCHAR- oder CHAR-Spalte die gleiche wie die aktuelle Sortierung auf Datenbankebene.

Im folgenden Beispiel wird eine Amazon-Redshift-Spectrum-Tabelle abgefragt:

```
SELECT ci_varchar FROM spectrum.test_collation
WHERE ci_varchar = 'AMAZON';

ci_varchar
----------
amazon
Amazon
AMAZON
AmaZon
(4 rows)
```

Weitere Informationen zur Erstellung von Tabellen mit der Datenbanksortierung finden Sie unter CREATE TABLE.

Weitere Informationen über die COLLATE-Funktionen finden Sie unter Funktion COLLATE.

Einschränkungen bei der Datenbanksortierung

Die folgenden Einschränkungen gelten für die Verwendung der Datenbanksortierung in Amazon Redshift:

- Bei allen Systemtabellen oder -ansichten, einschließlich PG-Katalogtabellen und Amazon-Redshift-Systemtabellen, wird die Groß- und Kleinschreibung beachtet.
- Wenn Consumer- und Producer-Datenbank auf Datenbankebene unterschiedliche Sortierungen verwenden, unterstützt Amazon Redshift keine datenbank- und clusterübergreifenden Abfragen.
- Amazon Redshift unterstützt keine groß-/kleinschreibungsneutrale Sortierung in Abfragen, die ausschließlich für Führungsknoten gelten.

  Das folgende Beispiel zeigt eine nicht unterstützte groß-/kleinschreibungsneutrale Abfrage und den Fehler, den Amazon Redshift sendet:

```
SELECT collate(usename, 'case_insensitive') FROM pg_user;
```

```
ERROR:  Case insensitive collation is not supported in leader node only query.
```

- Amazon Redshift unterstützt keine Interaktion zwischen Spalten mit und ohne Unterscheidung zwischen Groß-/Kleinschreibung, wie z. B. Vergleichs-, Funktions-, Join- oder Set-Operationen.

  Die folgenden Beispiele zeigen Fehler bei der Interaktion von Spalten mit und ohne Unterscheidung zwischen Groß-/Kleinschreibung:

```
CREATE TABLE test
  (ci_col varchar(10) COLLATE case_insensitive,
   cs_col varchar(10) COLLATE case_sensitive,
   cint int,
   cbigint bigint);
```

```
SELECT ci_col = cs_col FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT concat(ci_col, cs_col) FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT ci_col FROM test UNION SELECT cs_col FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT * FROM test a, test b WHERE a.ci_col = b.cs_col;
ERROR:  Query with different collations is not supported yet.
```

```
Select Coalesce(ci_col, cs_col) from test;
ERROR:  Query with different collations is not supported yet.
```

```
Select case when cint > 0 then ci_col else cs_col end from test;
ERROR:  Query with different collations is not supported yet.
```

- Amazon Redshift unterstützt keine Sortierung für den Datentyp SUPER. Das Erstellen von SUPER-Spalten in groß-/kleinschreibungsneutralen Datenbanken sowie Interaktionen zwischen SUPER-Spalten und groß-/kleinschreibungsneutralen Spalten werden nicht unterstützt.

  Das folgende Beispiel erstellt eine Tabelle mit dem Datentyp SUPER in der groß-/kleinschreibungsneutralen Datenbank:

```
CREATE TABLE super_table (a super);
ERROR: SUPER column is not supported in case insensitive database.
```

Im folgenden Beispiel werden Daten mit einer groß-/kleinschreibungsneutralen Zeichenfolge abgefragt und mit den SUPER-Daten verglichen:

```
CREATE TABLE test_super_collation
(s super, c varchar(10) COLLATE case_insensitive, i int);
```

```
SELECT s = c FROM test_super_collation;
ERROR:  Coercing from case insensitive string to SUPER is not supported.
```

Damit diese Abfragen funktionieren, können Sie mit der Funktion COLLATE die Sortierung der einen Spalte an die andere anpassen. Weitere Informationen finden Sie unter Funktion COLLATE.

Beispiele

Erstellen einer Datenbank

Im folgenden Beispiel wird eine Datenbank namens TICKIT erstellt und der Benutzer DWUSER ist der Besitzer:

```
create database tickit
with owner dwuser;
```

Führen Sie eine Abfrage für die Katalogtabelle PG_DATABASE_INFO aus, um Details zu Datenbanken anzuzeigen.

```
select datname, datdba, datconnlimit
from pg_database_info
where datdba > 1;

 datname     | datdba | datconnlimit
-------------+--------+-------------
 admin       |    100 | UNLIMITED
 reports     |    100 | 100
 tickit      |    100 | 100
```

Im folgenden Beispiel wird eine Datenbank namens **sampledb** mit der Isolationsstufe SNAPSHOT erstellt.

```
CREATE DATABASE sampledb ISOLATION LEVEL SNAPSHOT;
```

Im folgenden Beispiel wird die Datenbank sales_db aus dem Datashare salesshare erstellt.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Beispiele für die Datenbanksortierung

Erstellen einer groß-/kleinschreibungsneutralen Datenbank

Im folgenden Beispiel wird die Datenbank `sampledb` und die Tabelle T1 erstellt und anschließend werden Daten in die Tabelle T1 eingefügt.

```
create database sampledb collate case_insensitive;
```

Stellen Sie mit Ihrem SQL-Client eine Verbindung mit der neuen Datenbank her, die Sie gerade erstellt haben. Wenn Sie den Amazon Redshift Query Editor v2 verwenden, wählen Sie `sampledb` im Editor aus. Wenn Sie RSQL nutzen, verwenden Sie einen ähnlichen Befehl wie den folgenden.

```
\connect sampledb;
```

```
CREATE TABLE T1 (
  col1 Varchar(20) distkey sortkey
);
```

```
INSERT INTO T1 VALUES ('bob'), ('john'), ('Mary'), ('JOHN'), ('Bob');
```

Die Abfrage findet dann Ergebnisse mit dem Inhalt `John`.

```
SELECT * FROM T1 WHERE col1 = 'John';

 col1
 ------
 john
 JOHN
```

```
(2 row)
```

Groß-/kleinschreibungsneutrale Sortierung

Das folgende Beispiel zeigt eine groß-/kleinschreibungsneutrale Sortierung mit Tabelle T1. Die Reihenfolge von Bob und bob oder John und john ist nicht deterministisch, da sie in der groß-/kleinschreibungsneutralen Spalte gleich sind.

```
SELECT * FROM T1 ORDER BY 1;

 col1
 ------
 bob
 Bob
 JOHN
 john
 Mary
(5 rows)
```

In ähnlicher Weise zeigt das folgende Beispiel eine Sortierung ohne Berücksichtigung der Groß- und Kleinschreibung mit der GROUP BY-Klausel. Bob und bob sind gleich und gehören zur gleichen Gruppe. Es ist nicht deterministisch, was im Ergebnis auftaucht.

```
SELECT col1, count(*) FROM T1 GROUP BY 1;

 col1 | count
 -----+------
 Mary |  1
 bob  |  2
 JOHN |  2
(3 rows)
```

Abfragen von groß-/kleinschreibungsneutralen Spalten mit einer Fensterfunktion

Im folgenden Beispiel wird eine Fensterfunktion zur Abfrage einer Spalte verwendet, in der die Groß- und Kleinschreibung nicht berücksichtigt wird.

```
SELECT col1, rank() over (ORDER BY col1) FROM T1;

 col1 | rank
 -----+------
 bob  |   1
```

```
 Bob  |   1
 john |   3
 JOHN |   3
 Mary |   5
(5 rows)
```

Abfragen mit dem Schlüsselwort DISTINCT

Im folgenden Beispiel wird die Tabelle T1 mit dem Schlüsselwort DISTINCT abgefragt.

```
SELECT DISTINCT col1 FROM T1;

 col1
 ------
 bob
 Mary
 john
(3 rows)
```

Abfrage mit der UNION-Klausel

Das folgende Beispiel zeigt die Ergebnisse der UNION der Tabellen T1 und T2.

```
CREATE TABLE T2 AS SELECT * FROM T1;
```

```
SELECT col1 FROM T1 UNION SELECT col1 FROM T2;

 col1
 ------
 john
 bob
 Mary
(3 rows)
```

DATASHARE ERSTELLEN

Erzeugt ein neues Datashare in der aktuellen Datenbank. Der Besitzer dieses Datashares ist derjenige, der den Befehl CREATE DATASHARE eingegeben hat.

Amazon Redshift verknüpft jedes Datashare mit einer einzelnen Amazon-Redshift-Datenbank. Sie können nur Objekte aus dieser Datenbank dem zugeordneten Datashare hinzufügen. Sie können mehrere Datashares in derselben Amazon-Redshift-Datenbank erstellen.

Weitere Informationen zu Datashares finden Sie unter Verwalten von Aufgaben bei der Datenfreigabe.

Verwenden Sie zum Anzeigen von Informationen zu Datashares SHOW DATASHARES.

Erforderliche Berechtigungen

Im Folgenden sind die erforderlichen Berechtigungen für CREATE DATASHARE aufgeführt:

- Superuser
- Benutzer mit der Berechtigung CREATE DATASHARE
- Datenbankbesitzer

Syntax

```
CREATE DATASHARE datashare_name
[[SET] PUBLICACCESSIBLE [=] TRUE | FALSE ];
```

Parameter

datashare_name

Der Name des Datashares. Der Name des Datashares muss im Cluster-Namespace eindeutig sein.

[[SET] PUBLICACCESSIBLE]

Eine Klausel, die angibt, ob das Datashare für öffentlich zugängliche Cluster freigegeben werden kann.

Der Standardwert für den `SET PUBLICACCESSIBLE` beträgt `FALSE`.

Nutzungshinweise

Standardmäßig hat der Besitzer des Datashares Rechte nur für die Freigabe, nicht aber für die Objekte innerhalb der Freigabe.

Nur Superuser und der Datenbankbesitzer können CREATE DATASHARE verwenden und ALTER-Berechtigungen an andere Benutzer oder Gruppen delegieren.

Beispiele

Im folgenden Beispiel wird das Datashare `salesshare` erstellt.

```
CREATE DATASHARE salesshare;
```

Im folgenden Beispiel wird das Datashare `demoshare` erstellt, der von AWS Data Exchange verwaltet wird.

```
CREATE DATASHARE demoshare SET PUBLICACCESSIBLE TRUE, MANAGEDBY ADX;
```

CREATE EXTERNAL FUNCTION

Erstellt eine skalare benutzerdefinierte Funktion (UDF), die auf AWS Lambda Amazon Redshift basiert. Weitere Informationen zu benutzerdefinierten Lambda-Funktionen finden Sie unter Erstellen einer skalaren Lambda-UDF.

Erforderliche Berechtigungen

Für CREATE EXTERNAL FUNCTION sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung CREATE [OR REPLACE] EXTERNAL FUNCTION

Syntax

```
CREATE [ OR REPLACE ] EXTERNAL FUNCTION external_fn_name ( [data_type] [, ...] )
RETURNS data_type
{ VOLATILE | STABLE }
LAMBDA 'lambda_fn_name'
IAM_ROLE { default | ‘arn:aws:iam::<AWS-Konto-id>:role/<role-name>’
RETRY_TIMEOUT milliseconds
MAX_BATCH_ROWS count
MAX_BATCH_SIZE size [ KB | MB ];
```

Parameter

OR REPLACE

Die Klausel gibt an, dass die vorhandene Funktion ersetzt wird, wenn eine Funktion mit demselben Namen und denselben Eingabeargument-Datentypen bzw. derselben Signatur vorhanden ist. Sie können eine Funktion nur durch eine neue Funktion ersetzen, wenn diese einen identischen Satz von Datentypen definiert. Sie müssen Superuser sein, um eine Funktion zu ersetzen.

Wenn Sie eine Funktion definieren, die den gleichen Namen wie eine vorhandene Funktion, aber eine andere Signatur besitzt, erstellen Sie eine neue Funktion. Der Funktionsname wird also überladen. Weitere Informationen finden Sie unter Überladen von Funktionsnamen.

external_fn_name

Der Name der externen Funktion. Wenn Sie einen Schemanamen angeben (z. B. myschema.myfunction), wird die Funktion unter Verwendung des angegebenen Schemas erstellt. Andernfalls wird die Funktion im aktuellen Schema erstellt. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

Es wird empfohlen, dass Sie alle UDFs mit dem Präfix benennen `f_`. Amazon Redshift reserviert das Präfix `f_` für UDF-Namen. Durch die Verwendung des `f_`-Präfixes stellen Sie sicher, dass Ihr UDF-Name nicht mit den SQL-Funktionsnamen für Amazon Redshift kollidiert, weder jetzt noch in Zukunft. Weitere Informationen finden Sie unter Benennung von UDFs.

data_type

Der Datentyp der Eingabeargumente. Weitere Informationen finden Sie unter Datentypen.

RETURNS data_type

Der Datentyp des Werts, der von der Funktion zurückgegeben wird. Der RETURNS-Datentyp kann ein beliebiger Standard-Amazon-Redshift-Datentyp sein. Weitere Informationen finden Sie unter Python-UDF-Datentypen.

VOLATILE | STABLE

Informiert den Abfragenoptimierer über die Volatilität der Funktion.

Um die beste Optimierung zu erzielen, kennzeichnen Sie Ihre Funktion mit der strengsten Volatilitätskategorie, die für sie gültig ist. Beginnend mit der Volatilitätskategorie mit der geringsten Strenge sind dies die Volatilitätskategorien:

- VOLATILE
- STABLE

VOLATILE

Bei gleichen Argumenten kann die Funktion unterschiedliche Ergebnisse für aufeinanderfolgende Aufrufe zurückgeben, auch für die Zeilen in einer einzelnen Anweisung. Der Abfrageoptimierer kann keine Annahmen über das Verhalten einer volatilen Funktion treffen. Eine Abfrage, die eine volatile Funktion verwendet, muss die Funktion für jede Eingabe neu auswerten.

STABLE

Bei gleichen Argumenten gibt die Funktion garantiert die gleichen Ergebnisse für alle aufeinderfolgenden Aufrufe zurück, die innerhalb einer einzelnen Anweisung verarbeitet werden. Die Funktion kann unterschiedliche Ergebnisse zurückgeben, wenn sie in unterschiedlichen Anweisungen aufgerufen wird. Diese Kategorie ermöglicht es dem Optimierer, die Häufigkeit zu reduzieren, mit der die Funktion innerhalb einer einzigen Anweisung aufgerufen wird.

Beachten Sie, dass, wenn die ausgewählte Strenge für die Funktion nicht gültig ist, das Risiko besteht, dass der Optimierer aufgrund dieser Strenge einige Aufrufe überspringt. Dies kann zu einer falschen Ergebnismenge führen.

Die Klausel IMMUTABLE wird derzeit für Lambda-UDFs nicht unterstützt.

LAMBDA 'lambda_fn_name'

Der Name der Funktion, die Amazon Redshift aufruft.

Schritte zum Erstellen einer AWS Lambda Funktion finden Sie unter Erstellen einer Lambda-Funktion mit der Konsole im AWS Lambda Entwicklerhandbuch.

Informationen zu den für die Lambda-Funktion erforderlichen Berechtigungen finden Sie unter AWS Lambda -Berechtigungen im AWS Lambda -Entwicklerhandbuch.

IAM_ROLE { default | ‘arn:aws:iam::*<AWS-Konto-id>*:role/*<role-name>*’

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE EXTERNAL FUNCTION-Befehl ausgeführt wird.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Der Befehl CREATE EXTERNAL

FUNCTION ist berechtigt, über diese IAM-Rolle Lambda-Funktionen aufzurufen. Wenn Ihr Cluster über eine bestehende IAM-Rolle mit Berechtigungen zum Aufrufen von Lambda-Funktionen verfügt, können Sie den ARN Ihrer Rolle ersetzen. Weitere Informationen finden Sie unter Konfigurieren des Autorisierungsparameters für Lambda-UDFs.

Im Folgenden wird die Syntax für den Parameter IAM_ROLE gezeigt.

```
IAM_ROLE 'arn:aws:iam::aws-account-id:role/role-name'
```

RETRY_TIMEOUT Millisekunden

Die Gesamtzeit in Millisekunden, die Amazon Redshift für die Verzögerungen bei Wiederholungs-Backoffs verwendet.

Anstatt es bei fehlgeschlagenen Abfragen sofort erneut zu versuchen, führt Amazon Redshift Backoffs durch und wartet eine bestimmte Zeit zwischen den Wiederholungsversuchen. Dann wiederholt Amazon Redshift die Anfrage, um die fehlgeschlagene Abfrage erneut auszuführen, bis die Summe aller Verzögerungen gleich oder größer als der von Ihnen angegebene RETRY_TIMEOUT-Wert ist. Der Standardwert ist 20 000 Millisekunden.

Wenn eine Lambda-Funktion aufgerufen wird, wiederholt Amazon Redshift Abfragen, bei denen Fehler wie `TooManyRequestsException`, `EC2ThrottledException` und `ServiceException` auftreten.

Sie können den Parameter RETRY_TIMEOUT auf 0 Millisekunden setzen, um Wiederholungsversuche für eine Lambda-UDF zu verhindern.

MAX_BATCH_ROWS Anzahl

Die maximale Zeilenanzahl, die Amazon Redshift in einer einzelnen Batch-Anforderung für einen einzelnen Lambda-Aufruf sendet.

Der Mindestwert für diesen Parameter ist 1. Der maximale Wert lautet INT_MAX oder 2 147 483 647.

Dieser Parameter ist optional. Der Standardwert lautet INT_MAX oder 2 147 483 647.

MAX_BATCH_SIZE Größe [KB | MB]

Die maximale Größe der Datennutzlast, die Amazon Redshift in einer einzelnen Batch-Anforderung für einen einzelnen Lambda-Aufruf sendet.

Der Mindestwert für diesen Parameter ist 1 KB. Der maximale Wert ist 5 MB.

Der Standardwert für diesen Parameter ist 5 MB.

KB und MB sind optional. Wenn Sie die Maßeinheit nicht festlegen, verwendet Amazon Redshift standardmäßig KB.

Nutzungshinweise

Beachten Sie beim Erstellen von Lambda-UDFs Folgendes:

- Die Reihenfolge der Lambda-Funktionsaufrufe für die Eingabeargumente ist nicht festgelegt oder garantiert. Dies kann je nach Cluster-Konfiguration von Instance zu Instance variieren, für die Abfragen ausgeführt werden.
- Es wird nicht garantiert, dass die Funktionen tatsächlich nur einmal auf jedes Eingabeargument angewendet werden. Die Interaktion zwischen Amazon Redshift und AWS Lambda kann zu wiederholten Aufrufen mit denselben Eingaben führen.

Beispiele

Nachfolgend finden Sie Beispiele für die Verwendung skalarer benutzerdefinierter Lambda-Funktionen (UDFs).

Skalares Lambda-UDF-Beispiel mit einer Node.js-Lambda-Funktion

Im folgenden Beispiel wird eine externe Funktion namens `exfunc_sum` erstellt, die zwei Ganzzahlen als Eingabeargumente benötigt. Diese Funktion gibt die Summe als Ganzzahl aus. Der Name der aufzurufenden Lambda-Funktion lautet `lambda_sum`. Die für diese Lambda-Funktion verwendete Sprache ist Node.js 12.x. Stellen Sie sicher, dass Sie die IAM-Rolle angeben. Das Beispiel verwendet `'arn:aws:iam::123456789012:user/johndoe'` als IAM-Rolle.

```
CREATE EXTERNAL FUNCTION exfunc_sum(INT,INT)
RETURNS INT
VOLATILE
LAMBDA 'lambda_sum'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';
```

Die Lambda-Funktion verarbeitet die Nutzlast der Anfrage und iteriert über jede Zeile. Alle Werte in einer einzelnen Zeile werden addiert, um die Summe für diese Zeile zu berechnen, die im Antwort-

Array gespeichert wird. Die Anzahl der Zeilen im Ergebnis-Array entspricht der Anzahl der in der Nutzlast der Anfrage empfangenen Zeilen.

Die JSON-Antwort-Nutzlast muss die Ergebnisdaten im Feld „results“ enthalten, damit sie von der externen Funktion erkannt werden kann. Das Feld „arguments“ in der Anforderung, die an die Lambda Funktion gesendet wird, enthält die Datennutzlast. Bei einer Batch-Anforderung können mehrere Zeilen in den Nutzdaten vorhanden sein. Die folgende Lambda-Funktion iteriert über alle Zeilen in der Nutzlast der Anfragedaten. Sie durchläuft zudem alle Werte innerhalb einer einzelnen Zeile.

```
exports.handler = async (event) => {
    // The 'arguments' field in the request sent to the Lambda function contains the
 data payload.
    var t1 = event['arguments'];

    // 'len(t1)' represents the number of rows in the request payload.
    // The number of results in the response payload should be the same as the number
 of rows received.
    const resp = new Array(t1.length);

    // Iterating over all the rows in the request payload.
    for (const [i, x] of t1.entries())
    {
        var sum = 0;
        // Iterating over all the values in a single row.
        for (const y of x) {
            sum = sum + y;
        }
        resp[i] = sum;
    }
    // The 'results' field should contain the results of the lambda call.
    const response = {
        results: resp
    };
    return JSON.stringify(response);
};
```

Im folgenden Beispiel wird die externe Funktion mit Literalwerten aufgerufen.

```
select exfunc_sum(1,2);
exfunc_sum
------------
```

```
 3
(1 row)
```

Im folgenden Beispiel wird eine Tabelle namens t_sum mit zwei Spalten, c1 und c2, vom Datentyp Ganzzahl erstellt und zwei Datenzeilen werden eingefügt. Dann wird die externe Funktion aufgerufen, indem die Spaltennamen dieser Tabelle übergeben werden. Die beiden Tabellenzeilen werden in einer Batch-Anforderung in der Anforderungsnutzlast als einzelner Lambda-Aufruf gesendet.

```
CREATE TABLE t_sum(c1 int, c2 int);
INSERT INTO t_sum VALUES (4,5), (6,7);
SELECT exfunc_sum(c1,c2) FROM t_sum;
 exfunc_sum
---------------
 9
 13
(2 rows)
```

Skalares Lambda-UDF-Beispiel unter Verwendung des RETRY_TIMEOUT-Attributs

Im folgenden Abschnitt finden Sie ein Beispiel für die Verwendung des Attributs RETRY_TIMEOUT in Lambda-UDFs.

AWS Lambda Für Funktionen gibt es Grenzwerte für die Parallelität, die Sie für jede Funktion festlegen können. Weitere Informationen zu Parallelitätslimits finden Sie unter Managing Concurrency for a Lambda function im AWS Lambda Developer Guide und im Beitrag Managing AWS Lambda Function Concurrency im Compute-Blog. AWS

Wenn die Anzahl der Anfragen, die von einer Lambda-UDF bedient werden, die Nebenläufigkeitslimits überschreitet, wird bei neuen Anfragen der Fehler `TooManyRequestsException` ausgegeben. Die Lambda-UDF versucht es bei diesem Fehler so lange erneut, bis die Summe aller Verzögerungen zwischen den an die Lambda-Funktion gesendeten Anforderungen dem von Ihnen festgelegten Wert RETRY_TIMEOUT entspricht oder diesen überschreitet. Der Standardwert für RETRY_TIMEOUT ist 20 000 Millisekunden.

Im folgenden Beispiel wird die Lambda-Funktion mit dem Namen erstellt `exfunc_sleep_3`. Diese Funktion verarbeitet die Nutzdaten der Anfrage, iteriert über jede Zeile und konvertiert die Eingabe in Großbuchstaben. Dann wartet sie 3 Sekunden und gibt das Ergebnis zurück. Die für diese Lambda-Funktion verwendete Sprache ist Python 3.8.

Die Anzahl der Zeilen im Ergebnis-Array entspricht der Anzahl der in der Nutzlast der Anfrage empfangenen Zeilen. Die JSON-Antwort-Nutzlast muss die Ergebnisdaten im Feld `results`

enthalten, damit sie von der externen Funktion erkannt werden kann. Das Feld `arguments` in der Anforderung, die an die Lambda Funktion gesendet wird, enthält die Datennutzlast. Bei einer Batch-Anforderung können mehrere Zeilen in den Nutzdaten enthalten sein.

Das Nebenläufigkeitslimit für diese Funktion ist speziell auf 1 für die reservierte Nebenläufigkeit festgelegt, um die Verwendung des Attributs RETRY_TIMEOUT zu demonstrieren. Wenn das Attribut auf 1 gesetzt ist, kann die Lambda-Funktion jeweils nur eine Anfrage bedienen.

```
import json
import time
def lambda_handler(event, context):
    t1 = event['arguments']
    # 'len(t1)' represents the number of rows in the request payload.
    # The number of results in the response payload should be the same as the number of
 rows received.
    resp = [None]*len(t1)

    # Iterating over all rows in the request payload.
    for i, x in enumerate(t1):
        # Iterating over all the values in a single row.
        for j, y in enumerate(x):
            resp[i] = y.upper()

    time.sleep(3)
    ret = dict()
    ret['results'] = resp
    ret_json = json.dumps(ret)
    return ret_json
```

Im Folgenden werden zwei weitere Beispiele für das RETRY_TIMEOUT-Attribut beschrieben. Sie rufen jeweils eine einzelne Lambda-UDF auf. Beim Aufrufen der Lambda-UDF wird in jedem Beispiel dieselbe SQL-Abfrage ausgeführt, um die Lambda-UDF von zwei nebenläufigen Datenbanksitzungen aus aufzurufen. Wenn die erste Abfrage, die die Lambda-UDF aufruft, von der UDF bedient wird, wird für die zweite Abfrage der Fehler `TooManyRequestsException` angezeigt. Dieses Ergebnis tritt auf, weil Sie die reservierte Nebenläufigkeit in der UDF speziell auf 1 gesetzt haben. Informationen zum Festlegen der reservierten Nebenläufigkeit für Lambda-Funktionen finden Sie unter Konfigurieren reservierter Gleichzeitigkeit.

Im folgenden ersten Beispiel wird das RETRY_TIMEOUT-Attribut für die Lambda-UDF auf 0 Millisekunden gesetzt. Wenn die Lambda-Anforderung eine Ausnahme von der Lambda-Funktion

erhält, führt Amazon Redshift keine Wiederholungsversuche durch. Dieses Ergebnis tritt auf, weil das Attribut RETRY_TIMEOUT auf 0 gesetzt ist.

```
CREATE OR REPLACE EXTERNAL FUNCTION exfunc_upper(varchar)
RETURNS varchar
VOLATILE
LAMBDA 'exfunc_sleep_3'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test'
RETRY_TIMEOUT 0;
```

Wenn RETRY_TIMEOUT auf 0 gesetzt ist, können Sie die folgenden zwei Abfragen in verschiedenen Datenbanksitzungen ausführen, um unterschiedliche Ergebnisse zu erhalten.

Die erste SQL-Abfrage, die die Lambda-UDF verwendet, wird erfolgreich ausgeführt.

```
select exfunc_upper('Varchar');
 exfunc_upper
 --------------
 VARCHAR
(1 row)
```

Für die zweite Abfrage, die in einer separaten Datenbanksitzung zur gleichen Zeit ausgeführt wird, wird der Fehler `TooManyRequestsException` angezeigt.

```
select exfunc_upper('Varchar');
ERROR:  Rate Exceeded.; Exception: TooManyRequestsException; ShouldRetry: 1
DETAIL:
-----------------------------------------------
error:  Rate Exceeded.; Exception: TooManyRequestsException; ShouldRetry: 1
code:      32103
context:query:     0
location:  exfunc_client.cpp:102
process:   padbmaster [pid=26384]
-----------------------------------------------
```

Im folgenden zweiten Beispiel wird das RETRY_TIMEOUT-Attribut für die Lambda-UDF auf 3 000 Millisekunden gesetzt. Selbst wenn die zweite Abfrage gleichzeitig ausgeführt wird, wiederholt die Lambda-UDF den Vorgang, bis die Gesamtverzögerung 3 000 Millisekunden beträgt. Daher werden beide Abfragen erfolgreich ausgeführt.

```
CREATE OR REPLACE EXTERNAL FUNCTION exfunc_upper(varchar)
```

```
RETURNS varchar
VOLATILE
LAMBDA 'exfunc_sleep_3'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test'
RETRY_TIMEOUT 3000;
```

Wenn RETRY_TIMEOUT auf 3 000 Millisekunden eingestellt ist, können Sie die folgenden beiden Abfragen in verschiedenen Datenbanksitzungen ausführen, um dieselben Ergebnisse zu erhalten.

Die erste SQL-Abfrage, die die Lambda-UDF verwendet, wird erfolgreich ausgeführt.

```
select exfunc_upper('Varchar');
 exfunc_upper
 --------------
 VARCHAR
(1 row)
```

Die zweite Abfrage wird gleichzeitig ausgeführt, und die Lambda-UDF versucht es erneut, bis die Gesamtverzögerung 3 000 Millisekunden beträgt.

```
select exfunc_upper('Varchar');
 exfunc_upper
--------------
 VARCHAR
(1 row)
```

Skalares Lambda-UDF-Beispiel mit einer Python-Lambda-Funktion

Im folgenden Beispiel wird eine externe Funktion namens `exfunc_multiplication` erstellt, die Zahlen multipliziert und eine Ganzzahl zurückgibt. In diesem Beispiel sind die Erfolgs- und `error_msg`-Felder in der Lambda-Antwort enthalten. Das Erfolgsfeld wird auf false gesetzt, wenn ein Ganzzahlüberlauf im Multiplikationsergebnis vorliegt, und die Meldung `error_msg` wird auf `Integer multiplication overflow` gesetzt. Die `exfunc_multiplication`-Funktion nimmt drei Ganzzahlen als Eingabeargumente und gibt die Summe als Ganzzahl aus.

Der Name der Lambda-Funktion, die aufgerufen wird, lautet `lambda_multiplication`. Die für diese Lambda-Funktion verwendete Sprache ist Python 3.8. Stellen Sie sicher, dass Sie die IAM-Rolle angeben.

```
CREATE EXTERNAL FUNCTION exfunc_multiplication(int, int, int)
RETURNS INT
```

```
VOLATILE
LAMBDA 'lambda_multiplication'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';
```

Die Lambda-Funktion verarbeitet die Nutzlast der Anfrage und iteriert über jede Zeile. Die Werte in einer einzelnen Zeile werden multipliziert, um das Ergebnis für diese Zeile zu berechnen, das dann in der Antwortliste gespeichert wird. In diesem Beispiel wird ein boolescher Erfolgswert verwendet, der standardmäßig auf true gesetzt ist. Wenn das Multiplikationsergebnis für eine Zeile einen Ganzzahlüberlauf aufweist, wird der Erfolgswert auf false gesetzt. Dann bricht die Iterationsschleife ab.

Wenn der Erfolgswert beim Erstellen der Antwort-Nutzdaten falsch ist, fügt die folgende Lambda-Funktion das Feld `error_msg` in die Nutzdaten ein. Es setzt auch die Fehlermeldung auf `Integer multiplication overflow`. Wenn der Erfolgswert true ist, werden die Ergebnisdaten in das Ergebnisfeld eingefügt. Die Anzahl der Zeilen im Ergebnis-Array, sofern vorhanden, entspricht der Anzahl der in der Nutzlast der Anfrage empfangenen Zeilen.

Das Feld „arguments“ in der Anforderung, die an die Lambda Funktion gesendet wird, enthält die Datennutzlast. Bei einer Batch-Anforderung können mehrere Zeilen in den Nutzdaten vorhanden sein. Die folgende Lambda-Funktion iteriert über alle Zeilen in der Nutzlast der Anfragedaten und iteriert einzeln über alle Werte innerhalb einer einzelnen Zeile.

```
import json
def lambda_handler(event, context):
    t1 = event['arguments']
    # 'len(t1)' represents the number of rows in the request payload.
    # The number of results in the response payload should be the same as the number of
 rows received.
    resp = [None]*len(t1)

    # By default success is set to 'True'.
    success = True
    # Iterating over all rows in the request payload.
    for i, x in enumerate(t1):
        mul = 1
        # Iterating over all the values in a single row.
        for j, y in enumerate(x):
            mul = mul*y

        # Check integer overflow.
        if (mul >= 9223372036854775807 or mul <= -9223372036854775808):
```

```
            success = False
            break
        else:
            resp[i] = mul
    ret = dict()
    ret['success'] = success
    if not success:
        ret['error_msg'] = "Integer multiplication overflow"
    else:
        ret['results'] = resp
    ret_json = json.dumps(ret)

    return ret_json
```

Im folgenden Beispiel wird die externe Funktion mit Literalwerten aufgerufen.

```
SELECT exfunc_multiplication(8, 9, 2);
  exfunc_multiplication
---------------------------
          144
(1 row)
```

Im folgenden Beispiel wird eine Tabelle mit dem Namen t_multi mit drei Spalten, c1, c2 und c3, vom Datentyp Ganzzahl erstellt. Dann wird die externe Funktion aufgerufen, indem die Spaltennamen dieser Tabelle übergeben werden. Die Daten werden so eingefügt, dass sie einen Ganzzahlüberlauf verursachen, um zu zeigen, wie der Fehler verteilt ist.

```
CREATE TABLE t_multi (c1 int, c2 int, c3 int);
INSERT INTO t_multi VALUES (2147483647, 2147483647, 4);
SELECT exfunc_multiplication(c1, c2, c3) FROM t_multi;
DETAIL:
  -----------------------------------------------
  error:  Integer multiplication overflow
  code:      32004context:
  context:
  query:     38
  location:  exfunc_data.cpp:276
  process:   query2_16_38 [pid=30494]
  -----------------------------------------------
```

CREATE EXTERNAL SCHEMA

Erstellt ein neues externes Schema in der aktuellen Datenbank. Sie können dieses externe Schema verwenden, um sich mit Amazon-RDS-for-PostgreSQL- oder mit Amazon-Aurora-for-PostgreSQL-kompatible-Edition-Datenbanken zu verbinden. Sie können auch ein externes Schema erstellen, das auf eine Datenbank in einem externen Datenkatalog wie AWS Glue Athena oder auf eine Datenbank in einem Apache Hive-Metastore wie Amazon EMR verweist.

Der Besitzer dieses Schemas gibt den Befehl CREATE EXTERNAL SCHEMA aus. Mit dem Befehl ALTER SCHEMA können Sie den Besitzer eines externen Schemas ändern. Mit dem Befehl GRANT gewähren Sie anderen Benutzern oder Benutzergruppen Zugriff auf das Schema.

Sie können die Befehle GRANT oder REVOKE nicht für Berechtigungen in einer externen Tabelle verwenden. Gewähren oder widerrufen Sie stattdessen die Berechtigungen für das externe Schema.

> **Note**
>
> Wenn Sie derzeit externe Redshift-Spectrum-Tabellen im Amazon-Athena-Datenkatalog haben, können Sie Ihren Athena-Datenkatalog zu einem -Datenkatalog migrieren AWS Glue Data Catalog. Um den AWS Glue Datenkatalog mit Redshift Spectrum zu verwenden, müssen Sie möglicherweise Ihre AWS Identity and Access Management (IAM-) Richtlinien ändern. Weitere Informationen finden Sie unter Upgrade auf den AWS Glue Datenkatalog im Athena-Benutzerhandbuch.

Um Details zu externen Schemata anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_EXTERNAL_SCHEMAS aus.

Syntax

Die folgende Syntax beschreibt den Befehl CREATE EXTERNAL SCHEMA, der verwendet wird, um Daten mithilfe eines externen Datenkatalogs zu referenzieren. Weitere Informationen finden Sie unter Abfrage externer Daten mit Amazon Redshift Spectrum.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM { [ DATA CATALOG ] | HIVE METASTORE | POSTGRES | MYSQL | KINESIS | MSK |
 REDSHIFT }
[ DATABASE 'database_name' ]
[ SCHEMA 'schema_name' ]
[ REGION 'aws-region' ]
```

```
[ URI 'hive_metastore_uri' [ PORT port_number ] ]
IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>' }
[ SECRET_ARN 'ssm-secret-arn' ]
[ AUTHENTICATION { none | iam } ]
[ CLUSTER_ARN 'arn:aws:kafka:<region>:<AWS-Konto-id>:cluster/msk/<cluster uuid>' ]
[ CATALOG_ROLE { 'SESSION' | 'catalog-role-arn-string' } ]
[ CREATE EXTERNAL DATABASE IF NOT EXISTS ]
[ CATALOG_ID 'Amazon Web Services account ID containing Glue or Lake Formation
 database' ]
```

Die folgende Syntax beschreibt den Befehl CREATE EXTERNAL SCHEMA, der verwendet wird, um Daten mithilfe einer Verbundabfrage an RDS POSTGRES oder Aurora PostgreSQL zu referenzieren. Sie können auch ein externes Schema erstellen, das auf Streaming-Quellen wie Kinesis Data Streams verweist. Weitere Informationen finden Sie unter Abfragen von Daten mit Verbundabfragen in Amazon Redshift.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM POSTGRES
DATABASE 'federated_database_name' [SCHEMA 'schema_name']
URI 'hostname' [ PORT port_number ]
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>' }
SECRET_ARN 'ssm-secret-arn'
```

Die folgende Syntax beschreibt den Befehl CREATE EXTERNAL SCHEMA, der verwendet wird, um Daten mithilfe einer Verbundabfrage an RDS MySQL oder Aurora MySQL zu referenzieren. Weitere Informationen finden Sie unter Abfragen von Daten mit Verbundabfragen in Amazon Redshift.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM MYSQL
DATABASE 'federated_database_name'
URI 'hostname' [ PORT port_number ]
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>' }
SECRET_ARN 'ssm-secret-arn'
```

Die folgende Syntax beschreibt den Befehl CREATE EXTERNAL SCHEMA, der verwendet wird, um Daten in einem Kinesis-Stream zu referenzieren. Weitere Informationen finden Sie unter Streaming-Erfassung.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] schema_name
FROM KINESIS
```

```
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>' }
```

Die folgende Syntax beschreibt den Befehl CREATE EXTERNAL SCHEMA, der verwendet wird, um auf den Cluster von Amazon Managed Streaming for Apache Kafka und die zugehörigen Themen zu verweisen, aus denen Daten erfasst werden sollen. CLUSTER_ARN gibt den Amazon-MSK-Cluster an, aus dem Sie Daten lesen. Weitere Informationen finden Sie unter Streaming-Erfassung.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] schema_name
FROM MSK
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>' }
AUTHENTICATION { none | iam }
CLUSTER_ARN 'msk-cluster-arn';
```

Die folgende Syntax beschreibt den Befehl CREATE EXTERNAL SCHEMA, der verwendet wird, um Daten mithilfe einer datenbankübergreifenden Abfrage zu referenzieren.

```
CREATE EXTERNAL SCHEMA local_schema_name
FROM  REDSHIFT
DATABASE 'redshift_database_name' SCHEMA 'redshift_schema_name'
```

Parameter

IF NOT EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass das Schema vorhanden ist, statt mit einem Fehler beendet zu werden, wenn das angegebene Schema bereits vorhanden ist. Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn CREATE EXTERNAL SCHEMA versucht, ein Schema zu erstellen, das bereits vorhanden ist.

local_schema_name

Der Name des neuen externen Schemas. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

FROM [DATA CATALOG] | HIVE METASTORE | POSTGRES | MYSQL | KINESIS | MSK | REDSHIFT

Ein Schlüsselwort, das angibt, wo sich die externe Datenbank befindet.

DATA CATALOG gibt an, dass die externe Datenbank im Athena-Datenkatalog oder dem definiert ist AWS Glue Data Catalog.

Wenn die externe Datenbank in einem externen Datenkatalog in einer anderen AWS -Region definiert ist, ist der Parameter REGION erforderlich. DATA CATALOG ist der Standardwert.

HIVE METASTORE gibt an, dass die externe Datenbank in einem Apache Hive-Metastore definiert ist. Wenn HIVE METASTORE angegeben ist, ist der URI erforderlich.

POSTGRES gibt an, dass die externe Datenbank in RDS PostgreSQL oder Aurora PostgreSQL definiert ist.

MYSQL zeigt an, dass die externe Datenbank in RDS MySQL oder Aurora MySQL definiert ist.

KINESIS gibt an, dass die Datenquelle ein Stream aus dem Kinesis Data Streams ist.

MSK gibt an, dass die Datenquelle ein Thema aus Amazon MSK ist.

VON REDSHIFT

Ein Schlüsselwort, das angibt, dass sich die Datenbank in Amazon Redshift befindet.

DATABASE 'redshift_database_name' SCHEMA 'redshift_schema_name'

Der Name der Amazon-Redshift-Datenbank.

Der redshift_schema_name gibt das Schema in Amazon Redshift an. Standardmäßig lautet der redshift_schema_name auf `public`.

DATABASE 'federated_database_name'

Ein Schlüsselwort, das den Namen der externen Datenbank in einer unterstützten PostgreSQL- oder MySQL-Datenbank-Engine angibt.

[SCHEMA 'schema_name']

Der schema_name gibt das Schema in einer unterstützten PostgreSQL-Datenbank-Engine an. Der Standard-schema_name ist `public`.

Sie können kein SCHEMA angeben, wenn Sie eine Verbundabfrage an eine unterstützte MySQL-Datenbank-Engine einrichten.

REGION „aws-region“

Wenn die externe Datenbank in einem Athena-Datenkatalog oder in der AWS Region definiert ist AWS Glue Data Catalog, in der sich die Datenbank befindet. Dieser Parameter ist erforderlich, wenn die Datenbank in einem externen Datenkatalog definiert ist.

URI „hive_metastore_uri“ [PORT port_number]

Der Hostname-URI und die Portnummer einer unterstützten PostgreSQL- oder MySQL-Datenbank-Engine. Der Hostname ist der Haarknoten des Replikatsatzes. Der Endpunkt muss vom Amazon-Redshift-Cluster aus erreichbar (routingfähig) sein. Der Standardwert für port_number von PostgreSQL lautet 5432. Die Standarwert für port_number von MySQL lautet 3306.

Wenn sich die Datenbank in einem Hive-Metastore befindet, geben Sie den URI und optional die Portnummer des Metastore an. Die Standard-Portnummer ist 9083.

Ein URI enthält keine Protokollspezifikation („http://“). Beispiel für einen gültigen URI: `uri '172.10.10.10'`.

> **Note**
>
> Die unterstützte PostgreSQL- oder MySQL-Datenbank-Engine muss sich in der gleichen VPC befinden wie Ihr Amazon-Redshift-Cluster. Erstellen Sie eine Sicherheitsgruppe, die Amazon Redshift und RDS PostgreSQL oder Aurora PostgreSQL verbindet.

IAM_ROLE { default | 'SESSION' | 'arn:aws:iam::*<AWS-Konto-id>*:role/*<role-name>*' }

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE EXTERNAL SCHEMA-Befehl ausgeführt wird.

Verwenden Sie `'SESSION'`, wenn Sie über eine Verbundidentität eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellen und über das mit diesem Befehl erstellte externe Schema auf die Tabellen zugreifen. Weitere Informationen finden Sie unter Verwenden einer Verbundidentität zur Verwaltung des Amazon-Redshift-Zugriffs auf lokale Ressourcen und externe Amazon-Redshift-Spectrum-Tabellen. Darin wird erläutert, wie Sie eine Verbundidentität konfigurieren. Beachten Sie, dass diese Konfiguration (`'SESSION'` anstelle des ARN zu verwenden) nur verwendet werden kann, wenn das Schema mit `DATA CATALOG` erstellt wurde.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Die IAM-Rolle muss mindestens die Berechtigung besitzen, eine LIST-Operation für den Amazon-S3-Bucket auszuführen, auf den zugegriffen werden soll, und eine GET-Operation für die Amazon-S3-Objekte, die der Bucket enthält.

Nachfolgend ist die Syntax für die IAM_ROLE-Parameterzeichenfolge für einen einzelnen ARN aufgeführt.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Sie können Rollen miteinander verketten. Auf diese Weise kann der Cluster eine andere IAM-Rolle annehmen, die möglicherweise zu einem anderen Konto gehört. Es können bis zu 10 Rollen miteinander verkettet werden. Ein Beispiel für die Verkettung von Rollen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.

Fügen Sie dieser IAM-Rolle eine IAM-Berechtigungsrichtlinie ähnlich der folgenden an:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Informationen zu den Schritten für das Erstellen einer IAM-Rolle zur Verwendung mit der Verbundabfrage finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.

> **Note**
>
> Fügen Sie keine Leerzeichen in die Liste der verketteten Rollen ein.

Nachfolgend finden Sie die Syntax für die Verkettung von drei Rollen.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

SECRET_ARN 'ssm-secret-arn'

Der Amazon-Ressourcenname (ARN) eines unterstützten PostgreSQL- oder MySQL-Datenbank-Engine-Geheimnisses, das mit erstellt wurde. AWS Secrets Manager Informationen zum Erstellen und Abrufen eines ARNs für ein Secret finden Sie unter Erstellen eines Basis-Secrets und unter Abrufen des Secret-Werts im AWS Secrets Manager -Benutzerhandbuch.

CATALOG_ROLE { 'SESSION' | catalog-role-arn-string}

Verwenden Sie `'SESSION'`, um mithilfe einer Verbundidentität eine Verbindung mit Ihrem Amazon-Redshift-Cluster herzustellen, um den Datenkatalog zu authentifizieren und zu autorisieren. Weitere Informationen zur Durchführung der Schritte für eine Verbundidentität finden Sie unter Verwenden einer Verbundidentität zur Verwaltung des Amazon-Redshift-Zugriffs auf lokale Ressourcen und externe Amazon-Redshift-Spectrum-Tabellen. Beachten Sie, dass die `'SESSION'`-Rolle nur verwendet werden kann, wenn das Schema in DATA CATALOG erstellt wurde.

Verwenden Sie den Amazon-Ressourcenname (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung für den Datenkatalog verwendet wird.

Wenn CATALOG_ROLE nicht angegeben wird, verwendet Amazon Redshift die angegebene IAM_ROLE. Die Katalogrolle muss über die Berechtigung verfügen, auf den Datenkatalog in AWS Glue oder Athena zuzugreifen. Weitere Informationen finden Sie unter IAM-Richtlinien für Amazon Redshift Spectrum.

Nachfolgend ist die Syntax für die CATALOG_ROLE-Parameterzeichenfolge für einen einzelnen ARN aufgeführt.

```
CATALOG_ROLE 'arn:aws:iam::<aws-account-id>:role/<catalog-role>'
```

Sie können Rollen miteinander verketten. Auf diese Weise kann der Cluster eine andere IAM-Rolle annehmen, die möglicherweise zu einem anderen Konto gehört. Es können bis zu 10 Rollen miteinander verkettet werden. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.

> **Note**
>
> Die Liste der verketteten Rollen darf keine Leerstellen enthalten.

Nachfolgend finden Sie die Syntax für die Verkettung von drei Rollen.

```
CATALOG_ROLE 'arn:aws:iam::<aws-account-id>:role/<catalog-role-1-name>,arn:aws:iam::<aws-account-id>:role/<catalog-role-2-name>,arn:aws:iam::<aws-account-id>:role/<catalog-role-3-name>'
```

CREATE EXTERNAL DATABASE IF NOT EXISTS

Eine Klausel, die eine externe Datenbank mit dem Namen erstellt, der im DATABASE-Argument angegeben ist, wenn die angegebene externe Datenbank nicht vorhanden ist. Wenn die angegebene externe Datenbank vorhanden ist, führt der Befehl keine Änderungen aus. In diesem Fall gibt der Befehl die Meldung zurück, dass die externe Datenbank vorhanden ist, statt mit einem Fehler beendet zu werden.

> **Note**
>
> CREATE EXTERNAL DATABASE IF NOT EXISTS kann nicht mit HIVE METASTORE verwendet werden.
> Um CREATE EXTERNAL DATABASE IF NOT EXISTS mit einem für AWS Lake Formation aktivierten Datenkatalog zu verwenden, benötigen Sie die Berechtigung `CREATE_DATABASE` für den Datenkatalog.

CATALOG_ID ‚Amazon-Web-Services-Konto-ID mit Glue- oder Lake-Formation-Datenbank'

Die Konto-ID, in der die Datenkatalogdatenbank gespeichert ist.

`CATALOG_ID` kann nur angegeben werden, wenn Sie planen, für die Authentifizierung und Autorisierung beim Datenkatalog mithilfe einer Verbundidentität eine Verbindung mit Ihrem

Amazon-Redshift-Cluster oder Amazon Redshift Serverless herzustellen, indem Sie eine der folgenden Einstellungen festlegen:

- `CATALOG_ROLE` auf `'SESSION'`
- `IAM_ROLE` auf `'SESSION'` und `'CATALOG_ROLE'` auf die Standardeinstellung festgelegt

Weitere Informationen zur Durchführung der Schritte für Verbundidentität finden Sie unter Verwenden einer Verbundidentität zur Verwaltung des Amazon-Redshift-Zugriffs auf lokale Ressourcen und externe Amazon-Redshift-Spectrum-Tabellen

AUTHENTICATION

Der für die Streaming-Erfassung definierte Authentifizierungstyp. Die Streaming-Erfassung mit Authentifizierungstypen kann mit Amazon Managed Streaming for Apache Kafka verwendet werden. Es stehen folgende `AUTHENTICATION`-Typen zur Verfügung:

- none: Gibt an, dass es keinen Authentifizierungsschritt gibt.
- iam: Gibt an, dass eine IAM-Authentifizierung erfolgt. Stellen Sie bei Auswahl dieser Option sicher, dass die IAM-Rolle über Berechtigungen für die IAM-Authentifizierung verfügt. Weitere Informationen über die Definition des externen Schemas finden Sie unter Erste Schritte mit der Streaming-Erfassung aus Amazon Managed Streaming for Apache Kafka.

CLUSTER_ARN

Für die Streaming-Erfassung die Cluster-ID für den Cluster von Amazon Managed Streaming for Apache Kafka, von dem aus Sie streamen. Weitere Informationen finden Sie unter Streaming-Erfassung.

Nutzungshinweise

Informationen über Beschränkungen bei der Verwendung des Athena-Datenkatalogs finden Sie unter Athena-Limits in der Allgemeine AWS-Referenz.

Informationen zu den Beschränkungen bei der AWS Glue Data Catalog Verwendung von finden Sie unter AWS Glue Grenzwerte in der Allgemeine AWS-Referenz.

Diese Begrenzungen gelten nicht für einen Hive-Metastore.

Pro Datenbank sind höchstens 9 900 Schemata zulässig. Weitere Informationen finden Sie unter Kontingente und Einschränkungen im Amazon-Redshift-Verwaltungshandbuch.

Verwenden Sie den DROP SCHEMA-Befehl, um die Registrierung des Schemas aufzuheben.

Um sich die Details zu externen Schemata anzeigen zu lassen, fragen Sie die folgenden Systemansichten ab:

- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_EXTERNAL_COLUMNS

Beispiele

Im folgenden Beispiel wird ein externes Schema unter Verwendung einer Datenbank in einem Athena-Datenkatalog namens `sampledb` in der Region USA West (Oregon) erstellt. Verwenden Sie dieses Beispiel mit einem Athena- oder AWS Glue Datenkatalog.

```
create external schema spectrum_schema
from data catalog
database 'sampledb'
region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole';
```

Im folgenden Beispiel werden ein externes Schema und eine neue externe Datenbank namens erstellt `spectrum_db`.

```
create external schema spectrum_schema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
create external database if not exists;
```

Im folgenden Beispiel wird ein externes Schema mittels einer Hive-Metastore-Datenbank namens erstellt `hive_db`.

```
create external schema hive_schema
from hive metastore
database 'hive_db'
uri '172.10.10.10' port 99
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole';
```

Im folgenden Beispiel miteinander verketteter Rollen wird die Rolle `myS3Role` für den Zugriff auf Amazon S3 und `myAthenaRole` für den Datenkatalogzugriff verwendet. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.

```
create external schema spectrum_schema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myRedshiftRole,arn:aws:iam::123456789012:role/
myS3Role'
catalog_role 'arn:aws:iam::123456789012:role/myAthenaRole'
create external database if not exists;
```

Im folgenden Beispiel wird ein externes Schema erstellt, das auf eine Aurora-PostgreSQL-Datenbank verweist.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] myRedshiftSchema
FROM POSTGRES
DATABASE 'my_aurora_db' SCHEMA 'my_aurora_schema'
URI 'endpoint to aurora hostname' PORT 5432
IAM_ROLE 'arn:aws:iam::123456789012:role/MyAuroraRole'
SECRET_ARN 'arn:aws:secretsmanager:us-east-2:123456789012:secret:development/
MyTestDatabase-AbCdEf'
```

Im folgenden Beispiel wird ein externes Schema erstellt, das auf die im Konsumenten-Cluster importierte sales_db verweist.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA 'public';
```

Im folgenden Beispiel wird ein externes Schema erstellt, das auf eine Aurora-MySQL-Datenbank verweist.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] myRedshiftSchema
FROM MYSQL
DATABASE 'my_aurora_db'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyAuroraRole'
SECRET_ARN 'arn:aws:secretsmanager:us-east-2:123456789012:secret:development/
MyTestDatabase-AbCdEf'
```

CREATE EXTERNAL TABLE

Erstellt eine neue externe Tabelle im angegebenen Schema. Alle externen Tabellen müssen in einem externen Schema erstellt werden. Externe Schemata und externe Tabellen unterstützen keine Suchpfade. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

Zusätzlich zu externen Tabellen, die mit dem Befehl CREATE EXTERNAL TABLE erstellt wurden, kann Amazon Redshift auf externe Tabellen verweisen, die in einem AWS Lake Formation OR-Katalog AWS Glue oder einem Apache Hive-Metastore definiert sind. Verwenden Sie den Befehl CREATE EXTERNAL SCHEMA, um eine externe Datenbank zu registrieren, die im externen Katalog definiert ist, und um die externen Tabellen für die Verwendung in Amazon Redshift zur Verfügung zu stellen. Wenn die externe Tabelle in einem AWS Lake Formation Or-Katalog AWS Glue - oder Hive-Metastore vorhanden ist, müssen Sie die Tabelle nicht mit CREATE EXTERNAL TABLE erstellen. Um externe Tabellen anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_EXTERNAL_TABLES aus.

Durch das Ausführen des Befehls CREATE EXTERNAL TABLE AS können Sie eine externe Tabelle basierend auf der Spaltendefinition aus einer Abfrage erstellen und die Ergebnisse dieser Abfrage in Amazon S3 schreiben. Die Ergebnisse liegen im Apache Parquet- oder im Textformat mit Trennzeichen vor. Wenn die externe Tabelle über mindestens einen Partitionsschlüssel verfügt, partitioniert Amazon Redshift neue Dateien entsprechend diesen Partitionsschlüsseln und registriert neue Partitionen automatisch im externen Katalog. Weitere Hinweise zu CREATE EXTERNAL TABLE AS finden Sie unter Nutzungshinweise.

Sie können eine externe Tabelle mit der gleichen SELECT-Syntax abfragen, die Sie auch für andere Amazon-Redshift-Tabellen verwenden. Sie können auch die INSERT-Syntax verwenden, um neue Dateien in den Speicherort der externen Tabelle auf Amazon S3 zu schreiben. Weitere Informationen finden Sie unter INSERT (externe Tabelle).

Zum Erstellen einer Ansicht mit einer externen Tabelle fügen Sie die Klausel WITH NO SCHEMA BINDING in die CREATE VIEW-Aussage ein.

CREATE EXTERNAL TABLE kann nicht innerhalb einer Transaktion (BEGIN … END) ausgeführt werden. Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Erforderliche Berechtigungen

Zum Erstellen externer Tabellen müssen Sie der Eigentümer des externen Schemas oder ein Superuser sein. Mit dem Befehl ALTER SCHEMA können Sie den Besitzer eines externen Schemas ändern. Der Zugriff auf externe Tabellen wird durch den Zugriff auf die externen Schemata gesteuert.

Sie können für eine externe Tabelle keine Berechtigungen gewähren (GRANT) oder widerrufen (REVOKE). Sie gewähren oder widerrufen stattdessen das Recht USAGE für das externe Schema.

Die Nutzungshinweise enthalten zusätzliche Informationen zu spezifischen Berechtigungen für externe Tabellen.

Syntax

```
CREATE EXTERNAL TABLE
external_schema.table_name
(column_name data_type [, …] )
[ PARTITIONED BY (col_name data_type [, … ] )]
[ { ROW FORMAT DELIMITED row_format |
  ROW FORMAT SERDE 'serde_name'
  [ WITH SERDEPROPERTIES ( 'property_name' = 'property_value' [, ...] ) ] } ]
STORED AS file_format
LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }
[ TABLE PROPERTIES ( 'property_name'='property_value' [, ...] ) ]
```

Im Folgenden finden Sie die Syntax für CREATE EXTERNAL TABLE AS.

```
CREATE EXTERNAL TABLE
external_schema.table_name
[ PARTITIONED BY (col_name [, … ] ) ]
[ ROW FORMAT DELIMITED row_format ]
STORED AS file_format
LOCATION { 's3://bucket/folder/' }
[ TABLE PROPERTIES ( 'property_name'='property_value' [, ...] ) ]
 AS
 { select_statement }
```

Parameter

external_schema.table_name

Der Name der Tabelle, die erstellt werden soll, qualifiziert durch einen externen Schemanamen. Externe Tabellen müssen in einem externen Schema erstellt werden. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

Die maximale Länge des Tabellennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Sie können UTF-8-Multibyte-Zeichen bis zu einer Länge von vier Bytes

verwenden. Amazon Redshift erzwingt ein Limit von 9.900 Tabellen pro Cluster, einschließlich benutzerdefinierter temporärer Tabellen und temporärer Tabellen, die von Amazon Redshift während der Abfrageverarbeitung oder Systemwartung erstellt werden. Optional können Sie die Tabelle mit dem Datenbanknamen qualifizieren. Im folgenden Beispiel ist der Datenbankname `spectrum_db`, der Name des externen Schemas `spectrum_schema` und der Tabellenname `test`.

```
create external table spectrum_db.spectrum_schema.test (c1 int)
stored as parquet
location 's3://mybucket/myfolder/';
```

Wenn die angegebene Datenbank oder das angegebene Schema nicht vorhanden sind, wird die Tabelle nicht erstellt, und die Anweisung gibt einen Fehler zurück. Sie können in den Systemdatenbanken `template0`, `template1`, `padb_harvest` oder `sys:internal` keine Tabellen oder Ansichten erstellen.

Der Tabellenname muss für das angegebene Schema eindeutig sein.

Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

(column_name data_type)

Der Name und der Datentyp jeder Spalte, die erstellt wird.

Die maximale Länge des Spaltennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Sie können UTF-8-Multibyte-Zeichen bis zu einer Länge von vier Bytes verwenden. Sie können keine Spaltennamen `"$path"` oder `"$size"` festlegen. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

Standardmäßig erstellt Amazon Redshift externe Tabellen mit den Pseudospalten `$path` und `$size`. Sie können die Erstellung von Pseudospalten für eine Sitzung deaktivieren, indem Sie den `spectrum_enable_pseudo_columns`-Konfigurationsparameter auf `false` setzen. Weitere Informationen finden Sie unter Pseudospalten .

Falls Pseudospalten aktiviert sind, können Sie höchstens 1.598 Spalten in einer einzelnen Tabelle definieren. Wenn Pseudospalten nicht aktiviert sind, können Sie höchstens 1.600 Spalten in einer einzelnen Tabelle definieren.

Wenn Sie eine „breite Tabelle“ erstellen, achten Sie darauf, dass Ihre Spaltenliste nicht die Zeilenbreitengrenzen überschreitet, um während der Verarbeitung von Lasten und Abfragen sofort Ergebnisse bereitzustellen. Weitere Informationen finden Sie unter Nutzungshinweise.

Für den Befehl CREATE EXTERNAL TABLE AS ist keine Spaltenliste erforderlich, da Spalten von der Abfrage abgeleitet werden.

data_type

Die folgenden Datentypen werden unterstützt:

- SMALLINT (INT2)
- INTEGER (INT, INT4)
- BIGINT (INT8)
- DECIMAL (NUMERIC)
- REAL (FLOAT4)
- DOUBLE PRECISION (FLOAT8)
- BOOLEAN (BOOL)
- CHAR (CHARACTER)
- VARCHAR (CHARACTER VARYING)
- VARBYTE (CHARACTER VARYING) – kann mit Parquet- und ORC-Datendateien und nur mit nicht partitionierten Spalten benutzt werden.
- DATE – kann nur mit Text-, Parquet- oder ORC-Datendateien oder als Partitionsspalte verwendet werden.
- TIMESTAMP (ZEITSTEMPEL)

Für DATE können Sie die im Folgenden beschriebenen Formate verwenden. Für Monatswerte, die in Ziffern dargestellt werden, werden die folgenden Formate unterstützt:

- `mm-dd-yyyy`, beispielsweise `05-01-2017`. Dies ist die Standardeinstellung.
- `yyyy-mm-dd`, wenn das Jahr mit mehr als zwei Ziffern dargestellt werden soll. Zum Beispiel `2017-05-01`.

Für Monatswerte, die mit einer Abkürzung aus drei Buchstaben dargestellt werden, werden die folgenden Formate unterstützt:

- `mmm-dd-yyyy`, beispielsweise `may-01-2017`. Dies ist die Standardeinstellung.
- `dd-mmm-yyyy`, wenn das Jahr mit mehr als zwei Ziffern dargestellt werden soll. Zum Beispiel `01-may-2017`.
- `yyyy-mmm-dd`, wenn das Jahr mit mehr als zwei Ziffern dargestellt werden soll. Zum Beispiel `2017-may-01`.

Für Jahreswerte, die konstant unter 100 liegen, wird das Jahr wie folgt berechnet:

- Wenn das Jahr weniger als 70 beträgt, wird das Jahr als das Jahr plus 2000 berechnet. Im Format `mm-dd-yyyy` wird das Datum 05-01-17 beispielsweise zu `05-01-2017` konvertiert.
- Wenn das Jahr weniger als 100 beträgt, aber mehr als 69, wird das Jahr als das Jahr plus 1900 berechnet. Im Format `mm-dd-yyyy` wird das Datum 05-01-89 beispielsweise zu `05-01-1989` konvertiert.
- Für Jahreswerte, die mit zwei Ziffern dargestellt werden, müssen führende Nullen hinzugefügt werden, um das Jahr mit vier Ziffern darzustellen.

Die Zeitstempelwerte in Textdateien müssen das Format `yyyy-mm-dd HH:mm:ss.SSSSSS` haben, wie der folgende Zeitstempelwert zeigt: `2017-05-01 11:30:59.000000`.

Die Länge einer VARCHAR-Spalte wird in Bytes definiert, nicht in Zeichen. Eine VARCHAR(12)-Spalte kann z. B. 12 Einzelbyte-Zeichen oder 6 Zeichen mit einer Länge von je 2 Bytes enthalten. Wenn Sie eine externe Tabelle abfragen, werden die Ergebnisse gekürzt, damit sie der definierten Spaltengröße entsprechen, ohne dass ein Fehler zurückgegeben wird. Weitere Informationen finden Sie unter Speicherung und Bereiche.

Um eine optimale Leistung zu erzielen, empfehlen wir, die kleinste Spaltengröße anzugeben, die ihren Daten entspricht. Verwenden Sie die Funktion OCTET_LENGTH, um die maximale Größe in Bytes für Werte in einer Spalte zu suchen. Das folgende Beispiel gibt die maximale Größe von Werten in der Spalte „E-Mail“ zurück.

```
select max(octet_length(email)) from users;

max
---
 62
```

PARTITIONED BY (col_name data_type [, …])

Eine Klausel, die eine partitionierte Tabelle mit einer oder mehreren Partitionsspalten festlegt. Für jede angegebene Kombination wird ein eigenes Datenverzeichnis verwendet. Dies kann die Abfrageleistung in einigen Fällen verbessern. In den Tabellendaten selbst sind keine partitionierten Spalten vorhanden. Wenn Sie einen Wert für col_name verwenden, der mit einer Tabellenspalte identisch ist, erhalten Sie einen Fehler.

Nachdem Sie eine partitionierte Tabelle erstellt haben, ändern Sie die Tabelle mit der Anweisung ALTER TABLE ... ADD PARTITION, um neue Partitionen im externen Katalog zu registrieren.

Wenn Sie eine Partition hinzufügen, definieren Sie den Speicherort des Unterordners auf Amazon S3, der die Partitionsdaten enthält.

Wenn die Tabelle `spectrum.lineitem_part` beispielsweise mit `PARTITIONED BY (l_shipdate date)` definiert ist, führen Sie den folgenden ALTER TABLE-Befehl aus, um eine Partition hinzuzufügen.

```
ALTER TABLE spectrum.lineitem_part ADD PARTITION (l_shipdate='1992-01-29')
LOCATION 's3://spectrum-public/lineitem_partition/l_shipdate=1992-01-29';
```

Wenn Sie CREATE EXTERNAL TABLE AS verwenden, müssen Sie ALTER TABLE...ADD PARTITION nicht ausführen. Amazon Redshift registriert neue Partitionen automatisch im externen Katalog. Amazon Redshift schreibt auch automatisch die entsprechenden Daten in Partitionen in Amazon S3, basierend auf dem Partitionsschlüssel oder den Schlüsseln, die in der Tabelle definiert sind.

Um Partitionen anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_EXTERNAL_PARTITIONS aus.

> **Note**
>
> Für den Befehl CREATE EXTERNAL TABLE AS müssen Sie den Datentyp der Partitionsspalte nicht angeben, da diese Spalte von der Abfrage abgeleitet wird.

ROW FORMAT DELIMITED rowformat

Eine Klausel, die das Format der zugrundeliegenden Daten angibt. Die möglichen Werte für rowformat sind wie folgt:

- LINES TERMINATED BY 'Trennzeichen'
- FIELDS TERMINATED BY 'Trennzeichen'

Geben Sie ein ASCII-Zeichen für 'delimiter' an. Sie können nicht druckbare ASCII-Zeichen mithilfe von Oktal-Code im Format `'\`*ddd*`'` festlegen, wobei *d* eine Oktalziffer (0–7) bis ‘\177’ ist. Im folgenden Beispiel wird das BEL (Bell)-Zeichen anhand Oktalziffern angegeben.

```
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\007'
```

Wenn ROW FORMAT ausgelassen wird, lautet das Standardformat DELIMITED FIELDS TERMINATED BY '\A' (Anfang des Headings) und LINES TERMINATED BY '\n' (Zeilenumbruch).

ROW FORMAT SERDE 'serde_name' , [WITH SERDEPROPERTIES ('property_name' = 'property_value' [, ...])]

Eine Klausel, die das SERDE-Format der zugrundeliegenden Daten angibt.

'serde_name'

Der Name der SerDe. Sie können die folgenden Formate angeben:

- org.apache.hadoop.hive.serde2. RegexSerDe
- com.amazonaws.glue.serde. GrokSerDe
- org.apache.hadoop.hive.serde2.OpenCSVSerde

  Dieser Parameter unterstützt die folgende SerDe Eigenschaft für OpencsVSerde:

  ```
  'wholeFile' = 'true'
  ```

  Setzen Sie die Eigenschaft `wholeFile` auf `true`, um Neue-Zeile-Zeichen (\n) innerhalb von Zeichenfolgen in Anführungszeichen für OpenCSV-Anforderungen richtig zu parsen.
- org.openx.data.jsonserde. JsonSerDe
  - Das JSON SERDE-Format unterstützt auch ION-Dateien.
  - JSON muss wohl geformt sein.
  - Zeitstempel in Ion und JSON müssen sich im Format ISO8601 befinden.
  - Dieser Parameter unterstützt die folgende SerDe Eigenschaft für JsonSerDe:

    ```
    'strip.outer.array'='true'
    ```

    Verarbeitet Ion/JSON-Dateien mit einem sehr großen Array in äußeren Klammern ([…]) so, als ob mehrere JSON-Datensätze innerhalb des Arrays enthalten sind.
- com.amazon.ionhiveserde. IonHiveSerDe

  Das Amazon ION-Format bietet neben Datentypen Text- und Binärformate. Bei einer externen Tabelle, die auf Daten im ION-Format verweist, weisen Sie jede Spalte in der externen Tabelle dem entsprechenden Element in den ION-Formatdaten zu. Weitere Informationen finden Sie unter Amazon Ion. Sie müssen gegebenenfalls auch die Ein- und Ausgabeformate angeben.

WITH SERDEPROPERTIES ('property_name' = 'property_value' [, ...])]

Optional können Sie Namen und Werte der Eigenschaften getrennt durch Kommas angeben.

Wenn ROW FORMAT ausgelassen wird, lautet das Standardformat DELIMITED FIELDS TERMINATED BY '\A' (Anfang des Headings) und LINES TERMINATED BY '\n' (Zeilenumbruch).

STORED AS Dateiformat

Das Dateiformat für Datendateien.

Gültige Formate sind folgende:

- PARQUET
- RCFILE (ColumnarSerDe nur zur Verwendung von Daten, nicht LazyBinaryColumnarSerDe)
- SEQUENCEFILE
- TEXTFILE (für Textdateien, einschließlich JSON-Dateien).
- ORC
- AVRO
- INPUTFORMAT 'input_format_classname' OUTPUTFORMAT 'output_format_classname'

Der Befehl CREATE EXTERNAL TABLE AS unterstützt nur zwei Dateiformate: TEXTFILE und PARQUET.

Geben Sie für INPUTFORMAT und OUTPUTFORMAT einen Klassennamen wie im folgenden Beispiel ein.

```
'org.apache.hadoop.mapred.TextInputFormat'
```

LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file'}

Der Pfad zum -Bucket oder Amazon-S3-Ordner enthält die Datendateien oder eine Manifestdatei, die eine Liste der Amazon-S3-Objektpfade enthält. Die Buckets müssen sich in derselben AWS Region wie der Amazon Redshift Redshift-Cluster befinden. Eine Liste der unterstützten AWS Regionen finden Sie unter. Überlegungen zu Amazon Redshift Spectrum

Wenn im Pfad ein Bucket oder Ordner wie `'s3://mybucket/custdata/'` angegeben wird, scannt Redshift Spectrum die Dateien im angegebenen Bucket oder Ordner und in allen

Unterordnern. Redshift Spectrum ignoriert verborgene Dateien sowie Dateien, die mit einem Punkt oder Unterstrich beginnen.

Wenn der Pfad eine Manifestdatei angibt, muss das `'s3://bucket/manifest_file'`-Argument explizit auf eine einzelne Datei verweisen, zum Beispiel `'s3://mybucket/manifest.txt'`. Es darf kein Schlüsselpräfix referenzieren.

Das Manifest ist eine Textdatei im JSON-Format, die die URL jeder Datei, die aus Amazon S3 geladen werden soll, sowie die Größe der jeweiligen Datei in Bytes auflistet. Die URL enthält den Bucket-Namen und den vollständigen Objektpfad für die Datei. Die im Manifest angegebenen Dateien können sich in verschiedenen Buckets befinden, aber alle Buckets müssen sich in derselben AWS Region wie der Amazon Redshift Redshift-Cluster befinden. Wenn eine Datei zweimal aufgelistet wird, wird die Datei zweimal geladen. Im folgenden Beispiel wird der JSON-Code für ein Manifest gezeigt, das drei Dateien lädt.

```
{
  "entries": [
    {"url":"s3://mybucket-alpha/custdata.1", "meta": { "content_length":
 5956875 } },
    {"url":"s3://mybucket-alpha/custdata.2", "meta": { "content_length":
 5997091 } },
    {"url":"s3://mybucket-beta/custdata.1", "meta": { "content_length": 5978675 } }
  ]
}
```

Sie können die Aufnahme einer bestimmten Datei obligatorisch machen. Fügen Sie dazu eine `mandatory`-Option auf Dateiebene in das Manifest ein. Wenn Sie eine externe Tabelle mit einer obligatorischen Datei abfragen, die fehlt, schlägt die SELECT-Anweisung fehl. Stellen Sie sicher, dass alle in der Definition der externen Tabelle enthaltenen Dateien vorhanden sind. Wenn sie nicht alle vorhanden sind, wird ein Fehler angezeigt, in dem die erste obligatorische Datei angezeigt wird, die nicht gefunden wird. Das folgende Beispiel zeigt den JSON für ein Manifest, wobei die `mandatory`-Option auf `true` eingestellt ist.

```
{
  "entries": [
    {"url":"s3://mybucket-alpha/custdata.1", "mandatory":true, "meta":
 { "content_length": 5956875 } },
    {"url":"s3://mybucket-alpha/custdata.2", "mandatory":false, "meta":
 { "content_length": 5997091 } },
    {"url":"s3://mybucket-beta/custdata.1", "meta": { "content_length": 5978675 } }
```

```
  ]
}
```

Um auf Dateien zu verweisen, die mit UNLOAD erstellt wurden, können Sie das mit UNLOAD mit dem MANIFEST-Parameter erstellte Manifest verwenden. Die Manifestdatei ist mit einer Manifestdatei für COPY aus Amazon S3 kompatibel, es werden jedoch andere Schlüssel verwendet. Nicht verwendete Schlüssel werden ignoriert.

TABLE PROPERTIES ('property_name'='property_value' [, ...])

Eine Klausel, die die Tabellendefinition für Tabelleneigenschaften festlegt.

> **Note**
>
> Bei Tabelleneigenschaften muss die Groß-/Kleinschreibung beachtet werden.

'compression_type'='value'

Eine Eigenschaft, die den Komprimierungstyp festlegt, der verwendet wird, wenn der Dateiname keine Erweiterung enthält. Wenn Sie diese Eigenschaft festlegen und eine Dateierweiterung vorhanden ist, wird die Erweiterung ignoriert und der von der Eigenschaft festgelegte Wert verwendet. Gültige Werte für den Komprimierungstyp sind folgende:

- bzip2
- gzip
- Keine
- snappy

'data_cleansing_enabled'='true / false’

Diese Eigenschaft legt fest, ob die Datenverarbeitung für die Tabelle aktiviert ist. Wenn 'data_cleansing_enabled' auf „true“ festgelegt ist, ist die Datenverarbeitung für die Tabelle aktiviert. Wenn 'data_cleansing_enabled' auf „false“ gesetzt ist, ist die Datenverarbeitung für die Tabelle deaktiviert. Im Folgenden finden Sie eine Liste der Datenverarbeitungseigenschaften auf Tabellenebene, die von dieser Eigenschaft gesteuert werden:

- column_count_mismatch_handling
- invalid_char_handling
- numeric_overflow_handling

- replacement_char
- surplus_char_handling

Beispiele finden Sie unter Beispiele für die Datenverarbeitung.

'invalid_char_handling'='value'

Gibt die Aktion an, die ausgeführt werden soll, wenn Abfrageergebnisse ungültige UTF-8-Zeichenwerte enthalten. Sie können die folgenden Aktionen festlegen:

DISABLED (DEAKTIVIERT)

Führt keine ungültige Zeichenbehandlung durch.

FEHLER

Bricht Abfragen ab, die Daten zurückgeben, die ungültige UTF-8-Werte enthalten.

SET_TO_NULL

Ersetzt ungültige UTF-8-Werte durch null.

DROP_ROW

Ersetzt jeden Wert in der Zeile durch null.

REPLACE

Ersetzt das ungültige Zeichen durch das Ersetzungszeichen, das Sie mit `replacement_char` angeben.

'replacement_char'='character’

Gibt das Ersetzungszeichen an, das beim Festlegen von `invalid_char_handling` auf `REPLACE` verwendet werden soll.

'numeric_overflow_handling'='value’

Gibt die Aktion an, die ausgeführt werden soll, wenn ORC-Daten eine Ganzzahl (z. B. BIGINT oder int64) enthalten, die größer als die Spaltendefinition ist (z. B. SMALLINT oder int16). Sie können die folgenden Aktionen festlegen:

DISABLED (DEAKTIVIERT)

Die ungültige Zeichenbehandlung ist deaktiviert.

FEHLER

Brechen Sie die Abfrage ab, wenn die Daten ungültige Zeichen enthalten.

SET_TO_NULL

Setzen Sie ungültige Zeichen auf null.

DROP_ROW

Setzen Sie jeden Wert in der Zeile auf null.

'surplus_bytes_handling'='Wert'

Gibt an, wie geladene Daten behandelt werden, die die Länge des Datentyps überschreiten, der für Spalten mit VARBYTE-Daten definiert ist. Standardmäßig legt Redshift Spectrum den Wert für Daten auf null fest, die die Breite der Spalte überschreiten.

Sie können die folgenden Aktionen angeben, die ausgeführt werden sollen, wenn die Abfrage Daten zurückgibt, die die Länge des Datentyps überschreiten:

SET_TO_NULL

Ersetzt Daten, die die Spaltenbreite überschreiten, durch null.

DISABLED (DEAKTIVIERT)

Führt keine Behandlung für überschüssige Bytes durch.

FEHLER

Bricht Abfragen ab, die Daten zurückgeben, die die Spaltenbreite überschreiten.

DROP_ROW

Entfernt alle Zeilen, deren Daten die Spaltenbreite überschreiten.

TRUNCATE

Entfernt die Zeichen, die die maximale Anzahl der Zeichen überschreiten, die für die Spalte definiert sind.

'surplus_char_handling'='value'

Gibt an, wie geladene Daten behandelt werden, die die Länge des Datentyps überschreiten, der für Spalten definiert ist, die VARCHAR-, CHAR- oder Zeichenfolgen-Daten enthalten. Standardmäßig legt Redshift Spectrum den Wert für Daten auf null fest, die die Breite der Spalte überschreiten.

Sie können die folgenden Aktionen angeben, die ausgeführt werden sollen, wenn die Abfrage Daten zurückgibt, die die Spaltenbreite überschreiten:

SET_TO_NULL

Ersetzt Daten, die die Spaltenbreite überschreiten, durch null.

DISABLED (DEAKTIVIERT)

Führt keine überzählige Zeichenbehandlung durch.

FEHLER

Bricht Abfragen ab, die Daten zurückgeben, die die Spaltenbreite überschreiten.

DROP_ROW

Ersetzt jeden Wert in der Zeile durch null.

TRUNCATE

Entfernt die Zeichen, die die maximale Anzahl der Zeichen überschreiten, die für die Spalte definiert sind.

'column_count_mismatch_handling'='value’

Identifiziert, ob die Datei weniger oder mehr Werte für eine Zeile enthält als die in der externen Tabellendefinition angegebene Anzahl von Spalten. Diese Eigenschaft ist nur für ein unkomprimiertes Textdateiformat verfügbar. Sie können die folgenden Aktionen festlegen:

DISABLED (DEAKTIVIERT)

Die Eigenschaft zum Umgang mit nicht übereinstimmenden Spalten ist deaktiviert.

FEHLER

Die Abfrage schlägt fehl, wenn eine Nichtübereinstimmung bei der Spaltenanzahl festgestellt wird.

SET_TO_NULL

Fehlende Werte werden mit NULL aufgefüllt und die zusätzlichen Werte in jeder Zeile ignoriert.

DROP_ROW

Alle Zeilen, die einen Fehler bei der Spaltenanzahl enthalten, werden vom Scan ausgeschlossen.

'numRows'='row_count'

Eine Eigenschaft, die den Wert numRows für die Tabellendefinition festlegt. Um die Statistiken einer externen Tabelle explizit zu aktualisieren, legen Sie mit der Eigenschaft numRows

die Größe der Tabelle fest. Amazon Redshift analysiert keine externen Tabellen, um die Tabellenstatistiken zu generieren, die der Abfrageoptimierer verwendet, um einen Abfrageplan zu erstellen. Wenn für eine externe Tabelle keine Tabellenstatistiken festgelegt sind, generiert Amazon Redshift einen Abfrageausführungsplan basierend auf der Annahme, dass externe Tabellen die größeren Tabellen und lokale Tabellen die kleineren Tabellen sind.

'skip.header.line.count'='line_count'

Eine Eigenschaft, die die Anzahl der Reihen festgelegt, die am Anfang jeder Quelldatei übersprungen wird.

'serialization.null.format'=' '

Eine Eigenschaft, die Spectrum festlegt, muss einen NULL-Wert zurückgeben, wenn eine exakte Übereinstimmung mit dem in einem Feld angegebenen Text besteht.

'orc.schema.resolution'='mapping_type'

Eine Eigenschaft, die den Spalten-Mapping-Typ für Tabellen einrichtet, die das ORC-Datenformat verwenden. Für andere Datenformate wird diese Eigenschaft ignoriert.

Gültige Werte für den Spalten-Mapping-Typ sind folgende:

- Name
- position

Wenn die Eigenschaft orc.schema.resolution weggelassen wird, werden die Spalten standardmäßig nach Name zugewiesen. Wenn orc.schema.resolution auf einen anderen Wert als 'name' oder 'position' gesetzt wird, werden die Spalten nach Position zugewiesen. Weitere Informationen zum Spalten-Mapping finden Sie unter Mapping externer Tabellenspalten zu ORC-Spalten.

> **Note**
>
> Der COPY-Befehl weist ORC-Datendateien nur nach Position zu. Die Tabelleneigenschaft orc.schema.resolution hat keine Auswirkungen auf das Verhalten des COPY-Befehls.

'write.parallel'='on / off'

Eine Eigenschaft, die festlegt, ob CREATE EXTERNAL TABLE AS Daten parallel schreiben soll. Standardmäßig schreibt CREATE EXTERNAL TABLE AS die Daten parallel in mehrere

Dateien, je nach der Anzahl der Slices in dem Cluster. Die Standardoption ist eingeschaltet. Wenn 'write.parallel' auf „off“ gesetzt ist, schreibt CREATE EXTERNAL TABLE AS eine oder mehrere Datendateien seriell in Amazon S3. Diese Tabelleneigenschaft gilt auch für alle nachfolgenden INSERT-Anweisungen in derselben externen Tabelle.

‘write.maxfilesize.mb’=‘size’

Eine Eigenschaft, mit der die maximale Größe (in MB) jeder Datei festgelegt wird, die von CREATE EXTERNAL TABLE AS in Amazon S3 geschrieben wurde. Die Größe muss eine gültige Ganzzahl zwischen 5 und 6.200 sein. Die standardmäßige maximale Dateigröße beträgt 6.200 MB. Diese Tabelleneigenschaft gilt auch für alle nachfolgenden INSERT-Anweisungen in derselben externen Tabelle.

‘write.kms.key.id’=‘value’

Sie können einen AWS Key Management Service Schlüssel angeben, um serverseitige Verschlüsselung (SSE) für Amazon S3 S3-Objekte zu aktivieren, wobei der Wert einer der folgenden Werte ist:

- `autoum` den im Amazon S3 S3-Bucket gespeicherten AWS KMS Standardschlüssel zu verwenden.
- kms-key, den Sie zur Verschlüsselung von Daten angeben.

select-statement

Eine Anweisung, die mindestens eine Zeile in die externe Tabelle einfügt, indem eine beliebige Abfrage definiert wird. Alle von der Abfrage erzeugten Zeilen werden auf der Grundlage der Tabellendefinition entweder in Text- oder in Parquet-Format in Amazon S3 geschrieben.

Beispiele

Eine Sammlung von Beispielen ist unter Beispiele verfügbar.

Nutzungshinweise

Dieses Thema enthält Nutzungshinweise für CREATE EXTERNAL TABLE. Sie können die Details für Amazon-Redshift-Spectrum-Tabellen nicht mit den gleichen Ressourcen anzeigen, die Sie für Amazon-Redshift-Standardtabellen verwenden, wie PG_TABLE_DEF, STV_TBL_PERM, PG_CLASS oder information_schema. Wenn Ihr Business Intelligence- oder Analyse-Tool externe Redshift Spectrum-Tabellen nicht erkennt, konfigurieren Sie Ihre Anwendung für die Ausführung von Abfragen für SVV_EXTERNAL_TABLES und SVV_EXTERNAL_COLUMNS.

CREATE EXTERNAL TABLE AS

In einigen Fällen können Sie den Befehl CREATE EXTERNAL TABLE AS für einen AWS Glue Datenkatalog, einen AWS Lake Formation externen Katalog oder einen Apache Hive-Metastore ausführen. In solchen Fällen verwenden Sie eine AWS Identity and Access Management (IAM-) Rolle, um das externe Schema zu erstellen. Diese IAM-Rolle muss sowohl Lese- als auch Schreibberechtigungen für Amazon S3 haben.

Wenn Sie einen Lake-Formation-Katalog verwenden, muss die IAM-Rolle über die Berechtigung verfügen, eine Tabelle im Katalog zu erstellen. In diesem Fall muss sie auch über die Data Lake-Speicherort-Berechtigung für den Amazon-S3-Zielpfad verfügen. Diese IAM-Rolle wird zum Besitzer der neuen AWS Lake Formation -Tabelle.

Um sicherzustellen, dass Dateinamen eindeutig sind, verwendet Amazon Redshift das folgende Format für den Namen jeder Datei, die standardmäßig in Amazon S3 hochgeladen wurde.

*<date>*_*<time>*_*<microseconds>*_*<query_id>*_*<slice-number>*_part_*<part-number>*.*<format>*.

Ein Beispiel ist `20200303_004509_810669_1007_0001_part_00.parquet`.

Berücksichtigen Sie beim Ausführen des Befehls CREATE EXTERNAL TABLE AS Folgendes:

- Der Amazon-S3-Speicherort muss leer sein.
- Amazon Redshift unterstützt nur PARQUET- und TEXTFILE-Formate, wenn die STORED AS-Klausel verwendet wird.
- Sie müssen keine Spaltendefinitionsliste definieren. Spaltennamen und Spaltendatentypen der neuen externen Tabelle werden direkt aus der SELECT-Abfrage abgeleitet.
- Sie müssen den Datentyp der Partitionsspalte in der PARTITIONED BY-Klausel nicht definieren. Wenn Sie einen Partitionsschlüssel angeben, muss der Name dieser Spalte im SELECT-Abfrageergebnis vorhanden sein. Wenn mehrere Partitionsspalten vorhanden sind, spielt ihre Reihenfolge in der SELECT-Abfrage keine Rolle. Amazon Redshift verwendet die in der PARTITIONED BY-Klausel definierte Reihenfolge, um die externe Tabelle zu erstellen.
- Amazon Redshift partitioniert Ausgabedateien automatisch basierend auf den Partitionsschlüsselwerten in Partitionsordnern. Amazon Redshift entfernt standardmäßig Partitionsspalten aus den Ausgabedateien.
- Die LINES TERMINATED BY 'delimiter'-Klausel wird nicht unterstützt.

- Die Klausel ROW FORMAT SERDE 'serde_name' wird nicht unterstützt.
- Die Verwendung von Manifestdateien wird nicht unterstützt. Daher können Sie die LOCATION-Klausel in Amazon S3 nicht für eine Manifestdatei definieren.
- Amazon Redshift aktualisiert die Tabelleneigenschaft 'numRows' am Ende des Befehls automatisch.
- Die Tabelleneigenschaft 'compression_type' akzeptiert nur 'none' oder 'snappy' als PARQUET-Dateiformat.
- Amazon Redshift lässt die LIMIT-Klausel in der äußeren SELECT-Abfrage nicht zu. Stattdessen können Sie eine verschachtelte LIMIT-Klausel verwenden.
- Sie können STL_UNLOAD_LOG verwenden, um die Dateien zu verfolgen, die von jedem CREATE EXTERNAL TABLE AS-Vorgang in Amazon S3 geschrieben werden.

Berechtigungen, externe Tabellen zu erstellen und abzufragen

Um externe Tabellen zu erstellen, stellen Sie sicher, dass Sie der Besitzer des externen Schemas oder ein Superuser sind. Um das Eigentum an einem externen Schema zu übertragen, verwenden Sie ALTER SCHEMA. Das folgende Beispiel ändert den Eigentümer des Schemas `spectrum_schema` in `newowner`.

```
alter schema spectrum_schema owner to newowner;
```

Um eine Redshift Spectrum-Abfrage auszuführen, benötigen Sie die folgenden Berechtigungen:

- Nutzungsberechtigung für das Schema
- Berechtigung, temporäre Tabellen in der aktuellen Datenbank zu erstellen

Das folgende Beispiel erteilt der Benutzergruppe `spectrum_schema` Nutzungsberechtigungen für das Schema `spectrumusers`.

```
grant usage on schema spectrum_schema to group spectrumusers;
```

Das folgende Beispiel erteilt der Benutzergruppe `spectrumdb` temporäre Berechtigungen für die Datenbank `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Pseudospalten

Standardmäßig erstellt Amazon Redshift externe Tabellen mit den Pseudospalten $path und $size. Wählen Sie diese Spalten, um den Pfad zu den Datendateien auf dem Amazon S3 und die Größe der Datendateien für jede Zeile anzuzeigen, die von einer Abfrage zurückgegeben wird. Die Spaltennamen $path und $size müssen mit doppelten Anführungszeichen abgetrennt sein. Eine SELECT *-Klausel gibt die Pseudospalten nicht zurück. Sie müssen die Spaltennamen $path und $size explizit in Ihre Abfrage einfügen, wie im folgenden Beispiel gezeigt.

```
select "$path", "$size"
from spectrum.sales_part
where saledate = '2008-12-01';
```

Sie können die Erstellung von Pseudospalten für eine Sitzung deaktivieren, indem Sie den spectrum_enable_pseudo_columns-Konfigurationsparameter auf false setzen.

> **Important**
>
> Bei Auswahl von $size oder $path fallen Gebühren an, weil Redshift Spectrum die Datendateien in Amazon S3 scannt, um die Größe des Ergebnissatzes zu bestimmen. Weitere Informationen finden Sie unter Amazon-Redshift-Preise.

Festlegen von Optionen zur Datenverarbeitung

Sie können Tabellenparameter festlegen, um die Eingabebehandlung für Daten anzugeben, die in externen Tabellen abgefragt werden, einschließlich:

- Überschüssige Zeichen in Spalten, die VARCHAR-, CHAR- und Zeichenfolgen-Daten enthalten. Weitere Informationen finden Sie unter der externen Tabelleneigenschaft `surplus_char_handling`.
- Ungültige Zeichen in Spalten, die VARCHAR-, CHAR- und Zeichenfolgen-Daten enthalten. Weitere Informationen finden Sie unter der externen Tabelleneigenschaft `invalid_char_handling`.
- Ersatzzeichen, das verwendet wird, wenn Sie REPLACE für die externe Tabelleneigenschaft `invalid_char_handling` angeben.
- Wirft die Überlaufbehandlung in Spalten mit Ganzzahl- und Dezimaldaten um. Weitere Informationen finden Sie unter der externen Tabelleneigenschaft `numeric_overflow_handling`.

- Surplus_bytes_handling, um die Eingabebehandlung für überschüssige Bytes in Spalten anzugeben, die VARBYTE-Daten enthalten. Weitere Informationen finden Sie unter der externen Tabelleneigenschaft `surplus_bytes_handling`.

Beispiele

Im folgenden Beispiel wird eine Tabelle namens SALES im externen Amazon-Redshift-Schema namens erstellt `spectrum`. Die Daten befinden sich in Textdateien, die Tabulatoren als Trennzeichen verwenden. Die Klausel TABLE PROPERTIES legt die Eigenschaft numRows auf 170.000 Zeilen fest.

Abhängig von der Identität, die Sie zum Ausführen von CREATE EXTERNAL TABLE verwenden, müssen Sie möglicherweise IAM-Berechtigungen konfigurieren. Als bewährte Methode empfehlen wir, einer IAM-Rolle Berechtigungsrichtlinien anzufügen und sie dann nach Bedarf Benutzern und Gruppen zuzuweisen. Weitere Informationen finden Sie unter Identity and Access Management in Amazon Redshift.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
saledate date,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='170000');
```

Im folgenden Beispiel wird eine Tabelle erstellt, die verwendet, JsonSerDe um auf Daten im JSON-Format zu verweisen.

```
create external table spectrum.cloudtrail_json (
event_version int,
event_id bigint,
```

```
event_time timestamp,
event_type varchar(10),
awsregion varchar(20),
event_name varchar(max),
event_source varchar(max),
requesttime timestamp,
useragent varchar(max),
recipientaccountid bigint)
row format serde 'org.openx.data.jsonserde.JsonSerDe'
with serdeproperties (
'dots.in.keys' = 'true',
'mapping.requesttime' = 'requesttimestamp'
) location 's3://mybucket/json/cloudtrail';
```

Im folgenden CREATE EXTERNAL TABLE AS-Beispiel wird eine nicht partitionierte externe Tabelle erstellt.- Dann wird das Ergebnis der SELECT-Abfrage als Apache Parquet an die Amazon-S3-Zielposition geschrieben.

```
CREATE EXTERNAL TABLE spectrum.lineitem
STORED AS parquet
LOCATION 'S3://mybucket/cetas/lineitem/'
AS SELECT * FROM local_lineitem;
```

Im folgenden Beispiel wird eine partitionierte externe Tabelle erstellt und die Partitionsspalten werden in der SELECT-Abfrage eingeschlossen.

```
CREATE EXTERNAL TABLE spectrum.partitioned_lineitem
PARTITIONED BY (l_shipdate, l_shipmode)
STORED AS parquet
LOCATION 'S3://mybucket/cetas/partitioned_lineitem/'
AS SELECT l_orderkey, l_shipmode, l_shipdate, l_partkey FROM local_table;
```

Eine Liste der vorhandenen Datenbanken im externen Datenkatalog erhalten Sie durch Abfragen der SVV_EXTERNAL_DATABASES-Systemansicht.

```
select eskind,databasename,esoptions from svv_external_databases order by databasename;
```

```
eskind | databasename | esoptions
-------+--------------
+-------------------------------------------------------------------------------
```

```
     1 | default      | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
     1 | sampledb     | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
     1 | spectrumdb   | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

Um Details zu externen Tabellen anzuzeigen, führen Sie eine Abfrage für die Systemansichten SVV_EXTERNAL_TABLES und SVV_EXTERNAL_COLUMNS aus.

Im folgenden Beispiel wird eine Abfrage für die Ansicht SVV_EXTERNAL_TABLES ausgeführt.

```
select schemaname, tablename, location from svv_external_tables;
```

```
schemaname | tablename            | location
-----------+----------------------
+-----------------------------------------------------
spectrum   | sales                | s3://redshift-downloads/tickit/spectrum/sales
spectrum   | sales_part           | s3://redshift-downloads/tickit/spectrum/
sales_partition
```

Im folgenden Beispiel wird eine Abfrage für die Ansicht SVV_EXTERNAL_COLUMNS ausgeführt.

```
select * from svv_external_columns where schemaname like 'spectrum%' and tablename
 ='sales';
```

```
schemaname | tablename | columnname | external_type | columnnum | part_key
-----------+-----------+------------+---------------+-----------+---------
spectrum   | sales     | salesid    | int           |         1 |        0
spectrum   | sales     | listid     | int           |         2 |        0
spectrum   | sales     | sellerid   | int           |         3 |        0
spectrum   | sales     | buyerid    | int           |         4 |        0
spectrum   | sales     | eventid    | int           |         5 |        0
spectrum   | sales     | saledate   | date          |         6 |        0
spectrum   | sales     | qtysold    | smallint      |         7 |        0
spectrum   | sales     | pricepaid  | decimal(8,2)  |         8 |        0
spectrum   | sales     | commission | decimal(8,2)  |         9 |        0
spectrum   | sales     | saletime   | timestamp     |        10 |        0
```

Verwenden Sie die folgende Abfrage zum Anzeigen von Tabellenpartitionen.

```
select schemaname, tablename, values, location
from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values         | location
-----------+------------+----------------
+-------------------------------------------------------------------------
spectrum   | sales_part | ["2008-01-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-02-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-02
spectrum   | sales_part | ["2008-03-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
spectrum   | sales_part | ["2008-05-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-05
spectrum   | sales_part | ["2008-06-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-06
spectrum   | sales_part | ["2008-07-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-07
spectrum   | sales_part | ["2008-08-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-08
spectrum   | sales_part | ["2008-09-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-09
spectrum   | sales_part | ["2008-10-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-10
spectrum   | sales_part | ["2008-11-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-11
spectrum   | sales_part | ["2008-12-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-12
```

Das folgende Beispiel gibt die Gesamtgröße der entsprechenden Datendateien für eine externe Tabelle zurück.

```
select distinct "$path", "$size"
   from spectrum.sales_part;

 $path                                                                      | $size
----------------------------------------------------------------------------+-------
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/ |  1616
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
```

```
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
```

Beispiele für Partitionierungen

Führen Sie zur Erstellung einer nach Datum partitionierten Tabelle den folgenden Befehl aus.

```
create external table spectrum.sales_part(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (saledate date)
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales_partition/'
table properties ('numRows'='170000');
```

Führen Sie zum Hinzufügen der Partitionen die folgenden ALTER TABLE-Befehle aus.

```
alter table spectrum.sales_part
add if not exists partition (saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-02-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-04-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-04/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-05-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-05/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-06-01')
```

```
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-06/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-07-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-07/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-08-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-08/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-09-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-09/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-10-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-10/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-11-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-11/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-12-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-12/';
```

Führen Sie die folgende Abfrage aus, um Daten aus der partitionierten Tabelle auszuwählen.

```
select top 10 spectrum.sales_part.eventid, sum(spectrum.sales_part.pricepaid)
from spectrum.sales_part, event
where spectrum.sales_part.eventid = event.eventid
  and spectrum.sales_part.pricepaid > 30
  and saledate = '2008-12-01'
group by spectrum.sales_part.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    914 | 36173.00
   5478 | 27303.00
   5061 | 26383.00
   4406 | 26252.00
   5324 | 24015.00
   1829 | 23911.00
   3601 | 23616.00
   3665 | 23214.00
   6069 | 22869.00
   5638 | 22551.00
```

Um externe Tabellenpartitionen anzuzeigen, führen Sie eine Abfrage für die Systemansicht SVV_EXTERNAL_PARTITIONS aus.

```
select schemaname, tablename, values, location from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values         | location
-----------+------------+----------------
+-----------------------------------------------
spectrum   | sales_part | ["2008-01-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-02-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-02
spectrum   | sales_part | ["2008-03-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
spectrum   | sales_part | ["2008-05-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-05
spectrum   | sales_part | ["2008-06-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-06
spectrum   | sales_part | ["2008-07-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-07
spectrum   | sales_part | ["2008-08-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-08
spectrum   | sales_part | ["2008-09-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-09
spectrum   | sales_part | ["2008-10-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-10
spectrum   | sales_part | ["2008-11-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-11
spectrum   | sales_part | ["2008-12-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-12
```

Beispiele für Zeilenformate

Nachfolgend sehen Sie ein Beispiel der Angabe von ROW FORMAT SERDE-Parametern für Datendateien, die im AVRO-Format gespeichert sind.

```
create external table spectrum.sales(salesid int, listid int, sellerid int,
 buyerid int, eventid int, dateid int, qtysold int, pricepaid decimal(8,2), comment
 VARCHAR(255))
```

```
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'
WITH SERDEPROPERTIES ('avro.schema.literal'='{\"namespace\": \"dory.sample\",\"name\":
 \"dory_avro\",\"type\": \"record\", \"fields\": [{\"name\":\"salesid\", \"type\":\"int
\"},
{\"name\":\"listid\", \"type\":\"int\"},
{\"name\":\"sellerid\", \"type\":\"int\"},
{\"name\":\"buyerid\", \"type\":\"int\"},
{\"name\":\"eventid\",\"type\":\"int\"},
{\"name\":\"dateid\",\"type\":\"int\"},
{\"name\":\"qtysold\",\"type\":\"int\"},
{\"name\":\"pricepaid\", \"type\": {\"type\": \"bytes\", \"logicalType\": \"decimal\",
 \"precision\": 8, \"scale\": 2}}, {\"name\":\"comment\",\"type\":\"string\"}]}')
STORED AS AVRO
location 's3://mybucket/avro/sales' ;
```

Das Folgende zeigt ein Beispiel für die Angabe der ROW FORMAT SERDE-Parameter mithilfe RegEx von.

```
create external table spectrum.types(
cbigint bigint,
cbigint_null bigint,
cint int,
cint_null int)
row format serde 'org.apache.hadoop.hive.serde2.RegexSerDe'
with serdeproperties ('input.regex'='([^\\x01]+)\\x01([^\\x01]+)\\x01([^\\x01]+)\
\x01([^\\x01]+)')
stored as textfile
location 's3://mybucket/regex/types';
```

Nachfolgend sehen Sie ein Beispiel der Angabe von ROW FORMAT SERDE-Parametern mit Grok.

```
create external table spectrum.grok_log(
timestamp varchar(255),
pid varchar(255),
loglevel varchar(255),
progname varchar(255),
message varchar(255))
row format serde 'com.amazonaws.glue.serde.GrokSerDe'
with serdeproperties ('input.format'='[DFEWI], \\[%{TIMESTAMP_ISO8601:timestamp} #
%{POSINT:pid:int}\\] *(?<loglevel>:DEBUG|FATAL|ERROR|WARN|INFO) -- +%{DATA:progname}:
 %{GREEDYDATA:message}')
stored as textfile
```

```
location 's3://mybucket/grok/logs';
```

Die folgende Abbildung zeigt ein Beispiel für ein Amazon-S3-Server-Zugriffsprotokoll in einem S3 Bucket. Sie können Redshift Spectrum verwenden, um Amazon-S3-Zugriffsprotokolle abzufragen.

```
CREATE EXTERNAL TABLE spectrum.mybucket_s3_logs(
bucketowner varchar(255),
bucket varchar(255),
requestdatetime varchar(2000),
remoteip varchar(255),
requester varchar(255),
requested varchar(255),
operation varchar(255),
key varchar(255),
requesturi_operation varchar(255),
requesturi_key varchar(255),
requesturi_httpprotoversion varchar(255),
httpstatus varchar(255),
errorcode varchar(255),
bytessent bigint,
objectsize bigint,
totaltime varchar(255),
turnaroundtime varchar(255),
referrer varchar(255),
useragent varchar(255),
versionid varchar(255)
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
'input.regex' = '([^ ]*) ([^ ]*) \\[(.*?)\\] ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 \"([^ ]*)\\s*([^ ]*)\\s*([^ ]*)\" (- |[^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*) (\"[^\"]*\") ([^ ]*).*$')
LOCATION 's3://mybucket/s3logs’;
```

Nachfolgend sehen Sie ein Beispiel der Angabe von ROW FORMAT SERDE-Parametern für Daten im ION-Format.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'com.amazon.ionhiveserde.IonHiveSerDe'
STORED AS
INPUTFORMAT 'com.amazon.ionhiveserde.formats.IonInputFormat'
OUTPUTFORMAT 'com.amazon.ionhiveserde.formats.IonOutputFormat'
```

```
LOCATION 's3://s3-bucket/prefix'
```

Beispiele für die Datenverarbeitung

Die folgenden Beispiele haben Zugriff auf die Datei: spi_global_rankings.csv. Sie können die Datei `spi_global_rankings.csv` in einem Amazon-S3-Bucket hochladen, um diese Beispiele auszuprobieren.

Das folgende Beispiel erstellt ein externes Schema `schema_spectrum_uddh` unter Verwendung der externen Datenbank `spectrum_db_uddh`. Geben Sie für `aws-account-id` Ihre AWS Konto-ID und für Ihren Redshift Spectrum-Rollennamen `role-name` ein.

```
create external schema schema_spectrum_uddh
from data catalog
database 'spectrum_db_uddh'
iam_role 'arn:aws:iam::aws-account-id:role/role-name'
create external database if not exists;
```

Das folgende Beispiel erstellt die externe Tabelle `soccer_league` im externen Schema `schema_spectrum_uddh`.

```
CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league
(
  league_rank smallint,
  prev_rank   smallint,
  club_name   varchar(15),
  league_name varchar(20),
  league_off  decimal(6,2),
  league_def  decimal(6,2),
  league_spi  decimal(6,2),
  league_nspi integer
)
ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
    LINES TERMINATED BY '\n\l'
stored as textfile
LOCATION 's3://spectrum-uddh/league/'
table properties ('skip.header.line.count'='1');
```

Überprüfen Sie die Gesamtzahl der Zeilen in der `soccer_league`-Tabelle.

```
select count(*) from schema_spectrum_uddh.soccer_league;
```

Die Anzahl der Zeilen wird angezeigt.

```
count
645
```

Die folgende Abfrage zeigt die 10 besten Clubs an. Da für den Club `Barcelona` ein ungültiges Zeichen in der Zeichenfolge angegeben ist, wird für den Namen NULL angezeigt.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

```
league_rank club_name league_name   league_nspi
1  Manchester City Barclays Premier Lea  34595
2  Bayern Munich German Bundesliga  34151
3  Liverpool Barclays Premier Lea  33223
4  Chelsea  Barclays Premier Lea  32808
5  Ajax  Dutch Eredivisie  32790
6  Atletico  Madrid Spanish Primera Divi 31517
7  Real Madrid Spanish Primera Divi  31469
8  NULL         Spanish Primera Divi            31321
9  RB Leipzig German Bundesliga  31014
10  Paris Saint-Ger French Ligue 1   30929
```

Im folgenden Beispiel wird die `soccer_league`-Tabelle geändert, um die externen Tabelleneigenschaften `invalid_char_handling`, `replacement_char` und `data_cleansing_enabled` anzugeben und ein Fragezeichen (?) als Ersatz für unerwartete Zeichen einzufügen.

```
alter  table schema_spectrum_uddh.soccer_league
set table properties
 ('invalid_char_handling'='REPLACE','replacement_char'='?','data_cleansing_enabled'='true');
```

Das folgende Beispiel fragt die Tabelle `soccer_league` nach Teams mit einem Rang von 1 bis 10 ab.

```
select league_rank,club_name,league_name,league_nspi
```

```
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

Da die Tabelleneigenschaften geändert wurden, werden die 10 Clubs an der Spitze angezeigt, wobei das Fragezeichen (?) in der achten Zeile als Ersatz für den Club `Barcelona` steht.

```
league_rank club_name league_name  league_nspi
1  Manchester City Barclays Premier Lea 34595
2  Bayern Munich German Bundesliga 34151
3  Liverpool Barclays Premier Lea 33223
4  Chelsea  Barclays Premier Lea 32808
5  Ajax  Dutch Eredivisie 32790
6  Atletico Madrid Spanish Primera Divi 31517
7  Real Madrid Spanish Primera Divi 31469
8  Barcel?na Spanish Primera Divi 31321
9  RB Leipzig German Bundesliga 31014
10  Paris Saint-Ger French Ligue 1  30929
```

Das folgende Beispiel ändert die `soccer_league`-Tabelle, um die externen Tabelleneigenschaften `invalid_char_handling` anzugeben und Zeilen mit unerwarteten Zeichen zu löschen.

```
alter table schema_spectrum_uddh.soccer_league
set table properties
 ('invalid_char_handling'='DROP_ROW','data_cleansing_enabled'='true');
```

Das folgende Beispiel fragt die Tabelle `soccer_league` nach Teams mit einem Rang von 1 bis 10 ab.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

Die Ergebnisse zeigen die besten Clubs an, ohne die achte Zeile für den Club `Barcelona`.

```
league_rank   club_name         league_name            league_nspi
1             Manchester City   Barclays Premier Lea   34595
2             Bayern Munich     German Bundesliga      34151
3             Liverpool         Barclays Premier Lea   33223
4             Chelsea           Barclays Premier Lea   32808
5             Ajax              Dutch Eredivisie       32790
```

```
6             Atletico Madrid   Spanish Primera Divi   31517
7             Real Madrid       Spanish Primera Divi   31469
9             RB Leipzig        German Bundesliga      31014
10            Paris Saint-Ger   French Ligue 1         30929
```

CREATE EXTERNAL VIEW (Vorschau)

Hierbei handelt es sich um die vorab veröffentlichte Dokumentation im Datenkatalog für Amazon Redshift, derzeit in der Vorschauversion. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter Beta- und Vorschauversionen in den AWS -Servicebedingungen.

Sie können einen Amazon-Redshift-Cluster in der Vorschau erstellen, um neue Funktionen von Amazon Redshift zu testen. Sie haben nicht die Möglichkeit, diese Funktionen in der Produktion zu verwenden oder Ihren Vorschau-Cluster in einen Produktionscluster oder einen Cluster auf einem anderen Pfad zu verschieben. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Erstellen eines Clusters in der Vorschau

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Provisioned clusters dashboard (Dashboard für bereitgestellte Cluster) und dann Clusters (Cluster) aus. Die aktuellen Cluster für Ihr Konto AWS-Region sind aufgeführt. Eine Teilmenge der Eigenschaften jedes Clusters wird in den Spalten der Liste angezeigt.
3. Auf der Seite mit der Clusterliste (Clusters) wird ein Banner angezeigt, das die Vorschau vorstellt. Wählen Sie die Schaltfläche Create preview cluster (Vorschau-Cluster erstellen) aus, um die Seite zum Erstellen von Clustern zu öffnen.
4. Geben Sie Eigenschaften für Ihren Cluster ein. Wählen Sie den Vorschaupfad (Preview track) aus, der die zu testenden Funktionen enthält. Wir empfehlen, einen Namen für den Cluster zu verwenden, der darauf hinweist, dass sich dieser auf einem Vorschaupfad befindet. Wählen Sie Optionen für Ihren Cluster, einschließlich Optionen mit der Bezeichnung -preview (Vorschau), für die zu testenden Funktionen. Allgemeine Informationen zum Erstellen von Clustern finden Sie unter Erstellen eines Clusters im Amazon-Redshift-Verwaltungshandbuch.

5. Wählen Sie Vorschau-Cluster erstellen aus, um einen Cluster in der Vorschau zu erstellen.

> **Note**
>
> Der `preview_2023`-Track ist der neueste verfügbare Vorschau-Track. Dieser Track unterstützt nur die Erstellung von Clustern mit RA3-Knotentypen. Der Knotentyp DC2 und alle älteren Knotentypen werden nicht unterstützt.

6. Wenn Ihr Vorschau-Cluster verfügbar ist, verwenden Sie Ihren SQL-Client, um Daten zu laden und abzufragen.

Das Vorschau-Feature für den Datenkatalog ist nur in den folgenden Regionen verfügbar.

- USA Ost (Ohio): (us-east-2)
- USA Ost (Nord-Virginia): (us-east-1)
- USA West (Nordkalifornien) (us-west-1)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Sie können außerdem eine Vorschau-Arbeitsgruppe erstellen, um Datenkatalog-Ansichten zu testen. Sie können diese Features nicht in der Produktion verwenden und Ihre Vorschau-Arbeitsgruppe auch nicht in eine andere Arbeitsgruppe verschieben. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter „Beta- und Vorschauversionen“ in den AWS -Servicebedingungen. Eine Anleitung zur Erstellung einer Vorschau-Arbeitsgruppe finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe.

Erstellt eine Ansicht im Datenkatalog. Eine Datenkatalog-Ansicht ist ein einzelnes Ansichtsschema, das mit SQL-Engines wie Amazon Athena und Amazon EMR funktioniert. Sie können die Ansicht von der Engine Ihrer Wahl aus abfragen. Weitere Informationen zu Datenkatalog-Ansichten finden Sie unter Erstellen von Datenkatalog-Ansichten (Vorschau).

Syntax

```
CREATE EXTERNAL VIEW schema_name.view_name [ IF NOT EXISTS ]
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
AS query_definition;
```

Parameter

schema_name.view_name

Das Schema, das an Ihre AWS Glue Datenbank angehängt ist, gefolgt vom Namen der Ansicht.

GESCHÜTZT

Gibt an, dass der Befehl CREATE EXTERNAL VIEW nur abgeschlossen werden soll, wenn die Abfrage in der query_definition erfolgreich abgeschlossen werden kann.

IF NOT EXISTS

Erstellt die Ansicht, falls sie noch nicht vorhanden ist.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

Die Notation des Schemas, das beim Erstellen der Ansicht verwendet werden soll. Sie können angeben AWS Glue Data Catalog, ob Sie eine von Ihnen erstellte Glue-Datenbank oder ein von Ihnen erstelltes externes Schema verwenden möchten. Weitere Informationen finden Sie unter CREATE DATABASE und CREATE EXTERNAL SCHEMA.

query_definition

Die Definition der SQL-Abfrage, die Amazon Redshift ausführt, um die Ansicht zu ändern.

Beispiele

Im folgenden Beispiel wird eine Datenkatalog-Ansicht mit dem Namen sample_schema.glue_data_catalog_view erstellt.

```
CREATE EXTERNAL PROTECTED VIEW sample_schema.glue_data_catalog_view IF NOT EXISTS
AS SELECT * FROM sample_database.remote_table "remote-table-name";
```

CREATE FUNCTION

Erstellt eine neue skalare benutzerdefinierte Funktion (User-Defined Function; UDF) mit einer SQL SELECT-Klausel oder einem Python-Programm.

Weitere Informationen und Beispiele finden Sie unter Erstellung benutzerdefinierter Funktionen.

Erforderliche Berechtigungen

Sie benötigen eine der folgenden Möglichkeiten, um CREATE OR REPLACE FUNCTION ausführen zu können:

- Für CREATE FUNCTION:
  - Superuser kann sowohl vertrauenswürdige als auch nicht vertrauenswürdige Sprachen verwenden, um Funktionen zu erstellen.
  - Benutzer mit der Berechtigung CREATE [OR REPLACE] FUNCTION können Funktionen mit vertrauenswürdigen Sprachen erstellen.
- Für REPLACE FUNCTION:
  - Superuser
  - Benutzer mit der Berechtigung CREATE [OR REPLACE] FUNCTION
  - Funktionsbesitzer

Syntax

```
CREATE [ OR REPLACE ] FUNCTION f_function_name
( { [py_arg_name  py_arg_data_type |
sql_arg_data_type } [ , ... ] ] )
RETURNS data_type
{ VOLATILE | STABLE | IMMUTABLE }
AS $$
  { python_program | SELECT_clause }
$$ LANGUAGE { plpythonu | sql }
```

Parameter

OR REPLACE

Gibt an, dass die vorhandene Funktion ersetzt wird, wenn eine Funktion mit demselben Namen und denselben Eingabeargument-Datentypen bzw. derselben Signatur vorhanden ist. Sie können eine Funktion nur durch eine neue Funktion ersetzen, wenn diese einen identischen Satz von Datentypen definiert. Sie müssen Superuser sein, um eine Funktion zu ersetzen.

Wenn Sie eine Funktion definieren, die den gleichen Namen wie eine vorhandene Funktion, aber eine andere Signatur besitzt, erstellen Sie eine neue Funktion. Der Funktionsname wird also überladen. Weitere Informationen finden Sie unter Überladen von Funktionsnamen.

f_function_name

Der Name der Funktion. Wenn Sie einen Schemanamen angeben (wie `myschema.myfunction`), wird die Funktion mithilfe des angegebenen Schemas erstellt. Andernfalls wird die Funktion im aktuellen Schema erstellt. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

Es wird empfohlen, dass Sie alle UDFs mit dem Präfix benennen `f_`. Amazon Redshift reserviert das Präfix `f_` ausschließlich für UDF-Namen. Wenn Sie Ihren UDF-Namen `f_` voranstellen, stellen Sie daher sicher, dass Ihre UDF-Namen nicht im Widerspruch zu den Namen vorhandener oder zukünftiger, in Amazon Redshift integrierter SQL-Funktionen stehen. Weitere Informationen finden Sie unter Benennung von UDFs.

Sie können mehr als eine Funktion mit demselben Funktionsnamen definieren, wenn sich die Datentypen für die Eingabeargumente unterscheiden. Der Funktionsname wird also überladen. Weitere Informationen finden Sie unter Überladen von Funktionsnamen.

py_arg_name py_arg_data_type | sql_arg_data type

Für eine Python-UDF: eine Liste eingegebener Argumentnamen und Datentypen. Für eine SQL-UDF: eine Liste der Datentypen ohne Argumentnamen. Verweisen Sie in einer Python-UDF anhand der Argumentnamen auf Argumente. Verweisen Sie in einer SQL-UDF mit $1, $2 usw. auf Argumente, basierend auf der Reihenfolge der Argumente in der Argumentliste.

Die Ein- und Rückgabedaten für eine SQL-UDF können jeden standardmäßigen Amazon Redshift-Datentyp verwenden. Für eine Python-UDF kommen als Eingabe- und Rückgabedatentypen SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE oder TIMESTAMP infrage. Darüber hinaus unterstützen Python benutzerdefinierte Funktionen (UDFs) den Datentyp ANYELEMENT. Dieser wird basierend auf dem Datentyp des entsprechenden Arguments, das zur Laufzeit bereitgestellt wird, automatisch in einen Standarddatentyp umgewandelt. Wenn mehrere Argumente ANYELEMENT verwenden, werden sie alle zur Laufzeit zum gleichen Datentyp aufgelöst, basierend auf dem ersten ANYELEMENT-Argument in der Liste. Weitere Informationen erhalten Sie unter Python-UDF-Datentypen und Datentypen.

Sie können maximal 32 Argumente angeben.

RETURNS data_type

Der Datentyp des Werts, der von der Funktion zurückgegeben wird. Der RETURNS-Datentyp kann ein beliebiger Standard-Amazon-Redshift-Datentyp sein. Zusätzlich können Python-

UDFs den Datentyp ANYELEMENT verwenden. Dieser wird basierend auf dem Argument, das zur Laufzeit bereitgestellt wird, automatisch in einen Standarddatentyp umgewandelt. Wenn Sie ANYELEMENT für den Rückgabedatentyp angeben, muss mindestens ein Argument ANYELEMENT verwenden. Der tatsächliche Rückgabedatentyp entspricht dem Datentyp, der für das ANYELEMENT-Argument angegeben wird, wenn die Funktion aufgerufen wird. Weitere Informationen finden Sie unter Python-UDF-Datentypen.

VOLATILE | STABLE | IMMUTABLE

Informiert den Abfragenoptimierer über die Volatilität der Funktion.

Sie erhalten die beste Optimierung, wenn Sie Ihre Funktion mit der strengsten Volatilitätskategorie bezeichnen, die für sie zulässig ist. Wenn die Kategorie jedoch zu eng gefasst ist, besteht das Risiko, dass der Optimierer fälschlicherweise einige Aufrufe überspringt. Dies führt zu einem falschen Ergebnissatz. Beginnend mit der Volatilitätskategorie mit der geringsten Strenge sind dies die Volatilitätskategorien:

- VOLATILE
- STABLE
- IMMUTABLE

VOLATILE

Bei gleichen Argumenten kann die Funktion unterschiedliche Ergebnisse für aufeinanderfolgende Aufrufe zurückgeben, auch für die Zeilen in einer einzelnen Anweisung. Der Abfragenoptimierer kann keine Annahmen über das Verhalten einer volatilen Funktion machen. Daher muss eine Abfrage, die eine volatile Funktion verwendet, die Funktion für jede Eingabezeile erneut evaluieren.

STABLE

Bei gleichen Argumenten gibt die Funktion garantiert die gleichen Ergebnisse für alle Zeilen zurück, die innerhalb einer einzelnen Anweisung verarbeitet werden. Die Funktion kann unterschiedliche Ergebnisse zurückgeben, wenn sie in unterschiedlichen Anweisungen aufgerufen wird. Diese Kategorie ermöglicht dem Optimierer, mehrere Aufrufe der Funktion innerhalb einer einzelnen Anweisung zu einem einzigen Aufruf für die Anweisung zu optimieren.

IMMUTABLE

Bei gleichen Argumenten gibt die Funktion stets das gleiche Ergebnis zurück. Wenn eine Abfrage eine IMMUTABLE-Funktion mit konstanten Argumenten aufruft, evaluiert der Optimierer die Funktion im Voraus.

AS $$-Anweisung $$

Ein Konstrukt, das die Ausführung der Anweisung einschließt. Die Literalschlüsselwörter AS $$ und $$ sind erforderlich.

Amazon Redshift erfordert, dass Sie die Anweisung in Ihrer Funktion unter Verwendung eines Formats umschließen, das Dollaranführung genannt wird. Alles innerhalb der Umschließung wird exakt wie angegeben übergeben. Sie müssen für Sonderzeichen keine Escape-Zeichen verwenden, da der Inhalt der Zeichenfolge wie exakt wie angegeben geschrieben wird.

Bei der Dollaranführung verwenden Sie ein Dollarzeichenpaar ($$), um den Anfang und das Ende der auszuführenden Anweisung zu kennzeichnen, wie im folgenden Beispiel gezeigt.

```
$$ my statement $$
```

Optional können Sie zwischen den beiden Dollarzeichen eine Zeichenfolge angeben, um die Identifizierung der Anweisung zu unterstützen. Die von Ihnen im umschließenden Dollarzeichenpaar am Anfang und am Ende verwendete Zeichenfolge muss identisch sein. Diese Zeichenfolge unterscheidet zwischen Groß- und Kleinschreibung und unterliegt den gleichen Einschränkungen wie Bezeichner, die nicht in einer Anführung stehen, darf jedoch keine Dollarzeichen enthalten. Im folgenden Beispiel wird die Zeichenfolge verwendet test.

```
$test$ my statement $test$
```

Weitere Informationen zur Dollaranführung finden Sie unter "Zeichenfolgenkonstanten in Dollaranführung" im Abschnitt Lexikalische Struktur der PostgreSQL-Dokumentation.

python_program

Ein gültiges ausführbares Python-Programm, das einen Wert zurückgibt. Die Anweisung, die Sie mit der Funktion übergeben, muss den Einrückungsanforderungen entsprechen, die in den Vorgaben für Python-Code auf der Python-Website angegeben sind. Weitere Informationen finden Sie unter Python-Sprachunterstützung für UDFs.

SQL_clause

Eine SQL SELECT-Klausel.

Die SELECT-Klausel darf keinen der folgenden Klausel-Typen enthalten:

- FROM
- INTO
- WHERE
- GROUP BY
- ORDER BY
- LIMIT

LANGUAGE { plpythonu | sql }

Legen Sie für Python fest `plpythonu`. Legen Sie für SQL fest `sql`. Sie müssen eine Berechtigung für die Nutzung in der Sprache für SQL oder plpythonu besitzen. Weitere Informationen finden Sie unter UDF-Sicherheit und Rechte.

Nutzungshinweise

Verschachtelte Funktionen

Sie können eine weitere SQL-UDF innerhalb einer SQL-UDF aufrufen. Die verschachtelte Funktion muss existieren, wenn Sie den Befehl CREATE FUNCTION ausführen. Amazon Redshift verfolgt keine Abhängigkeiten für UDFs, wenn Sie also die verschachtelte Funktion löschen, gibt Amazon Redshift keinen Fehler zurück. Die UDF schlägt jedoch fehl, wenn die verschachtelte Funktion nicht vorhanden ist. Die folgende Funktion ruft beispielsweise die `f_sql_greater` -Funktion in der SELECT-Klausel auf.

```
create function f_sql_commission (float, float )
  returns float
stable
as $$
  select f_sql_greater ($1, $2)
$$ language sql;
```

UDF-Sicherheit und Rechte

Zum Erstellen einer UDF müssen Sie eine Berechtigung für die Nutzung in der Sprache für SQL oder plpythonu (Python) besitzen. USAGE ON LANGUAGE SQL wird standardmäßig PUBLIC gewährt. USAGE ON LANGUAGE PLPYTHONU müssen Sie jedoch spezifischen Benutzern oder Gruppen explizit gewähren.

Um die Nutzung für SQL zu widerrufen, widerrufen Sie zuerst die Nutzung von PUBLIC. Erlauben Sie dann die Nutzung in SQL nur spezifischen Benutzern oder Gruppen, die auch die Erlaubnis zum Erstellen von SQL-UDFs besitzen. Im folgenden Beispiel wird die Nutzung in SQL von PUBLIC widerrufen. Anschließend wird die Nutzung der Benutzergruppe erlaubt `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Um eine UDF auszuführen, benötigen Sie für die jeweilige Funktion eine Ausführungsberechtigung. Standardmäßig wird PUBLIC die Ausführungsberechtigung für neue UDFs gewährt. Um die Nutzung einzuschränken, widerrufen Sie die Ausführungsberechtigung für die Funktion von PUBLIC. Gewähren Sie die Berechtigung anschließend den spezifischen Einzelpersonen oder Gruppen.

Im folgenden Beispiel wird die Ausführungsberechtigung der Funktion `f_py_greater` von PUBLIC widerrufen. Anschließend wird die Nutzung der Benutzergruppe `udf_devs` erlaubt.

```
revoke execute on function f_py_greater(a float, b float) from PUBLIC;
grant execute on function f_py_greater(a float, b float) to group udf_devs;
```

Superuser haben standardmäßig alle Berechtigungen.

Weitere Informationen erhalten Sie unter GRANT und REVOKE.

Beispiele

Beispiel für eine skalare Python-UDF

Im folgenden Beispiel wird eine Python-UDF erstellt, mit der zwei Ganzzahlen verglichen werden und die den größeren Wert zurückgibt.

```
create function f_py_greater (a float, b float)
  returns float
stable
as $$
  if a > b:
    return a
  return b
$$ language plpythonu;
```

Im folgenden Beispiel werden eine Abfrage für die Tabelle SALES ausgeführt und die neue `f_py_greater`-Funktion aufgerufen, um entweder COMMISSION oder 20 Prozent von PRICEPAID zurückzugeben, je nachdem, welcher Wert größer ist.

```
select f_py_greater (commission, pricepaid*0.20) from sales;
```

Beispiel für eine skalare SQL-UDF

Im folgenden Beispiel wird eine Funktion erstellt, die zwei Zahlen vergleicht und die größere zurückgibt.

```
create function f_sql_greater (float, float)
  returns float
stable
as $$
  select case when $1 > $2 then $1
    else $2
  end
$$ language sql;
```

In der folgenden Abfrage wird die neue Funktion `f_sql_greater` aufgerufen, um eine Abfrage über der Tabelle SALES auszuführen und entweder COMMISSION oder 20 Prozent von PRICEPAID zurückzugeben, je nachdem, welcher Wert größer ist.

```
select f_sql_greater (commission, pricepaid*0.20) from sales;
```

CREATE GROUP

Definiert eine neue Benutzergruppe. Nur Superuser können Gruppen erstellen.

Syntax

```
CREATE GROUP group_name
[ [ WITH ] [ USER username ] [, ...] ]
```

Parameter

group_name

Der Name einer neuen Benutzergruppe. Gruppennamen, die mit zwei Unterstrichen beginnen, sind für die interne Verwendung durch Amazon Redshift reserviert. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

WITH

Optionale Syntax für die Angabe zusätzlicher Parameter für CREATE GROUP.

USER

Fügt der Gruppe einen oder mehrere Benutzer hinzu.

username (Benutzername

Der Name des Benutzers, der der Gruppe hinzugefügt werden soll.

Beispiele

Im folgenden Beispiel wird eine Benutzergruppe namens ADMIN_GROUP mit zwei Benutzern namens ADMIN1 und ADMIN2 hinzugefügt.

```
create group admin_group with user admin1, admin2;
```

CREATE IDENTITY PROVIDER

Definiert einen neuen Identitätsanbieter. Nur Superuser können Identitätsanbieter erstellen.

Syntax

```
CREATE IDENTITY PROVIDER identity_provider_name TYPE type_name
NAMESPACE namespace_name
[PARAMETERS parameter_string]
[APPLICATION_ARN arn]
[IAM_ROLE iam_role]
```

Parameter

identity_provider_name

Name des neuen Identitätsanbieters. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

type_name

Der Identitätsanbieter, zu dem eine Verbindung hergestellt werden soll. Azure ist derzeit der einzige unterstützte Identitätsanbieter.

namespace_name

Der Namespace Dies ist ein eindeutiger Kurzbezeichner für das Identitätsanbieter-Verzeichnis.

parameter_string

Eine Zeichenfolge mit einem ordnungsgemäß formatierten JSON-Objekt, das die für den Identitätsanbieter erforderlichen Parameter und Werte enthält.

arn

Der Amazon-Ressourcenname (ARN) für eine von IAM Identity Center verwaltete Anwendung. Dieser Parameter ist nur anwendbar, wenn der Identitätsanbietertyp ist. AWSIDC

iam_role

Die IAM-Rolle, die Berechtigungen zum Herstellen der Verbindung zum IAM Identity Center bereitstellt. Dieser Parameter ist nur anwendbar, wenn der Identitätsanbietertyp ist. AWSIDC

Beispiele

Im folgenden Beispiel wird ein Identitätsanbieter namens oauth_standard mit dem Typ azure erstellt, um eine Kommunikation mit Microsoft Azure Active Directory (AD) herzustellen.

```
CREATE IDENTITY PROVIDER oauth_standard TYPE azure
NAMESPACE 'aad'
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

Sie können eine von IAM Identity Center verwaltete Anwendung mit einem vorhandenen bereitgestellten Cluster oder einer Amazon Redshift Serverless-Arbeitsgruppe verbinden. Auf diese Weise können Sie den Zugriff auf eine Redshift-Datenbank über IAM Identity Center verwalten. Führen Sie dazu einen SQL-Befehl wie das folgende Beispiel aus. Sie müssen ein Datenbankadministrator sein.

```
CREATE IDENTITY PROVIDER "redshift-idc-app" TYPE AWSIDC
NAMESPACE 'awsidc'
APPLICATION_ARN 'arn:aws:sso::123456789012:application/ssoins-12345f67fe123d4/apl-
a0b0a12dc123b1a4'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyRedshiftRole';
```

Der Anwendungs-ARN identifiziert in diesem Fall die verwaltete Anwendung, mit der eine Verbindung hergestellt werden soll. Sie können es finden, indem Sie es ausführen`SELECT * FROM SVV_IDENTITY_PROVIDERS;`.

Weitere Informationen zur Verwendung von CREATE IDENTITY PROVIDER, einschließlich zusätzlicher Beispiele, finden Sie unter Nativer Identitätsanbieter (IDP)-Verbund für Amazon Redshift. Weitere Informationen zum Einrichten einer Verbindung zu IAM Identity Center von Redshift aus finden Sie unter Redshift mit IAM Identity Center Connect, um Benutzern ein Single-Sign-On-Erlebnis zu bieten.

CREATE LIBRARY

Installiert eine Python-Bibliothek, die der Benutzer einschließen kann, wenn mit dem Befehl CREATE FUNCTION eine benutzerdefinierte Funktion (User-Defined Function, UDF) erstellt wird. Die Gesamtgröße von Bibliotheken, die von Benutzern installiert werden, darf 100 MB nicht überschreiten.

CREATE LIBRARY kann nicht innerhalb eines Transaktionsblocks ausgeführt werden (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Amazon Redshift unterstützt Python Version 2.7. Weitere Informationen erhalten Sie unter www.python.org.

Weitere Informationen finden Sie unter Importieren von benutzerdefinierten Python-Bibliotheksmodulen.

Erforderliche Berechtigungen

Für CREATE LIBRARY sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung CREATE LIBRARY oder mit der Berechtigung für die angegebene Sprache

Syntax

```
CREATE [ OR REPLACE ] LIBRARY library_name LANGUAGE plpythonu
FROM
{ 'https://file_url'
| 's3://bucketname/file_name'
authorization
  [ REGION [AS] 'aws_region']
  IAM_ROLE { default | ‘arn:aws:iam::<AWS-Konto-id>:role/<role-name>’ }
}
```

Parameter

OR REPLACE

Gibt an, dass die vorhandene Bibliothek ersetzt wird, wenn bereits eine Bibliothek mit dem gleichen Namen vorhanden ist. REPLACE führt sofort einen Commit aus. Wenn gleichzeitig eine UDF ausgeführt wird, die von der Bibliothek abhängig ist, schlägt die UDF möglicherweise fehl oder gibt unerwartete Ergebnisse zurück, auch wenn die UDF innerhalb einer Transaktion ausgeführt wird. Sie müssen der Besitzer oder ein Superuser sein, um eine Bibliothek zu ersetzen.

library_name

Das Name der Bibliothek, die installiert werden soll. Die erstellten Bibliotheken dürfen jedoch nicht genauso heißen wie die integrierten Python-Standardbibliotheksmodule oder die vorinstallierten Amazon-Redshift-Python-Module. Wenn eine vorhandene, benutzerinstallierte Bibliothek dasselbe Python-Paket wie die Bibliothek verwendet, die installiert werden soll, müssen Sie die vorhandene Bibliothek entfernen, bevor Sie die neue Bibliothek installieren. Weitere Informationen finden Sie unter Python-Sprachunterstützung für UDFs.

LANGUAGE plpythonu

Die Sprache, die verwendet werden soll. Python (plpythonu) ist die einzige unterstützte Sprache. Amazon Redshift unterstützt Python Version 2.7. Weitere Informationen erhalten Sie unter www.python.org.

FROM

Der Speicherort der Bibliotheksdatei. Sie können einen Amazon-S3-Bucket und -Objektnamen angeben oder eine URL, um die Datei von einer öffentlichen Website herunterzuladen. Die Bibliothek muss als `.zip`-Datei gepackt sein. Weitere Informationen finden Sie unter Erstellen und Installieren von Python-Modulen in der Python-Dokumentation.

https://file_url

Die URL für den Download der Datei von einer öffentlichen Website. Die URL kann bis zu drei Umleitungen enthalten. Im Folgenden wird ein Beispiel für eine Datei-URL gezeigt.

```
'https://www.example.com/pylib.zip'
```

s3://bucket_name/file_name

Der Pfad zu einem einzelnen Amazon-S3-Objekt, das die Bibliotheksdatei enthält. Im Folgenden wird ein Beispiel für einen Amazon-S3-Objektpfad gezeigt.

```
's3://mybucket/my-pylib.zip'
```

Wenn Sie einen Amazon-S3-Bucket angeben, müssen Sie auch die Anmeldeinformationen für einen AWS -Benutzer angeben, der zum Herunterladen der Datei berechtigt ist.

> **Important**
>
> Wenn sich der Amazon S3 S3-Bucket nicht in derselben AWS Region wie Ihr Amazon Redshift Redshift-Cluster befindet, müssen Sie die Option REGION verwenden, um die AWS Region anzugeben, in der sich die Daten befinden. Der Wert für aws_region muss mit einer AWS Region übereinstimmen, die in der Tabelle in der REGION Parameterbeschreibung für den COPY-Befehl aufgeführt ist.

Autorisierung

Eine Klausel, die die Methode angibt, die der Cluster für die Authentifizierung und Autorisierung verwendet, um auf den Amazon-S3-Bucket zuzugreifen, der die Bibliotheksdatei enthält. Ihr Cluster muss die Berechtigung für den Zugriff auf Amazon S3 mithilfe der Aktionen LIST und GET besitzen.

Die Syntax für die Autorisierung ist identisch mit der Syntax für die Autorisierung des Befehls COPY. Weitere Informationen finden Sie unter Autorisierungsparameter.

```
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id>:role/<role-name>'
```

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE LIBRARY-Befehl ausgeführt wird.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Wenn Sie IAM_ROLE angeben, können Sie ACCESS_KEY_ID und SECRET_ACCESS_KEY, SESSION_TOKEN oder CREDENTIALS nicht verwenden.

Wenn der Amazon-S3-Bucket serverseitige Verschlüsselung verwendet, können Sie optional den Verschlüsselungsschlüssel in der Zeichenfolge credentials-args angeben. Wenn Sie temporäre Sicherheitsanmeldeinformationen verwenden, geben Sie das temporäre Token in die credentials-args-Zeichenfolge ein.

Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen.

REGION [AS] aws_region

Die AWS Region, in der sich der Amazon S3 S3-Bucket befindet. REGION ist erforderlich, wenn sich der Amazon S3 S3-Bucket nicht in derselben AWS Region wie der Amazon Redshift Redshift-Cluster befindet. Der Wert für aws_region muss mit einer AWS Region übereinstimmen, die in der Tabelle in der REGION Parameterbeschreibung für den COPY-Befehl aufgeführt ist.

Standardmäßig geht CREATE LIBRARY davon aus, dass sich der Amazon S3 S3-Bucket in derselben AWS Region wie der Amazon Redshift Redshift-Cluster befindet.

Beispiele

In den folgenden beiden Beispielen wird das Python-Modul urlparse installiert, das in einer Datei namens `urlparse3-1.0.3.zip` gepackt ist.

Der folgende Befehl installiert eine UDF-Bibliothek namens `f_urlparse` aus einem Paket, das zu einem Amazon-S3-Bucket hochgeladen wurde, der sich in der Region „US East“ befindet.

```
create library f_urlparse
language plpythonu
```

```
from 's3://mybucket/urlparse3-1.0.3.zip'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
region as 'us-east-1';
```

Im folgenden Beispiel wird eine Bibliothek namens `f_urlparse` aus einer Bibliotheksdatei auf einer Website installiert.

```
create library f_urlparse
language plpythonu
from 'https://example.com/packages/urlparse3-1.0.3.zip';
```

ERSTELLEN EINER MASKIERUNGSRICHTLINIE

Erstellt eine neue Richtlinie für die dynamische Datenmaskierung, um Daten eines bestimmten Formats zu verschleiern. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Maskierungsrichtlinie erstellen.

Syntax

```
CREATE MASKING POLICY
   policy_name [IF NOT EXISTS]
   WITH (input_columns)
   USING (masking_expression);
```

Parameter

policy_name

Der Name der Maskierungsrichtlinie. Die Maskierungsrichtlinie darf nicht denselben Namen wie eine andere Maskierungsrichtlinie haben, die bereits in der Datenbank vorhanden sind.

input_columns

Ein Tupel von Spaltennamen im Format (Spalte1 Typ, Spalte2 Typ ...).

Spaltennamen werden als Eingabe für den Maskierungsausdruck verwendet. Die Spaltennamen müssen nicht den Namen der zu maskierenden Spalten entsprechen, die Eingabe- und Ausgabedatentypen müssen jedoch übereinstimmen.

masking_expression

Der SQL-Ausdruck, der zur Transformation der Zielspalten verwendet wird. Er kann mithilfe von Datenmanipulationsfunktionen wie z. B. Funktionen zur Zeichenkettenmanipulation oder in Verbindung mit benutzerdefinierten Funktionen geschrieben werden, die in SQL, Python oder mit AWS Lambda geschrieben wurden. Sie können ein Tupel von Spaltenausdrücken für Maskierungsrichtlinien mit mehreren Ausgaben hinzufügen. Wenn Sie eine Konstante als Maskierungsausdruck verwenden, müssen Sie sie explizit in einen dem Eingabetyp entsprechenden Typ umwandeln.

Sie benötigen USAGE-Berechtigung für alle benutzerdefinierten Funktionen, die Sie im Maskierungsausdruck verwenden.

CREATE MATERIALIZED VIEW

Erstellt eine materialisierte Ansicht, die auf einer oder mehreren Amazon-Redshift-Tabellen basiert. Sie können materialisierte Ansichten auch auf externe Tabellen basieren, die mit Spectrum oder einer Verbundabfrage erstellt wurden. Weitere Informationen zu Spectrum finden Sie unter Abfrage externer Daten mit Amazon Redshift Spectrum. Weitere Informationen zur Verbundabfrage finden Sie unter Abfragen von Daten mit Verbundabfragen in Amazon Redshift.

Syntax

```
CREATE MATERIALIZED VIEW mv_name
[ BACKUP { YES | NO } ]
[ table_attributes ]
[ AUTO REFRESH { YES | NO } ]
AS query
```

Parameter

BACKUP

Eine Klausel, die angibt, ob die materialisierte Ansicht in automatisierten und manuellen Cluster-Snapshots enthalten ist, die in Amazon S3 gespeichert werden.

Der Standardwert für den BACKUP beträgt YES.

Sie können BACKUP NO angeben, um Verarbeitungszeit bei der Erstellung von Snapshots und der Wiederherstellung aus Snapshots zu sparen und den Speicherbedarf in Amazon S3 zu reduzieren.

> **Note**
>
> Die Einstellung BACKUP NO hat keinen Einfluss auf die automatische Replikation von Daten auf andere Knoten innerhalb des Clusters, sodass bei einem Knotenfehler Tabellen mit der Angabe BACKUP NO wiederhergestellt werden.

table_attribute

Eine Klausel, die angibt, wie die Daten in der materialisierten Ansicht verteilt werden, einschließlich der folgenden:

- Der Verteilungsstil für die materialisierte Ansicht im Format DISTSTYLE { EVEN | ALL | KEY }. Wenn Sie diese Klausel weglassen, ist der Verteilungsstil EVEN. Weitere Informationen finden Sie unter Verteilungsstile.
- Der Verteilungsschlüssel für die materialisierte Ansicht im Format DISTKEY (*distkey_identifier*). Weitere Informationen finden Sie unter Bezeichnen von Verteilungsstilen.
- Der Sortierschlüssel für die materialisierte Ansicht im Format SORTKEY (*column_name* [, ...]). Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.

AS query

Eine gültige SELECT-Anweisung, die die materialisierte Ansicht und ihren Inhalt definiert. Die Ergebnismenge aus der Abfrage definiert die Spalten und Zeilen der materialisierten Ansicht. Hinweise zu Einschränkungen beim Erstellen materialisierter Ansichten finden Sie unter Einschränkungen.

Darüber hinaus bestimmen einige SQL-Sprachkonstrukte, die in der Abfrage verwendet werden, ob die materialisierte Ansicht inkrementell oder vollständig aktualisiert werden kann. Hinweise zur Refresh-Methode finden Sie unter REFRESH MATERIALIZED VIEW. Hinweise zu den Einschränkungen für die inkrementelle Aktualisierung finden Sie unter Einschränkungen für die inkrementelle Aktualisierung.

Wenn die Abfrage einen SQL-Befehl enthält, der keine inkrementelle Aktualisierung unterstützt, zeigt Amazon Redshift eine Meldung an, die angibt, dass die materialisierte Ansicht eine

vollständige Aktualisierung verwendet. Je nach SQL-Clientanwendung kann die Meldung angezeigt werden oder nicht. Überprüfen Sie die `state`-Spalte von STV_MV_INFO, um den Aktualisierungstyp anzuzeigen, der von einer materialisierten Ansicht verwendet wird.

AUTO REFRESH

Eine Klausel, die definiert, ob die materialisierte Ansicht automatisch mit den neuesten Änderungen aus ihren Basistabellen aktualisiert werden soll. Der Standardwert ist NO. Weitere Informationen finden Sie unter Aktualisieren einer materialisierten Ansicht.

Nutzungshinweise

Um eine materialisierte Ansicht zu erstellen, müssen Sie über die folgenden Berechtigungen verfügen:

- CREATE-Berechtigungen für ein Schema.
- SELECT-Berechtigung auf Tabellen- oder Spaltenebene für die Basistabellen zum Erstellen einer materialisierten Ansicht. Wenn Sie über Berechtigungen auf Spaltenebene für bestimmte Spalten verfügen, können Sie nur für diese Spalten eine materialisierte Ansicht erstellen.

Inkrementelle Aktualisierung für materialisierte Ansichten in einem Datashare

Amazon Redshift unterstützt automatische und inkrementelle Aktualisierungen für materialisierte Ansichten in einem Consumer-Datashare, wenn die Basistabellen gemeinsam genutzt werden. Inkrementelle Aktualisierung ist ein Vorgang, bei dem Amazon Redshift Änderungen in der Basistabelle oder den Basistabellen identifiziert, die nach der vorherigen Aktualisierung vorgenommen wurden, und nur die entsprechenden Datensätze in der materialisierten Ansicht aktualisiert. Dieser Vorgang läuft schneller als eine vollständige Aktualisierung und verbessert die Workload-Leistung. Sie müssen Ihre Materialized-View-Definition nicht ändern, um die Vorteile der inkrementellen Aktualisierung nutzen zu können.

Bei der Nutzung der inkrementellen Aktualisierung mit einer materialisierten Ansicht sind einige Einschränkungen zu beachten:

- Die materialisierte Ansicht darf nur auf eine Datenbank verweisen, entweder lokal oder remote.
- Die inkrementelle Aktualisierung ist nur für neue materialisierte Ansichten verfügbar. Daher müssen Sie vorhandene materialisierte Ansichten löschen und neu erstellen, damit eine inkrementelle Aktualisierung erfolgt.

Weitere Informationen zum Erstellen materialisierter Ansichten in einem Datashare finden Sie unter Arbeiten mit Ansichten beim Amazon Redshift Redshift-Datenaustausch, der mehrere Abfragebeispiele enthält.

DDL-Aktualisierungen an materialisierten Ansichten oder Basistabellen

Wenn Sie materialisierte Ansichten in Amazon Redshift verwenden, folgen Sie diesen Verwendungshinweisen für DDL-Aktualisierungen (Data Definition Language) von materialisierten Ansichten oder Basistabellen.

- Sie können Spalten zu einer Basistabelle hinzufügen, ohne dass sich dies auf materialisierte Ansichten auswirkt, die auf die Basistabelle verweisen.
- Andere Operationen können die materialisierte Ansicht in einem Status belassen, der überhaupt nicht aktualisiert werden kann. Beispiele sind Operationen wie das Umbenennen oder Löschen einer Spalte, das Ändern des Spaltentyps und das Ändern des Schemanamens. Solche materialisierten Ansichten können abgefragt, aber nicht aktualisiert werden. In diesem Fall müssen Sie einen Drop für die materialisierte Ansicht durchführen und diese neu anlegen.
- Im Allgemeinen können Sie die Definition einer materialisierten Ansicht (deren SQL-Anweisung) nicht ändern.
- Sie können eine materialisierte Ansicht nicht umbenennen.

Einschränkungen

Sie können keine materialisierte Ansicht definieren, die Folgendes referenziert oder einschließt:

- Standardansichten oder Systemtabellen und Ansichten.
- Temporäre Tabellen.
- Benutzerdefinierte Funktionen.
- Die ORDER BY-, LIMIT- oder OFFSET-Klausel.
- Late-Binding-Referenzen auf Basistabellen. Mit anderen Worten, alle Basistabellen oder zugehörigen Spalten, die in der definierenden SQL-Abfrage der materialisierten Ansicht referenziert werden, müssen existieren und gültig sein.
- Auf den Führungsknoten beschränkte Funktionen: CURRENT_SCHEMA, CURRENT_SCHEMAS, HAS_DATABASE_PRIVILEGE, HAS_SCHEMA_PRIVILEGE, HAS_TABLE_PRIVILEGE.
- Sie können die Option AUTO REFRESH YES nicht verwenden, wenn die Definition der materialisierten Ansicht veränderbare Funktionen oder externe Schemata enthält. Sie können

sie auch nicht verwenden, wenn Sie eine materialisierte Ansicht in einer anderen materialisierten Ansicht definieren.

- Sie müssen ANALYZE für materialisierte Ansichten nicht manuell ausführen. Dies geschieht derzeit nur über AUTO ANALYZE. Weitere Informationen finden Sie unter Analysieren von Tabellen.

Beispiele

Im folgenden Beispiel wird eine materialisierte Ansicht aus drei Basistabellen erstellt, die verbunden und aggregiert sind. Jede Zeile stellt eine Kategorie mit der Anzahl der verkauften Tickets dar. Wenn Sie die materialisierte Ansicht tickets_mv abfragen, greifen Sie direkt auf die vorberechneten Daten in der materialisierten Ansicht tickets_mv zu.

```
CREATE MATERIALIZED VIEW tickets_mv AS
    select   catgroup,
    sum(qtysold) as sold
    from     category c, event e, sales s
    where    c.catid = e.catid
    and      e.eventid = s.eventid
    group by catgroup;
```

Im folgenden Beispiel wird eine materialisierte Ansicht ähnlich dem vorherigen Beispiel erstellt und die Aggregationsfunktion MAX() verwendet.

```
CREATE MATERIALIZED VIEW tickets_mv_max AS
    select   catgroup,
    max(qtysold) as sold
    from     category c, event e, sales s
    where    c.catid = e.catid
    and      e.eventid = s.eventid
    group by catgroup;

SELECT name, state FROM STV_MV_INFO;
```

Im folgenden Beispiel wird eine UNION ALL-Klausel verwendet, um die `public_sales`-Tabelle aus Amazon-Redshift und die `spectrum.sales`-Tabelle aus Redshift Spectrum zu verbinden, um die Materialansicht `mv_sales_vw` zu erstellen. Weitere Informationen zum Befehl CREATE EXTERNALE TABLE für Amazon Redshift Spectrum finden Sie unter CREATE EXTERNAL TABLE. Die externe Redshift-Spectrum-Tabelle verweist auf die Daten in Amazon S3.

```
CREATE MATERIALIZED VIEW mv_sales_vw as
select salesid, qtysold, pricepaid, commission, saletime from public.sales
union all
select salesid, qtysold, pricepaid, commission, saletime from spectrum.sales
```

Im folgenden Beispiel wird die materialisierte Ansicht `mv_fq` erstellt, die auf einer externen Tabelle für eine Verbundabfrage basiert. Weitere Informationen zur Verbundabfrage finden Sie unter CREATE EXTERNAL SCHEMA.

```
CREATE MATERIALIZED VIEW mv_fq as select firstname, lastname from apg.mv_fq_example;

select firstname, lastname from mv_fq;
 firstname | lastname
-----------+----------
 John      | Day
 Jane      | Doe
(2 rows)
```

Das folgende Beispiel zeigt die Definition einer materialisierten Ansicht.

```
SELECT pg_catalog.pg_get_viewdef('mv_sales_vw'::regclass::oid, true);

pg_get_viewdef
-------------------------------------------------
create materialized view mv_sales_vw as select a from t;
```

Das folgende Beispiel zeigt, wie AUTO REFRESH in der Definition der materialisierten Ansicht festgelegt wird, und gibt außerdem einen DISTSTYLE an. Erstellen Sie zunächst eine einfache Basistabelle.

```
CREATE TABLE baseball_table (ball int, bat int);
```

Erstellen Sie dann eine materialisierte Ansicht.

```
CREATE MATERIALIZED VIEW mv_baseball DISTSTYLE ALL AUTO REFRESH YES AS SELECT ball AS
 baseball FROM baseball_table;
```

Jetzt können Sie die materialisierte Ansicht mv_baseball abfragen. Informationen zum Überprüfen, ob AUTO REFRESH für eine materialisierte Ansicht aktiviert ist, finden Sie unter STV_MV_INFO.

Im folgenden Beispiel wird eine materialisierte Ansicht erstellt, die auf eine Quelltabelle in einer anderen Datenbank verweist. Es wird davon ausgegangen, dass sich die Datenbank, die die Quelltabelle, database_A, enthält, in demselben Cluster oder derselben Arbeitsgruppe befindet wie Ihre materialisierte Ansicht, die Sie in database_B erstellen. (Sie können das Beispiel durch Ihre eigenen Datenbanken ersetzen.) Erstellen Sie zunächst in database_A eine Tabelle mit dem Namen cities mit einer Spalte cityname. Ändern Sie den Datentyp der Spalte in VARCHAR. Nachdem Sie die Quelltabelle erstellt haben, führen Sie den folgenden Befehl in database_B aus, um eine materialisierte Ansicht zu erstellen, deren Quelle Ihre Tabelle cities ist. Stellen Sie sicher, dass Sie die Datenbank und das Schema der Quelltabelle in der FROM-Klausel angeben:

```
CREATE MATERIALIZED VIEW cities_mv AS
SELECT  cityname
FROM    database_A.public.cities;
```

Fragen Sie die materialisierte Ansicht ab, die Sie erstellt haben. Die Abfrage ruft Datensätze ab, deren ursprüngliche Quelle die Tabelle cities in database_A ist:

```
select * from cities_mv;
```

Wenn Sie die SELECT-Anweisung ausführen, gibt cities_mv die Datensätze zurück. Datensätze aus der Quelltabelle werden nur dann aktualisiert, wenn eine REFRESH-Anweisung ausgeführt wird. Beachten Sie außerdem, dass Sie Datensätze nicht direkt in der materialisierten Ansicht aktualisieren können. Hinweise zum Aktualisieren der Daten in einer materialisierten Ansicht finden Sie unter REFRESH MATERIALIZED VIEW.

Weitere Details zur Übersicht über materialisierte Ansichten und SQL-Befehle, die zum Aktualisieren und Löschen materialisierter Ansichten verwendet werden, finden Sie in den folgenden Themen:

- Erstellen von materialisierten Ansichten in Amazon Redshift
- REFRESH MATERIALIZED VIEW
- DROP MATERIALIZED VIEW

CREATE MODEL

Themen

- Voraussetzungen
- Erforderliche Berechtigungen

- Kontrolle der Kosten
- CREATE MODEL – vollständig
- Parameter
- Nutzungshinweise
- Anwendungsfälle

Voraussetzungen

Bevor Sie die CREATE MODEL-Anweisung verwenden, erfüllen Sie die Voraussetzungen in Cluster-Einrichtung für die Verwendung von Amazon Redshift ML. Im Folgenden finden Sie eine Übersicht über die Voraussetzungen.

- Erstellen Sie einen Amazon Redshift Redshift-Cluster mit der AWS Management Console oder der AWS Befehlszeilenschnittstelle (AWS CLI).
- Fügen Sie beim Erstellen des Clusters die AWS Identity and Access Management (IAM) -Richtlinie an.
- Um Amazon Redshift SageMaker zu ermöglichen und die Rolle für die Interaktion mit anderen Diensten zu übernehmen, fügen Sie der IAM-Rolle die entsprechende Vertrauensrichtlinie hinzu.

Einzelheiten zur IAM-Rolle, zur Vertrauensrichtlinie und zu anderen Voraussetzungen finden Sie unter Cluster-Einrichtung für die Verwendung von Amazon Redshift ML.

Im Folgenden finden Sie verschiedene Anwendungsfälle für die CREATE MODEL-Anweisung.

- Einfaches CREATE MODEL
- CREATE MODEL mit Benutzerführung
- CREATE XGBoost-Modelle mit AUTO OFF
- Bring Your Own Model (BYOM) - lokale Inferenz
- CREATE MODEL mit K-MEANS
- CREATE MODEL – vollständig

Erforderliche Berechtigungen

Für CREATE MODEL sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung CREATE MODEL
- Rollen mit der Berechtigung GRANT CREATE MODEL

Kontrolle der Kosten

Amazon Redshift ML verwendet vorhandene Cluster-Ressourcen, um Vorhersagemodelle zu erstellen, so dass Sie keine zusätzlichen Kosten bezahlen müssen. Möglicherweise fallen jedoch zusätzliche Kosten an, wenn Sie die Größe Ihres Clusters ändern oder Ihre Modelle trainieren möchten. Amazon Redshift ML verwendet Amazon, SageMaker um Modelle zu trainieren, was mit zusätzlichen Kosten verbunden ist. Es gibt Möglichkeiten, zusätzliche Kosten zu kontrollieren, z. B. die Begrenzung der maximalen Dauer des Trainings oder die Begrenzung der Anzahl der Trainingsbeispiele, die zum Trainieren Ihres Modells verwendet werden. Weitere Informationen finden Sie unter Kosten für die Verwendung von Amazon Redshift ML.

CREATE MODEL – vollständig

Im Folgenden werden die grundlegenden Optionen der vollständigen CREATE MODEL-Syntax zusammengefasst.

Vollständige CREATE MODEL-Syntax

Im Folgenden finden Sie die vollständige Syntax der CREATE MODEL-Anweisung.

> **Important**
>
> Wenn Sie ein Modell mit der CREATE MODEL-Anweisung erstellen, folgen Sie der Reihenfolge der Schlüsselwörter in der folgenden Syntax.

```
CREATE MODEL model_name
    FROM { table_name | ( select_statement )  | 'job_name' }
    [ TARGET column_name ]
    FUNCTION function_name ( data_type [, ...] )
    [ RETURNS super ]
    IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
    [ AUTO ON / OFF ]
      -- default is AUTO ON
    [ MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER | KMEANS | FORECAST } ]
      -- not required for non AUTO OFF case, default is the list of all supported types
```

```
      -- required for AUTO OFF
    [ PROBLEM_TYPE ( REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION ) ]
      -- not supported when AUTO OFF
    [ OBJECTIVE ( 'MSE' | 'Accuracy' | 'F1' | 'F1_Macro' | 'AUC' |
                  'reg:squarederror' | 'reg:squaredlogerror'| 'reg:logistic'|
                  'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' |
'binary:hinge',
                  'multi:softmax' | 'RMSE' | 'WAPE' | 'MAPE' | 'MASE' |
'AverageWeightedQuantileLoss' ) ]
      -- for AUTO ON: first 5 are valid
      -- for AUTO OFF: 6-13 are valid
      -- for FORECAST: 14-18 are valid
    [ PREPROCESSORS 'string' ]
      -- required for AUTO OFF, when it has to be 'none'
      -- optional for AUTO ON
    [ HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT ( Key 'value' (,...) ) } ]
      -- support XGBoost hyperparameters, except OBJECTIVE
      -- required and only allowed for AUTO OFF
      -- default NUM_ROUND is 100
      -- NUM_CLASS is required if objective is multi:softmax (only possible for AUTO
OFF)
     [ SETTINGS (
       S3_BUCKET 'bucket',  |
         -- required
      TAGS 'string', |
         -- optional
      KMS_KEY_ID 'kms_string', |
         -- optional
      S3_GARBAGE_COLLECT on / off, |
         -- optional, defualt is on.
      MAX_CELLS integer, |
         -- optional, default is 1,000,000
      MAX_RUNTIME integer (, ...) |
         -- optional, default is 5400 (1.5 hours)
      HORIZON integer, |
         -- required if creating a forecast model
      FREQUENCY integer, |
         -- required if creating a forecast model
      PERCENTILES string
         -- optional if creating a forecast model
    ) ]
```

Parameter

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

FROM { table_name | (select_query) | 'job_name'}

Der Tabellenname oder die Abfrage, die die Trainingsdaten spezifiziert. Dabei kann es sich entweder um eine bestehende Tabelle im System oder eine Amazon-Redshift-kompatible SELECT-Abfrage handeln, die in Klammern () eingeschlossen ist. Das Abfrageergebnis muss mindestens zwei Spalten enthalten.

TARGET column_name

Der Name der Spalte, die zum Vorhersageziel wird. Die Spalte muss in der FROM-Klausel vorhanden sein.

FUNCTION function_name (data_type [, ...])

Der Name der zu erstellenden Funktion und die Datentypen der Eingabeargumente. Sie können den Schemanamen eines Schemas in Ihrer Datenbank anstelle eines Funktionsnamens angeben.

RETURNS SUPER (Vorschau)

Der Datentyp, der von dem Modell zurückgegeben werden soll. Der zurückgegebene SUPER-Datentyp gilt nur für entfernte BYOM-Modelle.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE MODEL-Befehl ausgeführt wird. Alternativ können Sie einen ARN einer IAM-Rolle angeben, um diese Rolle zu verwenden.

[AUTO ON / OFF]

Deaktiviert die automatische Erkennung des Präprozessors, des Algorithmus und der Auswahl der Hyperparameter bei CREATE MODEL. Wenn Sie bei der Erstellung eines Prognosemodells ein angeben, bedeutet dies AutoPredictor, dass ein verwendet werden soll, wobei Amazon Forecast die optimalen Kombinationen von Algorithmen auf jede Zeitreihe in Ihrem Datensatz anwendet.

MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER | KMEANS | FORECAST }

(Optional) Gibt den Modelltyp an. Sie können angeben, ob Sie ein Modell eines bestimmten Modelltyps wie XGBoost, Multilayer Perceptron (MLP), KMEANS oder Linear Learner trainieren

möchten. Dies sind alles Algorithmen, die Amazon Autopilot unterstützt. SageMaker Wenn Sie den Parameter nicht angeben, werden beim Training alle unterstützten Modelltypen nach dem besten Modell durchsucht. Sie können in Redshift ML auch ein Prognosemodell erstellen, um präzise Zeitreihenprognosen zu erhalten.

PROBLEM_TYPE (REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION)

(Optional) Gibt den Problemtyp an. Wenn Sie den Problemtyp kennen, können Sie Amazon Redshift so einschränken, dass es nur nach dem besten Modell dieses bestimmten Modelltyps sucht. Wenn Sie diesen Parameter nicht angeben, wird während des Trainings auf der Grundlage Ihrer Daten ein Problemtyp ermittelt.

ZIEL ('MSE' | 'Genauigkeit' | 'F1' | 'F1Macro' | 'AUC' | 'reg:squarederror' | 'reg:squaredlogerror' | 'reg:logistic' | 'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' | 'binary:hinge' | 'multi:softmax' | 'RMM SE' | 'WAPE' | 'MAPE' | 'MASE' | 'AverageWeightedQuantileLoss')

(Optional) Gibt den Namen der objektiven Metrik an, die zur Messung der Vorhersagequalität eines Machine-Learning-Systems verwendet wird. Diese Metrik wird während des Trainings optimiert, um die beste Schätzung der Modellparameterwerte aus den Daten zu erhalten. Wenn Sie nicht explizit eine Metrik angeben, wird standardmäßig automatisch MSE: für Regression, F1: für die binäre Klassifikation und Accuracy: für die Multiklassen-Klassifikation verwendet. Weitere Informationen zu Zielen finden Sie unter AutoML JobObjective in der Amazon SageMaker API-Referenz und unter Lernaufgabenparametern in der XGBOOST-Dokumentation. Die Werte RMSE, WAPE, MAPE, MASE und AverageWeightedQuantileLoss gelten nur für Prognosemodelle. Weitere Informationen finden Sie unter der Predictor-API-Operation. CreateAuto

PREPROCESSORS 'string'

(Optional) Gibt bestimmte Kombinationen von Präprozessoren für bestimmte Spalten an. Das Format ist eine Liste von columnSets und entsprechenden Transformern, die auf jede Gruppe von Spalten angewendet werden sollen. Amazon Redshift wendet alle Transformatoren in einer bestimmten Transformer-Liste auf alle Spalten in den entsprechenden ColumnSet Spalten an. Um beispielsweise OneHotEncoder mit Imputer eine Anwendung auf die Spalten t1 und t2 anzuwenden, verwenden Sie den folgenden Beispielbefehl.

```
CREATE MODEL customer_churn
FROM customer_data
TARGET 'Churn'
FUNCTION predict_churn
```

```
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
PROBLEM_TYPE BINARY_CLASSIFICATION
OBJECTIVE 'F1'
PREPROCESSORS '[
...
  {"ColumnSet": [
      "t1",
      "t2"
    ],
    "Transformers": [
      "OneHotEncoder",
      "Imputer"
    ]
  },
  {"ColumnSet": [
      "t3"
    ],
    "Transformers": [
      "OneHotEncoder"
    ]
  },
  {"ColumnSet": [
      "temp"
    ],
    "Transformers": [
      "Imputer",
      "NumericPassthrough"
    ]
  }
]'
SETTINGS (
  S3_BUCKET 'bucket'
)
```

HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT (key ‘value’ (,..)) }

Gibt an, ob die Standard-XGBoost-Parameter verwendet oder durch benutzerdefinierte Werte überschrieben werden. Die Werte müssen in einfache Anführungszeichen eingeschlossen werden. Im Folgenden finden Sie Beispiele für Parameter für XGBoost und deren Standardwerte.

Parameter name	Parameterwert	Standa ert	Hinweise
num_class	Ganzzahl	Erforde ich für die Multicla s-Klassif izierun	N/A
num_round	Ganzzahl	100	N/A
tree_meth od	Zeichenfolge	Automa ch	N/A
max_depth	Ganzzahl	6	[0, 10]
min_child _weight	Gleitkommazahl	1	MinValue: 0, MaxValue: 120
subsample	Gleitkommazahl	1	MinValue: 0,5, MaxValue: 1
gamma	Gleitkommazahl	0	MinValue: 0, MaxValue: 5
alpha	Gleitkommazahl	0	MinValue: 0, MaxValue: 100
eta	Gleitkommazahl	0.3	MinValue: 0,1, MaxValue: 0,5
colsample _byleve	Gleitkommazahl	1	MinValue: 0,1, MaxValue: 1
colsample _bynode	Gleitkommazahl	1	MinValue: 0,1, MaxValue: 1
colsample _bytree	Gleitkommazahl	1	MinValue: 0,5, MaxValue: 1

Parameter name	Parameterwert	Standa ert	Hinweise
Lambda	Gleitkommazahl	1	MinValue: 0, MaxValue: 100
max_delta _step	Ganzzahl	0	[0, 10]

SETTINGS (S3_BUCKET 'bucket', | TAGS 'string', | KMS_KEY_ID 'kms_string' , | S3_GARBAGE_COLLECT on / off, | MAX_CELLS integer , | MAX_RUNTIME (,...) , | HORIZON integer, | FREQUENCY forecast_frequency, | PERCENTILES array of strings)

Die S3_BUCKET-Klausel gibt den Amazon-S3-Speicherort an, der zum Speichern von Zwischenergebnissen verwendet wird.

(Optional) Der TAGS-Parameter ist eine durch Kommas getrennte Liste von Schlüssel-Wert-Paaren, die Sie verwenden können, um in Amazon und Amazon Forecast erstellte Ressourcen zu taggen SageMaker. Mit Tags können Sie Ihre Ressourcen organisieren und Kosten zuordnen. Die Werte im Paar sind optional, sodass Sie Tags mithilfe des Formats `key=value` erstellen können oder indem Sie einfach einen Schlüssel erstellen. Weitere Informationen zu Tags in Amazon Redshift finden Sie unter Markieren – Übersicht.

(Optional) Die Klausel KMS_KEY_ID gibt an, ob Amazon Redshift serverseitige Verschlüsselung mit einem AWS KMS -Schlüssel verwendet, um Daten im Ruhezustand zu schützen. Bei der Übertragung von Daten werden diese mit Secure Sockets Layer (SSL) geschützt.

(Optional) S3_GARBAGE_COLLECT { ON | OFF } gibt an, ob Amazon Redshift eine Garbage Collection für die resultierenden Datensätze, die zum Trainieren von Modellen verwendet werden, und die Modelle durchführt. Wenn diese Option auf OFF gesetzt ist, werden die resultierenden Datensätze zum Trainieren von Modellen verwendet und die Modelle bleiben in Amazon S3 und können für andere Zwecke verwendet werden. Wenn die Option auf ON gesetzt ist, löscht Amazon Redshift die Artefakte in Amazon S3, nachdem das Training abgeschlossen ist. Die Standardeinstellung ist ON.

(Optional) MAX_CELLS gibt die Anzahl der Zellen in den Trainingsdaten an. Dieser Wert ist das Produkt aus der Anzahl der Datensätze (in der Trainingsabfrage oder Tabelle) mal der Anzahl der Spalten. Der Standardwert ist 1.000.000.

(Optional) MAX_RUNTIME gibt die Höchstdauer für das Training an. Je nach Größe des Datensatzes sind die Trainingsjobs oft schneller abgeschlossen. Damit legen Sie fest, wie lange das Training maximal dauern soll. Der Standardwert ist 5.400 (90 Minuten).

HORIZON gibt die maximale Anzahl von Vorhersagen an, die das Prognosemodell zurückgeben kann. Sobald das Modell trainiert ist, können Sie diese Ganzzahl nicht mehr ändern. Dieser Parameter ist erforderlich, wenn ein Prognosemodell trainiert wird.

FREQUENCY gibt an, wie detailliert die Prognosen im Hinblick auf Zeiteinheiten sein sollen. Verfügbare Optionen sind `Y | M | W | D | H | 30min | 15min | 10min | 5min | 1min`. Dieser Parameter ist erforderlich, wenn ein Prognosemodell trainiert wird.

(Optional) PERCENTILES ist eine durch Komma getrennte Zeichenfolge, die die Prognosetypen angibt, mit denen ein Prädiktor trainiert wird. Bei den Prognosetypen kann es sich um Quantile von 0,01 bis 0,99 handeln, und zwar in Schritten von 0,01 oder höher. Sie können den Mittelwert der Prognose auch mit Mittelwert angeben. Sie können maximal fünf Prognosetypen angeben.

Nutzungshinweise

Wenn Sie CREATE MODEL verwenden, beachten Sie Folgendes:

- Die Anweisung CREATE MODELL arbeitet in einem asynchronen Modus und kehrt beim Export von Trainingsdaten nach Amazon S3 zurück. Die verbleibenden Trainingsschritte bei Amazon SageMaker finden im Hintergrund statt. Während des Trainings ist die entsprechende Inferenzfunktion sichtbar, kann aber nicht ausgeführt werden. Sie können STV_ML_MODEL_INFO abfragen, um den Trainingsstand anzusehen.
- Das Training kann bis zu 90 Minuten im Hintergrund laufen, standardmäßig im Auto-Modell, und kann verlängert werden. Um das Training abzubrechen, führen Sie einfach den Befehl DROP MODEL aus.
- Der Amazon Redshift Redshift-Cluster, den Sie zum Erstellen des Modells verwenden, und der Amazon S3 S3-Bucket, der für die Bereitstellung der Trainingsdaten und Modellartefakte verwendet wird, müssen sich in derselben AWS Region befinden.
- Verwenden Sie während des Modelltrainings Amazon Redshift und SageMaker speichern Sie Zwischenartefakte in dem von Ihnen bereitgestellten Amazon S3 S3-Bucket. Standardmäßig führt Amazon Redshift die Garbage Collection am Ende des CREATE MODEL-Vorgangs durch. Amazon Redshift entfernt diese Objekte aus Amazon S3. Um diese Artefakte in Amazon S3 beizubehalten, legen Sie die Option S3_GARBAGE COLLECT OFF fest.

- Sie müssen mindestens 500 Zeilen in den Trainingsdaten verwenden, die mit der FROM-Klausel bereitgestellt werden.
- Bei Verwendung der CREATE MODEL-Anweisung können Sie in der Klausel FROM { table_name | (select_query) } nur bis zu 256 Merkmalsspalten für die Eingabe angeben.
- Für AUTO ON können Sie folgende Spaltentypen als Trainingssatz verwenden: SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, BOOLEAN, CHAR, VARCHAR, DATE, TIME, TIMETZ, TIMESTAMP und TIMESTAMPTZ. Für AUTO OFF können Sie folgende Spaltentypen als Trainingssatz verwenden: SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE und BOOLEAN.
- Sie können nicht DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, GEOMETRY, GEOGRAPHY, HLLSKETCH, SUPER oder VARBYTE als Zielspaltentyp verwenden.
- Um die Modellgenauigkeit zu verbessern, gehen Sie wie folgt vor:
  - Fügen Sie so viele relevante Spalten wie möglich in den Befehl CREATE MODEL ein, wenn Sie die Trainingsdaten in der FROM-Klausel angeben.
  - Verwenden Sie einen größeren Wert für MAX_RUNTIME und MAX_CELLS. Größere Werte für diesen Parameter erhöhen die Kosten für das Training eines Modells.
- Die Ausführung der CREATE MODEL-Anweisung wird zurückgegeben, sobald die Trainingsdaten berechnet und in den Amazon S3 Bucket exportiert werden. Danach können Sie den Status des Trainings mit dem Befehl SHOW MODEL überprüfen. Wenn ein Modell, das im Hintergrund trainiert wird, fehlschlägt, können Sie den Fehler mit SHOW MODEL überprüfen. Sie können ein fehlerhaftes Modell nicht wiederholen. Verwenden Sie DROP MODELL, um ein fehlerhaftes Modell zu entfernen und ein neues Modell neu zu erstellen Weitere Hinweise zu SHOW MODEL finden Sie unter SHOW MODEL.
- Lokales BYOM unterstützt die gleichen Modelle, die Amazon Redshift ML für Nicht-BYOM-Fälle unterstützt. Amazon Redshift unterstützt einfache XGBoost-Modelle (mit XGBoost-Version 1.0 oder höher), KMEANS-Modelle ohne Präprozessoren und XGBoost/MLP/Linear Learner-Modelle, die von Amazon Autopilot trainiert wurden. SageMaker Letzteres wird mit von Autopilot spezifizierten Präprozessoren unterstützt, die auch von Amazon Neo unterstützt werden. SageMaker
- Wenn in Ihrem Amazon Redshift Redshift-Cluster erweitertes Routing für Ihre Virtual Private Cloud (VPC) aktiviert ist, stellen Sie sicher, dass Sie einen Amazon S3 S3-VPC-Endpunkt und einen VPC-Endpunkt für die SageMaker VPC erstellen, in der sich Ihr Cluster befindet. Auf diese Weise kann der Datenverkehr zwischen diesen Diensten während der Ausführung von CREATE MODEL durch Ihre VPC laufen. Weitere Informationen finden Sie unter SageMaker Clarify Job Amazon VPC Subnets and Security Groups.

Anwendungsfälle

Die folgenden Anwendungsfälle demonstrieren die Verwendung von CREATE MODEL je nach Ihren Anforderungen.

Einfaches CREATE MODEL

Im Folgenden werden die grundlegenden Optionen der CREATE MODEL-Syntax zusammengefasst.

Einfache CREATE MODEL-Syntax

```
CREATE MODEL model_name
    FROM { table_name | ( select_query ) }
    TARGET column_name
    FUNCTION prediction_function_name
    IAM_ROLE { default }
    SETTINGS (
      S3_BUCKET 'bucket',
      [ MAX_CELLS integer ]
    )
```

Einfache CREATE MODEL-Parameter

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

FROM { table_name | (select_query) }

Der Tabellenname oder die Abfrage, die die Trainingsdaten spezifiziert. Dabei kann es sich entweder um eine bestehende Tabelle im System oder eine Amazon-Redshift-kompatible SELECT-Abfrage handeln, die in Klammern () eingeschlossen ist. Das Abfrageergebnis muss mindestens zwei Spalten enthalten.

TARGET column_name

Der Name der Spalte, die zum Vorhersageziel wird. Die Spalte muss in der FROM-Klausel vorhanden sein.

FUNCTION prediction_function_name

Ein Wert, der den Namen der Machine-Learning-Funktion von Amazon Redshift angibt, die von CREATE MODELL generiert und zur Erstellung von Prognosen mit diesem Modell verwendet

wird. Die Funktion wird im selben Schema wie das Modellobjekt erstellt und kann überladen werden.

Machine Learning in Amazon Redshift unterstützt Modelle wie Xtreme Gradient Boosted Tree (XGBoost) für Regression und Klassifizierung.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE MODEL-Befehl ausgeführt wird. Alternativ können Sie den ARN einer IAM-Rolle angeben, um diese Rolle zu verwenden.

S3_BUCKET 'bucket'

Der Name des Amazon S3 S3-Buckets, den Sie zuvor erstellt haben, um Trainingsdaten und Artefakte zwischen Amazon Redshift und SageMaker auszutauschen. Amazon Redshift erstellt einen Unterordner in diesem Bucket, bevor die Trainingsdaten entladen werden. Wenn das Training abgeschlossen ist, löscht Amazon Redshift den erstellten Unterordner und dessen Inhalt.

MAX_CELLS Ganzzahl

Die maximale Anzahl von Zellen, die aus der FROM-Klausel exportiert werden sollen. Der Standardwert ist 1 000 000.

Die Anzahl der Zellen ist das Produkt aus der Anzahl der Zeilen in den Trainingsdaten (erzeugt durch die FROM-Klausel-Tabelle oder -Abfrage) mal der Anzahl der Spalten. Wenn die Anzahl der Zellen in den Trainingsdaten die durch den Parameter max_cells angegebene Anzahl übersteigt, führt CREATE MODEL ein Downsampling der Trainingsdaten aus der FROM-Klausel durch, um die Größe des Trainingssatzes unter MAX_CELLS zu reduzieren. Größere Trainingsdatensätze können zu einer höheren Genauigkeit führen, aber auch bedeuten, dass das Modell länger zum Trainieren braucht und mehr kostet.

Informationen zu den Kosten für die Nutzung von Amazon Redshift finden Sie unter Kosten für die Verwendung von Amazon Redshift ML.

Weitere Informationen zu den Kosten für unterschiedliche Zellanzahlen und kostenlosen Testversionen finden Sie unter Amazon-Redshift-Preise.

CREATE MODEL mit Benutzerführung

Nachfolgend finden Sie eine Beschreibung der Optionen für CREATE MODEL, die zusätzlich zu den unter beschriebenen gelten Einfaches CREATE MODEL.

Standardmäßig sucht CREATE MODEL nach der besten Kombination aus Vorverarbeitung und Modell für Ihren jeweiligen Datensatz. Vielleicht benötigen Sie zusätzliche Kontrolle über das Modell oder möchten zusätzliches Fachwissen (z. B. Problemtyp oder Zielsetzung) einführen. Wenn in einem Kundenabwanderungsszenario das Ergebnis „Kunde ist nicht aktiv“ selten ist, wird das F1-Ziel häufig dem Ziel „Genauigkeit“ vorgezogen. Modelle mit einer hohen Genauigkeit sagen wahrscheinlich immer „Kunde ist aktiv“ voraus. Dies ergibt zwar eine hohe Genauigkeit, jedoch ist der geschäftliche Wert gering. Informationen zum F1-Ziel finden Sie unter AutoML JobObjective in der Amazon SageMaker API-Referenz.

Dann folgt CREATE MODEL Ihren Vorschlägen zu den angegebenen Aspekten, wie zum Beispiel dem Ziel. Gleichzeitig ermittelt CREATE MODEL automatisch die besten Präprozessoren und die besten Hyperparameter.

CREATE MODEL mit Benutzerführung – Syntax

CREATE MODEL bietet mehr Flexibilität bei den Aspekten, die Sie angeben können, und den Aspekten, die Amazon Redshift automatisch erkennt.

```
CREATE MODEL model_name
    FROM { table_name | ( select_statement ) }
    TARGET column_name
    FUNCTION function_name
    IAM_ROLE { default }
    [ MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER} ]
    [ PROBLEM_TYPE ( REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION ) ]
    [ OBJECTIVE ( 'MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC') ]
    SETTINGS (
      S3_BUCKET 'bucket', |
      S3_GARBAGE_COLLECT { ON | OFF }, |
      KMS_KEY_ID 'kms_key_id', |
      MAX_CELLS integer, |
      MAX_RUNTIME integer (, ...)
    )
```

CREATE MODEL mit Benutzerführung – Parameter

MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER }

(Optional) Gibt den Modelltyp an. Sie können angeben, ob Sie ein Modell eines bestimmten Modelltyps wie XGBoost, Multilayer Perceptron (MLP) oder Linear Learner trainieren möchten. Dies sind alles Algorithmen, die Amazon Autopilot unterstützt. SageMaker Wenn Sie den Parameter nicht angeben, werden beim Training alle unterstützten Modelltypen nach dem besten Modell durchsucht.

PROBLEM_TYPE (REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION)

(Optional) Gibt den Problemtyp an. Wenn Sie den Problemtyp kennen, können Sie Amazon Redshift so einschränken, dass es nur nach dem besten Modell dieses bestimmten Modelltyps sucht. Wenn Sie diesen Parameter nicht angeben, wird während des Trainings auf der Grundlage Ihrer Daten ein Problemtyp ermittelt.

OBJECTIVE ('MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC')

(Optional) Gibt den Namen der objektiven Metrik an, die zur Messung der Vorhersagequalität eines Machine-Learning-Systems verwendet wird. Diese Metrik wird während des Trainings optimiert, um die beste Schätzung der Modellparameterwerte aus den Daten zu erhalten. Wenn Sie nicht explizit eine Metrik angeben, wird standardmäßig automatisch MSE: für Regression, F1: für die binäre Klassifikation und Accuracy: für die Multiklassen-Klassifikation verwendet. Weitere Informationen zu Zielen finden Sie unter AutoML JobObjective in der Amazon SageMaker API-Referenz.

MAX_CELLS Ganzzahl

(Optional) Gibt die Anzahl der Zellen in den Trainingsdaten an. Dieser Wert ist das Produkt aus der Anzahl der Datensätze (in der Trainingsabfrage oder Tabelle) mal der Anzahl der Spalten. Der Standardwert ist 1 000 000.

MAX_RUNTIME Ganzzahl

(Optional) Gibt die Höchstdauer für das Training an. Je nach Größe des Datensatzes sind die Trainingsjobs oft schneller abgeschlossen. Damit legen Sie fest, wie lange das Training maximal dauern soll. Der Standardwert ist 5.400 (90 Minuten).

S3_GARBAGE_COLLECT { ON | OFF }

(Optional) Gibt an, ob Amazon Redshift eine Garbage Collection für die resultierenden Datensätze, die zum Trainieren von Modellen verwendet werden, und die Modelle durchführt.

Wenn diese Option auf OFF gesetzt ist, werden die resultierenden Datensätze zum Trainieren von Modellen verwendet und die Modelle bleiben in Amazon S3 und können für andere Zwecke verwendet werden. Wenn die Option auf ON gesetzt ist, löscht Amazon Redshift die Artefakte in Amazon S3, nachdem das Training abgeschlossen ist. Die Standardeinstellung ist ON.

KMS_KEY_ID 'kms_key_id'

(Optional) Gibt an, ob Amazon Redshift serverseitige Verschlüsselung mit einem AWS KMS -Schlüssel verwendet, um Data-at-Rest zu schützen. Bei der Übertragung von Daten werden mit Secure Sockets Layer (SSL) geschützt.

PREPROCESSORS 'string'

(Optional) Gibt bestimmte Kombinationen von Präprozessoren für bestimmte Spalten an. Das Format ist eine Liste von columnSets und entsprechenden Transformern, die auf jede Gruppe von Spalten angewendet werden sollen. Amazon Redshift wendet alle Transformatoren in einer bestimmten Transformer-Liste auf alle Spalten in den entsprechenden ColumnSet Spalten an. Um beispielsweise OneHotEncoder mit Imputer eine Anwendung auf die Spalten t1 und t2 anzuwenden, verwenden Sie den folgenden Beispielbefehl.

```
CREATE MODEL customer_churn
FROM customer_data
TARGET 'Churn'
FUNCTION predict_churn
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
PROBLEM_TYPE BINARY_CLASSIFICATION
OBJECTIVE 'F1'
PREPROCESSORS '[
...
{"ColumnSet": [
    "t1",
    "t2"
  ],
  "Transformers": [
    "OneHotEncoder",
    "Imputer"
  ]
},
{"ColumnSet": [
    "t3"
  ],
  "Transformers": [
    "OneHotEncoder"
```

```
  ]
},
{"ColumnSet": [
    "temp"
  ],
  "Transformers": [
    "Imputer",
    "NumericPassthrough"
  ]
}
]'
SETTINGS (
S3_BUCKET 'bucket'
)
```

Amazon Redshift unterstützt die folgenden Transformer:

- OneHotEncoder — Wird normalerweise verwendet, um einen diskreten Wert in einen binären Vektor mit einem Wert ungleich Null zu kodieren. Dieser Transformer ist für viele Machine-Learning-Modelle geeignet.
- OrdinalEncoder — Kodiert diskrete Werte in eine einzelne Ganzzahl. Dieser Transformer ist für bestimmte Machine-Learning-Modelle geeignet, wie MLP und Linear Learner.
- NumericPassthrough — Übergibt die Eingabe unverändert an das Modell.
- Imputer – Füllt mit fehlenden Werten und Not-a-Number(NaN)-Werten auf.
- ImputerWithIndicator — Füllt fehlende Werte und NaN-Werte aus. Dieser Transformer erzeugt auch einen Indikator, ob Werte gefehlt haben und aufgefüllt wurden.
- Normalizer – Normalisiert Werte, was die Leistung vieler ML-Algorithmen verbessern kann.
- DateTimeVectorizer — Erzeugt eine Vektoreinbettung, die eine Spalte vom Datentyp Datetime darstellt, die in Modellen für maschinelles Lernen verwendet werden kann.
- PCA – Projiziert die Daten in einen niedrigdimensionalen Raum, um die Anzahl der Merkmale zu reduzieren und gleichzeitig so viele Informationen wie möglich zu erhalten.
- StandardScaler — Standardisiert Merkmale, indem der Mittelwert entfernt und auf die Varianz pro Einheit skaliert wird.
- MinMax — Transformiert Merkmale, indem jedes Merkmal auf einen bestimmten Bereich skaliert wird.

Amazon Redshift ML speichert die trainierten Transformer und wendet sie automatisch als Teil der Vorhersageabfrage an. Sie müssen diese nicht angeben, wenn Sie Prognosen aus Ihrem Modell erstellen.

CREATE XGBoost-Modelle mit AUTO OFF

AUTO OFF CREATE MODEL hat generell andere Ziele als das normale CREATE MODEL.

Als fortgeschrittener Benutzer, der bereits den gewünschten Modelltyp und die Hyperparameter kennt, die beim Training dieser Modelle verwendet werden sollen, können Sie CREATE MODEL mit AUTO OFF verwenden, um die automatische Erkennung von Präprozessoren und Hyperparametern durch CREATE MODEL zu deaktivieren. Dazu geben Sie explizit den Modelltyp an. XGBoost ist derzeit der einzige Modelltyp, der unterstützt wird, wenn AUTO auf OFF gesetzt ist. Sie können Hyperparameter angeben. Amazon Redshift verwendet Standardwerte für alle von Ihnen angegebenen Hyperparameter.

CREATE XGBoost-Modelle mit AUTO OFF – Syntax

```
CREATE MODEL model_name
    FROM { table_name | (select_statement ) }
    TARGET column_name
    FUNCTION function_name
    IAM_ROLE { default }
    AUTO OFF
    MODEL_TYPE XGBOOST
    OBJECTIVE { 'reg:squarederror' | 'reg:squaredlogerror' | 'reg:logistic' |
                'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' |
 'binary:hinge' |
                'multi:softmax' | 'rank:pairwise' | 'rank:ndcg' }
    HYPERPARAMETERS DEFAULT EXCEPT (
        NUM_ROUND '10',
        ETA '0.2',
        NUM_CLASS '10',
        (, ...)
    )
    PREPROCESSORS 'none'
    SETTINGS (
      S3_BUCKET 'bucket', |
      S3_GARBAGE_COLLECT { ON | OFF }, |
      KMS_KEY_ID 'kms_key_id', |
      MAX_CELLS integer, |
      MAX_RUNTIME integer (, ...)
```

```
)
```

CREATE XGBoost-Modelle mit AUTO OFF – Parameter

AUTO OFF

Deaktiviert die automatische Erkennung des Präprozessors, des Algorithmus und der Auswahl der Hyperparameter bei CREATE MODEL.

MODEL_TYPE XGBOOST

Legt fest, dass XGBOOST zum Trainieren des Modells verwendet werden soll.

OBJECTIVE str

Gibt ein vom Algorithmus erkanntes Ziel an. Amazon Redshift unterstützt reg:squarederror, reg:squaredlogerror, reg:logistic, reg:pseudohubererror, reg:tweedie, binary:logistic, binary:hinge, multi:softmax. Weitere Informationen zu diesen Zielen finden Sie unter Parameter für die Lernaufgabe in der XGBoost-Dokumentation.

HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT (key ‘value’ (,..)) }

Gibt an, ob die Standard-XGBoost-Parameter verwendet oder durch benutzerdefinierte Werte überschrieben werden. Die Werte müssen in einfache Anführungszeichen eingeschlossen werden. Im Folgenden finden Sie Beispiele für Parameter für XGBoost und deren Standardwerte.

Parameter name	Parameterwert	Standa ert	Hinweise
num_class	Ganzzahl	Erforde ich für die Multicla s-Klassif izierun	N/A
num_round	Ganzzahl	100	N/A
tree_meth od	Zeichenfolge	Autom ch	N/A

Parameter name	Parameterwert	Standardwert	Hinweise
max_depth	Ganzzahl	6	[0, 10]
min_child_weight	Gleitkommazahl	1	MinValue: 0, MaxValue: 120
subsample	Gleitkommazahl	1	MinValue: 0,5, MaxValue: 1
gamma	Gleitkommazahl	0	MinValue: 0, MaxValue: 5
alpha	Gleitkommazahl	0	MinValue: 0, MaxValue: 100
eta	Gleitkommazahl	0.3	MinValue: 0,1, MaxValue: 0,5
colsample_byleve	Gleitkommazahl	1	MinValue: 0,1, MaxValue: 1
colsample_bynode	Gleitkommazahl	1	MinValue: 0,1, MaxValue: 1
colsample_bytree	Gleitkommazahl	1	MinValue: 0,5, MaxValue: 1
Lambda	Gleitkommazahl	1	MinValue: 0, MaxValue: 100
max_delta_step	Ganzzahl	0	[0, 10]

Im folgenden Beispiel werden Daten für XGBoost vorbereitet.

```
DROP TABLE IF EXISTS abalone_xgb;

    CREATE TABLE abalone_xgb (
    length_val float,
    diameter float,
    height float,
    whole_weight float,
    shucked_weight float,
```

```
    viscera_weight float,
    shell_weight float,
    rings int,
    record_number int);

    COPY abalone_xgb
    FROM 's3://redshift-downloads/redshift-ml/abalone_xg/'
    REGION 'us-east-1'
    IAM_ROLE default
    IGNOREHEADER 1 CSV;
```

Im folgenden Beispiel wird ein XGBoost-Modell mit erweiterten Optionen, wie MODEL_TYPE, OBJECTIVE und PREPROCESSORS, erstellt.

```
DROP MODEL abalone_xgboost_multi_predict_age;

    CREATE MODEL abalone_xgboost_multi_predict_age
    FROM ( SELECT length_val,
                  diameter,
                  height,
                  whole_weight,
                  shucked_weight,
                  viscera_weight,
                  shell_weight,
                  rings
   FROM abalone_xgb WHERE record_number < 2500 )
    TARGET rings FUNCTION ml_fn_abalone_xgboost_multi_predict_age
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE XGBOOST
    OBJECTIVE 'multi:softmax'
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT EXCEPT (NUM_ROUND '100', NUM_CLASS '30')
    SETTINGS (S3_BUCKET 'your-bucket');
```

Im folgenden Beispiel wird eine Inferenzabfrage verwendet, um das Alter der Fische mit einer Datensatznummer größer als 2 500 vorherzusagen. Sie verwendet die Funktion ml_fn_abalone_xgboost_multi_predict_age, die mit dem obigen Befehl erstellt wurde.

```
select ml_fn_abalone_xgboost_multi_predict_age(length_val,
                                               diameter,
                                               height,
```

```
                                                    whole_weight,
                                                    shucked_weight,
                                                    viscera_weight,
                                                    shell_weight)+1.5 as age
from abalone_xgb where record_number > 2500;
```

Bring Your Own Model (BYOM) - lokale Inferenz

Amazon Redshift ML unterstützt die Verwendung von „Bring Your Own Model“ (BYOM) bei der lokalen oder Remote-Inferenz.

Im Folgenden werden die grundlegenden Optionen der CREATE MODEL-Syntax für BYOM zusammengefasst. Sie können ein außerhalb von Amazon Redshift mit Amazon trainiertes Modell SageMaker für datenbankinterne Inferenzen lokal in Amazon Redshift verwenden.

CREATE MODEL-Syntax für lokale Inferenz

Im Folgenden wird die CREATE MODEL-Syntax für lokale Inferenz beschrieben.

```
CREATE MODEL model_name
    FROM ('job_name' | 's3_path' )
    FUNCTION function_name ( data_type [, ...] )
    RETURNS data_type
    IAM_ROLE { default }
    [ SETTINGS (
      S3_BUCKET 'bucket', | --required
      KMS_KEY_ID 'kms_string') --optional
    ];
```

Amazon Redshift unterstützt zurzeit nur vortrainierte XGBoost-, MLP- und Linear-Learner-Modelle für BYOM. Über diesen Pfad können Sie SageMaker Autopilot und Modelle importieren, die direkt in Amazon SageMaker für lokale Inferenzen trainiert wurden.

CREATE MODEL-Parameter für lokale Inferenz

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

FROM ('job_name' | 's3_path')

Der job_name verwendet einen SageMaker Amazon-Jobnamen als Eingabe. Der Jobname kann entweder ein SageMaker Amazon-Schulungsjobname oder ein Amazon SageMaker Autopilot-

Jobname sein. Der Job muss in demselben AWS Konto erstellt werden, dem der Amazon Redshift Redshift-Cluster gehört. Um den Jobnamen zu finden, starten Sie Amazon SageMaker. Wählen Sie im Drop-down-Menü Training Training jobs (Trainingsaufträge) aus.

Der 's3_pfad' gibt den S3-Speicherort der .tar.gz-Modellartefaktdatei an, die beim Erstellen des Modells verwendet werden soll.

FUNCTION function_name (data_type [, ...])

Der Name der zu erstellenden Funktion und die Datentypen der Eingabeargumente. Sie können einen Schemamen angeben.

RETURNS data_type

Der Datentyp des Werts, der von der Funktion zurückgegeben wird.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE MODEL-Befehl ausgeführt wird.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird.

SETTINGS (S3_BUCKET 'bucket', | KMS_KEY_ID 'kms_string')

Die S3_BUCKET-Klausel gibt den Amazon-S3-Speicherort an, der zum Speichern von Zwischenergebnissen verwendet wird.

(Optional) Die KMS_KEY_ID-Klausel gibt an, ob Amazon Redshift serverseitige Verschlüsselung mit einem AWS KMS Schlüssel verwendet, um Daten im Ruhezustand zu schützen. Bei der Übertragung von Daten werden mit Secure Sockets Layer (SSL) geschützt.

Weitere Informationen finden Sie unter CREATE MODEL mit Benutzerführung.

CREATE MODEL für lokale Inferenz – Beispiel

Im folgenden Beispiel wird ein Modell erstellt, das zuvor in Amazon außerhalb von Amazon SageMaker Redshift trainiert wurde. Da der Modelltyp von Amazon Redshift ML für die lokale Inferenz unterstützt wird, erstellt das folgende CREATE MODEL eine Funktion, die lokal in Amazon Redshift verwendet werden kann. Sie können einen Namen für den SageMaker Schulungsjob angeben.

```
CREATE MODEL customer_churn
    FROM 'training-job-customer-churn-v4'
    FUNCTION customer_churn_predict (varchar, int, float, float)
    RETURNS int
    IAM_ROLE default
    SETTINGS (S3_BUCKET 'your-bucket');
```

Nachdem das Modell erstellt wurde, können Sie die Funktion customer_churn_predict mit den angegebenen Argumenttypen verwenden, um Prognosen zu erstellen.

Bring Your Own Model (BYOM) - Remote-Inferenz

Amazon Redshift ML unterstützt die Verwendung von „Bring Your Own Model“ (BYOM) für die Remote-Inferenz.

Im Folgenden werden die grundlegenden Optionen der CREATE MODEL-Syntax für BYOM zusammengefasst.

⚠ Dies ist eine Vorabveröffentlichungsdokumentation für den SUPER-Datentyp für die Eingabe in BYOM-Modelle in Amazon Redshift ML, der sich in der Vorabversion befindet. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter Beta- und Vorschauversionen in den AWS - Servicebedingungen.

Wenn Sie angeben, dass der SUPER-Datentyp als Eingabedaten und der zurückgegebene Datentyp verwendet werden sollen, bedeutet dies, dass Sie ein in Amazon SageMaker JumpStart gehostetes Large Language Model (LLM) erstellen möchten. Das Erstellen von LLMs ist derzeit nur als Vorschau-Feature verfügbar. Diese Vorschau ist im Folgenden AWS-Regionen verfügbar.

- USA Ost (Ohio): (us-east-2)
- USA Ost (Nord-Virginia): (us-east-1)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Sie können einen Amazon-Redshift-Cluster in der Vorschau erstellen, um neue Funktionen von Amazon Redshift zu testen. Sie haben nicht die Möglichkeit, diese Funktionen in der Produktion zu verwenden oder Ihren Vorschau-Cluster in einen Produktionscluster oder einen Cluster auf einem anderen Pfad zu verschieben. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Erstellen eines Clusters in der Vorschau

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Provisioned clusters dashboard (Dashboard für bereitgestellte Cluster) und dann Clusters (Cluster) aus. Die aktuellen Cluster für Ihr Konto AWS-Region sind aufgeführt. Eine Teilmenge der Eigenschaften jedes Clusters wird in den Spalten der Liste angezeigt.
3. Auf der Seite mit der Clusterliste (Clusters) wird ein Banner angezeigt, das die Vorschau vorstellt. Wählen Sie die Schaltfläche Create preview cluster (Vorschau-Cluster erstellen) aus, um die Seite zum Erstellen von Clustern zu öffnen.
4. Geben Sie Eigenschaften für Ihren Cluster ein. Wählen Sie den Vorschaupfad (Preview track) aus, der die zu testenden Funktionen enthält. Wir empfehlen, einen Namen für den Cluster zu verwenden, der darauf hinweist, dass sich dieser auf einem Vorschaupfad befindet. Wählen Sie Optionen für Ihren Cluster, einschließlich Optionen mit der Bezeichnung -preview (Vorschau), für die zu testenden Funktionen. Allgemeine Informationen zum Erstellen von Clustern finden Sie unter Erstellen eines Clusters im Amazon-Redshift-Verwaltungshandbuch.
5. Wählen Sie Vorschau-Cluster erstellen aus, um einen Cluster in der Vorschau zu erstellen.

   > **Note**
   >
   > Der `preview_2023`-Track ist der neueste verfügbare Vorschau-Track. Dieser Track unterstützt nur die Erstellung von Clustern mit RA3-Knotentypen. Der Knotentyp DC2 und alle älteren Knotentypen werden nicht unterstützt.

6. Wenn Ihr Vorschau-Cluster verfügbar ist, verwenden Sie Ihren SQL-Client, um Daten zu laden und abzufragen.

Sie können außerdem eine Vorschau-Arbeitsgruppe erstellen, um ein LLM zu erstellen. Sie können diese Features nicht in der Produktion verwenden und Ihre Vorschau-Arbeitsgruppe auch nicht in

eine andere Arbeitsgruppe verschieben. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter „Beta- und Vorschauversionen“ in den AWS -Servicebedingungen. Eine Anleitung zur Erstellung einer Vorschau-Arbeitsgruppe finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe.

CREATE MODEL-Syntax für Remote-Inferenz

Im Folgenden wird die CREATE MODEL-Syntax für Remote-Inferenz beschrieben.

```
CREATE MODEL model_name
    FUNCTION function_name ( data_type [, ...] )
    RETURNS data_type
    SAGEMAKER 'endpoint_name'['model_name']
    IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
```

CREATE MODEL-Parameter für Remote-Inferenz

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

FUNCTION fn_name ([data_type] [, ...])

Der Name der Funktion und die Datentypen der Eingabeargumente. Alle unterstützten Datentypen finden Sie unter Datentypen. `Geography`, `geometry` und `hllsketch` werden nicht unterstützt. Wenn Sie angeben, dass der SUPER-Datentyp als Eingabedaten und der zurückgegebene Datentyp verwendet werden sollen, bedeutet dies, dass Sie ein in Amazon SageMaker JumpStart gehostetes Large Language Model (LLM) erstellen möchten.

Alternativ können Sie angeben, dass nur der SUPER-Datentyp als Eingabedaten verwendet werden soll, ohne ihn auch als zurückgegebenen Datentyp zu verwenden. Die Verwendung des SUPER-Datentyps als Eingabe ist nur als Vorschaufunktion verfügbar.

Sie können anstelle eines Funktionsnamens auch einen Schemanamen angeben.

RETURNS data_type

Der Datentyp des Werts, der von der Funktion zurückgegeben wird. Alle unterstützten Datentypen finden Sie unter Datentypen. `Geography`, `geometry` und `hllsketch` werden nicht unterstützt. Wenn Sie angeben, dass der SUPER-Datentyp als Eingabedaten und der zurückgegebene Datentyp verwendet werden sollen, bedeutet dies, dass Sie ein in Amazon SageMaker JumpStart gehostetes Large Language Model (LLM) erstellen möchten.

Alternativ können Sie angeben, dass nur der SUPER-Datentyp als zurückgegebenen Datentyp verwendet werden soll, ohne ihn auch als Eingabedaten zu verwenden.

SAGEMAKER 'endpoint_name'[:'model_name']

Der Name des SageMaker Amazon-Endpunkts. Wenn der Endpunktname auf einen Multimodell-Endpunkt verweist, fügen Sie den Namen des zu verwendenden Modells hinzu. Der Endpunkt muss in derselben AWS Region wie der Amazon Redshift Redshift-Cluster gehostet werden. Um Ihren Endpunkt zu finden, starten Sie Amazon SageMaker. Wählen Sie im Drop-down-Menü Inference (Inferenz) Endpoints (Endpunkte) aus.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE MODEL-Befehl ausgeführt wird. Alternativ können Sie den ARN einer IAM-Rolle angeben, um diese Rolle zu verwenden.

Wenn das Modell auf einem SageMaker Endpunkt bereitgestellt wird, werden die Informationen des Modells in Amazon Redshift SageMaker erstellt. Anschließend führt es eine Inferenz über die externe Funktion durch. Sie können den Befehl SHOW MODEL verwenden, um die Modellinformationen auf Ihrem Amazon-Redshift-Cluster anzuzeigen.

CREATE MODEL für Remote-Inferenz – Hinweise zur Verwendung

Bevor Sie CREATE MODEL für Remote-Inferenz verwenden, sollten Sie Folgendes beachten:

- BYOM-Modelle können nur ein Argument unterstützen, wenn Sie den SUPER-Datentyp als Eingabedaten verwenden, und die zurückgegebene Ausgabe muss ebenfalls den Datentyp SUPER haben.
- Das Modell muss Eingaben im Format kommagetrennter Werte (CSV) über den Inhaltstyp Text/CSV in akzeptieren. SageMaker Gilt nur, wenn Sie nicht den Datentyp SUPER als Eingabe verwenden.
- Der Endpunkt muss von demselben AWS Konto gehostet werden, dem der Amazon Redshift Redshift-Cluster gehört.
- Bei den Ausgaben von Modellen muss es sich um einen einzelnen Wert des Typs handeln, der bei der Erstellung der Funktion angegeben wurde, und zwar im Format von kommagetrennten Werten (CSV) über den Inhaltstyp Text/CSV in. SageMaker Varchar-Datentypen dürfen nicht in

Anführungszeichen gesetzt werden. Jede Ausgabe muss in einer neuen Zeile stehen. Gilt nur, wenn Sie angegeben haben, dass das Modell den SUPER-Datentyp nicht zurückgeben soll.

- Modelle akzeptieren Nullen als leere Zeichenfolgen.
- Stellen Sie sicher, dass der SageMaker Amazon-Endpunkt entweder über genügend Ressourcen verfügt, um Inferenzrufe von Amazon Redshift aufzunehmen, oder dass der SageMaker Amazon-Endpunkt automatisch skaliert werden kann.
- Wenn der zurückgegebene Typ SUPER ist, muss die Modellausgabe JSON und der Inhaltstyp `application/jsonlines` sein.
- Wenn der Ein- und der Ausgabetyp SUPER sind, muss das Modell JSON über den Inhaltstyp `application/json` akzeptieren und zurückgeben.

CREATE MODEL für Remote-Inferenz – Beispiel

Im folgenden Beispiel wird ein Modell erstellt, das einen SageMaker Endpunkt verwendet, um Vorhersagen zu treffen. Vergewissern Sie sich, dass der Endpunkt ausgeführt wird, um Prognosen zu erstellen, und geben Sie seinen Namen in dem Befehl CREATE MODEL an.

```
CREATE MODEL remote_customer_churn
    FUNCTION remote_fn_customer_churn_predict (varchar, int, float, float)
    RETURNS int
    SAGEMAKER 'customer-churn-endpoint'
    IAM_ROLE default;
```

Im folgenden Beispiel wird ein großes Sprachmodell (LLM) erstellt, indem der SUPER-Datentyp als Eingabedaten verwendet und der SUPER-Datentyp ausgegeben wird. LLMs werden in SageMaker Jumpstart gehostet.

```
CREATE MODEL sample_super_data_model
FUNCTION sample_super_data_model_predict(super)
RETURNS super
SAGEMAKER 'sample_super_data_model_endpoint'
IAM_ROLE default;
```

CREATE MODEL mit K-MEANS

Amazon Redshift unterstützt den K-Means-Algorithmus, der Daten gruppiert, die nicht markiert sind. Dieser Algorithmus löst Clusterprobleme, bei denen Sie Gruppierungen in den Daten erkennen

möchten. Nicht klassifizierte Daten werden aufgrund ihrer Ähnlichkeiten und Unterschiede gruppiert und partitioniert.

CREATE MODEL mit K-MEANS – Syntax

```
CREATE MODEL model_name
    FROM { table_name | ( select_statement ) }
    FUNCTION function_name
    IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
    AUTO OFF
    MODEL_TYPE KMEANS
    PREPROCESSORS 'string'
    HYPERPARAMETERS DEFAULT EXCEPT ( K 'val' [, ...] )
    SETTINGS (
      S3_BUCKET 'bucket',
      KMS_KEY_ID 'kms_string', |
        -- optional
      S3_GARBAGE_COLLECT on / off, |
        -- optional
      MAX_CELLS integer, |
        -- optional
      MAX_RUNTIME integer
        -- optional);
```

CREATE MODEL mit K-MEANS – Parameter

AUTO OFF

Deaktiviert die automatische Erkennung des Präprozessors, des Algorithmus und der Auswahl der Hyperparameter bei CREATE MODEL.

MODEL_TYPE KMEANS

Legt fest, dass KMEANS zum Trainieren des Modells verwendet werden soll.

PREPROCESSORS 'string'

Gibt bestimmte Kombinationen von Präprozessoren für bestimmte Spalten an. Das Format ist eine Liste von columnSets und entsprechenden Transformern, die auf jede Gruppe von Spalten angewendet werden sollen. Amazon Redshift unterstützt 3 K-Means-Präprozessoren, nämlich StandardScaler, und MinMax. NumericPassthrough Wenn Sie keine Vorverarbeitung für K-Means anwenden möchten, wählen Sie NumericPassthrough explizit als Transformator.

Weitere Informationen zu unterstützten Transformern finden Sie unter CREATE MODEL mit Benutzerführung – Parameter.

Der K-Means-Algorithmus verwendet einen euklidischen Abstand zum Berechnen der Ähnlichkeit. Die Vorverarbeitung der Daten stellt sicher, dass die Features des Modells im gleichen Maßstab bleiben und zuverlässige Ergebnisse liefern.

HYPERPARAMETERS DEFAULT EXCEPT (K 'val' [, ...])

Gibt an, ob die K-Means-Parameter verwendet werden. Sie müssen den K-Parameter angeben, wenn Sie den K-Means-Algorithmus verwenden. Weitere Informationen finden Sie unter K-Means Hyperparameters im Amazon SageMaker Developer Guide

Im folgenden Beispiel werden Daten für K-Means vorbereitet.

```
CREATE MODEL customers_clusters
FROM customers
FUNCTION customers_cluster
IAM_ROLE default
AUTO OFF
MODEL_TYPE KMEANS
PREPROCESSORS '[
{
  "ColumnSet": [ "*" ],
  "Transformers": [ "NumericPassthrough" ]
}
]'
HYPERPARAMETERS DEFAULT EXCEPT ( K '5' )
SETTINGS (S3_BUCKET 'bucket');

select customer_id, customers_cluster(...) from customers;
customer_id | customers_cluster
--------------------
12345            1
12346            2
12347            4
12348            0
```

CREATE MODEL mit Forecast

Prognosemodelle in Redshift ML verwenden Amazon Forecast, um präzise Zeitreihenprognosen zu erstellen. Auf diese Weise können Sie historische Daten über einen bestimmten Zeitraum verwenden,

um Vorhersagen über zukünftige Ereignisse zu treffen. Zu den häufigsten Anwendungsfällen von Amazon Forecast gehören die Verwendung von Einzelhandelsproduktdaten, um zu entscheiden, wie der Lagerbestand bewertet werden soll, die Vewendung von Herstellungsmengendaten, um vorherzusagen, wie viel von einem Artikel bestellt werden soll, und die Verwendung von Web-Traffic-Daten, um zu prognostizieren, wie viel Traffic ein Webserver empfangen könnte.

Kontingentlimits von Amazon Forecast werden in Amazon-Redshift-Prognosemodellen durchgesetzt. Die maximale Anzahl von Prognosen ist beispielsweise 100, wobei diese jedoch anpassbar ist. Wenn Sie ein Prognosemodell löschen, werden die zugehörigen Ressourcen in Amazon Forecast nicht automatisch entfernt. Wenn Sie einen Redshift-Cluster löschen, werden auch alle zugehörigen Modelle entfernt.

Beachten Sie, dass Prognosemodelle derzeit nur in den folgenden Regionen verfügbar sind:

- USA Ost (Ohio): (us-east-2)
- USA Ost (Nord-Virginia): (us-east-1)
- USA West (Oregon): (us-west-2)
- Asien-Pazifik (Mumbai): (ap-south-1)
- Asien-Pazifik (Seoul): (ap-northeast-2)
- Asien-Pazifik (Singapur): (ap-southeast-1)
- Asien-Pazifik (Sydney): (ap-southeast-2)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Frankfurt) (eu-central-1)
- Europa (Irland) (eu-west-1)

CREATE MODEL mit Forecast-Syntax

```
CREATE [ OR REPLACE ] MODEL forecast_model_name
FROM { table_name | ( select_query ) }
TARGET column_name
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}
AUTO ON
MODEL_TYPE FORECAST
SETTINGS (
  S3_BUCKET 'bucket',
  HORIZON integer,
  FREQUENCY forecast_frequency
```

```
  [PERCENTILES '0.1', '0.5', '0.9']
```

CREATE MODEL mit Forecast-Parametern

forecast_model_name

Der Name des Modells Der Modellname muss eindeutig sein.

FROM { table_name | (select_query) }

Der Tabellenname oder die Abfrage, die die Trainingsdaten spezifiziert. Dabei kann es sich entweder um eine bestehende Tabelle im System oder um eine mit Amazon Redshift kompatible SELECT-Abfrage handeln, die in Klammern eingeschlossen ist. Das Tabellen- oder Abfrageergebnis muss mindestens drei Spalten haben: (1) eine Varchar-Spalte, die den Namen der Zeitreihe angibt. Jeder Datensatz kann mehrere Zeitreihen, (2) eine datetime-Spalte und (3) die vorherzusagende Zielspalte enthalten. Diese Zielspalte muss entweder eine Ganzzahl oder ein Gleitkommawert sein. Wenn Sie einen Datensatz mit mehr als drei Spalten angeben, geht Amazon Redshift davon aus, dass alle zusätzlichen Spalten Teil einer verwandten Zeitreihe sind. Beachten Sie, dass verwandte Zeitreihen vom Typ „int“ (Ganzzahl) oder „float“ (Gleitkommawert) sein müssen. Weitere Informationen zu verwandten Zeitreihen finden Sie unter Verwenden von Datasets verwandter Zeitreihen.

TARGET column_name

Der Name der Spalte, die zum Vorhersageziel wird. Die Spalte muss in der FROM-Klausel vorhanden sein.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der CREATE MODEL-Befehl ausgeführt wird. Alternativ können Sie einen ARN einer IAM-Rolle angeben, um diese Rolle zu verwenden.

AUTO ON

Aktiviert die automatische Erkennung des Algorithmus und die Auswahl der Hyperparameter bei CREATE MODEL. Wenn Sie bei der Erstellung eines Prognosemodells ein angeben, wird eine Forecast verwendet AutoPredictor, bei der Amazon Forecast die optimalen Kombinationen von Algorithmen auf jede Zeitreihe in Ihrem Datensatz anwendet.

MODEL_TYPE FORECAST

Legt fest, dass FORECAST zum Trainieren des Modells verwendet werden soll.

S3_BUCKET 'bucket'

Der Name des Buckets von Amazon Simple Storage Service, den Sie zuvor erstellt haben und der verwendet wird, um Trainingsdaten und Artefakte zwischen Amazon Redshift und Amazon Forecast auszutauschen. Amazon Redshift erstellt einen Unterordner in diesem Bucket, bevor die Trainingsdaten entladen werden. Wenn das Training abgeschlossen ist, löscht Amazon Redshift den erstellten Unterordner und dessen Inhalt.

HORIZON-Ganzzahl

Die maximale Anzahl von Vorhersagen, die das Prognosemodell zurückgeben kann. Sobald das Modell trainiert ist, können Sie diese Ganzzahl nicht mehr ändern.

FREQUENCY forecast_frequency

Gibt an, wie detailliert die Prognosen sein sollen. Verfügbare Optionen sind `Y | M | W | D | H | 30min | 15min | 10min | 5min | 1min`. Erforderlich, wenn Sie ein Prognosemodell trainieren.

PERCENTILES-Zeichenfolge

Eine durch Komma getrennte Zeichenfolge, die die Prognosetypen angibt, mit denen ein Prädiktor trainiert wird. Bei den Prognosetypen kann es sich um Quantile von 0,01 bis 0,99 handeln, und zwar in Schritten von 0,01 oder höher. Sie können den Mittelwert der Prognose auch mit Mittelwert angeben. Sie können maximal fünf Prognosetypen angeben.

Das folgende Beispiel zeigt, wie ein einfaches Prognosemodell erstellt wird.

```
CREATE MODEL forecast_example
FROM forecast_electricity_
TARGET target
IAM_ROLE 'arn:aws:iam::<account-id>:role/<role-name>'
AUTO ON
MODEL_TYPE FORECAST
SETTINGS (S3_BUCKET 'redshift-ml-bucket',
          HORIZON 24,
          FREQUENCY 'H',
          PERCENTILES '0.25,0.50,0.75,mean',
          S3_GARBAGE_COLLECT OFF);
```

Nachdem Sie das Prognosemodell erstellt haben, können Sie eine neue Tabelle mit den Prognosedaten erstellen.

```
CREATE TABLE forecast_model_results as SELECT Forecast(forecast_example)
```

Anschließend können Sie die neue Tabelle abfragen, um Vorhersagen zu erhalten.

```
SELECT * FROM forecast_model_results
```

CREATE PROCEDURE

Erstellt eine neue gespeicherte Prozedur oder ersetzt eine vorhandene Prozedur für die aktuelle Datenbank.

Weitere Informationen und Beispiele finden Sie unter Erstellen von gespeicherten Prozeduren in Amazon Redshift.

Erforderliche Berechtigungen

Sie benötigen eine der folgenden Möglichkeiten, um CREATE OR REPLACE PROCEDURE ausführen zu können:

- Für CREATE PROCEDURE:
  - Superuser
  - Benutzer mit den Berechtigungen CREATE und USAGE für das Schema, in dem die gespeicherte Prozedur erstellt wird
- Für REPLACE PROCEDURE:
  - Superuser
  - Besitzer des Verfahrens

Syntax

```
CREATE [ OR REPLACE ] PROCEDURE sp_procedure_name
  ( [ [ argname ] [ argmode ] argtype [, ...] ] )
[ NONATOMIC ]
AS $$
  procedure_body
$$ LANGUAGE plpgsql
[ { SECURITY INVOKER | SECURITY DEFINER } ]
[ SET configuration_parameter { TO value | = value } ]
```

Parameter

OR REPLACE

Eine Klausel, die angibt, dass die vorhandene Prozedur ersetzt wird, wenn eine Prozedur mit demselben Namen und denselben Eingabeargument-Datentypen bzw. derselben Signatur vorhanden ist. Sie können eine Prozedur nur durch eine neue Prozedur ersetzen, wenn diese einen identischen Satz von Datentypen definiert.

Wenn Sie eine Prozedur mit dem gleichen Namen wie eine vorhandene Prozedur, aber mit einer anderen Signatur definieren, erstellen Sie eine neue Prozedur. Mit anderen Worten: Der Prozedurname ist überladen. Weitere Informationen finden Sie unter Überladen von Prozedurnamen.

sp_procedure_name

Der Name der Prozedur. Wenn Sie einen Schemanamen angeben (beispielsweise **`myschema.myprocedure`**), wird die Funktion im angegebenen Schema erstellt. Andernfalls wird die Prozedur im aktuellen Schema erstellt. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

Es wird empfohlen, dass Sie alle gespeicherten Prozeduren mit dem Präfix benennen `sp_`. Amazon Redshift reserviert das Präfix `sp_` für gespeicherte Prozeduren. Durch die Verwendung des Präfix `sp_` können Sie sicherstellen, dass keine Konflikte zwischen den Namen Ihrer gespeicherten Prozeduren und den Namen von bereits vorhandenen oder künftigen in Amazon Redshift integrierten gespeicherten Prozeduren oder Funktionen auftreten. Weitere Informationen finden Sie unter Benennen von gespeicherten Prozeduren.

Sie können mehr als eine Prozedur mit demselben Namen definieren, wenn sich die Datentypen für die Eingabeargumente bzw. Signaturen unterscheiden. Mit anderen Worten: In diesem Fall ist der Prozedurname überladen. Weitere Informationen finden Sie unter Überladen von Prozedurnamen

[argname] [argmode] argtype

Eine Liste von Argumentnamen, Argumentmodi und Datentypen. Nur der Datentyp ist erforderlich. Name und Modus sind optional und ihre Position kann vertauscht werden.

Der Argumentmodus kann IN, OUT oder INOUT sein. Der Standardwert ist IN.

Sie können OUT- und INOUT-Argumente verwenden, um bei einem Prozeduraufruf einen oder mehrere Werte zurückzugeben. Wenn OUT- oder INOUT-Argumente vorhanden sind, gibt der

Prozeduraufruf eine Ergebnisreihe mit n Spalten zurück, wobei n die Gesamtzahl der OUT- oder INOUT-Argumente ist.

INOUT-Argumente sind gleichzeitig Ein- und Ausgabeargumente. Eingabeargumente umfassen sowohl IN- als auch INOUT-Argumente und Ausgabeargumente umfassen sowohl OUT- als auch INOUT-Argumente.

OUT-Argumente werden im Rahmen der CALL-Anweisung nicht angegeben. Sie können INOUT-Argumente in der CALL-Anweisung für gespeicherte Prozeduren angeben. INOUT-Argumente können nützlich sein, wenn Werte übergeben oder von einem verschachtelten Aufruf zurückgegeben werden und auch wenn ein zurückgegeben wird `refcursor`. Weitere Informationen zu `refcursor`-Typen finden Sie unter Cursor.

Die Argumentdatentypen können jeder beliebige Amazon-Redshift-Standarddatentyp sein. Eine weitere Möglichkeit für Argumentdatentypen ist außerdem `refcursor`.

Sie können maximal 32 Eingabeargumente und 32 Ausgabeargumente angeben.

AS $$ procedure_body $$

Ein Konstrukt, das die auszuführende Anweisung umschließt. Die Literalschlüsselwörter AS $$ und $$ sind erforderlich.

Amazon Redshift erfordert, dass Sie die Anweisung in Ihrer Prozedur unter Verwendung eines Formats umschließen, das Dollaranführung genannt wird. Alles innerhalb der Umschließung wird exakt wie angegeben übergeben. Sie müssen für Sonderzeichen keine Escape-Zeichen verwenden, da der Inhalt der Zeichenfolge wie exakt wie angegeben geschrieben wird.

Bei der Dollaranführung verwenden Sie ein Dollarzeichenpaar ($$), um den Anfang und das Ende der auszuführenden Anweisung zu kennzeichnen, wie im folgenden Beispiel gezeigt.

```
$$ my statement $$
```

Optional können Sie zwischen den beiden Dollarzeichen eine Zeichenfolge angeben, um die Identifizierung der Anweisung zu unterstützen. Die von Ihnen im umschließenden Dollarzeichenpaar am Anfang und am Ende verwendete Zeichenfolge muss identisch sein. Diese Zeichenfolge unterscheidet zwischen Groß- und Kleinschreibung und unterliegt den gleichen Einschränkungen wie Bezeichner, die nicht in einer Anführung stehen, darf jedoch keine Dollarzeichen enthalten. Im folgenden Beispiel wird die Zeichenfolge "test" verwendet.

```
$test$ my statement $test$
```

Dieses Syntax ist auch für die verschachtelte Dollaranführung nützlich. Weitere Informationen zur Dollaranführung finden Sie unter "Zeichenfolgenkonstanten in Dollaranführung" im Abschnitt Lexikalische Struktur der PostgreSQL-Dokumentation.

procedure_body

Ein Satz gültiger PL/pgSQL-Anweisungen. PL/pgSQL-Anweisungen erweitern SQL-Befehle mit prozeduralen Konstrukten, einschließlich Schleifen- und bedingten Ausdrücken, um den logischen Fluss zu steuern. Die meisten SQL-Befehle können in der Prozedur verwendet werden, einschließlich DML-Befehlen wie COPY, UNLOAD und INSERT und DDL-Befehlen wie CREATE TABLE. Weitere Informationen finden Sie unter PL/pgSQL-Sprachreferenz.

LANGUAGE plpgsql

Ein Sprachenwert. Geben Sie an `plpgsql`. Sie müssen eine Berechtigung für die Nutzung der Sprache haben, um verwenden zu können `plpgsql`. Weitere Informationen finden Sie unter GRANT.

NONATOMIC

Erstellt die gespeicherte Prozedur in einem nicht atomaren Transaktionsmodus. Im Modus NONATOMIC werden die Anweisungen innerhalb der Prozedur automatisch festgeschrieben. Zudem wird ein Fehler innerhalb der NONATOMIC-Prozedur nicht erneut ausgelöst, wenn er durch einen Ausnahmeblock behandelt wird. Weitere Informationen finden Sie unter Verwalten von Transaktionen und RAISE.

Wenn Sie eine gespeicherte Prozedur als `NONATOMIC` definieren, beachten Sie Folgendes:

- Wenn Sie Aufrufe von gespeicherten Prozeduren verschachteln, müssen alle Prozeduren in demselben Transaktionsmodus erstellt werden.
- Die Optionen `SECURITY DEFINER` und `SET configuration_parameter` werden nicht unterstützt, wenn eine Prozedur im NONATOMIC-Modus erstellt wird.
- Jeder (explizit oder implizit) geöffnete Cursor wird automatisch geschlossen, wenn ein impliziter Commit verarbeitet wird. Daher müssen Sie eine explizite Transaktion öffnen, bevor Sie eine Cursor-Schleife starten, um sicherzustellen, dass SQL innerhalb der Iteration der Schleife nicht implizit festgeschrieben wird.

SECURITY INVOKER | SECURITY DEFINER

Die Option `SECURITY DEFINER` wird nicht unterstützt, wenn `NONATOMIC` angegeben ist.

Der Sicherheitsmodus für die Prozedur bestimmt die Zugriffsrechte der Prozedur zur Laufzeit. Die Prozedur muss über die Berechtigung zum Zugreifen auf die zugrunde liegenden Datenbankobjekte verfügen.

Im SECURITY INVOKER-Modus verwendet die Prozedur die Rechte des Benutzers, der die Prozedur aufruft. Der Benutzer muss über explizite Berechtigungen für die zugrunde liegenden Datenbankobjekte verfügen. Standardmäßig ist SECURITY INVOKER festgelegt.

Im Modus SECURITY DEFINER verwendet die Prozedur die Berechtigungen des Eigentümers der Prozedur. Der Eigentümer der Prozedur ist definiert als der Benutzer, der die Prozedur zur Laufzeit besitzt. Dies muss nicht zwingend der Benutzer sein, der die Prozedur ursprünglich definiert hat. Der Benutzer, der die Prozedur aufruft, muss über Ausführungsrechte für die Prozedur verfügen. Er benötigt aber keine Rechte für die zugrunde liegenden Objekte.

SET configuration_parameter { TO value | = value }

Diese Optionen werden nicht unterstützt, wenn `NONATOMIC` angegeben ist.

Die SET-Klausel bewirkt, dass der angegebene `configuration_parameter` beim Starten der Prozedur auf den angegebenen Wert gesetzt wird. Beim Beenden der Prozedur setzte die Klausel den `configuration_parameter` dann wieder auf seinen vorherigen Wert zurück.

Nutzungshinweise

Wenn eine gespeicherte Prozedur mit der Option SECURITY DEFINER erstellt wurde, gibt Amazon Redshift beim Aufrufen der CURRENT_USER-Funktion innerhalb der gespeicherten Prozedur den Benutzernamen des Besitzers der gespeicherten Prozedur zurück.

Beispiele

Note

Wenn beim Ausführen dieser Beispiele ein Fehler ähnlich dem folgenden auftritt:

```
ERROR: 42601: [Amazon](500310) unterminated dollar-quoted string at or near "$$
```

Siehe Übersicht über gespeicherte Prozeduren in Amazon Redshift.

Das folgende Beispiel erstellt eine Prozedur mit zwei Eingabeparametern.

```
CREATE OR REPLACE PROCEDURE test_sp1(f1 int, f2 varchar(20))
AS $$
DECLARE
  min_val int;
BEGIN
  DROP TABLE IF EXISTS tmp_tbl;
  CREATE TEMP TABLE tmp_tbl(id int);
  INSERT INTO tmp_tbl values (f1),(10001),(10002);
  SELECT INTO min_val MIN(id) FROM tmp_tbl;
  RAISE INFO 'min_val = %, f2 = %', min_val, f2;
END;
$$ LANGUAGE plpgsql;
```

Note

Wenn Sie gespeicherte Prozeduren schreiben, empfehlen wir eine bewährte Methode zum Sichern sensibler Werte:
Nehmen Sie keine Hardkodierung für sensible Informationen in der gespeicherten Prozedurlogik vor. Weisen Sie beispielsweise kein Benutzerkennwort in einer CREATE USER-Anweisung im Text einer gespeicherten Prozedur zu. Dies stellt ein Sicherheitsrisiko dar, da hartkodierte Werte als Schema-Metadaten in Katalogtabellen aufgezeichnet werden können. Übergeben Sie stattdessen mithilfe von Parametern sensible Werte wie Passwörter als Argumente an die gespeicherte Prozedur.
Weitere Informationen über gespeicherte Prozeduren finden Sie unter CREATE PROCEDURE und Erstellen von gespeicherten Prozeduren in Amazon Redshift. Weitere Informationen über die Katalogtabellen finden Sie unter Systemkatalogtabellen.

Das folgende Beispiel erstellt eine Prozedur mit einem IN-Parameter, einem OUT-Parameter und einem INOUT-Parameter.

```
CREATE OR REPLACE PROCEDURE test_sp2(f1 IN int, f2 INOUT varchar(256), out_var OUT
 varchar(256))
AS $$
DECLARE
  loop_var int;
BEGIN
  IF f1 is null OR f2 is null THEN
    RAISE EXCEPTION 'input cannot be null';
  END IF;
```

```
  DROP TABLE if exists my_etl;
  CREATE TEMP TABLE my_etl(a int, b varchar);
    FOR loop_var IN 1..f1 LOOP
        insert into my_etl values (loop_var, f2);
        f2 := f2 || '+' || f2;
    END LOOP;
  SELECT INTO out_var count(*) from my_etl;
END;
$$ LANGUAGE plpgsql;
```

CREATE RLS POLICY

Erstellt eine neue RLS-Richtlinie auf Zeilenebene, um granularen Zugriff auf Datenbankobjekte bereitzustellen.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Richtlinie erstellen.

Syntax

```
CREATE RLS POLICY policy_name
[ WITH (column_name data_type [, ...]) [ [AS] relation_alias ] ]
USING ( using_predicate_exp )
```

Parameter

policy_name

Der Name der -Richtlinie.

MIT (column_name data_type [, ...])

Bestimmt den column_name und den data_type, die sich auf die Spalten der Tabellen beziehen, denen die Richtlinie zugeordnet ist.

Sie können die WITH-Klausel nur weglassen, wenn die RLS-Richtlinie (Row-Level Security) auf keine Tabellenspalten verweist, denen die Richtlinie zugeordnet ist.

ALS relation_alias

Gibt einen optionalen Alias für die Tabelle an, an die die RLS-Richtlinie (Row-Level Security) angefügt wird.

MITHILFE VON (using_predicate_exp)

Gibt einen Filter an, der auf die WHERE-Klausel der Abfrage angewendet wird. Amazon Redshift wendet ein Richtlinienprädikat vor den Benutzerprädikaten auf Abfrageebene an. Beispielsweise schränkt **`current_user = ‘joe’ and price > 10`** ein, dass Joe nur Datensätze mit einem Preis von mehr als 10 USD sehen kann.

Nutzungshinweise

Beachten Sie beim Arbeiten mit der CREATE RLS POLICY-Anweisung Folgendes:

- Amazon Redshift unterstützt Filter, die Teil einer WHERE-Klausel einer Abfrage sein können.
- Alle Richtlinien, die an eine Tabelle angefügt werden, müssen mit demselben Tabellenalias erstellt worden sein.
- Sie benötigen keine SELECT-Berechtigung für Nachschlagetabellen. Wenn Sie eine Richtlinie erstellen, gewährt Amazon Redshift die SELECT-Berechtigung für die Nachschlagetabelle für die entsprechende Richtlinie. Eine Nachschlagetabelle ist ein Tabellenobjekt, das innerhalb einer Richtliniendefinition verwendet wird.
- Die Sicherheit auf Zeilenebene von Amazon Redshift unterstützt die folgenden Objekttypen innerhalb einer Richtliniendefinition nicht: Katalogtabellen, datenbankübergreifende Relationen, externe Tabellen, reguläre Ansichten, spät bindende Ansichten, Tabellen mit aktivierten RLS-Richtlinien (Row-Level Security) und temporäre Tabellen.

Beispiele

Die folgenden SQL-Anweisungen erstellen die Tabellen, Benutzer und Rollen für das CREATE RLS POLICY-Beispiel.

```
-- Create users and roles reference in the policy statements.
CREATE ROLE analyst;

CREATE ROLE consumer;

CREATE USER bob WITH PASSWORD 'Name_is_bob_1';

CREATE USER alice WITH PASSWORD 'Name_is_alice_1';

CREATE USER joe WITH PASSWORD 'Name_is_joe_1';
```

```
GRANT ROLE sys:secadmin TO bob;

GRANT ROLE analyst TO alice;

GRANT ROLE consumer TO joe;

GRANT ALL ON TABLE tickit_category_redshift TO PUBLIC;
```

Im folgenden Beispiel wird eine Richtlinie mit dem Namen policy_concerts erstellt.

```
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');
```

CREATE ROLE

Erstellt eine neue benutzerdefinierte Rolle, bei der es sich um eine Sammlung von Berechtigungen handelt. Eine Liste der systemdefinierten Amazon-Redshift-Rollen finden Sie unter the section called “Systemdefinierte Amazon-Redshift-Rollen”. Fragen Sie SVV_ROLES ab, um die aktuell erstellten Rollen in Ihrem Cluster oder Ihrer Arbeitsgruppe anzuzeigen.

Es gibt ein Kontingent für die Anzahl der Rollen, die erstellt werden können. Weitere Informationen finden Sie unter Kontingente und Limits in Amazon Redshift im Verwaltungshandbuch zu Amazon Redshift.

Erforderliche Berechtigungen

Im Folgenden sind die erforderlichen Berechtigungen für CREATE ROLE aufgeführt.

- Superuser
- Benutzer mit der Berechtigung CREATE ROLE

Syntax

```
CREATE ROLE role_name
[ EXTERNALID external_id ]
```

Parameter

rollen_name

Der Name der Rolle. Der Rollenname muss eindeutig sein und darf nicht mit Benutzernamen identisch sein. Ein Rollenname darf kein reserviertes Wort sein.

Ein Superuser oder regulärer Benutzer mit der Berechtigung CREATE ROLE kann Rollen erstellen. Ein Benutzer, der kein Superuser ist, aber dem USAGE für die Rolle WITH GRANT OPTION und die Berechtigung ALTER gewährt wurde, kann diese Rolle jedem zuweisen.

EXTERNALID external_id

Der Bezeichner für die Rolle, die einem Identitätsanbieter zugeordnet ist. Weitere Informationen finden Sie unter Nativer Identitätsanbieter(IdP)-Verbund für Amazon Redshift.

Beispiele

Im folgenden Beispiel wird eine `sample_role1`-Rolle erstellt.

```
CREATE ROLE sample_role1;
```

Im folgenden Beispiel wird eine Rolle `sample_role1` mit einer externen ID erstellt, die einem Identitätsanbieter zugeordnet ist.

```
CREATE ROLE sample_role1 EXTERNALID "ABC123";
```

CREATE SCHEMA

Definiert ein neues Schema für die aktuelle Datenbank.

Erforderliche Berechtigungen

Für CREATE SCHEMA sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung CREATE SCHEMA

Syntax

```
CREATE SCHEMA [ IF NOT EXISTS ] schema_name [ AUTHORIZATION username ]
           [ QUOTA {quota [MB | GB | TB] | UNLIMITED} ] [ schema_element [ ... ]

CREATE SCHEMA AUTHORIZATION username[ QUOTA {quota [MB | GB | TB] | UNLIMITED} ]
 [ schema_element [ ... ] ]
```

Parameter

IF NOT EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass das Schema vorhanden ist, statt mit einem Fehler beendet zu werden, wenn das angegebene Schema bereits vorhanden ist.

Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn CREATE SCHEMA versucht, ein Schema zu erstellen, das bereits vorhanden ist.

schema_name

Der Name des neuen Schemas. Der Schemaname darf nicht sein `PUBLIC`. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

> **Note**
>
> Die Liste der Schemata im Konfigurationsparameter search_path legt die Rangfolge von identisch benannten Objekten fest, wenn sie ohne Schemanamen referenziert werden.

AUTHORIZATION

Eine Klausel, die einem bestimmten Benutzer den Besitz gewährt.

username (Benutzername

Der Name des Schemabesitzers.

schema_element

Definition für ein oder mehrere Objekte, die innerhalb des Schemas erstellt werden sollen.

QUOTA

Die maximale Menge an Speicherplatz, die das angegebene Schema verwenden kann. Dieser Platz ist die gemeinsame Speichernutzung. Er enthält alle permanenten Tabellen, materialisierten Ansichten unter dem angegebenen Schema und doppelte Kopien aller Tabellen mit ALL-Verteilung auf jedem Rechenknoten. Das Schemakontingent berücksichtigt keine temporären Tabellen, die als Teil eines temporären Namespaces oder Schemas erstellt wurden.

Um die konfigurierten Schemakontingente anzuzeigen, siehe SVV_SCHEMA_QUOTA_STATE.

Um die Datensätze anzuzeigen, bei denen die Schemakontingente überschritten wurden, siehe STL_SCHEMA_QUOTA_VIOLATIONS.

Amazon Redshift konvertiert den ausgewählten Wert in Megabyte. Gigabyte ist die Standardmaßeinheit, wenn Sie keinen Wert angeben.

Sie müssen ein Datenbank-Superuser sein, um ein Schema-Kontingent festzulegen und zu ändern. Ein Benutzer, der kein Superuser ist, aber die CREATE SCHEMA-Berechtigung besitzt, kann ein Schema mit einem definierten Kontingent erstellen. Wenn Sie ein Schema erstellen, ohne ein Kontingent zu definieren, verfügt das Schema über ein unbegrenztes Kontingent. Wenn Sie das Kontingent unter den aktuellen Wert setzen, der vom Schema verwendet wird, erlaubt Amazon Redshift keine weitere Aufnahme, bis Sie Speicherplatz freigeben. Eine DELETE-Anweisung löscht Daten aus einer Tabelle, und Speicherplatz wird nur freigegeben, wenn VACUUM ausgeführt wird.

Amazon Redshift prüft jede Transaktion auf Kontingentverletzungen, bevor die Transaktion übertragen wird. Amazon Redshift vergleicht die Größe (den von allen Tabellen in einem Schema verwendeten Speicherplatz) jedes geänderten Schemas mit dem festgelegten Kontingent. Da die Kontingentverletzungsprüfung am Ende einer Transaktion erfolgt, kann die Größenbeschränkung das Kontingent vorübergehend innerhalb einer Transaktion überschreiten, bevor sie festgeschrieben wird. Wenn eine Transaktion das Kontingent überschreitet, bricht Amazon Redshift die Transaktion ab, verhindert nachfolgende Erfassungen und macht alle Änderungen rückgängig, bis wieder freier Speicherplatz vorhanden ist. Aufgrund von VACUUM im Hintergrund und internen Bereinigungen ist es möglich, dass ein Schema zu dem Zeitpunkt, zu dem Sie es nach einer abgebrochenen Transaktion prüfen, nicht voll ist.

Ausnahmsweise ignoriert Amazon Redshift die Kontingentverletzung und führt in bestimmten Fällen Transaktionen durch. Amazon Redshift geht so für Transaktionen vor, die nur aus einer oder mehreren der folgenden Anweisungen bestehen, wenn es keine INSERT- oder COPY-Erfassungsanweisung in derselben Transaktion gibt:

- DELETE
- TRUNCATE
- VACUUM
- DROP TABLE
- ALTER TABLE APPEND, nur beim Verschieben von Daten aus dem vollständigen Schema in ein anderes nicht vollständiges Schema

UNLIMITED

Amazon Redshift legt keine Begrenzung für das Wachstum der Gesamtgröße des Schemas fest.

Einschränkungen

Amazon Redshift erzwingt die folgenden Limits für Schemata.

- Pro Datenbank sind höchstens 9900 Schemata zulässig.

Beispiele

Im folgenden Beispiel wird ein Schema mit dem Namen US_SALES erstellt, und der Benutzer DWUSER ist der Besitzer:

```
create schema us_sales authorization dwuser;
```

Im folgenden Beispiel wird ein Schema mit dem Namen US_SALES erstellt, der Benutzer DWUSER als Besitzer eingerichtet und das Kontingent auf 50 GB festgelegt.

```
create schema us_sales authorization dwuser QUOTA 50 GB;
```

Um das neue Schema anzuzeigen, führen Sie eine Abfrage für die Katalogtabelle PG_NAMESPACE aus, wie nachfolgend gezeigt:

```
select nspname as schema, usename as owner
from pg_namespace, pg_user
where pg_namespace.nspowner = pg_user.usesysid
and pg_user.usename ='dwuser';

   schema |  owner
```

```
----------+----------
 us_sales | dwuser
(1 row)
```

Im folgenden Beispiel wird entweder das Schema US_SALES erstellt, oder es erfolgt keine Aktion und es wird eine Nachricht zurückgegeben, wenn das Schema bereits vorhanden ist:

```
create schema if not exists us_sales;
```

CREATE TABLE

Erstellt eine neue Tabelle in der aktuellen Datenbank. Sie definieren eine Liste von Spalten, die jeweils Daten eines bestimmten Typs enthalten. Der Besitzer der Tabelle gibt den Befehl CREATE TABLE aus.

Erforderliche Berechtigungen

Für CREATE TABLE sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung CREATE TABLE

Syntax

```
CREATE [ [LOCAL ] { TEMPORARY | TEMP } ] TABLE
[ IF NOT EXISTS ] table_name
( { column_name data_type [column_attributes] [ column_constraints ]
  | table_constraints
  | LIKE parent_table [ { INCLUDING | EXCLUDING } DEFAULTS ] }
  [, ... ]  )
[ BACKUP { YES | NO } ]
[table_attributes]

where column_attributes are:
  [ DEFAULT default_expr ]
  [ IDENTITY ( seed, step ) ]
  [ GENERATED BY DEFAULT AS IDENTITY ( seed, step ) ]
  [ ENCODE encoding ]
  [ DISTKEY ]
  [ SORTKEY ]
```

```
  [ COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE  ]

and column_constraints are:
  [ { NOT NULL | NULL } ]
  [ { UNIQUE  |  PRIMARY KEY } ]
  [ REFERENCES reftable [ ( refcolumn ) ] ]

and table_constraints  are:
  [ UNIQUE ( column_name [, ... ] ) ]
  [ PRIMARY KEY ( column_name [, ... ] )  ]
  [ FOREIGN KEY (column_name [, ... ] ) REFERENCES reftable [ ( refcolumn ) ]


and table_attributes are:
  [ DISTSTYLE { AUTO | EVEN | KEY | ALL } ]
  [ DISTKEY ( column_name ) ]
  [ [COMPOUND | INTERLEAVED ] SORTKEY ( column_name [,...]) |  [ SORTKEY AUTO ] ]
  [ ENCODE AUTO ]
```

Parameter

LOCAL

Optional. Obwohl dieses Schlüsselwort in der Anweisung akzeptiert wird, hat es in Amazon Redshift keine Auswirkungen.

TEMPORARY | TEMP

Ein Schlüsselwort, das eine temporäre Tabelle erstellt, die nur innerhalb der aktuellen Sitzung angezeigt wird. Die Tabelle wird am Ende der Sitzung, in der sie erstellt wird, automatisch entfernt. Die temporäre Tabelle kann denselben Namen wie eine permanente Tabelle haben. Die temporäre Tabelle wird in einem eigenen, sitzungsspezifischen Schema erstellt. (Sie können für dieses Schema keinen Namen angeben.) Dieses temporäre Schema wird das erste Schema im Suchpfad. Daher hat die temporäre Tabelle Vorrang vor der permanenten Tabelle, es sei denn, Sie qualifizieren den Tabellennamen mit dem Schemanamen, um auf die permanente Tabelle zuzugreifen. Weitere Informationen zu Schemata und Rangfolgen finden Sie unter search_path.

> **Note**
>
> Standardmäßig besitzen Datenbankbenutzer aufgrund ihrer automatischen Mitgliedschaft in der Gruppe PUBLIC die Berechtigung, temporäre Tabellen zu erstellen. Um dieses

Recht für einen Benutzer abzulehnen, widerrufen Sie das TEMP-Recht in der Gruppe PUBLIC und gewähren dann das TEMP-Recht explizit nur spezifischen Benutzern oder Benutzergruppen.

IF NOT EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass die Tabelle vorhanden ist, statt mit einem Fehler beendet zu werden, wenn die angegebene Tabelle bereits vorhanden ist. Beachten Sie, dass die vorhandene Tabelle möglicherweise überhaupt nicht der Tabelle entspricht, die erstellt worden wäre. Es wird nur der Tabellenname verglichen.

Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn CREATE TABLE versucht, eine Tabelle zu erstellen, die bereits vorhanden ist.

table_name

Der Name der Tabelle, die erstellt werden soll.

Important

Wenn Sie einen Tabellennamen angeben, der mit „#“ beginnt, wird die Tabelle als temporäre Tabelle erstellt. Im Folgenden wird ein Beispiel gezeigt:

```
create table #newtable (id int);
```

Sie verwenden „#“ auch für Verweise auf die Tabelle. Beispielsweise:

```
select * from #newtable;
```

Die maximale Länge des Tabellennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Sie können UTF-8-Multibyte-Zeichen bis zu einer Länge von vier Bytes verwenden. Amazon Redshift erzwingt ein Kontingent für die Anzahl der Tabellen pro Cluster nach Knotentyp, einschließlich benutzerdefinierter temporärer Tabellen und temporärer Tabellen, die von Amazon Redshift während der Abfrageverarbeitung oder Systemwartung erstellt werden. Optional kann der Tabellenname mit dem Datenbank- und Schemanamen qualifiziert werden.

Im folgenden Beispiel ist der Datenbankname `tickit`, der Schemaname `public` und der Tabellenname `test`.

```
create table tickit.public.test (c1 int);
```

Wenn die Datenbank oder das Schema nicht vorhanden sind, wird die Tabelle nicht erstellt, und die Anweisung gibt einen Fehler zurück. Sie können in den Systemdatenbanken `template0`, `template1`, `padb_harvest` oder `sys:internal` keine Tabellen oder Ansichten erstellen.

Wenn ein Schemaname vorhanden ist, wird die neue Tabelle in diesem Schema erstellt (vorausgesetzt, der Ersteller kann auf das Schema zugreifen). Der Tabellenname muss für dieses Schema eindeutig sein. Wenn kein Schema angegeben ist, wird die Tabelle anhand des aktuellen Datenbankschemas erstellt. Wenn Sie eine temporäre Tabelle erstellen, können Sie keinen Schemanamen angeben, da sich temporäre Tabellen in einem speziellen Schema befinden.

Mehrere temporäre Tabellen mit demselben Namen können in derselben Datenbank zur gleichen Zeit vorhanden sein, wenn sie in getrennten Sitzungen erstellt wurden, da die Tabellen unterschiedlichen Schemata zugewiesen werden. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

column_name

Der Name einer Spalte, die in der neuen Tabelle erstellt werden soll. Die maximale Länge des Spaltennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Sie können UTF-8-Multibyte-Zeichen bis zu einer Länge von vier Bytes verwenden. Die maximale Anzahl der Spalten, die Sie in einer einzelnen Tabelle definieren können, ist 1.600. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

> **Note**
>
> Wenn Sie eine „breite Tabelle“ erstellen, achten Sie darauf, dass Ihre Spaltenliste nicht die Zeilenbreitengrenzen überschreitet, um während der Verarbeitung von Lasten und Abfragen sofort Ergebnisse bereitzustellen. Weitere Informationen finden Sie unter Nutzungshinweise.

data_type

Der Datentyp der Spalte, die erstellt wird. Im Fall der Spalten CHAR und VARCHAR können Sie das Schlüsselwort MAX verwenden, statt eine maximale Länge zu deklarieren. MAX legt die

maximale Länge auf 4.096 Bytes für CHAR oder 65.535 Bytes für VARCHAR fest. Die maximale Größe eines GEOMETRY-Objekts beträgt 1.048.447 Byte..

Informationen zu den Datentypen, die Amazon Redshift unterstützt, finden Sie unter Datentypen.

DEFAULT default_expr

Eine Klausel, die der Spalte einen Standarddatenwert zuweist. Der Datentyp von default_expr muss dem Datentyp der Spalte entsprechen. Der Wert DEFAULT muss ein variablenloser Ausdruck sein. Unterabfragen, Querreferenzen auf andere Spalten in der aktuellen Tabelle und benutzerdefinierte Funktionen sind nicht zulässig.

Der Ausdruck default_expr wird in jeder INSERT-Operation verwendet, die keinen Wert für die Spalte angibt. Wenn kein Standardwert angegeben ist, ist der Standardwert für die Spalte null.

Wenn eine COPY-Operation mit einer definierten Spaltenliste eine Spalte mit einem DEFAULT-Wert auslässt, fügt der Befehl COPY den Wert von default_expr ein.

IDENTITY(seed, step)

Eine Klausel, die angibt, dass es sich bei der Spalte um eine IDENTITY-Spalte handelt. Eine IDENTITY-Spalte enthält eindeutige, automatisch generierte Werte. Der Datentyp für eine IDENTITY-Spalte muss entweder INT oder BIGINT sein.

Wenn Sie Zeilen mithilfe einer `INSERT`- oder `INSERT INTO [tablename] VALUES()`-Anweisung hinzufügen, beginnen diese Werte mit dem Wert, der als Startwert angegeben ist, und werden um die Zahl gesteigert, die als Schritt bezeichnet wird.

Wenn Sie die Tabelle mithilfe einer `INSERT INTO [tablename] SELECT * FROM`- oder `COPY`-Anweisung laden, werden die Daten parallel geladen und auf die Knoten-Slices verteilt. Um sicherzustellen, dass die IDENTITY-Werte eindeutig sind, überspringt Amazon Redshift beim Erstellen von IDENTITY-Werten eine Reihe von Werten. Identity-Werte sind eindeutig, ihre Reihenfolge stimmt jedoch möglicherweise nicht mit der in den Quelldateien überein.

GENERATED BY DEFAULT AS IDENTITY(Startwert, Schritt)

Eine Anweisung, die angibt, dass es sich bei der Spalte um eine Standard-IDENTITY-Spalte handelt, und dass Sie der Spalte automatisch einen eindeutigen Wert zuweisen können. Der Datentyp für eine IDENTITY-Spalte muss entweder INT oder BIGINT sein. Wenn Sie Zeilen ohne Werte hinzufügen, beginnen diese Werte mit dem Wert, der als Startwert angegeben ist, und werden um die Zahl erhöht, die als Schritt bezeichnet wird. Informationen zur Generierung von Werten finden Sie unter IDENTITY .

Außerdem können Sie während INSERT, UPDATE oder COPY einen Wert ohne EXPLICIT_IDS angeben. Amazon Redshift verwendet diesen Wert zum Einfügen in die Identitätsspalte, anstatt den vom System generierten Wert zu verwenden. Der Wert kann ein Duplikat, ein Wert unter dem Startwert oder ein Wert zwischen Schrittwerten sein. Amazon Redshift prüft nicht die Eindeutigkeit der Werte in der Spalte. Das Angeben eines Wertes hat keinen Einfluss auf den nächsten vom System generierten Wert.

> **Note**
>
> Wenn Sie Eindeutigkeit in der Spalte benötigen, fügen Sie keinen Duplikatwert hinzu. Fügen Sie stattdessen einen eindeutigen Wert hinzu, der unter dem Startwert oder zwischen Schrittwerten liegt.

Bedenken Sie Folgendes zu Standard-Identity-Spalten:

- Standard-IDENTITY-Spalten sind NOT NULL. NULL kann nicht eingefügt werden.
- Verwenden Sie zum Einfügen eines generierten Wertes in eine Standard-Identity-Spalte das Schlüsselwort DEFAULT.

```
INSERT INTO tablename (identity-column-name) VALUES (DEFAULT);
```

- Das Überschreiben von Werten einer Standard-Identity-Spalte hat keine Auswirkungen auf den nächsten generierten Wert.
- Sie können nicht eine Standard-Identity-Spalte mit der Anweisung ALTER TABLE ADD COLUMN anhängen.
- Sie können eine Standard-Identity-Spalte mit der Anweisung ALTER TABLE APPEND anhängen.

ENCODE encoding

Die Kompressionskodierung für eine Spalte. ENCODE AUTO ist die Standardeinstellung für Tabellen. Amazon Redshift verwaltet automatisch die Komprimierungskodierung für alle Spalten in der Tabelle. Wenn Sie die Komprimierungskodierung für eine Spalte in der Tabelle angeben, wird die Tabelle nicht mehr auf ENCODE AUTO festgelegt. Amazon Redshift verwaltet nicht mehr automatisch die Komprimierungskodierung für alle Spalten in der Tabelle. Sie können die Option ENCODE AUTO für die Tabelle angeben, damit Amazon Redshift die Komprimierungskodierung für alle Spalten in der Tabelle automatisch verwalten kann.

Amazon Redshift weist den Spalten, für die Sie keine Komprimierungskodierung angeben, automatisch eine anfängliche Komprimierungskodierung zu:

- Allen Spalten in temporären Tabellen wird standardmäßig die RAW-Kompression zugewiesen.
- Spalten, die als Sortierschlüssel definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als Datentyp BOOLEAN, REAL, DOUBLE PRECISION, GEOMETRY oder GEOGRAPHY definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP oder TIMESTAMPTZ definiert sind, wird die AZ64-Komprimierung zugewiesen.
- Spalten, die als CHAR, VARCHAR oder VARBYTE definiert sind, wird LZO-Komprimierung zugewiesen.

> **Note**
>
> Wenn Sie nicht möchten, dass eine Spalte komprimiert wird, geben Sie die RAW-Kodierung explizit an.

Die folgenden compression encodings (p. 70) werden unterstützt:

- AZ64
- BYTEDICT
- DELTA
- DELTA32K
- LZO
- MOSTLY8
- MOSTLY16
- MOSTLY32
- RAW (keine Kompression)
- RUNLENGTH
- TEXT255
- TEXT32K
- ZSTD

DISTKEY

Ein Schlüsselwort, das angibt, dass die Spalte der Verteilungsschlüssel für die Tabelle ist. In eine Tabelle kann nur eine Spalte der Verteilungsschlüssel sein. Sie können das DISTKEY-Schlüsselwort nach einem Spaltennamen oder als Teil der Tabellendefinition verwenden, indem Sie die DISTKEY-Syntax (column_name) verwenden. Beide Methoden haben die gleiche Wirkung. Weitere Informationen finden Sie unter „DISTSTYLE-Parameter“ später in diesem Thema.

Als Datentyp einer Verteilungsschlüsselspalte kommen BOOLEAN, REAL, DOUBLE PRECISION, SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP oder TIMESTAMPTZ, CHAR oder VARCHAR infrage.

SORTKEY

Ein Schlüsselwort, das angibt, dass die Spalte der Sortierschlüssel für die Tabelle ist. Wenn Daten in die Tabelle geladen werden, werden die Daten anhand einer oder mehrerer Spalten sortiert, die als Sortierschlüssel bezeichnet sind. Sie können das DISTKEY-Schlüsselwort nach einem Spaltennamen verwenden, um eine einzelne Spalte als Sortierschlüssel zu bezeichnen, oder eine oder mehrere Spalten als Sortierschlüsselspalten für die Tabelle angeben, indem Sie die SORTKEY (column_name [, ...]) -Syntax verwenden. Mit dieser Syntax werden nur zusammengesetzte Sortierschlüssel erstellt.

Sie können maximal 400 SORTKEY-Spalten pro Tabelle definieren.

Als Datentyp einer Sortierschlüsselspalte kommen BOOLEAN, REAL, DOUBLE PRECISION, SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP oder TIMESTAMPTZ, CHAR oder VARCHAR infrage.

COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE

Eine Klausel, die angibt, ob bei der Suche oder Vergleichen in der Spalte zwischen Groß- und Kleinschreibung unterschieden wird (CASE_SENSITIVE) oder nicht (CASE_INSENSITIVE). Der Standardwert entspricht der aktuellen Konfiguration für die Beachtung der Groß-/Kleinschreibung in der Datenbank.

Verwenden Sie den folgenden Befehl, um die Sortierinformationen der Datenbank zu ermitteln:

```
SELECT db_collation();

db_collation
----------------
```

```
 case_sensitive
(1 row)
```

NOT NULL | NULL

NOT NULL gibt an, dass die Spalte keine Null-Werte enthalten darf. Der Standardwert NULL gibt an, dass die Spalte Null-Werte akzeptiert. IDENTITY-Spalten werden standardmäßig als NOT NULL deklariert.

UNIQUE

Ein Schlüsselwort, das angibt, dass die Spalte nur eindeutige Werte enthalten darf. Das Verhalten der Tabelleneinschränkung in Bezug auf Eindeutigkeit ist das gleiche wie im Fall von Spalteneinschränkungen, kann jedoch zusätzlich mehrere Spalten umfassen. Um eine Tabelleneinschränkung in Bezug auf Eindeutigkeit zu definieren, verwenden Sie die UNIQUE-Syntax (column_name [, ...]).

> **Important**
>
> Eindeutigkeitseinschränkungen dienen Informationszwecken und werden nicht vom System erzwungen.

PRIMARY KEY

Ein Schlüsselwort, das angibt, dass die Spalte der Primärschlüssel für die Tabelle ist. Mittels einer Spaltendefinition kann nur eine Spalte als Primärschlüssel definiert werden. Um eine Tabelleneinschränkung mit einem mehrspaltigen Primärschlüssel zu definieren, verwenden Sie die PRIMARY KEY-Syntax (column_name [, ...]).

Die Identifizierung einer Spalte als Primärschlüssel stellt Metadaten über das Design des Schemas bereit. Ein Primärschlüssel impliziert, dass sich andere Tabellen auf diesen Spaltensatz als eindeutige Bezeichner von Zeilen verlassen können. Für eine Tabelle kann jeweils nur ein Primärschlüssel angegeben werden, ob als Spalten- oder als Tabelleneinschränkung. Die Primärschlüsseleinschränkung sollte einen Spaltensatz nennen, der sich von anderen Spaltensätzen unterscheidet, der von anderen eindeutigen Einschränkungen genannt wird, die für dieselbe Tabelle definiert sind.

PRIMARY KEY-Spalten sind ebenfalls als NOT NULL definiert.

> **Important**
>
> Primärschlüsseleinschränkungen dienen lediglich Informationszwecken. Sie werden nicht vom System erzwungen, werden jedoch vom Planer verwendet.

References reftable [(refcolumn)]

Eine Klausel, die eine Fremdschlüsseleinschränkung angibt, die impliziert, dass die Spalte nur Werte enthalten darf, die Werten in der referenzierten Spalte einer Zeile der referenzierten Tabelle entsprechen. Bei den referenzierten Spalten muss es sich um die Spalten einer Eindeutigkeits- oder Primärschlüsseleinschränkung in der referenzierten Tabelle handeln.

> **Important**
>
> Fremdschlüsseleinschränkungen dienen lediglich Informationszwecken. Sie werden nicht vom System erzwungen, werden jedoch vom Planer verwendet.

LIKE parent_table [{ INCLUDING | EXCLUDING } DEFAULTS]

Eine Klausel, die eine vorhandene Tabelle angibt, aus der die neue Tabelle automatisch Spaltennamen, Datentypen und NOT NULL-Einschränkungen kopiert. Die neue Tabelle und die übergeordnete Tabelle werden entkoppelt, und Änderungen, die in der übergeordneten Tabelle ausgeführt werden, werden nicht auf die neue Tabelle angewendet. Standardausdrücke für die kopierten Spaltendefinitionen werden nur kopiert, wenn INCLUDING DEFAULTS angegeben ist. Das Standardverhalten besteht darin, Standardausdrücke auszuschließen, sodass alle Spalten der neuen Tabelle keine Standardwerte besitzen.

Tabellen, die mit der LIKE-Option erstellt werden, erben keine Primär- und Fremdschlüsseleinschränkungen. Die Eigenschaften Verteilungstyp, Sortierschlüssel, BACKUP und NULL werden von LIKE-Tabellen geerbt. Sie können sie in der Anweisung CREATE TABLE ... LIKE jedoch nicht explizit festlegen.

BACKUP { YES | NO }

Eine Klausel, die angibt, ob die Tabelle in automatisierten und manuellen Cluster-Snapshots enthalten sein sollte. Geben Sie für Tabellen wie Staging-Tabellen, die keine kritischen Daten enthalten, BACKUP NO an, um beim Erstellen und Wiederherstellen von Snapshots Verarbeitungszeit zu sparen und den Speicherplatz auf Amazon Simple Storage Service zu

reduzieren. Die Einstellung BACKUP NO wirkt sich nicht auf die automatische Replikation von Daten zu anderen Knoten innerhalb des Clusters aus. Daher werden Tabellen, für die BACKUP NO angegeben ist, bei einem Knotenausfall wiederhergestellt. Der Standardwert ist BACKUP YES.

DISTSTYLE { AUTO | EVEN | KEY | ALL }

Schlüsselwort, das den Datenverteilungsstil für die gesamte Tabelle definiert. Amazon Redshift verteilt die Zeilen einer Tabelle an die Datenverarbeitungsknoten gemäß dem für die Tabelle angegebenen Verteilungstyp. Der Standardwert ist "AUTO".

Der von Ihnen für Tabellen ausgewählte Verteilungsstil wirkt sich auf die allgemeine Leistung Ihrer Datenbank aus. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen. Die möglichen Distributionsstile sind:

- AUTO: Amazon Redshift weist auf der Grundlage der Tabellendaten den optimalen Verteilungsstil aus. Wenn beispielsweise der Verteilungsstil AUTO angegeben ist, weist Amazon Redshift kleinen Tabellen zunächst den Verteilungsstil ALL zu. Wenn die Tabelle größer wird, ändert Amazon Redshift den Verteilungsstil möglicherweise in KEY und wählt den Primärschlüssel (oder eine Spalte des zusammengesetzten Primärschlüssels) als DISTKEY. Wenn die Tabelle größer wird und keine der Spalten als DISTKEY geeignet ist, ändert Amazon Redshift den Verteilungsstil in EVEN. Die Änderung des Verteilungsstils erfolgt im Hintergrund mit minimalen Auswirkungen auf die Benutzerabfragen.

  Um den Verteilungsstil einer Tabelle anzuzeigen, führen Sie eine Abfrage für die Systemkatalogtabelle PG_CLASS aus. Weitere Informationen finden Sie unter Anzeigen von Verteilungsstilen.
- EVEN: Die Daten in der Tabelle werden gleichmäßig in einer Round-Robin-Verteilung auf die Knoten in einem Cluster verteilt. Für die Festlegung der Verteilung werden Zeilen-IDs verwendet und jedem Knoten wird ungefähr die gleiche Zahl von Zeilen zugeteilt.
- KEY: Die Daten werden anhand der Werte in der DISTKEY-Spalte verteilt. Wenn Sie verknüpfte Spalten von verknüpften Tabellen als Verteilungsschlüssel festlegen, werden die verknüpften Zeilen aus beiden Tabellen auf den Datenverarbeitungsknoten zusammen platziert. Wenn Daten zusammen platziert werden, kann der Optimierer Verknüpfungen effizienter ausführen. Wenn Sie DISTSTYLE KEY angeben, müssen Sie eine DISTKEY-Spalte benennen, entweder für die Tabelle oder als Teil der Spaltendefinition. Weitere Informationen finden Sie unter „DISTKEY-Parameter“ früher in diesem Thema.
- ALL: Eine Kopie der gesamten Tabelle wird zu jedem Knoten verteilt. Dieser Verteilungsstil stellt sicher, dass alle Zeilen, die für eine Verknüpfung erforderlich sind, auf jedem Knoten vorhanden

sind. Die Speicheranforderungen werden jedoch multipliziert und die Lade- und Wartungszeiten für die Tabelle werden verlängert. Der Verteilungsstil ALL kann die Ausführungszeit verbessern, wenn er in Verbindung mit bestimmten Dimensionstabellen verwendet wird, für die der Verteilungsstil KEY nicht geeignet ist. Die Leistungsverbesserungen müssen jedoch gegen die Wartungskosten abgewogen werden.

DISTKEY (column_name)

Eine Einschränkung, die die Spalte angibt, die als Verteilungsschlüssel für die Tabelle verwendet werden soll. Sie können das DISTKEY-Schlüsselwort nach einem Spaltennamen oder als Teil der Tabellendefinition verwenden, indem Sie die DISTKEY-Syntax (column_name) verwenden. Beide Methoden haben die gleiche Wirkung. Weitere Informationen finden Sie unter „DISTSTYLE-Parameter“ früher in diesem Thema.

[COMPOUND | INTERLEAVED] SORTKEY (column_name [,...]) | [SORTKEY AUTO]

Gibt einen oder mehrere Sortierschlüssel für die Tabelle an. Wenn Daten in die Tabelle geladen werden, werden die Daten anhand der Spalten sortiert, die als Sortierschlüssel bezeichnet sind. Sie können das DISTKEY-Schlüsselwort nach einem Spaltennamen verwenden, um eine einzelne Spalte als Sortierschlüssel zu bezeichnen, oder eine oder mehrere Spalten als Sortierschlüsselspalten für die Tabelle angeben, indem Sie die `SORTKEY (column_name [ , ... ] )`-Syntax verwenden.

Sie können optional den COMPOUND- oder INTERLEAVED-Sortierstil angeben. Wenn Sie SORTKEY mit Spalten angeben, ist der Standardwert COMPOUND. Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.

Wenn Sie keine Sortierschlüsseloptionen angeben, ist die Standardeinstellung AUTO.

Sie können pro Tabelle maximal 400 COMPOUND SORTKEY-Spalten oder 8 INTERLEAVED SORTKEY-Spalten definieren.

AUTO

Legt fest, dass Amazon Redshift einen optimalen Sortierschlüssel basierend auf den Tabellendaten zuweist. Wenn zum Beispiel der Sortierschlüssel AUTO angegeben ist, weist Amazon Redshift einer Tabelle zunächst keinen Sortierschlüssel zu. Wenn Amazon Redshift feststellt, dass ein Sortierschlüssel die Leistung von Abfragen verbessert, kann Amazon Redshift den Sortierschlüssel Ihrer Tabelle ändern. Die eigentliche Sortierung der Tabelle erfolgt durch die automatische Tabellensortierung. Weitere Informationen finden Sie unter Automatische Tabellensortierung.

Amazon Redshift ändert keine Tabellen, die bereits Sortier- oder Verteilungsschlüssel haben. Wenn eine Tabelle über einen Verteilungsschlüssel verfügt, der nie in einem JOIN verwendet wurde, kann der Schlüssel geändert werden, wenn Amazon Redshift feststellt, dass ein besserer Schlüssel vorhanden ist.

Um den Sortierschlüssel einer Tabelle anzuzeigen, fragen Sie die Systemkatalogansicht SVV_TABLE_INFO ab. Weitere Informationen finden Sie unter SVV_TABLE_INFO.
Um die Empfehlungen von Amazon Redshift Advisor für Tabellen anzuzeigen, fragen Sie die Systemkatalogansicht SVV_ALTER_TABLE_RECOMMENDATIONS ab. Weitere Informationen finden Sie unter SVV_ALTER_TABLE_RECOMMENDATIONS.
Um die von Amazon Redshift durchgeführten Aktionen anzuzeigen, fragen Sie die Systemkatalogansicht SVL_AUTO_WORKER_ACTION ab. Weitere Informationen finden Sie unter SVL_AUTO_WORKER_ACTION.

COMPOUND

Gibt an, dass die Daten mittels eines zusammengesetzten Schlüssels sortiert werden, der aus allen aufgelisteten Spalten in der Reihenfolge ihrer Auflistung besteht. Ein zusammengesetzter Sortierschlüssel ist am nützlichsten, wenn eine Abfrage Zeilen in der Reihenfolge der Sortierspalten scannt. Die Leistungsvorteile einer Sortierung mit einem zusammengesetzten Schlüssel nehmen ab, wenn Abfragen von sekundären Sortierspalten abhängig sind. Sie können maximal 400 COMPOUND SORTKEY-Spalten pro Tabelle definieren.

INTERLEAVED

Gibt an, dass die Daten mittels eines überlappenden Sortierschlüssels sortiert werden. Für einen überlappenden Sortierschlüssel können maximal acht Spalten angegeben werden.

Eine überlappende Sortierung gewichtet jede Spalte bzw. jeden Subsatz von Spalten im Sortierschlüssel gleich, so dass Abfragen nicht von der Reihenfolge der Spalten im Sortierschlüssel abhängig sind. Wenn eine Abfrage eine oder mehrere sekundäre Sortierspalten verwendet, wird die Abfrageleistung durch die überlappende Sortierung deutlich verbessert. Die überlappende Sortierung führt zu geringfügigen Overhead-Kosten für das Laden von Daten und das Bereinigen von Operationen.

> **Important**
>
> Verwenden Sie keinen Interleaved-Sortierschlüssel in Spalten mit monoton ansteigenden Attributen, wie Identitätsspalten, Daten oder Zeitstempel.

ENCODE AUTO

Ermöglicht Amazon Redshift, den Kodierungstyp für alle Spalten in der Tabelle automatisch anzupassen, um die Abfrageleistung zu optimieren. ENCODE AUTO behält die anfänglichen Kodierungstypen bei, die Sie beim Erstellen der Tabelle angeben. Wenn Amazon Redshift dann feststellt, dass ein neuer Kodierungstyp die Abfrageleistung verbessern kann, kann Amazon Redshift den Kodierungstyp der Tabellenspalten ändern. ENCODE AUTO ist die Standardeinstellung, wenn Sie keinen Kodierungstyp für eine Spalte in der Tabelle angeben.

UNIQUE (column_name [,...])

Eine Einschränkung, die angibt, dass eine Gruppe aus einer oder mehreren Spalten einer Tabelle nur eindeutige Werte enthalten darf. Das Verhalten der Tabelleneinschränkung in Bezug auf Eindeutigkeit ist das gleiche wie im Fall von Spalteneinschränkungen, kann jedoch zusätzlich mehrere Spalten umfassen. Im Kontext von Eindeutigkeitseinschränkungen werden Null-Werte nicht als gleichwertig betrachtet. Jede Eindeutigkeitseinschränkung für eine Tabelle muss einen Spaltensatz nennen, der sich von dem Spaltensatz unterscheidet, der von einer anderen Eindeutigkeits- oder Primärschlüsseleinschränkung genannt wird, die für die Tabelle definiert ist.

> **Important**
>
> Eindeutigkeitseinschränkungen dienen Informationszwecken und werden nicht vom System erzwungen.

PRIMARY KEY (column_name [,...])

Eine Einschränkung, die angibt, dass eine Spalte oder eine Reihe von Spalten einer Tabelle nur eindeutige (nicht duplizierte) Nicht-Null-Werte enthalten darf. Die Identifizierung eines Spaltensatzes als Primärschlüssel stellt darüber hinaus Metadaten zum Design des Schemas bereit. Ein Primärschlüssel impliziert, dass sich andere Tabellen auf diesen Spaltensatz als eindeutige Bezeichner von Zeilen verlassen können. Für eine Tabelle kann jeweils nur ein Primärschlüssel angegeben werden, ob als Einzelspalten- oder als Tabelleneinschränkung. Die Primärschlüsseleinschränkung sollte einen Spaltensatz nennen, der sich von anderen Spaltensätzen unterscheidet, der von anderen eindeutigen Einschränkungen genannt wird, die für dieselbe Tabelle definiert sind.

> **Important**
>
> Primärschlüsseleinschränkungen dienen lediglich Informationszwecken. Sie werden nicht vom System erzwungen, werden jedoch vom Planer verwendet.

FOREIGN KEY (column_name [, ...]) REFERENCES reftable [(refcolumn)]

Eine Einschränkung, die eine Fremdschlüsseleinschränkung angibt, die erfordert, dass eine Gruppe aus einer oder mehreren Spalten der neuen Tabelle nur Werte enthalten darf, die Werten in der referenzierten Spalte oder den Spalten einer Zeile der referenzierten Tabelle entsprechen. Wenn refcolumn ausgelassen wird, wird der Primärschlüssel von reftable verwendet. Bei den referenzierten Spalten muss es sich um die Spalten einer Eindeutigkeits- oder Primärschlüsseleinschränkung in der referenzierten Tabelle handeln.

> **Important**
>
> Fremdschlüsseleinschränkungen dienen lediglich Informationszwecken. Sie werden nicht vom System erzwungen, werden jedoch vom Planer verwendet.

Nutzungshinweise

Einschränkungen hinsichtlich Eindeutigkeit, Primärschlüssel und Fremdschlüssel dienen lediglich Informationszwecken. Sie werden von Amazon Redshift nicht erzwungen, wenn Sie eine Tabelle ausfüllen. Wenn Sie beispielsweise Daten in eine Tabelle mit Abhängigkeiten einfügen, kann der Einfügevorgang erfolgreich sein, auch wenn er gegen die Einschränkung verstößt. Dennoch werden Primärschlüssel und Fremdschlüssel als Planungshilfen verwendet und sollten deklariert werden, wenn Ihr ETL-Prozess oder ein anderer Prozess in Ihrer Anwendung ihre Integrität erzwingt. Hinweise zum Entfernen einer Tabelle mit Abhängigkeiten finden Sie unter DROP TABLE.

Limits und Kontingente

Berücksichtigen Sie beim Erstellen einer Tabelle die folgenden Grenzwerte.

- Es gibt eine Begrenzung für die maximale Anzahl von Tabellen in einem Cluster nach Knotentyp. Weitere Informationen finden Sie unter Limits im Amazon-Redshift-Verwaltungshandbuch.
- Die maximale Anzahl von Zeichen für einen Tabellennamen ist 127.

- Die maximale Anzahl der Spalten, die Sie in einer einzelnen Tabelle definieren können, ist 1.600.
- Die maximale Anzahl der SORTKEY-Spalten, die Sie in einer einzelnen Tabelle definieren können, ist 400.

Übersicht über Einstellungen auf Spalten- und Tabellenebene

Auf Spalten- oder Tabellenebene können verschiedene Attribute und Einstellungen festgelegt werden. In einigen Fällen hat die Festlegung eines Attributs oder einer Einschränkung auf der Spalten- oder Tabellenebene die gleiche Wirkung. In anderen Fällen führt dies zu unterschiedlichen Ergebnissen.

Die folgende Liste bietet eine Übersicht über Einstellungen auf Spalten- und Tabellenebene:

DISTKEY

Es gibt keinen Unterschied hinsichtlich der Wirkung, ob auf Spalten- oder ob auf Tabellenebene festgelegt.

Wenn DISTKEY festgelegt auf der Spalten- oder Tabellenebene festgelegt wird, muss DISTSTYLE auf KEY oder darf überhaupt nicht festgelegt werden. DISTSTYLE kann nur auf Tabellenebene festgelegt werden.

SORTKEY

Bei Festlegung auf Spaltenebene muss SORTKEY aus einer einzelnen Spalte bestehen. Bei Festlegung von SORTKEY auf Tabellenebene können eine oder mehrere Spalten einen zusammengesetzten oder überlappenden Sortierschlüssel bilden.

COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE

Amazon Redshift bietet keine Unterstützung für die Änderung der Groß-/Kleinschreibungskonfiguration für eine Spalte. Wenn Sie eine neue Spalte an die Tabelle anhängen, verwendet Amazon Redshift den Standardwert für die Unterscheidung von Groß- und Kleinschreibung. Amazon Redshift unterstützt beim Anhängen einer neuen Spalte nicht das Schlüsselwort COLLATE.

Weitere Informationen zur Erstellung von Datenbanken mit der Datenbanksortierung finden Sie unter CREATE DATABASE.

Weitere Informationen über die COLLATE-Funktionen finden Sie unter Funktion COLLATE.

UNIQUE

Bei Festlegung auf Spaltenebene können ein oder mehrere Schlüssel als UNIQUE festgelegt werden. Die UNIQUE-Einschränkung gilt für jede Spalte einzeln. Bei Festlegung von UNIQUE auf Tabellenebene können eine oder mehrere Spalten eine zusammengesetzte UNIQUE-Einschränkung bilden.

PRIMARY KEY

Bei Festlegung auf Spaltenebene muss PRIMARY KEY aus einer einzelnen Spalte bestehen. Bei Festlegung von PRIMARY KEY auf Tabellenebene können eine oder mehrere Spalten einen zusammengesetzten Primärschlüssel bilden.

FOREIGN KEY

Es gibt keinen Unterschied hinsichtlich der Wirkung, unabhängig davon, ob FOREIGN KEY auf Spalten- oder auf Tabellenebene festgelegt wird. Auf Spaltenebene ist die Syntax einfach `REFERENCES` reftable [(refcolumn)].

Verteilung eingehender Daten

Wenn das Hash-Verteilungsschema der eingehenden Daten dem der Zieltabelle entspricht, ist keine physische Verteilung der Daten notwendig, wenn die Daten geladen werden. Wenn beispielsweise ein Verteilungsschlüssel für die neue Tabelle festgelegt wird und die Daten aus einer anderen Tabelle eingefügt werden, die anhand der gleichen Schlüsselspalte verteilt wird, werden die Daten unter Verwendung derselben Knoten und Slices entsprechend geladen. Wenn jedoch sowohl die Quell- als auch die Zieltabelle auf eine EVEN-Verteilung festgelegt sind, werden die Daten neu zur Zieltabelle verteilt.

Breite Tabellen

Möglicherweise können Sie eine sehr breite Tabelle erstellen, können jedoch für die Tabelle keine Abfragen wie INSERT oder SELECT verarbeiten. Die maximale Breite einer Tabelle mit Spalten fester Breite, z. B. CHAR, beträgt 64 KB - 1 (oder 65535 Bytes). Wenn eine Tabelle VARCHAR-Spalten enthält, kann die Tabelle eine größere deklarierte Breite haben, ohne einen Fehler zurückzugeben, da VARCHAR-Spalten nicht die gesamte deklarierte Breite zum berechneten Limit für die Abfrageverarbeitung beitragen. Das effektive Limit für die Abfrageverarbeitung für VARCHAR-Spalten ist von einer Reihe von Faktoren abhängig.

Wenn eine Tabelle zu breit für das Einfügen oder Auswählen ist, wird folgender Fehler gemeldet.

```
ERROR:  8001
DETAIL:  The combined length of columns processed in the SQL statement
exceeded the query-processing limit of 65535 characters (pid:7627)
```

Beispiele

Unter dem Thema Beispiele finden Sie Beispiele für die Verwendung des Befehls CREATE TABLE.

Beispiele

Im folgenden Beispiel werden verschiedene Spalten- und Tabellenattribute in CREATE TABLE-Anweisungen von Amazon Redshift gezeigt. Weitere Informationen zu CREATE TABLE, einschließlich Parameterdefinitionen, finden Sie unter CREATE TABLE.

In vielen der Beispiele werden Tabellen und Daten aus dem TICKIT-Beispieldatensatz verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

In einem Befehl CREATE TABLE können Sie dem Tabellennamen den Datenbanknamen und den Schemanamen voranstellen. Beispiel: `dev_database.public.sales` Der Datenbankname muss die Datenbank bezeichnen, mit der Sie verbunden sind. Jeder Versuch, Datenbankobjekte in einer anderen Datenbank zu erstellen, schlägt mit einem Fehler aufgrund einer ungültigen Operation fehl.

Erstellen einer Tabelle mit einem Verteilungsschlüssel, einem zusammengesetzten Sortierschlüssel und Kompression

Im folgenden Beispiel wird eine SALES-Tabelle in der Datenbank TICKIT erstellt, in der für verschiedene Tabellen Kompression definiert wird. LISTID ist als Verteilungsschlüssel deklariert und LISTID und SELLERID sind als zusammengesetzter mehrspaltiger Sortierschlüssel deklariert. Darüber hinaus sind Primärschlüssel- und Fremdschlüsseleinschränkungen für die Tabelle definiert. Bevor Sie die Tabelle in dem Beispiel erstellen, müssen Sie möglicherweise jeder Spalte, auf die durch einen Fremdschlüssel verwiesen wird, eine Einschränkung UNIQUE hinzufügen, falls keine Einschränkungen existieren.

```
create table sales(
salesid integer not null,
listid integer not null,
sellerid integer not null,
buyerid integer not null,
eventid integer not null encode mostly16,
```

```
dateid smallint not null,
qtysold smallint not null encode mostly8,
pricepaid decimal(8,2) encode delta32k,
commission decimal(8,2) encode delta32k,
saletime timestamp,
primary key(salesid),
foreign key(listid) references listing(listid),
foreign key(sellerid) references users(userid),
foreign key(buyerid) references users(userid),
foreign key(dateid) references date(dateid))
distkey(listid)
compound sortkey(listid,sellerid);
```

Es folgen die Ergebnisse:

```
schemaname | tablename | column     | type                          | encoding | distkey
 | sortkey | notnull
-----------+-----------+------------+-------------------------------+----------+---------
+---------+--------
public     | sales     | salesid    | integer                       | lzo      | false
 |       0 | true
public     | sales     | listid     | integer                       | none     | true
 |       1 | true
public     | sales     | sellerid   | integer                       | none     | false
 |       2 | true
public     | sales     | buyerid    | integer                       | lzo      | false
 |       0 | true
public     | sales     | eventid    | integer                       | mostly16 | false
 |       0 | true
public     | sales     | dateid     | smallint                      | lzo      | false
 |       0 | true
public     | sales     | qtysold    | smallint                      | mostly8  | false
 |       0 | true
public     | sales     | pricepaid  | numeric(8,2)                  | delta32k | false
 |       0 | false
public     | sales     | commission | numeric(8,2)                  | delta32k | false
 |       0 | false
public     | sales     | saletime   | timestamp without time zone | lzo      | false
 |       0 | false
```

Das folgende Beispiel erstellt die Tabelle t1 mit einer groß-/kleinschreibungsneutralen Spalte col1.

```
create table T1 (
```

```
  col1 Varchar(20) collate case_insensitive
 );

insert into T1 values ('bob'), ('john'), ('Tom'), ('JOHN'), ('Bob');
```

Fragen Sie die Tabelle ab:

```
select * from T1 where col1 = 'John';

col1
------
 john
 JOHN
(2 rows)
```

Erstellen einer Tabelle mit einem überlappenden Sortierschlüssel

Im folgenden Beispiel wird die Tabelle CUSTOMER mit einem überlappenden Sortierschlüssel erstellt.

```
create table customer_interleaved (
  c_custkey     integer        not null,
  c_name        varchar(25)    not null,
  c_address     varchar(25)    not null,
  c_city        varchar(10)    not null,
  c_nation      varchar(15)    not null,
  c_region      varchar(12)    not null,
  c_phone       varchar(15)    not null,
  c_mktsegment      varchar(10)    not null)
diststyle all
interleaved sortkey (c_custkey, c_city, c_mktsegment);
```

Erstellen einer Tabelle mit IF NOT EXISTS

Im folgenden Beispiel wird entweder die Tabelle CITIES erstellt oder es erfolgt keine Aktion und es wird eine Meldung zurückgegeben, wenn die Tabelle bereits vorhanden ist:

```
create table if not exists cities(
cityid integer not null,
city varchar(100) not null,
state char(2) not null);
```

Erstellen einer Tabelle mit der Verteilung ALL

Im folgenden Beispiel wird die Tabelle VENUE mit der Verteilung ALL erstellt.

```
create table venue(
venueid smallint not null,
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer,
primary key(venueid))
diststyle all;
```

Erstellen einer Tabelle mit der Verteilung EVEN

Im folgenden Befehl wird eine Tabelle namens MYEVENT mit drei Spalten erstellt.

```
create table myevent(
eventid int,
eventname varchar(200),
eventcity varchar(30))
diststyle even;
```

Die Tabelle wird gleichmäßig verteilt und ist nicht sortiert. Für die Tabelle wurden keine DISTKEY- oder SORTKEY-Spalten deklariert.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'myevent';

  column   |          type          | encoding | distkey | sortkey
-----------+------------------------+----------+---------+---------
 eventid   | integer                | lzo      | f       |       0
 eventname | character varying(200) | lzo      | f       |       0
 eventcity | character varying(30)  | lzo      | f       |       0
(3 rows)
```

Erstellen einer temporären Tabelle, die wie eine andere Tabelle ist (LIKE)

Im folgenden Befehl wird eine temporäre Tabelle namens TEMPEVENT erstellt, deren Spalten aus der Tabelle EVENT geerbt werden.

```
create temp table tempevent(like event);
```

Diese Tabelle erbt darüber hinaus die Attribute DISTKEY und SORTKEY der übergeordneten Tabelle:

```
select "column", type, encoding, distkey, sortkey
 from pg_table_def where tablename = 'tempevent';

  column   |            type             | encoding | distkey | sortkey
-----------+-----------------------------+----------+---------+---------
 eventid   | integer                     | none     | t       |       1
 venueid   | smallint                    | none     | f       |       0
 catid     | smallint                    | none     | f       |       0
 dateid    | smallint                    | none     | f       |       0
 eventname | character varying(200)      | lzo      | f       |       0
 starttime | timestamp without time zone | bytedict | f       |       0
(6 rows)
```

Erstellen einer Tabelle mit einer IDENTITY-Spalte

Im folgenden Beispiel wird eine Tabelle namens VENUE_IDENT erstellt, die eine IDENTITY-Spalte namens VENUEID besitzt. Diese Spalte beginnt mit 0 und wird für jeden Datensatz um 1 erhöht. VENUEID ist auch als Primärschlüssel der Tabelle deklariert.

```
create table venue_ident(venueid bigint identity(0, 1),
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer,
primary key(venueid));
```

Erstellen einer Tabelle mit einer Standard-IDENTITY-Spalte

Im folgenden Beispiel wird eine Tabelle mit dem Namen erstellt `t1`. Diese Tabelle hat eine IDENTITY-Spalte mit dem, Namen `hist_id` und eine Standard-IDENTITY-Spalte mit dem Namen `base_id`.

```
CREATE TABLE t1(
  hist_id BIGINT IDENTITY NOT NULL, /* Cannot be overridden */
  base_id BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL, /* Can be overridden */
  business_key varchar(10) ,
  some_field varchar(10)
);
```

Die Einfügung einer Zeile in die Tabelle zeigt, dass `hist_id`- und `base_id`-Werte generiert werden.

```
INSERT INTO T1 (business_key, some_field) values ('A','MM');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
```

Die Einfügung einer zweiten Zeile zeigt, dass der Standardwert für `base_id` generiert wird.

```
INSERT INTO T1 (base_id, business_key, some_field) values (DEFAULT, 'B','MNOP');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
       2 |       2 | B            | MNOP
```

Die Einfügung einer dritten Zeile zeigt, dass der Wert für `base_id` nicht eindeutig sein muss.

```
INSERT INTO T1 (base_id, business_key, some_field) values (2,'B','MNNN');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
       2 |       2 | B            | MNOP
       3 |       2 | B            | MNNN
```

Erstellen einer Tabelle mit DEFAULT-Spaltenwerten

Im folgenden Beispiel wird die Tabelle CATEGORYDEF erstellt, die für jede Spalte Standardwerte deklariert:

```
create table categorydef(
catid smallint not null default 0,
```

```
catgroup varchar(10) default 'Special',
catname varchar(10) default 'Other',
catdesc varchar(50) default 'Special events',
primary key(catid));

insert into categorydef values(default,default,default,default);
```

```
select * from categorydef;

 catid | catgroup | catname |    catdesc
-------+----------+---------+---------------
     0 | Special  | Other   | Special events
(1 row)
```

Die Optionen DISTSTYLE, DISTKEY und SORTKEY

Im folgenden Beispiel wird gezeigt, wie die Optionen DISTSTYLE, DISTKEY und SORTKEY funktionieren. In diesem Beispiel ist COL1 der Verteilungsschlüssel. Daher muss der Verteilungsstil entweder auf KEY festgelegt werden oder wird nicht festgelegt. Standardmäßig besitzt die Tabelle keinen Sortierschlüssel und ist daher nicht sortiert:

```
create table t1(col1 int distkey, col2 int) diststyle key;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't1';

column |  type   | encoding | distkey | sortkey
-------+---------+----------+---------+---------
col1   | integer | az64     | t       | 0
col2   | integer | az64     | f       | 0
```

Im folgenden Beispiel wird dieselbe Spalte als Verteilungs- und Sortierschlüssel definiert. Auch hier muss der Verteilungsstil entweder auf KEY festgelegt werden oder wird nicht festgelegt.

```
create table t2(col1 int distkey sortkey, col2 int);
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't2';

column |  type   | encoding | distkey | sortkey
```

```
-------+---------+----------+---------+---------
col1   | integer | none     | t       | 1
col2   | integer | az64     | f       | 0
```

Im folgenden Beispiel wird keine Spalte als Verteilungsschlüssel festgelegt, COL2 ist als Sortierschlüssel festgelegt und der Verteilungsstil ist auf ALL festgelegt:

```
create table t3(col1 int, col2 int sortkey) diststyle all;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't3';

Column |  Type   | Encoding | DistKey | SortKey
-------+---------+----------+---------+--------
col1   | integer | az64     | f       | 0
col2   | integer | none     | f       | 1
```

Im folgenden Beispiel ist der Verteilungsstil auf EVEN festgelegt, und es ist kein Sortierschlüssel explizit festgelegt. Daher wird die Tabelle gleichmäßig verteilt, jedoch nicht sortiert.

```
create table t4(col1 int, col2 int) diststyle even;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't4';

             column |  type   |encoding | distkey | sortkey
--------+---------+---------+---------+--------
col1    | integer | az64    | f       | 0
col2    | integer | az64    | f       | 0
```

Erstellen einer Tabelle mit der Option ENCODE AUTO

Im folgenden Beispiel wird die Tabelle `t1` mit automatischer Komprimierungskodierung erstellt. ENCODE AUTO ist der Standard für Tabellen, wenn Sie für keine Spalte einen Kodierungstyp angeben.

```
create table t1(c0 int, c1 varchar);
```

Im folgenden Beispiel wird die Tabelle `t2` mit automatischer Komprimierungskodierung erstellt, indem ENCODE AUTO angegeben wird.

```
create table t2(c0 int, c1 varchar) encode auto;
```

Im folgenden Beispiel wird die Tabelle `t3` mit automatischer Komprimierungskodierung erstellt, indem ENCODE AUTO angegeben wird. Spalte `c0` wird mit dem anfänglichen Kodierungstyp DELTA definiert. Amazon Redshift kann die Kodierung ändern, wenn eine andere Kodierung eine bessere Abfrageleistung bietet.

```
create table t3(c0 int encode delta, c1 varchar) encode auto;
```

Im folgenden Beispiel wird die Tabelle `t4` mit automatischer Komprimierungskodierung erstellt, indem ENCODE AUTO angegeben wird. Die Spalte `c0` wird mit einer Anfangskodierung von DELTA definiert, und die Spalte `c1` wird mit einer Anfangskodierung von LZO definiert. Amazon Redshift kann diese Kodierungen ändern, wenn andere Kodierungen eine bessere Abfrageleistung liefern.

```
create table t4(c0 int encode delta, c1 varchar encode lzo) encode auto;
```

CREATE TABLE AS

Themen

Erstellt eine neue Tabelle auf der Basis einer Abfrage. Der Besitzer dieser Tabelle ist der Benutzer, der den Befehl ausgibt.

Die neue Tabelle wird mit Daten geladen, die durch die Abfrage im Befehl definiert werden. Die Tabellenspalten haben Namen und Datentypen, die mit den Ausgabespalten der Abfrage verknüpft sind. Der Befehl CREATE TABLE AS (CTAS) erstellt eine neue Tabelle und evaluiert die Abfrage, um die neue Tabelle zu laden.

Syntax

```
CREATE [ [ LOCAL ] { TEMPORARY | TEMP } ]
TABLE table_name
```

```
[ ( column_name [, ... ] ) ]
[ BACKUP { YES | NO } ]
[ table_attributes ]
AS query

where table_attributes are:
[ DISTSTYLE { AUTO | EVEN | ALL | KEY } ]
[ DISTKEY( distkey_identifier ) ]
[ [ COMPOUND | INTERLEAVED ] SORTKEY( column_name [, ...] ) ]
```

Parameter

LOCAL

Obwohl dieses optionale Schlüsselwort in der Anweisung akzeptiert wird, hat es in Amazon Redshift keine Auswirkungen.

TEMPORARY | TEMP

Erstellt eine temporäre Tabelle. Eine temporäre Tabelle wird am Ende der Sitzung, in der sie erstellt wurde, automatisch entfernt.

table_name

Der Name der Tabelle, die erstellt werden soll.

> **Important**
>
> Wenn Sie einen Tabellennamen angeben, der mit „#“ beginnt, wird die Tabelle als temporäre Tabelle erstellt. Beispiel:
>
> ```
> create table #newtable (id) as select * from oldtable;
> ```

Die maximale Länge des Tabellennamens beträgt 127 Bytes; längere Namen werden bei 127 Bytes abgeschnitten. Amazon Redshift erzwingt ein Kontingent für die Anzahl der Tabellen pro Cluster nach Knotentyp. Der Tabellenname kann mit dem Datenbank- und Schemanamen qualifiziert werden, wie die folgende Tabelle zeigt.

```
create table tickit.public.test (c1) as select * from oldtable;
```

In diesem Beispiel ist `tickit` der Datenbankname und `public` ist der Schemaname. Wenn die Datenbank oder das Schema nicht vorhanden sind, gibt die Anweisung einen Fehler zurück.

Wenn ein Schemaname vorhanden ist, wird die neue Tabelle in diesem Schema erstellt (vorausgesetzt, der Ersteller kann auf das Schema zugreifen). Der Tabellenname muss für dieses Schema eindeutig sein. Wenn kein Schema angegeben ist, wird die Tabelle anhand des aktuellen Datenbankschemas erstellt. Wenn Sie eine temporäre Tabelle erstellen, können Sie keinen Schemanamen angeben, da sich temporäre Tabellen in einem speziellen Schema befinden.

Mehrere temporäre Tabellen mit demselben Namen können in derselben Datenbank zur gleichen Zeit vorhanden sein, wenn sie in getrennten Sitzungen erstellt wurden. Diese Tabellen werden unterschiedlichen Schemata zugewiesen.

column_name

Der Name einer Spalte in der neuen Tabelle. Wenn keine Spaltennamen angegeben werden, werden die Spaltennamen den Ausgabespaltennamen der Abfrage entnommen. Für Ausdrücke werden Standardspaltennamen verwendet. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

BACKUP { YES | NO }

Eine Klausel, die angibt, ob die Tabelle in automatisierten und manuellen Cluster-Snapshots enthalten sein sollte. Geben Sie für Tabellen wie Staging-Tabellen, die keine kritischen Daten enthalten werden, BACKUP NO an, um beim Erstellen und Wiederherstellen von Snapshots Verarbeitungszeit zu sparen und den Speicherplatz auf Amazon Simple Storage Service zu reduzieren. Die Einstellung BACKUP NO wirkt sich nicht auf die automatische Replikation von Daten zu anderen Knoten innerhalb des Clusters aus. Daher werden Tabellen mit BACKUP NO bei einem Knotenausfall wiederhergestellt. Der Standardwert ist BACKUP YES.

DISTSTYLE { AUTO | EVEN | KEY | ALL }

Definiert den Datenverteilungsstil für die gesamte Tabelle. Amazon Redshift verteilt die Zeilen einer Tabelle an die Datenverarbeitungsknoten gemäß dem für die Tabelle angegebenen Verteilungstyp. Der Standardwert ist DISTSTYLE AUTO.

Der von Ihnen für Tabellen ausgewählte Verteilungsstil wirkt sich auf die allgemeine Leistung Ihrer Datenbank aus. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

- AUTO: Amazon Redshift weist auf der Grundlage der Tabellendaten den optimalen Verteilungsstil aus. Um den Verteilungsstil einer Tabelle anzuzeigen, führen Sie eine Abfrage

für die Systemkatalogtabelle PG_CLASS aus. Weitere Informationen finden Sie unter Anzeigen von Verteilungsstilen.

- EVEN: Die Daten in der Tabelle werden gleichmäßig in einer Round-Robin-Verteilung auf die Knoten in einem Cluster verteilt. Für die Festlegung der Verteilung werden Zeilen-IDs verwendet und jedem Knoten wird ungefähr die gleiche Zahl von Zeilen zugeteilt. Dies ist die Standardverteilungsmethode.
- KEY: Die Daten werden anhand der Werte in der DISTKEY-Spalte verteilt. Wenn Sie verknüpfte Spalten von verknüpften Tabellen als Verteilungsschlüssel festlegen, werden die verknüpften Zeilen aus beiden Tabellen auf den Datenverarbeitungsknoten zusammen platziert. Wenn Daten zusammen platziert werden, kann der Optimierer Verknüpfungen effizienter ausführen. Wenn Sie einen DISTSTYLE KEY angeben, müssen Sie eine DISTKEY-Spalte benennen.
- ALL: Eine Kopie der gesamten Tabelle wird zu jedem Knoten verteilt. Dieser Verteilungsstil stellt sicher, dass alle Zeilen, die für eine Verknüpfung erforderlich sind, auf jedem Knoten vorhanden sind. Die Speicheranforderungen werden jedoch multipliziert und die Lade- und Wartungszeiten für die Tabelle werden verlängert. Der Verteilungsstil ALL kann die Ausführungszeit verbessern, wenn er in Verbindung mit bestimmten Dimensionstabellen verwendet wird, für die der Verteilungsstil KEY nicht geeignet ist. Die Leistungsverbesserungen müssen jedoch gegen die Wartungskosten abgewogen werden.

DISTKEY (column)

Gibt einen Spaltennamen oder eine Positionsnummer für den Verteilungsschlüssel an. Verwenden Sie den Namen, der in der Liste der optionalen Spalten für die Tabelle oder in der Auswahlliste der Abfrage angegeben ist. Alternativ können Sie eine Positionsnummer verwenden, wobei die erste ausgewählte Spalte 1 ist, die zweite ausgewählte Spalte 2 ist usw. In eine Tabelle kann nur eine Spalte der Verteilungsschlüssel sein:

- Wenn Sie eine Spalte als DISTKEY-Spalte deklarieren, muss DISTSTYLE auf KEY festgelegt werden oder darf überhaupt nicht festgelegt werden.
- Wenn Sie keine DISTKEY-Spalte deklarieren, können Sie DISTSTYLE auf EVEN festlegen.
- Wenn Sie weder DISTKEY noch DISTSTYLE angeben, legt CTAS den Verteilungsstil für die neue Tabelle anhand des Abfrageplans für die SELECT-Klausel fest. Weitere Informationen finden Sie unter Vererbung von Spalten- und Tabellenattributen.

Sie können dieselbe Spalte als Verteilungs- und Sortierschlüssel definieren. Dieser Ansatz tendiert dazu, Verknüpfungen zu beschleunigen, wenn die betreffende Spalte eine Verknüpfungsspalte in der Abfrage ist.

[COMPOUND | INTERLEAVED] SORTKEY (column_name [,...])

Gibt einen oder mehrere Sortierschlüssel für die Tabelle an. Wenn Daten in die Tabelle geladen werden, werden die Daten anhand der Spalten sortiert, die als Sortierschlüssel bezeichnet sind.

Sie können optional den COMPOUND- oder INTERLEAVED-Sortierstil angeben. Der Standard ist COMPOUND. Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.

Sie können pro Tabelle maximal 400 COMPOUND SORTKEY-Spalten oder 8 INTERLEAVED SORTKEY-Spalten definieren.

Wenn Sie SORTKEY nicht angeben, legt CTAS die Sortierschlüssel für die neue Tabelle anhand des Abfrageplans für die SELECT-Klausel fest. Weitere Informationen finden Sie unter Vererbung von Spalten- und Tabellenattributen.

COMPOUND

Gibt an, dass die Daten mittels eines zusammengesetzten Schlüssels sortiert werden, der aus allen aufgelisteten Spalten in der Reihenfolge ihrer Auflistung besteht. Ein zusammengesetzter Sortierschlüssel ist am nützlichsten, wenn eine Abfrage Zeilen in der Reihenfolge der Sortierspalten scannt. Die Leistungsvorteile einer Sortierung mit einem zusammengesetzten Schlüssel nehmen ab, wenn Abfragen von sekundären Sortierspalten abhängig sind. Sie können maximal 400 COMPOUND SORTKEY-Spalten pro Tabelle definieren.

INTERLEAVED

Gibt an, dass die Daten mittels eines überlappenden Sortierschlüssels sortiert werden. Für einen überlappenden Sortierschlüssel können maximal acht Spalten angegeben werden.

Eine überlappende Sortierung gewichtet jede Spalte bzw. jeden Subsatz von Spalten im Sortierschlüssel gleich, so dass Abfragen nicht von der Reihenfolge der Spalten im Sortierschlüssel abhängig sind. Wenn eine Abfrage eine oder mehrere sekundäre Sortierspalten verwendet, wird die Abfrageleistung durch die überlappende Sortierung deutlich verbessert. Die überlappende Sortierung führt zu geringfügigen Overhead-Kosten für das Laden von Daten und das Bereinigen von Operationen.

AS query

Alle Abfragen (SELECT-Anweisungen), die von Amazon Redshift unterstützt werden.

Nutzungshinweise für CTAS

Einschränkungen

Amazon Redshift erzwingt ein Kontingent für die Anzahl der Tabellen pro Cluster nach Knotentyp.

Die maximale Anzahl von Zeichen für einen Tabellennamen ist 127.

Die maximale Anzahl der Spalten, die Sie in einer einzelnen Tabelle definieren können, ist 1.600.

Vererbung von Spalten- und Tabellenattributen

CREATE TABLE AS (CTAS)-Tabellen erben keine Einschränkungen, Identitätsspalten, Standardspaltenwerte oder den Primärschlüssel aus der Tabelle, aus der sie erstellt wurden.

Sie können für CTAS-Tabellen keine Spaltenkomprimierungskodierungen angeben. Amazon Redshift weist die Komprimierungskodierung automatisch wie folgt zu:

- Spalten, die als Sortierschlüssel definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als Datentyp BOOLEAN, REAL, DOUBLE PRECISION, GEOMETRY oder GEOGRAPHY definiert sind, wird die RAW-Kompression zugewiesen.
- Spalten, die als SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP oder TIMESTAMPTZ definiert sind, wird die AZ64-Komprimierung zugewiesen.
- Spalten, die als CHAR, VARCHAR oder VARBYTE definiert sind, wird LZO-Komprimierung zugewiesen.

Weitere Informationen erhalten Sie unter Kompressionskodierungen und Datentypen.

Um Spaltenkodierungen explizit zuzuweisen, verwenden Sie CREATE TABLE.

CTAS legt den Verteilungsstil und Sortierschlüssel für die neue Tabelle anhand des Abfrageplans für die SELECT-Klausel fest.

Im Fall komplexer Abfragen wie Abfragen mit Joins, Aggregationen, einer ORDER BY-Klausel oder einer LIMIT-Klausel wählt CTAS den bestmöglichen Verteilungsstil und Sortierschlüssel auf der Basis des Abfrageplans aus.

> **Note**
>
> Um für große Datensätze oder komplexe Abfragen eine optimale Leistung zu erzielen, werden Tests mit typischen Datensätzen empfohlen.

Häufig können Sie vorhersagen, welchen Verteilungsstil und Sortierschlüssel CTAS wählen wird, indem Sie im Abfrageplan ermitteln, welche Spalten der Abfrageoptimierer zum Sortieren und Verteilen von Daten auswählt. Wenn es sich beim obersten Knoten des Abfrageplans um einen einfache sequenziellen Scan aus einer einzelnen Tabelle handelt (XN Seq Scan), verwendet CTAS in der Regel den Verteilungsstil und Sortierschlüssel der Quelltabelle. Wenn es sich bei dem obersten Knoten des Abfrageplans um einen anderen sequentiellen Scan handelt (z. B. XN Limit, XN Sort, XN usw.) HashAggregate, bemüht sich CTAS nach besten Kräften, den optimalen Verteilungsstil und den optimalen Sortierschlüssel auf der Grundlage des Abfrageplans auszuwählen.

Angenommen, Sie erstellen fünf Tabellen mit den folgenden Arten von SELECT-Klauseln:

- Eine einfache SELECT-Anweisung
- Eine Limit-Klausel
- Eine Reihenfolgenklausel unter Verwendung von LISTID
- Eine Reihenfolgenklausel unter Verwendung von QTYSOLD
- Eine SUM-Aggregationsfunktion mit einer Gruppierungsklausel.

In den folgenden Beispielen wird der Abfrageplan für die einzelnen CTAS-Anweisungen gezeigt.

```
explain create table sales1_simple as select listid, dateid, qtysold from sales;
                           QUERY PLAN
-----------------------------------------------------------------
 XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(1 row)


explain create table sales2_limit as select listid, dateid, qtysold from sales limit
 100;
                              QUERY PLAN
----------------------------------------------------------------------
 XN Limit  (cost=0.00..1.00 rows=100 width=8)
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(2 rows)
```

```
explain create table sales3_orderbylistid as select listid, dateid, qtysold from sales
 order by listid;
                               QUERY PLAN
------------------------------------------------------------------------
 XN Sort  (cost=1000000016724.67..1000000017155.81 rows=172456 width=8)
   Sort Key: listid
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(3 rows)


explain create table sales4_orderbyqty as select listid, dateid, qtysold from sales
 order by qtysold;
                               QUERY PLAN
------------------------------------------------------------------------
 XN Sort  (cost=1000000016724.67..1000000017155.81 rows=172456 width=8)
   Sort Key: qtysold
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(3 rows)


explain create table sales5_groupby as select listid, dateid, sum(qtysold) from sales
 group by listid, dateid;
                               QUERY PLAN
------------------------------------------------------------------------
 XN HashAggregate  (cost=3017.98..3226.75 rows=83509 width=8)
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(2 rows)
```

Um den Verteilungs- und Sortierschlüssel für jede Tabelle anzuzeigen, führen Sie eine Abfrage für die Systemkatalogtabelle PG_TABLE_DEF aus wie im Folgenden gezeigt.

```
select * from pg_table_def where tablename like 'sales%';

      tablename       |   column   | distkey | sortkey
----------------------+------------+---------+---------
 sales                | salesid    | f       |       0
 sales                | listid     | t       |       0
 sales                | sellerid   | f       |       0
 sales                | buyerid    | f       |       0
 sales                | eventid    | f       |       0
 sales                | dateid     | f       |       1
```

```
 sales                | qtysold    | f       |       0
 sales                | pricepaid  | f       |       0
 sales                | commission | f       |       0
 sales                | saletime   | f       |       0
 sales1_simple        | listid     | t       |       0
 sales1_simple        | dateid     | f       |       1
 sales1_simple        | qtysold    | f       |       0
 sales2_limit         | listid     | f       |       0
 sales2_limit         | dateid     | f       |       0
 sales2_limit         | qtysold    | f       |       0
 sales3_orderbylistid | listid     | t       |       1
 sales3_orderbylistid | dateid     | f       |       0
 sales3_orderbylistid | qtysold    | f       |       0
 sales4_orderbyqty    | listid     | t       |       0
 sales4_orderbyqty    | dateid     | f       |       0
 sales4_orderbyqty    | qtysold    | f       |       1
 sales5_groupby       | listid     | f       |       0
 sales5_groupby       | dateid     | f       |       0
 sales5_groupby       | sum        | f       |       0
```

In der folgenden Tabelle werden die Ergebnisse zusammengefasst. Aus Gründen der Einfachheit werden Details zu Kosten, Zeilen und zur Breite aus dem Beispielplan ausgelassen.

Tabelle	CTAS-SELECT-Anweisung	Oberster Knoten des Beispielplans	Verteilungsschlüssel	Sortierschlüssel
S1_SIMPLE	`select listid, dateid, qtysold from sales`	`XN Seq Scan on sales ...`	LISTID	DATEID
S2_LIMIT	`select listid, dateid, qtysold from sales limit 100`	`XN Limit ...`	Kein (EVEN)	Keine
S3_ORDER_BY_LISTID	`select listid, dateid, qtysold from sales order by listid`	`XN Sort ...` `Sort Key: listid`	LISTID	LISTID

Tabelle	CTAS-SELECT-Anweisung	Oberster Knoten des Beispielplans	Verteilungsschlüssel	Sortierschlüssel
S4_ORDER_BY_QTY	`select listid, dateid, qtysold from sales order by qtysold`	`XN Sort ...` `Sort Key: qtysold`	LISTID	QTYSOLD
S5_GROUP_BY	`select listid, dateid, sum(qtysold) from sales group by listid, dateid`	`XN HashAggregate ...`	Kein (EVEN)	Keine

Sie können den Verteilungsstil und Sortierschlüssel in der CTAS-Anweisung explizit angeben. Die folgende Anweisung erstellt beispielsweise eine Tabelle mit der Verteilung EVEN und gibt SALESID als Sortierschlüssel an.

```
create table sales_diseteven
diststyle even
sortkey (salesid)
as
select eventid, venueid, dateid, eventname
from event;
```

Kompressionskodierung

ENCODE AUTO wird Standardeinstellung für Tabellen verwendet. Amazon Redshift verwaltet automatisch die Komprimierungskodierung für alle Spalten in der Tabelle.

Verteilung eingehender Daten

Wenn das Hash-Verteilungsschema der eingehenden Daten dem der Zieltabelle entspricht, ist keine physische Verteilung der Daten notwendig, wenn die Daten geladen werden. Wenn beispielsweise ein Verteilungsschlüssel für die neue Tabelle festgelegt wird und die Daten aus einer anderen Tabelle eingefügt werden, die anhand der gleichen Schlüsselspalte verteilt wird, werden die Daten unter Verwendung derselben Knoten und Slices entsprechend geladen. Wenn jedoch sowohl die Quell- als

auch die Zieltabelle auf eine EVEN-Verteilung festgelegt sind, werden die Daten neu zur Zieltabelle verteilt.

Automatische ANALYZE-Operationen

Amazon Redshift analysiert automatisch Tabellen, die Sie mit CTAS-Befehlen erstellen. Sie müssen den Befehl ANALYZE nicht für diese Tabellen ausführen, wenn sie erstellt werden. Wenn Sie sie ändern, sollten Sie sie jedoch genauso wie andere Tabellen analysieren.

CTAS-Beispiele

Im folgenden Befehl wird eine Tabelle namens EVENT_BACKUP für die Tabelle EVENT erstellt:

```
create table event_backup as select * from event;
```

Die erstellte Tabelle erbt die Verteilungs- und Sortierschlüssel aus der Tabelle EVENT.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'event_backup';

column    | type                        | encoding | distkey | sortkey
----------+-----------------------------+----------+---------+--------
catid     | smallint                    | none     | false   |       0
dateid    | smallint                    | none     | false   |       1
eventid   | integer                     | none     | true    |       0
eventname | character varying(200)      | none     | false   |       0
starttime | timestamp without time zone | none     | false   |       0
venueid   | smallint                    | none     | false   |       0
```

Der folgende Befehl erstellt eine neue Tabelle namens EVENTDISTSORT, indem vier Spalten aus der Tabelle EVENT ausgewählt werden. Die neue Tabelle wird nach EVENTID verteilt und nach EVENTID und DATEID sortiert:

```
create table eventdistsort
distkey (1)
sortkey (1,3)
as
select eventid, venueid, dateid, eventname
from event;
```

Das Ergebnis ist wie folgt:

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdistsort';

column   |          type          | encoding | distkey | sortkey
---------+------------------------+----------+---------+-------
eventid   | integer               | none     | t       | 1
venueid   | smallint              | none     | f       | 0
dateid    | smallint              | none     | f       | 2
eventname | character varying(200)| none     | f       | 0
```

Sie könnten genau die gleiche Tabelle erstellen, indem Sie für die Verteilungs- und Sortierschlüssel Spaltennamen verwenden. Beispiel:

```
create table eventdistsort1
distkey (eventid)
sortkey (eventid, dateid)
as
select eventid, venueid, dateid, eventname
from event;
```

Die folgende Anweisung wendet eine gleichmäßige Verteilung auf die Tabelle an, definiert jedoch keinen expliziten Sortierschlüssel:

```
create table eventdisteven
diststyle even
as
select eventid, venueid, dateid, eventname
from event;
```

Die Tabelle übernimmt den Sortierschlüssel nicht aus der Tabelle EVENT (EVENTID), da für die neue Tabelle die Verteilung EVEN angegeben ist. Die neue Tabelle besitzt weder einen Sortierschlüssel noch einen Verteilungsschlüssel.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdisteven';

column    |          type          | encoding | distkey | sortkey
----------+------------------------+----------+---------+---------
eventid   | integer                | none     | f       | 0
venueid   | smallint               | none     | f       | 0
```

```
dateid    | smallint               | none     | f       | 0
eventname | character varying(200) | none     | f       | 0
```

Die folgende Anweisung wendet eine gleichmäßige Verteilung an und definiert einen Sortierschlüssel:

```
create table eventdistevensort diststyle even sortkey (venueid)
as select eventid, venueid, dateid, eventname from event;
```

Die erstellte Tabelle besitzt einen Sortierschlüssel, jedoch keinen Verteilungsschlüssel.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdistevensort';

column    |          type          | encoding | distkey | sortkey
----------+------------------------+----------+---------+-------
eventid   | integer                | none     | f       | 0
venueid   | smallint               | none     | f       | 1
dateid    | smallint               | none     | f       | 0
eventname | character varying(200) | none     | f       | 0
```

Die folgende Anweisung verteilt die Tabelle EVENT anhand einer anderen Schlüsselspalte aus den eingehenden Daten neu, die anhand der Spalte EVENTID sortiert sind, und definiert keine SORTKEY-Spalte. Daher ist die Tabelle nicht sortiert.

```
create table venuedistevent distkey(venueid)
as select * from event;
```

Das Ergebnis ist wie folgt:

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'venuedistevent';

 column   |            type             | encoding | distkey | sortkey
----------+-----------------------------+----------+---------+-------
eventid   | integer                     | none     | f       | 0
venueid   | smallint                    | none     | t       | 0
catid     | smallint                    | none     | f       | 0
dateid    | smallint                    | none     | f       | 0
eventname | character varying(200)      | none     | f       | 0
starttime | timestamp without time zone | none     | f       | 0
```

CREATE USER

Erstellt einen neuen Datenbankbenutzer. Je nach Berechtigungen und Rollen können Datenbankbenutzer Daten abrufen, Befehle ausführen und andere Aktionen in einer Datenbank ausführen. Sie müssen Datenbank-Superuser sein, um diesen Befehl auszuführen.

Erforderliche Berechtigungen

Für CREATE USER sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung CREATE USER

Syntax

```
CREATE USER name [ WITH ]
PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }
[ option [ ... ] ]

where option can be:

CREATEDB | NOCREATEDB
| CREATEUSER | NOCREATEUSER
| SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }
| IN GROUP groupname [, ... ]
| VALID UNTIL 'abstime'
| CONNECTION LIMIT { limit | UNLIMITED }
| SESSION TIMEOUT limit
| EXTERNALID external_id
```

Parameter

Name

Der Name des zu erstellenden Benutzers. Der Benutzername darf nicht sein `PUBLIC`. Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen.

WITH

Optionales Schlüsselwort. WITH wird von Amazon Redshift ignoriert

PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }

Legt das Benutzerpasswort fest.

Standardmäßig können Benutzer ihre eigenen Passwörter ändern, es sei denn, das Passwort ist deaktiviert. Legen Sie zum Deaktivieren des Passworts eines Benutzers DISABLE fest. Wenn das Passwort eines Benutzers deaktiviert ist, wird das Passwort aus dem System gelöscht und der Benutzer kann sich nur mit temporären Benutzeranmeldeinformationen AWS Identity and Access Management (IAM) anmelden. Weitere Informationen finden Sie unter Verwenden der IAM-Authentifizierung zum Generieren von Benutzeranmeldeinformationen für Datenbanken. Nur ein Superuser kann Passwörter aktivieren oder deaktivieren. Sie können das Passwort eines Superusers nicht deaktivieren. Führen Sie zum Aktivieren eines Passworts ALTER USER aus und legen Sie ein Passwort fest.

Sie können das Passwort als reinen Text, als MD5-Hash-Zeichenfolge oder als SHA256-Hash-Zeichenfolge angeben.

> **Note**
>
> Wenn Sie einen neuen Cluster mit der AWS Management Console AWS CLI, oder Amazon Redshift Redshift-API starten, müssen Sie ein Klartext-Passwort für den ersten Datenbankbenutzer angeben. Sie können das Passwort später mittels ändern ALTER USER.

Wenn Sie es als Klartext angeben, muss das Passwort den folgenden Einschränkungen entsprechen:

- Es muss 8 bis 64 Zeichen lang sein.
- Es muss mindestens einen Großbuchstaben, einen Kleinbuchstaben und eine Zahl enthalten.
- Es kann alle ASCII-Zeichen mit den ASCII-Codes 33–126 enthalten, außer einfachen Anführungszeichen ('), doppelten Anführungszeichen ("), \, / oder @.

Sicherer als die Übergabe des Passwortparameters CREATE USER als Klartext ist die Angabe einer MD5-Hash-Zeichenfolge, die das Passwort und den Benutzernamen enthält.

> **Note**
>
> Wenn Sie eine MD5-Hash-Zeichenfolge angeben, prüft der Befehl CREATE USER, ob eine gültige MD5-Hash-Zeichenfolge vorhanden ist, validiert den Passwortteil der

Zeichenfolge jedoch nicht. In diesem Fall ist es möglich, ein Passwort wie beispielsweise eine leere Zeichenfolge zu erstellen, das nicht für die Anmeldung an der Datenbank verwendet werden kann.

So geben Sie ein MD5-Passwort an:

1. Verketten Sie Passwort und Benutzername.

   Wenn das Passwort beispielsweise `ez` und der Benutzername `user1` ist, ist die verkettete Zeichenfolge `ezuser1`.

2. Wandeln Sie die verkettete Zeichenfolge in eine MD5-Hash-Zeichenfolge mit 32 Zeichen um. Sie können für die Erstellung der Hash-Zeichenfolge jedes MD5-Dienstprogramm verwenden. Im folgenden Beispiel werden Amazon Redshift Die Funktion MD5 und der Verkettungsoperator (||) verwendet, um eine MD5-Hash-Zeichenfolge mit 32 Zeichen zurückzugeben.

   ```
   select md5('ez' || 'user1');

   md5
   ---------------------------------
   153c434b4b77c89e6b94f12c5393af5b
   ```

3. Verketten Sie `md5` vor der MD5-Hash-Zeichenfolge, und geben Sie die verkettete Zeichenfolge als das Argument md5hash an.

   ```
   create user user1 password 'md5153c434b4b77c89e6b94f12c5393af5b';
   ```

4. Melden Sie sich mit den Anmeldeinformationen bei der Datenbank an.

   In diesem Beispiel melden Sie sich als `user1` mit dem Passwort `ez` an.

Eine andere sichere Alternative ist es, einen SHA-256-Hash eines Passwortstrings anzugeben. Sie können auch einen eigenen gültigen SHA-256-Digest und ein 256-Bit-Salt, das zur Erstellung des Digest verwendet wurde, bereitstellen.

- Digest – Die Ausgabe einer Hashing-Funktion.
- Salt – Zufällig generierte Daten, die mit dem Passwort kombiniert werden, um Muster in der Hashing-Funktion-Ausgabe zu reduzieren.

```
'sha256|Mypassword'
```

```
'sha256|digest|256-bit-salt'
```

Im folgenden Beispiel generiert und verwaltet Amazon Redshift das Salt.

```
CREATE USER admin PASSWORD 'sha256|Mypassword1';
```

Im folgenden Beispiel werden ein gültiger SHA-256-Digest und ein 256-Bit-Salt, das zur Erstellung des Digest verwendet wurde, bereitgestellt.

Gehen Sie wie folgt vor, um ein Passwort anzugeben und es mit Ihrem eigenen Salt zu hashen:

1. Erstellen Sie ein 256-Bit-Salt. Sie können ein Salt erhalten, indem Sie einen beliebigen hexadezimalen Zeichenfolgengenerator verwenden, um eine 64 Zeichen lange Zeichenfolge zu generieren. In diesem Beispiel lautet das Salt `c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6`.
2. Verwenden Sie die Funktion FROM_HEX, um Ihr Salt in eine Binärdatei umzuwandeln. Dies liegt daran, dass die SHA2-Funktion die binäre Darstellung des Salts erfordert. Sehen Sie sich die folgende Anweisung an.

   ```
   SELECT
    FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6');
   ```
3. Verwenden Sie die CONCAT-Funktion, um Ihr Salt an Ihr Passwort anzuhängen. In diesem Beispiel lautet das Passwort `Mypassword1`. Sehen Sie sich die folgende Anweisung an.

   ```
   SELECT
    CONCAT('Mypassword1',FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f951303
   ```
4. Verwenden Sie die SHA2-Funktion, um aus Ihrer Kombination aus Passwort und Salt einen Digest zu erstellen. Sehen Sie sich die folgende Anweisung an.

   ```
   SELECT
    SHA2(CONCAT('Mypassword1',FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f95
    0);
   ```
5. Erstellen Sie den Benutzer mit dem Digest und dem Salt aus den vorherigen Schritten. Sehen Sie sich die folgende Anweisung an.

```
CREATE USER admin PASSWORD 'sha256|
821708135fcc42eb3afda85286dee0ed15c2c461d000291609f77eb113073ec2|
c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6';
```

6. Melden Sie sich mit den Anmeldeinformationen bei der Datenbank an.

   In diesem Beispiel melden Sie sich als `admin` mit dem Passwort `Mypassword1` an.

Wenn Sie ein Nur-Text-Passwort festlegen, ohne eine Hashing-Funktion anzugeben, wird ein MD5-Digest generiert, der den Nutzernamen als Salt verwendet.

CREATEDB | NOCREATEDB

Mithilfe der Option CREATEDB kann der neue Benutzer Datenbanken erstellen. Der Standardwert ist NOCREATEDB.

CREATEUSER | NOCREATEUSER

Die Option CREATEUSER erstellt einen Superuser mit allen Datenbankrechten einschließlich CREATE USER. Der Standardwert ist NOCREATEUSER. Weitere Informationen finden Sie unter superuser.

SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }

Eine Klausel, über die die Zugriffsebene eines Benutzers auf die Amazon-Redshift-Systemtabellen und -Ansichten festgelegt wird.

Reguläre Benutzer mit der SYSLOG ACCESS RESTRICTED-Berechtigung können nur die Zeilen sehen, die von diesem Benutzer in für den Benutzer sichtbaren Systemtabellen und -ansichten generiert wurden. RESTRICTED ist der Standardwert.

Normale Benutzer mit der Berechtigung SYSLOG ACCESS UNRESTRICTED können alle Zeilen in für den Benutzer sichtbaren Systemtabellen und -ansichten sehen, einschließlich der Zeilen, die von einem anderen Benutzer generiert wurden. Über die Option UNRESTRICTED erhält ein Benutzer keinen Zugriff auf für Superuser sichtbare Tabellen. Nur Superuser können auf solche Tabellen zugreifen.

> **Note**
>
> Wenn Sie einem Benutzer uneingeschränkten Zugriff auf Systemtabellen gewähren, sieht der Benutzer auch Daten, die von anderen Benutzern generiert wurden. STL_QUERY und STL_QUERYTEXT enthalten beispielsweise den vollständigen Text von INSERT-,

UPDATE- und DELETE-Anweisungen, die möglicherweise sensible von Benutzern generierte Daten enthalten.

Alle Zeilen in SVV_TRANSACTIONS sind für alle Benutzer sichtbar.

Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

IN GROUP groupname

Gibt den Namen einer vorhandenen Gruppe an, zu der der Benutzer gehört. Es können mehrere Gruppennamen aufgelistet werden.

VALID UNTIL abstime

Die Option VALID UNTIL legt eine absolute Zeit fest, nach der das Passwort des Benutzers nicht länger gültig ist. Standardmäßig gibt es für das Passwort kein Zeitlimit.

CONNECTION LIMIT { Limit | UNLIMITED }

Die maximale Zahl von Datenbankverbindungen, die der Benutzer gleichzeitig geöffnet haben darf. Das Limit wird für Superuser nicht durchgesetzt. Mithilfe des Schlüsselworts UNLIMITED können Sie die maximale Zahl gleichzeitiger Verbindungen festlegen. Möglicherweise gilt auch ein Limit für die Zahl der Verbindungen für die einzelnen Datenbanken. Weitere Informationen finden Sie unter CREATE DATABASE. Der Standardwert ist UNLIMITED. Um die aktuellen Verbindungen anzuzeigen, führen Sie eine Abfrage für die Systemansicht STV_SESSIONS aus.

> **Note**
>
> Wenn sowohl für Benutzer- als auch für Datenbankverbindungen Limits gelten, muss ein ungenutzter Verbindungsplatz verfügbar sein, der sich innerhalb beider Grenzen befindet, wenn ein Benutzer versucht, eine Verbindung herzustellen.

SESSION TIMEOUT limit

Die maximale Zeit in Sekunden, die eine Sitzung inaktiv oder untätig bleibt. Der Bereich liegt zwischen 60 Sekunden (einer Minute) und 1.728.000 Sekunden (20 Tagen). Wenn für den Benutzer kein Sitzungstimeout eingestellt ist, gilt die Clustereinstellung. Weitere Informationen finden Sie unter Kontingente und Limits in Amazon Redshift im Verwaltungshandbuch zu Amazon Redshift.

Wenn Sie das Sitzungstimeout festlegen, wird es nur auf neue Sitzungen angewendet.

Um Informationen über aktive Benutzersitzungen, einschließlich der Startzeit, des Benutzernamens und des Sitzungstimeouts anzuzeigen, fragen Sie die STV_SESSIONS-Systemansicht ab. Um Informationen über den Verlauf von Benutzersitzungen anzuzeigen, fragen Sie die STL_SESSIONS-Ansicht an. Um Informationen über Datenbankbenutzer, einschließlich Sitzungstimeouts, abzurufen, fragen Sie die SVL_USER_INFO-Ansicht ab.

EXTERNALID external_id

Der Bezeichner für den Benutzer, der einem Identitätsanbieter zugeordnet ist. Der Benutzer muss sein Passwort deaktiviert haben. Weitere Informationen finden Sie unter Nativer Identitätsanbieter(IdP)-Verbund für Amazon Redshift.

Nutzungshinweise

Standardmäßig besitzen alle Benutzer CREATE- und USAGE-Rechte für das Schema PUBLIC. Um Benutzer daran zu hindern, Objekte im Schema PUBLIC einer Datenbank zu erstellen, verwenden Sie den Befehl REVOKE, um dieses Recht zu entfernen.

Wenn Sie die IAM-Authentifizierung verwenden, um Benutzeranmeldeinformationen für Datenbanken zu erstellen, können Sie einen Superuser erstellen, der sich nur anhand vorübergehender Anmeldeinformationen anmelden kann. Sie können das Passwort eines Superusers nicht deaktivieren, aber ein unbekanntes Passwort mithilfe einer willkürlich generierten MD5-Hash-Zeichenfolge erstellen.

```
create user iam_superuser password 'md5A1234567890123456780123456789012' createuser;
```

Die Groß-/Kleinschreibung eines in doppelte Anführungszeichen eingeschlossenen Benutzernamens bleibt unabhängig von der Einstellung der `enable_case_sensitive_identifier`-Konfigurationsoption immer erhalten. Weitere Informationen finden Sie unter enable_case_sensitive_identifier.

Beispiele

Der folgende Befehl erstellt einen Benutzer namens dbuser mit dem Passwort "abcD1234", Datenbankerstellungsrechten und einem Verbindungslimit von 30.

```
create user dbuser with password 'abcD1234' createdb connection limit 30;
```

Führen Sie eine Abfrage für die Katalogtabelle PG_USER_INFO aus, um Details zu einem Datenbankbenutzer anzuzeigen.

```
select * from pg_user_info;

 usename   | usesysid | usecreatedb | usesuper | usecatupd | passwd   | valuntil |
 useconfig | useconnlimit
-----------+----------+-------------+----------+-----------+----------+----------
+-----------+-------------
 rdsdb     |        1 | true        | true     | true      | ******** | infinity |
      |
 adminuser |      100 | true        | true     | false     | ******** |          |
      | UNLIMITED
 dbuser    |      102 | true        | false    | false     | ******** |          |
      | 30
```

Im folgenden Beispiel ist das Kontopasswort bis zum 10 Juni 2017 gültig.

```
create user dbuser with password 'abcD1234' valid until '2017-06-10';
```

Im folgenden Beispiel wird ein Benutzer mit einen Kennwort erstellt, das zwischen Groß- und Kleinschreibung unterscheidet und Sonderzeichen enthält.

```
create user newman with password '@AbC4321!';
```

Um in Ihrem MD5-Passwort einen Backslash („\“) zu verwenden, müssen Sie für den Backslash ein Escape-Zeichen in Form eines Backslash in Ihrer Quellzeichenfolge verwenden. Im folgenden Beispiel wird ein Benutzer namens `slashpass` mit einem einzelnen Backslash („\“) als Passwort erstellt.

```
select md5('\\'||'slashpass');

md5
--------------------------------
0c983d1a624280812631c5389e60d48c
```

Erstellen Sie einen Benutzer mit dem Passwort md5.

```
create user slashpass password 'md50c983d1a624280812631c5389e60d48c';
```

Im folgenden Beispiel wird ein Benutzer mit dem Namen dbuser erstellt, für den das Timeout für eine Leerlaufsitzung auf 120 Sekunden eingestellt ist.

```
CREATE USER dbuser password 'abcD1234' SESSION TIMEOUT 120;
```

Im folgenden Beispiel wird ein Benutzer mit dem Namen bob erstellt. Der Namespace lautet myco_aad. Dies ist nur ein Beispiel. Um den Befehl erfolgreich ausführen zu können, benötigen Sie einen registrierten Identitätsanbieter. Weitere Informationen finden Sie unter Nativer Identitätsanbieter(IdP)-Verbund für Amazon Redshift.

```
CREATE USER myco_aad:bob EXTERNALID "ABC123" PASSWORD DISABLE;
```

CREATE VIEW

Erstellt eine Ansicht in einer Datenbank. Die Ansicht wird nicht physisch umgesetzt; die Abfrage, mit der die Ansicht definiert wird, wird jedes Mal ausgeführt, wenn die Ansicht in einer Abfrage referenziert wird. Zum Erstellen einer Ansicht mit einer externen Tabelle fügen Sie die Klausel WITH NO SCHEMA BINDING ein.

Um eine Standardansicht erstellen zu können, benötigen Sie Zugriff auf die zugrunde liegenden Tabellen oder Ansichten. Zum Abfragen einer Standardansicht benötigen Sie ausgewählte Berechtigungen für die Ansicht selbst, jedoch nicht für die zugrundeliegenden Tabellen. Wenn Sie eine Ansicht erstellen, die auf eine Tabelle oder Ansicht in einem anderen Schema oder auf eine materialisierte Ansicht verweist, benötigen Sie Nutzungsberechtigungen. Zum Abfragen einer Late-Binding-Ansicht benötigen Sie SELECT-Berechtigungen für die Late-Binding-Ansicht selbst. Sie sollten sicherstellen, dass der Eigentümer der Ansicht mit später Bindung über Auswahlsonderrechte auf die verwiesenen Objekte (Tabellen, Ansichten oder benutzerdefinierte Funktionen) verfügt. Weitere Informationen zu Late-Binding-Ansichten finden Sie unter Nutzungshinweise.

Erforderliche Berechtigungen

Für CREATE VIEW sind folgende Berechtigungen erforderlich:

- Für CREATE VIEW:
  - Superuser
  - Benutzer mit der Berechtigung CREATE [OR REPLACE] VIEW
- Für REPLACE VIEW:
  - Superuser

- Benutzer mit der Berechtigung CREATE [OR REPLACE] VIEW
- Besitzer der Ansicht

Syntax

```
CREATE [ OR REPLACE ] VIEW name [ ( column_name [, ...] ) ] AS query
[ WITH NO SCHEMA BINDING ]
```

Parameter

OR REPLACE

Wenn bereits eine Ansicht mit demselben Namen vorhanden ist, wird die Ansicht ersetzt. Sie können eine Ansicht nur durch eine neue Abfrage ersetzen, die einen identischen Satz von Spalten generiert und dieselben Spaltennamen und Datentypen verwendet. CREATE OR REPLACE VIEW sperrt die Ansicht für Lese- und Schreibvorgänge, bis die Operation abgeschlossen ist.

Wenn eine Ansicht ersetzt wird, bleiben ihre anderen Eigenschaften, wie z. B. Eigentümerschaft und gewährte Berechtigungen, erhalten.

Name

Der Name der Ansicht. Wenn ein Schemaname angegeben ist (wie `myschema.myview`), wird die Ansicht mithilfe des angegebenen Schemas erstellt. Andernfalls wird die Ansicht im aktuellen Schema erstellt. Der Name der Ansicht muss sich von den Namen anderer Ansichten oder Tabellen im selben Schema unterscheiden.

Wenn Sie einen Ansichtnamen angeben, der mit „#“ beginnt, wird die Ansicht als temporäre Ansicht erstellt, die nur in der aktuellen Sitzung angezeigt wird.

Weitere Informationen zu gültigen Namen finden Sie unter Namen und Kennungen. Sie können in den Systemdatenbanken template0, template1, padb_harvest oder sys:internal keine Tabellen oder Ansichten erstellen.

column_name

Optionale Liste von Namen, die für die Spalten in der Ansicht verwendet werden sollen. Wenn keine Spaltennamen angegeben werden, werden die Spaltennamen aus der Abfrage abgeleitet. Die maximale Anzahl der Spalten, die Sie in einer einzelnen Ansicht definieren können, ist 1.600.

query

Eine Abfrage (in Form einer SELECT-Anweisung), die zu einer Tabelle evaluiert. Diese Tabelle definiert die Spalten und Zeilen in der Ansicht.

WITH NO SCHEMA BINDING

Klausel, die angibt, dass die Ansicht nicht an die zugrunde liegenden Datenbankobjekte gebunden ist, wie etwa Tabellen und benutzerdefinierte Funktionen. Deshalb besteht keine Abhängigkeit zwischen der Ansicht und den Objekten, auf die sie verweist. Sie können sogar eine Ansicht erstellen, wenn die referenzierten Objekte nicht vorhanden sind. Da keine Abhängigkeit besteht, können Sie ein referenziertes Objekt entfernen oder ändern, ohne die Ansicht zu beeinflussen. Amazon Redshift sucht erst nach Abhängigkeiten, wenn die Ansicht abgefragt wird. Um Details über Late-Binding-Ansichten anzuzeigen, führen Sie die Funktion PG_GET_LATE_BINDING_VIEW_COLS aus.

Wenn Sie die WITH NO SCHEMA BINDING-Klausel einbeziehen, müssen Tabellen und Ansichten, die in der SELECT-Anweisung referenziert werden, mit einem Schemanamen qualifiziert werden. Das Schema muss vorhanden sein, wenn die Ansicht erstellt wird, auch wenn die referenzierte Tabelle nicht vorhanden ist. Die folgende Anweisung gibt beispielsweise einen Fehler zurück.

```
create view myevent as select eventname from event
with no schema binding;
```

Die folgende Anweisung wird erfolgreich ausgeführt.

```
create view myevent as select eventname from public.event
with no schema binding;
```

> **Note**
>
> Sie können für eine Ansicht keine Aktualisierungen, Einfügungen oder Löschungen ausführen.

Nutzungshinweise

Ansichten mit später Bindung

Eine Ansicht mit später Bindung prüft zugrunde liegende Datenbankobjekte wie Tabellen und andere Ansichten erst, wenn die Ansicht abgefragt wird. Daher können Sie die zugrunde liegenden Objekte ändern oder weglassen, ohne die Ansicht wegzulassen oder neu zu erstellen. Wenn Sie zugrundeliegende Objekte weglassen, schlagen Anfragen an die Ansicht mit später Bindung fehl. Wenn die Abfrage der Ansicht mit später Bindung auf Spalten in dem zugrundeliegenden Objekt verweist, die nicht vorhanden sind, schlägt die Abfrage fehl.

Wenn Sie die zugrundeliegende Tabelle oder Ansicht einer Ansicht mit später Bildung weglassen und danach erneut erstellen, wird das neue Objekt mit Standard-Zugriffsberechtigungen erstellt. Sie müssen möglicherweise Benutzern, die die Ansicht abfragen werden, Berechtigungen für die zugrundeliegenden Objekte erteilen.

Zum Erstellen einer Ansicht mit später Bindung fügen Sie die Klausel WITH NO SCHEMA BINDING ein. Im folgenden Beispiel wird eine Ansicht ohne Schemabindung erstellt.

```
create view event_vw as select * from public.event
with no schema binding;
```

```
select * from event_vw limit 1;

eventid | venueid | catid | dateid | eventname     | starttime
--------+---------+-------+--------+---------------+--------------------
      2 |     306 |     8 |   2114 | Boris Godunov | 2008-10-15 20:00:00
```

Das folgende Beispiel zeigt, dass Sie eine zugrunde liegende Tabelle ändern können, ohne die Ansicht neu zu erstellen.

```
alter table event rename column eventname to title;
```

```
select * from event_vw limit 1;

eventid | venueid | catid | dateid | title         | starttime
--------+---------+-------+--------+---------------+--------------------
      2 |     306 |     8 |   2114 | Boris Godunov | 2008-10-15 20:00:00
```

Sie können externe Amazon-Redshift-Spectrum-Tabellen nur in Ansichten mit später Bindung referenzieren. Eine Anwendung von Ansichten mit später Bindung dient zum Abfragen der Amazon-

Redshift- und Redshift-Spectrum-Tabellen. Beispiel: Sie können den Befehl UNLOAD verwenden, um ältere Daten in Amazon S3 zu archivieren. Erstellen Sie anschließend eine externe Redshift-Spectrum-Tabelle, die die Daten in Amazon S3 referenziert, und erstellen Sie eine Ansicht, die beide Tabellen abfragt. Das folgende Beispiel verwendet eine UNION ALL-Klausel zur Verbindung der SALES-Tabelle aus Amazon Redshift und der SPECTRUM.SALES-Tabelle aus Redshift Spectrum.

```
create view sales_vw as
select * from public.sales
union all
select * from spectrum.sales
with no schema binding;
```

Weitere Informationen zum Erstellen externer Redshift Spectrum-Tabellen, einschließlich der SPECTRUM.SALES-Tabelle finden Sie unter Erste Schritte mit Amazon Redshift Spectrum.

Wenn Sie eine Standardansicht aus einer Late-Binding-Ansicht erstellen, enthält die Definition der Standardansicht die Definition der Late-Binding-Ansicht zu dem Zeitpunkt, an dem die Standardansicht erstellt wurde. Die Abhängigkeit der Late-Binding-Ansicht wird nicht nachverfolgt, so dass Änderungen an der Late-Binding-Ansicht in der Standardansicht nicht nachverfolgt werden.

Um die Standardansicht so zu aktualisieren, dass sie auf die neueste Definition der Late-Binding-Ansicht verweist, führen Sie CREATE OR REPLACE VIEW mit der ursprünglichen Ansichtsdefinition aus, mit der Sie die Standardansicht erstellt haben.

Im folgenden Beispiel sehen Sie das Erstellen einer Standardansicht aus einer Late-Binding-Ansicht.

```
create view sales_vw_lbv as
select * from public.sales
with no schema binding;

show view sales_vw_lbv;
                            Show View DDL statement
-------------------------------------------------------------------------------
 create view sales_vw_lbv as select * from public.sales with no schema binding;
(1 row)

create view sales_vw as
select * from sales_vw_lbv;

show view sales_vw;
                                                Show View DDL statement
```

```
----------------------------------------------------------------------------------------------
 SELECT sales_vw_lbv.price, sales_vw_lbv."region" FROM (SELECT sales.price,
 sales."region" FROM sales) sales_vw_lbv;
(1 row)
```

Beachten Sie, dass die Late-Binding-Ansicht, wie sie in der DDL-Anweisung für die Standardansicht gezeigt wird, beim Erstellen der Standardansicht definiert und nicht aktualisiert wird, wenn Sie danach Änderungen an der Late-Binding-Ansicht vornehmen.

Beispiele

In den Beispielbefehlen wird ein Beispielsatz von Objekten und Daten verwendet, die sogenannte TICKIT-Datenbank. Weitere Informationen finden Sie unter Beispieldatenbank.

Der folgende Befehl erstellt eine Ansicht namens myevent aus einer Tabelle namens EVENT.

```
create view myevent as select eventname from event
where eventname = 'LeAnn Rimes';
```

Der folgende Befehl erstellt eine Ansicht namens myuser aus einer Tabelle namens USERS.

```
create view myuser as select lastname from users;
```

Der folgende Befehl erstellt oder ersetzt eine Ansicht namens myuser aus einer Tabelle namens USERS.

```
create or replace view myuser as select lastname from users;
```

Im folgenden Beispiel wird eine Ansicht ohne Schemabindung erstellt.

```
create view myevent as select eventname from public.event
with no schema binding;
```

DEALLOCATE

Entfernt die Zuteilung einer vorbereiteten Anweisung.

Syntax

```
DEALLOCATE [PREPARE] plan_name
```

Parameter

PREPARE

Dieses Schlüsselwort ist optional und wird ignoriert.

plan_name

Der Name der vorbereiteten Anweisung, deren Zuteilung aufgehoben werden soll.

Nutzungshinweise

DEALLOCATE wird verwendet, um die Zuteilung einer zuvor vorbereiteten SQL-Anweisung aufzuheben. Wenn Sie die Zuteilung einer vorbereiteten Anweisung nicht explizit aufheben, wird die Zuteilung aufgehoben, wenn die Sitzung beendet wird. Weitere Informationen zu vorbereiteten Anweisungen finden Sie unter PREPARE.

Weitere Informationen finden Sie unter:

EXECUTE, PREPARE

DECLARE

Definiert einen neuen Cursor. Mit einem Cursor können Sie aus dem Ergebnissatz einer größeren Abfrage einige Zeilen gleichzeitig abrufen.

Wenn die erste Zeile eines Cursors abgerufen wird, wird der gesamte Ergebnissatz auf dem Führungsknoten im Arbeitsspeicher oder auf der Festplatte umgesetzt, wenn nötig. Aufgrund der potenziellen negativen Auswirkungen der Verwendung von Cursors mit großen Ergebnissätzen wird empfohlen, alternative Ansätze anzuwenden, wann immer möglich. Weitere Informationen finden Sie unter Überlegungen in Bezug auf die Leistung bei Verwendung von Cursors.

Sie müssen einen Cursor innerhalb eines Transaktionsblocks deklarieren. Pro Sitzung kann jeweils nur ein Cursor offen sein.

Weitere Informationen finden Sie unter FETCH, CLOSE.

Syntax

```
DECLARE cursor_name CURSOR FOR query
```

Parameter

cursor_name

Der Name des neuen Cursors.

query

Eine SELECT-Anweisung, die den Cursor ausfüllt.

Nutzungshinweise für DECLARE CURSOR

Wenn Ihre Clientanwendung eine ODBC-Verbindung verwendet und Ihre Abfrage eine Ergebnismenge erzeugt, die so groß ist, dass sie nicht in den Arbeitsspeicher passt, können Sie die Ergebnismenge unter Verwendung eines Cursors an die Clientanwendung übermitteln. Wenn Sie einen Cursor verwenden, wird der gesamte Ergebnissatz auf dem Führungsknoten umgesetzt. Anschließend kann Ihr Client die Ergebnisse inkrementell abrufen.

> **Note**
>
> Um Cursors in ODBC für Microsoft Windows zu unterstützen, aktivieren Sie die Option Use Declare/Fetch (Declare/Fetch verwenden) in der ODBC DSN, die Sie für Amazon Redshift verwenden. Es wird empfohlen, die Größe des ODBC-Zwischenspeichers im Feld Cache Size (Cache-Größe) des Dialogfelds für ODBC DSN-Optionen auf mindestens 4.000 für Mehrknoten-Cluster festzulegen, um Netzläufe zu minimieren. Legen Sie die Größe des Zwischenspeichers für Einzelknoten-Cluster auf 1.000 fest.

Aufgrund der potenziellen negativen Auswirkungen der Verwendung von Cursors wird empfohlen, alternative Ansätze anzuwenden, wann immer möglich. Weitere Informationen finden Sie unter Überlegungen in Bezug auf die Leistung bei Verwendung von Cursors.

Amazon-Redshift-Cursors werden mit den folgenden Einschränkungen unterstützt:

- Pro Sitzung kann jeweils nur ein Cursor offen sein.
- Cursors müssen innerhalb einer Transaktion (BEGIN … END) verwendet werden.
- Die maximal zulässige kumulative Größe für Ergebnissätze für alle Cursors wird auf der Basis des Typs des Clusterknotens eingeschränkt. Wenn Sie größere Ergebnissätze benötigen, können Sie die Größe in eine XL- oder 8XL-Knotenkonfiguration ändern.

Weitere Informationen finden Sie unter Einschränkungen für Cursors.

Einschränkungen für Cursors

Wenn die erste Zeile eines Cursors abgerufen wird, wird der gesamte Ergebnissatz auf dem Führungsknoten umgesetzt. Wenn der Ergebnissatz zu groß für den Arbeitsspeicher ist, wird er bei Bedarf auf die Festplatte geschrieben. Um die Integrität des Führungsknotens zu schützen, setzt Amazon Redshift Einschränkungen für die Größe aller Cursor-Ergebnissätze auf der Basis des Typs des Clusterknotens durch.

In der folgenden Tabelle wird die maximal zulässige Gesamtgröße von Ergebnissätzen für die einzelnen Cluster-Knotentypen gezeigt. Die maximale Größe von Ergebnissätzen ist in Megabyte angegeben.

Knotentyp	Maximal zulässige Größe des Ergebnissatzes pro Cluster (MB)
RA3 16XL-Mehrfachknoten	14400000
DC2 Large-Einzelknoten	8000
DC2 Large-Mehrfachknoten	192000
DC2 8XL-Mehrfachknoten	3200000
RA3 4XL-Mehrfachknoten	3200000
RA3 XLPLUS-Mehrfachknoten	1000000
RA3 XLPLUS-Einzelknoten	64000
Amazon Redshift Serverless	150000

Um die aktive Cursor-Konfiguration für ein Cluster anzuzeigen, führen Sie eine Abfrage für die Systemtabelle STV_CURSOR_CONFIGURATION als Superuser aus. Um den Status aktiver Cursors anzuzeigen, führen Sie eine Abfrage für die Systemtabelle STV_ACTIVE_CURSORS aus. Nur die Zeilen für die eigenen Cursors eines Benutzers werden diesem angezeigt. Superuser können jedoch alle Cursors anzeigen.

Überlegungen in Bezug auf die Leistung bei Verwendung von Cursors

Da Cursors den gesamten Ergebnissatz auf dem Führungsknoten umsetzen, bevor Ergebnisse an den Client zurückgegeben werden, kann sich die Verwendung von Cursors bei sehr großen Ergebnissätzen negativ auf die Leistung auswirken. Es wird nachdrücklich empfohlen, Cursors nicht bei sehr großen Ergebnissätzen zu verwenden. In einigen Fällen sind Cursors möglicherweise die einzige praktikable Lösung, beispielsweise, wenn Ihre Anwendung eine ODBC-Verbindung verwendet. Es wird empfohlen, die folgenden Alternativen zu verwenden, wenn möglich:

- Verwenden Sie UNLOAD, um große Tabellen zu exportieren. Wenn Sie UNLOAD verwenden, arbeiten die Datenverarbeitungsknoten parallel, um die Daten direkt an Datendateien in Amazon Simple Storage Service zu übertragen. Weitere Informationen finden Sie unter Entfernen von Daten.
- Legen Sie den JDBC-Parameter für die Abrufgröße in Ihre Clientanwendung fest. Wenn Sie eine JDBC-Verbindung verwenden und clientseitige out-of-memory Fehler auftreten, können Sie Ihrem Client ermöglichen, Ergebnismengen in kleineren Batches abzurufen, indem Sie den Parameter JDBC-Abrufgröße festlegen. Weitere Informationen finden Sie unter Festlegen des JDBC-Parameters für die Abrufgröße.

Beispiele für DECLARE CURSOR

Im folgenden Beispiel wird ein Cursor namens LOLLAPALOOZA deklariert, um Verkaufsinformationen für das Lollapalooza-Ereignis auszuwählen und dann mittels des Cursors Zeilen aus dem Ergebnissatz abzurufen:

```
-- Begin a transaction

begin;

-- Declare a cursor

declare lollapalooza cursor for
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Lollapalooza';

-- Fetch the first 5 rows in the cursor lollapalooza:
```

```
fetch forward 5 from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-05-01 19:00:00 |   92.00000000 |       3
 Lollapalooza | 2008-11-15 15:00:00 |  222.00000000 |       2
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       3
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       4
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       1
(5 rows)

-- Fetch the next row:

fetch next from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-10-06 14:00:00 |  114.00000000 |       2

-- Close the cursor and end the transaction:

close lollapalooza;
commit;
```

Im folgenden Beispiel wird eine Schleife über einen Refcursor mit allen Ergebnissen aus einer Tabelle durchgeführt:

```
CREATE TABLE tbl_1 (a int, b int);
INSERT INTO tbl_1 values (1, 2),(3, 4);

CREATE OR REPLACE PROCEDURE sp_cursor_loop() AS $$
DECLARE
    target record;
    curs1 cursor for select * from tbl_1;
BEGIN
    OPEN curs1;
    LOOP
        fetch curs1 into target;
        exit when not found;
        RAISE INFO 'a %', target.a;
    END LOOP;
    CLOSE curs1;
END;
```

```
$$ LANGUAGE plpgsql;

CALL sp_cursor_loop();

SELECT message
   from svl_stored_proc_messages
   where querytxt like 'CALL sp_cursor_loop()%';

  message
----------
      a 1
      a 3
```

DELETE

Löscht Zeilen aus Tabellen.

> **Note**
>
> Die maximal zulässige Größe für eine einzelne SQL-Anweisung ist 16 MB.

Syntax

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
DELETE [ FROM ] { table_name | materialized_view_name }
    [ { USING } table_name, ... ]
    [ WHERE condition ]
```

Parameter

WITH-Klausel

Optionale Klausel, die eine oder mehrere allgemeine Tabellenausdrücke (CET) angibt. Siehe WITH-Klausel.

FROM

Das Schlüsselwort FROM ist optional, außer in den Fällen, in denen die USING-Klausel angegeben ist. Die Anweisungen `delete from event;` und `delete event;` sind gleichwertige Operationen, die alle Zeilen aus der Tabelle EVENT entfernen.

> **Note**
>
> Um alle Zeilen in einer Tabelle zu löschen, führen Sie für die Tabelle den Befehl TRUNCATE aus. TRUNCATE ist sehr viel effizienter als DELETE und erfordert weder VACUUM noch ANALYZE. Denken Sie jedoch daran, dass TRUNCATE ein Commit für die Transaktion ausführt, in der er ausgeführt wird.

table_name

Eine temporäre oder persistente Tabelle. Nur der Besitzer der Tabelle oder ein Benutzer mit dem Recht DELETE für die Tabelle können Zeilen aus der Tabelle löschen.

Sie sollten den Befehl TRUNCATE für schnelle, nicht qualifizierte Löschoperationen für große Tabellen verwenden; siehe TRUNCATE.

> **Note**
>
> Nach dem Löschen einer großen Zahl von Zeilen aus einer Tabelle:
>
> - Führen Sie eine Vacuum-Operation für die Tabelle durch, um Speicherplatz zurückzugewinnen und die Zeilen neu zu sortieren.
> - Analysieren Sie die Tabelle, um Statistiken für den Abfrageplaner zu aktualisieren.

materialized_view_name

Eine materialisierte Ansicht. Die DELETE-Anweisung funktioniert in einer materialisierten Ansicht, die für Streaming-Erfassung benutzt wird. Nur der Besitzer der materialisierten Ansicht oder ein Benutzer mit DELETE-Privilegien für die materialisierte Ansicht kann Zeilen daraus löschen.

Sie können DELETE nicht für eine materialisierte Ansicht zur Streaming-Aufnahme mit einer RLS-Richtlinie (Row Level Security) ausführen, für die dem Benutzer nicht die Berechtigung IGNORE RLS erteilt wurde. Es gibt eine Ausnahme: Wenn dem Benutzer, der den Löschvorgang ausführt, die Option IGNORE RLS gewährt wurde, wird der Vorgang erfolgreich ausgeführt. Weitere Informationen finden Sie unter Eigentümerschaft und Verwaltung von RLS-Richtlinien.

USING table_name, ...

Das Schlüsselwort USING wird verwendet, um eine Tabellenliste einzuführen, wenn in der WHERE-Klauselbedingung zusätzliche Tabellen referenziert werden. Die folgende Anweisung

löscht beispielsweise alle Zeilen aus der Tabelle EVENT, die die Join-Bedingung für die Tabellen EVENT und SALES erfüllen. Die Tabelle SALES muss in der Liste FROM explizit genannt werden:

```
delete from event using sales where event.eventid=sales.eventid;
```

Wenn Sie den Namen der Zieltabelle in der USING-Klausel wiederholen, führt die DELETE-Operation einen Self-Join aus. Sie können anstelle der USING-Syntax eine Unterabfrage in der WHERE-Klausel verwenden, um die gleiche Abfrage auf eine andere Art zu schreiben.

WHERE condition

Optionale Klausel, die das Löschen von Zeilen auf die Zeilen einschränkt, die der Bedingung entsprechen. Bei der Bedingung kann es sich beispielsweise um eine Einschränkung für eine Spalte, eine Join-Bedingung oder eine Bedingung auf der Basis eines Abfrageergebnisses handeln. Die Abfrage kann andere Tabellen als das Ziel des Befehls DELETE referenzieren. Beispiel:

```
delete from t1
where col1 in(select col2 from t2);
```

Wenn keine Bedingung angegeben wird, werden alle Zeilen in der Tabelle gelöscht.

Beispiele

Löschen aller Zeilen aus der Tabelle CATEGORY:

```
delete from category;
```

Löschen von Zeilen mit CATID-Werten zwischen 0 und 9 aus der Tabelle CATEGORY:

```
delete from category
where catid between 0 and 9;
```

Löschen von Zeilen aus der Tabelle LISTING, deren SELLERID-Werte in der Tabelle SALES nicht vorhanden sind:

```
delete from listing
```

```
where listing.sellerid not in(select sales.sellerid from sales);
```

Die folgenden beiden Abfragen löschen jeweils eine einzelne Zeile aus der Tabelle CATEGORY, basierend auf einem Join mit der Tabelle EVENT und einer zusätzlichen Einschränkung für die Spalte CATID:

```
delete from category
using event
where event.catid=category.catid and category.catid=9;
```

```
delete from category
where catid in
(select category.catid from category, event
where category.catid=event.catid and category.catid=9);
```

Die folgende Abfrage löscht alle Zeilen aus der `mv_cities` materialisierten Ansicht. Der hier verwendete Name der materialisierten Ansicht ist ein Beispiel:

```
delete from mv_cities;
```

DESC DATASHARE

Zeigt eine Liste der Datenbankobjekte in einem Datashare an, die mit ALTER DATASHARE hinzugefügt werden. Amazon Redshift zeigt die Namen, Datenbanken, Schemata und Typen von Tabellen, Ansichten und Funktionen an.

Zusätzliche Informationen zu DataShare-Objekten finden Sie mithilfe von Systemansichten. Weitere Informationen finden Sie unter SVV_DATASHARE_OBJECTS und SVV_DATASHARES.

Syntax

```
DESC DATASHARE datashare_name [ OF [ ACCOUNT account_id ] NAMESPACE namespace_guid ]
```

Parameter

datashare_name

Der Name des Datashares.

NAMESPACE namespace_guid

Ein Wert, der den Namespace angibt, den das Datashare verwendet. Wenn Sie DESC DATASHARE als Administrator eines Konsumenten-Clusters ausführen, geben Sie die Option NAMESPACE an, um eingehende Datashares anzuzeigen.

ACCOUNT account_id

Ein Wert, der das Konto angibt, zu dem das Datashare gehört.

Nutzungshinweise

Wenn Sie als Administrator für Benutzerkonten DESC DATASHARE ausführen, um eingehende Datenfreigaben innerhalb des Kontos zu sehen, geben Sie die Option NAMESPACE an. AWS Wenn Sie DESC DATASHARE ausführen, um eingehende Datenfreigaben zwischen Konten zu sehen, geben Sie die Optionen ACCOUNT und NAMESPACE an. AWS

Beispiele

Im folgenden Beispiel werden Informationen zu den ausgehenden Datashares auf einem Produzenten-Cluster angezeigt.

```
DESC DATASHARE salesshare;

producer_account |          producer_namespace           | share_type  | share_name   |
 object_type |        object_name           |  include_new
-----------------+--------------------------------------+-------------+--------------
+-------------+------------------------------+--------------
 123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d  | OUTBOUND    |  salesshare  |
 TABLE       | public.tickit_sales_redshift |
 123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d  | OUTBOUND    |  salesshare  |
 SCHEMA      | public                       |   t
```

Im folgenden Beispiel werden Informationen zu den eingehenden Datashares auf einem Konsumenten-Cluster angezeigt.

```
DESC DATASHARE salesshare of ACCOUNT '123456789012' NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';

 producer_account |          producer_namespace          | share_type | share_name |
 object_type |         object_name          |  include_new
```

```
------------------+--------------------------------------+-----------+-----------
+-------------+------------------------------+-------------
 123456789012     | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare |
 table       | public.tickit_sales_redshift |
 123456789012     | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare |
 schema      | public                       |
(2 rows)
```

DESC IDENTITY PROVIDER

Zeigt Informationen über einen Identitätsanbieter an. Nur Superuser können Identitätsanbieter beschreiben.

Syntax

```
DESC IDENTITY PROVIDER identity_provider_name
```

Parameter

identity_provider_name

Name des Identitätsanbieters.

Beispiel

Im folgenden Beispiel werden Informationen zu dem Identitätsanbieter angezeigt.

```
DESC IDENTITY PROVIDER azure_idp;
```

Beispielausgabe.

```
  uid   |   name    | type  |             instanceid              | namespc |

                                          params

                          | enabled
--------+-----------+-------+--------------------------------------+---------
+-------------------------------------------------------------------------------------------
+---------
 126692 | azure_idp | azure | e40d4bb2-7670-44ae-bfb8-5db013221d73 | aad     |
 {"issuer":"https://login.microsoftonline.com/e40d4bb2-7670-44ae-bfb8-5db013221d73/
```

```
v2.0", "client_id":"871c010f-5e61-4fb1-83ac-98610a7e9110", "client_secret":'',
 "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift", "https://
analysis.windows.net/powerbi/connector/AWSRDS"]} | t
(1 row)
```

TRENNEN EINER MASKIERUNGSRICHTLINIE

Trennt eine bereits angefügte Richtlinie für die dynamische Datenmaskierung von einer Spalte. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Maskierungsrichtlinie trennen.

Syntax

```
DETACH MASKING POLICY policy_name
   ON { table_name }
   ( output_column_names )
   FROM { user_name | ROLE role_name | PUBLIC };
```

Parameter

policy_name

Der Name der zu trennenden Maskierungsrichtlinie.

table_name

Der Name der Tabelle, von der die Maskierungsrichtlinie getrennt werden soll.

output_column_names

Die Namen der Spalten, denen die Maskierungsrichtlinie angefügt wurde.

user_name

Der Name des Benutzers, dem die Maskierungsrichtlinie angefügt wurde.

In einer einzelnen Anweisung DETACH MASKING POLICY können Sie nur entweder user_name, role_name oder PUBLIC festlegen.

role_name

Der Name der Rolle, der die Maskierungsrichtlinie angefügt wurde.

In einer einzelnen Anweisung DETACH MASKING POLICY können Sie nur entweder user_name, role_name oder PUBLIC festlegen.

PUBLIC

Zeigt an, dass die Richtlinie allen Benutzern in der Tabelle angefügt wurde.

In einer einzelnen Anweisung DETACH MASKING POLICY können Sie nur entweder user_name, role_name oder PUBLIC festlegen.

DETACH RLS POLICY

Trennen Sie eine RLS-Richtlinie auf Zeilenebene für eine Tabelle von einem oder mehreren Benutzern oder Rollen.

Superuser und Benutzer oder Rollen, die die `sys:secadmin`-Rolle haben, können eine Richtlinie aufheben.

Syntax

```
DETACH RLS POLICY policy_name ON [TABLE] table_name [, ...]
FROM { user_name | ROLE role_name | PUBLIC } [, ...]
```

Parameter

policy_name

Der Name der -Richtlinie.

AUF [TABLE] table_name [, ...]

Die Tabelle oder Ansicht, von der die RLS-Richtlinie auf Zeilenebene getrennt ist.

VON { user_name | ROLE role_name | PUBLIC} [, ...]

Gibt an, ob die Richtlinie von einem oder mehreren angegebenen Benutzern oder Rollen getrennt ist.

Nutzungshinweise

Beachten Sie bei der Arbeit mit der DETACH RLS POLICY-Anweisung Folgendes:

- Sie können eine Richtlinie von einer Relation, einem Benutzer, einer Rolle oder einer Öffentlichkeit trennen.

Beispiele

Im folgenden Beispiel wird eine Richtlinie für eine Tabelle von einer Rolle getrennt.

```
DETACH RLS POLICY policy_concerts ON tickit_category_redshift FROM ROLE analyst, ROLE
 dbadmin;
```

DROP DATABASE

Entfernt eine Datenbank.

DROP DATABASE kann nicht innerhalb eines Transaktionsblocks (BEGIN ... END) ausgeführt werden. Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Syntax

```
DROP DATABASE database_name
```

Parameter

database_name

Der Name der Datenbank, die entfernt werden soll. Sie können die Datenbanken dev, padb_harvest, template0, template1, sys:internal oder die aktuelle Datenbank nicht entfernen.

Um eine externe Datenbank zu entfernen, entfernen Sie das externe Schema. Weitere Informationen finden Sie unter DROP SCHEMA.

Nutzungshinweise für DROP DATABASE

Wenn Sie die DROP DATABASE-Anweisung verwenden, beachten Sie Folgendes:

- Im Allgemeinen empfehlen wir, eine Datenbank, die eine Datenfreigabe enthält, nicht mit der DROP DATABASE-Anweisung zu löschen. AWS Data Exchange Wenn Sie dies tun, verlieren diejenigen, AWS-Konten die Zugriff auf die Datenfreigabe haben, den Zugriff. Diese Art der Änderung kann außerdem zu einer Verletzung der Datenproduktbedingungen in AWS Data Exchange führen.

Das folgende Beispiel zeigt einen Fehler, wenn eine Datenbank, die ein AWS Data Exchange Datashare enthält, gelöscht wird.

```
DROP DATABASE test_db;
ERROR:  Drop of database test_db that contains ADX-managed datashare(s)
 requires session variable datashare_break_glass_session_var to be set to value
 'ce8d280c10ad41'
```

Um das Entfernen einer Datenbank zu erlauben, legen Sie die folgende Variable fest und führen die DROP DATABASE-Anweisung erneut aus.

```
SET datashare_break_glass_session_var to 'ce8d280c10ad41';
```

```
DROP DATABASE test_db;
```

In diesem Fall generiert Amazon Redshift einen zufälligen Einmalwert zur Festlegung der Sitzungsvariable, um DROP DATABASE für eine Datenbank, die ein AWS Data Exchange - Datashare enthält, zu erlauben.

Beispiele

Im folgenden Beispiel wird die Datenbank mit dem Namen TICKIT_TEST entfernt:

```
drop database tickit_test;
```

DROP DATASHARE

Entfernt ein Datashare. Dieser Befehl kann nicht rückgängig gemacht werden.

Nur ein Superuser oder der Datashare-Besitzer kann ein Datashare entfernen.

Erforderliche Berechtigungen

Für DROP DATASHARE sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP DATASHARE

- Besitzer von Datashare

Syntax

```
DROP DATASHARE datashare_name;
```

Parameter

datashare_name

Der Name des Datashares, das entfernt werden soll.

Nutzungshinweise für DROP DATASHARE

Wenn Sie die DROP DATASHARE-Anweisung verwenden, beachten Sie Folgendes:

- Im Allgemeinen empfehlen wir, eine AWS Data Exchange Datenfreigabe nicht mit der DROP DATASHARE-Anweisung zu löschen. Wenn Sie dies tun, verlieren diejenigen, AWS-Konten die Zugriff auf den Datashare haben, den Zugriff. Diese Art der Änderung kann außerdem zu einer Verletzung der Datenproduktbedingungen in AWS Data Exchange führen.

  Das folgende Beispiel zeigt einen Fehler, wenn ein AWS Data Exchange Datashare gelöscht wird.

  ```
  DROP DATASHARE salesshare;
  ERROR:  Drop of ADX-managed datashare salesshare requires session variable
   datashare_break_glass_session_var to be set to value '620c871f890c49'
  ```

  Um das Löschen einer AWS Data Exchange Datenfreigabe zu ermöglichen, legen Sie die folgende Variable fest und führen Sie die DROP DATASHARE-Anweisung erneut aus.

  ```
  SET datashare_break_glass_session_var to '620c871f890c49';
  ```

  ```
  DROP DATASHARE salesshare;
  ```

  In diesem Fall generiert Amazon Redshift einen zufälligen Einmalwert, um die Sitzungsvariable so einzustellen, dass DROP DATASHARE für eine Datenfreigabe zugelassen wird. AWS Data Exchange

Beispiele

Im folgenden Beispiel wird ein Datashare namens `salesshare` entfernt.

```
DROP DATASHARE salesshare;
```

DROP EXTERNAL VIEW (Vorschau)

Hierbei handelt es sich um die vorab veröffentlichte Dokumentation im Datenkatalog für Amazon Redshift, derzeit in der Vorschauversion. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur mit Testclustern und nicht in Produktionsumgebungen zu verwenden. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter Beta- und Vorschauversionen in den AWS -Servicebedingungen.

Sie können einen Amazon-Redshift-Cluster in der Vorschau erstellen, um neue Funktionen von Amazon Redshift zu testen. Sie haben nicht die Möglichkeit, diese Funktionen in der Produktion zu verwenden oder Ihren Vorschau-Cluster in einen Produktionscluster oder einen Cluster auf einem anderen Pfad zu verschieben. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Erstellen eines Clusters in der Vorschau

1. Melden Sie sich bei der Amazon Redshift Redshift-Konsole an AWS Management Console und öffnen Sie sie unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im Navigationsmenü Provisioned clusters dashboard (Dashboard für bereitgestellte Cluster) und dann Clusters (Cluster) aus. Die aktuellen Cluster für Ihr Konto AWS-Region sind aufgeführt. Eine Teilmenge der Eigenschaften jedes Clusters wird in den Spalten der Liste angezeigt.
3. Auf der Seite mit der Clusterliste (Clusters) wird ein Banner angezeigt, das die Vorschau vorstellt. Wählen Sie die Schaltfläche Create preview cluster (Vorschau-Cluster erstellen) aus, um die Seite zum Erstellen von Clustern zu öffnen.
4. Geben Sie Eigenschaften für Ihren Cluster ein. Wählen Sie den Vorschaupfad (Preview track) aus, der die zu testenden Funktionen enthält. Wir empfehlen, einen Namen für den Cluster zu verwenden, der darauf hinweist, dass sich dieser auf einem Vorschaupfad befindet. Wählen Sie Optionen für Ihren Cluster, einschließlich Optionen mit der Bezeichnung -preview (Vorschau), für

die zu testenden Funktionen. Allgemeine Informationen zum Erstellen von Clustern finden Sie unter Erstellen eines Clusters im Amazon-Redshift-Verwaltungshandbuch.

5. Wählen Sie Vorschau-Cluster erstellen aus, um einen Cluster in der Vorschau zu erstellen.

> **Note**
>
> Der `preview_2023`-Track ist der neueste verfügbare Vorschau-Track. Dieser Track unterstützt nur die Erstellung von Clustern mit RA3-Knotentypen. Der Knotentyp DC2 und alle älteren Knotentypen werden nicht unterstützt.

6. Wenn Ihr Vorschau-Cluster verfügbar ist, verwenden Sie Ihren SQL-Client, um Daten zu laden und abzufragen.

Das Vorschau-Feature für den Datenkatalog ist nur in den folgenden Regionen verfügbar.

- USA Ost (Ohio): (us-east-2)
- USA Ost (Nord-Virginia): (us-east-1)
- USA West (Nordkalifornien) (us-west-1)
- Asien-Pazifik (Tokyo) (ap-northeast-1)
- Europa (Irland) (eu-west-1)
- Europa (Stockholm) (eu-north-1)

Sie können außerdem eine Vorschau-Arbeitsgruppe erstellen, um Datenkatalog-Ansichten zu testen. Sie können diese Features nicht in der Produktion verwenden und Ihre Vorschau-Arbeitsgruppe auch nicht in eine andere Arbeitsgruppe verschieben. Weitere Informationen zu den Nutzungsbedingungen finden Sie unter „Beta- und Vorschauversionen“ in den AWS -Servicebedingungen. Eine Anleitung zur Erstellung einer Vorschau-Arbeitsgruppe finden Sie unter https://docs.aws.amazon.com/redshift/latest/mgmt/serverless-workgroup-preview.html.

Entfernt eine externe Ansicht aus der Datenbank. Wenn Sie eine externe Ansicht entfernen, wird sie aus allen SQL-Engines entfernt, mit denen die Ansicht verknüpft ist, z. B. Amazon Athena und Amazon EMR Spark. Dieser Befehl kann nicht rückgängig gemacht werden. Weitere Informationen zu Datenkatalog-Ansichten finden Sie unter Erstellen von Datenkatalog-Ansichten (Vorschau).

Syntax

```
DROP EXTERNAL VIEW schema_name.view_name [ IF EXISTS ]
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
```

Parameter

schema_name.view_name

Das Schema, das an Ihre AWS Glue Datenbank angehängt ist, gefolgt vom Namen der Ansicht.

IF EXISTS

Löscht die Ansicht nur, wenn sie existiert.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

Die Notation des Schemas, das beim Löschen der Ansicht verwendet werden soll. Sie können angeben AWS Glue Data Catalog, ob Sie eine von Ihnen erstellte Glue-Datenbank oder ein von Ihnen erstelltes externes Schema verwenden möchten. Weitere Informationen finden Sie unter CREATE DATABASE und CREATE EXTERNAL SCHEMA.

query_definition

Die Definition der SQL-Abfrage, die Amazon Redshift ausführt, um die Ansicht zu ändern.

Beispiele

Im folgenden Beispiel wird eine Datenkatalog-Ansicht mit dem Namen sample_schema.glue_data_catalog_view entfernt.

```
DROP EXTERNAL VIEW sample_schema.glue_data_catalog_view IF EXISTS
```

DROP FUNCTION

Entfernt eine benutzerdefinierte Funktion (User-Defined Function, UDF) aus der Datenbank. Die Signatur der Funktion bzw. die Liste der Argumentdatentypen muss angegeben werden, da mehrere Funktionen mit demselben Namen, jedoch unterschiedlichen Signaturen vorhanden sein können. Sie können keine in Amazon Redshift integrierten Funktionen entfernen.

Dieser Befehl kann nicht rückgängig gemacht werden.

Erforderliche Berechtigungen

Für DROP FUNCTION sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP FUNCTION
- Funktionsbesitzer

Syntax

```
DROP FUNCTION name
( [arg_name] arg_type   [, ...] )
[ CASCADE | RESTRICT ]
```

Parameter

Name

Das Name der Funktion, die entfernt werden soll.

arg_name

Der Name eines Eingabeargumente. DROP FUNCTION ignoriert Argumentnamen, da nur die Argumentdatentypen benötigt werden, um die Identität der Funktion zu ermitteln.

arg_type

Der Datentyp des Eingabeargumente. Sie können eine durch Komma getrennte Liste mit maximal 32 Datentypen bereitstellen.

CASCADE

Ein Schlüsselwort, das angibt, dass von der Funktion abhängige Objekte, wie etwa Ansichten, automatisch entfernt werden sollen.

Zum Erstellen einer Ansicht, die von keiner Funktion abhängig ist, fügen Sie die Klausel WITH NO SCHEMA BINDING in die Ansichtsdefinition ein. Weitere Informationen finden Sie unter CREATE VIEW.

RESTRICT

Ein Schlüsselwort, das angibt, dass die Funktion nicht entfernt und eine Meldung zurückgegeben werden soll, wenn von der Funktion Objekte abhängig sind. Diese Aktion ist die Standardeinstellung.

Beispiele

Im folgenden Beispiel wird die Funktion namens entfernt `f_sqrt`:

```
drop function f_sqrt(int);
```

Um eine Funktion zu entfernen, von der Objekte abhängig sind, verwenden Sie die Option CASCADE, wie im folgenden Beispiel gezeigt:

```
drop function f_sqrt(int)cascade;
```

DROP GROUP

Löscht eine Benutzergruppe. Dieser Befehl kann nicht rückgängig gemacht werden. Dieser Befehl löscht nicht die einzelnen Benutzer in einer Gruppe.

Informationen zum Löschen eines einzelnen Benutzers finden Sie unter DROP USER.

Syntax

```
DROP GROUP name
```

Parameter

Name

Der Name der Benutzergruppe, die gelöscht werden soll.

Beispiel

Im folgenden Beispiel wird die `guests` Benutzergruppe gelöscht:

```
DROP GROUP guests;
```

Sie können eine Gruppe nicht entfernen, wenn die Gruppe Rechte für ein Objekt besitzt. Wenn Sie versuchen, eine solche Gruppe zu löschen, erhalten Sie die folgende Fehlermeldung.

```
ERROR: group "guests" can't be dropped because the group has a privilege on some object
```

Wenn die Gruppe über Rechte für ein Objekt verfügt, müssen Sie die Rechte entziehen, bevor Sie die Gruppe löschen. Verwenden Sie das folgende Beispiel, um die Objekte zu finden, für die die `guests` Gruppe Berechtigungen besitzt. Weitere Informationen zur im Beispiel verwendeten Metadatenansicht finden Sie unter SVV_RELATION_PRIVILEGES.

```
SELECT DISTINCT namespace_name, relation_name, identity_name, identity_type
FROM svv_relation_privileges
WHERE identity_type='group' AND identity_name='guests';

+----------------+---------------+---------------+---------------+
| namespace_name | relation_name | identity_name | identity_type |
+----------------+---------------+---------------+---------------+
| public         | table1        | guests        | group         |
+----------------+---------------+---------------+---------------+
| public         | table2        | guests        | group         |
+----------------+---------------+---------------+---------------+
```

Im folgenden Beispiel werden alle Rechte für alle Tabellen im Schema `public` von der Benutzergruppe `guests` entfernt und die Gruppe anschließend entfernt.

```
REVOKE ALL ON ALL TABLES IN SCHEMA public FROM GROUP guests;
DROP GROUP guests;
```

DROP IDENTITY PROVIDER

Löscht einen Identitätsanbieter. Dieser Befehl kann nicht rückgängig gemacht werden. Nur Superuser können Identitätsanbieter löschen.

Syntax

```
DROP IDENTITY PROVIDER identity_provider_name [ CASCADE ]
```

Parameter

identity_provider_name

Name des zu löschenden Identitätsanbieters.

CASCADE

Löscht Benutzer und Rollen, die mit dem Identitätsanbieter verbunden sind, wenn dieser gelöscht wird.

Beispiel

Im folgenden Beispiel wird der Identitätsanbieter oauth_provider gelöscht.

```
DROP IDENTITY PROVIDER oauth_provider;
```

Wenn Sie den Identitätsanbieter löschen, sind manche Benutzer möglicherweise nicht in der Lage, sich anzumelden oder Client-Tools zu verwenden, die für die Verwendung des Identitätsanbieters konfiguriert sind.

DROP LIBRARY

Entfernt eine angepasste Python-Bibliothek aus der Datenbank. Nur der Besitzer der Bibliothek oder ein Superuser können eine Bibliothek entfernen.

DROP LIBRARY kann nicht innerhalb eines Transaktionsblocks ausgeführt werden (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Dieser Befehl kann nicht rückgängig gemacht werden. Der Befehl DROP LIBRARY führt sofort ein Commit aus. Wenn gleichzeitig eine UDF ausgeführt wird, die von der Bibliothek abhängig ist, schlägt die UDF möglicherweise fehl, auch wenn die UDF innerhalb einer Transaktion ausgeführt wird.

Weitere Informationen finden Sie unter CREATE LIBRARY.

Erforderliche Berechtigungen

Für DROP LIBRARY sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP LIBRARY

- Besitzer der Bibliothek

Syntax

```
DROP LIBRARY library_name
```

Parameter

library_name

Der Name der Bibliothek.

ENTFERNEN EINER MASKIERUNGSRICHTLINIE

Entfernt eine Richtlinie für die dynamische Datenmaskierung aus allen Datenbanken. Eine Maskierungsrichtlinie, die immer noch mindestens einer Tabelle angefügt ist, kann nicht entfernt werden. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Richtlinie entfernen.

Syntax

```
DROP MASKING POLICY policy_name;
```

Parameter

policy_name

Der Name der zu entfernenden Maskierungsrichtlinie.

DROP MODEL

Entfernt ein Modell aus der Datenbank. Nur der Besitzer des Modells oder ein Superuser können ein Modell entfernen.

DROP MODELL löscht außerdem alle zugehörigen Vorhersagefunktionen, die von diesem Modell abgeleitet sind, alle Amazon-Redshift-Artefakte, die sich auf das Modell beziehen, sowie alle

Amazon-S3-Daten, die sich auf das Modell beziehen. Während das Modell noch in Amazon trainiert wird SageMaker, storniert DROP MODEL diese Operationen.

Dieser Befehl kann nicht rückgängig gemacht werden. Der Befehl DROP MODEL führt sofort ein Commit aus.

Erforderliche Berechtigungen

Für DROP MODEL sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP MODEL.
- Modellbesitzer
- Schemabesitzer

Syntax

```
DROP MODEL [ IF EXISTS ] model_name
```

Parameter

IF EXISTS

Eine Klausel, die angibt, dass, wenn das angegebene Schema bereits vorhanden ist, der Befehl keine Änderungen vornehmen und eine Meldung zurückgeben soll, dass das Schema existiert.

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

Beispiele

Im folgenden Beispiel wird das Modell demo_ml.customer_churn entfernt.

```
DROP MODEL demo_ml.customer_churn
```

DROP MATERIALIZED VIEW

Entfernt eine materialisierte Ansicht.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

Syntax

```
DROP MATERIALIZED VIEW [ IF EXISTS ] mv_name [ CASCADE | RESTRICT ]
```

Parameter

IF EXISTS

Eine Klausel, die angibt, dass überprüfen werden soll, ob die benannte materialisierte Ansicht vorhanden ist. Wenn die materialisierte Ansicht nicht existiert, gibt der Befehl `DROP MATERIALIZED VIEW` eine Fehlermeldung zurück. Diese Klausel ist nützlich beim Skripten, um zu verhindern, dass das Skript versagt, wenn Sie ein Drop für eine nicht vorhandene materialisierte Ansicht durchführen.

mv_name

Der Name der zu löschenden materialisierten Ansicht.

CASCADE

Eine Klausel, die angibt, dass Objekte, von denen die materialisierte Ansicht abhängt, wie z. B. andere Ansichten, automatisch gelöscht werden sollen.

RESTRICT

Eine Klausel, die angibt, dass die materialisierte Ansicht nicht gelöscht werden soll, wenn Objekte von ihr abhängen. Dies ist die Standardeinstellung.

Nutzungshinweise

Nur der Besitzer einer materialisierten Ansicht kann für diese Ansicht `DROP MATERIALIZED VIEW` verwenden. Ausnahmen hiervon können Superuser oder Benutzer sein, denen ausdrücklich DROP-Berechtigungen erteilt wurden.

Wenn Sie eine Drop-Anweisung für eine materialisierte Ansicht schreiben und eine Ansicht mit einem übereinstimmenden Namen vorhanden ist, führt dies zu einem Fehler mit der Anweisung, DROP VIEW zu verwenden. Ein Fehler tritt auch dann auf, wenn Sie `DROP MATERIALIZED VIEW IF EXISTS` verwenden.

Beispiel

Das folgende Beispiel löscht die materialisierte `tickets_mv`-Ansicht.

```
DROP MATERIALIZED VIEW tickets_mv;
```

DROP PROCEDURE

Entfernt eine Prozedur. Zum Entfernen einer Prozedur sind sowohl der Name der Prozedur als auch die Eingabeargument-Datentypen (Signatur) erforderlich. Optional können Sie alle Argumentdatentypen angeben, auch die der OUT-Argumente. Verwenden Sie den Befehl SHOW PROCEDURE, um die Signatur für eine Prozedur zu finden. Weitere Informationen zu Prozedursignaturen finden Sie unter PG_PROC_INFO.

Erforderliche Berechtigungen

Für DROP PROCEDURE sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP PROCEDURE
- Besitzer des Verfahrens

Syntax

```
DROP PROCEDURE sp_name ( [ [ argname ] [ argmode ] argtype [, ...] ] )
```

Parameter

sp_name

Der Name der Prozedur, die entfernt werden soll.

argname

Der Name eines Eingabeargumentes. DROP PROCEDURE ignoriert Argumentnamen, da nur die Argumentdatentypen benötigt werden, um die Identität der Prozedur zu ermitteln.

argmode

Der Modus eines Arguments. Dieser kann IN, OUT oder INOUT sein. OUT-Argumente sind optional, da diese nicht zum Identifizieren einer gespeicherten Prozedur verwendet werden.

argtype

Der Datentyp des Eingabearguments. Eine Liste der unterstützten Datentypen finden Sie unter Datentypen.

Beispiele

Im folgenden Beispiel wird eine Prozedur namens `quarterly_revenue` entfernt.

```
DROP PROCEDURE quarterly_revenue(volume INOUT bigint, at_price IN numeric,result OUT
 int);
```

DROP RLS POLICY

Entfernt eine RLS-Richtlinie auf Zeilenebene für alle Tabellen in allen Datenbanken.

Superuser und Benutzer oder Rollen mit der Rolle sys:secadmin können eine Richtlinie löschen.

Syntax

```
DROP RLS POLICY [ IF EXISTS ] policy_name [ CASCADE | RESTRICT ]
```

Parameter

IF EXISTS

Eine Klausel, die angibt, ob die angegebene Richtlinie bereits existiert.

policy_name

Der Name der -Richtlinie.

CASCADE

Eine Klausel, die angibt, dass die Richtlinie automatisch von allen angefügten Tabellen getrennt werden soll, bevor die Richtlinie gelöscht wird.

RESTRICT

Eine Klausel, die angibt, die Richtlinie nicht zu löschen, wenn sie einigen Tabellen zugeordnet ist. Dies ist die Standardeinstellung.

Beispiele

Im folgenden Beispiel wird eine RLS-Richtlinie auf Zeilinienebene entfernt.

```
DROP RLS POLICY policy_concerts;
```

DROP ROLE

Entfernt eine Rolle aus einer Datenbank. Nur der Rollenbesitzer, der die Rolle erstellt hat, ein Benutzer mit der Option WITH ADMIN oder ein Superuser kann eine Rolle löschen.

Rollen, die einem Benutzer zugewiesen sind, oder Rollen, von denen andere Rollen abhängig sind, können nicht gelöscht werden.

Erforderliche Berechtigungen

Für DROP ROLE sind folgende Berechtigungen erforderlich:

- Superuser
- Rollenbesitzer, der entweder der Benutzer ist, der die Rolle erstellt hat, oder ein Benutzer, dem die Rolle mit dem Privileg WITH ADMIN OPTION gewährt wurde.

Syntax

```
DROP ROLE role_name [ FORCE | RESTRICT ]
```

Parameter

rollen_name

Der Name der Rolle.

[FORCE | RESTRICT]

Die Standardeinstellung ist RESTRICT. Amazon Redshift löst einen Fehler aus, wenn Sie versuchen, eine Rolle zu löschen, die eine andere Rolle geerbt hat. Verwenden Sie FORCE, um alle Rollenzuweisungen zu entfernen, falls vorhanden.

Beispiele

Im folgenden Beispiel wird die Rolle `sample_role` gelöscht.

```
DROP ROLE sample_role FORCE;
```

Im folgenden Beispiel wird versucht, die Rolle sample_role1 zu löschen, die einem Benutzer mit der Standardoption RESTRICT zugewiesen wurde.

```
CREATE ROLE sample_role1;
GRANT sample_role1 TO user1;
DROP ROLE sample_role1;
ERROR:  cannot drop this role since it has been granted on a user
```

Verwenden Sie die Option FORCE, um die Rolle sample_role1, die einem Benutzer zugewiesen wurde, erfolgreich zu löschen.

```
DROP ROLE sample_role1 FORCE;
```

Im folgenden Beispiel wird versucht, die Rolle sample_role2 zu löschen, von der eine andere Rolle mit der Standardoption RESTRICT abhängig ist.

```
CREATE ROLE sample_role1;
CREATE ROLE sample_role2;
GRANT sample_role1 TO sample_role2;
DROP ROLE sample_role2;
ERROR:  cannot drop this role since it depends on another role
```

Um die sample_role2 erfolgreich zu löschen, von der eine andere Rolle abhängig ist, verwenden Sie die Option FORCE.

```
DROP ROLE sample_role2 FORCE;
```

DROP SCHEMA

Löscht ein Schema. Im Fall von externen Schemen können Sie auch die mit dem Schema verknüpfte externe Datenbank entfernen. Dieser Befehl kann nicht rückgängig gemacht werden.

Erforderliche Berechtigungen

Für DROP SCHEMA sind folgende Berechtigungen erforderlich:

- Superuser
- Schemabesitzer
- Benutzer mit der Berechtigung DROP SCHEMA

Syntax

```
DROP SCHEMA [ IF EXISTS ] name [, ...]
[ DROP EXTERNAL DATABASE ]
[ CASCADE | RESTRICT ]
```

Parameter

IF EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass das Schema nicht vorhanden ist, statt mit einem Fehler beendet zu werden, wenn das angegebene Schema nicht vorhanden ist.

Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn DROP SCHEMA für ein nicht vorhandenes Schema ausgeführt wird.

Name

Namen der Schemata, die entfernt werden sollen. Sie können mehrere Schemanamen durch Kommas getrennt eingeben.

DROP EXTERNAL DATABASE

Eine Klausel, die angibt, dass mit der Entfernung eines externen Schemas die mit diesem verknüpfte externe Datenbank entfernt werden soll, sofern vorhanden. Wenn keine externe Datenbank vorhanden ist, gibt der Befehl die Meldung zurück, dass keine externe Datenbank vorhanden ist. Wenn mehrere externe Schemata entfernt werden, werden auch alle mit den angegebenen Schemata verknüpften Datenbanken entfernt.

Wenn eine externe Datenbank abhängige Objekte wie beispielsweise Tabellen enthält, schließen Sie die Option CASCADE ein, um diese abhängigen Objekte ebenfalls zu entfernen.

Wenn Sie eine externe Datenbank entfernen, wird die Datenbank auch für alle anderen mit der Datenbank verknüpften externen Schemata entfernt. In anderen externen Schemata definierte Tabellen, die diese Datenbank verwenden, werden ebenfalls entfernt.

DROP EXTERNAL DATABASE unterstützt keine in einem HIVE-Metastore gespeicherten externen Datenbanken.

CASCADE

Ein Schlüsselwort, das angibt, dass alle Objekte im Schema automatisch entfernt werden sollen. Bei Angabe von DROP EXTERNAL DATABASE werden alle Objekte in der externen Datenbank ebenfalls entfernt.

RESTRICT

Ein Schlüsselwort, das angibt, dass ein Schema oder eine externe Datenbank nicht entfernt werden soll, wenn darin Objekte enthalten sind. Diese Aktion ist die Standardeinstellung.

Beispiel

Im folgenden Beispiel wird ein Schema namens S_SALES gelöscht. In diesem Beispiel wird RESTRICT als Sicherheitsmechanismus verwendet, damit das Schema nicht gelöscht wird, wenn es Objekte enthält. In diesem Fall müssen Sie die Schemaobjekte löschen, bevor Sie das Schema löschen.

```
drop schema s_sales restrict;
```

Im folgenden Beispiel werden ein Schema mit der Bezeichnung S_SALES und alle Objekte, die von diesem Schema abhängig sind, gelöscht.

```
drop schema s_sales cascade;
```

Im folgenden Beispiel wird entweder das Schema S_SALES gelöscht, sofern es vorhanden ist, oder es erfolgt keine Aktion, und es wird eine Meldung zurückgegeben, wenn das Schema nicht vorhanden ist.

```
drop schema if exists s_sales;
```

Im folgenden Beispiel werden ein externes Schema mit der Bezeichnung S_SPECTRUM und die mit diesem verknüpfte externe Datenbank gelöscht. In diesem Beispiel wird RESTRICT verwendet, damit

das Schema und die Datenbank nicht gelöscht werden, wenn sie Objekte enthalten. In diesem Fall müssen Sie die abhängigen Objekte löschen, bevor Sie das Schema und die Datenbank löschen.

```
drop schema s_spectrum drop external database restrict;
```

Im folgenden Beispiel werden verschiedene Schemata und die mit diesen verknüpften externen Datenbanken sowie alle abhängigen Objekte gelöscht.

```
drop schema s_sales, s_profit, s_revenue drop external database cascade;
```

DROP TABLE

Entfernt eine Tabelle aus einer Datenbank.

Wenn Sie alle Zeilen in einer Tabelle löschen möchten, ohne die Tabelle zu entfernen, verwenden Sie die Befehle DELETE oder TRUNCATE.

DROP TABLE entfernt Einschränkungen für die Zieltabelle. Mit einem einzigen DROP TABLE-Befehl können mehrere Tabellen entfernt werden.

DROP TABLE mit einer externen Tabelle kann nicht innerhalb einer Transaktion (BEGIN … END) ausgeführt werden. Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Ein Beispiel, in dem die DROP-Berechtigung an eine Gruppe vergeben wird, finden Sie unter GRANT Beispiele.

Erforderliche Berechtigungen

Für DROP TABLE sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP TABLE
- Tabellenbesitzer mit dem USAGE-Recht für das Schema

Syntax

```
DROP TABLE [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```

Parameter

IF EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass die Tabelle nicht vorhanden ist, statt mit einem Fehler beendet zu werden, wenn die angegebene Tabelle nicht vorhanden ist.

Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn DROP TABLE für eine nicht vorhandene Tabelle ausgeführt wird.

Name

Der Name der Tabelle, die entfernt werden soll.

CASCADE

Eine Klausel, die angibt, dass Objekte, die von der Tabelle abhängig sind, automatisch entfernt werden sollen, beispielsweise Ansichten.

Zum Erstellen einer Ansicht, die von keinen anderen Datenbankobjekten abhängig ist, wie etwa Ansichten und Tabellen, fügen Sie die Klausel WITH NO SCHEMA BINDING in die Ansichtsdefinition ein. Weitere Informationen finden Sie unter CREATE VIEW.

RESTRICT

Eine Klausel, die angibt, dass eine Tabelle nicht entfernt werden soll, wenn von ihr Objekte abhängig sind. Diese Aktion ist die Standardeinstellung.

Beispiele

Entfernen einer Tabelle, von der keine Objekte abhängig sind

Im folgenden Befehl wird eine Tabelle namens FEEDBACK erstellt und entfernt, von der keine Objekte abhängig sind:

```
create table feedback(a int);

drop table feedback;
```

Wenn eine Tabelle Spalten enthält, auf die von Ansichten oder anderen Tabellen verwiesen wird, zeigt Amazon Redshift eine Meldung wie die folgende an.

```
Invalid operation: cannot drop table feedback because other objects depend on it
```

Gleichzeitiges Entfernen von zwei Tabellen

Der folgende Befehlssatz erstellt eine Tabelle namens FEEDBACK und eine Tabelle namens BUYERS und entfernt anschließend beide Tabellen mit einem einzigen Befehl:

```
create table feedback(a int);

create table buyers(a int);

drop table feedback, buyers;
```

Entfernen einer Tabelle, von ein Objekt abhängig ist

Im Folgenden wird gezeigt, wie eine Tabelle namens FEEDBACK mithilfe der Option CASCADE entfernt wird.

Erstellen Sie zunächst mit dem Befehl CREATE TABLE eine einfache Tabelle namens FEEDBACK:

```
create table feedback(a int);
```

Erstellen Sie als Nächstes mit dem Befehl CREATE VIEW eine Ansicht namens FEEDBACK_VIEW, die von der Tabelle FEEDBACK abhängig ist:

```
create view feedback_view as select * from feedback;
```

Im folgenden Beispiel werden die Tabelle FEEDBACK und die Ansicht FEEDBACK_VIEW entfernt, da FEEDBACK_VIEW von der Tabelle FEEDBACK abhängig ist:

```
drop table feedback cascade;
```

Anzeigen der Abhängigkeiten für eine Tabelle

Verwenden Sie das folgende Beispiel, um die Abhängigkeiten für Ihre Tabelle zurückzugeben. Ersetzen Sie *my_schema* und *my_table* durch Ihr eigenes Schema und Ihre eigene Tabelle.

```
SELECT dependent_ns.nspname as dependent_schema
, dependent_view.relname as dependent_view
```

```
, source_ns.nspname as source_schema
, source_table.relname as source_table
, pg_attribute.attname as column_name
FROM pg_depend
JOIN pg_rewrite ON pg_depend.objid = pg_rewrite.oid
JOIN pg_class as dependent_view ON pg_rewrite.ev_class = dependent_view.oid
JOIN pg_class as source_table ON pg_depend.refobjid = source_table.oid
JOIN pg_attribute ON pg_depend.refobjid = pg_attribute.attrelid
    AND pg_depend.refobjsubid = pg_attribute.attnum
JOIN pg_namespace dependent_ns ON dependent_ns.oid = dependent_view.relnamespace
JOIN pg_namespace source_ns ON source_ns.oid = source_table.relnamespace
WHERE
source_ns.nspname = 'my_schema'
AND source_table.relname = 'my_table'
AND pg_attribute.attnum > 0
ORDER BY 1,2
LIMIT 10;
```

Verwenden Sie das folgende Beispiel, um *my_table* und seine Abhängigkeiten zu entfernen. In diesem Beispiel werden auch alle Abhängigkeiten für die Tabelle zurückgegeben, die entfernt wurde.

```
DROP TABLE my_table CASCADE;

SELECT dependent_ns.nspname as dependent_schema
, dependent_view.relname as dependent_view
, source_ns.nspname as source_schema
, source_table.relname as source_table
, pg_attribute.attname as column_name
FROM pg_depend
JOIN pg_rewrite ON pg_depend.objid = pg_rewrite.oid
JOIN pg_class as dependent_view ON pg_rewrite.ev_class = dependent_view.oid
JOIN pg_class as source_table ON pg_depend.refobjid = source_table.oid
JOIN pg_attribute ON pg_depend.refobjid = pg_attribute.attrelid
    AND pg_depend.refobjsubid = pg_attribute.attnum
JOIN pg_namespace dependent_ns ON dependent_ns.oid = dependent_view.relnamespace
JOIN pg_namespace source_ns ON source_ns.oid = source_table.relnamespace
WHERE
source_ns.nspname = 'my_schema'
AND source_table.relname = 'my_table'
AND pg_attribute.attnum > 0
ORDER BY 1,2
LIMIT 10;
```

```
+------------------+----------------+--------------+--------------+-------------+
| dependent_schema | dependent_view | source_schema | source_table | column_name |
+------------------+----------------+--------------+--------------+-------------+
```

Entfernen einer Tabelle mit IF EXISTS

Im folgenden Beispiel wird entweder die Tabelle FEEDBACK gelöscht, wenn sie vorhanden ist, oder es erfolgt keine Aktion, und es wird eine Meldung zurückgegeben, wenn die Tabelle nicht vorhanden ist:

```
drop table if exists feedback;
```

DROP USER

Entfernt einen Benutzer aus einer Datenbank. Mit einem einzigen DROP USER-Befehl können mehrere Benutzer entfernt werden. Sie müssen ein Datenbank-Superuser sein oder über die DROP USER-Berechtigung verfügen, um diesen Befehl ausführen zu können.

Syntax

```
DROP USER [ IF EXISTS ] name [, ... ]
```

Parameter

IF EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass der Benutzer nicht vorhanden ist, statt mit einem Fehler beendet zu werden, wenn der angegebene Benutzer nicht vorhanden ist.

Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn DROP USER für einen nicht vorhandenen Benutzer ausgeführt wird.

Name

Der Name des Benutzers, der entfernt werden soll. Sie können mehrere Benutzer angeben, indem Sie die einzelnen Benutzernamen durch Komma trennen.

Nutzungshinweise

Sie können den Benutzer namens `rdsdb` oder den Administratorbenutzer der Datenbank nicht löschen, der normalerweise den Namen `awsuser` oder `admin` hat.

Sie können einen Benutzer nicht entfernen, wenn der Benutzer ein Datenbankobjekt wie ein Schema, eine Datenbank, eine Tabelle oder eine Ansicht besitzt oder wenn der Benutzer über Berechtigungen für eine Datenbank, Tabelle, Spalte oder Gruppe verfügt. Wenn Sie versuchen, einen solchen Benutzer zu löschen, erhalten Sie eine der folgenden Fehlermeldungen.

```
ERROR: user "username" can't be dropped because the user owns some object [SQL
 State=55006]

ERROR: user "username" can't be dropped because the user has a privilege on some object
 [SQL State=55006]
```

Eine ausführliche Anleitung zum Suchen der Objekte, die einem Datenbankbenutzer gehören, finden Sie unter Wie behebe ich den Fehler „Benutzer kann nicht gelöscht werden“ in Amazon Redshift? im Wissenszentrum.

> **Note**
>
> Amazon Redshift überprüft nur die aktuelle Datenbank, bevor ein Benutzer entfernt wird. DROP USER gibt keinen Fehler zurück, wenn der Benutzer Datenbankobjekte oder Rechte für Objekte in einer anderen Datenbank besitzt. Wenn Sie einen Benutzer entfernen, der Objekte in einer anderen Datenbank besitzt, wird der Besitzer dieser Objekte in „unbekannt“ geändert.

Wenn ein Benutzer ein Objekt besitzt, entfernen Sie zunächst das Objekt oder ändern dessen Besitzer in einen anderen Besitzer, bevor Sie den ursprünglichen Besitzer entfernen. Wenn der Benutzer Rechte für ein Objekt besitzt, widerrufen Sie zunächst die Rechte, bevor Sie den Benutzer entfernen. Im folgenden Beispiel werden das Entfernen eines Objekts, das Ändern des Besitzers und das Widerrufen von Rechten gezeigt, bevor der Benutzer entfernt wird.

```
drop database dwdatabase;
alter schema dw owner to dwadmin;
revoke all on table dwtable from dwuser;
drop user dwuser;
```

Beispiele

Im folgenden Beispiel wird ein Benutzer namens „paulo“ entfernt:

```
drop user paulo;
```

Im folgenden Beispiel werden zwei Benutzer, „paulo“ und „martha“, entfernt:

```
drop user paulo, martha;
```

Im folgenden Beispiel wird entweder der Benutzer „paulo“ gelöscht, wenn er vorhanden ist, oder es erfolgt keine Aktion, und es wird eine Meldung zurückgegeben, wenn er nicht vorhanden ist:

```
drop user if exists paulo;
```

DROP VIEW

Entfernt eine Ansicht aus der Datenbank. Mit einem einzigen DROP VIEW-Befehl können mehrere Ansichten entfernt werden. Dieser Befehl kann nicht rückgängig gemacht werden.

Erforderliche Berechtigungen

Für DROP VIEW sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung DROP VIEW
- Besitzer der Ansicht

Syntax

```
DROP VIEW [ IF EXISTS ] name [, ... ] [ CASCADE | RESTRICT ]
```

Parameter

IF EXISTS

Eine Klausel, die angibt, dass der Befehl keine Änderungen ausführen und die Meldung zurückgeben soll, dass die Ansicht nicht vorhanden ist, statt mit einem Fehler beendet zu werden, wenn die angegebene Ansicht nicht vorhanden ist.

Diese Klausel ist beim Scripting nützlich, damit das Skript nicht fehlschlägt, wenn DROP VIEW für eine nicht vorhandene Ansicht ausgeführt wird.

Name

Das Name der Ansicht, die entfernt werden soll.

CASCADE

Eine Klausel, die angibt, dass Objekte, die von der Ansicht abhängig sind, automatisch entfernt werden sollen, beispielsweise andere Ansichten.

Zum Erstellen einer Ansicht, die von keinen anderen Datenbankobjekten abhängig ist, wie etwa Ansichten und Tabellen, fügen Sie die Klausel WITH NO SCHEMA BINDING in die Ansichtsdefinition ein. Weitere Informationen finden Sie unter CREATE VIEW.

Beachten Sie, dass Ihr Datenbankclient möglicherweise nicht alle gelöschten Objekte in den Übersichtsergebnissen auflistet, wenn Sie CASCADE einbeziehen und die Anzahl der gelöschten Datenbankobjekte auf zehn oder mehr steigt. Dies liegt in der Regel daran, dass SQL-Client-Tools Standardbeschränkungen für die zurückgegebenen Ergebnisse haben.

RESTRICT

Eine Klausel, die angibt, dass eine Ansicht nicht entfernt werden soll, wenn von ihr Objekte abhängig sind. Diese Aktion ist die Standardeinstellung.

Beispiele

Im folgenden Beispiel wird die Ansicht namens event entfernt:

```
drop view event;
```

Um eine Ansicht zu entfernen, von der Objekte abhängig sind, verwenden Sie die Option CASCADE. Nehmen Sie an, Sie beginnen mit einer Tabelle namens EVENT. Anschließend erstellen Sie die Ansicht „eventview“ der Tabelle EVENT mit dem Befehl CREATE VIEW wie im folgenden Beispiel gezeigt:

```
create view eventview as
select dateid, eventname, catid
from event where catid = 1;
```

Als Nächstes erstellen Sie eine zweite Ansicht namens myeventview, die auf der ersten Ansicht namens eventview basiert:

```
create view myeventview as
select eventname, catid
from eventview where eventname <> ' ';
```

An diesem Punkt wurden zwei Ansichten erstellt: eventview und myeventview.

Die Ansicht myeventview ist eine untergeordnete Ansicht der übergeordneten Ansicht eventview.

Um die Ansicht eventview zu löschen, verwenden Sie den folgenden Befehl:

```
drop view eventview;
```

Beachten Sie, dass der folgende Fehler angezeigt wird, wenn Sie diesen Befehl in dieser Situation ausführen:

```
drop view eventview;
ERROR: can't drop view eventview because other objects depend on it
HINT: Use DROP ... CASCADE to drop the dependent objects too.
```

Um dieses Problem zu lösen, führen Sie den folgenden Befehl aus (wie von der Fehlermeldung vorgeschlagen):

```
drop view eventview cascade;
```

Nun wurden sowohl eventview als auch myeventview erfolgreich entfernt.

Im folgenden Beispiel wird entweder die Ansicht eventview gelöscht, wenn sie vorhanden ist, oder es erfolgt keine Aktion, und es wird eine Meldung zurückgegeben, wenn die Ansicht nicht vorhanden ist:

```
drop view if exists eventview;
```

END

Führt einen Commit der aktuellen Transaktion aus. Dieser Befehl hat genau die gleiche Funktion wie der Befehl COMMIT.

Detailliertere Informationen finden Sie in COMMIT.

Syntax

```
END [ WORK | TRANSACTION ]
```

Parameter

WORK

Optionales Schlüsselwort.

TRANSACTION

Optionales Schlüsselwort; WORK und TRANSACTION sind Synonyme.

Beispiele

In den folgenden Beispielen werden der Transaktionsblock beendet und ein Commit für die Transaktion ausgeführt:

```
end;
```

```
end work;
```

```
end transaction;
```

Nach Ausführung jedes dieser Befehle beendet Amazon Redshift den Transaktionsblock und führt ein Commit für die Änderungen aus.

EXECUTE

Führt eine zuvor vorbereitete Anweisung aus.

Syntax

```
EXECUTE plan_name [ (parameter [, ...]) ]
```

Parameter

plan_name

Der Name der vorbereiteten Anweisung, die ausgeführt werden soll.

Parameter

Der tatsächliche Wert eines Parameters der vorbereiteten Anweisung. Es muss sich um einen Ausdruck handeln, der einen Wert mit einem Typ zum Ergebnis hat, der mit dem Datentyp kompatibel ist, der für diese Parameterposition im Befehl PREPARE, der die vorbereitete Anweisung erstellt hat, angegeben ist.

Nutzungshinweise

EXECUTE wird verwendet, um eine zuvor vorbereitete Anweisung auszuführen. Da vorbereitete Anweisungen nur für die Dauer einer Sitzung vorhanden sind, muss die vorbereitete Anweisung von einer PREPARE-Anweisung erstellt worden sein, die früher in der aktuellen Sitzung ausgeführt wurde.

Wenn die vorherige PREPARE-Anweisung Parameter angegeben hat, muss ein kompatibler Satz von Parametern an die EXECUTE-Anweisung übergeben werden. Andernfalls gibt Amazon Redshift einen Fehler zurück. Anders als bei Funktionen werden vorbereitete Anweisungen nicht auf der Basis des Typs oder der Zahl der angegebenen Parameter überladen. Der Name einer vorbereiteten Anweisung muss innerhalb einer Datenbanksitzung eindeutig sein.

Wenn ein EXECUTE-Befehl für die vorbereitete Anweisung ausgegeben wird, kann Amazon Redshift den Abfrageausführungsplan optional ändern (um die Leistung auf der Basis der angegebenen Parameterwerte zu verbessern), bevor die vorbereitete Anweisung ausgeführt wird. Für jede neue Ausführung einer vorbereiteten Anweisung kann Amazon Redshift den Abfrageausführungsplan basierend auf den unterschiedlichen Parameterwerten, die mit der EXECUTE-Anweisung angegeben werden, erneut ändern. Um den Abfrageausführungsplan zu überprüfen, den Amazon Redshift für bestimmte EXECUTE-Anweisungen ausgewählt hat, verwenden Sie den Befehl EXPLAIN.

Beispiele und weitere Informationen zur Erstellung und Nutzung vorbereiteter Anweisungen finden Sie unter PREPARE.

Weitere Informationen finden Sie auch unter

DEALLOCATE, PREPARE

EXPLAIN

Zeigt den Ausführungsplan für eine Abfrageanweisung an, ohne die Abfrage auszuführen. Informationen zum Arbeitsablauf für die Abfrageanalyse finden Sie unter Workflow zur Analyse von Abfragen.

Syntax

```
EXPLAIN [ VERBOSE ] query
```

Parameter

VERBOSE

Zeigt den vollständigen Abfrageplan und nicht nur eine Zusammenfassung an.

query

Die Abfrageanweisung, die erklärt werden soll. Bei der Abfrage kann es sich um eine SELECT-, INSERT-, CREATE TABLE AS-, UPDATE- oder DELETE-Anweisung handeln.

Nutzungshinweise

Die Leistung von EXPLAIN wird manchmal von der Zeit beeinflusst, die benötigt wird, um temporäre Tabellen zu erstellen. Beispielsweise müssen für eine Abfrage, die die allgemeine Unterausdruckoptimierung verwendet, temporäre Tabellen erstellt und analysiert werden, um die EXPLAIN-Ausgabe zurückgeben zu können. Der Abfrageplan ist vom Schema und der Statistik der temporären Tabellen abhängig. Daher benötigt der EXPLAIN-Befehl für diese Art von Abfrage möglicherweise länger für die Ausführung als erwartet.

Sie können EXPLAIN nur für die folgenden Befehle verwenden:

- SELECT
- SELECT INTO
- CREATE TABLE AS
- INSERT
- AKTUALISIERUNG
- DELETE

Der Befehl EXPLAIN ist nicht erfolgreich, wenn Sie ihn für andere SQL-Befehle verwenden, wie Data Definition Language (DDL)- oder Datenbankoperationen.

Die relativen Einheitskosten der EXPLAIN-Ausgabe werden von Amazon Redshift verwendet, um einen Abfrageplan auszuwählen. Amazon Redshift vergleicht die Größen verschiedener Ressourcenschätzungen, um den Plan zu ermitteln.

Abfrageplanung und Ausführungsschritte

Der Ausführungsplan für eine bestimmte Amazon-Redshift-Abfrageanweisung unterteilt Ausführung und Berechnung einer Abfrage in eine definierte Abfolge von Schritten und Tabellenoperationen, die schließlich einen endgültigen Ergebnissatz für die Abfrage zurückgeben. Informationen zur Abfrageplanung finden Sie unter Verarbeitung von Abfragen.

In der folgenden Tabelle wird eine Übersicht über die Schritte bereitgestellt, die Amazon Redshift für die Entwicklung eines Ausführungsplans für eine Abfrage verwenden kann, die von einem Benutzer zur Ausführung abgesendet wird.

EXPLAIN-Operatoren	Abfrageausführungsschritte	Beschreibung
SCAN:		
Sequenzieller Scan	scan	Amazon-Redshift-Beziehungs-Scan- oder Tabellen-Scan-Operator oder -Schritt. Scannt die gesamte Tabelle sequenziell von Anfang bis Ende und evaluiert darüber hinaus Abfrageeinschränkungen für jede Zeile (Filter), wenn mit der WHERE-Klausel angegeben. Wird auch für die Ausführung von INSERT-, UPDATE- und DELETE-Anweisungen verwendet.
JOINS: Amazon Redshift verwendet unterschiedliche Join-Operatoren, abhängig vom physischen Design der Tabellen, für die ein Join-Vorgang ausgeführt wird, vom Speicherort der Daten, die für den Join benötigt werden, und von den spezifischen Attributen der Abfrage selbst. Unterabfrage-Scan – Unterabfrage-Scans und Anfügungen werden verwendet, um UNION-Abfragen auszuführen.		

EXPLAIN-Operatoren	Abfrageausführungsschritte	Beschreibung
Nested Loop	nloop	Dies ist der am wenigsten optimale Join; wird vor allem für Kreuz-Joins (kartesische Produkte; ohne Join-Bedingung) und einige Ungleichheits-Joins verwendet.
Hash Join	hjoin	Wird auch für interne Joins sowie externe Joins nach links und rechts verwendet und in der Regel schneller ausgeführt als ein Join über eine verschachtelte Schleife. Hash-Joins lesen die externe Tabelle, führen einen Hash-Vorgang für die angeschlossene Spalte aus und suchen Übereinstimmungen in der internen Hash-Tabelle. Dieser Schritt kann auf die Festplatte übergreifen. (Der interne Input von hjoin ist ein Hash-Schritt, der festplattenbasiert sein kann.)
Merge Join	mjoin	Wird auch für interne und externe Joins verwendet (für Join-Tabellen, die anhand der Join-Spalten verteilt und sortiert werden). In der Regel ist dies der schnellste Join-Algorithmus von Amazon Redshift ohne Berücksichtigung anderer Kostenüberlegungen.
AGGREGATION: Operatoren und Schritte, die für Abfragen verwendet werden, die zusammengefasste Funktionen und GROUP BY-Vorgänge verwenden.		
Aggregate	aggr	Operator/Schritt für skalare Zusammenfassungsfunktionen.
HashAggregate	aggr	Operator/Schritt für gruppierte Zusammenfassungsfunktionen. Kann über die Festplatte ausgeführt werden, wenn die Hash-Tabelle auf die Festplatte übergreift.

EXPLAIN-Operatoren	Abfrageausführungsschritte	Beschreibung
GroupAggregate	aggr	Ein Operator, der manchmal für gruppierte Zusammenfassungsabfragen gewählt wird, wenn die Amazon-Redshift-Konfigurationseinstellung für die Einstellung force_hash_grouping nicht aktiviert ist.
SORT: Operatoren und Schritte, die verwendet werden, wenn Abfragen Ergebnissätze sortieren oder zusammenführen müssen.		
Sortierung	sort	„sort“ führt die von der ORDER BY-Klausel angegebene Sortierung sowie weitere Operationen wie UNION und Joins aus. Kann von der Festplatte aus ausgeführt werden.
Merge	merge	Erstellt die abschließenden sortierten Ergebnisse einer Abfrage auf der Basis zwischenzeitlicher sortierter Ergebnisse, die von parallel ausgeführten Operationen abgeleitet werden.
EXCEPT-, INTERSECT- und UNION-Operationen:		
SetOp Außer [Distinct]	hjoin	Wird für EXCEPT-Abfragen verwendet. Kann über die Festplatte ausgeführt werden, da der Eingabe-Hash festplattenbasiert sein kann.
Hash Intersect [Distinct]	hjoin	Wird für INTERSECT-Abfragen verwendet. Kann über die Festplatte ausgeführt werden, da der Eingabe-Hash festplattenbasiert sein kann.

EXPLAIN-Operatoren	Abfrageausführungsschritte	Beschreibung
Append [All |Distinct]	save	Eine Anfügung, die mit dem Unterabfrage-Scan ausgeführt wird, um UNION- und UNION ALL-Abfragen zu implementieren. Kann aufgrund von „save“ über die Festplatte ausgeführt werden.
Verschiedenes/Sonstiges:		
Hash	hash	Wird für interne Joins sowie externe Joins nach links und rechts verwendet (stellt Eingaben für einen Hash-Join bereit). Der Hash-Operator erstellt die Hash-Tabelle für die interne Tabelle eines Join. (Die interne Tabelle ist die Tabelle, die auf Übereinstimmungen überprüft wird und im Fall eines Join zweier Tabellen in der Regel die kleinere der beiden Tabellen ist.)
Limit	limit	Evaluiert die Limit-Klausel.
Materialize	save	Setzt Zeilen für die Eingabe für Joins mit verschachtelten Schleifen und einige Zusammenführungs-Joins um. Kann von der Festplatte aus ausgeführt werden.
--	parse	Wird verwendet, um während des Ladens Eingabedaten in Textform zu analysieren.
--	project	Wird verwendet, um Spalten und Datenverarbeitungsausdrücke, d. h. Projektdaten, neu anzuordnen.
Ergebnis	--	Führt skalare Funktionen aus, für die kein Tabellenzugriff erforderlich ist.
--	return	Gibt Zeilen an den Leader oder Client zurück.

EXPLAIN-Operatoren	Abfrageausführungsschritte	Beschreibung
Subplan	--	Wird für bestimmte Unterabfragen verwendet.
Unique	eindeutig	Beseitigt Duplikate aus SELECT DISTINCT- und UNION-Abfragen.
Window	window	Datenverarbeitung für Aggregation und Einstufung von Fensterfunktionen. Kann von der Festplatte aus ausgeführt werden.
Netzwerkoperationen:		
Network (Broadcast)	bcast	Broadcast ist auch ein Attribut von Join Explain-Operatoren und -Schritten.
Network (Distribute)	dist	Verteilung von Zeilen an Datenverarbeitungsknoten für die parallele Verarbeitung durch ein Data Warehouse-Cluster.
Network (Send to Leader)	return	Sendet die Ergebnisse zur weiteren Verarbeitung an den Leader zurück.
DML-Operationen (Operatoren, die Daten ändern):		
Insert (unter Verwendung des Ergebnisses)	insert	Fügt Daten ein.
Delete (Scan + Filter)	delete	Löscht Daten. Kann von der Festplatte aus ausgeführt werden.
Update (Scan + Filter)	delete, insert	Wird als Lösch- und Einfügevorgang implementiert.

Verwendung von EXPLAIN für RLS

Wenn eine Abfrage eine Tabelle enthält, die RLS-Richtlinien (Row Level Security) unterliegt, zeigt EXPLAIN einen speziellen RLS-Knoten an. SecureScan Amazon Redshift protokolliert denselben

Knotentyp auch in der STL_EXPLAIN-Systemtabelle. EXPLAIN enthüllt nicht das RLS-Prädikat, das für dim_tbl gilt. Der SecureScan RLS-Knotentyp dient als Indikator dafür, dass der Ausführungsplan zusätzliche Operationen enthält, die für den aktuellen Benutzer unsichtbar sind.

Das folgende Beispiel zeigt einen RLS-Knoten SecureScan .

```
EXPLAIN
SELECT D.cint
FROM fact_tbl F INNER JOIN dim_tbl D ON F.k_dim = D.k
WHERE F.k_dim / 10 > 0;
                              QUERY PLAN
------------------------------------------------------------------------
 XN Hash Join DS_DIST_ALL_NONE  (cost=0.08..0.25 rows=1 width=4)
   Hash Cond: ("outer".k_dim = "inner"."k")
   ->  *XN* *RLS SecureScan f  (cost=0.00..0.14 rows=2 width=4)*
         Filter: ((k_dim / 10) > 0)
   ->  XN Hash  (cost=0.07..0.07 rows=2 width=8)
         ->  XN Seq Scan on dim_tbl d  (cost=0.00..0.07 rows=2 width=8)
               Filter: (("k" / 10) > 0)
```

Um eine vollständige Untersuchung von Abfrageplänen zu ermöglichen, die RLS unterliegen, bietet Amazon Redshift die EXPLAIN-RLS-Systemberechtigungen an. Benutzer, denen diese Berechtigung erteilt wurde, können vollständige Abfragepläne prüfen, die auch RLS-Prädikate enthalten.

Das folgende Beispiel zeigt, dass ein zusätzlicher Seq-Scan unterhalb des SecureScan RLS-Knotens auch das RLS-Richtlinienprädikat (k_dim > 1) beinhaltet.

```
EXPLAIN SELECT D.cint
FROM fact_tbl F INNER JOIN dim_tbl D ON F.k_dim = D.k
WHERE F.k_dim / 10 > 0;
                                  QUERY PLAN
------------------------------------------------------------------------------
 XN Hash Join DS_DIST_ALL_NONE  (cost=0.08..0.25 rows=1 width=4)
   Hash Cond: ("outer".k_dim = "inner"."k")
   *->  XN RLS SecureScan f  (cost=0.00..0.14 rows=2 width=4)
         Filter: ((k_dim / 10) > 0)*
         ->  *XN* *Seq Scan on fact_tbl rls_table  (cost=0.00..0.06 rows=5 width=8)
               Filter: (k_dim > 1)*
   ->  XN Hash  (cost=0.07..0.07 rows=2 width=8)
         ->  XN Seq Scan on dim_tbl d  (cost=0.00..0.07 rows=2 width=8)
               Filter: (("k" / 10) > 0)
```

Während die Berechtigung EXPLAIN RLS einem Benutzer erteilt wird, protokolliert Amazon Redshift den vollständigen Abfrageplan einschließlich RLS-Prädikate in der STL_EXPLAIN-Systemtabelle. Abfragen, die ausgeführt werden, während diese Berechtigung nicht erteilt ist, werden ohne RLS-Interna protokolliert. Das Erteilen oder Entfernen der EXPLAIN RLS-Berechtigung ändert nichts daran, was Amazon Redshift in STL_EXPLAIN für frühere Abfragen protokolliert hat.

AWS Lake Formation-RLS-geschützte Redshift-Relationen

Das folgende Beispiel zeigt einen SecureScan LF-Knoten, mit dem Sie die Beziehungen zwischen Lake Formation und RLS anzeigen können.

```
EXPLAIN
SELECT *
FROM lf_db.public.t_share
WHERE a > 1;
QUERY PLAN
------------------------------------------------------------
XN LF SecureScan t_share  (cost=0.00..0.02 rows=2 width=11)
(2 rows)
```

Beispiele

> **Note**
>
> Für diese Beispiele kann sich die Beispielausgabe abhängig von der Amazon-Redshift-Konfiguration unterscheiden.

Im folgenden Beispiel wird der Abfrageplan für eine Abfrage zurückgegeben, die EVENTID, EVENTNAME, VENUEID und VENUENAME aus den Tabellen EVENT und VENUE auswählt:

```
explain
select eventid, eventname, event.venueid, venuename
from event, venue
where event.venueid = venue.venueid;
```

```
                                QUERY PLAN
------------------------------------------------------------------------------
XN Hash Join DS_DIST_OUTER  (cost=2.52..58653620.93 rows=8712 width=43)
```

```
Hash Cond: ("outer".venueid = "inner".venueid)
->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=23)
->  XN Hash  (cost=2.02..2.02 rows=202 width=22)
->  XN Seq Scan on venue  (cost=0.00..2.02 rows=202 width=22)
(5 rows)
```

Im folgenden Beispiel wird der Abfrageplan für dieselbe Abfrage mit Verbose-Ausgabe zurückgegeben:

```
explain verbose
select eventid, eventname, event.venueid, venuename
from event, venue
where event.venueid = venue.venueid;
```

```
                                QUERY PLAN
------------------------------------------------------------------------------
{HASHJOIN
:startup_cost 2.52
:total_cost 58653620.93
:plan_rows 8712
:plan_width 43
:best_pathkeys <>
:dist_info DS_DIST_OUTER
:dist_info.dist_keys (
TARGETENTRY
{
VAR
:varno 2
:varattno 1
...

XN Hash Join DS_DIST_OUTER  (cost=2.52..58653620.93 rows=8712 width=43)
Hash Cond: ("outer".venueid = "inner".venueid)
->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=23)
->  XN Hash  (cost=2.02..2.02 rows=202 width=22)
->  XN Seq Scan on venue  (cost=0.00..2.02 rows=202 width=22)
(519 rows)
```

Im folgenden Beispiel wird der Abfrageplan für eine CREATE TABLE AS (CTAS)-Anweisung zurückgegeben:

```
explain create table venue_nonulls as
```

```
select * from venue
where venueseats is not null;

QUERY PLAN
--------------------------------------------------------------
XN Seq Scan on venue  (cost=0.00..2.02 rows=187 width=45)
Filter: (venueseats IS NOT NULL)
(2 rows)
```

FETCH

Ruft Zeilen mittels eines Cursors ab. Weitere Informationen zum Deklarieren eines Cursors finden Sie unter DECLARE.

FETCH ruft Zeilen auf der Basis der aktuellen Position innerhalb des Cursors ab. Wenn ein Cursor erstellt wird, wird er vor der ersten Zeile positioniert. Nach einem FETCH befindet sich der Cursor in der letzten abgerufenen Zeile. Wenn FETCH das Ende der verfügbaren Zeilen erreicht, beispielsweise infolge eines FETCH ALL, befindet sich der Cursor hinter der letzten Zeile.

FORWARD 0 ruft die aktuelle Zeile ab, ohne den Cursor zu verschieben, d. h., sie ruft die zuletzt abgerufene Zeile ab. Wenn sich der Cursor vor der ersten oder nach der letzten Zeile befindet, wird keine Zeile zurückgegeben.

Wenn die erste Zeile eines Cursors abgerufen wird, wird der gesamte Ergebnissatz auf dem Führungsknoten im Arbeitsspeicher oder auf der Festplatte umgesetzt, wenn nötig. Aufgrund der potenziellen negativen Auswirkungen der Verwendung von Cursors mit großen Ergebnissätzen wird empfohlen, alternative Ansätze anzuwenden, wann immer möglich. Weitere Informationen finden Sie unter Überlegungen in Bezug auf die Leistung bei Verwendung von Cursors.

Weitere Informationen finden Sie unter DECLARE, CLOSE.

Syntax

```
FETCH [ NEXT | ALL | {FORWARD [ count | ALL ] } ] FROM cursor
```

Parameter

NEXT

Ruft die nächste Zeile ab. Dies ist die Standardeinstellung.

ALL

Ruft alle verbleibenden Zeilen ab. (Verhält sich wie FORWARD ALL.) ALL wird nicht für Einzelknoten-Cluster unterstützt.

FORWARD [count | ALL]

Holt die nächste Zahl von Zeilen oder alle erbleibenden Zeilen. `FORWARD 0` holt die aktuelle Zeile. Im Fall von Einzelknoten-Clustern ist der maximal zulässige Wert für „count“ `1000`. FORWARD ALL wird nicht für Einzelknoten-Cluster unterstützt.

cursor

Der Name des neuen Cursors.

Beispiel für FETCH

Im folgenden Beispiel wird ein Cursor namens LOLLAPALOOZA deklariert, um Verkaufsinformationen für das Lollapalooza-Ereignis auszuwählen und dann mittels des Cursors Zeilen aus dem Ergebnissatz abzurufen:

```
-- Begin a transaction

begin;

-- Declare a cursor

declare lollapalooza cursor for
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Lollapalooza';

-- Fetch the first 5 rows in the cursor lollapalooza:

fetch forward 5 from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-05-01 19:00:00 |   92.00000000 |       3
 Lollapalooza | 2008-11-15 15:00:00 |  222.00000000 |       2
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       3
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       4
```

```
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       1
(5 rows)

-- Fetch the next row:

fetch next from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-10-06 14:00:00 |  114.00000000 |       2

-- Close the cursor and end the transaction:

close lollapalooza;
commit;
```

GRANT

Definiert die Zugriffsberechtigungen für einen Benutzer oder eine Rolle.

Zu den Berechtigungen gehören Zugriffsoptionen wie die Möglichkeit zum Lesen von Daten in Tabellen und Ansichten, zum Schreiben von Daten, zum Erstellen von Tabellen sowie zum Entfernen von Tabellen. Verwenden Sie diesen Befehl, um bestimmte Berechtigungen für eine Tabelle, eine Datenbank, ein Schema, eine Funktion, eine Prozedur, eine Sprache oder eine Spalte zu erteilen. Um Berechtigungen für ein Datenbankobjekt zu widerrufen, verwenden Sie den Befehl REVOKE.

Zu den Berechtigungen gehören auch die folgenden Zugriffsoptionen für Datashare-Produzenten:

- Gewährung von Datashare-Zugriff auf Konsumenten-Namespaces und -Konten.
- Gewähren der Berechtigung zum Ändern eines Datashares durch Hinzufügen oder Entfernen von Objekten aus dem Datashare.
- Gewähren der Berechtigung zum Freigeben eines Datashares durch Hinzufügen oder Entfernen von Konsumenten-Namespaces aus dem Datashare.

Die Optionen für den Datashare-Zugriff für Konsumenten sind wie folgt:

- Gewähren des vollen Zugriffs für Benutzer auf Datenbanken, die aus einem Datashare erstellt wurden, oder auf externe Schemata, die auf solche Datenbanken verweisen.
- Gewähren von Berechtigungen auf Objektebene für Benutzer für Datenbanken, die aus einem Datashare erstellt wurden, wie dies für lokale Datenbankobjekte möglich ist. Um diese

Berechtigungsebene zu gewähren, müssen Sie die WITH PERMISSIONS-Klausel verwenden, wenn Sie eine Datenbank aus dem Datashare erstellen. Weitere Informationen finden Sie unter CREATE DATABASE.

Weitere Informationen zu Datashare-Berechtigungen finden Sie unter Freigeben von Datashares.

Sie können auch Rollen zuweisen, um die Datenbankberechtigungen zu verwalten und zu kontrollieren, welche Aktionen Benutzer in Bezug auf Ihre Daten durchführen können. Durch die Definition von Rollen und die Zuweisung von Rollen für Benutzer können Sie die Aktionen beschränken, die diese Benutzer ausführen können. So können Sie beispielsweise die Aktionen der Benutzer auf die Befehle CREATE TABLE und INSERT beschränken. Weitere Informationen zum Befehl CREATE ROLE finden Sie unter the section called “CREATE ROLE”. In Amazon Redshift gibt es systemdefinierte Rollen, die Sie ebenfalls verwenden können, um Ihren Benutzern bestimmte Berechtigungen zu erteilen. Weitere Informationen finden Sie unter the section called “Systemdefinierte Amazon-Redshift-Rollen”.

Sie können Nutzungsberechtigungen in einem externen Schema nur Datenbankbenutzern und Benutzergruppen gewähren oder entziehen, die die ON SCHEMA-Syntax verwenden. Wenn Sie ON EXTERNAL SCHEMA with verwenden AWS Lake Formation, können Sie nur die Berechtigungen GRANT und REVOKE für eine AWS Identity and Access Management (IAM-) Rolle gewähren. Eine Liste der Berechtigungen finden Sie in der Syntax.

Für gespeicherte Prozeduren kann nur die Berechtigung EXECUTE erteilt werden.

GRANT (auf einer externen Ressource) kann nicht innerhalb eines Transaktionsblocks (BEGIN ... END) ausgeführt werden. Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Wenn Sie sehen möchten, welche Berechtigungen Benutzern für eine Datenbank erteilt wurden, verwenden Sie HAS_DATABASE_PRIVILEGE. Wenn Sie sehen möchten, welche Berechtigungen Benutzern für ein Schema erteilt wurden, verwenden Sie HAS_SCHEMA_PRIVILEGE. Wenn Sie sehen möchten, welche Berechtigungen Benutzern für eine Tabelle erteilt wurden, verwenden Sie HAS_TABLE_PRIVILEGE.

Syntax

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | ALTER | TRUNCATE }
 [,...] | ALL [ PRIVILEGES ] }
```

```
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { CREATE | TEMPORARY | TEMP | ALTER } [,...] | ALL [ PRIVILEGES ] }
    ON DATABASE db_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { CREATE | USAGE | ALTER } [,...] | ALL [ PRIVILEGES ] }
    ON SCHEMA schema_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 FUNCTIONS IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 PROCEDURES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT USAGE
    ON LANGUAGE language_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Erteilen von Berechtigungen auf Spaltenebene für Tabellen

Im Folgenden finden Sie die Syntax für Berechtigungen auf Spaltenebene für Amazon-Redshift-Tabellen und -Ansichten.

```
GRANT { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
     ON { [ TABLE ] table_name [, ...] }

     TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Erteilen von Berechtigungen ASSUMEROLE

Im Folgenden finden Sie die Syntax für die Berechtigungen ASSUMEROLE, die Benutzern und Gruppen mit einer spezifizierten Rolle erteilt werden. Informationen zur Verwendung der ASSUMEROLE-Berechtigung finden Sie unter Hinweise zur Erteilung der Berechtigung ASSUMEROLE.

```
GRANT ASSUMEROLE
       ON { 'iam_role' [, ...] | default | ALL }
       TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
       FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]
```

Erteilen von Berechtigungen für die Integration von Redshift Spectrum in Lake Formation

Im Folgenden finden Sie die Syntax für die Integration von Redshift Spectrum in Lake Formation.

```
GRANT { SELECT | ALL [ PRIVILEGES ] } ( column_list )
    ON EXTERNAL TABLE schema_name.table_name
    TO { IAM_ROLE iam_role } [, ...] [ WITH GRANT OPTION ]

GRANT { { SELECT | ALTER | DROP | DELETE | INSERT }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL TABLE schema_name.table_name [, ...]
    TO { { IAM_ROLE iam_role } [, ...] | PUBLIC } [ WITH GRANT OPTION ]

GRANT { { CREATE | ALTER | DROP }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL SCHEMA schema_name [, ...]
    TO { IAM_ROLE iam_role } [, ...] [ WITH GRANT OPTION ]
```

Erteilen von Datashare-Berechtigungen

Produzentenseitige Datashare-Berechtigungen

Im Folgenden finden Sie die Syntax zur Verwendung von GRANT zum Gewähren von ALTER- oder SHARE-Berechtigungen für einen Benutzer oder eine Rolle. Der Benutzer kann den Datashare mit der ALTER-Berechtigung ändern oder einem Konsumenten mit der SHARE-Berechtigung die Nutzung gestatten. ALTER und SHARE sind die einzigen Berechtigungen, die Sie Benutzern und Rollen für Datashares erteilen können.

```
GRANT { ALTER | SHARE } ON DATASHARE datashare_name
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Im Folgenden finden Sie die Syntax für die Verwendung von GRANT für Datashare-Nutzungsberechtigungen in Amazon Redshift. Mit der Berechtigung USAGE gewähren Sie einem Konsumenten Zugriff auf ein Datashare. Sie können diese Berechtigung nicht an Benutzer oder Benutzergruppen erteilen. Diese Berechtigung unterstützt auch nicht die WITH GRANT OPTION für die GRANT-Anweisung. Nur Benutzer oder Benutzergruppen, denen zuvor die Berechtigung SHARE für das Datashare erteilt wurde, können diese Art von GRANT-Anweisung ausführen.

```
GRANT USAGE
    ON DATASHARE datashare_name
    TO NAMESPACE 'namespaceGUID' | ACCOUNT 'accountnumber' [ VIA DATA CATALOG ]
```

Im Folgenden finden Sie ein Beispiel dafür, wie Sie einem Lake-Formation-Konto die Berechtigung zur Nutzung eines Datashares gewähren können.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012' VIA DATA CATALOG;
```

Konsumentenseitige Datashare-Berechtigungen

Im Folgenden finden Sie die Syntax für die GRANT-Datashare-Nutzungsberechtigungen für eine bestimmte Datenbank oder ein bestimmtes Schema, die auf einem Datashare erstellt wurden.

Weitere Berechtigungen, die Konsumenten für den Zugriff auf eine aus einem Datashare erstellte Datenbank benötigen, hängen davon ab, ob der CREATE DATABASE-Befehl, mit dem die Datenbank aus dem Datashare erstellt wurde, die WITH PERMISSIONS-Klausel verwendet hat oder nicht. Weitere Informationen über den CREATE DATABASE-Befehl und die WITH PERMISSIONS-Klausel finden Sie unter CREATE DATABASE.

Ohne die WITH PERMISSIONS-Klausel erstellte Datenbanken

Wenn Sie die USAGE-Berechtigung für eine Datenbank gewähren, die aus einem Datashare ohne die WITH PERMISSIONS-Klausel erstellt wurde, müssen Sie Berechtigungen für die Objekte in der gemeinsam genutzten Datenbank nicht separat gewähren. Entitäten, denen die Verwendung für Datenbanken gewährt wurde, die aus Datashares ohne die WITH PERMISSIONS-Klausel erstellt wurden, haben automatisch Zugriff auf alle Objekte in der Datenbank.

Mit der WITH PERMISSIONS-Klausel erstellte Datenbanken

Wenn Sie die USAGE-Berechtigung für eine Datenbank gewähren, wobei die gemeinsam genutzte Datenbank mit der WITH PERMISSIONS-Klausel aus einem Datashare erstellt wurde, müssen konsumentenseitigen Identitäten dennoch die entsprechenden Berechtigungen für Datenbankobjekte

in der gemeinsam genutzten Datenbank erteilt werden, um auf sie zugreifen zu können, genauso wie Sie Berechtigungen für lokale Datenbankobjekte gewähren würden. Verwenden Sie die dreiteilige Syntax `database_name.schema_name.object_name`, um Objekten in einer Datenbank, die aus einem Datashare erstellt wurde, Berechtigungen zu gewähren. Verwenden Sie die zweiteilige Syntax `schema_name.object_name`, um Objekten in einem externen Schema, das auf ein gemeinsam genutztes Schema innerhalb der gemeinsam genutzten Datenbank verweist, Berechtigungen zu gewähren.

```
GRANT USAGE ON { DATABASE shared_database_name [, ...] | SCHEMA shared_schema}
    TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Gewähren von bereichsbezogenen Berechtigungen

Mit bereichsbezogenen Berechtigungen können Sie einem Benutzer oder einer Rolle Berechtigungen für alle Objekte eines Typs innerhalb einer Datenbank oder eines Schemas gewähren. Benutzer und Rollen mit bereichsbezogenen Berechtigungen verfügen über die angegebenen Berechtigungen für alle aktuellen und future Objekte innerhalb der Datenbank oder des Schemas.

Im Folgenden sehen Sie die Syntax zum Erteilen bereichsbezogener Berechtigungen für Benutzer oder Rollen. Weitere Hinweise zu bereichsbezogenen Berechtigungen finden Sie unter. Bereichsbeschränkte Berechtigungen

```
GRANT { CREATE | USAGE | ALTER } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE db_name
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

GRANT
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }
FOR TABLES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name} [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name | } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
FOR PROCEDURES IN
```

```
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name | } [, ...]

GRANT USAGE
FOR LANGUAGES IN
{DATABASE db_name}
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]
```

Beachten Sie, dass bei bereichsbezogenen Berechtigungen nicht zwischen Berechtigungen für Funktionen und für Prozeduren unterschieden wird. Die folgende Anweisung gewährt beispielsweise bob die EXECUTE Berechtigung sowohl für Funktionen als auch für Prozeduren im Schema. `Sales_schema`

```
GRANT EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema TO bob;
```

Erteilen von Machine-Learning-Berechtigungen

Im Folgenden finden Sie die Syntax für Berechtigungen für Machine-Learning-Modelle in Amazon Redshift.

```
GRANT CREATE MODEL
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON MODEL model_name [, ...]

    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Erteilen von Rollenberechtigungen

Im Folgenden finden Sie die Syntax zum Erteilen von Rollenberechtigungen in Amazon Redshift.

```
GRANT { ROLE role_name } [, ...] TO { { user_name [ WITH ADMIN OPTION ] } |
 ROLE role_name }[, ...]
```

Im Folgenden finden Sie die Syntax zum Erteilen von Systemberechtigungen für Rollen in Amazon Redshift.

```
GRANT
```

```
  {
    { CREATE USER | DROP USER | ALTER USER |
    CREATE SCHEMA | DROP SCHEMA |
    ALTER DEFAULT PRIVILEGES |
    ACCESS CATALOG |
    CREATE TABLE | DROP TABLE | ALTER TABLE |
    CREATE OR REPLACE FUNCTION | CREATE OR REPLACE EXTERNAL FUNCTION |
    DROP FUNCTION |
    CREATE OR REPLACE PROCEDURE | DROP PROCEDURE |
    CREATE OR REPLACE VIEW | DROP VIEW |
    CREATE MODEL | DROP MODEL |
    CREATE DATASHARE | ALTER DATASHARE | DROP DATASHARE |
    CREATE LIBRARY | DROP LIBRARY |
    CREATE ROLE | DROP ROLE |
    TRUNCATE TABLE
    VACUUM | ANALYZE | CANCEL }[, ...]
  }
  | { ALL [ PRIVILEGES ] }
TO { ROLE role_name } [, ...]
```

Erteilen von Explain-Berechtigungen für RLS-Richtlinienfilter

Im Folgenden finden Sie die Syntax zum Erteilen von Berechtigungen zum Erklären der RLS-Richtlinienfilter auf Zeilenebene einer Abfrage im EXPLAIN-Plan. Sie können die Berechtigung mit der REVOKE-Anweisung entziehen.

```
GRANT EXPLAIN RLS TO ROLE rolename
```

Im Folgenden finden Sie die Syntax zum Erteilen von Berechtigungen zum Umgehen von RLS-Richtlinien auf Zeilenebene für eine Abfrage.

```
GRANT IGNORE RLS TO ROLE rolename
```

Gewähren von Berechtigungen für RLS-Suchtabellen an ein Richtlinienobjekt

Im Folgenden finden Sie die Syntax zum Erteilen von Berechtigungen für die angegebene RLS-Richtlinie auf Zeilenebene.

```
GRANT SELECT ON [ TABLE ] table_name [, ...]
TO RLS POLICY policy_name [, ...]
```

Parameter

SELECT

Erteilt die Berechtigung, Daten aus einer Tabelle oder Ansicht mittels einer SELECT-Anweisung auszuwählen. Die Berechtigung SELECT ist auch erforderlich, um vorhandene Spaltenwerte für UPDATE- oder DELETE-Operationen zu referenzieren.

INSERT

Erteilt die Berechtigung, Daten in eine Tabelle mittels einer INSERT- oder COPY-Anweisung zu laden.

UPDATE

Erteilt die Berechtigung, eine Tabellenspalte mittels einer UPDATE-Anweisung zu aktualisieren. UPDATE-Operationen erfordern ebenfalls die Berechtigung SELECT, da sie Tabellenspalten referenzieren müssen, um zu ermitteln, welche Zeilen aktualisiert werden sollen, oder um neue Werte für Spalten zu berechnen.

DELETE

Erteilt die Berechtigung, eine Datenzeile aus einer Tabelle zu löschen. DELETE-Operationen erfordern ebenfalls die Berechtigung SELECT, da sie Tabellenspalten referenzieren müssen, um zu ermitteln, welche Zeilen gelöscht werden sollen.

DROP

Erteilt die Berechtigung zum Entfernen einer Tabelle Diese Berechtigung gilt in Amazon Redshift und in einer AWS Glue Data Catalog , die für Lake Formation aktiviert ist.

REFERENCES

Erteilt die Berechtigung, eine Fremdschlüsseleinschränkung zu erstellen. Sie müssen diese Berechtigung sowohl für die referenzierte als auch für die referenzierende Tabelle erteilen. Andernfalls kann der Benutzer die Einschränkung nicht erstellen.

ALTER

Erteilt dem Benutzer oder der Benutzergruppe die folgenden Berechtigungen, abhängig vom Datenbankobjekt:

- Für Tabellen erteilt ALTER die Berechtigung, eine Tabelle oder Ansicht zu ändern. Weitere Informationen finden Sie unter ALTER TABLE.

- Für Datenbanken erteilt ALTER die Berechtigung, eine Datenbank zu ändern. Weitere Informationen finden Sie unter ALTER DATABASE.
- Für Schemata erteilt ALTER die Berechtigung, ein Schema zu ändern. Weitere Informationen finden Sie unter ALTER SCHEMA.
- Für externe Tabellen erteilt ALTER die Erlaubnis, eine Tabelle in einer Tabelle zu ändern AWS Glue Data Catalog , die für Lake Formation aktiviert ist. Diese Berechtigung gilt nur bei Verwendung von Lake Formation.

TRUNCATE

Gewährt die Berechtigung zum Kürzen einer Tabelle. Ohne diese Berechtigung kann nur der Eigentümer einer Tabelle oder ein Superuser eine Tabelle kürzen. Weitere Informationen zur Verwendung des TRUNCATE-Befehls finden Sie unter the section called “TRUNCATE”.

ALL [PRIVILEGES]

Erteilt dem angegebenen Benutzer oder der angegebenen Benutzergruppe alle verfügbaren Berechtigungen auf einmal. Das Schlüsselwort PRIVILEGES ist optional.

GRANT ALL ON SCHEMA erteilt keine Berechtigungen CREATE für externe Schemata.

Sie können einer Tabelle in einer, die für Lake Formation aktiviert ist AWS Glue Data Catalog , die ALL-Berechtigung erteilen. In diesem Fall werden die einzelnen Berechtigungen (wie SELECT, ALTER usw.) im Datenkatalog aufgezeichnet.

ASSUMEROLE

Erteilt Benutzern, Rollen oder Gruppen mit einer angegebenen Rolle die Berechtigung zum Ausführen der Befehle COPY, UNLOAD, EXTERNAL FUNCTION und CREATE MODEL. Der Benutzer, die Rolle oder die Gruppe übernimmt diese Rolle beim Ausführen des jeweiligen Befehls. Informationen zur Verwendung der Berechtigung ASSUMEROLE finden Sie unter Hinweise zur Erteilung der Berechtigung ASSUMEROLE.

ON [TABLE] table_name

Erteilt die angegebenen Berechtigungen für eine Tabelle oder Ansicht. Das Schlüsselwort TABLE ist optional. Sie können in einer einzelnen Anweisung mehrere Tabellen und Ansichten auflisten.

ON ALL TABLES IN SCHEMA schema_name

Erteilt die angegebenen Berechtigungen für alle Tabellen und Ansichten im referenzierten Schema.

(column_name [,...]) ON TABLE table_name

Erteilt Benutzern, Gruppen oder PUBLIC die angegebenen Berechtigungen für die angegebenen Spalten der Amazon-Redshift-Tabelle oder -Ansicht.

(column_list) ON EXTERNAL TABLE schema_name.table_name

Erteilt einer IAM-Rolle die angegebenen Berechtigungen für die angegebenen Spalten der Lake-Formation-Tabelle im referenzierten Schema.

ON EXTERNAL TABLE schema_name.table_name

Erteilt einer IAM-Rolle die angegebenen Berechtigungen für die angegebenen Lake-Formation-Tabellen im referenzierten Schema.

ON EXTERNAL SCHEMA schema_name

Erteilt einer IAM-Rolle die angegebenen Berechtigungen für das referenzierte Schema.

ON iam_role

Erteilt einer IAM-Rolle die angegebenen Berechtigungen.

TO username

Gibt den Benutzer an, der die Berechtigungen erhält.

TO IAM_ROLE iam_role

Gibt die IAM-Rolle an, die die Berechtigungen erhält.

WITH GRANT OPTION

Gibt an, dass der Benutzer, der die Berechtigungen erhält, anderen Benutzern dieselben Berechtigungen gewähren kann. Die Berechtigung WITH GRANT OPTION kann keiner Gruppe oder PUBLIC gewährt werden.

ROLE rollen_name

Erteilt die Berechtigungen einer Rolle.

GROUP group_name

Erteilt die Berechtigungen einer Benutzergruppe. Dies kann eine durch Kommata getrennte Liste sein, wenn mehrere Benutzergruppen angegeben werden.

PUBLIC

Erteilt allen Benutzern die angegebenen Berechtigungen, einschließlich Benutzern, die später erstellt werden. PUBLIC stellt eine Gruppe dar, die stets alle Benutzer enthält. Die Berechtigungen eines einzelnen Benutzers sind die Summe der Berechtigungen, die PUBLIC gewährt werden, der Berechtigungen, die Gruppen gewährt werden, zu denen der Benutzer gehört, und der Berechtigungen, die dem einzelnen Benutzer gewährt werden.

Wenn Sie die Berechtigung PUBLIC einer EXTERNAL TABLE von Lake Formation gewähren, wird die Berechtigung der Lake-Formation-Gruppe everyone erteilt.

CREATE

Erteilt dem Benutzer oder der Benutzergruppe die folgenden Berechtigungen, abhängig vom Datenbankobjekt:

- Im Fall von Datenbanken ermöglicht CREATE Benutzern das Erstellen von Schemata innerhalb der Datenbank.
- Im Fall von Schemata ermöglicht CREATE Benutzern das Erstellen von Objekten innerhalb eines Schemas. Um ein Objekt umzubenennen, muss der Benutzer über die Berechtigung CREATE verfügen und Eigentümer des Objekts sein, das umbenannt werden soll.
- CREATE ON SCHEMA wird für externe Amazon-Redshift-Spectrum-Schematas nicht unterstützt. Um Rechte zur Nutzung externer Tabellen in einem externen Schema zu gewähren, müssen Sie den Benutzern, die diesen Zugriff benötigen, USAGE ON SCHEMA gewähren. Nur der Eigentümer eines externen Schemas oder ein Superuser darf externe Tabellen im externen Schema erstellen. Mit dem Befehl ALTER SCHEMA können Sie den Besitzer eines externen Schemas ändern.

TEMPORARY | TEMP

Erteilt die Berechtigung, in der angegebenen Datenbank temporäre Tabellen zu erstellen. Um Amazon Redshift Spectrum-Abfragen auszuführen, benötigt der Datenbankbenutzer die Berechtigung, temporäre Tabellen in der Datenbank zu erstellen.

> **Note**
>
> Standardmäßig wird Benutzern aufgrund ihrer automatischen Mitgliedschaft in der Gruppe PUBLIC die Berechtigung gewährt, temporäre Tabellen zu erstellen. Um die Berechtigung zum Erstellen temporärer Tabellen für alle Benutzer zu entfernen, entziehen Sie der

Gruppe PUBLIC die Berechtigung TEMP. Erteilen Sie dann explizit die Berechtigung, temporäre Tabellen für bestimmte Benutzer oder Benutzergruppen zu erstellen.

ON DATABASE db_name

Erteilt die angegebenen Berechtigungen für eine Datenbank.

USAGE

Erteilt die Berechtigung USAGE für ein bestimmtes Schema, um Objekte in diesem Schema für Benutzer zugänglich zu machen. Spezifische Aktionen für diese Objekte müssen für lokale Amazon-Redshift-Schemata getrennt gewährt werden (z. B. Berechtigung SELECT oder UPDATE für Tabellen). Standardmäßig besitzen alle Benutzer die Berechtigungen CREATE und USAGE für das Schema PUBLIC.

Wenn Sie externen Schemas mithilfe der ON SCHEMA-Syntax USAGE-Berechtigung zuweisen, müssen Sie Aktionen für die Objekte in dem externen Schema nicht separat gewähren. Die entsprechenden Katalogberechtigungen steuern differenzierte Berechtigungen für die externen Schemaobjekte.

ON SCHEMA schema_name

Erteilt die angegebenen Berechtigungen für ein Schema.

GRANT CREATE ON SCHEMA und die Berechtigung CREATE in GRANT ALL ON SCHEMA werden für externe Amazon-Redshift-Spectrum-Schemata nicht unterstützt. Um Rechte zur Nutzung externer Tabellen in einem externen Schema zu gewähren, müssen Sie den Benutzern, die diesen Zugriff benötigen, USAGE ON SCHEMA gewähren. Nur der Eigentümer eines externen Schemas oder ein Superuser darf externe Tabellen im externen Schema erstellen. Mit dem Befehl ALTER SCHEMA können Sie den Besitzer eines externen Schemas ändern.

EXECUTE ON ALL FUNCTIONS IN SCHEMA schema_name

Erteilt die angegebenen Berechtigungen für alle Funktionen im referenzierten Schema.

Amazon Redshift unterstützt keine Anweisungen für GRANT (erteilen) oder REVOKE (widerrufen) für integrierte pg_proc-Einträge, die im Namespace pg_catalog definiert sind.

EXECUTE ON PROCEDURE procedure_name

Erteilt die Berechtigung EXECUTE für eine spezifische gespeicherte Prozedur. Da die Namen gespeicherter Prozeduren überladen sein können, müssen Sie die Argumentliste für die Prozedur angeben. Weitere Informationen finden Sie unter Benennen von gespeicherten Prozeduren.

EXECUTE ON ALL PROCEDURES IN SCHEMA schema_name

Erteilt die angegebenen Berechtigungen für alle gespeicherten Prozeduren im referenzierten Schema.

USAGE ON LANGUAGE language_name

Erteilt die Berechtigung USAGE für eine Sprache.

Die Berechtigung USAGE ON LANGUAGE ist erforderlich, um benutzerdefinierte Funktionen (User-defined Functions, UDFs) durch Ausführen des Befehls CREATE FUNCTION zu erstellen. Weitere Informationen finden Sie unter UDF-Sicherheit und Rechte.

Die Berechtigung USAGE ON LANGUAGE ist erforderlich, um gespeicherte Prozeduren durch Ausführen des Befehls CREATE PROCEDURE zu erstellen. Weitere Informationen finden Sie unter Sicherheit und Berechtigungen für gespeicherte Prozeduren .

Für Python-UDFs verwenden Sie `plpythonu`. Für SQL-UDFs verwenden Sie `sql`. Für gespeicherte Prozeduren verwenden Sie `plpgsql`.

FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]

Gibt den SQL-Befehl an, für den die Berechtigung erteilt wird. Sie können ALL angeben, um die Berechtigung für die Anweisungen COPY, UNLOAD, EXTERNAL FUNCTION und CREATE MODEL zu erteilen. Diese Klausel gilt nur für die Erteilung der Berechtigung ASSUMEROLE.

ALTER

Erteilt Benutzern die Berechtigung ALTER zum Hinzufügen von Objekten zu einem Datashare oder zum Entfernen daraus oder zum Festlegen der Eigenschaft PUBLICACCESSIBLE. Weitere Informationen finden Sie unter ALTER DATASHARE.

SHARE

Erteilt Benutzern und Benutzergruppen Berechtigungen zum Hinzufügen von Datenkonsumenten zu einem Datashare. Diese Berechtigung ist erforderlich, damit der betreffende Konsument (Konto oder Namespace) von seinen Clustern aus auf das Datashare zugreifen kann. Bei dem Verbraucher kann es sich um dasselbe oder ein anderes AWS Konto mit demselben oder

einem anderen Cluster-Namespace handeln, der durch eine GUID (Globally Unique Identifier) angegeben wird.

ON DATASHARE datashare_name

Erteilt die angegebenen Berechtigungen für das referenzierte Datashare. Informationen über die Granularität der Zugriffskontrolle für Verbraucher finden Sie unter Freigeben von Daten auf verschiedenen Ebenen in Amazon Redshift.

USAGE

Wenn USAGE einem Konsumenten-Konto oder einem Namespace innerhalb desselben Kontos gewährt wird, kann dieses Konsumenten-Konto oder der Namespace innerhalb des Kontos schreibgeschützt auf das Datashare und die Objekte auf dem Datashare zugreifen.

TO NAMESPACE 'clusternamespace GUID'

Gibt einen Namespace in demselben Konto an, in dem Konsumenten die angegebenen Berechtigungen für das Datashare erhalten können. Namespaces verwenden eine alphanumerische 128-Bit-GUID.

TO ACCOUNT 'Kontonummer' [VIA DATA CATALOG]

Gibt die Nummer eines anderen Kontos an, dessen Konsumenten die angegebenen Berechtigungen für das Datashare erhalten können. Die Angabe „VIA DATA CATALOG“ bedeutet, dass Sie einem Lake-Formation-Konto die Berechtigung zur Nutzung des Datashares gewähren. Wenn Sie diesen Parameter weglassen, gewähren Sie die Nutzungsberechtigung einem Konto, dem der Cluster gehört.

ON DATABASE shared_database_name> [, ...]

Erteilt die angegebenen Nutzungsberechtigungen für die angegebene Datenbank, die im angegebenen Datashare erstellt wird.

ON SCHEMA shared_schema

Erteilt die angegebenen Berechtigungen für das angegebene Schema, das im angegebenen Datashare erstellt wird.

FOR { SCHEMAS | TABLES | FUNCTIONS | PROCEDURES | LANGUAGES } IN

Gibt die Datenbankobjekte an, für die Berechtigungen gewährt werden sollen. Die Parameter nach IN definieren den Bereich der gewährten Berechtigung.

CREATE MODEL

Erteilt bestimmten Benutzern oder Benutzergruppen die Berechtigung CREATE MODEL.

ON MODEL model_name

Erteilt die Berechtigung EXECUTE für ein spezifisches Modell.

ACCESS CATALOG

Gewährt die Berechtigung, relevante Metadaten von Objekten anzuzeigen, auf die die Rolle Zugriff hat.

{ role } [, ...]

Die Rolle, die einer anderen Rolle, einem Benutzer oder PUBLIC gewährt werden soll.

PUBLIC stellt eine Gruppe dar, die stets alle Benutzer enthält. Die Berechtigungen eines einzelnen Benutzers sind die Summe der Berechtigungen, die PUBLIC gewährt werden, der Berechtigungen, die Gruppen gewährt werden, zu denen der Benutzer gehört, und der Berechtigungen, die dem einzelnen Benutzer gewährt werden.

TO { { benutzer_name [WITH ADMIN OPTION] } | role }[, ...]

Gewährt die angegebene Rolle einem bestimmten Benutzer mit WITH ADMIN OPTION, einer anderen Rolle oder PUBLIC.

Die WITH ADMIN OPTION-Klausel bietet allen Berechtigungsempfängern die Verwaltungsoptionen für alle erteilten Rollen.

EXPLAIN RLS TO ROLE Rollenname

Erteilt die Berechtigung, einer Rolle die RLS-Richtlinienfilter einer Abfrage im EXPLAIN-Plan zu erläutern.

IGNORE RLS TO ROLE Rollenname

Erteilt die Berechtigung, RLS-Richtlinien für eine Abfrage an eine Rolle zu umgehen.

Nutzungshinweise

Weitere Informationen zu den Nutzungshinweisen für GRANT finden Sie unter the section called “Nutzungshinweise”.

Beispiele

Beispiele für die Verwendung von GRANT finden Sie unter the section called “Beispiele”.

Nutzungshinweise

Um Rechte für ein Objekt zu gewähren, müssen Sie mindestens eines der folgenden Kriterien erfüllen:

- Sie müssen der Besitzer des Objekts sein.
- Sie müssen ein Superuser sein.
- Sie müssen das Recht besitzen, Rechte für dieses Objekt und Recht zu gewähren.

Der folgende Befehl gewährt beispielsweise dem Benutzer HR die erforderlichen Rechte, um SELECT-Befehle für die Mitarbeitertabelle ausführen, anderen Benutzern dieselben Recht gewähren und diese Rechte für andere Benutzer widerrufen zu können.

```
grant select on table employees to HR with grant option;
```

Beachten Sie, dass HR keine Rechte für andere Operationen als SELECT oder für andere Tabellen als die Mitarbeitertabelle gewähren kann.

Der folgende Befehl gewährt beispielsweise dem Benutzer HR die erforderlichen Rechte, um ALTER-Befehle für die Mitarbeitertabelle ausführen, anderen Benutzern dieselben Rechte gewähren und diese Rechte für andere Benutzer widerrufen zu können.

```
grant ALTER on table employees to HR with grant option;
```

Beachten Sie, dass HR keine Rechte für andere Operationen als ALTER oder für andere Tabellen als die Mitarbeitertabelle gewähren kann.

Die Gewährung von Rechten für eine Ansicht impliziert nicht, dass auch Rechte in Bezug auf die zugrunde liegenden Tabellen gewährt werden. Gleichermaßen impliziert die Gewährung von Rechten für ein Schema nicht, dass auch Rechte in Bezug auf die zugrunde liegenden Tabellen gewährt werden. Sie müssen den Zugriff auf die zugrunde liegenden Tabellen explizit gewähren.

Um einer AWS Lake Formation Tabelle Berechtigungen zu gewähren, muss die IAM-Rolle, die dem externen Schema der Tabelle zugeordnet ist, über die Berechtigung verfügen, der externen Tabelle Berechtigungen zu gewähren. Im folgenden Beispiel wird ein externes Schema mit einer zugeordneten IAM-Rolle erstellt `myGrantor`. Die IAM-Rolle `myGrantor` verfügt über die Berechtigung, anderen Berechtigungen zu gewähren. Der Befehl GRANT nutzt die Berechtigung der

IAM-Rolle `myGrantor`, die dem externen Schema zugeordnet ist, um der IAM-Rolle `myGrantee` Berechtigungen zu gewähren.

```
create external schema mySchema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myGrantor'
create external database if not exists;
```

```
grant select
on external table mySchema.mytable
to iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

Wenn Sie einer IAM-Rolle mit GRANT ALL alle Berechtigungen gewähren, werden individuelle Berechtigungen im zugehörigen Lake Formation-fähigen Datenkatalog gewährt. Mit dem folgenden GRANT ALL-Befehl werden beispielsweise die einzelnen Berechtigungen (SELECT, ALTER, DROP, DELETE und INSERT) gewährt, wie in der Lake Formation-Konsole gezeigt.

```
grant all
on external table mySchema.mytable
to iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

Superuser können auf alle Objekte zugreifen, unabhängig von GRANT- und REVOKE-Befehlen, mit denen Objektrechte festgelegt werden.

Verwendungshinweise für die Zugriffskontrolle auf Spaltenebene

Die folgenden Verwendungshinweise gelten für Berechtigungen auf Spaltenebene für Amazon-Redshift-Tabellen und -Ansichten. Diese Hinweise beschreiben Tabellen; die gleichen Hinweise gelten für Ansichten, sofern nicht ausdrücklich anderweitig vermerkt.

- Für eine Amazon-Redshift-Tabelle können Sie nur die SELECT- und UPDATE-Berechtigungen Spaltenebene erteilen. Für eine Amazon-Redshift-Ansicht können Sie nur die SELECT-Berechtigung auf Spaltenebene erteilen.
- Das Schlüsselwort ALL ist ein Synonym für kombinierte SELECT- und UPDATE-Berechtigungen, wenn es im Kontext eines GRANT auf Spaltenebene für eine Tabelle verwendet wird.
- Wenn Sie nicht über die Berechtigung SELECT für alle Spalten in einer Tabelle verfügen, werden bei Ausführung einer Operation SELECT * nur die Spalten zurückgegeben, auf die Sie Zugriff

haben. Wenn Sie eine Ansicht verwenden, versucht eine SELECT *-Operation, auf alle Spalten in der Ansicht zuzugreifen. Wenn Sie nicht berechtigt sind, auf alle Spalten zuzugreifen, schlagen diese Abfragen fehl und es wird die Fehlermeldung „Zugriff verweigert“ angezeigt.

- SELECT * wird in den folgenden Fällen nicht nur auf die zugänglichen Spalten erweitert:
  - Sie können mit SELECT * keine reguläre Ansicht nur mit zugänglichen Spalten erstellen.
  - Sie können mit SELECT * keine materialisierte Ansicht nur mit zugänglichen Spalten erstellen.
- Wenn Sie über SELECT- oder UPDATE-Berechtigungen für eine Tabelle oder Ansicht verfügen und eine Spalte hinzufügen, haben Sie weiterhin die gleichen Berechtigungen für die Tabelle oder Ansicht und damit alle ihre Spalten.
- Nur der Besitzer einer Tabelle oder ein Superuser kann Berechtigungen auf Spaltenebene erteilen.
- Die WITH GRANT OPTION-Klausel wird für Berechtigungen auf Spaltenebene nicht unterstützt.
- Sie können nicht dieselbe Berechtigung sowohl auf Tabellenebene als auch auf Spaltenebene besitzen. Beispielsweise kann der Benutzer `data_scientist` nicht sowohl über die SELECT-Berechtigung für die Tabelle `employee` als auch über die SELECT-Berechtigung für die Spalte `employee.department` verfügen. Berücksichtigen Sie die folgenden Ergebnisse, wenn Sie einer Tabelle und einer Spalte innerhalb der Tabelle dieselbe Berechtigung erteilen:
  - Wenn ein Benutzer über eine Berechtigung auf Tabellenebene für eine Tabelle verfügt, hat das Gewähren derselben Berechtigung auf Spaltenebene keine Auswirkungen.
  - Wenn ein Benutzer über eine Berechtigung auf Tabellenebene für eine Tabelle verfügt, wird beim Widerrufen derselben Berechtigung für eine oder mehrere Spalten der Tabelle ein Fehler zurückgegeben. Widerrufen Sie stattdessen die Berechtigung auf Tabellenebene.
  - Wenn ein Benutzer über eine Berechtigung auf Spaltenebene verfügt, dann wird beim Erteilen derselben Berechtigung auf Tabellenebene ein Fehler zurückgegeben.
  - Wenn ein Benutzer über eine Berechtigung auf Spaltenebene verfügt, werden beim Widerrufen derselben Berechtigung auf Tabellenebene sowohl Spalten- als auch Tabellenberechtigungen für alle Spalten in der Tabelle entzogen.
- Sie können für späte Bindungsansichten keine Berechtigungen auf Spaltenebene erteilen.
- Sie müssen über SELECT-Berechtigungen auf Tabellenebene für die Basistabellen verfügen, um eine materialisierte Ansicht zu erstellen. Selbst, wenn Sie über Berechtigungen auf Spaltenebene für bestimmte Spalten verfügen, können Sie keine materialisierte Ansicht für nur diese Spalten erstellen. Sie können jedoch Spalten einer materialisierten Ansicht, ähnlich wie regulären Ansichten, SELECT-Berechtigungen erteilen.
- Verwenden Sie die Ansicht PG_ATTRIBUTE_INFO, um Berechtigungen auf Spaltenebene zu suchen.

Hinweise zur Erteilung der Berechtigung ASSUMEROLE

Die folgenden Nutzungshinweise gelten für die Erteilung der Berechtigung ASSUMEROLE in Amazon Redshift.

Mit der Berechtigung ASSUMEROLE steuern Sie IAM-Rollenzugriffsberechtigungen für Datenbankbenutzer, -rollen oder -gruppen auf Befehle wie COPY, UNLOAD, EXTERNAL FUNCTION oder CREATE MODEL. Nachdem Sie einem Benutzer, einer Rolle oder einer Gruppe die Berechtigung ASSUMEROLE für eine IAM-Rolle erteilt haben, kann der Benutzer, die Rolle oder die Gruppe diese Rolle übernehmen, wenn der Befehl ausgeführt wird. Mit der Berechtigung ASSUMEROLE können Sie bei Bedarf Zugriff auf die entsprechenden Befehle gewähren.

Nur ein Datenbank-Superuser kann die Berechtigung ASSUMEROLE für Benutzer, Rollen und Gruppen erteilen oder entziehen. Ein Superuser behält immer die Berechtigung ASSUMEROLE.

Um die Verwendung der Berechtigung ASSUMEROLE für Benutzer, Rollen und Gruppen zu aktivieren, führt ein Superuser die beiden folgenden Aktionen aus:

- Führen Sie die folgende Anweisung einmalig auf dem Cluster aus:

  ```
  revoke assumerole on all from public for all;
  ```

- Erteilen der Berechtigung ASSUMEROLE an Benutzer, Rollen und Gruppen für die entsprechenden Befehle.

Sie können eine Rollenverkettung in der ON-Klausel angeben, wenn Sie die Berechtigung ASSUMEROLE erteilen. Verwenden Sie Kommas, um Rollen in einer Rollenkette zu trennen, z. B. ., `Role1,Role2,Role3`. Wenn beim Erteilen der Berechtigung ASSUMEROLE eine Rollenverkettung angegeben wurde, müssen Sie die Rollenkette angeben, wenn Vorgänge ausgeführt werden, die durch die Berechtigung ASSUMEROLE gewährt werden. Sie können keine einzelnen Rollen innerhalb der Rollenkette angeben, wenn Vorgänge ausgeführt werden, die durch die Berechtigung ASSUMEROLE gewährt werden. Wenn einem Benutzer, einer Rolle oder einer Gruppe beispielsweise die Rollenkette `Role1,Role2,Role3` zugewiesen wurde, können Sie nicht nur `Role1` für die Ausführung von Vorgängen angeben.

Wenn ein Benutzer versucht, eine Operation COPY, UNLOAD, EXTERNAL FUNCTION oder CREATE MODEL auszuführen, und die Berechtigung ASSUMEROLE nicht erteilt wurde, wird eine Meldung ähnlich der folgenden angezeigt.

```
ERROR:  User awsuser does not have ASSUMEROLE permission on IAM role
 "arn:aws:iam::123456789012:role/RoleA" for COPY
```

Informationen zum Auflisten von Benutzern, denen über die Berechtigung ASSUMEROLE Zugriff auf IAM-Rollen und -Befehle gewährt wurde, finden Sie unter HAS_ASSUMEROLE_PRIVILEGE. Informationen zum Auflisten von Berechtigungen für IAM-Rollen und -Befehle, die einem von Ihnen spezifizierten Benutzer erteilt wurden, finden Sie unter PG_GET_IAM_ROLE_BY_USER. Informationen zum Auflisten von Benutzern, Rollen und Gruppen, denen Zugriff auf eine von Ihnen spezifizierte IAM-Rolle erteilt wurde, finden Sie unter PG_GET_GRANTEE_BY_IAM_ROLE.

Hinweise zum Erteilen von Machine-Learning-Berechtigungen

Sie können Berechtigungen für eine ML-Funktion nicht direkt erteilen oder widerrufen. Eine ML-Funktion gehört zu einem ML-Modell, und die Berechtigungen werden über das Modell gesteuert. Stattdessen erteilen Sie Berechtigungen für das ML-Modell. Das folgende Beispiel zeigt, wie allen Benutzern Berechtigungen zum Ausführen der dem Modell `customer_churn` zugeordneten ML-Funktion erteilt werden.

```
GRANT EXECUTE ON MODEL customer_churn TO PUBLIC;
```

Sie können einem Benutzer auch alle Berechtigungen für das ML-Modell `customer_churn` erteilen.

```
GRANT ALL on MODEL customer_churn TO ml_user;
```

Das Erteilen der Berechtigung `EXECUTE` für eine ML-Funktion schlägt fehl, wenn das Schema eine ML-Funktion enthält, auch wenn diese ML-Funktion bereits über `GRANT EXECUTE ON MODEL` über die Berechtigung `EXECUTE` verfügt. Wir empfehlen, bei Verwendung des Befehls `CREATE MODEL` ein separates Schema zu verwenden, um die ML-Funktionen in einem separaten Schema zu behalten. Das folgende Beispiel veranschaulicht die Vorgehensweise hierfür.

```
CREATE MODEL ml_schema.customer_churn
FROM customer_data
TARGET churn
FUNCTION ml_schema.customer_churn_prediction
IAM_ROLE default
SETTINGS (
 S3_BUCKET 'your-s3-bucket'
);
```

Beispiele

Im folgenden Beispiel erhält der Benutzer das SELECT-Recht für die Tabelle SALES `fred`.

```
grant select on table sales to fred;
```

Im folgenden Beispiel erhält der Benutzer das SELECT-Recht für alle Tabellen im Schema QA_TICKIT `fred`.

```
grant select on all tables in schema qa_tickit to fred;
```

Im folgenden Beispiel erhält die Benutzergruppe QA_USERS alle Schemarechte für das Schema QA_TICKIT. Die Schemarechte sind CREATE und USAGE. USAGE gewährt Benutzern den Zugriff auf die Objekte im Schema, jedoch keine Rechte wie INSERT oder SELECT für diese Objekte. Gewähren Sie für jedes Objekt separat Berechtigungen.

```
create group qa_users;
grant all on schema qa_tickit to group qa_users;
```

Im folgenden Beispiel erhalten alle Benutzer in der Gruppe QA_USERS alle Rechte für die Tabelle SALES im Schema QA_TICKIT.

```
grant all on table qa_tickit.sales to group qa_users;
```

Im folgenden Beispiel erhalten alle Benutzer in den Gruppen QA_USERS und RO_USERS alle Berechtigungen für die Tabelle SALES im Schema QA_TICKIT.

```
grant all on table qa_tickit.sales to group qa_users, group ro_users;
```

Im folgenden Beispiel erhalten alle Benutzer in der Gruppe QA_USERS die DROP-Berechtigung für die Tabelle SALES im Schema QA_TICKIT.

```
grant drop on table qa_tickit.sales to group qa_users;>
```

Die folgende Abfolge von Befehlen zeigt, dass der Zugriff auf ein Schema keine Rechte in Bezug auf eine Tabelle im Schema gewährt.

```
create user schema_user in group qa_users password 'Abcd1234';
create schema qa_tickit;
```

```
create table qa_tickit.test (col1 int);
grant all on schema qa_tickit to schema_user;

set session authorization schema_user;
select current_user;


current_user
--------------
schema_user
(1 row)


select count(*) from qa_tickit.test;


ERROR: permission denied for relation test [SQL State=42501]


set session authorization dw_user;
grant select on table qa_tickit.test to schema_user;
set session authorization schema_user;
select count(*) from qa_tickit.test;


count
-------
0
(1 row)
```

Die folgende Abfolge von Befehlen zeigt, dass der Zugriff auf eine Ansicht keinen Zugriff auf die zugrunde liegenden Tabellen impliziert. Der Benutzer namens VIEW_USER kann nicht aus der Tabelle DATE auswählen, obwohl diesem Benutzer alle Rechte in Bezug auf VIEW_DATE gewährt wurden.

```
create user view_user password 'Abcd1234';
create view view_date as select * from date;
grant all on view_date to view_user;
set session authorization view_user;
select current_user;


current_user
```

```
--------------
view_user
(1 row)


select count(*) from view_date;


count
-------
365
(1 row)


select count(*) from date;


ERROR:  permission denied for relation date
```

Im folgenden Beispiel wird dem Benutzer `cust_name` die SELECT-Berechtigung für die Spalten `cust_phone` und `cust_profile` der Tabelle `user1` gewährt.

```
grant select(cust_name, cust_phone) on cust_profile to user1;
```

Im folgenden Beispiel wird der Gruppe `cust_name` die SELECT-Berechtigung für die Spalten `cust_phone` und `cust_contact_preference` und die UPDATE-Berechtigung für die Spalte `cust_profile` der Tabelle `sales_group` gewährt.

```
grant select(cust_name, cust_phone), update(cust_contact_preference) on cust_profile to
 group sales_group;
```

Das folgende Beispiel zeigt die Verwendung des Schlüsselworts ALL, um der Gruppe `cust_profile` sowohl SELECT- als auch UPDATE-Berechtigungen für drei Spalten der Tabelle `sales_admin` zu erteilen.

```
grant ALL(cust_name, cust_phone,cust_contact_preference) on cust_profile to group
 sales_admin;
```

Im folgenden Beispiel wird dem Benutzer `cust_name` die SELECT-Berechtigung für die Spalte `cust_profile_vw` der Ansicht `user2` gewährt.

```
grant select(cust_name) on cust_profile_vw to user2;
```

Beispiele für die Gewährung von Zugriff auf Datashares

Im Folgenden finden Sie Beispiele für die GRANT-Datashare-Nutzungsberechtigungen für eine bestimmte Datenbank oder ein bestimmtes Schema, die auf einem Datashare erstellt wurden.

Im folgenden Beispiel erteilt ein Administrator auf Produzentenseite dem angegebenen Namespace die USAGE-Berechtigung für den `salesshare`-Datashare.

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Im folgenden Beispiel erteilt ein konsumentenseitiger Administrator die USAGE-Berechtigung auf `sales_db` für `Bob`.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

Im folgenden Beispiel erteilt ein konsumentenseitiger Administrator der `Analyst_role`-Rolle die GRANT USAGE-Berechtigung für das `sales_schema`-Schema. `sales_schema` ist ein externes Schema, das auf sales_db verweist.

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Zu diesem Zeitpunkt können `Bob` und `Analyst_role` auf alle Datenbankobjekte in `sales_schema` und `sales_db` zugreifen.

Das folgende Beispiel zeigt, wie zusätzliche Berechtigungen auf Objektebene für Objekte in einer gemeinsam genutzten Datenbank erteilt werden. Diese zusätzlichen Berechtigungen sind nur erforderlich, wenn der Befehl CREATE DATABASE, mit dem die gemeinsam genutzte Datenbank erstellt wurde, die WITH PERMISSIONS-Klausel verwendet hat. Wenn der Befehl CREATE DATABASE die Klausel WITH PERMISSIONS nicht verwendet hat, beinhaltet die Gewährung der USAGE-Berechtigung für die gemeinsam genutzte Datenbank vollen Zugriff auf alle Objekte in dieser Datenbank.

```
GRANT SELECT ON sales_db.sales_schema.tickit_sales_redshift to Bob;
```

Beispiele für die Gewährung von bereichsbezogenen Berechtigungen

Im folgenden Beispiel wird der `Sales`-Rolle die Verwendung aller derzeitigen und zukünftigen Schemata in der `Sales_db`-Datenbank gestattet.

```
GRANT USAGE FOR SCHEMAS IN DATABASE Sales_db TO ROLE Sales;
```

Im folgenden Beispiel wird dem Benutzer `alice` die SELECT-Berechtigung für alle derzeitigen und zukünftigen Tabellen in der `Sales_db`-Datenbank sowie `alice` die Berechtigung erteilt, anderen Benutzern bereichsbezogene Berechtigungen für Tabellen in `Sales_db` zu gewähren.

```
GRANT SELECT FOR TABLES IN DATABASE Sales_db TO alice WITH GRANT OPTION;
```

Im folgenden Beispiel wird dem Benutzer bob die EXECUTE-Berechtigung für Funktionen im `Sales_schema`-Schema gewährt.

```
GRANT EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema TO bob;
```

Im folgenden Beispiel werden der `Sales`-Rolle alle Berechtigungen für alle Tabellen im Schema `ShareSchema` der Datenbank `ShareDb` gewährt. Bei der Angabe des Schemas können Sie die Datenbank des Schemas im zweiteiligen Format `database.schema` angeben.

```
GRANT ALL FOR TABLES IN SCHEMA ShareDb.ShareSchema TO ROLE Sales;
```

Das folgende Beispiel zeigt dieselbe Abfrage wie im vorherigen Beispiel. Sie können die Datenbank mit dem `DATABASE`-Schlüsselwort angeben, anstatt ein zweiteiliges Format zu verwenden.

```
GRANT ALL FOR TABLES IN SCHEMA ShareSchema DATABASE ShareDb TO ROLE Sales;
```

Beispiele für die Erteilung der ASSUMEROLE-Berechtigung

Im Folgenden finden Sie Beispiele für die Erteilung der ASSUMEROLE-Berechtigung.

Das folgende Beispiel zeigt die REVOKE-Anweisung, die ein Superuser einmal auf dem Cluster ausführt, um die Verwendung der ASSUMEROLE-Berechtigung für Benutzer und Gruppen zu ermöglichen. Anschließend erteilt der Superuser Benutzern und Gruppen die Berechtigung ASSUMEROLE für die entsprechenden Befehle. Informationen zum Aktivieren der Verwendung der ASSUMEROLE-Berechtigung für Benutzer und Gruppen finden Sie unter Hinweise zur Erteilung der Berechtigung ASSUMEROLE.

```
revoke assumerole on all from public for all;
```

Im folgenden Beispiel erhält der Benutzer `reg_user1` die ASSUMEROLE-Berechtigung für die IAM-Rolle `Redshift-S3-Read`, um COPY-Vorgänge auszuführen.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-S3-Read'
to reg_user1 for copy;
```

Im folgenden Beispiel erhält der Benutzer `reg_user1` die ASSUMEROLE-Berechtigung für die IAM-Rollenkette `RoleA`, `RoleB`, um UNLOAD-Vorgänge auszuführen.

```
grant assumerole
on 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB'
to reg_user1
for unload;
```

Im Folgenden finden Sie ein Beispiel für den Befehl UNLOAD unter Verwendung der IAM-Rollenkette `RoleA`, `RoleB`.

```
unload ('select * from venue limit 10')
to 's3://companyb/redshift/venue_pipe_'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

Im folgenden Beispiel erhält der Benutzer `reg_user1` die ASSUMEROLE-Berechtigung für die IAM-Rolle `Redshift-Exfunc`, um externe Funktionen auszuführen.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-Exfunc'
to reg_user1 for external function;
```

Im folgenden Beispiel erhält der Benutzer `reg_user1` die ASSUMEROLE-Berechtigung für die IAM-Rolle `Redshift-model`, um Machine-Learning-Modelle zu erstellen.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-ML'
to reg_user1 for create model;
```

Beispiele für die Erteilung der ROLE-Berechtigung

Im folgenden Beispiel wird die Rolle sample_role1 einem Benutzer mit dem Namen user1 erteilt.

```
CREATE ROLE sample_role1;
GRANT ROLE sample_role1 TO user1;
```

Im folgenden Beispiel wird dem Benutzer user1 die Rolle sample_role1 mit WITH ADMIN OPTION gewährt, die aktuelle Sitzung für user1 festgelegt und user1 gewährt die Rolle sample_role1 für user2.

```
GRANT ROLE sample_role1 TO user1 WITH ADMIN OPTION;
SET SESSION AUTHORIZATION user1;
GRANT ROLE sample_role1 TO user2;
```

Im folgenden Beispiel wird die Rolle sample_role1 der Rolle sample_role2 gewährt.

```
GRANT ROLE sample_role1 TO ROLE sample_role2;
```

Im folgenden Beispiel wird die Rolle sample_role2 für sample_role3 und sample_role4 gewährt. Dann wird versucht, sample_role3 für sample_role1 zu gewähren.

```
GRANT ROLE sample_role2 TO ROLE sample_role3;
GRANT ROLE sample_role3 TO ROLE sample_role2;
ERROR: cannot grant this role, a circular dependency was detected between these roles
```

Im folgenden Beispiel werden die CREATE USER-Systemberechtigungen für sample_role1 gewährt.

```
GRANT CREATE USER TO ROLE sample_role1;
```

Im folgenden Beispiel wird die vom System definierte Rolle `sys:dba` für user1 gewährt.

```
GRANT ROLE sys:dba TO user1;
```

Im folgenden Beispiel wird versucht, sample_role3 in einer kreisförmigen Abhängigkeit von sample_role2 zu gewähren.

```
CREATE ROLE sample_role3;
GRANT ROLE sample_role2 TO ROLE sample_role3;
GRANT ROLE sample_role3 TO ROLE sample_role2; -- fail
ERROR:  cannot grant this role, a circular dependency was detected between these roles
```

INSERT

Themen

Fügt neue Zeilen in eine Tabelle ein. Sie können eine einzelne Zeile mit der VALUES-Syntax, mehrere Zeilen mit der VALUES-Syntax oder eine oder mehrere Zeilen einfügen, die durch die Ergebnisse einer Abfrage definiert werden (INSERT INTO...SELECT).

> **Note**
>
> Es wird nachdrücklich empfohlen, den Befehl COPY zu verwenden, um große Mengen von Daten zu laden. Die Verwendung einzelner INSERT-Anweisungen, um eine Tabelle auszufüllen, kann äußerst langsam sein. Wenn Ihre Daten in anderen Amazon-Redshift-Datenbanktabellen bereits vorhanden sind, können Sie alternativ INSERT INTO SELECT oder CREATE TABLE AS verwenden, um die Leistung zu verbessern. Weitere Informationen zur Verwendung des Befehls COPY zum Laden von Tabellen finden Sie unter Laden von Daten.

> **Note**
>
> Die maximal zulässige Größe für eine einzelne SQL-Anweisung ist 16 MB.

Syntax

```
INSERT INTO table_name [ ( column [, ...] ) ]
{DEFAULT VALUES |
VALUES ( { expression | DEFAULT } [, ...] )
[, ( { expression | DEFAULT } [, ...] )
[, ...] ] |
query }
```

Parameter

table_name

Eine temporäre oder persistente Tabelle. Nur der Besitzer der Tabelle oder ein Benutzer mit dem Recht INSERT für die Tabelle können Zeilen einfügen. Wenn Sie die Klausel query verwenden, um Zeilen einzufügen, müssen Sie das SELECT-Recht für die in der Abfrage genannten Tabellen besitzen.

> **Note**
>
> Verwenden Sie INSERT (externe Tabelle), um Ergebnisse einer SELECT-Abfrage in vorhandene Tabellen im externen Katalog einzufügen. Weitere Informationen finden Sie unter INSERT (externe Tabelle).

column

Sie können in eine oder mehrere Spalten der Tabelle Werte einfügen. Sie können die Zielspaltennamen in beliebiger Reihenfolge auflisten. Wenn Sie keine Spaltenliste angeben, müssen die Werte, die eingefügt werden sollen, den Tabellenspalten in der Reihenfolge entsprechen, in der sie in der Anweisung CREATE TABLE deklariert wurden. Wenn die Anzahl der Spalten, die eingefügt werden sollen, kleiner als die Anzahl der Spalten in der Tabelle ist, werden die ersten n Spalten geladen.

In jede nicht in der Anweisung INSERT aufgelistete Spalte wird entweder der deklarierte Standardwert oder ein Null-Wert geladen (implizit oder explizit).

DEFAULT VALUES

Wenn den Spalten in der Tabelle beim Erstellen der Tabelle Standardwerte zugewiesen wurden, verwenden Sie diese Schlüsselwörter, um eine Zeile einzufügen, die ausschließlich aus Standardwerten besteht. Wenn eine oder mehrere Spalten keine Standardwerte aufweisen, werden in diese Spalten Null-Werte eingefügt. Wenn eine oder mehrere Spalten als NOT NULL deklariert wurden, gibt die Anweisung INSERT einen Fehler zurück.

VALUES

Verwenden Sie dieses Schlüsselwort, um eine oder mehrere Zeilen einzufügen, wobei jede Zeile aus einem oder mehreren Werten besteht. Die VALUES-Liste für jede Zeile muss der Spaltenliste

entsprechen. Um mehrere Zeilen einzufügen, verwenden Sie ein Kommatrennzeichen zwischen den einzelnen Listen von Ausdrücken. Wiederholen Sie das Schlüsselwort VALUES nicht. Alle VALUES-Listen für eine INSERT-Anweisung für mehrere Zeilen müssen die gleiche Zahl von Werten enthalten.

expression

Ein einzelner Wert oder ein Ausdruck, der in einen einzelnen Wert evaluiert wird. Jeder Wert muss mit dem Datentyp der Spalte, in die er eingefügt wird, kompatibel sein. Wenn möglich, wird ein Wert, dessen Datentyp nicht dem deklarierten Datentyp der Spalte entspricht, automatisch in einen kompatiblen Datentyp umgewandelt. Beispiel:

- Der Dezimalwert `1.1` wird als `1` in eine INT-Spalte eingefügt.
- Der Dezimalwert `100.8976` wird als `100.90` in eine DEC(5,2)-Spalte eingefügt.

Sie können einen Wert explizit in einen kompatiblen Datentyp umwandeln, indem Sie eine Typumwandlungssyntax in den Ausdruck einschließen. Wenn beispielsweise die Spalte COL1 in Tabelle T1 eine CHAR(3)-Spalte ist:

```
insert into t1(col1) values('Incomplete'::char(3));
```

Diese Anweisung fügt den Wert `Inc` in die Spalte ein.

Für eine INSERT VALUES-Anweisung für eine einzelne Zeile können Sie eine skalare Unterabfrage als Ausdruck verwenden. Das Ergebnis der Unterabfrage wird in die entsprechende Tabelle eingefügt.

> **Note**
>
> Unterabfragen werden für INSERT VALUES-Anweisungen für mehrere Zeilen nicht als Ausdrücke unterstützt.

DEFAULT

Verwenden Sie dieses Schlüsselwort, um den Standardwert für eine Spalte einzufügen, der beim Erstellen der Tabelle definiert wurde. Wenn für eine Spalte kein Standardwert vorhanden ist, wird eine Null eingefügt. Sie können in eine Spalte, für die eine NOT NULL-Einschränkung vorhanden ist, keinen Standardwert einfügen, wenn dieser Spalte in der Anweisung CREATE TABLE nicht explizit ein Standardwert zugewiesen wurde.

query

Sie können eine oder mehrere Zeilen in die Tabelle einfügen, indem Sie eine Abfrage definieren. Alle Zeilen, die von der Abfrage erstellt werden, werden in die Tabelle eingefügt. Die Abfrage muss eine Spaltenliste zurückgeben, die mit den Spalten in der Tabelle kompatibel ist. Die Spaltennamen müssen dabei nicht übereinstimmen.

Nutzungshinweise

> **Note**
>
> Es wird nachdrücklich empfohlen, den Befehl COPY zu verwenden, um große Mengen von Daten zu laden. Die Verwendung einzelner INSERT-Anweisungen, um eine Tabelle auszufüllen, kann äußerst langsam sein. Wenn Ihre Daten in anderen Amazon-Redshift-Datenbanktabellen bereits vorhanden sind, können Sie alternativ INSERT INTO SELECT oder CREATE TABLE AS verwenden, um die Leistung zu verbessern. Weitere Informationen zur Verwendung des Befehls COPY zum Laden von Tabellen finden Sie unter Laden von Daten.

Das Datenformat für die eingefügten Werte muss mit dem Datenformat übereinstimmen, das von der Definition CREATE TABLE angegeben wird.

Nach dem Einfügen einer großen Zahl neuer Zeilen in eine Tabelle:

- Führen Sie eine Vacuum-Operation für die Tabelle durch, um Speicherplatz zurückzugewinnen und die Zeilen neu zu sortieren.
- Analysieren Sie die Tabelle, um Statistiken für den Abfrageplaner zu aktualisieren.

Wenn Werte in DECIMAL-Spalten eingefügt werden und die angegebene Skala überschritten wird, werden die geladenen Werte entsprechend gerundet. Wenn beispielsweise der Wert `20.259` in eine DECIMAL(8,2)-Spalte eingefügt wird, ist der gespeicherte Wert `20.26`.

Sie können in eine GENERATED BY DEFAULT AS IDENTITY-Spalte einfügen. Sie können als GENERATED BY DEFAULT AS IDENTITY-Spalten mit von Ihnen angegebenen Werten aktualisieren. Weitere Informationen finden Sie unter GENERATED BY DEFAULT AS IDENTITY.

Beispiele für INSERT

Die Tabelle CATEGORY in der Datenbank TICKIT enthält die folgenden Zeilen:

```
 catid | catgroup |  catname  |                  catdesc
-------+----------+-----------+--------------------------------------------
     1 | Sports   | MLB       | Major League Baseball
     2 | Sports   | NHL       | National Hockey League
     3 | Sports   | NFL       | National Football League
     4 | Sports   | NBA       | National Basketball Association
     5 | Sports   | MLS       | Major League Soccer
     6 | Shows    | Musicals  | Musical theatre
     7 | Shows    | Plays     | All non-musical theatre
     8 | Shows    | Opera     | All opera and light opera
     9 | Concerts | Pop       | All rock and pop music concerts
    10 | Concerts | Jazz      | All jazz singers and bands
    11 | Concerts | Classical | All symphony, concerto, and choir concerts
(11 rows)
```

Erstellen Sie eine Tabelle CATEGORY_STAGE mit einem ähnlichen Schema wie die Tabelle CATEGORY. Definieren Sie jedoch die Standardwerte für die Spalten:

```
create table category_stage
(catid smallint default 0,
catgroup varchar(10) default 'General',
catname varchar(10) default 'General',
catdesc varchar(50) default 'General');
```

Die folgende INSERT-Anweisung wählt alle Zeilen aus der Tabelle CATEGORY aus und fügt Sie in die Tabelle CATEGORY_STAGE ein.

```
insert into category_stage
(select * from category);
```

Die Klammern, in die die Abfrage eingeschlossen ist, sind optional.

Dieser Befehl fügt eine neue Zeile in die Tabelle CATEGORY_STAGE ein, in der für jede Spalte der Reihenfolge nach ein Wert angegeben ist:

```
insert into category_stage values
(12, 'Concerts', 'Comedy', 'All stand-up comedy performances');
```

Sie können auch eine neue Zeile einfügen, die spezifische Werte und Standardwerte kombiniert:

```
insert into category_stage values
(13, 'Concerts', 'Other', default);
```

Führen Sie die folgende Abfrage aus, um die eingefügten Zeilen zurückzugeben:

```
select * from category_stage
where catid in(12,13) order by 1;

 catid | catgroup | catname |             catdesc
-------+----------+---------+---------------------------------
    12 | Concerts | Comedy  | All stand-up comedy performances
    13 | Concerts | Other   | General
(2 rows)
```

In den folgenden Beispielen werden INSERT VALUES-Anweisungen für mehrere Zeilen gezeigt. Im ersten Beispiel werden spezifische CATID-Werte für zwei Zeilen und für die übrigen Spalten in den beiden Zeilen Standardwerte eingefügt.

```
insert into category_stage values
(14, default, default, default),
(15, default, default, default);

select * from category_stage where catid in(14,15) order by 1;
 catid | catgroup | catname | catdesc
-------+----------+---------+---------
    14 | General  | General | General
    15 | General  | General | General
(2 rows)
```

Im nächsten Beispiel werden drei Zeilen mit verschiedenen Kombinationen aus spezifischen und Standardwerten eingefügt:

```
insert into category_stage values
(default, default, default, default),
(20, default, 'Country', default),
(21, 'Concerts', 'Rock', default);

select * from category_stage where catid in(0,20,21) order by 1;
 catid | catgroup | catname | catdesc
-------+----------+---------+---------
```

```
     0 | General  | General | General
    20 | General  | Country | General
    21 | Concerts | Rock    | General
(3 rows)
```

Der erste Satz von VALUES in diesem Beispiel führt zu den gleichen Ergebnissen wie die Angabe von DEFAULT VALUES für eine INSERT-Anweisung für eine einzelne Zeile.

In den folgenden Beispielen wird das INSERT-Verhalten gezeigt, wenn eine Tabelle eine IDENTITY-Spalte besitzt. Erstellen Sie zunächst eine neue Version der Tabelle CATEGORY und fügen Sie dann Zeilen aus CATEGORY ein:

```
create table category_ident
(catid int identity not null,
catgroup varchar(10) default 'General',
catname varchar(10) default 'General',
catdesc varchar(50) default 'General');


insert into category_ident(catgroup,catname,catdesc)
select catgroup,catname,catdesc from category;
```

Beachten Sie, dass Sie in die Spalte CATID IDENTITY keine spezifischen ganzzahligen Werte einfügen können. Die Werte in der IDENTITY-Spalte werden automatisch generiert.

Im folgenden Beispiel wird gezeigt, dass Unterabfragen nicht als Ausdrücke in INSERT VALUES-Anweisungen für mehrere Zeilen verwendet werden können:

```
insert into category(catid) values
((select max(catid)+1 from category)),
((select max(catid)+2 from category));

ERROR: can't use subqueries in multi-row VALUES
```

Das folgende Beispiel zeigt eine Einfügung in eine temporäre Tabelle, die mithilfe der `WITH SELECT`-Klausel mit Daten aus der `venue`-Tabelle gefüllt wird. Weitere Informationen zur Tabelle `venue` finden Sie unter Beispieldatenbank.

Erstellen Sie zunächst die temporäre Tabelle `#venuetemp`.

```
CREATE TABLE #venuetemp AS SELECT * FROM venue;
```

Listen Sie die Zeilen in der #venuetemp-Tabelle auf.

```
SELECT * FROM #venuetemp ORDER BY venueid;

venueid | venuename                | venuecity  | venuestate| venueseats
--------+--------------------------+------------+-----------+------------
1       Toyota Park                Bridgeview   IL          0
2       Columbus Crew Stadium      Columbus     OH          0
3       RFK Stadium                Washington   DC          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
5       Gillette Stadium           Foxborough   MA          68756
...
```

Fügen Sie mithilfe der WITH SELECT-Klausel 10 doppelte Zeilen in die #venuetemp-Tabelle ein.

```
INSERT INTO #venuetemp (WITH venuecopy AS (SELECT * FROM venue) SELECT * FROM venuecopy
 ORDER BY 1 LIMIT 10);
```

Listen Sie die Zeilen in der #venuetemp-Tabelle auf.

```
SELECT * FROM #venuetemp ORDER BY venueid;

venueid | venuename                | venuecity  | venuestate| venueseats
--------+--------------------------+------------+-----------+------------
1       Toyota Park                Bridgeview   IL          0
1       Toyota Park                Bridgeview   IL          0
2       Columbus Crew Stadium      Columbus     OH          0
2       Columbus Crew Stadium      Columbus     OH          0
3       RFK Stadium                Washington   DC          0
3       RFK Stadium                Washington   DC          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
5       Gillette Stadium           Foxborough   MA          68756
5       Gillette Stadium           Foxborough   MA          68756
...
```

INSERT (externe Tabelle)

Fügt die Ergebnisse einer SELECT-Abfrage in bestehende externe Tabellen in einem externen Katalog ein, z. B. für AWS Glue AWS Lake Formation, oder einen Apache Hive-Metastore. Verwenden Sie dieselbe AWS Identity and Access Management (IAM-) Rolle, die für den Befehl

CREATE EXTERNAL SCHEMA verwendet wurde, um mit externen Katalogen und Amazon S3 zu interagieren.

Bei nicht partitionierten Tabellen schreibt der Befehl INSERT (externe Tabelle) Daten basierend auf den angegebenen Tabelleneigenschaften und dem Dateiformat in den Amazon-S3-Speicherort, der in der Tabelle definiert ist.

Bei partitionierten Tabellen schreibt INSERT (externe Tabelle) Daten gemäß dem in der Tabelle angegebenen Partitionsschlüssel in den Amazon-S3-Speicherort. Außerdem werden neue Partitionen automatisch im externen Katalog registriert, nachdem der INSERT-Vorgang abgeschlossen ist.

Sie können INSERT (externe Tabelle) nicht innerhalb eines Transaktionsblocks ausführen (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.

Syntax

```
INSERT INTO external_schema.table_name
{ select_statement }
```

Parameter

external_schema.table_name

Der Name eines vorhandenen externen Schemas und einer externen Zieltabelle, in die eingefügt werden soll.

select-statement

Eine Anweisung, die mindestens eine Zeile in die externe Tabelle einfügt, indem eine beliebige Abfrage definiert wird. Alle von der Abfrage erzeugten Zeilen werden auf Grundlage der Tabellendefinition entweder im Text- oder im Parquet-Format in Amazon S3 geschrieben. Die Abfrage muss eine Spaltenliste zurückgeben, die mit den Spaltendatentypen in der externen Tabelle kompatibel ist. Die Spaltennamen müssen jedoch nicht übereinstimmen.

Nutzungshinweise

Die Anzahl der Spalten in der SELECT-Abfrage muss mit der Summe der Datenspalten und Partitionsspalten übereinstimmen. Der Speicherort und der Datentyp jeder Datenspalte müssen mit dem der externen Tabelle übereinstimmen. Die Position der Partitionsspalten muss am Ende der

SELECT-Abfrage liegen, in derselben Reihenfolge, in der sie im Befehl CREATE EXTERNAL TABLE definiert wurden. Die Spaltennamen müssen nicht übereinstimmen.

In einigen Fällen möchten Sie möglicherweise den Befehl INSERT (externe Tabelle) für einen AWS Glue -Datenkatalog oder einen Hive-Metastore ausführen. Im Fall von muss die IAM-Rolle AWS Glue, die zur Erstellung des externen Schemas verwendet wurde, sowohl Lese- als auch Schreibberechtigungen für Amazon S3 und AWS Glue haben. Wenn Sie einen AWS Lake Formation Katalog verwenden, wird diese IAM-Rolle der Besitzer der neuen Lake Formation-Tabelle. Diese IAM-Rolle muss mindestens über die folgenden Berechtigungen verfügen:

- SELECT-, INSERT-, UPDATE-Berechtigung für die externe Tabelle
- Datenspeicherort-Berechtigung für den Amazon-S3-Pfad der externen Tabelle

Um sicherzustellen, dass Dateinamen eindeutig sind, verwendet Amazon Redshift das folgende Format für den Namen jeder Datei, die standardmäßig in Amazon S3 hochgeladen wurde.

*<date>*_*<time>*_*<microseconds>*_*<query_id>*_*<slice-number>*_part_*<part-number>*.*<format>*.

Ein Beispiel ist `20200303_004509_810669_1007_0001_part_00.parquet`.

Berücksichtigen Sie Folgendes, wenn Sie den Befehl INSERT (externe Tabelle) ausführen:

- Externe Tabellen, die ein anderes Format als PARQUET oder TEXTFILE haben, werden nicht unterstützt.
- Dieser Befehl unterstützt vorhandene Tabelleneigenschaften wie 'write.parallel', 'write.maxfilesize.mb', 'compression_type' und 'serialization.null.format'. Um diese Werte zu aktualisieren, führen Sie den Befehl ALTER TABLE SET TABLE PROPERTIES aus.
- Die Tabelleneigenschaft 'numRows' wird automatisch gegen Ende des INSERT-Vorgangs aktualisiert. Die Tabelleneigenschaft muss bereits definiert oder der Tabelle hinzugefügt werden, wenn sie nicht durch den CREATE EXTERNAL TABLE AS-Vorgang erstellt wurde.
- Die LIMIT-Klausel wird in der äußeren SELECT-Abfrage nicht unterstützt. Verwenden Sie stattdessen eine verschachtelte LIMIT-Klausel.
- Sie können die STL_UNLOAD_LOG-Tabelle verwenden, um die Dateien zu verfolgen, die von jedem INSERT (externe Tabelle)-Vorgang in Amazon S3 geschrieben wurden.
- Amazon Redshift unterstützt nur die Amazon-S3-Standardverschlüsselung für INSERT (externe Tabelle).

Beispiele für INSERT (externe Tabelle)

Im folgenden Beispiel werden die Ergebnisse der SELECT-Anweisung in die externe Tabelle eingefügt.

```
INSERT INTO spectrum.lineitem
SELECT * FROM local_lineitem;
```

Im folgenden Beispiel werden die Ergebnisse der SELECT-Anweisung mithilfe der statischen Partitionierung in eine partitionierte externe Tabelle eingefügt. Die Partitionsspalten sind in der SELECT-Anweisung fest kodiert. Die Partitionsspalten müssen sich am Ende der Abfrage befinden.

```
INSERT INTO spectrum.customer
SELECT name, age, gender, 'May', 28 FROM local_customer;
```

Im folgenden Beispiel werden die Ergebnisse der SELECT-Anweisung mithilfe der dynamischen Partitionierung in eine partitionierte externe Tabelle eingefügt. Die Partitionsspalten sind nicht fest kodiert. Daten werden automatisch zu den vorhandenen Partitionsordnern oder zu neuen Ordnern hinzugefügt, wenn eine neue Partition hinzugefügt wird.

```
INSERT INTO spectrum.customer
SELECT name, age, gender, month, day FROM local_customer;
```

LOCK

Schränkt den Zugriff auf eine Datenbanktabelle ein. Dieser Befehl ist nur sinnvoll, wenn er innerhalb eines Transaktionsblocks ausgeführt wird.

Der LOCK-Befehl bewirkt eine Sperrung auf Tabellenebene im Modus ACCESS EXCLUSIVE und wartet auf die Freigabe widersprüchlicher Sperren, wenn notwendig. Die explizite Sperrung einer Tabelle auf diese Weise führt dazu, dass Lese- und Schreibvorgänge für die Tabelle abwarten, wenn sie über andere Transaktionen oder Sitzungen versucht werden. Eine explizite Tabellensperre, die von einem Benutzer erstellt wird, hindert andere Benutzer vorübergehend daran, Daten aus dieser Tabelle auszuwählen oder Daten in diese Tabelle zu laden. Die Sperre wird aufgehoben, wenn die Transaktion, die den LOCK-Befehl enthält, abgeschlossen ist.

Befehle, die Tabellen referenzieren, wie Schreibvorgänge, bewirken implizit weniger einschränkende Tabellensperren. Wenn ein Benutzer beispielsweise versucht, Daten aus einer Tabelle zu lesen, während ein anderer Benutzer die Tabelle aktualisiert, stellen die gelesenen Daten einen Snapshot

der Daten dar, für die bereits ein Commit ausgeführt wurde. (In einigen Fällen werden Abfragen abgebrochen, wenn sie Regeln für die serialisierbare Isolierung verletzen.) Siehe Verwalten gleichzeitiger Schreiboperationen.

Einige DDL-Operationen, wie z. B. DROP TABLE und TRUNCATE, erstellen exklusive Sperren. Diese Operationen verhindern, dass Daten gelesen werden.

Wenn ein Sperrkonflikt auftritt, zeigt Amazon Redshift eine Fehlermeldung an, um den Benutzer zu warnen, der die konfliktbehaftete Transaktion gestartet hat. Die Transaktion, die den Sperrkonflikt empfangen hat, wird abgebrochen. Jedes Mal, wenn ein Sperrkonflikt eintritt, schreibt Amazon Redshift einen Eintrag in die Tabelle STL_TR_CONFLICT.

Syntax

```
LOCK [ TABLE ] table_name [, ...]
```

Parameter

TABLE

Optionales Schlüsselwort.

table_name

Der Name der Tabelle, die gesperrt werden soll. Sie können mehrere Tabellen sperren, indem Sie eine Liste verwenden, in der Tabellennamen durch Komma getrennt werden. Sie können keine Ansichten sperren.

Beispiel

```
begin;

lock event, sales;

...
```

MERGE

Führt Zeilen aus einer Quelltabelle bedingt in einer Zieltabelle zusammen. Herkömmlicherweise ist dies nur durch separate Verwendung mehrerer Insert-, Update- oder Delete-Anweisungen möglich.

Weitere Informationen zu den Operationen, die Sie mithilfe von MERGE kombinieren können, finden Sie unter UPDATE, DELETE und INSERT.

Syntax

```
MERGE INTO target_table
USING source_table [ [ AS ] alias ]
ON match_condition
[ WHEN MATCHED THEN { UPDATE SET col_name = { expr } [,...] | DELETE }
WHEN NOT MATCHED THEN INSERT [ ( col_name [,...] ) ] VALUES ( { expr } [, ...] ) |
REMOVE DUPLICATES ]
```

Parameter

target_table

Die temporäre oder permanente Tabelle, in die die MERGE-Anweisung zusammengeführt wird.

source_table

Die temporäre oder permanente Tabelle, die die Zeilen bereitstellt, die in target_table zusammengeführt werden sollen. Als source_table kommen auch Spectrum-Tabellen infrage. source_table kann keine Ansicht oder Unterabfrage sein.

alias

Der temporäre alternative Name für source_table.

Dieser Parameter ist optional. Optional kann dem Alias auch AS vorangestellt sein.

match_condition

Gibt übereinstimmende Prädikate zwischen der Quelltabellenspalte und der Zieltabellenspalte an, anhand derer bestimmt wird, ob die Zeilen in source_table mit Zeilen in target_table abgeglichen werden können. Wenn die Bedingung erfüllt ist, führt MERGE matched_clause für diese Zeile aus. Andernfalls führt MERGE not_matched_clause für diese Zeile aus.

WHEN MATCHED

Gibt die Aktion an, die ausgeführt werden soll, wenn die Übereinstimmungsbedingung zwischen einer Quell- und einer Zielzeile „True“ (Wahr) ergibt. Sie können entweder eine UPDATE-Aktion oder eine DELETE-Aktion angeben.

UPDATE

Aktualisiert die übereinstimmende Zeile in target_table. Nur Werte in dem von Ihnen angegebenen col_name werden aktualisiert.

DELETE

Löscht die übereinstimmende Zeile in target_table.

WHEN NOT MATCHED

Gibt die Aktion an, die ausgeführt werden soll, wenn die Übereinstimmungsbedingung „False“ (Falsch) oder „Unknown“ (Unbekannt) ergibt. Sie können nur die INSERT-Einfügeaktion für diese Klausel angeben.

INSERT

Fügt eine Zeile in target_table ein. Der Ziel-col_name kann in beliebiger Reihenfolge aufgelistet werden. Wenn Sie keine col_name-Werte angeben, werden standardmäßig alle Spalten der Tabelle in ihrer deklarierten Reihenfolge angeordnet.

col_name

Ein oder mehrere Spaltennamen, die Sie ändern möchten. Beziehen Sie den Tabellennamen bei der Angabe der Zielspalte nicht ein.

expr

Der Ausdruck, der den neuen Wert für col_name definiert.

REMOVE DUPLICATES

Gibt an, dass der MERGE-Befehl im vereinfachten Modus ausgeführt wird. Für den vereinfachten Modus gelten folgende Anforderungen:

- target_table und source_table müssen dieselbe Anzahl von Spalten und kompatible Spaltentypen haben.
- Lassen Sie die WHEN-Klausel und die UPDATE- und INSERT-Klauseln in Ihrem MERGE-Befehl weg.
- Verwenden Sie die REMOVE DUPLICATES-Klausel in Ihrem MERGE-Befehl.

Im vereinfachten Modus bewirkt MERGE Folgendes:

- Zeilen in target_table, die eine Übereinstimmung in source_table haben, werden aktualisiert, so dass sie den Werten in source_table entsprechen.

- Zeilen in source_table, die keine Übereinstimmung in target_table haben, werden in target_table eingefügt.
- Wenn mehrere Zeilen in target_table mit derselben Zeile in source_table übereinstimmen, werden die doppelten Zeilen entfernt. Amazon Redshift behält eine Zeile bei und aktualisiert sie. Doppelte Zeilen, die keiner Zeile in source_table entsprechen, bleiben unverändert.

Die Verwendung von REMOVE DUPLICATES bietet eine bessere Leistung als die Verwendung von WHEN MATCHED und WHEN NOT MATCHED. Wir empfehlen die Verwendung von REMOVE DUPLICATES, wenn target_table und source_table kompatibel sind und Sie keine doppelten Zeilen in target_table beibehalten müssen.

Nutzungshinweise

- Um MERGE-Anweisungen ausführen zu können, müssen Sie sowohl Besitzer von source_table als auch von target_table sein oder über die SELECT-Berechtigung für diese Tabellen verfügen. Darüber hinaus benötigen Sie UPDATE-, DELETE- und INSERT-Berechtigungen für target_table, je nachdem, welche Operationen in Ihrer MERGE-Anweisung enthalten sind.
- target_table kann keine Systemtabelle, Katalogtabelle oder externe Tabelle sein.
- source_table und target_table können nicht dieselbe Tabelle sein.
- Sie können die WITH-Klausel nicht in einer MERGE-Anweisung verwenden.
- Zeilen in target_table können nicht mit mehreren Zeilen in source_table übereinstimmen.

Betrachten Sie das folgende Beispiel:

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (1, 'Bob'), (2, 'John');
INSERT INTO source VALUES (1, 'Tony'), (1, 'Alice'), (3, 'Bill');

MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN UPDATE SET id = source.id, name = source.name
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
ERROR: Found multiple matches to update the same tuple.

MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN DELETE
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
```

```
ERROR: Found multiple matches to update the same tuple.
```

Bei beiden MERGE-Anweisungen schlägt der Vorgang fehl, da es mehrere Zeilen in der `source`-Tabelle mit einem ID-Wert von 1 gibt.

- match_condition und expr können nicht teilweise auf Spalten vom Typ SUPER verweisen. Wenn Ihr Objekt vom Typ SUPER beispielsweise ein Array oder eine Struktur ist, können Sie nicht einzelne Elemente dieser Spalte für match_condition oder expr verwenden, die gesamte Spalte dagegen schon.

  Betrachten Sie das folgende Beispiel:

```
CREATE TABLE IF NOT EXISTS target (key INT, value SUPER);
CREATE TABLE IF NOT EXISTS source (key INT, value SUPER);

INSERT INTO target VALUES (1, JSON_PARSE('{"key": 88}'));
INSERT INTO source VALUES (1, ARRAY(1, 'John')), (2, ARRAY(2, 'Bill'));

MERGE INTO target USING source ON target.key = source.key
WHEN matched THEN UPDATE SET value = source.value[0]
WHEN NOT matched THEN INSERT VALUES (source.key, source.value[0]);
ERROR: Partial reference of SUPER column is not supported in MERGE statement.
```

  Weitere Informationen zum Typ SUPER finden Sie unter Typ SUPER.

- Bei einer großen source_table kann die Leistung verbessert werden, indem die Join-Spalten von target_table und source_table als Verteilungsschlüssel definiert werden.
- Um die REMOVE DUPLICATES-Klausel benutzen zu können, benötigen Sie die SELECT-, INSERT- und DELETE-Berechtigungen für target_table.

Beispiele

Im folgenden Beispiel werden zwei Tabellen erstellt und anschließend wird eine MERGE-Operation für sie ausgeführt. Dabei werden übereinstimmende Zeilen in der Zieltabelle aktualisiert und nicht übereinstimmende Zeilen eingefügt. Daraufhin wird ein weiterer Wert in die Quelltabelle eingefügt und eine weitere MERGE-Operation ausgeführt. Dieses Mal werden dabei übereinstimmende Zeilen gelöscht und die neue Zeile aus der Quelltabelle wird eingefügt.

Erstellen Sie zunächst die Quell- und Zieltabellen und füllen Sie sie aus.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (101, 'Bob'), (102, 'John'), (103, 'Susan');
INSERT INTO source VALUES (102, 'Tony'), (103, 'Alice'), (104, 'Bill');

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 102 | John
 103 | Susan
(3 rows)

SELECT * FROM source;
 id  |    name
-----+------------
 102 | Tony
 103 | Alice
 104 | Bill
(3 rows)
```

Führen Sie anschließend die Quelltabelle mit der Zieltabelle zusammen. Aktualisieren Sie dabei die Zieltabelle mit übereinstimmenden Zeilen und fügen Sie Zeilen aus der Quelltabelle ein, zu denen es keine Übereinstimmung gibt.

```
MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN UPDATE SET id = source.id, name = source.name
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 102 | Tony
 103 | Alice
 104 | Bill
(4 rows)
```

Beachten Sie, dass die Zeilen mit den ID-Werten 102 und 103 aktualisiert werden, sodass sie den Namenswerten aus der Zieltabelle entsprechen. Außerdem wird eine neue Zeile mit dem ID-Wert 104 und dem Namenswert „Bill“ in die Zieltabelle eingefügt.

Fügen Sie als Nächstes eine neue Zeile in die Quelltabelle ein.

```
INSERT INTO source VALUES (105, 'David');

SELECT * FROM source;
 id  |    name
-----+------------
 102 | Tony
 103 | Alice
 104 | Bill
 105 | David
(4 rows)
```

Führen Sie abschließend eine Merge-Operation aus. Löschen Sie dabei übereinstimmende Zeilen in der Zieltabelle und fügen Sie nicht übereinstimmende Zeilen ein.

```
MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN DELETE
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 105 | David
(2 rows)
```

Die Zeilen mit den ID-Werten 102, 103 und 104 werden aus der Zieltabelle gelöscht, und eine neue Zeile mit dem ID-Wert 105 und dem Namenswert „David“ wird in die Zieltabelle eingefügt.

Das folgende Beispiel zeigt einen MERGE-Befehl unter Verwendung der REMOVE DUPLICATES-Klausel.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (30, 'Tony'), (11, 'Alice'), (23, 'Bill');
INSERT INTO source VALUES (23, 'David'), (22, 'Clarence');

MERGE INTO target USING source ON target.id = source.id REMOVE DUPLICATES;

SELECT * FROM target;
```

```
id | name
---+------------
30 | Tony
11 | Alice
23 | David
22 | Clarence
(4 rows)
```

Das folgende Beispiel zeigt einen MERGE-Befehl, der die REMOVE DUPLICATES-Klausel benutzt und doppelte Zeilen aus target_table entfernt, wenn sie übereinstimmende Zeilen in source_table haben.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (30, 'Tony'), (30, 'Daisy'), (11, 'Alice'), (23, 'Bill'),
 (23, 'Nikki');
INSERT INTO source VALUES (23, 'David'), (22, 'Clarence');

MERGE INTO target USING source ON target.id = source.id REMOVE DUPLICATES;

SELECT * FROM target;
id | name
---+------------
30 | Tony
30 | Daisy
11 | Alice
23 | David
22 | Clarence
(5 rows)
```

Nach der Ausführung von MERGE gibt es in target_table nur eine Zeile mit dem ID-Wert 23. Da es in source_table keine Zeile mit dem ID-Wert 30 gab, verbleiben die beiden doppelten Zeilen mit den ID-Werten 30 in target_table.

Weitere Informationen finden Sie auch unter

INSERT, UPDATE, DELETE

PREPARE

Bereitet eine Anweisung für die Ausführung vor.

PREPARE erstellt eine vorbereitete Anweisung. Wenn die PREPARE-Anweisung ausgeführt wird, wird die angegebene Anweisung (SELECT, INSERT, UPDATE oder DELETE) analysiert, neu geschrieben und geplant. Wenn dann ein EXECUTE-Befehl für die vorbereitete Anweisung ausgegeben wird, kann Amazon Redshift den Abfrageausführungsplan optional ändern (um die Leistung auf der Basis der angegebenen Parameterwerte zu verbessern), bevor die vorbereitete Anweisung ausgeführt wird.

Syntax

```
PREPARE plan_name [ (datatype [, ...] ) ] AS statement
```

Parameter

plan_name

Ein zufällig ausgewählter Name, der dieser bestimmten vorbereiteten Anweisung gegeben wird. Er muss innerhalb einer einzelnen Sitzung eindeutig sein und wird anschließend verwendet, um eine zuvor vorbereitete Anweisung auszuführen oder deren Zuteilung aufzuheben.

datatype

Der Datentyp eines Parameters der vorbereiteten Anweisung. Um auf die Parameter in der vorbereiteten Anweisung selbst zu verweisen, verwenden Sie $1, $2 usw.

Nachricht sehen

Eine SELECT-, INSERT-, UPDATE- oder DELETE-Anweisung.

Nutzungshinweise

Vorbereitete Anweisungen können Parameter besitzen: Werte, die in die Anweisung ersetzt werden, wenn sie ausgeführt wird. Um Parameter in eine vorbereitete Anweisung einzuschließen, stellen Sie in der PREPARE-Anweisung eine Liste von Datentypen bereit und verweisen Sie in der vorzubereitenden Anweisung selbst auf die Parameter nach Position, in der Schreibweise $1, $2... Wenn die Anweisung ausgeführt wird, geben Sie die tatsächlichen Werte für diese Parameter in der Anweisung EXECUTE an. Weitere Details finden Sie unter EXECUTE.

Vorbereitete Anweisungen sind nur für die Dauer der aktuellen Sitzung gültig. Nach Ende der Sitzung wird die vorbereitete Anweisung verworfen. Daher muss sie erneut erstellt werden, um erneut

verwendet zu werden. Das bedeutet auch, dass eine einzelne vorbereitete Anweisung nicht von mehreren Datenbankclients gleichzeitig verwendet werden kann. Jeder Client kann jedoch eine eigene vorbereitete Anweisung zur Verwendung erstellen. Die vorbereitete Anweisung kann mittels des Befehls DEALLOCATE manuell entfernt werden.

Vorbereitete Anweisungen bieten den größten Leistungsvorteil, wenn eine einzelne Sitzung zur Ausführung einer großen Zahl ähnlicher Anweisungen verwendet wird. Wie bereits erwähnt, kann Amazon Redshift für jede neue Ausführung einer vorbereiteten Anweisung den Abfrageausführungsplan ändern, um die Leistung auf der Basis der angegebenen Parameterwerte zu verbessern. Um den Abfrageausführungsplan zu überprüfen, den Amazon Redshift für spezifische EXECUTE-Anweisungen ausgewählt hat, verwenden Sie den Befehl EXPLAIN.

Weitere Informationen zur Abfrageplanung und zu den Statistiken, die von Amazon Redshift für die Optimierung von Abfragen gesammelt werden, finden Sie unter dem Befehl ANALYZE.

Beispiele

Erstellung einer temporären Tabelle, Vorbereitung einer INSERT-Anweisung vor und anschließende Ausführung:

```
DROP TABLE IF EXISTS prep1;
CREATE TABLE prep1 (c1 int, c2 char(20));
PREPARE prep_insert_plan (int, char)
AS insert into prep1 values ($1, $2);
EXECUTE prep_insert_plan (1, 'one');
EXECUTE prep_insert_plan (2, 'two');
EXECUTE prep_insert_plan (3, 'three');
DEALLOCATE prep_insert_plan;
```

Vorbereitung einer SELECT-Anweisung und anschließende Ausführung:

```
PREPARE prep_select_plan (int)
AS select * from prep1 where c1 = $1;
EXECUTE prep_select_plan (2);
EXECUTE prep_select_plan (3);
DEALLOCATE prep_select_plan;
```

Weitere Informationen finden Sie auch unter

DEALLOCATE, EXECUTE

REFRESH MATERIALIZED VIEW

Aktualisiert eine materialisierte Ansicht.

Wenn Sie eine materialisierte Ansicht erstellen, spiegelt ihr Inhalt den Zustand der zugrundeliegenden Datenbanktabelle(n) zu diesem Zeitpunkt wider. Die Daten in der materialisierten Ansicht bleiben unverändert, auch wenn Anwendungen Änderungen an den Daten in den zugrundeliegenden Tabellen vornehmen. Um die Daten in einer materialisierten Ansicht zu aktualisieren, können Sie jederzeit die Anweisung `REFRESH MATERIALIZED VIEW` verwenden. Wenn Sie diese Anweisung verwenden, identifiziert Amazon Redshift Änderungen, die in der oder den Basistabellen stattgefunden haben, und wendet diese Änderungen dann auf die materialisierte Ansicht an.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

Syntax

```
REFRESH MATERIALIZED VIEW mv_name
```

Parameter

mv_name

Der Name der zu aktualisierenden materialisierten Ansicht.

Nutzungshinweise

Nur der Besitzer einer materialisierten Ansicht kann eine `REFRESH MATERIALIZED VIEW`-Operation für diese Ansicht ausführen. Darüber hinaus muss der Besitzer über SELECT-Berechtigungen für die zugrunde liegenden Basistabellen verfügen, um erfolgreich ausführen zu können `REFRESH MATERIALIZED VIEW`.

Der Befehl `REFRESH MATERIALIZED VIEW` wird als Transaktion von sich selbst ausgeführt. Die Amazon-Redshift-Transaktionssemantik wird befolgt, um zu bestimmen, welche Daten aus Basistabellen für den Befehl `REFRESH` sichtbar sind oder wann die durch den Befehl `REFRESH` vorgenommenen Änderungen für andere Transaktionen sichtbar gemacht werden, die in Amazon Redshift ausgeführt werden.

- Für inkrementelle materialisierte Ansichten verwendet `REFRESH MATERIALIZED VIEW` nur die Zeilen der Basistabelle, die bereits festgeschrieben wurden. Wenn der Aktualisierungsvorgang nach einer DML-Anweisung (Data Manipulation Language) in derselben Transaktion ausgeführt wird, sind Änderungen dieser DML-Anweisung nicht sichtbar, um zu aktualisieren.
- Für eine vollständige Aktualisierung einer materialisierten Ansicht zeigt `REFRESH MATERIALIZED VIEW` alle Basistabellenzeilen an, die für die Aktualisierungstransaktion sichtbar sind, entsprechend der üblichen Amazon-Redshift-Transaktionssemantik.
- Abhängig vom Eingabeargumenttyp unterstützt Amazon Redshift weiterhin die inkrementelle Aktualisierung für materialisierte Ansichten für die folgenden Funktionen mit bestimmten Eingabeargumenttypen: DATE (Zeitstempel), DATE_PART (Datum, Uhrzeit, Intervall, Zeitzone), DATE_TRUNC (Zeitstempel, Intervall).
- Die inkrementelle Aktualisierung wird auch für eine materialisierte Ansicht unterstützt, bei der sich die Basistabelle in einem Datashare befindet.

Einige Operationen in Amazon Redshift interagieren mit materialisierten Ansichten. Einige dieser Vorgänge erzwingen möglicherweise einen Vorgang vom Typ `REFRESH MATERIALIZED VIEW`, um die materialisierte Ansicht vollständig neu zu berechnen, auch wenn die Abfrage, die die materialisierte Ansicht definiert, nur die SQL-Features verwendet, die sich für eine inkrementelle Aktualisierung eignen. Beispiel:

- Im Hintergrund ausgeführte Bereinigungsoperationen können blockiert werden, wenn materialisierte Ansichten nicht aktualisiert werden. Nach einem intern definierten Zeitraum kann eine Bereinigungsoperation ausgeführt werden. Bei dieser Bereinigungsoperation werden alle abhängigen materialisierten Ansichten bei der nächsten Aktualisierung zur Neuberechnung markiert (auch wenn sie inkrementell sind). Weitere Informationen zu VACUUM finden Sie unter VACUUM. Weitere Hinweise zu Ereignissen und Statusänderungen finden Sie unter STL_MV_STATE.
- Einige vom Benutzer initiierte Operationen für Basistabellen zwingen eine materialisierte Ansicht dazu, bei der nächsten Ausführung einer REFRESH-Operation vollständig neu berechnet zu werden. Beispiele für solche Operationen sind eine manuell aufgerufene VACUUM-Operation, eine klassische Größenänderung, eine ALTER DISTKEY-Operation, eine ALTER SORTKEY-Operation und eine Operation zum Kürzen. Weitere Hinweise zu Ereignissen und Statusänderungen finden Sie unter STL_MV_STATE.

Inkrementelle Aktualisierung für materialisierte Ansichten in einer Datenfreigabe

Amazon Redshift unterstützt automatische und inkrementelle Aktualisierungen für materialisierte Ansichten in einem Consumer-Datashare, wenn die Basistabellen gemeinsam genutzt werden. Inkrementelle Aktualisierung ist ein Vorgang, bei dem Amazon Redshift Änderungen in der Basistabelle oder den Basistabellen identifiziert, die nach der vorherigen Aktualisierung vorgenommen wurden, und nur die entsprechenden Datensätze in der materialisierten Ansicht aktualisiert. Weitere Informationen zu diesem Verhalten finden Sie unter CREATE MATERIALIZED VIEW.

Einschränkungen für die inkrementelle Aktualisierung

Amazon Redshift unterstützt derzeit keine inkrementelle Aktualisierung für materialisierte Ansichten, die mit einer Abfrage mit einem der folgenden SQL-Elemente definiert sind:

- OUTER JOIN (RIGHT, LEFT oder FULL).
- Set-Operationen: UNION, INTERSECT, EXCEPT, MINUS.
- UNION ALL, wenn es in einer Unterabfrage vorkommt und eine Aggregatfunktion oder eine GROUP BY-Klausel in der Abfrage vorhanden ist.
- Aggregatfunktionen: MEDIAN, PERCENTILE_CONT, LISTAGG, STDDEV_SAMP, STDDEV_POP, APPROXIMATE COUNT, APPROXIMATE PERCENTILE sowie bitweise Aggregatfunktionen.

  > **Note**
  >
  > Die Aggregatfunktionen COUNT, SUM, MIN, MAX und AVG werden unterstützt.

- DISTINCT-Aggregatfunktionen, wie DISTINCT COUNT, DISTINCT SUM usw.
- Fensterfunktionen.
- Eine Abfrage, die temporäre Tabellen für die Abfrageoptimierung verwendet, z. B. das Optimieren allgemeiner Unterausdrücke.
- Unterabfragen
- Externe Tabellen, die in der Abfrage, die die materialisierte Ansicht definiert, auf die folgenden Formate verweisen.
  - Delta Lake
  - Hudi

Die inkrementelle Aktualisierung wird für materialisierte Ansichten unterstützt, die externe Tabellen verwenden, die im Vorschau-Track auf andere Formate verweisen. Weitere Informationen zum Einrichten von Vorschau-Clustern finden Sie unter Erstellen eines Vorschau-Clusters im Amazon-Redshift-Verwaltungshandbuch. Informationen zum Einrichten von Vorschau-Arbeitsgruppen finden Sie unter Erstellen einer Vorschau-Arbeitsgruppe im Amazon-Redshift-Verwaltungshandbuch.

- Veränderbare Funktionen, wie Datum-Uhrzeit-Funktionen, RANDOM und nicht-STABLE benutzerdefinierte Funktionen.
- Einschränkungen in Bezug auf die inkrementelle Aktualisierung für Zero-ETL-Integrationen finden Sie unter Überlegungen bei der Verwendung von Zero-ETL-Integrationen mit Amazon Redshift.

Weitere Informationen zu Einschränkungen bei materialisierten Ansichten, einschließlich der Auswirkungen von Hintergrundoperationen wie VACUUM auf Aktualisierungsvorgänge in materialisierten Ansichten, finden Sie unter Nutzungshinweise.

Beispiele

Das folgende Beispiel aktualisiert die materialisierte `tickets_mv`-Ansicht.

```
REFRESH MATERIALIZED VIEW tickets_mv;
```

RESET

Setzt den Wert eines Konfigurationsparameter auf den Standardwert zurück.

Sie können einen einzelnen angegebenen Parameter oder alle Parameter auf einmal zurücksetzen. Um einen Parameter auf einen spezifischen Wert festzulegen, verwenden Sie den Befehl SET. Um den aktuellen Wert eines Parameters anzuzeigen, verwenden Sie den Befehl ZEIGEN.

Syntax

```
RESET { parameter_name | ALL }
```

Die folgende Anweisung legt den Wert einer Sitzungskontextvariablen auf NULL fest.

```
RESET { variable_name | ALL }
```

Parameter

parameter_name

Der Name des Parameters, der zurückgesetzt werden soll. Weitere Informationen zu Parametern finden Sie unter Modifizieren der Serverkonfiguration.

ALL

Setzt alle Laufzeitparameter zurück, einschließlich aller Sitzungskontextvariablen.

Variable

Der Name der Variablen, die zurückgesetzt werden soll. Wenn der Wert für RESET eine Sitzungskontextvariable ist, setzt Amazon Redshift ihn auf NULL.

Beispiele

Im folgenden Beispiel wird der Parameter `query_group` auf den Standardwert zurückgesetzt:

```
reset query_group;
```

Im folgenden Beispiel werden alle Laufzeitparameter auf die Standardwerte zurückgesetzt.

```
reset all;
```

Im folgenden Beispiel wird die Kontextvariable zurückgesetzt.

```
RESET app_context.user_id;
```

REVOKE

Entfernt Zugriffsberechtigungen wie beispielsweise die Berechtigungen zum Erstellen, Entfernen oder Aktualisieren von Tabellen von einem Benutzer oder einer Rolle.

Sie können nur mit der ON SCHEMA-Syntax GRANT- oder REVOKE USAGE-Berechtigungen auf einem externen Schema für Datenbankbenutzern und Rollen einrichten. Wenn Sie ON EXTERNAL SCHEMA mit verwenden AWS Lake Formation, können Sie nur Berechtigungen für eine (IAM-) Rolle ERTEILEN und WIDERRUFEN. AWS Identity and Access Management Eine Liste der Berechtigungen finden Sie in der Syntax.

Für gespeicherte Prozeduren werden die Berechtigungen USAGE ON LANGUAGE `plpgsql` standardmäßig PUBLIC gewährt. Die Berechtigung EXECUTE ON PROCEDURE wird standardmäßig nur dem Besitzer und Superusern gewährt.

Geben Sie im Befehl REVOKE die Berechtigungen an, die Sie entfernen möchten. Verwenden Sie den Befehl GRANT, um Berechtigungen zu erteilen.

Syntax

```
REVOKE [ GRANT OPTION FOR ]
{ { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | ALTER | TRUNCATE } [,...] |
 ALL [ PRIVILEGES ] }
ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]


REVOKE [ GRANT OPTION FOR ]
{ { CREATE | TEMPORARY | TEMP | ALTER } [,...] | ALL [ PRIVILEGES ] }
ON DATABASE db_name [, ...]
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]


REVOKE [ GRANT OPTION FOR ]
{ { CREATE | USAGE | ALTER } [,...] | ALL [ PRIVILEGES ] }
ON SCHEMA schema_name [, ...]
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]


REVOKE [ GRANT OPTION FOR ]
EXECUTE
    ON FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]


REVOKE [ GRANT OPTION FOR ]
{ { EXECUTE } [,...] | ALL [ PRIVILEGES ] }
    ON PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]
```

```
REVOKE [ GRANT OPTION FOR ]
USAGE
    ON LANGUAGE language_name [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]
```

Widerrufen von Berechtigungen auf Spaltenebene für Tabellen

Im Folgenden finden Sie die Syntax für Berechtigungen auf Spaltenebene für Amazon-Redshift-Tabellen und -Ansichten.

```
REVOKE { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
     ON { [ TABLE ] table_name [, ...] }
     FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
     [ RESTRICT ]
```

Widerrufen von Berechtigungen ASSUMEROLE

Im Folgenden finden Sie die Syntax zum Widerrufen der Berechtigung ASSUMEROLE von Benutzern und Gruppen mit einer spezifizierten Rolle.

```
REVOKE ASSUMEROLE
    ON { 'iam_role' [, ...]  | default | ALL }
    FROM { user_name | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL }
```

Widerrufen von Berechtigungen für Redshift Spectrum für Lake Formation

Im Folgenden finden Sie die Syntax für die Integration von Redshift Spectrum in Lake Formation.

```
REVOKE [ GRANT OPTION FOR ]
{ SELECT | ALL [ PRIVILEGES ] } ( column_list )
    ON EXTERNAL TABLE schema_name.table_name
    FROM { IAM_ROLE iam_role } [, ...]

REVOKE [ GRANT OPTION FOR ]
{ { SELECT | ALTER | DROP | DELETE | INSERT }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL TABLE schema_name.table_name [, ...]
```

```
    FROM { { IAM_ROLE iam_role } [, ...] | PUBLIC }

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | ALTER | DROP }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL SCHEMA schema_name [, ...]
    FROM { IAM_ROLE iam_role } [, ...]
```

Widerrufen von Datashare-Berechtigungen

Produzentenseitige Datashare-Berechtigungen

Im Folgenden finden Sie die Syntax zur Verwendung von REVOKE zum Entfernen von ALTER- oder SHARE-Berechtigungen von einem Benutzer oder einer Rolle. Der Benutzer, dessen Berechtigungen widerrufen wurden, kann das Datashare nicht mehr ändern oder einem Konsumenten dessen Nutzung gewähren.

```
REVOKE { ALTER | SHARE } ON DATASHARE datashare_name
 FROM { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Im Folgenden finden Sie die Syntax zur Verwendung von REVOKE, um einem Konsumenten den Zugriff auf ein Datashare zu verweigern.

```
REVOKE USAGE
 ON DATASHARE datashare_name
 FROM NAMESPACE 'namespaceGUID' [, ...] | ACCOUNT 'accountnumber' [ VIA DATA CATALOG ]
 [, ...]
```

Im Folgenden finden Sie ein Beispiel dafür, wie Sie die Berechtigung eines Lake-Formation-Kontos zur Nutzung eines Datashares widerrufen können.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012' VIA DATA CATALOG;
```

Konsumentenseitige Datashare-Berechtigungen

Im Folgenden finden Sie die REVOKE-Syntax für Datashare-Nutzungsberechtigungen für eine bestimmte Datenbank oder ein bestimmtes Schema, die auf einem Datashare erstellt wurden. Durch den Widerruf der Nutzungsberechtigung für eine Datenbank, die mit der WITH PERMISSIONS-Klausel erstellt wurde, werden keine zusätzlichen Berechtigungen, die Sie einem Benutzer oder

einer Rolle erteilt haben, entzogen, einschließlich Berechtigungen auf Objektebene, die für zugrunde liegende Objekte gewährt wurden. Wenn Sie diesem Benutzer oder dieser Rolle erneut die Nutzungsberechtigung gewähren, behält dieser Benutzer oder diese Rolle alle zusätzlichen Berechtigungen bei, die er/sie hatte, bevor Sie die Nutzung widerrufen haben.

```
REVOKE USAGE ON { DATABASE shared_database_name [, ...] | SCHEMA shared_schema}
 FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Widerrufen von bereichsbezogenen Berechtigungen

Mit bereichsbezogenen Berechtigungen können Sie einem Benutzer oder einer Rolle Berechtigungen für alle Objekte eines Typs innerhalb einer Datenbank oder eines Schemas gewähren. Benutzer und Rollen mit bereichsbezogenen Berechtigungen verfügen über die angegebenen Berechtigungen für alle aktuellen und future Objekte innerhalb der Datenbank oder des Schemas.

Im Folgenden sehen Sie die Syntax zum Widerrufen bereichsbezogener Berechtigungen für Benutzer oder Rollen. Weitere Hinweise zu bereichsbezogenen Berechtigungen finden Sie unter. Bereichsbeschränkte Berechtigungen

```
REVOKE [ GRANT OPTION ]
{ CREATE | USAGE | ALTER } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE db_name
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ]
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }
FOR TABLES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username] | ROLE role_name} [, ...]

REVOKE [ GRANT OPTION ] { EXECUTE | ALL [ PRIVILEGES ] }
FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name | } [, ...]

REVOKE [ GRANT OPTION ] { EXECUTE | ALL [ PRIVILEGES ] }
FOR PROCEDURES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name | } [, ...]
```

```
REVOKE [ GRANT OPTION ] USAGE
FOR LANGUAGES IN
{DATABASE db_name}
FROM { username | ROLE role_name } [, ...]
```

Beachten Sie, dass bei bereichsbezogenen Berechtigungen nicht zwischen Berechtigungen für Funktionen und für Prozeduren unterschieden wird. Mit der folgenden Anweisung werden beispielsweise EXECUTE Berechtigungen für Funktionen und Prozeduren aus dem bob Schema entzogen. `Sales_schema`

```
REVOKE EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema FROM bob;
```

Widerrufen von Machine-Learning-Berechtigungen

Im Folgenden finden Sie die Syntax für Berechtigungen für Machine-Learning-Modelle in Amazon Redshift.

```
REVOKE [ GRANT OPTION FOR ]
    CREATE MODEL FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
    { EXECUTE | ALL [ PRIVILEGES ] }
    ON MODEL model_name [, ...]

    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]
```

Widerrufen von Rollenberechtigungen

Im Folgenden finden Sie die Syntax zum Widerrufen von Rollenberechtigungen in Amazon Redshift.

```
REVOKE [ ADMIN OPTION FOR ] { ROLE role_name } [, ...] FROM { user_name } [, ...]
```

```
REVOKE { ROLE role_name } [, ...] FROM { ROLE role_name } [, ...]
```

Im Folgenden finden Sie die Syntax zum Widerrufen von Systemberechtigungen für Rollen in Amazon Redshift.

```
REVOKE
  {
    { CREATE USER | DROP USER | ALTER USER |
    CREATE SCHEMA | DROP SCHEMA |
    ALTER DEFAULT PRIVILEGES |
    ACCESS CATALOG |
    CREATE TABLE | DROP TABLE | ALTER TABLE |
    CREATE OR REPLACE FUNCTION | CREATE OR REPLACE EXTERNAL FUNCTION |
    DROP FUNCTION |
    CREATE OR REPLACE PROCEDURE | DROP PROCEDURE |
    CREATE OR REPLACE VIEW | DROP VIEW |
    CREATE MODEL | DROP MODEL |
    CREATE DATASHARE | ALTER DATASHARE | DROP DATASHARE |
    CREATE LIBRARY | DROP LIBRARY |
    CREATE ROLE | DROP ROLE
    TRUNCATE TABLE
    VACUUM | ANALYZE | CANCEL }[, ...]
  }
  | { ALL [ PRIVILEGES ] }
FROM { ROLE role_name } [, ...]
```

Widerrufen von Explain-Berechtigungen für RLS-Richtlinienfilter

Im Folgenden finden Sie die Syntax zum Entzug von Berechtigungen, um die RLS-Richtlinienfilter auf Zeilenebene einer Abfrage im EXPLAIN-Plan zu erläutern. Sie können die Berechtigung mit der REVOKE-Anweisung entziehen.

```
REVOKE EXPLAIN RLS FROM ROLE rolename
```

Im Folgenden finden Sie die Syntax zum Erteilen von Berechtigungen zum Umgehen von RLS-Richtlinien auf Zeilenebene für eine Abfrage.

```
REVOKE IGNORE RLS FROM ROLE rolename
```

Im Folgenden finden Sie die Syntax zum Entzug von Berechtigungen aus der angegebenen RLS-Richtlinie auf Zeilenebene.

```
REVOKE SELECT ON [ TABLE ] table_name [, ...]
            FROM RLS POLICY policy_name [, ...]
```

Parameter

GRANT OPTION FOR

Widerruft nur die Option, anderen Benutzern eine spezifische Berechtigung zu gewähren, und nicht die Berechtigung selbst. Sie können GRANT OPTION nicht für eine Gruppe oder PUBLIC widerrufen.

SELECT

Widerruft die Berechtigung, Daten aus einer Tabelle oder Ansicht mittels einer SELECT-Anweisung auszuwählen.

INSERT

Widerruft die Berechtigung, Daten in eine Tabelle mittels einer INSERT- oder COPY-Anweisung zu laden.

UPDATE

Widerruft die Berechtigung, eine Tabellenspalte mittels einer UPDATE-Anweisung zu aktualisieren.

DELETE

Widerruft die Berechtigung, eine Datenzeile aus einer Tabelle zu löschen.

REFERENCES

Widerruft die Berechtigung, eine Fremdschlüsseleinschränkung zu erstellen. Sie sollten diese Berechtigung sowohl für die referenzierte als auch für die referenzierende Tabelle widerrufen.

TRUNCATE

Widerruft die Berechtigung zum Kürzen einer Tabelle. Ohne diese Berechtigung kann nur der Eigentümer einer Tabelle oder ein Superuser eine Tabelle kürzen. Weitere Informationen zur Verwendung des TRUNCATE-Befehls finden Sie unter the section called “TRUNCATE”.

ALL [PRIVILEGES]

Widerruft für den angegebenen Benutzer oder die angegebenen Benutzergruppe alle verfügbaren Berechtigungen auf einmal. Das Schlüsselwort PRIVILEGES ist optional.

ALTER

Widerruft die folgenden Berechtigungen für den Benutzer oder die Benutzergruppe, abhängig vom Datenbankobjekt:

- Bei Tabellen widerruft ALTER die Erlaubnis, eine Tabelle oder Ansicht zu ändern. Weitere Informationen finden Sie unter ALTER TABLE.
- Bei Datenbanken widerruft ALTER die Erlaubnis, eine Datenbank zu ändern. Weitere Informationen finden Sie unter ALTER DATABASE.
- Bei Schemata widerruft ALTER die Erlaubnis zum Ändern eines Schemas. Weitere Informationen finden Sie unter ALTER SCHEMA.
- Für externe Tabellen widerruft ALTER die Erlaubnis, eine Tabelle in einer Tabelle zu ändern AWS Glue Data Catalog , die für Lake Formation aktiviert ist. Diese Berechtigung gilt nur bei Verwendung von Lake Formation.

DROP

Widerruft die Berechtigung zum Löschen einer Tabelle. Diese Berechtigung gilt in Amazon Redshift und in einer AWS Glue Data Catalog , die für Lake Formation aktiviert ist.

ASSUMEROLE

Widerruft die Berechtigung zum Ausführen der Befehle COPY, UNLOAD, EXTERNAL FUNCTION und CREATE MODEL für Benutzer, Rollen oder Gruppen mit einer angegebenen Rolle.

ON [TABLE] table_name

Widerruft die angegebenen Rechte für eine Tabelle oder Ansicht. Das Schlüsselwort TABLE ist optional.

ON ALL TABLES IN SCHEMA schema_name

Widerruft die angegebenen Rechte für alle Tabellen im referenzierten Schema.

(column_name [,...]) ON TABLE table_name

Widerruft die angegebenen Berechtigungen von Benutzern, Gruppen oder PUBLIC für die angegebenen Spalten der Amazon-Redshift-Tabelle oder -Ansicht.

(column_list) ON EXTERNAL TABLE schema_name.table_name

Widerruft die angegebenen Berechtigungen einer IAM-Rolle für die angegebenen Spalten der Lake Formation-Tabelle im referenzierten Schema.

ON EXTERNAL TABLE schema_name.table_name

Widerruft die angegebenen Berechtigungen einer IAM-Rolle für die angegebenen Lake Formation-Tabellen im referenzierten Schema.

ON EXTERNAL SCHEMA schema_name

Widerruft die angegebenen Berechtigungen einer IAM-Rolle für das referenzierte Schema.

FROM IAM_ROLE iam_role

Gibt die IAM-Rolle an, die die Berechtigungen verliert.

ROLE role_name

Widerruft die Berechtigungen für die angegebene Rolle.

GROUP group_name

Widerruft die Berechtigungen für den angegebenen Benutzer oder die angegebene Benutzergruppe.

PUBLIC

Widerruft die angegebenen Berechtigungen für alle Benutzer. PUBLIC stellt eine Gruppe dar, die stets alle Benutzer enthält. Die Berechtigungen eines einzelnen Benutzers sind die Summe der Berechtigungen, die PUBLIC gewährt werden, der Berechtigungen, die Gruppen gewährt werden, zu denen der Benutzer gehört, und der Berechtigungen, die dem einzelnen Benutzer gewährt werden.

Indem Sie die Berechtigung PUBLIC für eine externe Lake-Formation-Tabelle widerrufen, widerrufen Sie die Berechtigung für die Lake-Formation-Gruppe everyone.

CREATE

Widerruft die folgenden Berechtigungen für den Benutzer oder die Benutzergruppe, abhängig vom Datenbankobjekt:

- Im Fall von Datenbanken verhindert die Verwendung der CREATE-Klausel für REVOKE, dass Benutzer innerhalb der Datenbank Schemata erstellen.
- Im Fall von Schemata verhindert die Verwendung der CREATE-Klausel für REVOKE, dass Benutzer innerhalb eines Schemas Objekte erstellen. Um ein Objekt umzubenennen, muss der Benutzer über die Berechtigung CREATE verfügen und Eigentümer des Objekts sein, das umbenannt werden soll.

> **Note**
>
> Standardmäßig besitzen alle Benutzer CREATE- und USAGE-Berechtigungen für das Schema PUBLIC einer Datenbank.

TEMPORARY | TEMP

Widerruft die Berechtigung, in der angegebenen Datenbank temporäre Tabellen zu erstellen.

> **Note**
>
> Standardmäßig wird Benutzern aufgrund ihrer automatischen Mitgliedschaft in der Gruppe PUBLIC die Berechtigung gewährt, temporäre Tabellen zu erstellen. Um die Berechtigung zum Erstellen temporärer Tabellen für alle Benutzer zu entfernen, widerrufen Sie die Berechtigung TEMP für die Gruppe PUBLIC und gewähren Sie dann explizit die Berechtigung zum Erstellen temporärer Tabellen für spezifische Benutzer oder Benutzergruppen.

ON DATABASE db_name

Widerruft die Berechtigungen für die angegebene Datenbank.

USAGE

Widerruft Berechtigungen USAGE für Objekte innerhalb eines bestimmten Schemas, wodurch diese Objekte für Benutzer nicht zugänglich sind. Spezifische Aktionen für diese Objekte müssen getrennt widerrufen werden (z. B. die Berechtigung EXECUTE für Funktionen).

> **Note**
>
> Standardmäßig besitzen alle Benutzer CREATE- und USAGE-Berechtigungen für das Schema PUBLIC einer Datenbank.

ON SCHEMA schema_name

Widerruft die Berechtigungen für das angegebene Schema. Sie können Schemaberechtigungen verwenden, um die Erstellung von Tabellen zu steuern. Die Berechtigung CREATE für eine Datenbank steuert nur die Erstellung von Schemata.

RESTRICT

Widerruft nur die Berechtigungen, die der Benutzer direkt erteilt hat. Dieses Verhalten ist das Standardverhalten.

EXECUTE ON PROCEDURE procedure_name

Widerruft die Berechtigung EXECUTE für eine spezifische gespeicherte Prozedur. Da die Namen gespeicherter Prozeduren überladen sein können, müssen Sie die Argumentliste für die Prozedur angeben. Weitere Informationen finden Sie unter Benennen von gespeicherten Prozeduren.

EXECUTE ON ALL PROCEDURES IN SCHEMA procedure_name

Widerruft die angegebenen Berechtigungen für alle Prozeduren im referenzierten Schema.

USAGE ON LANGUAGE language_name

Widerruft die Berechtigung USAGE für eine Sprache. Für Python-UDFs verwenden Sie `plpythonu`. Für SQL-UDFs verwenden Sie `sql`. Für gespeicherte Prozeduren verwenden Sie `plpgsql`.

Zum Erstellen einer UDF müssen Sie über die Berechtigung für die Sprachennutzung für SQL oder `plpythonu` (Python) verfügen. USAGE ON LANGUAGE SQL wird standardmäßig PUBLIC gewährt. USAGE ON LANGUAGE PLPYTHONU müssen Sie jedoch spezifischen Benutzern oder Gruppen explizit gewähren.

Um die Nutzung für SQL zu widerrufen, widerrufen Sie zuerst die Nutzung von PUBLIC. Erlauben Sie dann die Nutzung in SQL nur spezifischen Benutzern oder Gruppen, die auch die Erlaubnis zum Erstellen von SQL-UDFs besitzen. Im folgenden Beispiel wird die Nutzung in SQL von PUBLIC widerrufen. Anschließend wird die Nutzung der Benutzergruppe erlaubt `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Weitere Informationen finden Sie unter UDF-Sicherheit und Rechte.

Um die Nutzung für gespeicherte Prozeduren zu widerrufen, widerrufen Sie zuerst die Nutzung von PUBLIC. Erlauben Sie dann die Nutzung von `plpgsql` nur spezifischen Benutzern oder Gruppen, die auch gespeicherte Prozeduren erstellen dürfen. Weitere Informationen finden Sie unter Sicherheit und Berechtigungen für gespeicherte Prozeduren .

FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]

Gibt den SQL-Befehl an, für den die Berechtigung widerrufen wird. Sie können ALL angeben, um die Berechtigung für die Anweisungen COPY, UNLOAD, EXTERNAL FUNCTION und CREATE MODEL zu widerrufen. Diese Klausel gilt nur für das Widerrufen der Berechtigung ASSUMEROLE.

ALTER

Widerruft die Berechtigung ALTER für Benutzer oder Benutzergruppen, die es ihnen ermöglicht, ein Datashare zu ändern, das ihnen nicht gehört. Diese Berechtigung ist erforderlich, um Objekte zu einem Datashare hinzuzufügen oder daraus zu entfernen oder um die Eigenschaft PUBLICACCESSIBLE festzulegen. Weitere Informationen finden Sie unter ALTER DATASHARE.

SHARE

Widerruft Berechtigungen von Benutzern und Benutzergruppen zum Hinzufügen von Konsumenten zu einem Datashare. Das Widerrufen dieser Berechtigung ist erforderlich, um den Zugriff des jeweiligen Konsumenten auf das Datashare von seinen Clustern aus zu unterbinden.

ON DATASHARE datashare_name

Erteilt die angegebenen Berechtigungen für das referenzierte Datashare.

FROM Benutzername

Gibt den Benutzer an, der die Berechtigungen verliert.

FROM GROUP group_name

Gibt die Benutzergruppe an, die die Berechtigungen verliert.

WITH GRANT OPTION

Gibt an, dass der Benutzer, der die Berechtigungen verliert, dieselben Berechtigungen von anderen Benutzern widerrufen kann. Sie können WITH GRANT OPTION nicht für eine Gruppe oder PUBLIC widerrufen.

USAGE

Wenn USAGE einem Konsumenten-Konto oder einem Namespace innerhalb desselben Kontos entzogen wird, kann dieses Konsumenten-Konto oder der Namespace innerhalb des Kontos nicht mehr schreibgeschützt auf das Datashare und die Objekte auf dem Datashare zugreifen.

Wenn Sie die Berechtigung USAGE widerrufen, wird Konsumenten der Zugriff auf ein Datashare entzogen.

FROM NAMESPACE 'clusternamespace GUID'

Gibt den Namespace in demselben Konto an, in dem Konsumenten die Berechtigungen für das Datashare verlieren. Namespaces verwenden einen 128 Bit großen alphanumerischen globalen eindeutigen Bezeichner (Globally Unique Identifier, GUID).

FROM ACCOUNT 'Kontonummer' [VIA DATA CATALOG]

Gibt die Kontonummer eines anderen Kontos an, in dem Konsumenten die Berechtigungen für das Datashare verlieren. Die Angabe „VIA DATA CATALOG“ bedeutet, dass Sie die Berechtigung eines Lake-Formation-Kontos zur Nutzung des Datashares widerrufen. Wenn Sie die Kontonummer weglassen, bedeutet das, dass Sie die Berechtigung des Kontos widerrufen, dem der Cluster gehört.

ON DATABASE shared_database_name> [, ...]

Widerruft die angegebenen Nutzungsberechtigungen für die angegebene Datenbank, die im angegebenen Datashare erstellt wurde.

ON SCHEMA shared_schema

Widerruft die angegebenen Berechtigungen für das angegebene Schema, das im angegebenen Datashare erstellt wurde.

FOR { SCHEMAS | TABLES | FUNCTIONS | PROCEDURES | LANGUAGES } IN

Gibt die Datenbankobjekte an, für die Berechtigungen widerrufen werden sollen. Die Parameter nach IN definieren den Bereich der widerrufenen Berechtigung.

CREATE MODEL

Widerruft die Berechtigung CREATE MODEL, mit der sich Machine-Learning-Modelle in der angegebenen Datenbank erstellen lassen.

ON MODEL model_name

Widerruft die Berechtigung EXECUTE für ein spezifisches Modell.

ACCESS CATALOG

Widerruft die Berechtigung, relevante Metadaten von Objekten anzuzeigen, auf die die Rolle Zugriff hat.

[ADMIN OPTION FOR] { role } [, ...]

Die Rolle, die Sie für einen bestimmten Benutzer widerrufen, der über die WITH ADMIN-OPTION verfügt.

FROM { role } [, ...]

Die Rolle, für die Sie die angegebene Rolle widerrufen.

Nutzungshinweise

Weitere Informationen zu den Nutzungshinweisen für REVOKE finden Sie unter the section called “Nutzungshinweise”.

Beispiele

Beispiele für die Verwendung von REVOKE finden Sie unter the section called “Beispiele”.

Nutzungshinweise

Um Rechte für ein Objekt zu widerrufen, müssen Sie mindestens eines der folgenden Kriterien erfüllen:

- Sie müssen der Besitzer des Objekts sein.
- Sie müssen ein Superuser sein.
- Sie müssen das Recht besitzen, Rechte für dieses Objekt und Recht zu gewähren.

  Der folgende Befehl gewährt beispielsweise dem Benutzer HR die erforderlichen Rechte, um SELECT-Befehle für die Mitarbeitertabelle ausführen, anderen Benutzern dieselben Recht gewähren und diese Rechte für andere Benutzer widerrufen zu können.

  ```
  grant select on table employees to HR with grant option;
  ```

  HR kann keine Rechte für andere Operationen als SELECT oder für andere Tabellen als die Mitarbeitertabelle widerrufen.

Superuser können auf alle Objekte zugreifen, unabhängig von GRANT- und REVOKE-Befehlen, mit denen Objektrechte festgelegt werden.

PUBLIC stellt eine Gruppe dar, die stets alle Benutzer enthält. Standardmäßig besitzen alle Mitglieder von PUBLIC CREATE- und USAGE-Berechtigungen für das Schema PUBLIC. Zum Einschränken sämtlicher Benutzerberechtigungen im PUBLIC-Schema müssen Sie zuerst alle Berechtigungen von PUBLIC im PUBLIC-Schema widerrufen und die Berechtigungen dann spezifischen Benutzern oder Gruppen erteilen. Im folgenden Beispiel werden Rechte für die Tabellenerstellung im Schema PUBLIC gesteuert.

```
revoke create on schema public from public;
```

Um Berechtigungen für eine Lake Formation-Tabelle zu widerrufen, muss die mit dem externen Schema der Tabelle verknüpfte IAM-Rolle über die Berechtigung verfügen, Rechte für die externe Tabelle zu widerrufen. Im folgenden Beispiel wird ein externes Schema mit einer zugeordneten IAM-Rolle erstellt `myGrantor`. Die IAM-Rolle `myGrantor` verfügt über die Berechtigung, anderen Berechtigungen zu widerrufen. Der Befehl REVOKE nutzt die Berechtigung der IAM-Rolle `myGrantor`, die dem externen Schema zugeordnet ist, um Berechtigungen der IAM-Rolle `myGrantee` zu widerrufen.

```
create external schema mySchema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myGrantor'
create external database if not exists;
```

```
revoke select
on external table mySchema.mytable
from iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

> **Note**
>
> Wenn die IAM-Rolle auch über die `ALL` Berechtigung in einer verfügt AWS Glue Data Catalog , die für Lake Formation aktiviert ist, wird die `ALL` Berechtigung nicht widerrufen. Nur die Berechtigung `SELECT` wird widerrufen. Sie können die Lake Formation Berechtigungen in der Lake Formation-Konsole anzeigen.

Hinweise zum Widerrufen der Berechtigung ASSUMEROLE

Die folgenden Hinweise gelten für den Entzug der ASSUMEROLE-Berechtigung in Amazon Redshift.

Nur ein Datenbank-Superuser kann die Berechtigung ASSUMEROLE für Benutzer und Gruppen entziehen. Ein Superuser hat immer die Berechtigung ASSUMEROLE.

Um die Verwendung der ASSUMEROLE-Berechtigung für Benutzer und Gruppen zu aktivieren, führt ein Superuser die folgende Anweisung einmalig auf dem Cluster aus. Bevor die ASSUMEROLE-Berechtigung an Benutzer und Gruppen vergeben wird, muss ein Superuser die folgende Anweisung einmalig auf dem Cluster ausführen.

```
revoke assumerole on all from public for all;
```

Hinweise zum Widerrufen von Machine-Learning-Berechtigungen

Sie können Berechtigungen für eine ML-Funktion nicht direkt erteilen oder widerrufen. Eine ML-Funktion gehört zu einem ML-Modell, und die Berechtigungen werden über das Modell gesteuert. Stattdessen können Sie Berechtigungen im Zusammenhang mit dem ML-Modell widerrufen. Das folgende Beispiel zeigt, wie für alle Benutzer im Zusammenhang mit dem Modell `customer_churn` die Ausführungsberechtigung widerrufen wird.

```
REVOKE EXECUTE ON MODEL customer_churn FROM PUBLIC;
```

Sie können auch alle Berechtigungen für einen Benutzer für das ML-Modell `customer_churn` widerrufen.

```
REVOKE ALL on MODEL customer_churn FROM ml_user;
```

Das Erteilen oder Widerrufen der Berechtigung `EXECUTE` für eine ML-Funktion schlägt fehl, wenn das Schema eine ML-Funktion enthält, auch wenn diese ML-Funktion bereits über `GRANT EXECUTE ON MODEL` über die Berechtigung `EXECUTE` verfügt. Wir empfehlen, bei Verwendung des Befehls `CREATE MODEL` ein separates Schema zu verwenden, um die ML-Funktionen in einem separaten Schema zu behalten. Das folgende Beispiel veranschaulicht die Vorgehensweise hierfür.

```
CREATE MODEL ml_schema.customer_churn
FROM customer_data
TARGET churn
FUNCTION ml_schema.customer_churn_prediction
IAM_ROLE default
SETTINGS (
 S3_BUCKET 'your-s3-bucket'
);
```

Beispiele

Im folgenden Beispiel werden INSERT-Rechte für die Tabelle SALES für die Benutzergruppe GUESTS widerrufen. Dieser Befehl verhindert, dass Mitglieder von GUESTS mittels des Befehls INSERT Daten in die Tabelle SALES laden können.

```
revoke insert on table sales from group guests;
```

Im folgenden Beispiel wird für den Benutzer die SELECT-Berechtigung für alle Tabellen im Schema QA_TICKIT widerrufen `fred`.

```
revoke select on all tables in schema qa_tickit from fred;
```

Im folgenden Beispiel wird für den Benutzer das Recht widerrufen, aus einer Ansicht auszuwählen `bobr`.

```
revoke select on table eventview from bobr;
```

Im folgenden Beispiel wird für alle Benutzer das Recht widerrufen, in der TICKIT-Datenbank temporäre Tabellen zu erstellen.

```
revoke temporary on database tickit from public;
```

Im folgenden Beispiel wird dem Benutzer `cust_name` die SELECT-Berechtigung für die Spalten `cust_phone` und `cust_profile` der Tabelle `user1` entzogen.

```
revoke select(cust_name, cust_phone) on cust_profile from user1;
```

Im folgenden Beispiel wird die SELECT-Berechtigung für die Spalten `cust_name` und `cust_phone` und die UPDATE-Berechtigung für die Spalte `cust_contact_preference` der Tabelle `cust_profile` der Gruppe `sales_group` entzogen.

```
revoke select(cust_name, cust_phone), update(cust_contact_preference) on cust_profile
 from group sales_group;
```

Das folgende Beispiel zeigt die Verwendung des Schlüsselworts ALL, um SELECT- und UPDATE-Berechtigungen für drei Spalten der Tabelle `cust_profile` aus der `sales_admin` Gruppe zu entziehen.

```
revoke ALL(cust_name, cust_phone,cust_contact_preference) on cust_profile from group
 sales_admin;
```

Im folgenden Beispiel wird dem Benutzer `cust_name` die SELECT-Berechtigung für die Spalte `cust_profile_vw` der Ansicht `user2` entzogen.

```
revoke select(cust_name) on cust_profile_vw from user2;
```

Beispiele für den Entzug der USAGE-Berechtigung in Datenbanken, die aus Datashares erstellt wurden

Im folgenden Beispiel wird der Zugriff auf das `salesshare`-Datashare vom Namespace `13b8833d-17c6-4f16-8fe4-1a018f5ed00d` aus widerrufen.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Im folgenden Beispiel wird die USAGE-Berechtigung auf `sales_db` von Bob widerrufen.

```
REVOKE USAGE ON DATABASE sales_db FROM Bob;
```

Im folgenden Beispiel wird die REVOKE USAGE-Berechtigung auf `sales_schema` von `Analyst_role` verwendet.

```
REVOKE USAGE ON SCHEMA sales_schema FROM ROLE Analyst_role;
```

Beispiele für den Widerruf von bereichsbezogenen Berechtigungen

Im folgenden Beispiel wird für die `Sales`-Rolle die Verwendung aller derzeitigen und zukünftigen Schemata in der `Sales_db`-Datenbank widerrufen.

```
REVOKE USAGE FOR SCHEMAS IN DATABASE Sales_db FROM ROLE Sales;
```

Im folgenden Beispiel wird dem Benutzer `alice` die Fähigkeit entzogen, die SELECT-Berechtigung für alle derzeitigen und zukünftigen Tabellen in der `Sales_db`-Datenbank zu erteilen. `alice` behält den Zugriff auf alle Tabellen in `Sales_db` bei.

```
REVOKE GRANT OPTION SELECT FOR TABLES IN DATABASE Sales_db FROM alice;
```

Im folgenden Beispiel wird für den Benutzer bob die EXECUTE-Berechtigung für Funktionen im `Sales_schema`-Schema widerrufen.

```
REVOKE EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema FROM bob;
```

Im folgenden Beispiel werden für die `Sales`-Rolle alle Berechtigungen für alle Tabellen im Schema `ShareSchema` der Datenbank `ShareDb` widerrufen. Bei der Angabe des Schemas können Sie die Datenbank des Schemas auch im zweiteiligen Format `database.schema` angeben.

```
REVOKE ALL FOR TABLES IN SCHEMA ShareDb.ShareSchema FROM ROLE Sales;
```

Das folgende Beispiel zeigt dieselbe Abfrage wie im vorherigen Beispiel. Sie können die Datenbank des Schemas mit dem DATABASE-Schlüsselwort angeben, anstatt ein zweiteiliges Format zu verwenden.

```
REVOKE ALL FOR TABLES IN SCHEMA ShareSchema DATABASE ShareDb FROM ROLE Sales;
```

Beispiele für den Entzug der ASSUMEROLE-Berechtigung

Im Folgenden finden Sie Beispiele für den Entzug der ASSUMEROLE-Berechtigung.

Ein Superuser muss die Verwendung der ASSUMEROLE-Berechtigung für Benutzer und Gruppen aktivieren, indem er die folgende Anweisung einmal auf dem Cluster ausführt:

```
revoke assumerole on all from public for all;
```

Die folgende Anweisung entzieht dem Benutzer reg_user1 die ASSUMEROLE-Berechtigung für alle Rollen für alle Vorgänge.

```
revoke assumerole on all from reg_user1 for all;
```

Beispiele für den Entzug der ROLE-Berechtigung

Im folgenden Beispiel wird die Rolle sample_role1, die der Rolle sample_role2 zugewiesen ist, widerrufen.

```
CREATE ROLE sample_role2;
GRANT ROLE sample_role1 TO ROLE sample_role2;
REVOKE ROLE sample_role1 FROM ROLE sample_role2;
```

Im folgenden Beispiel werden Systemberechtigungen von user1 widerrufen.

```
GRANT ROLE sys:DBA TO user1;
REVOKE ROLE sys:DBA FROM user1;
```

Im folgenden Beispiel werden sample_role1 und sample_role2 von user1 widerrufen.

```
CREATE ROLE sample_role1;
```

```
CREATE ROLE sample_role2;
GRANT ROLE sample_role1, ROLE sample_role2 TO user1;
REVOKE ROLE sample_role1, ROLE sample_role2 FROM user1;
```

Im folgenden Beispiel wird sample_role2 mit der ADMIN OPTION von user1 widerrufen.

```
GRANT ROLE sample_role2 TO user1 WITH ADMIN OPTION;
REVOKE ADMIN OPTION FOR ROLE sample_role2 FROM user1;
REVOKE ROLE sample_role2 FROM user1;
```

Im folgenden Beispiel werden sample_role1 und sample_role2 von sample_role5 widerrufen.

```
CREATE ROLE sample_role5;
GRANT ROLE sample_role1, ROLE sample_role2 TO ROLE sample_role5;
REVOKE ROLE sample_role1, ROLE sample_role2 FROM ROLE sample_role5;
```

Im folgenden Beispiel werden die Systemberechtigungen CREATE SCHEMA und DROP SCHEMA für sample_role1 widerrufen.

```
GRANT CREATE SCHEMA, DROP SCHEMA TO ROLE sample_role1;
REVOKE CREATE SCHEMA, DROP SCHEMA FROM ROLE sample_role1;
```

ROLLBACK

Bricht die aktuelle Transaktion ab und verwirft alle Aktualisierungen, die durch diese Transaktion ausgeführt wurden.

Dieser Befehl hat die gleiche Funktion wie der Befehl ABORT.

Syntax

```
ROLLBACK [ WORK | TRANSACTION ]
```

Parameter

WORK

Optionales Schlüsselwort. Dieses Schlüsselwort wird in einer gespeicherten Prozedur nicht unterstützt.

TRANSACTION

Optionales Schlüsselwort. WORK und TRANSACTION sind Synonyme. Beide werden in einer gespeicherten Prozedur nicht unterstützt.

Informationen zur Verwendung von ROLLBACK innerhalb einer gespeicherten Prozedur finden Sie unter Verwalten von Transaktionen.

Beispiel

Im folgenden Beispiel werden eine Tabelle erstellt und eine Transaktion gestartet, bei der Daten in die Tabelle eingefügt werden. Der Befehl ROLLBACK nimmt die Dateneinfügung anschließend zurück, sodass die Tabelle leer ist.

Mit dem folgenden Befehl wird eine Beispieltabelle namens MOVIE_GROSS erstellt:

```
create table movie_gross( name varchar(30), gross bigint );
```

Mit dem nächsten Satz von Befehlen wird eine Transaktion gestartet, die zwei Datenzeilen in die Tabelle einfügt:

```
begin;

insert into movie_gross values ( 'Raiders of the Lost Ark', 23400000);

insert into movie_gross values ( 'Star Wars', 10000000 );
```

Als Nächstes werden mit dem folgenden Befehl die Daten aus der Tabelle ausgewählt, um zu zeigen, dass sie erfolgreich eingefügt wurden:

```
select * from movie_gross;
```

Die Befehlsausgabe zeigt, dass beide Zeilen erfolgreich eingefügt wurden:

```
name           |  gross
-------------------------+----------
Raiders of the Lost Ark | 23400000
Star Wars               | 10000000
```

```
(2 rows)
```

Mit diesem Befehl werden nun die Datenänderungen auf den Zeitpunkt zurückgesetzt, an dem die Transaktion gestartet wurde:

```
rollback;
```

Wenn nun Daten aus der Tabelle ausgewählt werden, wird eine leere Tabelle gezeigt:

```
select * from movie_gross;

name | gross
------+-------
(0 rows)
```

SELECT

Gibt Zeilen aus Tabellen, Ansichten und benutzerdefinierten Funktionen zurück.

> **Note**
>
> Die maximal zulässige Größe für eine einzelne SQL-Anweisung ist 16 MB.

Syntax

```
[ WITH with_subquery [, ...] ]
SELECT
[ TOP number | [ ALL | DISTINCT ]
* | expression [ AS output_name ] [, ...] ]
[ FROM table_reference [, ...] ]
[ WHERE condition ]
[ [ START WITH expression ] CONNECT BY expression ]
[ GROUP BY expression [, ...] ]
[ HAVING condition ]
[ QUALIFY condition ]
[ { UNION | ALL | INTERSECT | EXCEPT | MINUS } query ]
[ ORDER BY expression [ ASC | DESC ] ]
[ LIMIT { number | ALL } ]
[ OFFSET start ]
```

Themen

WITH-Klausel

Eine WITH-Klausel ist eine optionale Klausel, die der SELECT-Liste in einer Abfrage vorangeht. Die WITH-Klausel definiert einen oder mehrere allgemeine Tabellenausdrücke (CTE). Jeder allgemeine Tabellenausdruck (CTE) definiert eine temporäre Tabelle, die einer Ansichtdefinition ähnelt. Sie können diese temporären Tabellen in der FROM-Klausel referenzieren. Sie werden nur verwendet, während die Abfrage, zu der sie gehören, ausgeführt wird. Jede CTE in der WITH-Klausel gibt einen Tabellennamen, eine optionale Liste von Spaltennamen und einen Abfrageausdruck an, der in eine Tabelle evaluiert wird (eine SELECT-Anweisung). Wenn Sie den Namen der temporären Tabelle in der FROM-Klausel desselben Abfrageausdrucks referenzieren, der ihn definiert, ist der CTE rekursiv.

Unterabfragen mit einer WITH-Klausel sind eine effiziente Art, Tabellen zu definieren, die während der Ausführung einer einzelnen Abfrage verwendet werden können. In allen Fällen können dieselben Ergebnisse erzielt werden, indem im Hauptteil der SELECT-Anweisung Unterabfragen verwendet werden. Unterabfragen mit WITH-Klauseln können jedoch leichter geschrieben und gelesen werden. Wenn möglich, werden Unterabfragen mit WITH-Klauseln, die mehrmals referenziert werden, als gemeinsame Unterausdrücke optimiert. Das bedeutet, dass es möglich sein kann, eine WITH-Unterabfrage einmal zu evaluieren und die Ergebnisse wiederzuverwenden. (Beachten Sie, dass gemeinsame Unterausdrücke nicht auf diejenigen begrenzt sind, die in der WITH-Klausel definiert sind.)

Syntax

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
```

Der allgemeine Tabellenausdruck where kann entweder nicht rekursiv oder rekursiv sein. Dies ist die nicht-rekursive Form:

```
CTE_table_name [ ( column_name [, ...] ) ] AS ( query )
```

Dies ist die rekursive Form des allgemeinen Tabellenausdrucks:

```
CTE_table_name (column_name [, ...] ) AS ( recursive_query )
```

Parameter

RECURSIVE

Schlüsselwort, das die Abfrage als rekursiven CTE identifiziert. Dieses Schlüsselwort ist erforderlich, wenn ein in der WITH-Klausel definierter allgemeiner Tabellenausdruck rekursiv ist. Sie können das Schlüsselwort RECURSIVE nur einmal angeben, unmittelbar nach dem WITH-Schlüsselwort, selbst wenn die WITH-Klausel mehrere rekursive CTEs enthält. Im Allgemeinen ist ein rekursiver CTE eine UNION ALL-Unterabfrage mit zwei Teilen.

common_table_expression

Definiert eine temporäre Tabelle, auf die Sie in der FROM-Klausel verweisen können und die nur während der Ausführung der Abfrage verwendet wird, zu der sie gehört.

CTE_table_name

Ein eindeutiger Name für eine temporäre Tabelle, die die Ergebnisse einer Unterabfrage mit WITH-Klausel definiert. Sie können in einer einzelnen WITH-Klausel keine duplizierten Namen verwenden. Jede Unterabfrage muss einen Tabellennamen erhalten, der in der referenziert werden kann FROM-Klausel.

column_name

Eine Liste von Ausgabespaltennamen für die Unterabfrage der WITH-Klausel, kommagetrennt. Die Anzahl der angegebenen Spaltennamen muss größer als oder gleich der Anzahl der Spalten sein, die von der Unterabfrage definiert wird. Bei einem CTE, das nicht rekursiv ist, ist die column_name-Klausel optional. Für einen rekursiven CTE ist die column_name-Liste erforderlich.

query

Alle SELECT-Abfragen, die Amazon Redshift unterstützt. Siehe SELECT.

recursive_query

Eine UNION ALL-Abfrage, die aus zwei SELECT-Unterabfragen besteht:

- Die erste SELECT-Unterabfrage hat keinen rekursiven Verweis auf denselben CTE_table_name. Sie gibt eine Ergebnismenge zurück, die den Ausgangspunkt der Rekursion bildet. Dieses Teil wird als erstes Element oder Ausgangselement bezeichnet.
- Die zweite SELECT-Unterabfrage verweist in ihrer FROM-Klausel auf denselben CTE_table_name. Dies wird als rekursives Mitglied bezeichnet. Die recursive_query enthält eine WHERE-Bedingung, um die recursive_query zu beenden.

Nutzungshinweise

Sie können in den folgenden SQL-Anweisungen eine WITH-Klausel verwenden:

- SELECT
- SELECT INTO
- CREATE TABLE AS
- CREATE VIEW
- DECLARE
- EXPLAIN
- INSERT INTO...SELECT
- PREPARE
- UPDATE (innerhalb einer WHERE-Klausel-Unterabfrage. Sie können keinen rekursiven CTE in der Unterabfrage definieren. Der rekursive CTE muss der UPDATE-Klausel vorausgehen).
- DELETE

Wenn die FROM-Klausel einer Abfrage, die eine WITH-Klausel enthält, keine der Tabellen referenziert, die von der WITH-Klausel definiert werden, wird die WITH-Klausel ignoriert, und die Abfrage wird wie normal ausgeführt.

Eine Tabelle, die von einer Unterabfrage mit WITH-Klausel definiert ist, kann nur im Bereich der SELECT-Abfrage referenziert werden, die die WITH-Klausel beginnt. Sie können beispielsweise eine solche Tabelle in der FROM-Klausel einer Unterabfrage in der SELECT-Liste, in einer WHERE-

Klausel oder in einer HAVING-Klausel referenzieren. Sie können eine WITH-Klausel nicht in einer Unterabfrage verwenden und ihre Tabelle in der FROM-Klausel der Hauptabfrage oder einer anderen Unterabfrage referenzieren. Dieses Abfragemuster führt zu einer Fehlermeldung der Art `relation table_name doesn't exist` für die Tabelle der WITH-Klausel.

Sie können innerhalb einer Unterabfrage mit WITH-Klausel keine weitere WITH-Klausel angeben.

Sie können keine Vorausreferenzen auf Tabellen erstellen, die durch Unterabfragen mit WITH-Klauseln definiert werden. Die folgende Abfrage gibt beispielsweise aufgrund der Vorausreferenz auf die Tabelle W2 in der Definition der Tabelle W1 einen Fehler zurück:

```
with w1 as (select * from w2), w2 as (select * from w1)
select * from sales;
ERROR:  relation "w2" does not exist
```

Eine Unterabfrage mit WITH-Klausel darf nicht aus einer SELECT INTO-Anweisung bestehen. Sie können jedoch eine WITH-Klausel in einer SELECT-Anweisung verwenden.

Rekursive allgemeine Tabellenausdrücke

Eine rekursiver allgemeiner Tabellenausdruck (CTE) ist ein CTE, der sich selbst referenziert. Ein rekursiver CTE ist nützlich für die Abfrage hierarchischer Daten, wie z. B. Organigramme, die Hierarchien zwischen Mitarbeitern und Führungskräften zeigen. Siehe Beispiel: Rekursiver CTE.

Eine weitere häufige Anwendung ist eine hierarchisch aufgebaute Stückliste, wenn ein Produkt aus vielen Komponenten besteht und jede Komponente selbst auch aus anderen Komponenten oder Unterbaugruppen besteht.

Achten Sie darauf, die Rekursionstiefe zu begrenzen, indem Sie eine WHERE-Klausel in die zweite SELECT-Unterabfrage der rekursiven Abfrage aufnehmen. Ein Beispiel finden Sie unter Beispiel: Rekursiver CTE. Andernfalls kann ein Fehler wie der folgende auftreten:

- `Recursive CTE out of working buffers.`
- `Exceeded recursive CTE max rows limit, please add correct CTE termination predicates or change the max_recursion_rows parameter.`

> **Note**
>
> `max_recursion_rows` ist ein Parameter, der die maximale Anzahl von Zeilen festlegt, die ein rekursiver CTE zurückgeben kann, um endlose Rekursionsschleifen zu verhindern. Wir

raten davon ab, diesen Wert in einen höheren Wert als den Standardwert zu ändern. Dadurch wird verhindert, dass aufgrund von Problemen mit endlosen Rekursionen in Ihren Abfragen übermäßig viel Speicherplatz in Ihrem Cluster beansprucht wird.

Sie können eine Sortierreihenfolge und ein Limit für das Ergebnis des rekursiven CTE angeben. Sie können die group by- und distinct-Optionen auf das Endergebnis des rekursiven CTE anwenden.

Sie können keine WITH RECURSIVE-Klausel innerhalb einer Unterabfrage angeben. Das recursive_query-Mitglied kann keine order by- oder limit-Klausel enthalten.

Beispiele

Im folgenden Beispiel wird der einfachste mögliche Fall einer Abfrage gezeigt, die eine WITH-Klausel enthält. Die WITH-Abfrage namens VENUECOPY wählt alle Zeilen aus der Tabelle VENUE aus. Die Hauptabfrage wählt anschließend alle Zeilen aus VENUECOPY aus. Die Tabelle VENUECOPY besteht nur für die Dauer dieser Abfrage.

```
with venuecopy as (select * from venue)
select * from venuecopy order by 1 limit 10;
```

```
 venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
1 | Toyota Park                | Bridgeview      | IL         |          0
2 | Columbus Crew Stadium      | Columbus        | OH         |          0
3 | RFK Stadium                | Washington      | DC         |          0
4 | CommunityAmerica Ballpark  | Kansas City     | KS         |          0
5 | Gillette Stadium           | Foxborough      | MA         |      68756
6 | New York Giants Stadium    | East Rutherford | NJ         |      80242
7 | BMO Field                  | Toronto         | ON         |          0
8 | The Home Depot Center      | Carson          | CA         |          0
9 | Dick's Sporting Goods Park | Commerce City   | CO         |          0
v     10 | Pizza Hut Park             | Frisco          | TX         |          0
(10 rows)
```

Im folgenden Beispiel wird eine WITH-Klausel gezeigt, die zwei Tabellen namens VENUE_SALES und TOP_VENUES erstellt. Die zweite WITH-Abfragetabelle wählt aus der ersten aus. Die WHERE-Klausel des Hauptabfrageblocks enthält eine Unterabfrage, die die Tabelle TOP_VENUES einschränkt.

```
with venue_sales as
(select venuename, venuecity, sum(pricepaid) as venuename_sales
from sales, venue, event
where venue.venueid=event.venueid and event.eventid=sales.eventid
group by venuename, venuecity),

top_venues as
(select venuename
from venue_sales
where venuename_sales > 800000)

select venuename, venuecity, venuestate,
sum(qtysold) as venue_qty,
sum(pricepaid) as venue_sales
from sales, venue, event
where venue.venueid=event.venueid and event.eventid=sales.eventid
and venuename in(select venuename from top_venues)
group by venuename, venuecity, venuestate
order by venuename;
```

```
        venuename       |   venuecity   | venuestate | venue_qty | venue_sales
------------------------+---------------+------------+-----------+-------------
August Wilson Theatre   | New York City | NY         |      3187 |  1032156.00
Biltmore Theatre        | New York City | NY         |      2629 |   828981.00
Charles Playhouse       | Boston        | MA         |      2502 |   857031.00
Ethel Barrymore Theatre | New York City | NY         |      2828 |   891172.00
Eugene O'Neill Theatre  | New York City | NY         |      2488 |   828950.00
Greek Theatre           | Los Angeles   | CA         |      2445 |   838918.00
Helen Hayes Theatre     | New York City | NY         |      2948 |   978765.00
Hilton Theatre          | New York City | NY         |      2999 |   885686.00
Imperial Theatre        | New York City | NY         |      2702 |   877993.00
Lunt-Fontanne Theatre   | New York City | NY         |      3326 |  1115182.00
Majestic Theatre        | New York City | NY         |      2549 |   894275.00
Nederlander Theatre     | New York City | NY         |      2934 |   936312.00
Pasadena Playhouse      | Pasadena      | CA         |      2739 |   820435.00
Winter Garden Theatre   | New York City | NY         |      2838 |   939257.00
(14 rows)
```

In den folgenden beiden Beispielen werden die Regeln für den Bereich der Tabellenreferenzen auf der Basis von Unterabfragen mit WITH-Klausel gezeigt. Die erste Abfrage wird ausgeführt. Die zweite Abfrage schlägt jedoch mit einem erwarteten Fehler fehl. Die erste Abfrage enthält eine Unterabfrage mit WITH-Klausel innerhalb der SELECT-Liste der Hauptabfrage. Die von der WITH-

Klausel definierte Tabelle (HOLIDAYS) wird in der FROM-Klausel der Unterabfrage in der SELECT-Liste referenziert:

```
select caldate, sum(pricepaid) as daysales,
(with holidays as (select * from date where holiday ='t')
select sum(pricepaid)
from sales join holidays on sales.dateid=holidays.dateid
where caldate='2008-12-25') as dec25sales
from sales join date on sales.dateid=date.dateid
where caldate in('2008-12-25','2008-12-31')
group by caldate
order by caldate;

caldate   | daysales | dec25sales
-----------+----------+------------
2008-12-25 | 70402.00 |   70402.00
2008-12-31 | 12678.00 |   70402.00
(2 rows)
```

Die zweite Abfrage schlägt fehl, weil sie versucht, die Tabelle HOLIDAYS in der Hauptabfrage und in der Unterabfrage der SELECT-Liste zu referenzieren. Die Referenzen der Hauptabfrage liegen außerhalb des Bereichs.

```
select caldate, sum(pricepaid) as daysales,
(with holidays as (select * from date where holiday ='t')
select sum(pricepaid)
from sales join holidays on sales.dateid=holidays.dateid
where caldate='2008-12-25') as dec25sales
from sales join holidays on sales.dateid=holidays.dateid
where caldate in('2008-12-25','2008-12-31')
group by caldate
order by caldate;

ERROR:  relation "holidays" does not exist
```

Beispiel: Rekursiver CTE

Im Folgenden finden Sie ein Beispiel für einen rekursiven CTE, der die Mitarbeiter zurückgibt, die direkt oder indirekt an John berichten. Die rekursive Abfrage enthält eine WHERE-Klausel, um die Rekursionstiefe auf weniger als 4 Ebenen zu begrenzen.

```
--create and populate the sample table
```

```
  create table employee (
  id int,
  name varchar (20),
  manager_id int
  );

  insert into employee(id, name, manager_id)  values
(100, 'Carlos', null),
(101, 'John', 100),
(102, 'Jorge', 101),
(103, 'Kwaku', 101),
(110, 'Liu', 101),
(106, 'Mateo', 102),
(110, 'Nikki', 103),
(104, 'Paulo', 103),
(105, 'Richard', 103),
(120, 'Saanvi', 104),
(200, 'Shirley', 104),
(201, 'Sofía', 102),
(205, 'Zhang', 104);

--run the recursive query
  with recursive john_org(id, name, manager_id, level) as
( select id, name, manager_id, 1 as level
  from employee
  where name = 'John'
  union all
  select e.id, e.name, e.manager_id, level + 1 as next_level
  from employee e, john_org j
  where e.manager_id = j.id and level < 4
  )
 select distinct id, name, manager_id from john_org order by manager_id;
```

Nachfolgend das Ergebnis der Abfrage.

```
    id        name      manager_id
  ------+-----------+--------------
   101    John           100
   102    Jorge          101
   103    Kwaku          101
   110    Liu            101
   201    Sofía          102
   106    Mateo          102
```

```
  110    Nikki          103
  104    Paulo          103
  105    Richard        103
  120    Saanvi         104
  200    Shirley        104
  205    Zhang          104
```

Im Folgenden finden Sie ein Organigramm für Johns Abteilung.

SELECT-Liste

Themen

- Syntax
- Parameter
- Nutzungshinweise
- Beispiele

Die SELECT-Liste nennt die Spalten, Funktionen und Ausdrücke, die die Abfrage zurückgeben soll. Der Liste stellt die Ausgabe der Abfrage dar.

Weitere Hinweise zu SQL-Funktionen finden Sie unter SQL-Funktionsreferenz. Weitere Informationen zu Ausdrücken finden Sie unter Bedingte Ausdrücke.

Syntax

```
SELECT
[ TOP number ]
[ ALL | DISTINCT ] * | expression [ AS column_alias ] [, ...]
```

Parameter

TOP number

TOP verwendet als Argument eine positive Ganzzahl, die die Anzahl der Zeilen definiert, die an den Client zurückgegeben werden. Das Verhalten mit TOP-Klausel ist mit dem Verhalten mit LIMIT-Klausel identisch. Die Anzahl der zurückgegebenen Zeilen ist fest, der Zeilensatz jedoch nicht. Verwenden Sie zur Rückgabe eines konsistenten Zeilensatzes TOP oder LIMIT in Verbindung mit einer ORDER BY-Klausel.

ALL

Ein redundantes Schlüsselwort, das das Standardverhalten definiert, wenn Sie nicht DISTINCT angeben. `SELECT ALL *` bedeutet das gleiche wie `SELECT *` (Auswahl aller Zeilen für alle Spalten und Beibehaltung von Duplikaten).

DISTINCT

Eine Option, die duplizierte Zeilen aus dem Ergebnissatz entfernt, basierend auf übereinstimmenden Werten in einer oder mehreren Spalten.

> **Note**
>
> Wenn Ihre Anwendung ungültige Fremdschlüssel oder Primärschlüssel zulässt, kann dies dazu führen, dass Abfragen falsche Ergebnisse zurückgeben. Beispielsweise kann eine SELECT DISTINCT-Abfrage doppelte Zeilen zurückgeben, wenn die Primärschlüsselspalte nicht alle eindeutigen Werte enthält. Weitere Informationen finden Sie unter Definieren von Tabelleneinschränkungen.

* (Sternchen)

Gibt den gesamten Inhalt der Tabelle zurück (alle Spalten und alle Zeilen).

expression

Ein Ausdruck, der aus einer oder mehreren Spalten gebildet wird, die in den Tabellen vorhanden sind, die von der Abfrage referenziert werden. Ein Ausdruck kann SQL-Funktionen enthalten. Beispiel:

```
avg(datediff(day, listtime, saletime))
```

AS column_alias

Ein temporärer Name für die Spalte, der im endgültigen Ergebnissatz verwendet wird. Das Schlüsselwort AS ist optional. Beispiel:

```
avg(datediff(day, listtime, saletime)) as avgwait
```

Wenn Sie keinen Alias für einen Ausdruck angeben, bei dem es sich nicht um einen einfachen Spaltennamen handelt, wendet der Ergebnissatz einen Standardnamen auf diese Spalte an.

> **Note**
>
> Der Alias wird sofort nach seiner Definition in der Zielliste erkannt. Sie können einen Alias in anderen danach definierten Ausdrücken in derselben Zielliste verwenden. Das folgende Beispiel illustriert dies.
>
> ```
> select clicks / impressions as probability, round(100 * probability, 1) as
>  percentage from raw_data;
> ```
>
> Der Nutzen der lateralen Alias-Referenz ist es, dass Sie den Alias-Ausdruck nicht wiederholen müssen, wenn Sie komplexere Ausdrücke in derselben Zielliste erstellen. Wenn Amazon Redshift diesen Typ einer Referenz analysiert, stimmt es lediglich die zuvor definierten Aliase aufeinander ab. Falls eine Spalte mit demselben Namen wie der vorherige Alias-Ausdruck in der FROM-Klausel definiert wurde, erhält die Spalte in der FROM-Klausel Priorität. Beispiel: Wenn in der Abfrage oben eine Spalte mit der Bezeichnung „Wahrscheinlichkeit“ in den Rohdaten der Tabelle enthalten ist, bezieht sich die „Wahrscheinlichkeit“ im zweiten Ausdruck in der Zielliste auf diese Spalte anstatt auf den Aliasnamen „Wahrscheinlichkeit“.

Nutzungshinweise

TOP ist eine SQL-Erweiterung. Sie bietet eine Alternative zum LIMIT-Verhalten. Sie können TOP und LIMIT nicht in derselben Abfrage verwenden.

Beispiele

Im folgenden Beispiel werden 10 Zeilen aus der Tabelle SALES zurückgegeben. Trotz Verwendung der TOP-Klausel in der Abfrage wird dennoch ein unvorhersehbarer Satz von Zeilen zurückgegeben, da keine Klausel ORDER BY angegeben ist.

```
select top 10 *
from sales;
```

Die folgende Abfrage ist funktionell gleichwertig, verwendet jedoch anstelle einer TOP-Klausel eine LIMIT-Klausel:

```
select *
from sales
limit 10;
```

Im folgenden Beispiel werden unter Verwendung der TOP-Klausel die ersten 10 Zeilen aus der Tabelle SALES zurückgegeben, in absteigender Reihenfolge nach der Spalte QTYSOLD angeordnet.

```
select top 10 qtysold, sellerid
from sales
order by qtysold desc, sellerid;

qtysold | sellerid
--------+----------
8 |      518
8 |      520
8 |      574
8 |      718
8 |      868
8 |     2663
8 |     3396
8 |     3726
8 |     5250
8 |     6216
(10 rows)
```

Im folgenden Beispiel werden die ersten beiden QTYSOLD- und SELLERID-Werte aus der Tabelle SALES zurückgegeben, nach der Spalte QTYSOLD angeordnet:

```
select top 2 qtysold, sellerid
from sales
order by qtysold desc, sellerid;

qtysold | sellerid
--------+----------
8 |      518
8 |      520
(2 rows)
```

Im folgenden Beispiel ist die Liste der verschiedenen Kategoriegruppen aus der Tabelle CATEGORY zu sehen:

```
select distinct catgroup from category
order by 1;

catgroup
----------
Concerts
Shows
Sports
(3 rows)

--the same query, run without distinct
select catgroup from category
order by 1;

catgroup
----------
Concerts
Concerts
Concerts
Shows
Shows
Shows
Sports
Sports
Sports
Sports
Sports
(11 rows)
```

Im folgenden Beispiel wird der eindeutige Satz von Wochenzahlen für Dezember 2008 zurückgegeben. Ohne die DISTINCT-Klausel würde die Anweisung 31 Zeilen bzw. eine Zeile für jeden Tag des Monats zurückgeben.

```
select distinct week, month, year
from date
where month='DEC' and year=2008
order by 1, 2, 3;

week | month | year
-----+-------+------
49 | DEC   | 2008
50 | DEC   | 2008
```

```
51 | DEC   | 2008
52 | DEC   | 2008
53 | DEC   | 2008
(5 rows)
```

FROM-Klausel

Die -Klausel in einer Abfrage listet die Tabellenreferenzen (Tabellen, Ansichten und Unterabfragen) auf, aus denen Daten ausgewählt werden. Wenn mehrere Tabellenreferenzen aufgelistet werden, muss ein Join für die Tabellen ausgeführt werden, indem entweder in der FROM-Klausel oder in der WHERE-Klausel die entsprechende Syntax verwendet wird. Wenn keine Join-Kriterien angegeben werden, verarbeitet das System die Abfrage als Kreuz-Join (kartesisches Produkt).

Themen

Syntax

```
FROM table_reference [, ...]
```

wobei table_reference eins der folgenden ist:

```
with_subquery_table_name [ table_alias ]
table_name [ * ] [ table_alias ]
( subquery ) [ table_alias ]
table_reference [ NATURAL ] join_type table_reference
   [ ON join_condition | USING ( join_column [, ...] ) ]
table_reference PIVOT (
   aggregate(expr) [ [ AS ] aggregate_alias ]
   FOR column_name IN ( expression [ AS ] in_alias [, ...] )
) [ table_alias ]
table_reference UNPIVOT [ INCLUDE NULLS | EXCLUDE NULLS ] (
   value_column_name
```

```
   FOR name_column_name IN ( column_reference [ [ AS ]
   in_alias ] [, ...] )
) [ table_alias ]
UNPIVOT expression AS value_alias [ AT attribute_alias ]
```

Der optionale table_alias kann verwendet werden, um Tabellen und komplexen Tabellenverweisen und, falls gewünscht, auch ihren Spalten temporäre Namen zuzuweisen, wie im Folgenden dargestellt:

```
[ AS ] alias [ ( column_alias [, ...] ) ]
```

Parameter

with_subquery_table_name

Eine Tabelle, die von einer Unterabfrage in der definiert wird WITH-Klausel.

table_name

Der Name einer Tabelle oder Ansicht.

alias

Der temporäre alternative Name für eine Tabelle oder Ansicht. Für eine Tabelle, die von einer Unterabfrage abgeleitet wird, muss ein Alias bereitgestellt werden. In anderen Tabellenreferenzen sind Aliasnamen optional. Das Schlüsselwort AS ist stets optional. Tabellenaliasnamen stellen eine bequeme Abkürzung für die Identifizierung von Tabellen in anderen Teilen einer Abfrage dar, beispielsweise in der WHERE-Klausel. Beispiel:

```
select * from sales s, listing l
where s.listid=l.listid
```

column_alias

Der temporäre alternative Name für eine Spalte in einer Tabelle oder Ansicht.

subquery

Ein Abfrageausdruck, der zu einer Tabelle evaluiert wird. Die Tabelle ist nur für die Dauer der Abfrage vorhanden und erhält in der Regel einen Namen oder einen Alias. Ein Alias ist jedoch nicht erforderlich. Sie können auch Spaltennamen für Tabellen definieren, die von Unterabfragen abgeleitet werden. Die Vergabe von Spaltenaliasnamen ist wichtig, wenn Sie für die Ergebnisse

von Unterabfragen einen Join mit anderen Tabellen ausführen möchten und wenn Sie diese Spalten an anderer Stelle in der Abfrage auswählen oder einschränken möchten.

Eine Unterabfrage kann eine ORDER BY-Klausel enthalten. Diese Klausel hat jedoch keine Auswirkungen, wenn nicht auch eine LIMIT- oder OFFSET-Klausel angegeben ist.

NATURAL

Definiert einen Join, der automatisch alle Paare identisch benannter Spalten in den beiden Tabellen als Joining-Spalten verwendet. Es ist keine explizite Join-Bedingung erforderlich. Wenn die Tabellen CATEGORY und EVENT beispielsweise beide Spalten namens CATID besitzen, ist ein Join ihrer CATID-Spalten ein NATURAL-Join dieser Tabellen.

> **Note**
>
> Wenn ein NATURAL-Join angegeben ist, in den Tabellen, für die ein Join ausgeführt werden soll, jedoch keine identisch benannten Spaltenpaare vorhanden sind, wird für die Abfrage standardmäßig ein Kreuz-Join ausgeführt.

join_type

Geben Sie eine der folgenden Join-Arten an:

- [INNER] JOIN
- LEFT [OUTER] JOIN
- RIGHT [OUTER] JOIN
- FULL [OUTER] JOIN
- CROSS JOIN

Kreuz-Joins sind nicht qualifizierte Joins. Sie geben das kartesische Produkt der beiden Tabellen zurück.

Interne und externe Joins sind qualifizierte Joins. Sie sind entweder implizit (in natürlichen Joins), mit der ON- oder USING-Syntax in der FROM-Klausel oder mit einer WHERE-Klauselbedingung qualifiziert.

Ein interner Join gibt nur übereinstimmende Zeilen zurück, basierend auf der Join-Bedingung oder der Liste der Joining-Spalten. Ein externer Join gibt alle Zeilen zurück, die der entsprechende

interne Join zurückgeben würde, und zusätzlich nicht übereinstimmende Zeilen aus der Tabelle „links“, aus der Tabelle „rechts“ oder aus beiden Tabellen. Die linke Tabelle wird zuerst aufgelistet. Die rechte Tabelle wird als zweite Tabelle aufgelistet. Die nicht übereinstimmenden Zeilen enthalten NULL-Werte, um die Lücken in den Ausgabespalten zu füllen.

ON join_condition

Eine Join-Spezifikation, in der die Joining-Spalten als eine Bedingung angegeben werden, die dem Schlüsselwort ON folgt. Beispiel:

```
sales join listing
on sales.listid=listing.listid and sales.eventid=listing.eventid
```

USING (join_column [, ...])

Eine Join-Spezifikation, in der die Joining-Spalten in Klammern angegeben werden. Wenn mehrere Joining-Spalten angegeben werden, werden sie durch Komma abgetrennt. Das Schlüsselwort USING muss der Liste vorangestellt werden. Beispielsweise:

```
sales join listing
using (listid,eventid)
```

PIVOT

Dreht die Ausgabe von Zeilen zu Spalten, um tabellarische Daten in einem einfach lesbaren Format darzustellen. Die Ausgabe wird horizontal über mehrere Spalten hinweg dargestellt. PIVOT ähnelt einer GROUP BY-Abfrage mit einer Aggregation, wobei ein Aggregatausdruck verwendet wird, um ein Ausgabeformat anzugeben. Im Gegensatz zu GROUP BY werden die Ergebnisse jedoch in Spalten anstelle von Zeilen zurückgegeben.

Beispiele, die Abfragen mit PIVOT und UNPIVOT veranschaulichen, finden Sie unter Beispiele für PIVOT und UNPIVOT.

UNPIVOT

Spalten mit UNPIVOT in Zeilen drehen — Der Operator wandelt Ergebnisspalten aus einer Eingabetabelle oder von Abfrageergebnissen in Zeilen um, um die Ausgabe lesbarer zu machen. UNPIVOT fasst die Daten der Eingabespalten in zwei Ergebnisspalten zusammen: eine Namensspalte und eine Wertspalte. Die Namensspalte enthält die Spaltennamen aus der Eingabe als Zeileneinträge. Die Wertspalte enthält Werte aus den Eingabespalten, wie etwa

die Ergebnisse einer Aggregation. Zum Beispiel die Anzahl der Elemente in verschiedenen Kategorien.

Entpivotieren von Objekten mit UNPIVOT (SUPER) — Sie können eine Objektpivotierung durchführen, wobei Ausdruck ein SUPER-Ausdruck ist, der auf ein anderes FROM-Klauselelement verweist. Weitere Informationen finden Sie unter Entpivotieren von Objekten. Es enthält auch Beispiele, die zeigen, wie semistrukturierte Daten, z. B. Daten im JSON-Format, abgefragt werden.

Nutzungshinweise

Joining-Spalten müssen vergleichbare Datentypen haben.

Ein NATURAL- oder -USING-Join enthält jeweils nur eine Spalte jedes Joining-Spaltenpaars im Zwischenergebnissatz.

Ein Join mit der ON-Syntax enthält beide Joining-Spalten im Zwischenergebnissatz.

Weitere Informationen finden Sie auch unter WITH-Klausel.

Beispiele für PIVOT und UNPIVOT

PIVOT und UNPIVOT sind Parameter in der FROM-Klausel, die die Abfrageausgabe von Zeilen in Spalten bzw. von Spalten in Zeilen drehen. Sie stellen tabellarische Abfrageergebnisse in einem leicht lesbaren Format dar. In den folgenden Beispielen wird anhand von Testdaten und Abfragen ihre Verwendung dargestellt.

Weitere Informationen zu diesen und anderen Parametern finden Sie unter FROM-Klausel.

Beispiele für PIVOT

Richten Sie die Beispieltabelle und -daten ein und verwenden Sie sie, um die folgenden Beispielabfragen auszuführen.

```
CREATE TABLE part (
    partname varchar,
    manufacturer varchar,
    quality int,
    price decimal(12, 2)
);

INSERT INTO part VALUES ('prop', 'local parts co', 2, 10.00);
```

```
INSERT INTO part VALUES ('prop', 'big parts co', NULL, 9.00);
INSERT INTO part VALUES ('prop', 'small parts co', 1, 12.00);

INSERT INTO part VALUES ('rudder', 'local parts co', 1, 2.50);
INSERT INTO part VALUES ('rudder', 'big parts co', 2, 3.75);
INSERT INTO part VALUES ('rudder', 'small parts co', NULL, 1.90);

INSERT INTO part VALUES ('wing', 'local parts co', NULL, 7.50);
INSERT INTO part VALUES ('wing', 'big parts co', 1, 15.20);
INSERT INTO part VALUES ('wing', 'small parts co', NULL, 11.80);
```

PIVOT auf `partname` mit einer AVG-Aggregation auf `price`.

```
SELECT *
FROM (SELECT partname, price FROM part) PIVOT (
    AVG(price) FOR partname IN ('prop', 'rudder', 'wing')
);
```

Die Abfrage führt zur folgenden Ausgabe.

```
  prop   |  rudder  |  wing
---------+----------+---------
 10.33   | 2.71     | 11.50
```

Im vorherigen Beispiel werden die Ergebnisse in Spalten umgewandelt. Das folgende Beispiel zeigt eine GROUP BY-Abfrage, die die Durchschnittspreise in Zeilen und nicht in Spalten zurückgibt.

```
SELECT partname, avg(price)
FROM (SELECT partname, price FROM part)
WHERE partname IN ('prop', 'rudder', 'wing')
GROUP BY partname;
```

Die Abfrage führt zur folgenden Ausgabe.

```
 partname |  avg
----------+-------
 prop     | 10.33
 rudder   |  2.71
 wing     | 11.50
```

Ein PIVOT-Beispiel mit `manufacturer` als impliziter Spalte.

```
SELECT *
FROM (SELECT quality, manufacturer FROM part) PIVOT (
    count(*) FOR quality IN (1, 2, NULL)
);
```

Die Abfrage führt zur folgenden Ausgabe.

```
 manufacturer      | 1  | 2  | null
-------------------+----+----+------
 local parts co    | 1  | 1  |  1
 big parts co      | 1  | 1  |  1
 small parts co    | 1  | 0  |  2
```

Eingabetabellenspalten, die nicht in der PIVOT-Definition referenziert sind, werden implizit zur Ergebnistabelle hinzugefügt. Dies ist zum Beispiel bei der manufacturer-Spalte im vorigen Beispiel der Fall. Das Beispiel zeigt auch, dass NULL ein gültiger Wert für den IN-Operator ist.

PIVOT im obigen Beispiel gibt ähnliche Informationen zurück wie die folgende Abfrage, die GROUP BY umfasst. Der Unterschied besteht darin, dass PIVOT den Wert 0 für Spalte 2 und den Hersteller small parts co zurückgibt. Die GROUP BY-Abfrage enthält keine entsprechende Zeile. In dem meisten Fällen fügt PIVOT NULL ein, wenn eine Zeile keine Eingabedaten für eine bestimmte Spalte aufweist. Das Zählaggregat gibt jedoch nicht NULL zurück und 0 ist der Standardwert.

```
SELECT manufacturer, quality, count(*)
FROM (SELECT quality, manufacturer FROM part)
WHERE quality IN (1, 2) OR quality IS NULL
GROUP BY manufacturer, quality
ORDER BY manufacturer;
```

Die Abfrage führt zur folgenden Ausgabe.

```
 manufacturer        | quality | count
---------------------+---------+-------
 big parts co        |         |     1
 big parts co        |       2 |     1
 big parts co        |       1 |     1
 local parts co      |       2 |     1
 local parts co      |       1 |     1
 local parts co      |         |     1
 small parts co      |       1 |     1
```

```
 small parts co      |         |     2
```

Der PIVOT-Operator akzeptiert optionale Aliase auf dem Aggregatausdruck und jedem Wert für den IN-Operator. Verwenden Sie Aliase, um die Spaltennamen anzupassen. Wenn kein Aggregatalias vorliegt, werden nur IN-Listenaliase verwendet. Andernfalls wird das Aggregatalias an den Spaltennamen angefügt (mit einem Unterstrich, um die Namen auseinanderzuhalten).

```
SELECT *
FROM (SELECT quality, manufacturer FROM part) PIVOT (
    count(*) AS count FOR quality IN (1 AS high, 2 AS low, NULL AS na)
);
```

Die Abfrage führt zur folgenden Ausgabe.

```
 manufacturer      | high_count  | low_count | na_count
-------------------+-------------+-----------+----------
 local parts co    |           1 |         1 |        1
 big parts co      |           1 |         1 |        1
 small parts co    |           1 |         0 |        2
```

Richten Sie die folgende Beispieltabelle und -daten ein und verwenden Sie sie, um die folgenden Beispielabfragen auszuführen. Die Daten stellen Buchungstermine für eine Reihe von Hotels dar.

```
CREATE TABLE bookings (
    booking_id int,
    hotel_code char(8),
    booking_date date,
    price decimal(12, 2)
);

INSERT INTO bookings VALUES (1, 'FOREST_L', '02/01/2023', 75.12);
INSERT INTO bookings VALUES (2, 'FOREST_L', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (3, 'FOREST_L', '02/04/2023', 85.54);

INSERT INTO bookings VALUES (4, 'FOREST_L', '02/08/2023', 75.00);
INSERT INTO bookings VALUES (5, 'FOREST_L', '02/11/2023', 75.00);
INSERT INTO bookings VALUES (6, 'FOREST_L', '02/14/2023', 90.00);

INSERT INTO bookings VALUES (7, 'FOREST_L', '02/21/2023', 60.00);
INSERT INTO bookings VALUES (8, 'FOREST_L', '02/22/2023', 85.00);
INSERT INTO bookings VALUES (9, 'FOREST_L', '02/27/2023', 90.00);
```

```
INSERT INTO bookings VALUES (10, 'DESERT_S', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (11, 'DESERT_S', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (12, 'DESERT_S', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (13, 'DESERT_S', '02/05/2023', 75.00);
INSERT INTO bookings VALUES (14, 'DESERT_S', '02/06/2023', 34.00);
INSERT INTO bookings VALUES (15, 'DESERT_S', '02/09/2023', 85.00);

INSERT INTO bookings VALUES (16, 'DESERT_S', '02/12/2023', 23.00);
INSERT INTO bookings VALUES (17, 'DESERT_S', '02/13/2023', 76.00);
INSERT INTO bookings VALUES (18, 'DESERT_S', '02/14/2023', 85.00);

INSERT INTO bookings VALUES (19, 'OCEAN_WV', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (20, 'OCEAN_WV', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (21, 'OCEAN_WV', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (22, 'OCEAN_WV', '02/06/2023', 75.00);
INSERT INTO bookings VALUES (23, 'OCEAN_WV', '02/09/2023', 34.00);
INSERT INTO bookings VALUES (24, 'OCEAN_WV', '02/12/2023', 85.00);

INSERT INTO bookings VALUES (25, 'OCEAN_WV', '02/13/2023', 23.00);
INSERT INTO bookings VALUES (26, 'OCEAN_WV', '02/14/2023', 76.00);
INSERT INTO bookings VALUES (27, 'OCEAN_WV', '02/16/2023', 85.00);

INSERT INTO bookings VALUES (28, 'CITY_BLD', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (29, 'CITY_BLD', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (30, 'CITY_BLD', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (31, 'CITY_BLD', '02/12/2023', 75.00);
INSERT INTO bookings VALUES (32, 'CITY_BLD', '02/13/2023', 34.00);
INSERT INTO bookings VALUES (33, 'CITY_BLD', '02/17/2023', 85.00);

INSERT INTO bookings VALUES (34, 'CITY_BLD', '02/22/2023', 23.00);
INSERT INTO bookings VALUES (35, 'CITY_BLD', '02/23/2023', 76.00);
INSERT INTO bookings VALUES (36, 'CITY_BLD', '02/24/2023', 85.00);
```

In dieser Beispielabfrage werden die Buchungsdatensätze zusammengezählt, um eine Gesamtsumme für jede Woche zu erhalten. Das Enddatum für jede Woche wird zu einem Spaltennamen.

```
SELECT * FROM
    (SELECT
```

```
       booking_id,
       (date_trunc('week', booking_date::date) + '5 days'::interval)::date as enddate,
       hotel_code AS "hotel code"
FROM bookings
) PIVOT (
    count(booking_id) FOR enddate IN ('2023-02-04','2023-02-11','2023-02-18')
);
```

Die Abfrage führt zur folgenden Ausgabe.

```
 hotel code | 2023-02-04  | 2023-02-11 | 2023-02-18
------------+-------------+------------+----------
 FOREST_L   |           3 |          2 |        1
 DESERT_S   |           4 |          3 |        2
 OCEAN_WV   |           3 |          3 |        3
 CITY_BLD   |           3 |          1 |        2
```

Amazon Redshift unterstützt CROSSTAB, um mehrere Spalten zu pivotieren. Mit einer Abfrage wie der folgenden können Sie jedoch Zeilendaten in Spalten ändern, ähnlich wie bei einer Aggregation mit PIVOT. Dabei werden dieselben Buchungsbeispieldaten wie im vorherigen Beispiel verwendet.

```
SELECT
  booking_date,
  MAX(CASE WHEN hotel_code = 'FOREST_L' THEN 'forest is booked' ELSE '' END) AS
 FOREST_L,
  MAX(CASE WHEN hotel_code = 'DESERT_S' THEN 'desert is booked' ELSE '' END) AS
 DESERT_S,
  MAX(CASE WHEN hotel_code = 'OCEAN_WV' THEN 'ocean is booked' ELSE '' END)  AS
 OCEAN_WV
FROM bookings
GROUP BY booking_date
ORDER BY booking_date asc;
```

Die Beispielabfrage ergibt Buchungsdaten, die neben kurzen Ausdrücken aufgeführt sind und angeben, welche Hotels gebucht wurden.

```
 booking_date  | forest_l         | desert_s         | ocean_wv
---------------+------------------+------------------+--------------------
 2023-02-01    | forest is booked | desert is booked |  ocean is booked
 2023-02-02    | forest is booked | desert is booked |  ocean is booked
 2023-02-04    | forest is booked | desert is booked |  ocean is booked
 2023-02-05    |                  | desert is booked |
```

```
 2023-02-06    |                 | desert is booked |
```

Im Folgenden finden Sie Nutzungshinweise für PIVOT:

- PIVOT kann auf Tabellen, Unterabfragen und allgemeine Tabellenausdrücke (CTEs) angewendet werden. PIVOT kann nicht auf JOIN-Ausdrücke, rekursive CTEs, PIVOT- oder UNPIVOT-Ausdrücke angewendet werden. Ebenfalls nicht unterstützt werden nicht verschachtelte SUPER-Ausdrücke sowie verschachtelte Redshift-Spectrum-Tabellen.
- PIVOT unterstützt die Aggregatfunktionen COUNT, SUM, MIN, MAX und AVG.
- Der PIVOT-Aggregatausdruck muss ein Aufruf einer unterstützten Aggregatfunktion sein. Komplexe Ausdrücke, die auf dem Aggregat aufbauen, werden nicht unterstützt. Die Aggregatargumente können keine Verweise auf Tabellen enthalten, bei denen es sich nicht um die PIVOT-Eingabetabelle handelt. Korrelierte Verweise auf eine übergeordnete Abfrage werden ebenfalls nicht unterstützt. Das Aggregatargument kann Unterabfragen enthalten. Diese können intern oder in der PIVOT-Eingabetabelle korreliert werden.
- Die PIVOT IN-Listenwerte können keine Spaltenverweise oder Unterabfragen sein. Alle Werte müssen mit dem FOR-Spaltenverweis typenkompatibel sein.
- Wenn die IN-Listenwerte keine Aliase haben, generiert PIVOT Standardspaltennamen. Bei konstanten IN-Werten wie etwa „abc“ oder „5“ ist der Spaltennamen die Konstante. Bei komplexen Ausdrücken ist der Spaltenname ein Amazon-Redshift-Standardname wie etwa ?column?.

Beispiele für UNPIVOT

Richten Sie die Beispieldaten ein und verwenden Sie sie, um die folgenden Beispiele auszuführen.

```
CREATE TABLE count_by_color (quality varchar, red int, green int, blue int);

INSERT INTO count_by_color VALUES ('high', 15, 20, 7);
INSERT INTO count_by_color VALUES ('normal', 35, NULL, 40);
INSERT INTO count_by_color VALUES ('low', 10, 23, NULL);
```

UNPIVOT in den Eingabespalten rot, grün und blau.

```
SELECT *
FROM (SELECT red, green, blue FROM count_by_color) UNPIVOT (
    cnt FOR color IN (red, green, blue)
);
```

Die Abfrage führt zur folgenden Ausgabe.

```
 color | cnt
-------+-----
 red   |  15
 red   |  35
 red   |  10
 green |  20
 green |  23
 blue  |   7
 blue  |  40
```

Standardmäßig werden NULL-Werte in der Eingabespalte übersprungen und ergeben keine Ergebniszeile.

Das folgende Beispiel zeigt UNPIVOT mit INCLUDE NULLS.

```
SELECT *
FROM (
    SELECT red, green, blue
    FROM count_by_color
) UNPIVOT INCLUDE NULLS (
    cnt FOR color IN (red, green, blue)
);
```

Die resultierende Ausgabe sieht wie folgt aus.

```
 color | cnt
-------+-----
 red   |  15
 red   |  35
 red   |  10
 green |  20
 green |
 green |  23
 blue  |   7
 blue  |  40
 blue  |
```

Wenn der Parameter INCLUDING NULLS festgelegt wurde, generieren NULL-Eingabewerte Ergebniszeilen.

The following query shows UNPIVOT mit quality als impliziter Spalte.

```
SELECT *
FROM count_by_color UNPIVOT (
    cnt FOR color IN (red, green, blue)
);
```

Die Abfrage führt zur folgenden Ausgabe.

```
 quality | color | cnt
---------+-------+-----
 high    | red   |  15
 normal  | red   |  35
 low     | red   |  10
 high    | green |  20
 low     | green |  23
 high    | blue  |   7
 normal  | blue  |  40
```

Spalten der Eingabetabelle, die nicht in der UNPIVOT-Definition referenziert sind, werden implizit zur Ergebnistabelle hinzugefügt. Im Beispiel ist dies bei der quality-Spalte der Fall.

Das folgende Beispiel zeigt UNPIVOT mit Aliasen für Werte in der IN-Liste.

```
SELECT *
FROM count_by_color UNPIVOT (
    cnt FOR color IN (red AS r, green AS g, blue AS b)
);
```

Die vorige Abfrage führt zu der folgenden Ausgabe.

```
 quality | color | cnt
---------+-------+-----
 high    | r     |  15
 normal  | r     |  35
 low     | r     |  10
 high    | g     |  20
 low     | g     |  23
 high    | b     |   7
 normal  | b     |  40
```

Der UNPIVOT-Operator akzeptiert optionale Aliase auf jedem IN-Listenwert. Jeder Alias ermöglicht die Anpassung der Daten in jeder value-Spalte.

Im Folgenden finden Sie Nutzungshinweise für UNPIVOT.

- UNPIVOT kann auf Tabellen, Unterabfragen und allgemeine Tabellenausdrücke (CTEs) angewendet werden. UNPIVOT kann nicht auf JOIN-Ausdrücke, rekursive CTEs, PIVOT- oder UNPIVOT-Ausdrücke angewendet werden. Ebenfalls nicht unterstützt werden nicht verschachtelte SUPER-Ausdrücke sowie verschachtelte Redshift-Spectrum-Tabellen.
- Die Liste UNPIVOT IN darf nur Spaltenverweise auf Eingabetabellen enthalten. Die IN-Listenspalten müssen einen gemeinsamen Typ haben, mit dem sie alle kompatibel sind. Die UNPIVOT-Wertspalte hat diesen gemeinsamen Typ. Der UNPIVOT-Spaltenname hat den Typ VARCHAR.
- Wenn ein IN-Listenwert keinen Alias hat, verwendet UNPIVOT den Spaltennamen als Standardwert.

JOIN-Beispiele

Eine SQL JOIN-Klausel wird verwendet, um die Daten aus zwei oder mehr Tabellen basierend auf gemeinsamen Feldern zu kombinieren. Die Ergebnisse können sich je nach festgelegter Join-Methode ändern oder nicht. Weitere Informationen zur Syntax einer JOIN-Klausel finden Sie unter Parameter.

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen über das Datenbankschema finden Sie unter Beispieldatenbank. Informationen zum Laden von Beispieldaten finden Sie unter Laden von Daten im Amazon Redshift Getting Started Guide.

Die folgende Abfrage ist ein innerer Join (ohne das Schlüsselwort JOIN) zwischen den Tabellen LISTING und SALES, wobei die LISTID aus der Tabelle LISTING zwischen 1 und 5 liegt. Diese Abfrage gleicht LISTID-Spaltenwerte in der Tabelle LISTING (linke Tabelle) und der Tabelle SALES (rechte Tabelle) ab. Die Ergebnisse zeigen, dass LISTID 1, 4 und 5 den Kriterien entsprechen.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing, sales
where listing.listid = sales.listid
and listing.listid between 1 and 5
group by 1
order by 1;
```

```
listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

Bei der folgenden Abfrage handelt es sich um einen linken, externen Join. Externe Joins nach links und rechts behalten die Werte aus einer der Tabellen, für die ein Join ausgeführt wurde, wenn in der anderen Tabelle keine Übereinstimmung gefunden wurde. Die Tabellen links und rechts werden in der Syntax als erste und zweite Tabelle aufgelistet. Es werden NULL-Werte verwendet, um die „Lücken“ im Ergebnissatz zu füllen. Diese Abfrage gleicht LISTID-Spaltenwerte in der Tabelle LISTING (linke Tabelle) und der Tabelle SALES (rechte Tabelle) ab. Die Ergebnisse zeigen, dass die LISTIDs 2 und 3 nicht zu Verkäufen führten.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing left outer join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     2 | NULL   | NULL
     3 | NULL   | NULL
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

Bei der folgenden Abfrage handelt es sich um einen rechten, externen Join. Diese Abfrage gleicht LISTID-Spaltenwerte in der Tabelle LISTING (linke Tabelle) und der Tabelle SALES (rechte Tabelle) ab. Die Ergebnisse zeigen, dass die LISTIDs 1, 4 und 5 den Kriterien entsprechen.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing right outer join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
```

```
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

Bei der folgenden Abfrage handelt es sich um einen vollständigen Join. Vollständige Joins behalten die Werte aus einer der Tabellen bei, für die ein Join ausgeführt wurde, wenn in der anderen Tabelle keine Übereinstimmung gefunden wurde. Die Tabellen links und rechts werden in der Syntax als erste und zweite Tabelle aufgelistet. Es werden NULL-Werte verwendet, um die „Lücken“ im Ergebnissatz zu füllen. Diese Abfrage gleicht LISTID-Spaltenwerte in der Tabelle LISTING (linke Tabelle) und der Tabelle SALES (rechte Tabelle) ab. Die Ergebnisse zeigen, dass die LISTIDs 2 und 3 nicht zu Verkäufen führten.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing full join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     2 | NULL   | NULL
     3 | NULL   | NULL
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

Bei der folgenden Abfrage handelt es sich um einen vollständigen Join. Diese Abfrage gleicht LISTID-Spaltenwerte in der Tabelle LISTING (linke Tabelle) und der Tabelle SALES (rechte Tabelle) ab. In den Ergebnissen sind nur Zeilen enthalten, die zu keinen Verkäufen führen (LISTIDs 2 und 3).

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing full join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
and (listing.listid IS NULL or sales.listid IS NULL)
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     2 | NULL   | NULL
     3 | NULL   | NULL
```

Bei dem folgenden Beispiel handelt es sich um einen inneren Join mit der ON-Klausel. In diesem Fall werden NULL-Zeilen nicht zurückgegeben.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from sales join listing
on sales.listid=listing.listid and sales.eventid=listing.eventid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

Bei der folgenden Abfrage handelt es sich um einen Cross Join oder kartesischen Join der LISTING- und der SALES-Tabelle mit einem Prädikat zur Begrenzung der Ergebnisse. Diese Abfrage gleicht LISTID-Spaltenwerte in der SALES- und der LISTING-Tabelle für LISTIDs 1, 2, 3, 4 und 5 in beiden Tabellen ab. Die Ergebnisse zeigen, dass 20 Zeilen den Kriterien entsprechen.

```
select sales.listid as sales_listid, listing.listid as listing_listid
from sales cross join listing
where sales.listid between 1 and 5
and listing.listid between 1 and 5
order by 1,2;

sales_listid | listing_listid
-------------+---------------
1            | 1
1            | 2
1            | 3
1            | 4
1            | 5
4            | 1
4            | 2
4            | 3
4            | 4
4            | 5
5            | 1
5            | 1
5            | 2
```

```
5            | 2
5            | 3
5            | 3
5            | 4
5            | 4
5            | 5
5            | 5
```

Das folgende Beispiel ist ein NATURAL-Join zwischen zwei Tabellen. In diesem Fall haben die Spalten listid, sellerid, eventid und dateid identische Namen und Datentypen in beiden Tabellen und werden daher als Join-Spalten verwendet. Die Ergebnisse sind auf 5 Zeilen begrenzt.

```
select listid, sellerid, eventid, dateid, numtickets
from listing natural join sales
order by 1
limit 5;

listid | sellerid  | eventid | dateid | numtickets
-------+-----------+---------+--------+-----------
113    | 29704     | 4699    | 2075   | 22
115    | 39115     | 3513    | 2062   | 14
116    | 43314     | 8675    | 1910   | 28
118    | 6079      | 1611    | 1862   | 9
163    | 24880     | 8253    | 1888   | 14
```

Das folgende Beispiel ist ein Join zwischen zwei Tabellen mit der USING-Klausel. In diesem Fall werden die Spalten listid und eventid als Join-Spalten verwendet. Die Ergebnisse sind auf 5 Zeilen begrenzt.

```
select listid, listing.sellerid, eventid, listing.dateid, numtickets
from listing join sales
using (listid, eventid)
order by 1
limit 5;

listid | sellerid | eventid | dateid | numtickets
-------+----------+---------+--------+-----------
1      | 36861    | 7872    | 1850   | 10
4      | 8117     | 4337    | 1970   | 8
5      | 1616     | 8647    | 1963   | 4
5      | 1616     | 8647    | 1963   | 4
6      | 47402    | 8240    | 2053   | 18
```

Die folgende Abfrage ist ein interner Join zweiter Unterabfragen in der FROM-Klausel. Die Abfrage ermittelt die Zahl der verkauften und nicht verkauften Tickets für verschiedene Veranstaltungskategorien (Konzerte und Shows). Die Unterabfragen mit FROM-Klausel sind Tabellen-Unterabfragen und können mehrere Spalten und Zeilen zurückgeben.

```
select catgroup1, sold, unsold
from
(select catgroup, sum(qtysold) as sold
from category c, event e, sales s
where c.catid = e.catid and e.eventid = s.eventid
group by catgroup) as a(catgroup1, sold)
join
(select catgroup, sum(numtickets)-sum(qtysold) as unsold
from category c, event e, sales s, listing l
where c.catid = e.catid and e.eventid = s.eventid
and s.listid = l.listid
group by catgroup) as b(catgroup2, unsold)

on a.catgroup1 = b.catgroup2
order by 1;

catgroup1 |  sold  | unsold
----------+--------+--------
Concerts  | 195444 |1067199
Shows     | 149905 | 817736
```

WHERE-Klausel

Die WHERE-Klausel enthält Bedingungen, die entweder einen Join für Tabellen ausführen oder Prädikate auf Spalten in Tabellen anwenden. Für Tabellen können interne Joins ausgeführt werden, indem entweder in der WHERE-Klausel oder in der FROM-Klausel die entsprechende Syntax verwendet wird. Die Kriterien für externe Joins müssen in der FROM-Klausel angegeben werden.

Syntax

```
[ WHERE condition ]
```

Bedingung

Jede Suchbedingung mit einem Booleschen Ergebnis, wie eine Join-Bedingung oder ein Prädikat für eine Tabellenspalte. In den folgenden Beispielen werden gültige Join-Bedingungen gezeigt:

```
sales.listid=listing.listid
sales.listid<>listing.listid
```

In den folgenden Beispielen werden gültige Bedingungen für Spalten in Tabellen gezeigt:

```
catgroup like 'S%'
venueseats between 20000 and 50000
eventname in('Jersey Boys','Spamalot')
year=2008
length(catdesc)>25
date_part(month, caldate)=6
```

Bedingungen können einfach oder komplex sein. Im Fall komplexer Bedingungen können Sie Klammern verwenden, um logische Einheiten zu isolieren. Im folgenden Beispiel wird die Join-Bedingung durch Klammern umschlossen.

```
where (category.catid=event.catid) and category.catid in(6,7,8)
```

Nutzungshinweise

Sie können in der WHERE-Klausel Aliase verwenden, um Auswahllistenausdrücke zu referenzieren.

Sie können die Ergebnisse aggregierter Funktionen in der WHERE-Klausel nicht einschränken. Verwenden Sie für diesen Zweck die HAVING-Klausel.

Spalten, die in der WHERE-Klausel eingeschränkt sind, müssen von Tabellenreferenzen in der FROM-Klausel abgeleitet werden.

Beispiel

Die folgende Abfrage verwendet eine Kombination aus verschiedenen WHERE-Klauseleinschränkungen, einschließlich einer Join-Bedingung für die Tabellen SALES und EVENT, eines Prädikats für die EVENTNAME-Spalte und zweier Prädikate für die STARTTIME-Spalte.

```
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Hannah Montana'
and date_part(quarter, starttime) in(1,2)
```

```
and date_part(year, starttime) = 2008
order by 3 desc, 4, 2, 1 limit 10;

eventname    |      starttime      |   costperticket   | qtysold
---------------+---------------------+-------------------+---------
Hannah Montana | 2008-06-07 14:00:00 |     1706.00000000 |       2
Hannah Montana | 2008-05-01 19:00:00 |     1658.00000000 |       2
Hannah Montana | 2008-06-07 14:00:00 |     1479.00000000 |       1
Hannah Montana | 2008-06-07 14:00:00 |     1479.00000000 |       3
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       1
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       2
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       4
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       1
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       2
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       4
(10 rows)
```

Externe Joins nach Oracle in der WHERE-Klausel

Um Kompatibilität mit Oracle zu erzielen, unterstützt Amazon Redshift die Verwendung des Oracle-Operators für externe Joins (+) in WHERE-Klausel-Join-Bedingungen. Dieser Operator ist ausschließlich für die Verwendung zur Definition von Bedingungen für externe Joins vorgesehen. Verwenden Sie ihn nicht in anderen Zusammenhängen. In den meisten Fällen werden andere Verwendungen dieses Operators stillschweigend ignoriert.

Ein externer Join gibt alle Zeilen zurück, die der entsprechende interne Join zurückgeben würde, und zusätzlich nicht übereinstimmende Zeilen aus einer oder beiden Tabellen. In der FROM-Klausel können Sie externe Joins nach links, rechts und in beide Richtungen angeben. In der WHERE-Klausel können Sie nur externe Joins nach links und rechts angeben.

Um externe Joins für die Tabellen TABLE1 und TABLE2 auszuführen und nicht übereinstimmende Zeilen aus TABLE1 zurückzugeben (ein externer Join nach links), geben Sie `TABLE1 LEFT OUTER JOIN TABLE2` in der FROM-Klausel an oder wenden den Operator (+) auf alle Joining-Spalten aus TABLE2 in der WHERE-Klausel an. Das Ergebnis der Abfrage enthält für alle Zeilen in TABLE1, für die es keine übereinstimmenden Zeilen in TABLE2 gibt, Null-Werte für alle Auswahllistenausdrücke, die Spalten aus TABLE2 enthalten.

Um dasselbe Verhalten für alle Zeilen in TABLE2 zu erhalten, für die es keine übereinstimmenden Zeilen in TABLE1 gibt, geben Sie `TABLE1 RIGHT OUTER JOIN TABLE2` in der FROM-Klausel an oder wenden den Operator (+) auf alle Joining-Spalten aus TABLE1 in der WHERE-Klausel an.

Basissyntax

```
[ WHERE {
[ table1.column1 = table2.column1(+) ]
[ table1.column1(+) = table2.column1 ]
}
```

Die erste Bedingung entspricht:

```
from table1 left outer join table2
on table1.column1=table2.column1
```

Die zweite Bedingung entspricht:

```
from table1 right outer join table2
on table1.column1=table2.column1
```

> **Note**
>
> Die hier gezeigte Syntax deckt den einfachen Fall eines equijoin für ein einzelnes Paar von Joining-Spalten ab. Andere Arten von Vergleichsbedingungen und mehrere Paare von Joining-Spalten sind jedoch ebenfalls gültig.

Beispielsweise definiert die folgende WHERE-Klausel einen externen Join über zwei Paare von Spalten. Der Operator (+) muss in beiden Bedingungen der gleichen Tabelle angefügt werden:

```
where table1.col1 > table2.col1(+)
and table1.col2 = table2.col2(+)
```

Nutzungshinweise

Verwenden Sie, wenn möglich, die OUTER JOIN-Standardsyntax für die FROM-Klausel anstelle des Operators (+) in der WHERE-Klausel. Abfragen, die den Operator (+) enthalten, unterliegen den folgenden Regeln:

- Sie können den Operator (+) nur in der WHERE-Klausel verwenden und nur in Bezug auf Spalten aus Tabellen oder Ansichten.

- Sie können den Operator (+) nicht auf Ausdrücke anwenden. Ein Ausdruck kann jedoch Spalten enthalten, die den Operator (+) verwenden. Beispielsweise gibt die folgende Join-Bedingung einen Syntaxfehler zurück:

  ```
  event.eventid*10(+)=category.catid
  ```

  Die folgende Join-Bedingungen ist jedoch gültig:

  ```
  event.eventid(+)*10=category.catid
  ```

- Sie können den Operator (+) nicht in einem Abfrageblock verwenden, der auch Join-Syntax der FROM-Klausel enthält.
- Wenn für zwei Tabellen ein Join über mehrere Join-Bedingungen ausgeführt wird, müssen Sie den Operator (+) entweder in allen oder in keinen dieser Bedingungen verwenden. Ein Join mit gemischten Syntaxstilen wird ohne Warnung als interner Join ausgeführt.
- Der Operator (+) erstellt keinen externen Join, wenn Sie für eine Tabelle in der externen Abfrage einen Join mit einer Tabelle ausführen, die das Ergebnis einer internen Abfrage ist.
- Um den Operator (+) zu verwenden, um einen externen Join einer Tabelle mit sich selbst auszuführen, müssen Sie in der FROM-Klausel Tabellenaliase definieren und diese in der Join-Bedingung referenzieren:

  ```
  select count(*)
  from event a, event b
  where a.eventid(+)=b.catid;

  count
  -------
  8798
  (1 row)
  ```

- Sie können keine Join-Bedingung kombinieren, die den Operator (+) mit einer OR-Bedingung oder einer IN-Bedingung enthält. Beispiel:

  ```
  select count(*) from sales, listing
  where sales.listid(+)=listing.listid or sales.salesid=0;
  ERROR:  Outer join operator (+) not allowed in operand of OR or IN.
  ```

- Der Operator (+) kann in einer WHERE-Klausel, die einen Join für mehr als zwei Tabellen ausführt, nur einmal auf eine bestimmte Tabelle angewendet werden. Im folgenden Beispiel kann die Tabelle SALES nicht mit dem Operator (+) in zwei aufeinanderfolgenden Joins referenziert werden.

```
select count(*) from sales, listing, event
where sales.listid(+)=listing.listid and sales.dateid(+)=date.dateid;
ERROR:  A table may be outer joined to at most one other table.
```

- Wenn die WHERE-Klauselbedingung für externe Joins eine Spalte aus TABLE2 mit einer Konstante vergleicht, wenden Sie den Operator (+) auf die Spalte an. Wenn Sie den Operator nicht verwenden, werden die Zeilen aus TABLE1 aus dem externen Join, die Null-Werte für die eingeschränkte Spalte enthalten, entfernt. Beispiele hierfür finden Sie unten im Beispielabschnitt.

Beispiele

Die folgende Join-Abfrage gibt einen externen Join nach links für die Tabellen SALES und LISTING für ihre LISTID-Spalten an:

```
select count(*)
from sales, listing
where sales.listid = listing.listid(+);

count
--------
172456
(1 row)
```

Die folgende gleichwertige Abfrage führt zum gleichen Ergebnis, verwendet jedoch die Join-Syntax der FROM-Klausel:

```
select count(*)
from sales left outer join listing on sales.listid = listing.listid;

count
--------
172456
(1 row)
```

Die Tabelle SALES enthält nicht für alle Einträge in der Tabelle LISTING Datensätze, da nicht alle Einträge zu Verkäufen führen. Die folgende Abfrage führt einen externen Join für SALES

und LISTING aus und gibt Zeilen aus LISTING zurück, auch wenn in der Tabelle SALES für eine bestimmte Listen-ID keine Verkäufe eingetragen sind. Die Spalten PRICE und COMM, abgeleitet von der Tabelle SALES, enthalten im Ergebnissatz Null-Werte für diese nicht übereinstimmenden Zeilen.

```
select listing.listid, sum(pricepaid) as price,
sum(commission) as comm
from listing, sales
where sales.listid(+) = listing.listid and listing.listid between 1 and 5
group by 1 order by 1;

listid | price  |  comm
--------+--------+--------
1 | 728.00 | 109.20
2 |        |
3 |        |
4 |  76.00 |  11.40
5 | 525.00 |  78.75
(5 rows)
```

Wenn der Join-Operator der WHERE-Klausel verwendet wird, spielt die Reihenfolge der Tabellen in der FROM-Klausel keine Rolle.

Ein Beispiel für eine komplexere Bedingung in der WHERE-Klausel für einen externen Join ist der Fall, in dem die Bedingung aus einem Vergleich zwischen zwei Tabellenspalten und einem Vergleich mit einer Konstante besteht:

```
where category.catid=event.catid(+) and eventid(+)=796;
```

Beachten Sie, dass der Operator (+) an zwei Stellen verwendet wird: zunächst im Gleichheits-Vergleich zwischen den Tabellen und dann in der Vergleichsbedingungen für die Spalte EVENTID. Das Ergebnis dieser Syntax ist die Bewahrung der Zeilen mit externen Joins, wenn die Einschränkung für EVENTID evaluiert wird. Wenn Sie den Operator (+) aus der Einschränkung EVENTID entfernen, behandelt die Abfrage diese Einschränkung als Filter und nicht als Teil der Bedingung für den externen Join. Daher Zeilen mit dem externen Join, die Null-Werte für EVENTID enthalten, aus dem Ergebnissatz entfernt.

Im folgenden sehen Sie eine vollständige Abfrage, die dieses Verhalten illustriert:

```
select catname, catgroup, eventid
from category, event
```

```
where category.catid=event.catid(+) and eventid(+)=796;

catname | catgroup | eventid
-----------+----------+---------
Classical | Concerts |
Jazz | Concerts |
MLB | Sports   |
MLS | Sports   |
Musicals | Shows    | 796
NBA | Sports   |
NFL | Sports   |
NHL | Sports   |
Opera | Shows    |
Plays | Shows    |
Pop | Concerts |
(11 rows)
```

Die entsprechende Abfrage, die die Syntax der FROM-Klausel verwendet, ist wie folgt:

```
select catname, catgroup, eventid
from category left join event
on category.catid=event.catid and eventid=796;
```

Wenn Sie den zweiten Operator (+) aus der WHERE-Klauselversion dieser Abfrage entfernen, wird nur 1 Zeile zurückgegeben (die Zeile mit `eventid=796`).

```
select catname, catgroup, eventid
from category, event
where category.catid=event.catid(+) and eventid=796;

catname | catgroup | eventid
-----------+----------+---------
Musicals | Shows    | 796
(1 row)
```

GROUP BY-Klausel

Die GROUP BY-Klausel identifiziert die Gruppierungsspalten für die Abfrage. Gruppierungsspalten müssen deklariert werden, wenn die Abfrage aggregierte Werte mit Standardfunktionen wie SUM, AVG und COUNT berechnet. Weitere Informationen finden Sie unter Aggregationsfunktionen.

Syntax

```
GROUP BY group_by_clause [, ...]

group_by_clause := {
    expr |
    GROUPING SETS ( () | group_by_clause [, ...] ) |
    ROLLUP ( expr [, ...] ) |
    CUBE ( expr [, ...] )
    }
```

Parameter

expr

Der Liste der Spalten oder Ausdrücke muss der Liste der nicht aggregierten Ausdrücke in der Auswahlliste der Abfrage entsprechen. Betrachten Sie beispielsweise die folgende einfache Abfrage.

```
select listid, eventid, sum(pricepaid) as revenue,
count(qtysold) as numtix
from sales
group by listid, eventid
order by 3, 4, 2, 1
limit 5;

listid | eventid | revenue | numtix
-------+---------+---------+--------
89397  |      47 |   20.00 |      1
106590 |      76 |   20.00 |      1
124683 |     393 |   20.00 |      1
103037 |     403 |   20.00 |      1
147685 |     429 |   20.00 |      1
(5 rows)
```

In dieser Abfrage besteht die Auswahlliste aus zwei aggregierten Ausdrücken. Der erste verwendet die SUM-Funktion und der zweite verwendet die COUNT-Funktion. Die übrigen beiden Spalten, LISTID und EVENTID, müssen als Gruppierungsspalten deklariert werden.

Ausdrücke in der -Klausel können ebenfalls die Auswahlliste durch Verwendung von Ordinalzahlen referenzieren. Das vorherige Beispiel könnte beispielsweise wie folgt abgekürzt werden.

```
select listid, eventid, sum(pricepaid) as revenue,
count(qtysold) as numtix
from sales
group by 1,2
order by 3, 4, 2, 1
limit 5;

listid | eventid | revenue | numtix
-------+---------+---------+--------
89397  |      47 |   20.00 |      1
106590 |      76 |   20.00 |      1
124683 |     393 |   20.00 |      1
103037 |     403 |   20.00 |      1
147685 |     429 |   20.00 |      1
(5 rows)
```

GROUPING SETS/ROLLUP/CUBE

Sie können die Aggregationserweiterungen GROUPING SETS, ROLLUP und CUBE verwenden, um die Arbeit mehrerer GROUP BY-Operationen in einer einzigen Anweisung auszuführen. Weitere Informationen zu Aggregationserweiterungen und verwandten Funktionen finden Sie unter Aggregationserweiterungen.

Aggregationserweiterungen

Amazon Redshift unterstützt Aggregationserweiterungen, um die Arbeit mehrerer GROUP BY-Operationen in einer einzigen Anweisung zu erledigen.

In den Beispielen zu Aggregationserweiterungen wird die Tabelle `orders` verwendet. Diese enthält Verkaufsdaten für ein Elektronikunternehmen. Sie können `orders` wie folgt erstellen.

```
CREATE TABLE ORDERS (
    ID INT,
    PRODUCT CHAR(20),
    CATEGORY CHAR(20),
    PRE_OWNED CHAR(1),
    COST DECIMAL
);

INSERT INTO ORDERS VALUES
    (0, 'laptop',       'computers',    'T', 1000),
```

```
    (1, 'smartphone',   'cellphones',   'T', 800),
    (2, 'smartphone',   'cellphones',   'T', 810),
    (3, 'laptop',       'computers',    'F', 1050),
    (4, 'mouse',        'computers',    'F', 50);
```

GROUPING SETS

Berechnet einen oder mehrere Gruppierungssätze in einer einzigen Anweisung. Ein Gruppierungssatz ist die Menge einer einzelnen GROUP BY-Klausel, eine Menge von 0 oder mehr Spalten, nach denen Sie die Ergebnismenge einer Abfrage gruppieren können. GROUP BY GROUPING SETS entspricht der Ausführung einer UNION ALL-Abfrage für eine Ergebnismenge, die nach verschiedenen Spalten gruppiert ist. Beispielsweise entspricht GROUP BY GROUPING SETS((a), (b)) GROUP BY a UNION ALL GROUP BY b.

Das folgende Beispiel gibt die Kosten der Produkte der Bestelltabelle zurück, gruppiert sowohl nach den Produktkategorien als auch nach der Art der verkauften Produkte.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY GROUPING SETS(category, product);

       category       |       product        | total
----------------------+----------------------+-------
 computers            |                      |  2100
 cellphones           |                      |  1610
                      | laptop               |  2050
                      | smartphone           |  1610
                      | mouse                |    50

(5 rows)
```

ROLLUP

Geht von einer Hierarchie aus, bei der vorangehende Spalten als übergeordnete Spalten der nachfolgenden Spalten betrachtet werden. ROLLUP gruppiert Daten nach den bereitgestellten Spalten und gibt zusätzlich zu den gruppierten Zeilen weitere Zwischensummenzeilen zurück, die die Summen auf allen Ebenen der Gruppierungsspalten darstellen. Beispielsweise können Sie GROUP BY ROLLUP((a), (b)) verwenden, um eine Ergebnismenge zurückzugeben, die zuerst nach a und dann nach b gruppiert ist, wobei angenommen wird, dass b ein Unterabschnitt von a ist. ROLLUP gibt auch eine Zeile mit der gesamten Ergebnismenge ohne Gruppierungsspalten zurück.

GROUP BY ROLLUP((a), (b)) entspricht GROUP BY GROUPING SETS((a,b), (a), ()).

Im folgenden Beispiel werden die Kosten der Produkte der Bestelltabelle zurückgegeben, zuerst nach Kategorie und dann nach Produkt gruppiert, wobei „product“ (Produkt) eine Unterteilung von „category“ (Kategorie) darstellt.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY ROLLUP(category, product) ORDER BY 1,2;

       category       |       product        | total
----------------------+----------------------+-------
 cellphones           | smartphone           |  1610
 cellphones           |                      |  1610
 computers            | laptop               |  2050
 computers            | mouse                |    50
 computers            |                      |  2100
                      |                      |  3710
(6 rows)
```

CUBE

Gruppiert Daten nach den bereitgestellten Spalten und gibt zusätzlich zu den gruppierten Zeilen weitere Zwischensummenzeilen zurück, die die Summen auf allen Ebenen der Gruppierungsspalten darstellen. CUBE gibt dieselben Zeilen wie ROLLUP zurück und fügt zusätzliche Zwischensummenzeilen für jede Kombination von Gruppierungsspalten hinzu, die nicht von ROLLUP abgedeckt wird. Beispielsweise können Sie GROUP BY CUBE ((a), (b)) verwenden, um eine Ergebnismenge zurückzugeben, die zuerst nach a und dann nach b – unter der Annahme, dass b ein Unterabschnitt von a ist – und dann nur nach b gruppiert ist. CUBE gibt auch eine Zeile mit der gesamten Ergebnismenge ohne Gruppierungsspalten zurück.

GROUP BY CUBE((a), (b)) entspricht GROUP BY GROUPING SETS((a, b), (a), (b), ()).

Im folgenden Beispiel werden die Kosten der Produkte der Bestelltabelle zurückgegeben, zuerst nach Kategorie und dann nach Produkt gruppiert, wobei „product“ (Produkt) eine Unterteilung von „category“ (Kategorie) darstellt. Im Gegensatz zum vorherigen Beispiel für ROLLUP gibt die Anweisung Ergebnisse für jede Kombination von Gruppierungsspalten zurück.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY CUBE(category, product) ORDER BY 1,2;
```

```
       category       |       product        | total
----------------------+----------------------+-------
 cellphones           | smartphone           |  1610
 cellphones           |                      |  1610
 computers            | laptop               |  2050
 computers            | mouse                |    50
 computers            |                      |  2100
                      | laptop               |  2050
                      | mouse                |    50
                      | smartphone           |  1610
                      |                      |  3710
(9 rows)
```

GROUPING/GROUPING_ID-Funktionen

ROLLUP und CUBE fügen der Ergebnismenge NULL-Werte hinzu, um Zwischensummenzeilen anzugeben. So gibt GROUP BY ROLLUP((a), (b)) beispielsweise eine oder mehrere Zeilen zurück, die in der Gruppierungsspalte b den Wert NULL haben, um anzugeben, dass es sich um Zwischensummen von Feldern in der Gruppierungsspalte a handelt. Diese NULL-Werte dienen nur dazu, das Format der Rückgabe-Tupel einzuhalten.

Wenn Sie GROUP BY-Operationen mit ROLLUP und CUBE für Relationen ausführen, die selbst NULL-Werte speichern, kann dies zu Ergebnismengen mit Zeilen führen, die identische Gruppierungsspalten zu haben scheinen. Zurück zum vorherigen Beispiel: Wenn die Gruppierungsspalte b einen gespeicherten NULL-Wert enthält, gibt GROUP BY ROLLUP((a), (b)) eine Zeile mit dem Wert NULL in Gruppierungsspalte b zurück, bei der es sich nicht um eine Zwischensumme handelt.

Um zwischen NULL-Werten, die von ROLLUP und CUBE erstellt wurden, und den in den Tabellen selbst gespeicherten NULL-Werten zu unterscheiden, können Sie die GROUPING-Funktion oder ihren Alias GROUPING_ID verwenden. GROUPING verwendet einen einzelnen Gruppierungssatz als Argument und gibt für jede Zeile in der Ergebnismenge einen 0- oder 1-Bit-Wert entsprechend der Gruppierungsspalte an der betreffenden Position zurück und wandelt diesen Wert dann in eine Ganzzahl um. Wenn der Wert an dieser Position ein NULL-Wert ist, der durch eine Aggregationserweiterung erstellt wurde, gibt GROUPING 1 zurück. Für alle anderen Werte einschließlich gespeicherter NULL-Werte wird 0 zurückgegeben.

Beispielsweise kann GROUPING(Kategorie, Produkt) die folgenden Werte für eine bestimmte Zeile zurückgeben, je nach Gruppierungsspaltenwerten für diese Zeile. Für die Zwecke dieses Beispiels

sind alle NULL-Werte in der Tabelle NULL-Werte, die durch eine Aggregationserweiterung erstellt wurden.

Kategoriespalte	Produktspalte	Bitwert der GROUPING-Funktion	Dezimalwert
nicht NULL	nicht NULL	00	0
nicht NULL	NULL	01	1
NULL	nicht NULL	10	2
NULL	NULL	11	3

GROUPING-Funktionen werden im SELECT-Listenbereich der Abfrage im folgenden Format angezeigt.

```
SELECT ... [GROUPING( expr )...] ...
  GROUP BY ... {CUBE | ROLLUP| GROUPING SETS} ( expr ) ...
```

Das folgende Beispiel entspricht dem vorherigen Beispiel für CUBE, enthält jedoch zusätzliche GROUPING-Funktionen für die Gruppierungssätze

```
SELECT category, product,
       GROUPING(category) as grouping0,
       GROUPING(product) as grouping1,
       GROUPING(category, product) as grouping2,
       sum(cost) as total
FROM orders
GROUP BY CUBE(category, product) ORDER BY 3,1,2;

       category       |       product        | grouping0 | grouping1 | grouping2 |
 total
----------------------+----------------------+-----------+-----------+-----------
+-------
 cellphones           | smartphone           |         0 |         0 |         0 |
 1610
 cellphones           |                      |         0 |         1 |         1 |
 1610
```

```
 computers            | laptop               |         0 |         0 |         0 |
 2050
 computers            | mouse                |         0 |         0 |         0 |
 50
 computers            |                      |         0 |         1 |         1 |
 2100
                      | laptop               |         1 |         0 |         2 |
 2050
                      | mouse                |         1 |         0 |         2 |
 50
                      | smartphone           |         1 |         0 |         2 |
 1610
                      |                      |         1 |         1 |         3 |
 3710
(9 rows)
```

Partielle ROLLUP- und CUBE-Operationen

Sie können ROLLUP- und CUBE-Operationen nur mit einem Teil der Zwischensummen ausführen.

Die Syntax für partielle ROLLUP- und CUBE-Operationen lautet wie folgt.

```
GROUP BY expr1, { ROLLUP | CUBE }(expr2, [, ...])
```

Hier erstellt die GROUP BY-Klausel nur Zwischensummenzeilen auf der Ebene expr2 und höher.

Die folgenden Beispiele zeigen partielle ROLLUP- und CUBE-Operationen in der Bestelltabelle, wobei zuerst danach gruppiert wird, ob ein Produkt gebraucht ist, und dann ROLLUP und CUBE für die Kategorie- und Produktspalten ausgeführt werden.

```
SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY pre_owned, ROLLUP(category, product) ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
 F         | computers            |                      |        2 |  1100
```

```
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
 F         |                      |                      |        6 |  1100
 T         |                      |                      |        6 |  2610
(9 rows)

SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY pre_owned, CUBE(category, product) ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
 F         | computers            |                      |        2 |  1100
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
 F         |                      | laptop               |        4 |  1050
 F         |                      | mouse                |        4 |    50
 T         |                      | laptop               |        4 |  1000
 T         |                      | smartphone           |        4 |  1610
 F         |                      |                      |        6 |  1100
 T         |                      |                      |        6 |  2610
(13 rows)
```

Da die Spalte für gebrauchte Produkte nicht in den ROLLUP- und CUBE-Operationen enthalten ist, gibt es keine Gesamtsummenzeile, die alle anderen Zeilen enthält.

Verkettete Gruppierung

Sie können mehrere GROUPING SETS-/ROLLUP-/CUBE-Klauseln verketten, um verschiedene Ebenen von Zwischensummen zu berechnen. Verkettete Gruppierungen geben das kartesische Produkt der bereitgestellten Gruppierungssätze zurück.

Die Syntax für die Verkettung von GROUPING SETS-/ROLLUP-/CUBE-Klauseln lautet wie folgt.

```
GROUP BY {ROLLUP|CUBE|GROUPING SETS}(expr1[, ...]),
         {ROLLUP|CUBE|GROUPING SETS}(expr1[, ...])[, ...]
```

Im folgenden Beispiel sehen Sie, wie eine kleine verkettete Gruppierung eine große endgültige Ergebnismenge ergeben kann.

```
SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY CUBE(category, product), GROUPING SETS(pre_owned, ())
ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
           | cellphones           | smartphone           |        1 |  1610
           | computers            | laptop               |        1 |  2050
           | computers            | mouse                |        1 |    50
 F         | computers            |                      |        2 |  1100
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
           | cellphones           |                      |        3 |  1610
           | computers            |                      |        3 |  2100
 F         |                      | laptop               |        4 |  1050
 F         |                      | mouse                |        4 |    50
 T         |                      | laptop               |        4 |  1000
 T         |                      | smartphone           |        4 |  1610
           |                      | laptop               |        5 |  2050
           |                      | mouse                |        5 |    50
           |                      | smartphone           |        5 |  1610
 F         |                      |                      |        6 |  1100
 T         |                      |                      |        6 |  2610
           |                      |                      |        7 |  3710
(22 rows)
```

Verschachtelte Gruppierung

Sie können GROUPING SETS-/ROLLUP-/CUBE-Operationen als Ihre GROUPING SETS expr verwenden, um eine verschachtelte Gruppierung zu bilden. Die Untergruppierung innerhalb verschachtelter GROUPING SETS ist abgeflacht.

Die Syntax für die verschachtelte Gruppierung lautet wie folgt.

```
GROUP BY GROUPING SETS({ROLLUP|CUBE|GROUPING SETS}(expr[, ...])[, ...])
```

Betrachten Sie das folgende Beispiel.

```
SELECT category, product, pre_owned,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY GROUPING SETS(ROLLUP(category), CUBE(product, pre_owned))
ORDER BY 4,1,2,3;

       category       |       product        | pre_owned | group_id | total
----------------------+----------------------+-----------+----------+-------
 cellphones           |                      |           |        3 |  1610
 computers            |                      |           |        3 |  2100
                      | laptop               | F         |        4 |  1050
                      | laptop               | T         |        4 |  1000
                      | mouse                | F         |        4 |    50
                      | smartphone           | T         |        4 |  1610
                      | laptop               |           |        5 |  2050
                      | mouse                |           |        5 |    50
                      | smartphone           |           |        5 |  1610
                      |                      | F         |        6 |  1100
                      |                      | T         |        6 |  2610
                      |                      |           |        7 |  3710
                      |                      |           |        7 |  3710
(13 rows)
```

Beachten Sie, dass die Zeile, die die Gesamtsumme darstellt, dupliziert wird, da sowohl ROLLUP(categorie) als auch CUBE(product, pre_owned) den Gruppierungssatz () enthalten.

Nutzungshinweise

- Die GROUP BY-Klausel unterstützt bis zu 64 Gruppierungssätze. Im Falle von ROLLUP und CUBE oder einer Kombination von GROUPING SETS, ROLLUP und CUBE gilt diese Einschränkung für die implizite Anzahl an Gruppierungssätzen. So zählt beispielsweise GROUP BY CUBE((a), (b)) als 4 und nicht als 2 Gruppierungssätze.
- Bei Verwendung von Aggregationserweiterungen können Sie keine Konstanten als Gruppierungsspalten verwenden.
- Sie können keinen Gruppierungssatz erstellen, der doppelte Spalten enthält.

HAVING-Klausel

Die HAVING-Klausel wendet eine Bedingung auf den gruppierten Zwischenergebnissatz an, den eine Abfrage zurückgibt.

Syntax

```
[ HAVING condition ]
```

Sie können beispielsweise die Ergebnisse einer SUM-Funktion einschränken:

```
having sum(pricepaid) >10000
```

Die HAVING-Bedingung wird angewendet, nachdem alle WHERE-Klauselbedingungen angewendet wurden und die GROUP BY-Operationen abgeschlossen sind.

Die Bedingung selbst hat das gleiche Format wie eine WHERE-Klauselbedingung.

Nutzungshinweise

- Bei jeder, in einer -Klauselbedingung referenzierten Spalte muss es sich entweder um eine Gruppierungsspalte handeln oder um eine Spalte, die sich auf das Ergebnis einer aggregierten Funktion bezieht.
- In einer HAVING-Klausel können Sie Folgendes nicht angeben:
  - Eine Ordinalzahl, die ein Auswahllistenelement referenziert. Nur die Klauseln GROUP BY und ORDER BY akzeptieren Ordinalzahlen.

Beispiele

Die folgende Abfrage berechnet den Ticket-Gesamtverkauf für alle Veranstaltungen nach Namen. Anschließend werden Veranstaltungen entfernt, deren Gesamtverkauf weniger als 800.000 USD betrug. Die HAVING-Bedingung wird auf die Ergebnisse der Aggregierungsfunktion in der Auswahlliste angewendet: `sum(pricepaid)`.

```
select eventname, sum(pricepaid)
from sales join event on sales.eventid = event.eventid
group by 1
having sum(pricepaid) > 800000
order by 2 desc, 1;
```

```
eventname        |    sum
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
```

Die folgende Abfrage berechnet einen ähnlichen Ergebnissatz. In diesem Fall wird die HAVING-Bedingung jedoch auf ein Aggregat angewendet, das nicht in der Auswahlliste angegeben ist: `sum(qtysold)`. Veranstaltungen, für weniger als 2.000 Tickets verkauft wurden, werden aus dem Endergebnis entfernt.

```
select eventname, sum(pricepaid)
from sales join event on sales.eventid = event.eventid
group by 1
having sum(qtysold) >2000
order by 2 desc, 1;

eventname        |    sum
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
Chicago          |  790993.00
Spamalot         |  714307.00
```

Die folgende Abfrage berechnet den Ticket-Gesamtverkauf für alle Veranstaltungen nach Namen. Anschließend werden Veranstaltungen entfernt, deren Gesamtverkauf weniger als 800.000 USD betrug. Die HAVING-Bedingung wird auf die Ergebnisse der Aggregatfunktion in der Auswahlliste angewendet, wobei der Alias pp für `sum(pricepaid)` verwendet wird.

```
select eventname, sum(pricepaid) as pp
from sales join event on sales.eventid = event.eventid
group by 1
having pp > 800000
order by 2 desc, 1;
```

```
eventname        |    pp
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
```

QUALIFY-Klausel

Die QUALIFY-Klausel filtert die Ergebnisse einer zuvor berechneten Fensterfunktion anhand benutzerdefinierter Suchbedingungen. Sie können die Klausel verwenden, um Filterbedingungen auf das Ergebnis einer Fensterfunktion anzuwenden, ohne eine Unterabfrage zu verwenden.

Die Klausel ähnelt der HAVING-Klausel, die eine Bedingung anwendet, um Zeilen aus einer WHERE-Klausel weiter zu filtern. Der Unterschied zwischen QUALIFY und HAVING besteht darin, dass gefilterte Ergebnisse aus der QUALIFY-Klausel auf dem Ergebnis der Ausführung von Fensterfunktionen für die Daten basieren können. Sie können sowohl die QUALIFY- als auch die HAVING-Klausel in einer Abfrage verwenden.

Syntax

```
QUALIFY condition
```

> **Note**
>
> Wenn Sie die QUALIFY-Klausel direkt nach der FROM-Klausel benutzen, muss vor der QUALIFY-Klausel vor dem Namen der FROM-Beziehung ein Alias angegeben werden.

Beispiele

Für die Beispiele in diesem Abschnitt werden die unten angegebenen Beispieldaten verwendet.

```
create table store_sales (ss_sold_date date, ss_sold_time time,
               ss_item text, ss_sales_price float);
insert into store_sales values ('2022-01-01', '09:00:00', 'Product 1', 100.0),
                               ('2022-01-01', '11:00:00', 'Product 2', 500.0),
```

```
                                ('2022-01-01', '15:00:00', 'Product 3', 20.0),
                                ('2022-01-01', '17:00:00', 'Product 4', 1000.0),
                                ('2022-01-01', '18:00:00', 'Product 5', 30.0),
                                ('2022-01-02', '10:00:00', 'Product 6', 5000.0),
                                ('2022-01-02', '16:00:00', 'Product 7', 5.0);
```

Das folgende Beispiel zeigt, wie Sie die beiden teuersten Artikel finden, die täglich nach 12:00 Uhr verkauft werden.

```
SELECT *
FROM store_sales ss
WHERE ss_sold_time > time '12:00:00'
QUALIFY row_number()
OVER (PARTITION BY ss_sold_date ORDER BY ss_sales_price DESC) <= 2


 ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price
--------------+--------------+-----------+----------------
 2022-01-01   | 17:00:00     | Product 4 |           1000
 2022-01-01   | 18:00:00     | Product 5 |             30
 2022-01-02   | 16:00:00     | Product 7 |              5
```

Sie können dann den letzten Artikel finden, der an jedem Tag verkauft wurde.

```
SELECT *
FROM store_sales ss
QUALIFY last_value(ss_item)
OVER (PARTITION BY ss_sold_date ORDER BY ss_sold_time ASC
      ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) = ss_item;

ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price
--------------+--------------+-----------+----------------
 2022-01-01   | 18:00:00     | Product 5 |             30
 2022-01-02   | 16:00:00     | Product 7 |              5
```

Im folgenden Beispiel werden dieselben Datensätze wie bei der vorherigen Abfrage zurückgegeben, d. h. der letzte verkaufte Artikel an jedem Tag, aber die QUALIFY-Klausel wird nicht verwendet.

```
SELECT * FROM (
  SELECT *,
  last_value(ss_item)
  OVER (PARTITION BY ss_sold_date ORDER BY ss_sold_time ASC
```

```
        ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) ss_last_item
  FROM store_sales ss
)
WHERE ss_last_item = ss_item;

 ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price | ss_last_item
--------------+--------------+-----------+----------------+--------------
 2022-01-02   | 16:00:00     | Product 7 |              5 | Product 7
 2022-01-01   | 18:00:00     | Product 5 |             30 | Product 5
```

UNION, INTERSECT und EXCEPT

Themen

- Syntax
- Parameter
- Reihenfolge der Evaluierung für Satzoperatoren
- Nutzungshinweise
- Beispiel für UNION-Abfragen
- Beispiel für die UNION ALL-Abfrage
- Beispiel für INTERSECT-Abfragen
- Beispiel für die EXCEPT-Abfrage

Die Satzoperatoren UNION, INTERSECT und EXCEPT werden verwendet, um die Ergebnisse von zwei getrennten Abfrageausdrücken zu vergleichen und zusammenzuführen. Wenn Sie beispielsweise wissen möchten, welche Benutzer einer Website sowohl Käufer als auch Verkäufer sind, die Namen jedoch in getrennten Spalten oder Tabellen gespeichert sind, können Sie die Überschneidung zwischen diesen beiden Arten von Benutzern finden. Wenn Sie wissen möchten, welche Benutzer einer Website Käufer, jedoch nicht Verkäufer sind, können Sie den Operator EXCEPT verwenden, um den Unterschied zwischen diesen beiden Listen von Benutzern zu finden. Wenn Sie eine Liste aller Benutzer unabhängig von der Rolle erstellen möchten, können Sie den Operator UNION verwenden.

Syntax

```
query
{ UNION [ ALL ] | INTERSECT | EXCEPT | MINUS }
query
```

Parameter

query

Ein Abfrageausdruck, der in Form seiner Auswahlliste einem zweiten Abfrageausdruck entspricht, der dem Operator UNION, INTERSECT oder EXCEPT folgt. Die beiden Ausdrücke müssen die gleiche Zahl von Ausgabespalten mit kompatiblen Datentypen enthalten. Andernfalls können die beiden Ergebnissätze nicht verglichen und zusammengeführt werden. Satzoperationen lassen die implizite Umwandlung zwischen unterschiedlichen Kategorien von Datentypen nicht zu. Weitere Informationen finden Sie unter Kompatibilität von Typen und Umwandlung zwischen Typen.

Sie können Abfragen erstellen, die eine unbegrenzte Anzahl von Abfrageausdrücken enthalten, und sie mithilfe der Operatoren UNION, INTERSECT und EXCEPT in beliebigen Kombinationen verbinden. Beispielsweise ist die folgende Abfragestruktur gültig, wenn die Tabellen T1, T2 und T3 kompatible Sätze von Spalten enthalten:

```
select * from t1
union
select * from t2
except
select * from t3
order by c1;
```

UNION

Satzoperation, die Zeilen aus zwei Abfrageausdrücken zurückgibt, unabhängig davon, ob die Zeilen von einem oder von beiden Ausdrücken abgeleitet werden.

INTERSECT

Satzoperation, die Zeilen zurückgibt, die von zwei Abfrageausdrücken abgeleitet werden. Zeilen, die nicht von beiden Ausdrücken zurückgegeben werden, werden verworfen.

EXCEPT | MINUS

Satzoperation, die Zeilen zurückgibt, die von einem von zwei Abfrageausdrücken abgeleitet werden. Um sich für das Ergebnis zu qualifizieren, dürfen Zeilen zwar in der ersten Ergebnistabelle, nicht jedoch in der zweiten vorhanden sein. MINUS und EXCEPT sind exakte Synonyme.

ALL

Das Schlüsselwort ALL behält alle duplizierten Zeilen, die von UNION erstellt werden. Wenn das Schlüsselwort ALL nicht verwendet wird, besteht das Standardverhalten darin, diese Duplikate zu verwerfen. INTERSECT ALL, EXCEPT ALL und MINUS ALL werden nicht unterstützt.

Reihenfolge der Evaluierung für Satzoperatoren

Die Satzoperatoren UNION und EXCEPT sind links-assoziativ. Wenn keine Klammern angegeben werden, um die Reihenfolge zu beeinflussen, wird eine Kombination dieser Satzoperatoren von links nach rechts ausgewertet. Beispielsweise wird in der folgenden Abfrage der Operator UNION von T1 und T2 zuerst ausgewertet. Anschließend wird die Operation EXCEPT für das UNION-Ergebnis ausgeführt:

```
select * from t1
union
select * from t2
except
select * from t3
order by c1;
```

Der Operator INTERSECT hat Vorrang vor den Operatoren UNION und EXCEPT, wenn in derselben Abfrage eine Kombination von Operatoren verwendet wird. Beispielsweise wird in der folgenden Abfrage die Schnittmenge von T2 und T3 ausgewertet und anschließend mit T1 vereinigt:

```
select * from t1
union
select * from t2
intersect
select * from t3
order by c1;
```

Durch die Hinzufügung von Klammern können Sie eine andere Reihenfolge für die Auswertung erzwingen. Im folgenden Fall wird für das Ergebnis von UNION für T1 und T2 eine Überschneidung mit T3 ausgewertet. Die Abfrage führt wahrscheinlich zu einem anderen Ergebnis.

```
(select * from t1
union
select * from t2)
```

```
intersect
(select * from t3)
order by c1;
```

Nutzungshinweise

- Die Spaltennamen, die im Ergebnis einer Satzoperationsabfrage zurückgegeben werden, sind die Spaltennamen (Spaltenaliase) aus den Tabellen im ersten Abfrageausdruck. Da diese Spaltennamen potenziell irreführend sein können, da die Werte in der Spalte aus Tabellen auf beiden Seiten des Satzoperators abgeleitet werden, sollten Sie möglicherweise sinnvolle Aliase für den Ergebnissatz bereitstellen.
- Ein Abfrageausdruck, der einem Operator vorangeht, sollte keine ORDER BY-Klausel enthalten. Eine ORDER BY-Klausel gibt nur dann sinnvolle, sortierte Ergebnisse zurück, wenn sie am Ende einer Abfrage mit Satzoperatoren verwendet wird. In diesem Fall wird die ORDER BY-Klausel auf die Endergebnisse aller Satzoperationen angewendet. Die Abfrage ganz außen kann auch LIMIT- und OFFSET-Standardklauseln enthalten.
- Wenn Abfragen mit Satzoperatoren Dezimalergebnisse zurückgeben, geben die entsprechenden Ergebnisspalten Werte mit derselben Genauigkeit und Skalierung zurück. In der folgenden Abfrage, in der T1.REVENUE eine DECIMAL(10,2)-Spalte ist und T2.REVENUE eine DECIMAL(8,4)-Spalte ist, ist das Dezimalergebnis DECIMAL(12,4):

  ```
  select t1.revenue union select t2.revenue;
  ```

  Die Skalierung ist 4, da dies die maximale Skalierung der beiden Spalten ist. Die Genauigkeit ist 12, da T1.REVENUE 8 Stellen links vom Dezimalkomma erfordert (12 – 4 = 8). Dieser Vorgang stellt sicher, dass alle Werte aus beiden Seiten der UNION-Operation in das Ergebnis passen. Für 64-Bit-Werte ist die maximale Ergebnisgenauigkeit 19 und die maximale Ergebnisskalierung 18. Für 128-Bit-Werte ist die maximale Ergebnisgenauigkeit 38 und die maximale Ergebnisskalierung 37.

  Wenn der Ergebnisdatentyp die Genauigkeits- und Skalierungslimits von Amazon Redshift überschreitet, gibt die Abfrage einen Fehler zurück.
- Bei Satzoperationen werden zwei Zeilen als identisch behandelt, wenn für jedes korrespondierendes Spaltenpaar die beiden Datenwerte beide gleich oder beide NULL sind. Wenn beispielsweise die Tabellen T1 und T2 beide nur eine Spalte und eine Zeile enthalten und diese Zeile in beiden Tabellen NULL ist, gibt eine INTERSECT-Operation für diese Tabellen diese Zeile zurück.

Beispiel für UNION-Abfragen

In der folgenden UNION-Abfrage werden Zeilen in der Tabelle SALES mit Zeilen in der Tabelle LISTING zusammengeführt. Aus jeder Tabelle werden drei kompatible Spalten ausgewählt. In diesem Fall haben die korrespondierenden Spalten die gleichen Namen und Datentypen.

Das Endergebnis wird nach der ersten Spalte in der Tabelle LISTING angeordnet und ist auf die 5 Zeilen mit dem höchsten LISTID-Wert begrenzt.

```
select listid, sellerid, eventid from listing
union select listid, sellerid, eventid from sales
order by listid, sellerid, eventid desc limit 5;

listid | sellerid | eventid
--------+----------+---------
1 |    36861 |    7872
2 |    16002 |    4806
3 |    21461 |    4256
4 |     8117 |    4337
5 |     1616 |    8647
(5 rows)
```

Das folgende Beispiel zeigt, wie Sie der Ausgabe einer UNION-Abfrage einen Literalwert hinzufügen können, um zu sehen, durch welche Abfrageausdrücke die einzelnen Zeilen im Ergebnissatz jeweils generiert wurden. Die Abfrage identifiziert Zeilen aus dem ersten Abfrageausdruck als „B“ (für Käufer) und Zeilen aus dem zweiten Abfrageausdruck als „S“ (für Verkäufer).

Die Abfrage identifiziert Käufer und Verkäufer für Tickettransaktionen, die einen Wert von mindestens 10.000 USD haben. Der einzige Unterschied zwischen den beiden Abfrageausdrücken auf beiden Seiten des UNION-Operators besteht in der Joining-Spalte für die Tabelle SALES.

```
select listid, lastname, firstname, username,
pricepaid as price, 'S' as buyorsell
from sales, users
where sales.sellerid=users.userid
and pricepaid >=10000
union
select listid, lastname, firstname, username, pricepaid,
'B' as buyorsell
from sales, users
where sales.buyerid=users.userid
and pricepaid >=10000
```

```
order by 1, 2, 3, 4, 5;

listid | lastname | firstname | username |   price   | buyorsell
--------+----------+-----------+----------+-----------+-----------
209658 | Lamb     | Colette   | VOR15LYI |  10000.00 | B
209658 | West     | Kato      | ELU81XAA |  10000.00 | S
212395 | Greer    | Harlan    | GXO71KOC |  12624.00 | S
212395 | Perry    | Cora      | YWR73YNZ |  12624.00 | B
215156 | Banks    | Patrick   | ZNQ69CLT |  10000.00 | S
215156 | Hayden   | Malachi   | BBG56AKU |  10000.00 | B
(6 rows)
```

Das folgende Beispiel verwendet einen UNION ALL-Operator, da duplizierte Zeilen im Ergebnis beibehalten werden müssen, wenn gefunden. Die Abfrage gibt für eine spezifische Reihe von Ereignis-IDs 0 oder mehr Zeilen für jeden Verkauf zurück, der mit den einzelnen Ereignissen verknüpft ist, und 0 oder 1 Zeile für jede Auflistung dieses Ereignisses. Die Ereignis-IDs sind für die einzelnen Zeilen in den Tabellen LISTING und EVENT eindeutig. Es gibt jedoch möglicherweise mehrere Verkäufe für dieselbe Kombination von Ereignis- und Auflistungs-IDs in der Tabelle SALES.

Die dritte Spalte im Ergebnissatz identifiziert die Quelle der Zeile. Wenn sie aus der Tabelle SALES stammt, wird sie in der Spalte SALESROW mit „Ja“ markiert. (SALESROW ist ein Alias für SALES.LISTID.) Wenn sie aus der Tabelle LISTING stammt, wird sie in der Spalte SALESROW mit „Nein“ markiert.

In diesem Fall besteht der Ergebnissatz aus drei Verkaufszeilen für Auflistung 500, Ereignis 7787. Mit anderen Worten, für diese Kombination von Auflistung und Ereignis fanden drei verschiedene Transaktionen statt. Die beiden anderen Auflistungen, 501 und 502, generierten keine Verkäufe. Daher stammt die einzige Zeile, die die Abfrage für diese Auflistungs-IDs generiert, aus der Tabelle LISTING (SALESROW = „Nein“).

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union all
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
```

```
7787 |    500 | No
7787 |    500 | Yes
7787 |    500 | Yes
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(6 rows)
```

Wenn Sie die gleiche Abfrage ohne das Schlüsselwort ALL ausführen, gibt das Ergebnis nur eine der Verkaufstransaktionen zurück.

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(4 rows)
```

Beispiel für die UNION ALL-Abfrage

Das folgende Beispiel verwendet einen UNION ALL-Operator, da duplizierte Zeilen im Ergebnis beibehalten werden müssen, wenn gefunden. Die Abfrage gibt für eine spezifische Reihe von Ereignis-IDs 0 oder mehr Zeilen für jeden Verkauf zurück, der mit den einzelnen Ereignissen verknüpft ist, und 0 oder 1 Zeile für jede Auflistung dieses Ereignisses. Die Ereignis-IDs sind für die einzelnen Zeilen in den Tabellen LISTING und EVENT eindeutig. Es gibt jedoch möglicherweise mehrere Verkäufe für dieselbe Kombination von Ereignis- und Auflistungs-IDs in der Tabelle SALES.

Die dritte Spalte im Ergebnissatz identifiziert die Quelle der Zeile. Wenn sie aus der Tabelle SALES stammt, wird sie in der Spalte SALESROW mit „Ja“ markiert. (SALESROW ist ein Alias für SALES.LISTID.) Wenn sie aus der Tabelle LISTING stammt, wird sie in der Spalte SALESROW mit „Nein“ markiert.

In diesem Fall besteht der Ergebnissatz aus drei Verkaufszeilen für Auflistung 500, Ereignis 7787. Mit anderen Worten, für diese Kombination von Auflistung und Ereignis fanden drei verschiedene Transaktionen statt. Die beiden anderen Auflistungen, 501 und 502, generierten keine Verkäufe. Daher stammt die einzige Zeile, die die Abfrage für diese Auflistungs-IDs generiert, aus der Tabelle LISTING (SALESROW = „Nein“).

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union all
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
7787 |    500 | Yes
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(6 rows)
```

Wenn Sie die gleiche Abfrage ohne das Schlüsselwort ALL ausführen, gibt das Ergebnis nur eine der Verkaufstransaktionen zurück.

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
6473 |    501 | No
```

```
5108 |    502 | No
(4 rows)
```

Beispiel für INTERSECT-Abfragen

Vergleichen Sie das folgende Beispiel mit dem ersten UNION-Beispiel. Der einzige Unterschied zwischen den beiden Beispielen besteht im verwendeten Satzoperator. Die Ergebnisse unterscheiden sich jedoch stark. Nur eine Zeile ist identisch:

```
235494 |    23875 |    8771
```

Dies ist die einzige Zeile im begrenzten Ergebnis von 5 Zeilen, die in beiden Tabellen gefunden wurde.

```
select listid, sellerid, eventid from listing
intersect
select listid, sellerid, eventid from sales
order by listid desc, sellerid, eventid
limit 5;

listid | sellerid | eventid
--------+----------+---------
235494 |    23875 |    8771
235482 |     1067 |    2667
235479 |     1589 |    7303
235476 |    15550 |     793
235475 |    22306 |    7848
(5 rows)
```

Die folgende Abfrage sucht Veranstaltungen (für die Tickets verkauft wurden), die im März sowohl in New York City als auch in Los Angeles stattfanden. Der Unterschied zwischen den beiden Abfrageausdrücken auf beiden Seiten des UNION-Operators besteht in der Einschränkung für die Spalte VENUECITY.

```
select distinct eventname from event, sales, venue
where event.eventid=sales.eventid and event.venueid=venue.venueid
and date_part(month,starttime)=3 and venuecity='Los Angeles'
intersect
select distinct eventname from event, sales, venue
where event.eventid=sales.eventid and event.venueid=venue.venueid
```

```
and date_part(month,starttime)=3 and venuecity='New York City'
order by eventname asc;

eventname
----------------------------
A Streetcar Named Desire
Dirty Dancing
Electra
Running with Annalise
Hairspray
Mary Poppins
November
Oliver!
Return To Forever
Rhinoceros
South Pacific
The 39 Steps
The Bacchae
The Caucasian Chalk Circle
The Country Girl
Wicked
Woyzeck
(16 rows)
```

Beispiel für die EXCEPT-Abfrage

Die Tabelle CATEGORY in der Datenbank TICKIT enthält die folgenden 11 Zeilen:

```
 catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
   1   | Sports   | MLB       | Major League Baseball
   2   | Sports   | NHL       | National Hockey League
   3   | Sports   | NFL       | National Football League
   4   | Sports   | NBA       | National Basketball Association
   5   | Sports   | MLS       | Major League Soccer
   6   | Shows    | Musicals  | Musical theatre
   7   | Shows    | Plays     | All non-musical theatre
   8   | Shows    | Opera     | All opera and light opera
   9   | Concerts | Pop       | All rock and pop music concerts
  10   | Concerts | Jazz      | All jazz singers and bands
  11   | Concerts | Classical | All symphony, concerto, and choir concerts
(11 rows)
```

Angenommen, eine Tabelle namens CATEGORY_STAGE (eine Staging-Tabelle) enthält eine einzige zusätzliche Zeile:

```
 catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports   | MLB       | Major League Baseball
2 | Sports   | NHL       | National Hockey League
3 | Sports   | NFL       | National Football League
4 | Sports   | NBA       | National Basketball Association
5 | Sports   | MLS       | Major League Soccer
6 | Shows    | Musicals  | Musical theatre
7 | Shows    | Plays     | All non-musical theatre
8 | Shows    | Opera     | All opera and light opera
9 | Concerts | Pop       | All rock and pop music concerts
10 | Concerts | Jazz      | All jazz singers and bands
11 | Concerts | Classical | All symphony, concerto, and choir concerts
12 | Concerts | Comedy    | All stand up comedy performances
(12 rows)
```

Gibt den Unterschied zwischen den beiden Tabellen zurück. Mit anderen Worten, gibt Zeilen zurück, die in der Tabelle CATEGORY_STAGE, jedoch nicht in der Tabelle CATEGORY enthalten sind:

```
select * from category_stage
except
select * from category;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
12 | Concerts | Comedy  | All stand up comedy performances
(1 row)
```

Die folgende gleichwertige Abfrage verwendet das Synonym MINUS.

```
select * from category_stage
minus
select * from category;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
12 | Concerts | Comedy  | All stand up comedy performances
(1 row)
```

Wenn Sie die Reihenfolge der SELECT-Ausdrücke umkehren, gibt die Abfrage keine Zeilen zurück.

ORDER BY-Klausel

Themen

Die ORDER BY-Klausel sortiert den Ergebnissatz einer Abfrage.

Syntax

```
[ ORDER BY expression [ ASC | DESC ] ]
[ NULLS FIRST | NULLS LAST ]
[ LIMIT { count | ALL } ]
[ OFFSET start ]
```

Parameter

expression

Ausdruck, der die Sortierreihenfolge des Abfrageergebnissatzes definiert, in der Regel durch die Angabe mindestens einer Spalte in der Auswahlliste. Die Ergebnisse werden auf der Basis einer binären UTF-8-Reihenfolge zurückgegeben. Sie können auch Folgendes angeben:

- Spalten, die nicht in der Auswahlliste vorhanden sind
- Ausdrücke, die aus einer oder mehreren Spalten gebildet werden, die in den Tabellen vorhanden sind, die von der Abfrage referenziert werden
- Ordinalzahlen, die die Position der Auswahllisteneinträge darstellen (oder die Position der Spalten in der Tabelle, wenn keine Auswahlliste vorhanden ist)
- Aliase, die Auswahllisteneinträge definieren

Wenn die -Klausel mehrere Ausdrücke enthält, wird der Ergebnissatz nach dem ersten Ausdruck sortiert. Anschließend wird der zweite Ausdruck auf Zeilen mit übereinstimmenden Werten aus dem ersten Ausdruck angewendet usw.

ASC | DESC

Eine Option, die die Sortierreihenfolge für den Ausdruck wie folgt definiert:

- ASC: aufsteigend (beispielsweise niedrig nach hoch für numerische Werte und A bis Z für Zeichenfolgen). Wenn keine Option angegeben wird, werden die Daten standardmäßig in aufsteigender Reihenfolge sortiert.
- DESC: absteigend (beispielsweise hoch nach niedrig für numerische Werte und Z bis A für Zeichenfolgen).

NULLS FIRST | NULLS LAST

Option, die angibt, ob NULL-Werte vor Nicht-Null-Werten oder nach Nicht-Null-Werten aufgelistet werden sollen. Standardmäßig werden NULL-Werte in einer ASC-Reihenfolge an letzter Stelle sortiert und aufgeführt und in einer DESC-Reihenfolge an erster Stelle sortiert und aufgeführt.

LIMIT number | ALL

Option, die die Anzahl der sortierten Zeilen steuert, die von der Abfrage zurückgegeben werden. Bei der LIMIT-Zahl muss es sich um eine positive Ganzzahl handeln. Der maximal zulässige Wert ist `2147483647`.

LIMIT 0 gibt keine Zeilen zurück. Sie können diese Syntax für Testzwecke verwenden: um zu prüfen, ob eine Abfrage ausgeführt wird (ohne Zeilen anzuzeigen) oder um eine Spaltenliste aus einer Tabelle zurückzugeben. Eine -Klausel ist redundant, wenn Sie LIMIT 0 verwenden, um eine Spaltenliste zurückzugeben. Der Standardwert ist LIMIT ALL.

OFFSET start

Option, die die Anzahl der Zeilen vor start angibt, die übersprungen werden sollen, bevor Zeilen zurückgegeben werden. Bei der OFFSET-Zahl muss es sich um eine positive Ganzzahl handeln. Der maximal zulässige Wert ist `2147483647`. Bei der Verwendung mit der Option LIMIT werden OFFSET-Zeilen übersprungen, bevor die Zahl der LIMIT-Zeilen gezählt werden, die zurückgegeben werden. Wenn die LIMIT-Option nicht verwendet wird, wird die Zahl der Zeilen im Ergebnissatz um die Zahl der übersprungenen Zeilen reduziert. Die von einer OFFSET-Klausel übersprungenen Zeilen müssen dennoch gescannt werden. Daher ist es möglicherweise ineffizient, einen großen OFFSET-Wert zu verwenden.

Nutzungshinweise

Beachten Sie das folgende erwartete Verhalten bei Verwendung von ORDER BY-Klauseln:

- NULL-Werte gelten als „höher“ als alle anderen Werte. Bei Verwendung der standardmäßigen aufsteigenden Sortierfolge befinden sich NULL-Werte am Ende. Um dieses Verhalten zu ändern, wählen Sie die Option NULLS FIRST.
- Wenn eine Anfrage keine ORDER BY-Klausel enthält, gibt das System Ergebnissätze ohne vorhersagbare Anordnung der Zeilen zurück. Wenn dieselbe Abfrage zweimal ausgeführt wird, wird der Ergebnissatz möglicherweise in einer anderen Reihenfolge zurückgegeben.
- Die Optionen LIMIT und OFFSET können ohne ORDER BY-Klausel verwendet werden. Um jedoch einen konsistenten Satz von Zeilen zurückzugeben, verwenden Sie diese Optionen in Verbindung mit ORDER BY.
- In einem parallelen System wie Amazon Redshift, in dem ORDER BY keine spezifische Anordnung generiert, ist die Reihenfolge der Zeilen nicht deterministisch. Wenn der ORDER BY-Ausdruck duplizierte Werte produziert, kann sich die Rückgabereihenfolge dieser Zeilen von der anderer Systeme oder zwischen Ausführungen in Amazon Redshift unterscheiden.
- Amazon Redshift unterstützt keine Zeichenfolgeliterale in ORDER-BY-Klauseln.

Beispiele mit ORDER BY

Gibt alle 11 Zeilen aus der Tabelle CATEGORY geordnet nach der zweiten Spalte, CATGROUP, zurück. Ergebnisse, die denselben CATGROUP-Wert haben, ordnen die CATDESC-Spaltenwerte nach der Länge der Zeichenfolge. Dann wird nach Spalten CATID und CATNAME geordnet.

```
select * from category order by 2, length(catdesc), 1, 3;

catid | catgroup |  catname  |                 catdesc
------+----------+-----------+--------------------------------------
10    | Concerts | Jazz      | All jazz singers and bands
9     | Concerts | Pop       | All rock and pop music concerts
11    | Concerts | Classical | All symphony, concerto, and choir conce
6     | Shows    | Musicals  | Musical theatre
7     | Shows    | Plays     | All non-musical theatre
8     | Shows    | Opera     | All opera and light opera
5     | Sports   | MLS       | Major League Soccer
1     | Sports   | MLB       | Major League Baseball
2     | Sports   | NHL       | National Hockey League
3     | Sports   | NFL       | National Football League
4     | Sports   | NBA       | National Basketball Association
(11 rows)
```

Gibt ausgewählte Spalten aus der Tabelle SALES zurück, geordnet nach den höchsten QTYSOLD-Werten. Begrenzt das Ergebnis auf die obersten 10 Zeilen:

```
select salesid, qtysold, pricepaid, commission, saletime from sales
order by qtysold, pricepaid, commission, salesid, saletime desc
limit 10;

salesid | qtysold | pricepaid | commission |      saletime
--------+---------+-----------+------------+---------------------
15401   |       8 |    272.00 |      40.80 | 2008-03-18 06:54:56
61683   |       8 |    296.00 |      44.40 | 2008-11-26 04:00:23
90528   |       8 |    328.00 |      49.20 | 2008-06-11 02:38:09
74549   |       8 |    336.00 |      50.40 | 2008-01-19 12:01:21
130232  |       8 |    352.00 |      52.80 | 2008-05-02 05:52:31
55243   |       8 |    384.00 |      57.60 | 2008-07-12 02:19:53
16004   |       8 |    440.00 |      66.00 | 2008-11-04 07:22:31
489     |       8 |    496.00 |      74.40 | 2008-08-03 05:48:55
4197    |       8 |    512.00 |      76.80 | 2008-03-23 11:35:33
16929   |       8 |    568.00 |      85.20 | 2008-12-19 02:59:33
(10 rows)
```

Gibt unter Verwendung der LIMIT 0-Syntax eine Spaltenliste, aber keine Zeilen zurück:

```
select * from venue limit 0;
venueid | venuename | venuecity | venuestate | venueseats
---------+-----------+-----------+------------+------------
(0 rows)
```

CONNECT BY-Klausel

Die CONNECT BY-Klausel gibt die Beziehung zwischen Zeilen in einer Hierarchie an. Sie können CONNECT BY verwenden, um Zeilen in einer hierarchischen Reihenfolge auszuwählen, indem Sie die Tabelle mit sich selbst verbinden und die hierarchischen Daten verarbeiten. Sie können dies beispielsweise verwenden, um rekursiv ein Organigramm zu durchlaufen und Daten aufzulisten.

Hierarchische Abfragen werden in der folgenden Reihenfolge verarbeitet:

1. Wenn die FROM-Klausel eine Verknüpfung enthält, wird diese zuerst verarbeitet.
2. Die CONNECT BY-Klausel wird ausgewertet.
3. Die WHERE-Klausel wird ausgewertet.

Syntax

```
[START WITH start_with_conditions]
CONNECT BY connect_by_conditions
```

> **Note**
>
> START und CONNECT sind zwar keine reservierten Wörter, Sie sollten jedoch abgegrenzte Kennungen (doppelte Anführungszeichen) oder AS verwenden, wenn Sie START und CONNECT als Tabellenaliasse in Ihrer Abfrage verwenden, um Fehler während der Laufzeit zu vermeiden.

```
SELECT COUNT(*)
FROM Employee "start"
CONNECT BY PRIOR id = manager_id
START WITH name = 'John'
```

```
SELECT COUNT(*)
FROM Employee AS start
CONNECT BY PRIOR id = manager_id
START WITH name = 'John'
```

Parameter

start_with_conditions

Bedingungen, die die Stammzeile(n) der Hierarchie angeben.

connect_by_conditions

Bedingungen, die die Beziehung zwischen übergeordneten Zeilen und untergeordneten Zeilen der Hierarchie angeben. Mindestens eine Bedingung muss mit dem unären Operator qualifiziert werden, der für den Verweis auf die übergeordnete Zeile verwendet wird.

```
PRIOR column = expression
-- or
expression > PRIOR column
```

Operatoren

Sie können die folgenden Operatoren in einer CONNECT BY-Abfrage verwenden.

LEVEL

Pseudospalte, die die aktuelle Zeilenebene in der Hierarchie zurückgibt. Gibt 1 für die Stammzeile, 2 für das untergeordnete Element der Stammzeile zurück usw.

PRIOR

Unärer Operator, der den Ausdruck für die übergeordnete Zeile der aktuellen Zeile in der Hierarchie auswertet.

Beispiele

Im Folgenden finden Sie eine CONNECT BY-Abfrage, die die Anzahl der Mitarbeiter zurückgibt, die direkt oder indirekt John unterstellt sind, und zwar auf 4 Ebenen begrenzt.

```
SELECT id, name, manager_id
FROM employee
WHERE LEVEL < 4
START WITH name = 'John'
CONNECT BY PRIOR id = manager_id;
```

Nachfolgend das Ergebnis der Abfrage.

```
id      name      manager_id
------+----------+--------------
  101     John        100
  102     Jorge       101
  103     Kwaku       101
  110     Liu         101
  201     Sofía       102
  106     Mateo       102
  110     Nikki       103
  104     Paulo       103
  105     Richard     103
  120     Saanvi      104
  200     Shirley     104
  205     Zhang       104
```

Die Tabellendefinition für dieses Beispiel lautet wie folgt:

```
CREATE TABLE employee (
   id INT,
   name VARCHAR(20),
   manager_id INT
   );
```

Im Folgenden werden die Zeilen in die Tabelle eingefügt.

```
INSERT INTO employee(id, name, manager_id)  VALUES
(100, 'Carlos', null),
(101, 'John', 100),
(102, 'Jorge', 101),
(103, 'Kwaku', 101),
(110, 'Liu', 101),
(106, 'Mateo', 102),
(110, 'Nikki', 103),
(104, 'Paulo', 103),
(105, 'Richard', 103),
(120, 'Saanvi', 104),
(200, 'Shirley', 104),
(201, 'Sofía', 102),
(205, 'Zhang', 104);
```

Im Folgenden finden Sie ein Organigramm für Johns Abteilung.

Beispiele für Unterabfragen

In den folgenden Beispielen zeigen verschiedene Möglichkeiten, wie Unterabfragen in SELECT-Abfragen integriert werden können. Ein weiteres Beispiel für die Verwendung von Unterabfragen finden Sie unter JOIN-Beispiele.

Unterabfragen in der SELECT-Liste

Das folgende Beispiel enthält eine Unterabfrage in der SELECT-Liste. Diese Unterabfrage ist skalar: Sie gibt nur eine Spalte und einen Wert zurück. Dies wird im Ergebnis für jede Zeile wiederholt, die von der umschließenden Abfrage zurückgegeben wird. Die Abfrage vergleicht den von der Unterabfrage berechneten Q1SALES-Wert mit den Verkaufswerten für zwei andere Quartale (2 und 3) im Jahr 2008 wie von der umschließenden Abfrage definiert.

```
select qtr, sum(pricepaid) as qtrsales,
(select sum(pricepaid)
from sales join date on sales.dateid=date.dateid
where qtr='1' and year=2008) as q1sales
from sales join date on sales.dateid=date.dateid
where qtr in('2','3') and year=2008
group by qtr
order by qtr;

qtr  |  qtrsales   |   q1sales
-------+-------------+-------------
2     | 30560050.00 | 24742065.00
3     | 31170237.00 | 24742065.00
(2 rows)
```

Unterabfragen in der WHERE-Klausel

Das folgende Beispiel enthält eine Tabellenunterabfrage in der WHERE-Klausel. Diese Unterabfrage produziert mehrere Zeilen. In diesem Fall enthalten die Zeilen nur eine Spalte. Tabellenunterabfragen können jedoch mehrere Spalten und Zeilen enthalten, genau wie jede andere Tabelle.

Die Abfrage sucht die 10 Top-Verkäufer in Bezug die meisten verkauften Tickets. Die Liste der Top 10 wird durch die Unterabfrage eingeschränkt, die Benutzer entfernt, die in Städten mit Ticketverkaufsstellen leben. Diese Abfrage kann auf verschiedene Arten geschrieben werden. Beispielsweise könnte die Unterabfrage als ein Join innerhalb der Hauptabfrage geschrieben werden.

```
select firstname, lastname, city, max(qtysold) as maxsold
from users join sales on users.userid=sales.sellerid
where users.city not in(select venuecity from venue)
group by firstname, lastname, city
order by maxsold desc, city desc
limit 10;

firstname | lastname  |      city      | maxsold
-----------+-----------+----------------+---------
Noah       | Guerrero | Worcester      |       8
Isadora    | Moss     | Winooski       |       8
Kieran     | Harrison | Westminster    |       8
Heidi      | Davis    | Warwick        |       8
Sara       | Anthony  | Waco           |       8
Bree       | Buck     | Valdez         |       8
Evangeline | Sampson  | Trenton        |       8
```

```
Kendall    | Keith    | Stillwater     |       8
Bertha     | Bishop   | Stevens Point  |       8
Patricia   | Anderson | South Portland |       8
(10 rows)
```

Unterabfragen in der WITH-Klausel

Siehe WITH-Klausel.

Korrelierte Unterabfragen

Das folgende Beispiel enthält eine korrelierte Unterabfrage in der WHERE-Klausel. Diese Art von Unterabfrage enthält mindestens eine Korrelation zwischen ihren Spalten und den Spalten, die von der umschließenden Abfrage produziert werden. In diesem Fall ist die Korrelation `where s.listid=l.listid`. Die Unterabfrage wird für jede Zeile ausgeführt, die die umschließende Abfrage produziert, um die Zeile zu qualifizieren oder zu disqualifizieren.

```
select salesid, listid, sum(pricepaid) from sales s
where qtysold=
(select max(numtickets) from listing l
where s.listid=l.listid)
group by 1,2
order by 1,2
limit 5;

salesid | listid |   sum
--------+--------+----------
 27     |     28 | 111.00
 81     |    103 | 181.00
 142    |    149 | 240.00
 146    |    152 | 231.00
 194    |    210 | 144.00
(5 rows)
```

Muster für korrelierte Unterabfragen, die nicht unterstützt werden

Der Abfrageplaner verwendet eine Methode für das Neuschreiben von Abfragen, die als Entkorrelierung von Unterabfragen bezeichnet wird, um verschiedene Muster korrelierter Unterabfragen für die Ausführung in einer MPP-Umgebung zu optimieren. Einige Arten von korrelierten Unterabfragen folgen Mustern, die Amazon Redshift nicht entkorrelieren kann und nicht unterstützt. Abfragen, die die folgenden Korrelierungsreferenzen enthalten, geben Fehler zurück:

- Korrelierungsreferenzen, die einen Abfrageblock überspringen, auch als „überspringende Korrelierungsreferenzen“ bekannt. Beispielsweise sind in der folgenden Abfrage der Block mit der Korrelierungsreferenz und der übersprungene Block durch ein NOT EXISTS-Prädikat verbunden:

```
select event.eventname from event
where not exists
(select * from listing
where not exists
(select * from sales where event.eventid=sales.eventid));
```

  Der übersprungende Block ist in diesem Fall die Unterabfrage für die LISTING-Tabelle. Die Korrelierungsreferenz korreliert die Tabellen EVENT und SALES.

- Korrelierungsreferenzen aus einer Unterabfrage, die Teil einer ON-Klausel in einer externen Abfrage ist:

```
select * from category
left join event
on category.catid=event.catid and eventid =
(select max(eventid) from sales where sales.eventid=event.eventid);
```

  Die ON-Klausel enthält eine Korrelierungsreferenz aus SALES in der Unterabfrage für EVENT in der umschließenden Abfrage.

- Null-sensible Korrelierungsreferenzen für eine Amazon-Redshift-Systemtabelle. Beispiel:

```
select attrelid
from stv_locks sl, pg_attribute
where sl.table_id=pg_attribute.attrelid and 1 not in
(select 1 from pg_opclass where sl.lock_owner = opcowner);
```

- Korrelierungsreferenzen aus einer Unterabfrage, die eine Fensterfunktion enthält.

```
select listid, qtysold
from sales s
where qtysold not in
(select sum(numtickets) over() from listing l where s.listid=l.listid);
```

- Referenzen in einer GROUP BY-Spalte zu den Ergebnissen einer korrelierten Unterabfrage. Beispiel:

```
select listing.listid,
(select count (sales.listid) from sales where sales.listid=listing.listid) as list
from listing
group by list, listing.listid;
```

- Korrelierungsreferenzen aus einer Unterabfrage mit einer Aggregationsfunktion und einer GROUP BY-Klausel, die durch ein IN-Prädikat mit der umschließenden Abfrage verbunden sind. (Diese Einschränkung gilt nicht für die Aggregationsfunktionen MIN und MAX.) Beispielsweise:

```
select * from listing where listid in
(select sum(qtysold)
from sales
where numtickets>4
group by salesid);
```

SELECT INTO

Wählt Zeilen aus, die durch eine beliebige Abfrage definiert sind, und fügt sie in eine neue Tabelle ein. Sie können angeben, ob eine temporäre oder eine persistente Tabelle erstellt werden soll.

Syntax

```
[ WITH with_subquery [, ...] ]
SELECT
[ TOP number ] [ ALL | DISTINCT ]
* | expression [ AS output_name ] [, ...]
INTO [ TEMPORARY | TEMP ] [ TABLE ] new_table
[ FROM table_reference [, ...] ]
[ WHERE condition ]
[ GROUP BY expression [, ...] ]
[ HAVING condition [, ...] ]
[ { UNION | INTERSECT | { EXCEPT | MINUS } } [ ALL ] query ]
[ ORDER BY expression
[ ASC | DESC ]
[ LIMIT { number | ALL } ]
[ OFFSET start ]
```

Details zu den Parametern dieses Befehls finden Sie unter SELECT.

Beispiele

Wählen Sie alle Zeilen aus der Tabelle EVENT aus und erstellen Sie die Tabelle NEWEVENT:

```
select * into newevent from event;
```

Fügen Sie das Ergebnis einer Aggregatabfrage in eine temporäre Tabelle namens PROFITS ein:

```
select username, lastname, sum(pricepaid-commission) as profit
into temp table profits
from sales, users
where sales.sellerid=users.userid
group by 1, 2
order by 3 desc;
```

SET

Legt den Wert eines Serverkonfigurationsparameters fest. Mit dem SET-Befehl setzen Sie eine Einstellung nur für die Dauer der aktuellen Sitzung oder Transaktion außer Kraft.

Mit dem RESET-Befehl setzen Sie einen Parameter auf dessen Standardwert zurück.

Sie können die Serverkonfigurationsparameter auf verschiedene Arten ändern. Weitere Informationen finden Sie unter Modifizieren der Serverkonfiguration.

Syntax

```
SET { [ SESSION | LOCAL ]
{ SEED | parameter_name } { TO | = }
{ value | 'value' | DEFAULT } |
SEED TO value }
```

Die folgende Anweisung legt den Wert einer Sitzungskontextvariablen fest.

```
SET { [ SESSION | LOCAL ]
variable_name { TO | = }
{ value | 'value'  }
```

Parameter

SESSION

Gibt an, dass die Einstellung für die aktuelle Sitzung gültig ist. Standardwert.

variable_name

Gibt den Namen der Kontextvariablen an, die für die Sitzung festgelegt wurde.

Die Benennungskonvention ist ein zweiteiliger Name, der durch einen Punkt getrennt ist, z.B. identifier.identifier. Es ist nur ein Punkttrennzeichen zulässig. Verwenden Sie eine Kennung, die den Standardkennungsregeln für Amazon Redshift folgt. Weitere Informationen finden Sie unter Namen und Kennungen. Begrenzte Kennungen sind nicht zulässig.

LOCAL

Gibt an, dass die Einstellung für die aktuelle Transaktion gültig ist.

SEED TO value

Legt einen internen Seed fest, der von der Funktion RANDOM für die Zufallsgenerierung einer Zahl verwendet werden soll.

SET SEED verwendet einen numerischen Wert zwischen 0 und 1 und multipliziert diese Zahl mit (2^{31}-1) zur Verwendung mit der Funktion Die Funktion RANDOM. Wenn Sie SET SEED verwenden, bevor Sie mehrere RANDOM-Aufrufe ausführen, generiert RANDOM Zahlen in einer vorhersagbaren Sequenz.

parameter_name

Der Name des Parameters, der festgelegt werden soll. Weitere Informationen über Parameter finden Sie unter Modifizieren der Serverkonfiguration.

Wert

Neuer Parameterwert. Verwenden Sie einfache Anführungszeichen, um den Wert für eine bestimmte Zeichenfolge festzulegen. Wenn Sie SET SEED verwenden, enthält dieser Parameter den SEED-Wert.

DEFAULT

Legt den Parameter auf den Standardwert fest.

Beispiele

Ändern eines Parameters für die aktuelle Sitzung

Im folgenden Beispiel wird der Datenstil festgelegt:

```
set datestyle to 'SQL,DMY';
```

Einrichten einer Abfragegruppe für das Workload-Management

Wenn Abfragegruppen in einer Warteschlangendefinition als Teil der WLM-Konfiguration des Clusters aufgeführt werden, können Sie den Parameter QUERY_GROUP auf einen aufgeführten Abfragegruppennamen festlegen. Nachfolgende Abfragen werden der verknüpften Abfragewarteschlange zugewiesen. Die Einstellung QUERY_GROUP bleibt für die Dauer der Sitzung gültig oder bis sie auf den Befehl RESET QUERY_GROUP trifft.

In diesem Beispiel werden zwei Abfragen als Teil der Abfragegruppe „Priority“ ausgeführt. Anschließend wird die Abfragegruppe zurückgesetzt.

```
set query_group to 'priority';
select tbl, count(*)from stv_blocklist;
select query, elapsed, substring from svl_qlog order by query desc limit 5;
reset query_group;
```

Weitere Informationen finden Sie unter Implementierung von Workload Management.

Ändern Sie den Standard-Identitätsnamespace für die Sitzung

Ein Datenbankbenutzer kann festlegen`default_identity_namespace`. Dieses Beispiel zeigt, wie Sie `SET SESSION` die Einstellung für die Dauer der aktuellen Sitzung überschreiben und dann den neuen Identitätsanbieter-Wert anzeigen können. Dies wird am häufigsten verwendet, wenn Sie einen Identitätsanbieter mit Redshift und IAM Identity Center verwenden. Weitere Informationen zur Verwendung eines Identitätsanbieters mit Redshift finden Sie unter Connect von Redshift mit IAM Identity Center, um Benutzern eine Single-Sign-On-Erfahrung zu bieten.

```
SET SESSION default_identity_namespace = 'MYCO';

SHOW default_identity_namespace;
```

Nachdem Sie den Befehl ausgeführt haben, können Sie eine GRANT-Anweisung oder eine CREATE-Anweisung wie die folgende ausführen:

```
GRANT SELECT ON TABLE mytable TO alice;

GRANT UPDATE ON TABLE mytable TO salesrole;

CREATE USER bob password 'md50c983d1a624280812631c5389e60d48c';
```

In diesem Fall entspricht die Einstellung des standardmäßigen Identitätsnamespaces dem Präfix des Namespaces für jede Identität. In diesem Beispiel `alice` wird ersetzt durch. `MYCO:alice` Weitere Informationen zu Einstellungen, die sich auf die Redshift-Konfiguration mit IAM Identity Center beziehen, finden Sie unter und. ALTER SYSTEM ALTER IDENTITY PROVIDER

Festlegen einer Bezeichnung für eine Gruppe von Abfragen

Der Parameter QUERY_GROUP definiert eine Bezeichnung für mindestens eine Abfrage, die in derselben Sitzung nach einem SET-Befehl ausgeführt wird. Diese Bezeichnung wird protokolliert, wenn Abfragen ausgeführt werden, und kann zur Einschränkung der Ergebnisse verwendet werden, die von den Systemtabellen STL_QUERY und STV_INFLIGHT sowie der Ansicht SVL_QLOG zurückgegeben werden.

```
show query_group;
query_group
-------------
unset
(1 row)

set query_group to '6 p.m.';


show query_group;
query_group
-------------
6 p.m.
(1 row)

select * from sales where salesid=500;
salesid | listid | sellerid | buyerid | eventid | dateid | ...
---------+--------+----------+---------+---------+--------+-----
500 |    504 |     3858 |    2123 |    5871 |   2052 | ...
(1 row)

reset query_group;
```

```
select query, trim(label) querygroup, pid, trim(querytxt) sql
from stl_query
where label ='6 p.m.';
query | querygroup |  pid  |                  sql
-------+------------+-------+---------------------------------------
57 | 6 p.m.     | 30711 | select * from sales where salesid=500;
(1 row)
```

Bezeichnungen von Abfragegruppen sind ein nützlicher Mechanismus, um einzelne Abfragen oder Gruppen von Abfragen zu isolieren, die als Teil von Skripts ausgeführt werden. Sie müssen Abfragen nicht nach ID identifizieren und nachverfolgen. Sie können sie anhand ihrer Bezeichnungen nachverfolgen.

Festlegen eines Seed-Wert für die Zufallsgenerierung von Zahlen

Im folgenden Beispiel wird die SEED-Option mit SET verwendet, damit die Funktion RANDOM Zahlen in einer vorhersagbaren Sequenz generiert.

Geben Sie zunächst drei RANDOM-Ganzzahlen zurück, ohne zuerst den SEED-Wert festzulegen:

```
select cast (random() * 100 as int);
int4
------
6
(1 row)

select cast (random() * 100 as int);
int4
------
68
(1 row)

select cast (random() * 100 as int);
int4
------
56
(1 row)
```

Legen Sie nun den SEED-Wert auf `.25` fest und geben Sie drei weitere RANDOM-Zahlen zurück:

```
set seed to .25;

select cast (random() * 100 as int);
```

```
int4
------
21
(1 row)

select cast (random() * 100 as int);
int4
------
79
(1 row)

select cast (random() * 100 as int);
int4
------
12
(1 row)
```

Setzen Sie zum Schluss den SEED-Wert auf `.25` zurück und überprüfen Sie, ob RANDOM dieselben Ergebnisse wie in den vorherigen drei Aufrufen zurückgibt:

```
set seed to .25;

select cast (random() * 100 as int);
int4
------
21
(1 row)

select cast (random() * 100 as int);
int4
------
79
(1 row)

select cast (random() * 100 as int);
int4
------
12
(1 row)
```

Im folgenden Beispiel wird eine benutzerdefinierte Kontextvariable festgelegt.

```
SET app_context.user_id TO 123;
```

```
SET app_context.user_id TO 'sample_variable_value';
```

SET SESSION AUTHORIZATION

Legt den Benutzernamen für die aktuelle Sitzung fest.

Sie können beispielsweise den Befehl SET SESSION AUTHORIZATION verwenden, um den Datenbankzugriff zu testen, indem Sie eine Sitzung oder Transaktion vorübergehend als nicht berechtigter Benutzer ausführen. Sie müssen Datenbank-Superuser sein, um diesen Befehl auszuführen.

Syntax

```
SET [ LOCAL ] SESSION AUTHORIZATION { user_name | DEFAULT }
```

Parameter

LOCAL

Gibt an, dass die Einstellung für die aktuelle Transaktion gültig ist. Durch Weglassen dieses Parameters wird angegeben, dass die Einstellung für die aktuelle Sitzung gültig ist.

user_name

Der Name des Benutzers, der festgelegt werden soll. Der Benutzernamen kann als Bezeichner oder Zeichenfolgeliteral geschrieben werden.

DEFAULT

Legt den Benutzernamen für die Sitzung auf den Standardwert fest.

Beispiele

Im folgenden Beispiel wird der Benutzername für die aktuelle Sitzung auf festgelegt `dwuser`:

```
SET SESSION AUTHORIZATION 'dwuser';
```

Im folgenden Beispiel wird der Benutzername für die aktuelle Transaktion auf festgelegt `dwuser`:

```
SET LOCAL SESSION AUTHORIZATION 'dwuser';
```

In diesem Beispiel wird der Benutzername für die aktuelle Sitzung auf den Standardbenutzernamen festgelegt:

```
SET SESSION AUTHORIZATION DEFAULT;
```

SET SESSION CHARACTERISTICS

Dieser Befehl ist veraltet.

ZEIGEN

Zeigt den aktuellen eines Serverkonfigurationsparameters an. Dieser Wert ist möglicherweise spezifisch für die aktuelle Sitzung, wenn ein SET-Befehl in Kraft ist. Eine Liste der Konfigurationsparameter finden Sie unter Konfigurationsreferenz.

Syntax

```
SHOW { parameter_name | ALL }
```

Die folgende Anweisung zeigt den aktuellen Wert einer Sitzungskontextvariablen an. Wenn die Variable nicht existiert, löst Amazon Redshift einen Fehler aus.

```
SHOW variable_name
```

Parameter

parameter_name

Zeigt den aktuellen Wert des angegebenen Parameters an.

ALL

Zeigt die aktuellen Werte aller Parameter an.

variable_name

Zeigt den aktuellen Wert der angegebenen Variablen an.

Beispiele

Im folgenden Beispiel wird der Wert für den Parameter query_group angezeigt:

```
show query_group;

query_group

unset
(1 row)
```

Im folgenden Beispiel wird eine Liste aller Parameter und ihrer Werte angezeigt:

```
show all;
name        |   setting
--------------------+--------------
datestyle          | ISO, MDY
extra_float_digits | 0
query_group        | unset
search_path        | $user,public
statement_timeout  | 0
```

Im folgenden Beispiel wird der aktuelle Wert der angegebenen Variablen angezeigt.

```
SHOW app_context.user_id;
```

SHOW_COLUMNS

Zeigt eine Liste der Spalten in einer Tabelle zusammen mit einigen Spaltenattributen an.

Jede Ausgabezeile besteht aus einer durch Kommas getrennten Liste mit Datenbanknamen, Schemanamen, Tabellennamen, Spaltennamen, Ordinalposition, Spaltenstandard, Nullwert, Datentyp, maximaler Zeichenlänge, numerischer Genauigkeit und Anmerkungen. Weitere Informationen zu diesen Attributen finden Sie unter SVV_ALL_COLUMNS.

Wenn der Befehl SHOW COLUMNS mehr als 10 000 Spalten ergeben würde, wird ein Fehler zurückgegeben.

Syntax

```
SHOW COLUMNS FROM TABLE database_name.schema_name.table_name [LIKE 'filter_pattern']
 [LIMIT row_limit ]
```

Parameter

database_name

Der Name der Datenbanktabelle, welche die aufzulistenden Tabellen enthält.

Um Tabellen in einer anzuzeigen AWS Glue Data Catalog, geben Sie (`awsdatacatalog`) als Datenbanknamen an und stellen Sie sicher, dass die Systemkonfiguration auf eingestellt `data_catalog_auto_mount` ist. `true` Weitere Informationen finden Sie unter ALTER SYSTEM.

schema_name

Der Name des Schemas, das die aufzulistenden Tabellen enthält.

Um AWS Glue Data Catalog Tabellen anzuzeigen, geben Sie den AWS Glue Datenbanknamen als Schemanamen an.

table_name

Der Name der Tabelle, welche die aufzulistenden Spalten enthält.

filter_pattern

Ein gültiger UTF-8-Zeichenfolgenausdruck mit einem Muster zum Abgleich der Tabellennamen. Die Option LIKE führt eine Suche durch, bei der zwischen Groß- und Kleinschreibung unterschieden wird und welche die folgenden Metazeichen für den Mustervergleich unterstützt:

Metazeichen	Beschreibung
%	Entspricht einer Folge von 0 oder mehr Zeichen.
_	Entspricht einem beliebigen Zeichen.

Wenn filter-pattern keine Metazeichen enthält, repräsentiert das Muster die Zeichenfolge selbst. In diesem Fall liefert LIKE dasselbe Ergebnis wie der Gleichheitsoperator.

row_limit

Die maximale Anzahl der zurückzugebenden Zeilen. Der Wert row_limit kann 0–10 000 betragen.

Beispiele

Das folgende Beispiel zeigt die Spalten in der Amazon-Redshift-Datenbank mit dem Namen `dev`, die sich im Schema `public` und in der Tabelle `tb` befinden.

```
SHOW COLUMNS FROM TABLE dev.public.tb;

 database_name | schema_name | table_name | column_name | ordinal_position
 | column_default | is_nullable | data_type | character_maximum_length |
 numeric_precision | remarks
---------------+-------------+-----------+-------------+-----------------
+----------------+-------------+-----------+-------------------------
+-------------------+---------
 dev           | public      | tb         | col         |                1 |
     | YES         | integer   |                          |                32 |
```

Das folgende Beispiel zeigt die Tabellen in der genannten AWS Glue Data Catalog Datenbank`awsdatacatalog`, die sich in Schema `batman` und Tabelle befinden`nation`. Die Ausgabe ist auf 2 Zeilen begrenzt.

```
SHOW COLUMNS FROM TABLE awsdatacatalog.batman.nation LIMIT 2;

 database_name  | schema_name | table_name | column_name | ordinal_position
 | column_default | is_nullable | data_type | character_maximum_length |
 numeric_precision | remarks
----------------+-------------+-----------+-------------+-----------------
+----------------+-------------+-----------+-------------------------
+-------------------+---------
 awsdatacatalog | batman      | nation     | n_nationkey |                1 |
      |             | integer   |                          |                  |
 awsdatacatalog | batman      | nation     | n_name      |                2 |
      |             | character |                          |                  |
```

SHOW EXTERNAL TABLE

Zeigt die Definition einer externen Tabelle an, einschließlich der Tabellenattribute und Spaltenattribute. Sie können die Ausgabe der SHOW EXTERNAL TABLE-Anweisung verwenden, um die Tabelle neu zu erstellen.

Weitere Informationen zur Erstellung einer externen Tabelle finden Sie unter CREATE EXTERNAL TABLE.

Syntax

```
SHOW EXTERNAL TABLE [external_database].external_schema.table_name [ PARTITION ]
```

Parameter

external_database

Der Name der verbundenen externen Datenbank. Dieser Parameter ist optional.

external_schema

Der Name des verbundenen externen Schemas.

table_name

Der Name der Tabelle, die angezeigt werden soll.

PARTITION

Zeigt ALTER TABLE-Anweisungen zum Hinzufügen von Partitionen zur Tabellendefinition an.

Beispiele

Die folgenden Beispiele basieren auf einer externen Tabelle, die wie folgt definiert ist:

```
CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
     csmallint smallint,
     cint int,
     cbigint bigint,
     cfloat float4,
     cdouble float8,
     cchar char(10),
     cvarchar varchar(255),
     cdecimal_small decimal(18,9),
     cdecimal_big decimal(30,15),
     ctimestamp TIMESTAMP,
     cboolean boolean,
     cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime TIMESTAMP)
STORED AS PARQUET
LOCATION 's3://mybucket-test-copy/alldatatypes_parquet_partitioned';
```

Es folgt ein Beispiel für den Befehl SHOW EXTERNAL TABLE und die Ausgabe für die Tabelle `my_schema.alldatatypes_parquet_test_partitioned`.

```
SHOW EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned;
```

```
"CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime timestamp)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://mybucket-test-copy/alldatatypes_parquet_partitioned';"
```

Im Folgenden finden Sie ein Beispiel für den Befehl SHOW EXTERNAL TABLE und die Ausgabe für dieselbe Tabelle, wobei jedoch auch die Datenbank im Parameter angegeben ist.

```
SHOW EXTERNAL TABLE my_database.my_schema.alldatatypes_parquet_test_partitioned;
```

```
"CREATE EXTERNAL TABLE my_database.my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
```

```
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime timestamp)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://mybucket-test-copy/alldatatypes_parquet_partitioned';"
```

Es folgt ein Beispiel für den Befehl SHOW EXTERNAL TABLE und die Ausgabe bei Verwendung des Parameters PARTITION. Die Ausgabe enthält ALTER TABLE-Anweisungen zum Hinzufügen von Partitionen zur Tabellendefinition.

```
SHOW EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned PARTITION;
```

```
"CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://mybucket-test-copy/alldatatypes_parquet_partitioned';
ALTER TABLE my_schema.alldatatypes_parquet_test_partitioned ADD IF NOT
 EXISTS PARTITION (cdate='2021-01-01') LOCATION 's3://mybucket-test-copy/
alldatatypes_parquet_partitioned2/cdate=2021-01-01';
ALTER TABLE my_schema.alldatatypes_parquet_test_partitioned ADD IF NOT
 EXISTS PARTITION (cdate='2021-01-02') LOCATION 's3://mybucket-test-copy/
alldatatypes_parquet_partitioned2/cdate=2021-01-02';"
```

SHOW DATABASES

Zeigt Datenbanken von einer bestimmten Konto-ID an.

Syntax

```
SHOW DATABASES FROM
DATA CATALOG [ ACCOUNT  '<id1>', '<id2>', ... ]
[ LIKE '<expression>' ]
[ IAM_ROLE default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' ]
```

Parameter

ACCOUNT '<id1>', '<id2>', ...

Die AWS Glue Data Catalog Konten, aus denen Datenbanken aufgelistet werden sollen. Wenn Sie diesen Parameter weglassen, bedeutet das, dass Amazon Redshift die Datenbanken des Kontos anzeigen soll, dem der Cluster gehört.

LIKE '<Ausdruck>'

Filtert die Liste der Datenbanken nach denen, die dem von Ihnen angegebenen Ausdruck entsprechen. Dieser Parameter unterstützt Muster, die die Platzhalterzeichen % (Prozentzeichen) und _ (Unterstrich) verwenden.

IAM_ROLE default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>'

Wenn Sie beim Ausführen des SHOW DATABASE-Befehls eine IAM-Rolle angeben, die dem Cluster zugeordnet ist, benutzt Amazon Redshift die Anmeldeinformationen der Rolle, wenn Sie Abfragen in der Datenbank ausführen.

Die Angabe des Schlüsselworts `default` bedeutet, die IAM-Rolle zu verwenden, die als Standard festgelegt und mit dem Cluster verknüpft ist.

Benutzen Sie `'SESSION'`, wenn Sie über eine Verbundidentität eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellen und über das mit dem the section called “CREATE DATABASE”-Befehl erstellte externe Schema auf die Tabellen zugreifen. Ein Beispiel zur Verwendung einer Verbundidentität finden Sie unter Verwenden einer Verbundidentität zur Verwaltung des Amazon-Redshift-Zugriffs auf lokale Ressourcen und externe Amazon-Redshift-Spectrum-Tabellen. Darin wird erläutert, wie Sie eine Verbundidentität konfigurieren.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Die IAM-Rolle muss mindestens die Berechtigung besitzen, eine LIST-Operation für den Amazon-S3-Bucket auszuführen, auf den zugegriffen werden soll, und eine GET-Operation für die Amazon-S3-Objekte, die der Bucket enthält. Weitere Informationen zu Datenbanken, die aus den AWS Glue Data Catalog for Datashares und mithilfe von IAM_ROLE erstellt wurden, finden Sie unter Working with Lake Formation-managed datashares as a consumer.

Nachfolgend ist die Syntax für die IAM_ROLE-Parameterzeichenfolge für einen einzelnen ARN aufgeführt.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Sie können Rollen miteinander verketten. Auf diese Weise kann der Cluster eine andere IAM-Rolle annehmen, die möglicherweise zu einem anderen Konto gehört. Es können bis zu 10 Rollen miteinander verkettet werden. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.

Fügen Sie dieser IAM-Rolle eine IAM-Berechtigungsrichtlinie ähnlich der folgenden an:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
```

```
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Informationen zu den Schritten für das Erstellen einer IAM-Rolle zur Verwendung mit der Verbundabfrage finden Sie unter Erstellen eines Secrets und einer IAM-Rolle für die Verwendung von Verbundabfragen.

> **Note**
>
> Fügen Sie keine Leerzeichen in die Liste der verketteten Rollen ein.

Nachfolgend finden Sie die Syntax für die Verkettung von drei Rollen.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

Beispiele

Im folgenden Beispiel werden alle Datenkatalogdatenbanken der Konto-ID 123456789012 angezeigt.

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012'

  catalog_id  | database_name |                          database_arn
 |     type     |                                              target_database
                            | location | parameters
--------------+---------------+-----------------------------------------------------
+--------------
+-------------------------------------------------------------------------------------------------
+----------+------------
 123456789012 |   database1   | arn:aws:glue:us-east-1:123456789012:database/database1
 | Data Catalog |
                            |          |
 123456789012 |   database2   | arn:aws:glue:us-east-1:123456789012:database/database2
 | Data Catalog | arn:aws:redshift:us-
```

```
east-1:123456789012:datashare:035c45ea-61ce-86f0-8b75-19ac6102c3b7/database2 |
 |
```

Die folgenden Beispiele veranschaulichen, wie Sie mithilfe der Anmeldeinformationen einer IAM-Rolle alle Datenkatalog-Datenbanken aus Konto-ID 123456789012 anzeigen.

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012' IAM_ROLE default;
```

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012' IAM_ROLE <iam-role-arn>;
```

SHOW MODEL

Zeigt nützliche Informationen über ein Machine-Learning-Modell an, einschließlich dessen Status, die Parameter, die zum Erstellen verwendet werden, und die Vorhersagefunktion mit seinen Eingabeargumenttypen. Sie können die Informationen aus SHOW MODEL verwenden, um das Modell neu zu erstellen. Wenn Basistabellen geändert wurden, führt das Ausführen von CREATE MODEL mit derselben SQL-Anweisung zu einem anderen Modell. Die von SHOW MODEL zurückgegebenen Informationen unterscheiden sich für den Modellbesitzer und für einen Benutzer mit der Berechtigung EXECUTE. SHOW MODELL zeigt unterschiedliche Ausgaben an, wenn ein Modell von Amazon Redshift trainiert wird oder wenn es sich um ein BYOM-Modell handelt.

Syntax

```
SHOW MODEL ( ALL | model_name )
```

Parameter

ALL

Gibt alle Modelle zurück, die der Benutzer verwenden kann, und ihre Schemata.

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

Nutzungshinweise

Der Befehl SHOW MODEL erzeugt die folgende Ausgabe.

- Der Modellname.
- Das Schema, in dem das Modell erstellt wurde.
- Der Besitzer des Modells.
- Zeitpunkt der Erstellung des Modells.
- Der Status des Modells, z. B. READY, TRAINING oder FAILED.
- Der Grund für den Fehlschlag eines Modells.
- Der Validierungsfehler, wenn das Modell das Training abgeschlossen hat.
- Die geschätzten Kosten, die erforderlich sind, um das Modell für einen Nicht-BYOM-Ansatz abzuleiten. Nur der Besitzer des Modells kann diese Informationen anzeigen.
- Eine Liste der benutzerdefinierten Parameter und deren Werte, insbesondere die folgenden:
  - Die angegebene TARGET-Spalte.
  - Der Modelltyp, AUTO oder XGBoost.
  - Der Problemtyp, z. B. REGRESSION, BINARY_CLASSIFICATION, MULTICLASS_CLASSIFICATION. Dieser Parameter ist spezifisch für AUTO.
  - Der Name des SageMaker Amazon-Schulungsjobs oder des Amazon SageMaker Autopilot-Jobs, mit dem das Modell erstellt wurde. Sie können diesen Jobnamen verwenden, um weitere Informationen über das Modell bei Amazon zu finden SageMaker.
  - Das Ziel, wie MSE, F1, Genauigkeit. Dieser Parameter ist spezifisch für AUTO.
  - Der Name der erstellten Funktion.
  - Die Art der Inferenz, lokal oder remote.
  - Die Eingabeargumente der Vorhersagefunktion.
  - Die Eingabeargumenttypen der Vorhersagefunktion für Modelle, die nicht BYOM sind.
  - Der Rückgabewert der Vorhersagefunktion. Dieser Parameter ist spezifisch für BYOM.
  - Der Name des SageMaker Amazon-Endpunkts für ein BYOM-Modell mit Remote-Inferenz.
  - Die IAM-Rolle. Kann nur der Besitzer des Modells sehen.
  - Der Name des S3 Buckets. Kann nur der Besitzer des Modells sehen.
  - Der AWS KMS Schlüssel, falls einer angegeben wurde. Kann nur der Besitzer des Modells sehen.
  - Die maximale Ausführungsdauer des Modells.
- Wenn der Modelltyp nicht AUTO ist, zeigt Amazon Redshift auch die Liste der bereitgestellten Hyperparameter und deren Werte an.

Sie können auch einige der von SHOW MODEL bereitgestellten Informationen in anderen Katalogtabellen wie pg_proc anzeigen. Amazon Redshift gibt Informationen über die Vorhersagefunktion zurück, die in der Katalogtabelle pg_proc registriert ist. Zu diesen Informationen gehören die Namen der Eingabeargumente und deren Typen für die Vorhersagefunktion. Amazon Redshift liefert dieselben Informationen mit dem Befehl SHOW MODEL.

```
SELECT * FROM pg_proc WHERE proname ILIKE '%<function_name>%';
```

Beispiele

Das folgende Beispiel zeigt die Ausgabe für SHOW MODEL.

```
SHOW MODEL ALL;

Schema Name |  Model Name
------------+---------------
 public     | customer_churn
```

Der Besitzer von customer_churn kann die folgende Ausgabe sehen. Ein Benutzer, der nur die Berechtigung EXECUTE hat, kann die IAM-Rolle, den Amazon-S3-Bucket und die geschätzten Kosten des Modells nicht sehen.

```
SHOW MODEL customer_churn;

       Key                  |           Value
----------------------------+-----------------------------------
 Model Name                 | customer_churn
 Schema Name                | public
 Owner                      | 'owner'
 Creation Time              | Sat, 15.01.2000 14:45:20
 Model State                | READY
 validation:F1              | 0.855
 Estimated Cost             | 5.7
                            |
 TRAINING DATA:             |
 Table                      | customer_data
 Target Column              | CHURN
                            |
 PARAMETERS:                |
 Model Type                 | auto
 Problem Type               | binary_classification
```

```
Objective                | f1
Function Name            | predict_churn
Function Parameters      | age zip average_daily_spend average_daily_cases
Function Parameter Types | int int float float
IAM Role                 | 'iam_role'
KMS Key                  | 'kms_key'
Max Runtime              | 36000
```

SHOW DATASHARES

Zeigt die eingehenden und ausgehenden Datashares in einem Cluster aus demselben Konto oder kontenübergreifend an. Wenn Sie keinen Datashare-Namen angeben, zeigt Amazon Redshift alle Datashares in allen Datenbanken im Cluster an. Benutzer mit ALTER- und SHARE-Berechtigungen können die Datashares anzeigen, für die sie über Berechtigungen verfügen.

Syntax

```
SHOW DATASHARES [ LIKE 'namepattern' ]
```

Parameter

LIKE

Eine optionale Klausel, die das angegebene Namensmuster mit der Beschreibung des Datashares vergleicht. Wenn diese Klausel verwendet wird, zeigt Amazon Redshift nur die Datashares an, deren Namen dem angegebenen Namensmuster entsprechen.

namepattern

Der Name des angeforderten Datashares oder ein Teil des Namens, der mit Platzhalterzeichen übereinstimmt.

Beispiele

Im folgenden Beispiel werden die eingehenden und ausgehenden Datashares in einem Cluster angezeigt.

```
SHOW DATASHARES;
SHOW DATASHARES LIKE 'sales%';
```

```
share_name   | share_owner | source_database | consumer_database | share_type |
 createdate          | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-------------+-------------+-----------------+-------------------+------------
+---------------------+---------------------+-----------+-----------------
+--------------------------------------
'salesshare' | 100         | dev             |                   | outbound
 | 2020-12-09 01:22:54.| False               |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

SHOW PROCEDURE

Zeigt die Definition einer bestimmten gespeicherten Prozedur, einschließlich ihrer Signatur. Sie können die Ausgabe von SHOW PROCEDURE verwenden, um die gespeicherte Prozedur neu zu erstellen.

Syntax

```
SHOW PROCEDURE sp_name [( [ [ argname ] [ argmode ] argtype [, ...] ] )]
```

Parameter

sp_name

Der Name der Prozedur, die angezeigt werden soll.

[argname] [argmode] argtype

Eingabeargumenttypen zum Identifizieren der gespeicherten Prozedur. Optional können Sie alle Argumentdatentypen angeben, auch die der OUT-Argumente. Dieser Teil ist optional, wenn der Name der gespeicherten Prozedur eindeutig (also nicht überladen) ist.

Beispiele

Das folgende Beispiel zeigt die Definition der Prozedur `test_spl2`.

```
show procedure test_sp2(int, varchar);
                                        Stored Procedure Definition
-------------------------------------------------------------------------------------------------
```

```
CREATE OR REPLACE PROCEDURE public.test_sp2(f1 integer, INOUT f2 character varying, OUT
 character varying)
LANGUAGE plpgsql
AS $_$
DECLARE
out_var alias for $3;
loop_var int;
BEGIN
IF f1 is null OR f2 is null THEN
RAISE EXCEPTION 'input cannot be null';
END IF;
CREATE TEMP TABLE etl(a int, b varchar);
FOR loop_var IN 1..f1 LOOP
insert into etl values (loop_var, f2);
f2 := f2 || '+' || f2;
END LOOP;
SELECT INTO out_var count(*) from etl;
END;
$_$

(1 row)
```

SHOW SCHEMAS

Zeigt eine Liste von Schemata in einer Datenbank zusammen mit einigen Schemaattributen an.

Jede Ausgabezeile besteht aus dem Datenbanknamen, dem Schemanamen, dem Schemabesitzer, dem Schematyp, der Schema-ACL, der Quelldatenbank und der Schemaoption. Weitere Informationen zu diesen Attributen finden Sie unter SVV_ALL_SCHEMAS.

Wenn der Befehl SHOW SCHEMAS mehr als 10 000 Schemas ergeben würde, wird ein Fehler zurückgegeben.

Syntax

```
SHOW SCHEMAS FROM DATABASE database_name [LIKE 'filter_pattern'] [LIMIT row_limit ]
```

Parameter

database_name

Der Name der Datenbanktabelle, welche die aufzulistenden Tabellen enthält.

Um Tabellen in einer anzuzeigen AWS Glue Data Catalog, geben Sie (`awsdatacatalog`) als Datenbanknamen an und stellen Sie sicher, dass die Systemkonfiguration auf eingestellt `data_catalog_auto_mount` ist`true`. Weitere Informationen finden Sie unter ALTER SYSTEM.

filter_pattern

Ein gültiger UTF-8-Zeichenfolgenausdruck mit einem Muster zum Abgleich von Schemanamen. Die Option LIKE führt eine Suche durch, bei der zwischen Groß- und Kleinschreibung unterschieden wird und welche die folgenden Metazeichen für den Mustervergleich unterstützt:

Metazeichen	Beschreibung
%	Entspricht einer Folge von 0 oder mehr Zeichen.
_	Entspricht einem beliebigen Zeichen.

Wenn filter-pattern keine Metazeichen enthält, repräsentiert das Muster die Zeichenfolge selbst. In diesem Fall liefert LIKE dasselbe Ergebnis wie der Gleichheitsoperator.

row_limit

Die maximale Anzahl der zurückzugebenden Zeilen. Der Wert row_limit kann 0–10 000 betragen.

Beispiele

Das folgende Beispiel zeigt die Schemata aus der Amazon-Redshift-Datenbank mit dem Namen `dev`.

```
SHOW SCHEMAS FROM DATABASE dev;

 database_name |     schema_name      | schema_owner | schema_type |         schema_acl
          | source_database | schema_option
---------------+----------------------+--------------+-------------
+---------------------------+-----------------+--------------
 dev           | pg_automv            |            1 | local       |
          |                 |
 dev           | pg_catalog           |            1 | local       | jpuser=UC/
jpuser~=U/jpuser  |                 |
 dev           | public               |            1 | local       | jpuser=UC/
jpuser~=UC/jpuser |                 |
 dev           | information_schema   |            1 | local       | jpuser=UC/
jpuser~=U/jpuser  |                 |
```

```
 dev           | schemad79cd6d93bf043 |           1 | local       |
          |                 |
```

Das folgende Beispiel zeigt die Schemas in der AWS Glue Data Catalog Datenbank mit dem Namen`awsdatacatalog`. Die maximale Anzahl von Ausgabezeilen ist 5.

```
SHOW SCHEMAS FROM DATABASE awsdatacatalog LIMIT 5;

 database_name  |     schema_name      | schema_owner | schema_type | schema_acl |
 source_database | schema_option
----------------+----------------------+--------------+-------------+------------
+-----------------+---------------
 awsdatacatalog | 000_too_many_glue_db |              | EXTERNAL    |            |
            |
 awsdatacatalog | 123_default          |              | EXTERNAL    |            |
            |
 awsdatacatalog | adhoc                |              | EXTERNAL    |            |
            |
 awsdatacatalog | all_shapes_10mb      |              | EXTERNAL    |            |
            |
 awsdatacatalog | all_shapes_1g        |              | EXTERNAL    |            |
            |
```

SHOW TABLE

Zeigt die Definition einer Tabelle an, einschließlich Tabellenattribute, Tabelleneinschränkungen, Spaltenattribute und Spalteneinschränkungen. Sie können die Ausgabe der SHOW TABLE-Anweisung verwenden, um die Tabelle neu zu erstellen.

Weitere Informationen zum Erstellen einer Tabelle finden Sie unter CREATE TABLE.

Syntax

```
SHOW TABLE [schema_name.]table_name
```

Parameter

schema_name

(Optional) Der Name des zugehörigen Schemas.

table_name

Der Name der Tabelle, die angezeigt werden soll.

Beispiele

Es folgt ein Beispiel für die Ausgabe von SHOW TABLE für die Tabelle `sales`.

```
show table sales;
```

```
CREATE TABLE public.sales (
salesid integer NOT NULL ENCODE az64,
listid integer NOT NULL ENCODE az64 distkey,
sellerid integer NOT NULL ENCODE az64,
buyerid integer NOT NULL ENCODE az64,
eventid integer NOT NULL ENCODE az64,
dateid smallint NOT NULL,
qtysold smallint NOT NULL ENCODE az64,
pricepaid numeric(8,2) ENCODE az64,
commission numeric(8,2) ENCODE az64,
saletime timestamp without time zone ENCODE az64
)
DISTSTYLE KEY SORTKEY ( dateid );
```

Es folgt ein Beispiel für die Ausgabe von SHOW TABLE für die Tabelle `category` im Schema `public`.

```
show table public.category;
```

```
CREATE TABLE public.category (
catid smallint NOT NULL distkey,
catgroup character varying(10) ENCODE lzo,
catname character varying(10) ENCODE lzo,
catdesc character varying(50) ENCODE lzo
) DISTSTYLE KEY SORTKEY ( catid );
```

Im folgenden Beispiel wird die Tabelle `foo` mit einem Primärschlüssel erstellt.

```
create table foo(a int PRIMARY KEY, b int);
```

Die SHOW TABLE-Ergebnisse zeigen die create-Anweisung mit allen Eigenschaften der `foo`-Tabelle.

```
show table foo;
```

```
CREATE TABLE public.foo ( a integer NOT NULL ENCODE az64, b integer ENCODE az64,
 PRIMARY KEY (a) ) DISTSTYLE AUTO;
```

SHOW TABLES

Zeigt eine Liste von Tabellen in einem Schema zusammen mit einigen Tabellenattributen an.

Jede Ausgabezeile besteht aus Datenbankname, Schemaname, Tabellenname, Tabellentyp, Tabellen-ACL und Anmerkungen. Weitere Informationen zu diesen Attributen finden Sie unter SVV_ALL_TABLES.

Wenn der Befehl SHOW TABLES mehr als 10 000 Spalten ergeben würde, wird ein Fehler zurückgegeben.

Syntax

```
SHOW TABLES FROM SCHEMA database_name.schema_name [LIKE 'filter_pattern']
 [LIMIT row_limit ]
```

Parameter

database_name

Der Name der Datenbanktabelle, welche die aufzulistenden Tabellen enthält.

Um Tabellen in einer anzuzeigen AWS Glue Data Catalog, geben Sie (`awsdatacatalog`) als Datenbanknamen an und stellen Sie sicher, dass die Systemkonfiguration auf `true` eingestellt `data_catalog_auto_mount` ist. Weitere Informationen finden Sie unter ALTER SYSTEM.

schema_name

Der Name des Schemas, das die aufzulistenden Tabellen enthält.

Um AWS Glue Data Catalog Tabellen anzuzeigen, geben Sie den AWS Glue Datenbanknamen als Schemanamen an.

filter_pattern

Ein gültiger UTF-8-Zeichenfolgenausdruck mit einem Muster zum Abgleich der Tabellennamen. Die Option LIKE führt eine Suche durch, bei der zwischen Groß- und Kleinschreibung unterschieden wird und welche die folgenden Metazeichen für den Mustervergleich unterstützt:

Metazeichen	Beschreibung
%	Entspricht einer Folge von 0 oder mehr Zeichen.
_	Entspricht einem beliebigen Zeichen.

Wenn filter-pattern keine Metazeichen enthält, repräsentiert das Muster die Zeichenfolge selbst. In diesem Fall liefert LIKE dasselbe Ergebnis wie der Gleichheitsoperator.

row_limit

Die maximale Anzahl der zurückzugebenden Zeilen. Der Wert row_limit kann 0–10 000 betragen.

Beispiele

Das folgende Beispiel zeigt die Tabellen in der Amazon-Redshift-Datenbank mit dem Namen `dev` an, die sich im Schema `public` befinden.

```
SHOW TABLES FROM SCHEMA dev.public;

 database_name | schema_name | table_name | table_type | table_acl | remarks
---------------+-------------+------------+------------+-----------+---------
 dev           | public      | tb         | TABLE      |           |
 dev           | public      | tb2        | TABLE      |           |
 dev           | public      | tb3        | TABLE      |           |
```

Das folgende Beispiel zeigt die Tabellen in der genannten AWS Glue Data Catalog Datenbank`awsdatacatalog`, die sich im Schema befinden`batman`.

```
SHOW TABLES FROM SCHEMA awsdatacatalog.batman;

 database_name  | schema_name |    table_name    | table_type | table_acl | remarks
----------------+-------------+------------------+------------+-----------+---------
 awsdatacatalog | batman      | nation           | EXTERNAL   |           |
```

```
 awsdatacatalog | batman      | part             | EXTERNAL   |           |
 awsdatacatalog | batman      | partsupp         | EXTERNAL   |           |
 awsdatacatalog | batman      | region           | EXTERNAL   |           |
 awsdatacatalog | batman      | supplier         | EXTERNAL   |           |
 awsdatacatalog | batman      | automount_nation | EXTERNAL   |           |
```

SHOW VIEW

Zeigt die Definition einer Ansicht, auch für materialisierte Ansichten und Ansichten mit später Bindung. Sie können die Ausgabe der SHOW VIEW-Anweisung verwenden, um die Tabelle neu zu erstellen.

Syntax

```
SHOW VIEW [schema_name.]view_name
```

Parameter

schema_name

(Optional) Der Name des zugehörigen Schemas.

view_name

Der Name der Ansicht, die angezeigt werden soll.

Beispiele

Nachfolgend sehen Sie die Ansichtsdefinition für `LA_Venues_v`.

```
create view LA_Venues_v as select * from venue where venuecity='Los Angeles';
```

Es folgt ein Beispiel für den Befehl SHOW VIEW und die Ausgabe für die zuvor definierte Ansicht.

```
show view LA_Venues_v;
```

```
SELECT venue.venueid,
venue.venuename,
venue.venuecity,
venue.venuestate,
venue.venueseats
```

```
FROM venue WHERE ((venue.venuecity)::text = 'Los Angeles'::text);
```

Nachfolgend sehen Sie die Ansichtsdefinition für die Ansicht `public.Sports_v` im Schema `public`.

```
create view public.Sports_v as select * from category where catgroup='Sports';
```

Es folgt ein Beispiel für den Befehl SHOW VIEW und die Ausgabe für die zuvor definierte Ansicht.

```
show view public.Sports_v;
```

```
SELECT category.catid,
category.catgroup,
category.catname,
category.catdesc
FROM category WHERE ((category.catgroup)::text = 'Sports'::text);
```

START TRANSACTION

Synonym der Funktion BEGIN.

Siehe BEGIN.

TRUNCATE

Löscht alle Zeilen aus einer Tabelle, ohne die Tabelle zu scannen: Dieser Vorgang stellt eine schnellere Alternative zu einer nicht qualifizierten DELETE-Operation dar. Um einen TRUNCATE-Befehl ausführen zu können, müssen Sie über die TRUNCATE TABLE-Berechtigung verfügen, Eigentümer der Tabelle oder Superuser sein. Mit dem GRANT-Befehl können Sie Berechtigungen zum Kürzen einer Tabelle erteilen.

TRUNCATE ist sehr viel effizienter als DELETE und erfordert weder VACUUM noch ANALYZE. Denken Sie jedoch daran, dass TRUNCATE ein Commit für die Transaktion ausführt, in der er ausgeführt wird.

Syntax

```
TRUNCATE [ TABLE ] table_name
```

Der Befehl funktioniert auch in einer materialisierten Ansicht.

```
TRUNCATE materialized_view_name
```

Parameter

TABLE

Optionales Schlüsselwort.

table_name

Eine temporäre oder persistente Tabelle. Nur der Besitzer der Tabelle oder ein Superuser können sie verkürzen.

Sie können jede Tabelle abschneiden, auch Tabellen, die in Fremdschlüsseleinschränkungen referenziert werden.

Sie müssen eine Tabelle nicht bereinigen, nachdem Sie sie abgeschnitten haben.

materialized_view_name

Eine materialisierte Ansicht.

Sie können eine materialisierte Ansicht, die für Streaming-Erfassung verwendet wird, kürzen.

Nutzungshinweise

Der Befehl TRUNCATE führt ein Commit für die Transaktion aus, in der er ausgeführt wird. Daher können Sie für eine TRUNCATE-Operation kein Rollback ausführen. Ein TRUNCATE-Befehl kann ein Commit für andere Operationen ausführen, wenn er ein Commit für sich selbst ausführt.

Beispiele

Verwenden Sie den Befehl TRUNCATE, um alle Zeilen aus der Tabelle CATEGORY zu löschen:

```
truncate category;
```

Im folgenden Beispiel wird versucht, ein Rollback für eine TRUNCATE-Operation auszuführen:

```
begin;

truncate date;
```

```
rollback;

select count(*) from date;
count
-------
0
(1 row)
```

Die Tabelle DATE bleibt nach dem ROLLBACK-Befehl leer, da der TRUNCATE-Befehl automatisch ein Commit ausgeführt hat.

Im folgenden Beispiel wird der Befehl TRUNCATE verwendet, um alle Zeilen einer materialisierten Ansicht zu löschen.

```
truncate my_materialized_view;
```

Er löscht alle Datensätze in der materialisierten Ansicht und lässt die materialisierte Ansicht und ihr Schema unverändert. In der Abfrage ist der Name der materialisierten Ansicht ein Beispiel.

UNLOAD

Entlädt das Ergebnis einer Abfrage in eine oder mehrere Textdateien, JSON-Dateien oder in Apache-Parquet-Dateien in Amazon S3 unter Verwendung der serverseitigen Amazon-S3-Verschlüsselung (SSE-S3). Sie können auch eine serverseitige Verschlüsselung mit einem AWS Key Management Service -Schlüssel (SSE-KMS) oder eine clientseitige Verschlüsselung mit einem vom Kunden verwalteten Schlüssel angeben.

Standardmäßig ist das Format der entladenen Datei ein mit senkrechten Strichen (Pipeoperator) getrennter Text (|).

Sie können die Größe von Dateien in Amazon S3 und damit auch die Anzahl von Dateien verwalten, indem Sie den Parameter MAXFILESIZE festlegen. Stellen Sie sicher, dass die S3-IP-Bereiche zu Ihrer Zulassungsliste hinzugefügt werden. Weitere Informationen zu den erforderlichen S3-IP-Bereichen finden Sie unter Netzwerkisolierung.

Sie können das Ergebnis einer Abfrage in Amazon Redshift in Ihr Amazon S3-Data Lake in Apache Parquet entladen, einem effizienten offenen spaltenbasierten Speicherformat für Analysen. Das Parquet-Format ist bis zu 2x schneller zu entladen und belegt in Amazon S3 bis zu 6x weniger Speicherplatz als Textformate. Auf diese Weise können Sie die Datentransformation und -anreicherung, die Sie in Amazon S3 vorgenommen haben, in Ihrem Amazon S3-Data Lake in einem

offenen Format speichern. Anschließend können Sie Ihre Daten mit Redshift Spectrum und anderen AWS Diensten wie Amazon Athena, Amazon EMR und Amazon analysieren. SageMaker

Weitere Informationen und Beispielszenarien zur Verwendung des Befehls UNLOAD finden Sie unter Entfernen von Daten.

Erforderliche Rechte und Berechtigungen

Damit der UNLOAD-Befehl erfolgreich ausgeführt werden kann, sind mindestens das SELECT-Recht für die Daten in der Datenbank sowie die Berechtigung zum Schreiben in den Amazon-S3-Speicherort erforderlich. Die erforderlichen Berechtigungen ähneln denen beim COPY-Befehl. Weitere Informationen zu Berechtigungen beim COPY-Befehl finden Sie unter Berechtigungen für den Zugriff auf andere AWS -Ressourcen.

Syntax

```
UNLOAD ('select-statement')
TO 's3://object-path/name-prefix'
authorization
[ option, ...]

where authorization is
IAM_ROLE { default | 'arn:aws:iam::<AWS-Konto-id-1>:role/<role-
name>[,arn:aws:iam::<AWS-Konto-id-2>:role/<role-name>][,...]' }

where option is
| [ FORMAT [ AS ] ] CSV | PARQUET | JSON
| PARTITION BY ( column_name [, ... ] ) [ INCLUDE ]
| MANIFEST [ VERBOSE ]
| HEADER
| DELIMITER [ AS ] 'delimiter-char'
| FIXEDWIDTH [ AS ] 'fixedwidth-spec'
| ENCRYPTED [ AUTO ]
| BZIP2
| GZIP
| ZSTD
| ADDQUOTES
| NULL [ AS ] 'null-string'
| ESCAPE
| ALLOWOVERWRITE
| CLEANPATH
| PARALLEL [ { ON | TRUE } | { OFF | FALSE } ]
| MAXFILESIZE [AS] max-size [ MB | GB ]
```

```
| ROWGROUPSIZE [AS] size [ MB | GB ]
| REGION [AS] 'aws-region' }
| EXTENSION 'extension-name'
```

Parameter

('select-statement')

Eine SELECT-Abfrage. Die Ergebnisse der Abfrage werden entladen. In den meisten Fällen lohnt es sich, Daten in sortierter Reihenfolge zu entladen, indem in der Abfrage eine ORDER BY-Klausel angegeben wird. Dieser Ansatz spart die für das Sortieren der Daten beim erneuten Laden benötigte Zeit.

Die Abfrage muss in einfache Anführungszeichen gesetzt werden, wie im Folgenden gezeigt:

```
('select * from venue order by venueid')
```

> **Note**
>
> Wenn Ihre Abfrage Anführungszeichen enthält (z. B. um Literalwerte einzuschließen), setzen Sie das Literal zwischen zwei Sätze einfacher Anführungszeichen. Sie müssen auch die Abfrage in einfache Anführungszeichen einschließen:
>
> ```
> ('select * from venue where venuestate=''NV''')
> ```

TO 's3://object-path/name-prefix'

Der vollständige Pfad, einschließlich des Bucket-Namens, zum Speicherort in Amazon S3, an dem Amazon Redshift die ausgegebenen Dateiobjekte schreibt, einschließlich der Manifest-Datei, wenn MANIFEST angegeben ist. Den Objektnamen wird name-prefix vorangestellt. Wenn Sie `PARTITION BY` verwenden, wird bei Bedarf automatisch ein Schrägstrich (/) am Ende des Namens-Präfix-Wertes eingefügt. Um zusätzliche Sicherheit bereitzustellen, stellt UNLOAD eine Verbindung zu Amazon S3 über HTTPS her. Standardmäßig schreibt UNLOAD eine oder mehrere Dateien pro Slice. UNLOAD fügt eine Slice-Nummer und Teilenummer an das angegebene Namenspräfix wie im Folgenden gezeigt an:

<object-path>/*<name-prefix><slice-number>*_part_*<part-number>*.

Wenn MANIFEST angegeben ist, wird die Manifestdatei wie folgt geschrieben:

<object_path>/*<name_prefix>*manifest.

Wenn PARALLEL auf OFF festgelegt ist, werden die Datendateien wie folgt geschrieben:

<object_path>/*<name_prefix>**<part-number>*.

UNLOAD erstellt automatisch mithilfe der serverseitigen Amazon-S3-Verschlüsselung (SSE) verschlüsselte Dateien, einschließlich der Manifestdatei, wenn MANIFEST verwendet wird. Der Befehl COPY liest automatisch serverseitig verschlüsselte Daten während der Ladeoperationen. Sie können serverseitig verschlüsselte Dateien über die Amazon-S3-Managementkonsole oder die API transparent aus Ihrem Bucket herunterladen. Weitere Informationen finden Sie unter Schutz von Daten mittels serverseitiger Verschlüsselung.

Um die clientseitige Amazon-S3-Verschlüsselung zu verwenden, geben Sie die ENCRYPTED-Option an.

> **Important**
>
> REGION ist erforderlich, wenn sich der Amazon-S3-Bucket nicht in derselben AWS-Region wie die Amazon-Redshift-Datenbank befindet.

Autorisierung

Der Befehl UNLOAD benötigt eine Autorisierung, um Daten zu Amazon S3 zu schreiben. Der Befehl UNLOAD verwendet für die Autorisierung dieselben Parameter wie der Befehl COPY. Weitere Informationen finden Sie unter Autorisierungsparameter in der COPY-Befehlssyntaxreferenz.

IAM_ROLE { default | 'arn:aws:iam::*<AWS-Konto-id-1>*:role/*<role-name>*'

Verwenden Sie das Standardstichwort, damit Amazon Redshift die IAM-Rolle verwendet, die als Standard festgelegt und mit dem Cluster verknüpft ist, wenn der UNLOAD-Befehl ausgeführt wird.

Verwenden Sie den Amazon-Ressourcennamen (ARN) für eine IAM-Rolle, die von Ihrem Cluster für Authentifizierung und Autorisierung verwendet wird. Wenn Sie IAM_ROLE angeben, können Sie ACCESS_KEY_ID und SECRET_ACCESS_KEY, SESSION_TOKEN oder CREDENTIALS nicht verwenden. IAM_ROLE kann verkettet werden. Weitere Informationen finden Sie unter Verketten von IAM-Rollen im Amazon-Redshift-Verwaltungshandbuch.

[FORMAT [AS]] CSV | PARQUET | JSON

Schlüsselwörter zur Angabe des Entladeformats, die das Standardformat außer Kraft setzen.

Bei CSV wird eine Entladung in eine Textdatei im CSV-Format durchgeführt, bei der ein Kommazeichen (,) als Standard-Trennzeichen verwendet wird. Wenn ein Feld Trennzeichen, doppelte Anführungszeichen, Zeilenvorschubzeichen oder Zeilenumbrüche enthält, wird das Feld in der entladenen Datei in doppelte Anführungszeichen eingeschlossen. Einem doppelten Anführungszeichen innerhalb eines Datenfeldes wird ein zusätzliches doppeltes Anführungszeichen vorangestellt. Wenn null Zeilen entladen werden, schreibt Amazon Redshift möglicherweise leere Amazon-S3-Objekte.

Im Falle von PARQUET wird in eine Datei im Apache-Parquet-Version-1.0-Format entladen. Standardmäßig wird jede Zeilengruppe mittels SNAPPY-Komprimierung komprimiert. Weitere Informationen zum Apache-Parquet-Format finden Sie unter Parquet.

Im Fall von JSON erfolgt eine Entladung in eine JSON-Datei, wobei jede Zeile ein JSON-Objekt enthält, das einen vollständigen Datensatz im Abfrageergebnis darstellt. Amazon Redshift unterstützt das Schreiben von verschachteltem JSON, wenn das Abfrageergebnis SUPER-Spalten enthält. Um ein gültiges JSON-Objekt zu erstellen, muss der Name jeder Spalte in der Abfrage eindeutig sein. In der JSON-Datei werden boolesche Werte als `t` oder `f` und NULL-Werte als `null` entladen. Wenn null Zeilen entladen werden, schreibt Amazon Redshift keine Amazon-S3-Objekte.

Die Schlüsselwörter FORMAT und AS sind optional. Sie können CSV nicht mit FIXEDWIDTH oder ADDQUOTES verwenden. Sie können PARQUET nicht mit DELIMITER, FIXEDWIDTH, ADDQUOTES, ESCAPE, NULL AS, HEADER, GZIP, BZIP2 oder ZSTD verwenden. PARQUET with ENCRYPTED wird nur bei serverseitiger Verschlüsselung mit einem AWS Key Management Service Schlüssel (SSE-KMS) unterstützt. Sie können JSON nicht mit DELIMITER, HEADER, FIXEDWIDTH, ADDQUOTES, ESCAPE oder NULL AS verwenden.

PARTITION BY (column_name [, ...]) [INCLUDE]

Gibt die Partitionsschlüssel für die Entladeoperation an. Über UNLOAD werden die Ausgabedateien entsprechend der Apache-Hive-Konvention automatisch in Partitionsordner basierend auf den Partitionsschlüsselwerten partitioniert. Beispielsweise verfügt eine Parquet-Datei, die zum Partitionsjahr 2019 und dem Monat September gehört, über das folgende Präfix: `s3://my_bucket_name/my_prefix/year=2019/month=September/000.parquet`.

Der Wert für column_name muss eine Spalte in den Abfrageergebnissen sein, die entladen werden.

Wenn Sie PARTITION BY mit der Option INCLUDE angeben, werden Partitionsspalten nicht aus den entladenen Dateien entfernt.

Amazon Redshift unterstützt keine Zeichenfolgeliterale in PARTITION-BY-Klauseln.

MANIFEST [VERBOSE]

Erstellt eine Manifestdatei, die explizit Details der Datendateien auflistet, die durch den UNLOAD-Prozess erstellt werden. Das Manifest ist eine Textdatei im JSON-Format, die die URL jeder einzelnen Datei auflistet, die auf Amazon S3 geschrieben wurde.

Bei Angabe von MANIFEST mit der Option VERBOSE enthält das Manifest die folgenden Details:

- Die Spaltennamen und Datentypen und für die Datentypen CHAR, VARCHAR oder NUMERIC Dimension für die einzelnen Spalten. Im Fall der Datentypen CHAR und VARCHAR entspricht die Dimension der Länge. Im Fall der Datentypen DECIMAL oder NUMERIC data type, sind die Dimensionen Genauigkeit und Skalierung.
- Die Anzahl der in die einzelnen Dateien entladenen Zeilen. Bei Angabe der Option HEADER schließt die Zeilenanzahl die Kopfzeile ein.
- Die gesamte Dateigröße aller entladenen Dateien und die Gesamtzahl der in alle Dateien entladenen Zeilen. Bei Angabe der Option HEADER schließt die Zeilenanzahl die Kopfzeilen ein.
- Den Autor. Der Autor ist immer „Amazon Redshift“.

Sie können VERBOSE nur auf MANIFEST folgend angeben.

Die Manifestdatei wird im gleichen Amazon-S3-Pfadpräfix wie die entladenen Dateien im Format geschrieben `<object_path_prefix>manifest`. Wenn UNLOAD beispielsweise das Amazon-S3-Pfadpräfix '`s3://mybucket/venue_`' angibt, ist der Speicherort der Manifestdatei '`s3://mybucket/venue_manifest`'.

HEADER

Fügt eine Kopfzeile mit den Spaltennamen oben in jeder Ausgabedatei hinzu. Texttransformationsoptionen wie CSV, DELIMITER, ADDQUOTES und ESCAPE gelten auch für die Kopfzeile. Sie können HEADER nicht mit FIXEDWIDTH verwenden.

DELIMITER AS 'delimiter_character'

Gibt ein einzelnes ASCII-Zeichen an, das verwendet wird, um Felder in der Ausgabedatei zu trennen, beispielsweise ein Pipe-Zeichen (|), ein Komma (,) oder ein Tabulatorzeichen (\t). Das Standardtrennzeichen für Textdateien ist ein Pipe-Zeichen. Das Standardtrennzeichen für

CSV-Dateien ist ein Kommazeichen. Das Schlüsselwort AS ist optional. Sie können DELIMITER nicht mit FIXEDWIDTH verwenden. Wenn die Daten das Trennzeichen enthalten, müssen Sie die ESCAPE-Option angeben, um ein Escape-Zeichen für das Trennzeichen zu verwenden, oder ADDQUOTES verwenden, um die Daten in doppelte Anführungszeichen einzuschließen. Alternativ können Sie ein Trennzeichen angeben, das nicht in den Daten enthalten ist.

FIXEDWIDTH 'fixedwidth_spec'

Entlädt die Daten in eine Datei, in der jede Spalte eine feste Breite hat, statt durch ein Trennzeichen abgetrennt zu werden. fixedwidth_spec ist eine Zeichenfolge, die die Anzahl der Spalten und die Breite der Spalten angibt. Das Schlüsselwort AS ist optional. Da FIXEDWIDTH keine Daten abschneidet, muss die Spezifikation für jede Spalte in der UNLOAD-Anweisung mindestens so lang wie die Länge des längsten Eintrags für die Spalte sein. Das Format für fixedwidth_spec wird im Folgenden gezeigt:

```
'colID1:colWidth1,colID2:colWidth2, ...'
```

Sie können FIXEDWIDTH nicht mit DELIMITER oder HEADER verwenden.

VERSCHLÜSSELT [AUTO]

Gibt an, dass die Ausgabedateien auf Amazon S3 mit der serverseitigen Verschlüsselung von Amazon S3 oder der clientseitigen Verschlüsselung verschlüsselt werden. Wenn MANIFEST angegeben ist, wird die Manifestdatei ebenfalls verschlüsselt. Weitere Informationen finden Sie unter Entladen verschlüsselter Datendateien. Wenn Sie den ENCRYPTED-Parameter nicht angeben, erstellt UNLOAD automatisch verschlüsselte Dateien mithilfe der serverseitigen Amazon S3 S3-Verschlüsselung mit AWS verwalteten Verschlüsselungsschlüsseln (SSE-S3).

Für ENCRYPTED möchten Sie möglicherweise mithilfe einer serverseitigen Verschlüsselung mit einem AWS KMS Schlüssel (SSE-KMS) nach Amazon S3 entladen. Wenn dies der Fall ist, verwenden Sie den KMS_KEY_ID-Parameter, um die Schlüssel-ID anzugeben. Sie können den Parameter CREDENTIALS nicht mit dem Parameter KMS_KEY_ID verwenden. Wenn Sie einen UNLOAD-Befehl für Daten mit KMS_KEY_ID ausführen, können Sie dann einen COPY-Vorgang für dieselben Daten ausführen, ohne einen Schlüssel anzugeben.

Um die clientseitige Verschlüsselung mit einem vom Kunden bereitgestellten symmetrischen Schlüssel nach Amazon S3 zu entladen, geben Sie den Schlüssel auf zwei Arten an. Verwenden Sie zum Bereitstellen des Schlüssels den MASTER_SYMMETRIC_KEY-Parameter oder den `master_symmetric_key`-Teil einer CREDENTIALS-Anmeldeinformationszeichenfolge. Wenn Sie Daten mit einem symmetrischen Root-Schlüssel entladen, stellen Sie sicher, dass Sie

denselben Schlüssel angeben, wenn Sie einen COPY-Vorgang für die verschlüsselten Daten ausführen.

UNLOAD unterstützt keine serverseitige Amazon-S3-Verschlüsselung mit von Kunden bereitgestellten Schlüsseln (SSE-C).

Wenn ENCRYPTED AUTO verwendet wird, ruft der Befehl UNLOAD den AWS KMS Standardverschlüsselungsschlüssel auf der Zieleigenschaft des Amazon S3-Buckets ab und verschlüsselt die in Amazon S3 geschriebenen Dateien mit dem Schlüssel. AWS KMS Wenn der Bucket nicht über den AWS KMS Standard-Verschlüsselungsschlüssel verfügt, erstellt UNLOAD automatisch verschlüsselte Dateien mithilfe der serverseitigen Amazon Redshift Redshift-Verschlüsselung mit AWS verwalteten Verschlüsselungsschlüsseln (SSE-S3). Sie können diese Option nicht mit KMS_KEY_ID, MASTER_SYMMETRIC_KEY oder CREDENTIALS verwenden, in denen master_symmetric_key enthalten ist.

KMS_KEY_ID 'key-id'

Gibt die Schlüssel-ID für einen AWS Key Management Service (AWS KMS) -Schlüssel an, der zum Verschlüsseln von Datendateien auf Amazon S3 verwendet werden soll. Weitere Informationen finden Sie unter Was ist AWS Key Management Service? Wenn Sie KMS_KEY_ID angeben, müssen Sie auch den Parameter ENCRYPTED angeben. Wenn Sie KMS_KEY_ID angeben, können Sie keine Authentifizierungen mittels des Parameters CREDENTIALS ausführen. Verwenden Sie stattdessen IAM_ROLE oder ACCESS_KEY_ID and SECRET_ACCESS_KEY.

MASTER_SYMMETRIC_KEY 'root_key'

Gibt den symmetrischen Root-Schlüssel an, der für die Verschlüsselung von Datendateien auf Amazon S3 verwendet werden soll. Wenn Sie MASTER_SYMMETRIC_KEY angeben, müssen Sie auch den Parameter ENCRYPTED angeben. MASTER_SYMMETRIC_KEY kann nicht mit dem Parameter CREDENTIALS verwendet werden. Weitere Informationen finden Sie unter Laden verschlüsselter Datendateien aus Amazon S3.

BZIP2

Entlädt Daten zu einer oder mehreren bzip2-komprimierten Dateien pro Slice. Jeder resultierenden Datei wird die Erweiterung `.bz2` angefügt.

GZIP

Entlädt Daten in eine oder mehrere gzip-komprimierte Dateien pro Slice. Jeder resultierenden Datei wird die Erweiterung `.gz` angefügt.

ZSTD

Entlädt Daten zu einer oder mehreren Zstandard-komprimierten Dateien pro Slice. Jeder resultierenden Datei wird die Erweiterung `.zst` angefügt.

ADDQUOTES

Platziert Anführungszeichen um jedes entladene Datenfeld, sodass Amazon Redshift Datenwerte entladen kann, die das Trennzeichen selbst enthalten. Wenn beispielsweise das Trennzeichen ein Komma ist, könnten Sie die folgenden Daten erfolgreich entladen und neu laden:

```
"1","Hello, World"
```

Ohne die hinzugefügten Anführungszeichen würde die Zeichenfolge `Hello, World` als zwei getrennte Felder analysiert werden.

Einige Ausgabeformate unterstützen ADDQUOTES nicht.

Wenn Sie ADDQUOTES verwenden, müssen Sie REMOVEQUOTES in der COPY-Operation angeben, wenn Sie die Daten neu laden.

NULL AS 'null-string'

Gibt eine Zeichenfolge zurück, die einen Null-Wert in entladenen Dateien darstellt. Wenn diese Option verwendet wird, enthalten alle Ausgabedateien die angegebene Zeichenfolge statt Null-Werten, die in den ausgewählten Daten gefunden werden. Wenn diese Option nicht angegeben ist, werden Null-Werte wie folgt entladen:

- Zeichenfolgen mit null Länge für Ausgaben mit Trennzeichen
- Leerzeichen-Zeichenfolgen für Ausgaben mit festen Spaltenbreiten

Wenn eine Null-Zeichenfolge angegeben ist, um eine Ausgabe mit fester Spaltenbreite zu entladen, und die Breite einer Ausgabespalte kleiner als die Breite der Null-Zeichenfolge ist, tritt das folgende Verhalten auf:

- Ein leeres Feld als Ausgabe für Spalten, die keine Zeichen enthalten
- Ein Fehler für Spalten, die Zeichen enthalten

Im Gegensatz zu anderen Datentypen, bei denen eine benutzerdefinierte Zeichenfolge einen Nullwert darstellt, exportiert Amazon Redshift die SUPER-Datenspalten im JSON-Format und stellt sie als null dar, wie durch das JSON-Format bestimmt. Daher ignorieren SUPER-Datenspalten die Option NULL [AS], die in UNLOAD-Befehlen verwendet wird.

ESCAPE

In CHAR- und VARCHAR-Spalten in entladenen Dateien mit Trennzeichen wird ein Escape-Zeichen (\) vor jedes Auftreten der folgenden Zeichen platziert:

- Zeilenvorschub: `\n`
- Zeilenumschaltung: `\r`
- Das Trennzeichen, das für die entladenen Daten angegeben ist.
- Das Escape-Zeichen: `\`
- Ein Anführungszeichen: `"` oder `'` (wenn sowohl ESCAPE als auch ADDQUOTES im Befehl UNLOAD angegeben sind)

> **Important**
>
> Wenn Sie Ihre Daten mittels einer COPY-Operation mit der ESCAPE-Option geladen haben, müssen Sie die ESCAPE-Option auch für Ihren UNLOAD-Befehl angeben, um die reziproke Ausgabedatei zu generieren. Wenn Sie UNLOAD unter Verwendung der Option ESCAPE ausführen, müssen Sie ESCAPE verwenden, wenn Sie eine COPY-Operation für dieselben Daten ausführen.

ALLOWOVERWRITE

Standardmäßig schlägt UNLOAD fehl, wenn die Operation auf Dateien trifft, die sie möglicherweise überschreiben könnte. Bei Angabe von ALLOWOVERWRITE überschreibt UNLOAD vorhandene Dateien einschließlich der Manifestdatei.

CLEANPATH

Die Option CLEANPATH entfernt vorhandene Dateien, die sich in dem in der TO-Klausel angegebenen Amazon-S3-Pfad befinden, bevor Dateien an den angegebenen Speicherort entladen werden.

Wenn Sie die PARTITION BY-Klausel einschließen, werden vorhandene Dateien nur aus den Partitionsordnern entfernt, die neue, durch die UNLOAD-Operation erzeugte Dateien aufnehmen sollen.

Sie müssen über `s3:DeleteObject`-Leseberechtigung für den Amazon-S3-Bucket verfügen. Weitere Informationen finden Sie unter Richtlinien und Berechtigungen in Amazon S3 im Amazon-

Simple-Storage-Service-Benutzerhandbuch. Dateien, die Sie mithilfe der CLEANPATH-Option entfernen, werden dauerhaft gelöscht und können nicht wiederhergestellt werden.

Sie können die Option CLEANPATH nicht angeben, wenn Sie die Option ALLOVOVERWRITE verwenden.

PARALLEL

Standardmäßig schreibt UNLOAD die Daten parallel in mehrere Dateien, je nach der Anzahl der Slices in dem Cluster. Die Standardoption ist ON oder TRUE. Wenn PARALLEL OFF oder FALSE ist, schreibt UNLOAD seriell zu einer oder mehreren Datendateien und ist absolut entsprechend der ORDER BY-Klausel sortiert, wenn eine solche Klausel verwendet wird. Die maximale Größe für eine Datendatei ist 6,2 GB. Wenn Sie beispielsweise 13,4 GB Daten entladen, erstellt UNLOAD die folgenden drei Dateien.

```
s3://mybucket/key000    6.2 GB
s3://mybucket/key001    6.2 GB
s3://mybucket/key002    1.0 GB
```

> **Note**
>
> Der UNLOAD-Befehl ist zur Verwendung der parallelen Verarbeitung gedacht. Wir empfehlen, in den meisten Fällen PARALLEL aktiviert zu lassen, besonders, wenn die Dateien zum Laden von Tabellen mit einem COPY-Befehl verwendet werden sollen.

MAXFILESIZE [AS] max-size [MB | GB]

Gibt die maximale Größe von Dateien an, die UNLOAD in Amazon S3 erstellt. Geben Sie einen Dezimalwert zwischen 5 MB und 6,2 GB an. Das Schlüsselwort AS ist optional. Die Standardeinheit ist MB. Wenn MAXFILESIZE nicht angegeben wird, ist die standardmäßig zulässige Dateigröße 6,2 GB. Die Größe der Manifestdatei, falls verwendet, ist von MAXFILESIZE nicht betroffen.

ROWGROUPSIZE [AS] size [MB | GB]

Gibt die Größe von Zeilengruppen an. Die Auswahl einer größeren Größe kann die Anzahl der Zeilengruppen und somit den Umfang der Netzwerkkommunikation verringern. Geben Sie einen Ganzzahlwert zwischen 32 MB und 128 MB an. Das Schlüsselwort AS ist optional. Die Standardeinheit ist MB.

Wenn ROWGROUPSIZE nicht angegeben ist, beträgt die Standardgröße 32 MB. Um diesen Parameter verwenden zu können, muss das Speicherformat Parquet und der Knotentyp ra3.4xlarge, ra3.16xlarge oder dc2.8xlarge sein.

REGION [AS] 'aws-region'

Gibt an AWS-Region , wo sich der Amazon S3 S3-Ziel-Bucket befindet. REGION ist für UNLOAD in einen Amazon S3 S3-Bucket erforderlich, der sich nicht in derselben AWS-Region Amazon Redshift Redshift-Datenbank befindet.

Der Wert für aws_region muss mit einer AWS Region übereinstimmen, die in der Amazon Redshift Redshift-Tabelle für Regionen und Endpunkte in der aufgeführt ist. Allgemeine AWS-Referenz

Standardmäßig geht UNLOAD davon aus, dass sich der Amazon S3 S3-Ziel-Bucket in derselben AWS-Region Amazon Redshift Redshift-Datenbank befindet.

EXTENSION 'extension-name'

Gibt die Dateierweiterung an, die an die Namen der entladenen Dateien angehängt werden soll. Amazon Redshift führt keine Validierung durch, daher müssen Sie selbst überprüfen, ob die angegebene Dateierweiterung korrekt ist. Wenn Sie eine Komprimierungsmethode wie GZIP verwenden, müssen Sie dennoch `.gz` im Erweiterungsparameter angeben. Wenn Sie keine Erweiterung angeben, fügt Amazon Redshift dem Dateinamen nichts hinzu. Wenn Sie eine Komprimierungsmethode angeben, ohne eine Erweiterung anzugeben, fügt Amazon Redshift dem Dateinamen nur die Erweiterung der Komprimierungsmethode hinzu.

Nutzungshinweise

Verwendung von ESCAPE für alle UNLOAD-Textoperationen mit Trennzeichen

Wenn Sie eine UNLOAD-Operation mit Trennzeichen ausführen, können Ihre Daten dieses Trennzeichen oder eines der Zeichen enthalten, die in der Beschreibung der ESCAPE-Option aufgelistet werden. In diesem Fall müssen Sie die ESCAPE-Option mit der UNLOAD-Anweisung verwenden. Wenn Sie die ESCAPE-Option nicht mit der UNLOAD-Anweisung verwenden, schlagen nachfolgende COPY-Operationen, die die entladenen Daten verwenden, möglicherweise fehl.

> **Important**
>
> Wir empfehlen nachdrücklich, ESCAPE immer mit den Anweisungen UNLOAD und COPY zu verwenden. Eine Ausnahme liegt vor, wenn Sie sich sicher sind, dass Ihre Daten keine

Trennzeichen oder andere Zeichen enthalten, für die möglicherweise ein Escape-Zeichen verwendet werden muss.

Verlust der Gleitkommawert-Präzision

Möglicherweise stellen Sie einen Präzisionsverlust bei Gleitkommawerten fest, die in Folge entladen und neu geladen werden.

Limit-Klausel

Die SELECT-Abfrage kann als umschließende SELECT-Abfrage keine LIMIT-Klausel verwenden. Beispielweise schlägt die folgende UNLOAD-Anweisung fehl.

```
unload ('select * from venue limit 10')
to 's3://mybucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole';
```

Verwenden Sie stattdessen eine verschachtelte LIMIT-Klausel, wie im folgenden Beispiel dargestellt.

```
unload ('select * from venue where venueid in
(select venueid from venue order by venueid desc limit 10)')
to 's3://mybucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole';
```

Sie können eine Tabelle außerdem mit SELECT...INTO oder CREATE TABLE AS über eine LIMIT-Klausel füllen und dann aus dieser Tabelle entladen.

Entladen einer Spalte vom Datentyp GEOMETRY

Sie können GEOMETRY-Spalten nur im Text- oder CSV-Format entladen. Sie können GEOMETRY-Daten nicht mit der `FIXEDWIDTH`-Option entladen. Die Daten werden in hexadezimaler Form des EWKB-Formats entladen. Wenn die Größe der EWKB-Daten mehr als 4 MB beträgt, erfolgt eine Warnung, da die Daten später nicht mehr in eine Tabelle geladen werden können.

Entladen des Datentyps HLLSKETCH

Sie können HLLSKETCH-Spalten nur im Text- oder CSV-Format entladen. Sie können HLLSKETCH-Daten nicht mit der `FIXEDWIDTH`-Option entladen. Die Daten werden im Base64-Format für dichte

HyperLogLog Skizzen oder im JSON-Format für dünn besetzte Skizzen entladen. HyperLogLog Weitere Informationen finden Sie unter HyperLogLog Funktionen.

Im folgenden Beispiel wird eine Tabelle mit HLLSKETCH-Spalten in eine Datei exportiert.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

UNLOAD ('select * from hll_table') TO 's3://mybucket/unload/'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole' NULL AS 'null' ALLOWOVERWRITE
 CSV;
```

Entladen einer Spalte des Datentyps VARBYTE

Sie können VARBYTE-Spalten nur im Text- oder CSV-Format entladen. Die Daten werden in hexadezimaler Form entladen. Sie können keine VARBYTE-Daten mit der `FIXEDWIDTH`-Option entladen. Die `ADDQU0TES`-Option von UNLOAD in das CSV-Format wird nicht unterstützt. VARBYTE-Spalten können keine PARTITIONED BY-Spalten sein.

Klausel FORMAT AS PARQUET

Beachten Sie bei der Verwendung von FORMAT AS PARQUET Folgendes:

- Für das Entladen nach Parquet kommt keine Komprimierung auf Dateieben zur Anwendung. Jede Zeilengruppe wird mit SNAPPY komprimiert.
- Wenn MAXFILESIZE nicht angegeben wird, ist die standardmäßig zulässige Dateigröße 6,2 GB. Sie können MAXFILESIZE verwenden, um eine Dateigröße von 5 MB bis 6,2 GB anzugeben. Die tatsächliche Dateigröße wird beim Schreiben der Datei annähernd beziffert, sodass sie möglicherweise nicht genau der von Ihnen angegebenen Zahl entspricht.

  Um die Scan-Leistung zu maximieren, versucht Amazon Redshift, Parquet-Dateien zu erstellen, die gleich große Zeilengruppen von 32 MB enthalten. Der von Ihnen angegebene MAXFILESIZE -Wert wird automatisch auf das nächste Vielfache von 32 MB abgerundet. Wenn Sie beispielsweise als MAXFILESIZE 200 MB angeben, beträgt jede entladene Parquet-Datei etwa 192 MB (Zeilengruppe von 32 MB x 6 = 192 MB).

- Wenn eine Spalte das Datenformat TIMESTAMPTZ verwendet, werden nur die Zeitstempelwerte entladen. Die Zeitzoneninformationen werden nicht entladen.
- Geben Sie keine Dateinamenpräfixe an, die mit Unterstrich (_) oder Punkt (.) beginnen. Redshift Spectrum behandelt Dateien, die mit diesen Zeichen beginnen, als versteckte Dateien und ignoriert sie.

Klausel PARTITION BY

Beachten Sie bei der Verwendung von PARTITION BY Folgendes:

- Partitionsspalten sind nicht in der Ausgabedatei enthalten.
- Stellen Sie sicher, dass die Partitionsspalten in der SELECT-Abfrage, die in der UNLOAD-Anweisung verwendet wird, enthalten sind. Sie können beliebig viele Partitionsspalten im UNLOAD-Befehl angeben. Allerdings gibt es eine Einschränkung, nämlich dass mindestens eine Nicht-Partitionsspalte vorhanden sein sollte, die Teil der Datei sein sollte.
- Wenn der Wert des Partitionsschlüssels null ist, entlädt Amazon Redshift diese Daten automatisch in eine Standardpartition namens `partition_column=__HIVE_DEFAULT_PARTITION__`.
- Der Befehl UNLOAD führt keine Aufrufe an einen externen Katalog durch. Um Ihre neuen Partitionen als Teil Ihrer vorhandenen externen Tabelle zu registrieren, verwenden Sie einen separaten Befehl ALTER TABLE... ADD PARTITION... Alternativ können Sie einen Befehl CREATE EXTERNAL TABLE ausführen, um die entladenen Daten als neue externe Tabelle zu registrieren. Sie können Ihren Datenkatalog auch mit einem AWS Glue Crawler auffüllen. Weitere Informationen finden Sie unter Definieren von Crawlern im AWS Glue -Entwicklerhandbuch.
- Wenn Sie die MANIFEST-Option verwenden, generiert Amazon Redshift nur eine Manifestdatei im Amazon-S3-Stammordner.
- Die Spaltendatentypen, die Sie als Partitionsschlüssel verwenden können, sind SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, BOOLEAN, CHAR, VARCHAR, DATE und TIMESTAMP.

Verwenden der ASSUMEROLE-Berechtigung, um einer IAM-Rolle Zugriff auf UNLOAD-Vorgänge zu gewähren

Um bestimmten Benutzern und Gruppen den Zugriff auf eine IAM-Rolle für UNLOAD-Vorgänge zu ermöglichen, kann ein Superuser Benutzern und Gruppen das ASSUMEROLE-Recht für eine IAM-Rolle gewähren. Weitere Informationen finden Sie unter GRANT.

UNLOAD unterstützt keine Amazon-S3-Zugriffspunkt-Aliase

Sie können keine Amazon-S3-Zugriffspunkt-Aliase mit dem UNLOAD-Befehl verwenden.

Beispiele

Beispiele zur Verwendung des UNLOAD-Befehls finden Sie unter UNLOAD-Beispiele.

UNLOAD-Beispiele

Diese Beispiele veranschaulichen verschiedene Parameter des UNLOAD-Befehls. Der TICKIT-Beispieldatensatz wird in vielen Beispielen verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

> **Note**
>
> In den folgenden Beispielen werden aus Gründen der Lesbarkeit Zeilenumbrüche verwendet. Verwenden Sie in Ihrer Zeichenfolge credentials-args keine Zeilenumbrüche oder Leerzeichen.

Entladen von VENUE in eine Datei, die das Pipe-Zeichen als Trennzeichen enthält (Standardtrennzeichen)

Im folgenden Beispiel werden die Tabelle VENUE entladen und die Daten in geschrieben `s3://mybucket/unload/`:

```
unload ('select * from venue')
to 's3://mybucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Standardmäßig schreibt UNLOAD eine oder mehrere Dateien pro Slice. Unter Annahme eines Clusters mit zwei Knoten und zwei Slices pro Knoten werden im vorherigen Beispiel in die folgenden Dateien erstellt `mybucket`:

```
unload/0000_part_00
unload/0001_part_00
unload/0002_part_00
unload/0003_part_00
```

Um die Ausgabedateien besser zu differenzieren, können Sie für den Speicherort ein Präfix verwenden. Im folgenden Beispiel werden die Tabelle VENUE entladen und die Daten in `s3://mybucket/unload/venue_pipe_` geschrieben:

```
unload ('select * from venue')
to 's3://mybucket/unload/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Das Ergebnis dieser vier Dateien ist der Ordner `unload`, erneut unter Annahme von vier Slices.

```
venue_pipe_0000_part_00
venue_pipe_0001_part_00
venue_pipe_0002_part_00
venue_pipe_0003_part_00
```

Tabelle LINEITEM in partitionierte Parquet-Dateien entladen

Im folgenden Beispiel wird die Tabelle LINEITEM im Parquet-Format entladen, das durch die `l_shipdate`-Spalte unterteilt wird.

```
unload ('select * from lineitem')
to 's3://mybucket/lineitem/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
PARQUET
PARTITION BY (l_shipdate);
```

Unter der Annahme von vier Slices werden die resultierenden Parquet-Dateien dynamisch in verschiedene Ordner aufgeteilt.

```
s3://mybucket/lineitem/l_shipdate=1992-01-02/0000_part_00.parquet
                                             0001_part_00.parquet
                                             0002_part_00.parquet
                                             0003_part_00.parquet
s3://mybucket/lineitem/l_shipdate=1992-01-03/0000_part_00.parquet
                                             0001_part_00.parquet
                                             0002_part_00.parquet
                                             0003_part_00.parquet
s3://mybucket/lineitem/l_shipdate=1992-01-04/0000_part_00.parquet
                                             0001_part_00.parquet
```

```
                                                0002_part_00.parquet
                                                0003_part_00.parquet
...
```

> **Note**
>
> In einigen Fällen verwendet der UNLOAD-Befehl die INCLUDE-Option, wie in der folgenden SQL-Anweisung gezeigt.
>
> ```
> unload ('select * from lineitem')
> to 's3://mybucket/lineitem/'
> iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
> PARQUET
> PARTITION BY (l_shipdate) INCLUDE;
> ```
>
> In diesen Fällen befindet sich die `l_shipdate`-Spalte auch in den Daten in den Parquet-Dateien. Andernfalls befinden sich die `l_shipdate`-Spaltendaten nicht in den Parquet-Dateien.

Entladen der VENUE-Tabelle in eine JSON-Datei

Im folgenden Beispiel wird die VENUE-Tabelle entladen und die Daten im JSON-Format in `s3://mybucket/unload/` geschrieben.

```
unload ('select * from venue')
to 's3://mybucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
JSON;
```

Im Folgenden sind die Beispielzeilen der VENUE-Tabelle aufgeführt.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-----------
      1 | Pinewood Racetrack         | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Nach dem Entladen in JSON sieht das Format der Datei in etwa wie folgt aus.

```
{"venueid":1,"venuename":"Pinewood
 Racetrack","venuecity":"Akron","venuestate":"OH","venueseats":0}
{"venueid":2,"venuename":"Columbus \"Crew\" Stadium
 ","venuecity":"Columbus","venuestate":"OH","venueseats":0}
{"venueid":4,"venuename":"Community, Ballpark, Arena","venuecity":"Kansas
 City","venuestate":"KS","venueseats":0}
```

Entladen von VENUE in eine CSV-Datei

Im folgenden Beispiel wird die VENUE-Tabelle entladen und die Daten im CSV-Format in geschrieben `s3://mybucket/unload/`.

```
unload ('select * from venue')
to 's3://mybucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
CSV;
```

Angenommen, die Tabelle VENUE enthält die folgenden Zeilen.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-----------
      1 | Pinewood Racetrack         | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Die Entladedatei sieht ähnlich wie folgt aus.

```
1,Pinewood Racetrack,Akron,OH,0
2,"Columbus ""Crew"" Stadium",Columbus,OH,0
4,"Community, Ballpark, Arena",Kansas City,KS,0
```

Entladen von VENUE in eine CSV-Datei mit einem Trennzeichen

Im folgenden Beispiel wird die VENUE-Tabelle entladen und die Daten im CSV-Format mit dem Pipe-Zeichen (|) als Trennzeichen geschrieben. Die entladene Datei wird in `s3://mybucket/unload/` geschrieben. Die VENUE-Tabelle in diesem Beispiel enthält das Pipe-Zeichen im Wert der ersten Zeile (`Pinewood Race|track`). Es tut dies, um zu zeigen, dass der Wert im Ergebnis in doppelte Anführungszeichen eingeschlossen ist. Ein doppeltes Anführungszeichen wird durch ein

doppeltes Anführungszeichen maskiert, und das gesamte Feld ist in doppelte Anführungszeichen eingeschlossen.

```
unload ('select * from venue')
to 's3://mybucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
CSV DELIMITER AS '|';
```

Angenommen, die Tabelle VENUE enthält die folgenden Zeilen.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-------------
      1 | Pinewood Race|track        | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Die Entladedatei sieht ähnlich wie folgt aus.

```
1|"Pinewood Race|track"|Akron|OH|0
2|"Columbus ""Crew"" Stadium"|Columbus|OH|0
4|Community, Ballpark, Arena|Kansas City|KS|0
```

Entladen von VENUE mit einer Manifestdatei

Um eine Manifestdatei zu erstellen, schließen Sie die Option MANIFEST ein. Im folgenden Beispiel werden die Tabelle VENUE entladen und eine Manifestdatei zusammen mit den Datendateien in s3://mybucket/venue_pipe_ geschrieben:

> **Important**
>
> Wenn Sie Dateien mithilfe der Option MANIFEST entladen, sollten Sie die Option MANIFEST mit dem Befehl COPY verwenden, wenn Sie die Dateien laden. Wenn Sie dasselbe Präfix verwenden, um die Dateien zu laden, und die Option MANIFEST nicht angeben, schlägt der Befehl COPY fehl, da er annimmt, dass es sich bei der Manifestdatei um eine Datendatei handelt.

```
unload ('select * from venue')
```

```
to 's3://mybucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole'
manifest;
```

Das Ergebnis sind diese fünf Dateien:

```
s3://mybucket/venue_pipe_0000_part_00
s3://mybucket/venue_pipe_0001_part_00
s3://mybucket/venue_pipe_0002_part_00
s3://mybucket/venue_pipe_0003_part_00
s3://mybucket/venue_pipe_manifest
```

Im Folgenden wird der Inhalt der Manifestdatei gezeigt.

```
{
  "entries": [
    {"url":"s3://mybucket/tickit/venue_0000_part_00"},
    {"url":"s3://mybucket/tickit/venue_0001_part_00"},
    {"url":"s3://mybucket/tickit/venue_0002_part_00"},
    {"url":"s3://mybucket/tickit/venue_0003_part_00"}
  ]
}
```

Entladen von VENUE mit MANIFEST VERBOSE

Bei Angabe der Option MANIFEST VERBOSE enthält die Manifestdatei die folgenden Abschnitte:

- Im Abschnitt `entries` werden der Amazon S3-Pfad, die Dateigröße und die Zeilenanzahl für die einzelnen Dateien aufgeführt.
- Im Abschnitt `schema` werden die Spaltennamen, Datentypen und die Dimension für die einzelnen Spalten aufgeführt.
- Im Abschnitt `meta` wird die gesamte Dateigröße und die Zeilenanzahl für alle Dateien aufgeführt.

Im folgenden Beispiel wird die Tabelle VENUE mithilfe der Option MANIFEST VERBOSE entladen.

```
unload ('select * from venue')
to 's3://mybucket/unload_venue_folder/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest verbose;
```

Im Folgenden wird der Inhalt der Manifestdatei gezeigt.

```
{
  "entries": [
    {"url":"s3://mybucket/venue_pipe_0000_part_00", "meta": { "content_length": 32295,
 "record_count": 10 }},
    {"url":"s3://mybucket/venue_pipe_0001_part_00", "meta": { "content_length": 32771,
 "record_count": 20 }},
    {"url":"s3://mybucket/venue_pipe_0002_part_00", "meta": { "content_length": 32302,
 "record_count": 10 }},
    {"url":"s3://mybucket/venue_pipe_0003_part_00", "meta": { "content_length": 31810,
 "record_count": 15 }}
  ],
  "schema": {
    "elements": [
      {"name": "venueid", "type": { "base": "integer" }},
      {"name": "venuename", "type": { "base": "character varying", 25 }},
      {"name": "venuecity", "type": { "base": "character varying", 25 }},
      {"name": "venuestate", "type": { "base": "character varying", 25 }},
      {"name": "venueseats", "type": { "base": "character varying", 25 }}
    ]
  },
  "meta": {
    "content_length": 129178,
    "record_count": 55
  },
  "author": {
    "name": "Amazon Redshift",
    "version": "1.0.0"
  }
}
```

Entladen von VENUE mit einer Kopfzeile

Das folgende Beispiel entlädt VENUE mit einer Kopfzeile.

```
unload ('select * from venue where venueseats > 75000')
to 's3://mybucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
header
parallel off;
```

Nachfolgend sehen Sie den Inhalt der Ausgabedatei mit einer Kopfzeile.

```
venueid|venuename|venuecity|venuestate|venueseats
6|New York Giants Stadium|East Rutherford|NJ|80242
78|INVESCO Field|Denver|CO|76125
83|FedExField|Landover|MD|91704
79|Arrowhead Stadium|Kansas City|MO|79451
```

Entladen von VENUE zu kleineren Dateien

Die maximal zulässige Dateigröße ist standardmäßig 6,2 GB. Wenn die Größe der Datei mit den entladenen Daten 6,2 GB überschreitet, erstellt UNLOAD eine neue Datei für jedes Datensegment von 6,2 GB. Um kleinere Dateien zu erstellen, verwenden Sie den Parameter MAXFILESIZE. Wenn die Größe der Daten im vorherigen Beispiel ist 20 GB betragen würde, würde der folgende UNLOAD-Befehl 20 Dateien erstellen, jede mit einer Größe von 1 GB.

```
unload ('select * from venue')
to 's3://mybucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
maxfilesize 1 gb;
```

Serielles Entladen von VENUE

Um eine serielle Entladung auszuführen, geben Sie PARALLEL OFF an. UNLOAD schreibt anschließend in jeweils eine einzelne Datei bis zu maximal 6,2 GB pro Datei.

Im folgenden Beispiel werden die Tabelle VENUE entladen und die Daten seriell in geschrieben `s3://mybucket/unload/`.

```
unload ('select * from venue')
to 's3://mybucket/unload/venue_serial_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off;
```

Das Ergebnis ist eine einzelne Datei namens venue_serial_000.

Wenn die Größe der Datei mit den entladenen Daten 6,2 GB überschreitet, erstellt UNLOAD eine neue Datei für jedes Datensegment von 6,2 GB. Im folgenden Beispiel werden die Tabelle LINEORDER entladen und die Daten seriell in geschrieben `s3://mybucket/unload/`.

```
unload ('select * from lineorder')
to 's3://mybucket/unload/lineorder_serial_'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off gzip;
```

Das Ergebnis ist die folgende Reihe von Dateien.

```
lineorder_serial_0000.gz
lineorder_serial_0001.gz
lineorder_serial_0002.gz
lineorder_serial_0003.gz
```

Um die Ausgabedateien besser zu differenzieren, können Sie für den Speicherort ein Präfix verwenden. Im folgenden Beispiel werden die Tabelle VENUE entladen und die Daten in `s3://mybucket/venue_pipe_` geschrieben:

```
unload ('select * from venue')
to 's3://mybucket/unload/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Das Ergebnis dieser vier Dateien ist der Ordner `unload`, erneut unter Annahme von vier Slices.

```
venue_pipe_0000_part_00
venue_pipe_0001_part_00
venue_pipe_0002_part_00
venue_pipe_0003_part_00
```

Laden von VENUE aus entladenen Dateien

Um eine Tabelle aus einem Satz entladener Dateien zu laden, kehren Sie den Befehl einfach unter Verwendung eines COPY-Befehls um. Im folgenden Beispiel werden eine neue Tabelle namens LOADVENUE erstellt und diese Tabelle aus den im vorherigen Beispiel erstellten Datendateien geladen.

```
create table loadvenue (like venue);

copy loadvenue from 's3://mybucket/venue_pipe_' iam_role
 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Wenn Sie die Option MANIFEST verwendet haben, um mit den entladenen Dateien eine Manifestdatei zu erstellen, können Sie die Daten unter Verwendung derselben Manifestdatei laden.

Führen Sie dazu einen COPY-Befehl mit der Option MANIFEST aus. Im folgenden Beispiel werden Daten unter Verwendung einer Manifestdatei geladen.

```
copy loadvenue
from 's3://mybucket/venue_pipe_manifest' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole'
manifest;
```

Entladen von VENUE zu verschlüsselten Dateien

Im folgenden Beispiel wird die VENUE-Tabelle mithilfe eines Schlüssels in eine Reihe verschlüsselter Dateien entladen. AWS KMS Wenn Sie eine Manifestdatei mit der Option ENCRYPTED angeben, wird die Manifestdatei ebenfalls verschlüsselt. Weitere Informationen finden Sie unter Entladen verschlüsselter Datendateien.

```
unload ('select * from venue')
to 's3://mybucket/venue_encrypt_kms'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
kms_key_id '1234abcd-12ab-34cd-56ef-1234567890ab'
manifest
encrypted;
```

Im folgenden Beispiel wird die Tabelle VENUE zu einem Satz verschlüsselter Dateien unter Verwendung eines symmetrischen Root-Schlüssels entladen.

```
unload ('select * from venue')
to 's3://mybucket/venue_encrypt_cmk'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key 'EXAMPLEMASTERKEYtkbjk/OpCwtYSx/M4/t7DMCDIK722'
encrypted;
```

Laden von VENUE aus verschlüsselten Dateien

Um Tabellen aus einem Satz von Dateien zu laden, die unter Verwendung von UNLOAD mit der Option ENCRYPT erstellt wurden, kehren Sie den Prozess einfach unter Verwendung eines COPY-Befehls um. Verwenden Sie die Option ENCRYPTED mit diesem Befehl und geben Sie denselben symmetrischen Root-Schlüssel an, der für den UNLOAD-Befehl verwendet wurde. Im folgenden Beispiel wird die Tabelle LOADVENUE aus den im vorherigen Beispiel erstellten verschlüsselten Datendateien geladen.

```
create table loadvenue (like venue);

copy loadvenue
from 's3://mybucket/venue_encrypt_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key 'EXAMPLEMASTERKEYtkbjk/OpCwtYSx/M4/t7DMCDIK722'
manifest
encrypted;
```

Entladen von VENUE zu einer Datei mit Tabulatoren als Trennzeichen

```
unload ('select venueid, venuename, venueseats from venue')
to 's3://mybucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t';
```

Die Ausgabedatendateien sehen wie folgt aus:

```
1 Toyota Park Bridgeview IL 0
2 Columbus Crew Stadium Columbus OH 0
3 RFK Stadium Washington DC 0
4 CommunityAmerica Ballpark Kansas City KS 0
5 Gillette Stadium Foxborough MA 68756
...
```

Entladen von VENUE zu einer Datei mit festen Spaltenbreiten

```
unload ('select * from venue')
to 's3://mybucket/venue_fw_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth as 'venueid:3,venuename:39,venuecity:16,venuestate:2,venueseats:6';
```

Die Ausgabedatendateien sehen wie folgt aus.

```
1  Toyota Park              Bridgeview  IL0
2  Columbus Crew Stadium    Columbus    OH0
3  RFK Stadium              Washington  DC0
4  CommunityAmerica BallparkKansas City KS0
5  Gillette Stadium         Foxborough  MA68756
...
```

Entladen von VENUE zu einem Satz von GZIP-komprimierten Dateien mit Tabulatoren als Trennzeichen

```
unload ('select * from venue')
to 's3://mybucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t'
gzip;
```

Entladen von VENUE in eine GZIP-komprimierte Textdatei

```
unload ('select * from venue')
to 's3://mybucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
extension 'txt.gz'
gzip;
```

Entladen von Daten, die ein Trennzeichen enthalten

In diesem Beispiel wird die Option ADDQUOTES verwendet, um durch Komma getrennte Daten zu entladen, in denen einige der tatsächlichen Datenfelder ein Komma enthalten.

Erstellen Sie zunächst eine Tabelle, die Anführungszeichen enthält.

```
create table location (id int, location char(64));

insert into location values (1,'Phoenix, AZ'),(2,'San Diego, CA'),(3,'Chicago, IL');
```

Entladen Sie die Daten anschließend unter Verwendung der Option ADDQUOTES.

```
unload ('select id, location from location')
to 's3://mybucket/location_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter ',' addquotes;
```

Die entladenen Daten sehen wie folgt aus:

```
1,"Phoenix, AZ"
2,"San Diego, CA"
3,"Chicago, IL"
...
```

Entladen der Ergebnisse einer Join-Abfrage

Im folgenden Beispiel werden die Ergebnisse einer Join-Abfrage entladen, die eine Fensterfunktion enthält.

```
unload ('select venuecity, venuestate, caldate, pricepaid,
sum(pricepaid) over(partition by venuecity, venuestate
order by caldate rows between 3 preceding and 3 following) as winsum
from sales join date on sales.dateid=date.dateid
join event on event.eventid=sales.eventid
join venue on event.venueid=venue.venueid
order by 1,2')
to 's3://mybucket/tickit/winsum'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Die Ausgabedateien sehen wie folgt aus:

```
Atlanta|GA|2008-01-04|363.00|1362.00
Atlanta|GA|2008-01-05|233.00|2030.00
Atlanta|GA|2008-01-06|310.00|3135.00
Atlanta|GA|2008-01-08|166.00|8338.00
Atlanta|GA|2008-01-11|268.00|7630.00
...
```

Entladen unter Verwendung von NULL AS

UNLOAD gibt Null-Werte standardmäßig als leere Zeichenfolgen zurück. Die folgenden Beispiele zeigen, wie NULL AS verwendet wird, um Null-Werte durch Textzeichenfolgen zu ersetzen.

Hierzu fügen wir der Tabelle VENUE einige Null-Werte hinzu.

```
update venue set venuestate = NULL
where venuecity = 'Cleveland';
```

Wählen Sie aus VENUE aus, wobei VENUESTATE null ist, um zu überprüfen, ob die Spalten NULL enthalten.

```
select * from venue where venuestate is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      22 | Quicken Loans Arena      | Cleveland |            |          0
```

```
     101 | Progressive Field        | Cleveland |            |      43345
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
```

Entladen Sie nun die Tabelle VENUE unter Verwendung der Option NULL AS, um Null-Werte durch die Zeichenfolge zu ersetzen. '`fred`'.

```
unload ('select * from venue')
to 's3://mybucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
null as 'fred';
```

Das folgende Beispiel aus der entladenen Datei zeigt, dass Null-Werte durch ersetzt wurden `fred`. Es stellt sich heraus, dass einige Werte für VENUESEATS ebenfalls null waren und durch ersetzt wurden `fred`. Auch wenn es sich beim Datentyp für VENUESEATS um Ganzzahl handelt, wandelt UNLOAD die Werte in den entladenen Dateien in Text um. Anschließend wandelt COPY sie wieder in Ganzzahlen um. Wenn Sie Daten zu einer Date mit festen Spaltenbreiten entladen, darf die NULL AS-Zeichenfolge nicht länger als die Feldbreite sein.

```
248|Charles Playhouse|Boston|MA|0
251|Paris Hotel|Las Vegas|NV|fred
258|Tropicana Hotel|Las Vegas|NV|fred
300|Kennedy Center Opera House|Washington|DC|0
306|Lyric Opera House|Baltimore|MD|0
308|Metropolitan Opera|New York City|NY|0
  5|Gillette Stadium|Foxborough|MA|5
 22|Quicken Loans Arena|Cleveland|fred|0
101|Progressive Field|Cleveland|fred|43345
...
```

Um eine Tabelle aus den entladenen Dateien zu laden, verwenden Sie einen COPY-Befehl mit derselben NULL AS-Option.

> **Note**
>
> Wenn Sie versuchen, Null-Werte in eine Spalte zu laden, die als NOT NULL definiert ist, schlägt der Befehl COPY fehl.

```
create table loadvenuenulls (like venue);
```

```
copy loadvenuenulls from 's3://mybucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
null as 'fred';
```

Um zu überprüfen, ob die Spalten Null-Werte enthalten und nicht nur leere Zeichenfolgen, wählen Sie aus LOADVENUENULLS aus und filtern nach Null-Werten.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
     251 | Paris Hotel              | Las Vegas | NV         |

...
```

Sie können eine Tabelle entladen, die Null-Werte enthält, indem Sie das NULL AS-Standardverhalten verwenden und anschließend die Daten unter Verwendung des NULL AS-Standardverhaltens wieder in eine Tabelle kopieren. Alle nicht numerischen Felder in der Zieltabelle werden jedoch leere Zeichenfolgen und keine Null-Werte enthalten. Standardmäßig wandelt UNLOAD Null-Werte in leere Zeichenfolgen um (Leerzeichen oder Null-Länge). COPY wandelt leere Zeichenfolgen für numerische Spalten in NULL um, fügt in nichtnumerische Spalten jedoch leere Zeichenfolgen ein. Im folgenden Beispiel wird gezeigt, wie Sie eine UNLOAD-Operation gefolgt von einer COPY-Operation unter Verwendung des NULL AS-Standardverhaltens ausführen.

```
unload ('select * from venue')
to 's3://mybucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' allowoverwrite;

truncate loadvenuenulls;
copy loadvenuenulls from 's3://mybucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

In diesem Fall enthalten nur die Zeilen mit VENUESEATS Null-Werte, wenn Sie nach Null-Werten filtern. Wenn VENUESTATE in der Tabelle (VENUE) Null-Werte enthält, enthält VENUESTATE in der Zieltabelle (LOADVENUENULLS) leere Zeichenfolgen.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
     251 | Paris Hotel              | Las Vegas | NV         |
...
```

Um leere Zeichenfolgen in nichtnumerische Spalten als NULL zu laden, schließen Sie die Optionen EMPTYASNULL oder BLANKSASNULL ein. Sie können beide Optionen gleichzeitig verwenden.

```
unload ('select * from venue')
to 's3://mybucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' allowoverwrite;

truncate loadvenuenulls;
copy loadvenuenulls from 's3://mybucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' EMPTYASNULL;
```

Um zu überprüfen, ob die Spalten NULL enthalten und nicht nur Leerzeichen oder leere Zeichenfolgen, wählen Sie aus LOADVENUENULLS aus und filtern nach Null-Werten.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
     251 | Paris Hotel              | Las Vegas | NV         |
     ...
```

Entladen mit dem Parameter ALLOWOVERWRITE

Standardmäßig überschreibt UNLOAD keine Dateien, die im Ziel-Bucket vorhanden sind. Wenn Sie beispielsweise dieselbe UNLOAD-Anweisung zweimal ausführen, ohne die Dateien im Ziel-Bucket zu ändern, schlägt die zweite UNLOAD-Anweisung fehl. Um die vorhandenen Dateien, einschließlich der Manifestdatei, zu überschreiben, geben Sie die Option ALLOWOVERWRITE an.

```
unload ('select * from venue')
to 's3://mybucket/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest allowoverwrite;
```

Entladen der EVENT-Tabelle mithilfe der Parameter PARALLEL und MANIFEST

Sie können eine Tabelle mit UNLOAD parallel entladen und eine Manifestdatei anlegen. Die Amazon-S3-Datendateien werden alle auf derselben Ebene erstellt und Namen werden mit dem Suffix des Musters `0000_part_00` versehen. Die Manifestdatei befindet sich auf derselben Ordnerebene wie die Datendateien und ist mit dem Suffixtext `manifest` versehen. Die folgende SQL-Anweisung entlädt die EVENT-Tabelle und erstellt Dateien mit dem Basisnamen `parallel`.

```
unload ('select * from mytickit1.event')
to 's3://my-s3-bucket-name/parallel'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
parallel on
manifest;
```

Die Amazon-S3-Dateiliste sieht etwa wie folgt aus.

```
 Name                        Last modified                         Size
 parallel0000_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 52.1 KB
 parallel0001_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 53.4 KB
 parallel0002_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 52.1 KB
 parallel0003_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 51.1 KB
 parallel0004_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 54.6 KB
 parallel0005_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 53.4 KB
 parallel0006_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 54.1 KB
 parallel0007_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 55.9 KB
 parallelmanifest       -   August 2, 2023, 14:54:39 (UTC-07:00) 886.0 B
```

Der Inhalt der Datei `parallelmanifest` ähnelt dem folgenden Beispiel.

```
{
  "entries": [
    {"url":"s3://my-s3-bucket-name/parallel0000_part_00", "meta": { "content_length":
 53316 }},
```

```
    {"url":"s3://my-s3-bucket-name/parallel0001_part_00", "meta": { "content_length":
 54704 }},
    {"url":"s3://my-s3-bucket-name/parallel0002_part_00", "meta": { "content_length":
 53326 }},
    {"url":"s3://my-s3-bucket-name/parallel0003_part_00", "meta": { "content_length":
 52356 }},
    {"url":"s3://my-s3-bucket-name/parallel0004_part_00", "meta": { "content_length":
 55933 }},
    {"url":"s3://my-s3-bucket-name/parallel0005_part_00", "meta": { "content_length":
 54648 }},
    {"url":"s3://my-s3-bucket-name/parallel0006_part_00", "meta": { "content_length":
 55436 }},
    {"url":"s3://my-s3-bucket-name/parallel0007_part_00", "meta": { "content_length":
 57272 }}
  ]
}
```

Entladen der EVENT-Tabelle mithilfe der Parameter PARALLEL OFF und MANIFEST

Sie können eine Tabelle mit UNLOAD seriell entladen (PARALLEL OFF) und eine Manifestdatei anlegen. Die Amazon-S3-Datendateien werden alle auf derselben Ebene erstellt und Namen werden mit dem Suffix des Musters `0000` versehen. Die Manifestdatei befindet sich auf derselben Ordnerebene wie die Datendateien und ist mit dem Suffixtext `manifest` versehen.

```
unload ('select * from mytickit1.event')
to 's3://my-s3-bucket-name/serial'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
parallel off
manifest;
```

Die Amazon-S3-Dateiliste sieht etwa wie folgt aus.

```
 Name                    Last modified                         Size
 serial0000          -   August 2, 2023, 15:54:39 (UTC-07:00) 426.7 KB
 serialmanifest      -   August 2, 2023, 15:54:39 (UTC-07:00) 120.0 B
```

Der Inhalt der Datei `serialmanifest` ähnelt dem folgenden Beispiel.

```
{
  "entries": [
    {"url":"s3://my-s3-bucket-name/serial000", "meta": { "content_length": 436991 }}
  ]
}
```

Entladen der EVENT-Tabelle mithilfe der Parameter PARTITION BY und MANIFEST

Sie können eine Tabelle mit UNLOAD partitionsweise entladen und eine Manifestdatei anlegen. In Amazon S3 wird ein neuer Ordner mit untergeordneten Partitionsordnern erstellt und die Datendateien in den untergeordneten Ordnern haben ein Namensmuster ähnlich wie `0000_par_00`. Die Manifestdatei befindet sich auf derselben Ordnerebene wie die untergeordneten Ordner mit dem Namen `manifest`.

```
unload ('select * from mytickit1.event')
to 's3://my-s3-bucket-name/partition'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
partition by (eventname)
manifest;
```

Die Amazon-S3-Dateiliste sieht etwa wie folgt aus.

```
Name                    Type     Last modified                     Size

partition            Folder
```

Im Ordner `partition` befinden sich untergeordnete Ordner mit dem Partitionsnamen und der Manifestdatei. Im Folgenden sehen Sie das Ende der Liste der Ordner im Ordner `partition`, ähnlich wie folgt.

```
Name                    Type      Last modified                      Size

...
eventname=Zucchero/    Folder
eventname=Zumanity/    Folder
eventname=ZZ Top/      Folder
manifest               -     August 2, 2023, 15:54:39 (UTC-07:00) 467.6 KB
```

Im Ordner `eventname=Zucchero/` befinden sich die Datendateien, ähnlich wie folgt.

```
 Name               Last modified                          Size
 0000_part_00 -   August 2, 2023, 15:59:19 (UTC-07:00) 70.0 B
 0001_part_00 -   August 2, 2023, 15:59:16 (UTC-07:00) 106.0 B
 0002_part_00 -   August 2, 2023, 15:59:15 (UTC-07:00) 70.0 B
 0004_part_00 -   August 2, 2023, 15:59:17 (UTC-07:00) 141.0 B
 0006_part_00 -   August 2, 2023, 15:59:16 (UTC-07:00) 35.0 B
 0007_part_00 -   August 2, 2023, 15:59:19 (UTC-07:00) 108.0 B
```

Das Ende der Datei `manifest` ähnelt dem folgenden Beispiel.

```
{
  "entries": [
    ...
    {"url":"s3://my-s3-bucket-name/partition/eventname=Zucchero/0007_part_00", "meta":
 { "content_length": 108 }},
    {"url":"s3://my-s3-bucket-name/partition/eventname=Zumanity/0007_part_00", "meta":
 { "content_length": 72 }}
  ]
}
```

Entladen der EVENT-Tabelle mithilfe der Parameter MAXFILESIZE, ROWGROUPSIZE und MANIFEST

Sie können eine Tabelle mit UNLOAD parallel entladen und eine Manifestdatei anlegen. Die Amazon-S3-Datendateien werden alle auf derselben Ebene erstellt und Namen werden mit dem Suffix des Musters `0000_part_00` versehen. Die generierten Parquet-Datendateien sind auf 256 MB und die Zeilengruppengröße ist auf 128 MB begrenzt. Die Manifestdatei befindet sich auf derselben Ordnerebene wie die Datendateien und ist mit dem Suffix `manifest` versehen.

```
unload ('select * from mytickit1.event')
to 's3://my-s3-bucket-name/eventsize'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
maxfilesize 256 MB
rowgroupsize 128 MB
parallel on
parquet
manifest;
```

Die Amazon-S3-Dateiliste sieht etwa wie folgt aus.

```
 Name                            Type      Last modified                          Size
 eventsize0000_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.5 KB
 eventsize0001_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.8 KB
 eventsize0002_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.4 KB
 eventsize0003_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.0 KB
 eventsize0004_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.3 KB
 eventsize0005_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.8 KB
 eventsize0006_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.0 KB
 eventsize0007_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.6 KB
 eventsizemanifest                  -       August 2, 2023, 17:35:21 (UTC-07:00) 958.0 B
```

Der Inhalt der Datei `eventsizemanifest` ähnelt dem folgenden Beispiel.

```
{
  "entries": [
    {"url":"s3://my-s3-bucket-name/eventsize0000_part_00.parquet", "meta":
 { "content_length": 25130 }},
    {"url":"s3://my-s3-bucket-name/eventsize0001_part_00.parquet", "meta":
 { "content_length": 25428 }},
    {"url":"s3://my-s3-bucket-name/eventsize0002_part_00.parquet", "meta":
 { "content_length": 25025 }},
    {"url":"s3://my-s3-bucket-name/eventsize0003_part_00.parquet", "meta":
 { "content_length": 24554 }},
    {"url":"s3://my-s3-bucket-name/eventsize0004_part_00.parquet", "meta":
 { "content_length": 25918 }},
    {"url":"s3://my-s3-bucket-name/eventsize0005_part_00.parquet", "meta":
 { "content_length": 25362 }},
    {"url":"s3://my-s3-bucket-name/eventsize0006_part_00.parquet", "meta":
 { "content_length": 25647 }},
    {"url":"s3://my-s3-bucket-name/eventsize0007_part_00.parquet", "meta":
 { "content_length": 26256 }}
  ]
}
```

UPDATE

Themen

Aktualisiert Werte in einer oder mehreren Tabellenspalten, wenn eine Bedingung erfüllt wird.

> **Note**
>
> Die maximal zulässige Größe für eine einzelne SQL-Anweisung ist 16 MB.

Syntax

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
            UPDATE table_name [ [ AS ] alias ] SET column = { expression | DEFAULT }
 [,...]

[ FROM fromlist ]
[ WHERE condition ]
```

Parameter

WITH-Klausel

Optionale Klausel, die eine oder mehrere allgemeine Tabellenausdrücke (CET) angibt. Siehe WITH-Klausel.

table_name

Eine temporäre oder persistente Tabelle. Nur der Besitzer der Tabelle oder ein Benutzer mit dem Recht UPDATE für die Tabelle können Zeilen aktualisieren. Wenn Sie in einem Ausdruck oder einer Bedingung die FROM-Klausel verwenden oder aus Tabellen auswählen, müssen Sie das SELECT-Recht für diese Tabellen besitzen. Sie können der Tabelle hier keinen Aliasnamen geben. Sie können jedoch in der FROM-Klausel einen Alias angeben.

> **Note**
>
> Externe Tabellen von Amazon Redshift Spectrum sind schreibgeschützt. Sie können eine externe Tabelle nicht aktualisieren.

alias

Der temporäre alternative Name für eine Zieltabelle. Aliase sind optional. Das Schlüsselwort AS ist stets optional.

SET column =

Eine oder mehrere Spalten, die Sie ändern möchten. Spalten, die nicht aufgelistet sind, behalten ihre aktuellen Werte bei. Schließen Sie den Tabellennamen nicht in der Spezifikation einer Zielspalte ein. Beispielsweise ist `UPDATE tab SET tab.col = 1` nicht gültig.

expression

Ein Ausdruck der den neuen Wert für die angegebene Spalte definiert.

DEFAULT

Aktualisiert die Spalte mit dem Standardwert, der der Spalte in der Anweisung CREATE TABLE zugewiesen wurde.

FROM tablelist

Sie können eine Tabelle aktualisieren, indem Sie Informationen in anderen Tabellen referenzieren. Listen Sie diese anderen Tabellen in der FROM-Klausel auf oder verwenden Sie eine Unterabfrage als Teil der WHERE-Bedingung. In der FROM-Klausel aufgelistete Tabellen können Aliase haben. Verwenden Sie einen Alias, wenn Sie die Zieltabelle der UPDATE-Anweisung in die Liste einschließen müssen.

WHERE condition

Optionale Klausel, die Aktualisierungen für Zeilen einschränkt, die einer Bedingung entsprechen. Wenn die Bedingung `true` zurückgibt, werden die angegebenen SET-Spalten aktualisiert. Bei der Bedingung kann es sich um ein einfaches Prädikat für eine Spalte oder eine Bedingung auf der Basis des Ergebnisses einer Unterabfrage handeln.

Sie können jede Tabelle in der Unterabfrage nennen, einschließlich der Zieltabelle für UPDATE.

Nutzungshinweise

Nach dem Aktualisieren einer großen Zahl von Zeilen in einer Tabelle:

- Führen Sie eine Vacuum-Operation für die Tabelle durch, um Speicherplatz zurückzugewinnen und die Zeilen neu zu sortieren.
- Analysieren Sie die Tabelle, um Statistiken für den Abfrageplaner zu aktualisieren.

Die FROM-Klausel einer UPDATE-Anweisung unterstützt keine Joins nach links oder rechts oder vollständige externe Joins. Diese geben den folgenden Fehler zurück:

```
ERROR: Target table must be part of an equijoin predicate
```

Wenn Sie einen externen Join angeben müssen, verwenden Sie eine Unterabfrage in der WHERE-Klausel der UPDATE-Anweisung.

Wenn Ihre UPDATE-Anweisung einen Selbst-Join für die Zieltabelle erfordert, müssen Sie die Join-Bedingung und die Kriterien für die WHERE-Klausel angeben, die Zeilen für die Update-Operation qualifizieren. Wenn für die Zieltabelle ein Join mit sich selbst oder einer anderen Tabelle ausgeführt wird, stellt es im Allgemeinen eine bewährte Methode dar, eine Unterabfrage zu verwenden, die die Join-Bedingungen klar von den Kriterien trennt, die Zeilen für Aktualisierungen qualifizieren.

UPDATE-Abfragen mit mehreren Übereinstimmungen pro Zeile führen zu einem Fehler, wenn der Konfigurationsparameter `error_on_nondeterministic_update` auf true gesetzt ist. Weitere Informationen finden Sie unter error_on_nondeterministic_update.

Sie können eine GENERATED BY DEFAULT AS IDENTITY-Spalte aktualisieren. Sie können als GENERATED BY DEFAULT AS IDENTITY-Spalten mit von Ihnen angegebenen Werten aktualisieren. Weitere Informationen finden Sie unter GENERATED BY DEFAULT AS IDENTITY.

Beispiele für UPDATE-Anweisungen

Weitere Hinweise zu den Tabellen, die in den folgenden Beispielen verwendet werden, finden Sie unter Beispieldatenbank.

Die Tabelle CATEGORY in der Datenbank TICKIT enthält die folgenden Zeilen:

```
+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
```

```
+-------+----------+-----------+-------------------------------------------+
| 5     | Sports   | MLS       | Major League Soccer                       |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
| 1     | Sports   | MLB       | Major League Baseball                     |
| 6     | Shows    | Musicals  | Musical theatre                           |
| 3     | Sports   | NFL       | National Football League                  |
| 8     | Shows    | Opera     | All opera and light opera                 |
| 2     | Sports   | NHL       | National Hockey League                    |
| 9     | Concerts | Pop       | All rock and pop music concerts           |
| 4     | Sports   | NBA       | National Basketball Association           |
| 7     | Shows    | Plays     | All non-musical theatre                   |
| 10    | Concerts | Jazz      | All jazz singers and bands                |
+-------+----------+-----------+-------------------------------------------+
```

Aktualisieren einer Tabelle auf der Basis eines Bereichs von Werten

Aktualisieren Sie die Spalte CATGROUP auf der Basis eines Bereichs von Werten in der Spalte CATID.

```
UPDATE category
SET catgroup='Theatre'
WHERE catid BETWEEN 6 AND 8;

SELECT * FROM category
WHERE catid BETWEEN 6 AND 8;

+-------+----------+----------+---------------------------+
| catid | catgroup | catname  |          catdesc          |
+-------+----------+----------+---------------------------+
| 6     | Theatre  | Musicals | Musical theatre           |
| 7     | Theatre  | Plays    | All non-musical theatre   |
| 8     | Theatre  | Opera    | All opera and light opera |
+-------+----------+----------+---------------------------+
```

Aktualisieren einer Tabelle auf der Basis eines aktuellen Werts

Aktualisieren Sie die Spalten CATNAME und CATDESC auf der Basis ihres aktuellen CATGROUP-Werts:

```
UPDATE category
SET catdesc=default, catname='Shows'
WHERE catgroup='Theatre';
```

```
SELECT * FROM category
WHERE catname='Shows';

+-------+----------+---------+---------+
| catid | catgroup | catname | catdesc |
+-------+----------+---------+---------+
| 6     | Theatre  | Shows   | NULL    |
| 7     | Theatre  | Shows   | NULL    |
| 8     | Theatre  | Shows   | NULL    |
+-------+----------+---------+---------+)
```

In diesem Fall wurde die Spalte CATDESC auf null festgelegt, da zum Zeitpunkt der Erstellung der Tabelle kein Standardwert definiert wurde.

Führen Sie die folgenden Befehle aus, um die Daten der Tabelle CATEGORY auf die ursprünglichen Werte zurückzusetzen:

```
TRUNCATE category;

COPY category
FROM 's3://redshift-downloads/tickit/category_pipe.txt'
DELIMITER '|'
IGNOREHEADER 1
REGION 'us-east-1'
IAM_ROLE default;
```

Aktualisieren einer Tabelle auf der Basis des Ergebnisses einer Unterabfrage in der WHERE-Klausel

Aktualisieren Sie die Tabelle CATEGORY auf der Basis des Ergebnisses einer Unterabfrage in der WHERE-Klausel:

```
UPDATE category
SET catdesc='Broadway Musical'
WHERE category.catid IN
(SELECT category.catid FROM category
JOIN event ON category.catid = event.catid
JOIN venue ON venue.venueid = event.venueid
JOIN sales ON sales.eventid = event.eventid
WHERE venuecity='New York City' AND catname='Musicals');
```

Zeigen Sie die aktualisierte Tabelle an:

```
SELECT * FROM category ORDER BY catid;

+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
+-------+----------+-----------+--------------------------------------------+
| 2     | Sports   | NHL       | National Hockey League                     |
| 3     | Sports   | NFL       | National Football League                   |
| 4     | Sports   | NBA       | National Basketball Association            |
| 5     | Sports   | MLS       | Major League Soccer                        |
| 6     | Shows    | Musicals  | Broadway Musical                           |
| 7     | Shows    | Plays     | All non-musical theatre                    |
| 8     | Shows    | Opera     | All opera and light opera                  |
| 9     | Concerts | Pop       | All rock and pop music concerts            |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
+-------+----------+-----------+--------------------------------------------+
```

Aktualisieren einer Tabelle auf der Basis des Ergebnisses einer Unterabfrage in der WITH-Klausel

Verwenden Sie das folgende Beispiel, um die CATEGORY-Tabelle mithilfe der WITH-Klausel zu aktualisieren.

```
WITH u1 as (SELECT catid FROM event ORDER BY catid DESC LIMIT 1)
UPDATE category SET catid='200' FROM u1 WHERE u1.catid=category.catid;

SELECT * FROM category ORDER BY catid DESC LIMIT 1;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
| 200   | Concerts | Pop     | All rock and pop music concerts |
+-------+----------+---------+---------------------------------+
```

Aktualisieren einer Tabelle auf der Basis des Ergebnisses einer Join-Bedingung

Aktualisieren Sie die ursprünglichen 11 Zeilen in der Tabelle CATEGORY auf der Basis der entsprechenden CATID-Zeilen in der Tabelle EVENT:

```
UPDATE category SET catid=100
FROM event
WHERE event.catid=category.catid;
```

```
SELECT * FROM category ORDER BY catid;

+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
+-------+----------+-----------+--------------------------------------------+
| 2     | Sports   | NHL       | National Hockey League                     |
| 3     | Sports   | NFL       | National Football League                   |
| 4     | Sports   | NBA       | National Basketball Association            |
| 5     | Sports   | MLS       | Major League Soccer                        |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
| 100   | Concerts | Pop       | All rock and pop music concerts            |
| 100   | Shows    | Plays     | All non-musical theatre                    |
| 100   | Shows    | Opera     | All opera and light opera                  |
| 100   | Shows    | Musicals  | Broadway Musical                           |
+-------+----------+-----------+--------------------------------------------+
```

Beachten Sie, dass die Tabelle EVENT in der FROM-Klausel aufgelistet wird und die Join-Bedingung für die Zieltabelle in der WHERE-Klausel definiert ist. Für die Aktualisierung wurden nur vier Zeilen qualifiziert. Dies sind die vier Zeilen, deren CATID-Werte ursprünglich 6, 7, 8 und 9 waren. Nur diese vier Kategorien werden in der Tabelle EVENT repräsentiert:

```
SELECT DISTINCT catid FROM event;

+-------+
| catid |
+-------+
| 6     |
| 7     |
| 8     |
| 9     |
+-------+
```

Aktualisieren Sie die ursprünglichen 11 Zeilen in der Tabelle CATEGORY, indem Sie das vorherige Beispiel erweitern und der WHERE-Klausel eine andere Bedingung hinzufügen. Aufgrund der Einschränkung für die Spalte CATGROUP ist nur eine Zeile für die Aktualisierung qualifiziert (auch wenn vier Zeilen für den Join qualifiziert sind).

```
UPDATE category SET catid=100
FROM event
WHERE event.catid=category.catid
AND catgroup='Concerts';
```

```
SELECT * FROM category WHERE catid=100;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
| 100   | Concerts | Pop     | All rock and pop music concerts |
+-------+----------+---------+---------------------------------+
```

Eine alternative Möglichkeit, dieses Beispiel zu schreiben, ist:

```
UPDATE category SET catid=100
FROM event JOIN category cat ON event.catid=cat.catid
WHERE cat.catgroup='Concerts';
```

Der Vorteil dieses Ansatzes besteht darin, dass die Join-Kriterien klar von allen anderen Kriterien getrennt sind, die Zeilen für die Aktualisierung qualifizieren. Beachten Sie die Verwendung des Alias CAT für die Tabelle CATEGORY in der FROM-Klausel.

Aktualisierungen mit externen Joins in der FROM-Klausel

Im vorherigen Beispiel wurde ein interner Join gezeigt, der in der FROM-Klausel einer UPDATE-Anweisung angegeben ist. Das folgende Beispiel gibt einen Fehler zurück da die FROM-Klausel keine externen Joins für die Zieltabelle unterstützt:

```
UPDATE category SET catid=100
FROM event LEFT JOIN category cat ON event.catid=cat.catid
WHERE cat.catgroup='Concerts';
ERROR:  Target table must be part of an equijoin predicate
```

Wenn Sie für die UPDATE-Anweisung einen externen Join angeben müssen, können Sie die Syntax für externe Joins in eine Unterabfrage verschieben:

```
UPDATE category SET catid=100
FROM
(SELECT event.catid FROM event LEFT JOIN category cat ON event.catid=cat.catid)
 eventcat
WHERE category.catid=eventcat.catid
AND catgroup='Concerts';
```

Updates mit Spalten aus einer anderen Tabelle in der SET-Klausel

Verwenden Sie das folgende Beispiel, um die listing-Tabelle in der TICKIT-Beispieldatenbank mit Werten aus der sales-Tabelle zu aktualisieren.

```
SELECT listid, numtickets FROM listing WHERE sellerid = 1 ORDER BY 1 ASC LIMIT 5;

+--------+------------+
| listid | numtickets |
+--------+------------+
| 100423 | 4          |
| 108334 | 24         |
| 117150 | 4          |
| 135915 | 20         |
| 205927 | 6          |
+--------+------------+

UPDATE listing
SET numtickets = sales.sellerid
FROM sales
WHERE sales.sellerid = 1 AND listing.sellerid = sales.sellerid;

SELECT listid, numtickets FROM listing WHERE sellerid = 1 ORDER BY 1 ASC LIMIT 5;

+--------+------------+
| listid | numtickets |
+--------+------------+
| 100423 | 1          |
| 108334 | 1          |
| 117150 | 1          |
| 135915 | 1          |
| 205927 | 1          |
+--------+------------+
```

VACUUM

Sortiert Zeilen neu und gewinnt Platz in einer bestimmten Tabelle oder in allen Tabellen der aktuellen Datenbank zurück.

> **Note**
>
> Nur Benutzer mit den erforderlichen Tabellenberechtigungen können eine Tabelle effektiv bereinigen. Wenn VACUUM ohne die notwendigen Tabellenberechtigungen ausgeführt, wird die Operation zwar erfolgreich abgeschlossen, hat jedoch keine Wirkung. Eine Liste der gültigen Tabellenberechtigungen zum effektiven Ausführen von VACUUM finden Sie im folgenden Abschnitt zu erforderlichen Berechtigungen..

Amazon Redshift sortiert im Hintergrund automatisch die Daten und führt VACUUM DELETE aus. Dadurch entfällt die Notwendigkeit, den Befehl VACUUM auszuführen. Weitere Informationen finden Sie unter Bereinigen von Tabellen.

Standardmäßig überspringt VACUUM die Sortierungsphase für alle Tabellen, in denen mehr als 95 Prozent der Tabellenzeilen bereits sortiert sind. Das Überspringen der Sortierungsphase kann die Leistung von VACUUM deutlich verbessern. Um den Standardschwellenwert für die Sortierung oder Löschung für eine einzelne Tabelle zu ändern, verwenden Sie den Tabellennamen und den Parameter TO threshold PERCENT, wenn Sie VACUUM ausführen.

Benutzer können auf Tabellen zugreifen, während sie bereinigt werden. Sie können Abfragen und Schreiboperationen ausführen, während eine Tabelle bereinigt wird. Wenn jedoch Data Manipulation Language (DML)-Befehle und eine Bereinigung gleichzeitig ausgeführt werden, dauern beide Vorgänge möglicherweise länger. Wenn Sie während einer Bereinigung UPDATE- und DELETE-Anweisungen ausführen, wird die Systemleistung möglicherweise reduziert. VACUUM DELETE blockiert vorübergehend Aktualisierungs- und Löschoperationen.

Amazon Redshift führt automatisch eine DELETE ONLY-Bereinigung im Hintergrund aus. Die automatische Bereinigungsoperation wird angehalten, wenn die Benutzer Data Definition Language (DDL)-Operationen ausführen, wie etwa ALTER TABLE.

> **Note**
>
> Syntax und Verhalten des Amazon-Redshift-Befehls VACUUM unterscheiden sich wesentlich von Syntax und Verhalten der PostgreSQL-Operation VACUUM. Beispielsweise ist die standardmäßige VACUUM-Operation in Amazon Redshift VACUUM FULL, die Festplattenspeicher zurückgewinnt und alle Zeilen neu sortiert. Die VACUUM-Operation in PostgreSQL gewinnt hingegen nur Festplattenplatz zurück und stellt ihn zur Wiederverwendung bereit.

Weitere Informationen finden Sie unter Bereinigen von Tabellen.

Erforderliche Berechtigungen

Für VACUUM sind folgende Berechtigungen erforderlich:

- Superuser
- Benutzer mit der Berechtigung VACUUM
- Tabellenbesitzer
- Datenbankbesitzer, für den die Tabelle freigegeben wird.

Syntax

```
VACUUM [ FULL | SORT ONLY | DELETE ONLY | REINDEX | RECLUSTER ]
[ [ table_name ] [ TO threshold PERCENT ] [ BOOST ] ]
```

Parameter

FULL

Sortiert die angegebene Tabelle (oder alle Tabellen in der aktuellen Datenbank) und gewinnt Festplattenplatz zurück, der von Zeilen besetzt wird, die von vorherigen UPDATE- und DELETE-Operationen zur Löschung markiert wurden. Der Standardwert ist VACUUM FULL.

Eine vollständige Bereinigung führt keine Neuindizierung überlappender Tabellen aus. Um überlappende Tabellen neu zu indizieren und anschließend vollständig zu bereinigen, verwenden Sie die Option VACUUM REINDEX.

Standardmäßig überspringt VACUUM FULL die Sortierphase für alle Tabellen, die bereits zu mindestens 95 Prozent sortiert sind. Wenn VACUUM die Sortierphase überspringen kann, führt die Operation eine DELETE ONLY-Operation aus und gewinnt während der Löschphase Festplattenplatz zurück, wenn mindestens 95 Prozent der verbleibenden Zeilen nicht zur Löschung markiert sind.

Wenn der Schwellenwert für die Sortierung nicht erreicht wird (wenn beispielsweise 90 Prozent der Zeilen sortiert sind) und VACUUM eine vollständige Sortierung ausführt, wird auch eine vollständige Löschoperation ausgeführt, so dass der Festplattenplatz von 100 Prozent der gelöschten Zeilen zurückgewonnen wird.

Sie können den Standardschwellenwert für die Bereinigung nur für einzelne Tabellen ändern. Um den Standardschwellenwert für eine Bereinigung für eine einzelne Tabelle zu ändern, verwenden Sie den Tabellennamen und den Parameter TO threshold PERCENT.

SORT ONLY

Sortiert die angegebene Tabelle (oder alle Tabellen in der aktuellen Datenbank), ohne den Festplattenplatz zurückzugewinnen, der von gelöschten Zeilen freigegeben wird. Diese Option ist nützlich, wenn das Wiederherstellen von Speicherplatz nicht wichtig ist, aber das erneute Sortieren neuer Zeilen von Bedeutung ist. Eine Bereinigung mit SORT ONLY reduziert die Zeit, die für Bereinigungsoperationen benötigt wird, wenn der nicht sortierte Bereich keine große Zahl gelöschter Zeilen enthält und nicht den gesamten sortierten Bereich umfasst. Für Anwendungen, für die es keine Einschränkungen in Bezug auf den Festplattenplatz gibt, die aber von Abfrageoptimierungen abhängig sind, für die sortierte Tabellenzeilen Voraussetzung sind, kann diese Art von Bereinigung nützlich sein.

Standardmäßig überspringt VACUUM SORT ONLY alle Tabellen, die bereits zu mindestens 95 Prozent sortiert sind. Um den Standardschwellenwert für die Sortierung für eine einzelne Tabelle zu ändern, verwenden Sie den Tabellennamen und den Parameter TO threshold PERCENT, wenn Sie VACUUM ausführen.

DELETE ONLY

Amazon Redshift führt automatisch eine DELETE ONLY-Bereinigung im Hintergrund aus, sodass Sie nur selten (wenn überhaupt) eine DELETE ONLY-Bereinigung ausführen müssen.

VACUUM DELETE gewinnt Festplattenplatz zurück, der von Zeilen besetzt wird, die von vorherigen UPDATE- und DELETE-Operationen zur Löschung markiert wurden, und macht die Tabelle kompakter, um den verbrauchten Festplattenplatz freizugeben. Eine Bereinigungsoperation mit DELETE ONLY sortiert die Tabellendaten nicht.

Diese Option reduziert den Zeitaufwand für Vacuum-Operationen, wenn das Wiederherstellen von Festplattenspeicher wichtig ist, aber das Neusortieren neuer Zeilen nicht wichtig ist. Diese Option kann auch nützlich sein, wenn Ihre Abfrageleistung bereits optimal ist. Eine Neusortierung von Zeilen zur Optimierung der Abfrageleistung ist nicht erforderlich.

Standardmäßig gewinnt VACUUM DELETE ONLY Festplattenplatz zurück, wenn mindestens 95 Prozent der verbleibenden Zeilen nicht zur Löschung markiert sind. Um den Standardschwellenwert für die Löschung für eine einzelne Tabelle zu ändern, verwenden Sie den Tabellennamen und den Parameter TO threshold PERCENT, wenn Sie VACUUM ausführen.

Einige Operationen, wie `ALTER TABLE APPEND`, können dazu führen, dass Tabellen fragmentiert werden. Wenn Sie die `DELETE ONLY`-Klausel verwenden, gibt die Bereinigungsoperation Speicherplatz von fragmentierten Tabellen frei. Derselbe Schwellenwert von 95 Prozent gilt für die Defragmentierungsoperation.

REINDEX tablename

Analysiert die Verteilung der Werte in überlappenden Sortierschlüsselspalten und führt anschließend eine vollständige VACUUM-Operation aus. Wenn REINDEX verwendet wird, ist ein Tabellenname erforderlich.

VACUUM REINDEX benötigt deutlich mehr Zeit als VACUUM FULL, da ein zusätzlicher Schritt zur Analyse der überlappenden Sortierschlüssel ausgeführt wird. Die Sortier- und Zusammenführungsoperationen für überlappende Tabellen kann länger dauern, weil die überlappende Sortierung möglicherweise mehr Zeilen als eine zusammengesetzte Sortierung neu anordnen muss.

Wenn eine VACUUM REINDEX-Operation beendet wird, bevor sie abgeschlossen ist, setzt die nächste VACUUM-Operation die Neuindizierungsoperation fort, bevor eine vollständige Bereinigung ausgeführt wird.

VACUUM REINDEX wird bei Verwendung von TO threshold PERCENT nicht unterstützt.

table_name

Der Name der zu bereinigenden Tabelle. Wenn Sie keinen Tabellennamen angeben, wird die Bereinigungsoperation auf alle Tabellen in der aktuellen Datenbank angewendet. Sie können jede permanente oder temporäre Tabelle angeben, die von Benutzern erstellt wurde. Der Befehl gilt nicht für andere Objekte wie Ansichten und Systemtabellen.

Wenn Sie den Parameter TO threshold PERCENT einschließen, ist ein Tabellenname erforderlich.

RECLUSTER tablename

Sortiert die Teile der Tabelle, die nicht sortiert sind. Teile der Tabelle, die bereits nach automatischer Tabellensortierung sortiert sind, bleiben intakt. Mit diesem Befehl werden die neu sortierten Daten nicht mit dem sortierten Bereich zusammengeführt. Es wird auch nicht der gesamte Speicherplatz zurückgewonnen, der zum Löschen markiert ist. Nach Beendigung dieses Befehls wird die Tabelle möglicherweise nicht vollständig sortiert angezeigt, was durch das Feld `unsorted` in SVV_TABLE_INFO angezeigt wird.

Wir empfehlen die Verwendung von VACUUM RECLUSTER für große Tabellen mit häufigen Zugriffen und Abfragen, die nur auf die neuesten Daten zugreifen.

VACUUM RECLUSTER wird bei Verwendung von TO threshold PERCENT nicht unterstützt. Wenn RECLUSTER verwendet wird, ist ein Tabellenname erforderlich.

VACUUM RECLUSTER wird bei Tabellen mit verschachtelten Sortierschlüsseln und Tabellen mit dem Verteilungsstil ALL nicht unterstützt.

table_name

Der Name der zu bereinigenden Tabelle. Sie können jede permanente oder temporäre Tabelle angeben, die von Benutzern erstellt wurde. Der Befehl gilt nicht für andere Objekte wie Ansichten und Systemtabellen.

TO threshold PERCENT

Eine Klausel, die den Schwellenwert, oberhalb dessen VACUUM die Sortierphase überspringt, und den Zielschwellenwert für die Rückgewinnung von Festplattenplatz während der Löschphase angibt. Der Sortierschwellenwert ist der Prozentsatz der Gesamtzahl der Zeilen, die sich bereits in der richtigen Sortierreihenfolge für die angegebene Tabelle befinden, bevor die Bereinigung gestartet wird. Der Löschschwellenwert ist der Mindestprozentsatz der Gesamtzahl der Zeilen, die nach der Bereinigung nicht zur Löschung markiert sind.

Da VACUUM die Zeilen nur dann neu sortiert, wenn der Prozentsatz der sortierten Zeilen in einer Tabelle kleiner als der Sortierschwellenwert ist, kann Amazon Redshift die VACUUM-Zeiten oft deutlich reduzieren. Wenn VACUUM nicht der Einschränkung unterliegt, den Festplattenplatz von 100 Prozent der zur Löschung markierten Zeilen zurückzugewinnen, kann die Operation häufig das Neuschreiben von Blöcken überspringen, die nur einige wenige zur Löschung markierte Zeilen enthalten.

Wenn Sie beispielsweise 75 für den Schwellenwert angeben, überspringt VACUUM die Sortierphase, wenn sich mindestens 75 % der Tabellenzeilen bereits in der richtigen Sortierreihenfolge befinden. In Bezug auf die Löschphase gewinnt VACUUM Festplattenplatz zurück, wenn nach der Bereinigung mindestens 75 Prozent der Tabellenzeilen nicht zur Löschung markiert sind. Der Schwellenwert muss eine Ganzzahl zwischen 0 und 100 sein. Der Standardwert ist 95. Wenn Sie den Wert als 100 angeben, sortiert VACUUM die Tabelle immer, wenn sie nicht bereits vollständig sortiert ist, und gewinnt Festplattenplatz für alle Zeilen zurück, die zur Löschung markiert sind. Wenn Sie den Wert als 0 angeben, sortiert VACUUM die Tabelle niemals und gewinnt niemals Festplattenplatz zurück.

Wenn Sie den Parameter TO threshold PERCENT einschließen, müssen Sie auch einen Tabellennamen angeben. Wenn kein Tabellenname angegeben wird, ist VACUUM erfolglos.

Sie können den Parameter TO threshold PERCENT nicht mit REINDEX verwenden.

BOOST

Führt den Befehl VACUUM mit zusätzlichen Ressourcen wie Arbeitsspeicher und Festplattenspeicher aus, sobald diese verfügbar sind. Mit der Option BOOST arbeitet VACUUM in einem Fenster und blockiert gleichzeitige Lösch- und Aktualisierungsvorgänge für die Dauer der VACUUM-Operation. Wenn Sie die BOOST-Option nutzen, werden Systemressourcen gesucht, die die Abfrageleistung beeinträchtigen können. Führen Sie VACUUM BOOST aus, wenn die Last des Systems gering ist, z. B. bei Wartungsarbeiten.

Beachten Sie Folgendes, wenn Sie die Option BOOST verwenden:

- Wenn BOOST angegeben wird, wird der Wert table_name benötigt.
- BOOST wird von REINDEX nicht unterstützt.
- BOOST wird bei DELETE ONLY ignoriert.

Nutzungshinweise

Für die Mehrzahl der Amazon-Redshift-Anwendungen wird eine vollständige Bereinigung empfohlen. Weitere Informationen finden Sie unter Bereinigen von Tabellen.

Beachten Sie das folgende Verhalten, bevor Sie eine Bereinigungsoperation ausführen:

- Sie können VACUUM nicht innerhalb eines Transaktionsblocks ausführen (BEGIN ... END). Weitere Informationen Transaktionen finden Sie unter Serialisierbare Isolierung.
- Sie können jeweils nur einen VACUUM-Befehl für ein Cluster ausführen. Wenn Sie versuchen, mehrere Bereinigungsoperationen gleichzeitig auszuführen, gibt Amazon Redshift einen Fehler zurück.
- Es kann vorkommen, dass die Größe von Tabellen zunimmt, wenn sie bereinigt werden. Dieses Verhalten ist erwartet, wenn es keine gelöschten Zeilen gibt, für die Festplattenplatz zurückgewonnen werden könnte, oder wenn die neue Sortierreihenfolge der Tabelle zu einem niedrigeren Datenkompressionsverhältnis führt.
- Während Bereinigungsoperationen ist eine gewisse Abnahme der Abfrageleistung ein erwartetes Verhalten. Die normale Leistung wird wiederhergestellt, sobald die Bereinigungsoperation abgeschlossen ist.

- Gleichzeitig ausgeführte Schreiboperationen werden während Bereinigungsoperationen fortgesetzt. Es wird jedoch davon abgeraten, während Bereinigungsoperationen Schreiboperationen auszuführen. Es ist effizienter, Schreiboperationen abzuschließen, bevor die Bereinigung ausgeführt wird. Außerdem können Daten, die nach dem Starten einer Bereinigungsoperation geschrieben werden, von dieser Operation nicht bereinigt werden. In diesem Fall ist eine zweite Bereinigungsoperation erforderlich.
- Eine Bereinigungsoperation kann möglicherweise nicht gestartet werden, wenn bereits eine Lade- oder Einfügeoperation ausgeführt werden. Bereinigungsoperationen erfordern vorübergehend exklusiven Zugriff auf Tabellen, um gestartet werden zu können. Dieser exklusive Zugriff ist nur kurz erforderlich. Bereinigungsoperationen blockieren daher keine gleichzeitig ausgeführten Lade- und Einfügeoperationen über einen wesentlichen Zeitraum.
- Bereinigungsoperationen werden übersprungen, wenn es für eine bestimmte Tabelle keinen Handlungsbedarf gibt. Mit der Feststellung, dass die Operation übersprungen werden kann, ist jedoch ein gewisser Overhead verbunden. Wenn Ihnen bekannt ist, dass eine Tabelle optimiert ist oder den Schwellenwert für die Bereinigung nicht erfüllt, führen Sie keine Bereinigungsoperation für sie aus.
- Eine Bereinigungsoperation mit DELETE ONLY für eine kleine Tabelle reduziert möglicherweise die Zahl der Blöcke nicht, die für das Speichern der Daten verwendet werden, besonders, wenn die Tabelle eine große Zahl von Spalten besitzt oder der Cluster eine große Zahl von Slices pro Knoten verwendet. Diese Bereinigungsoperationen fügen einen Block pro Spalte und Slice hinzu, um gleichzeitig ausgeführte Einfügungen in die Tabelle zu berücksichtigen. Es ist möglich, dass dieser Overhead die Reduzierung der Blockzahl aufgrund der Rückgewinnung von Festplattenplatz aufwiegt. Wenn beispielsweise eine Tabelle mit 10 Spalten in einem Cluster mit 8 Knoten vor einer Bereinigung 1000 Blöcke besetzt, reduziert die Bereinigung die tatsächliche Blockzahl nicht, es sei denn, aufgrund der Löschung von Zeilen werden mehr als 80 Blöcke Festplattenplatz zurückgewonnen. (Jeder Datenblock nimmt 1 MB in Anspruch.)

Automatische Bereinigungsvorgänge werden angehalten, wenn eine der folgenden Bedingungen erfüllt ist:

- Ein Benutzer führt eine Data Definition Language (DDL)-Operation wie etwa ALTER TABLE aus, die eine exklusive Sperre für eine Tabelle erforderlich macht, an der die automatische Bereinigung aktuell arbeitet.
- Ein Benutzer löst VACUUM für eine Tabelle im Cluster aus (es kann nur jeweils ein VACUUM-Vorgang ausgeführt werden).

- Ein Zeitraum hoher Cluster-Auslastung.

Beispiele

Stellen Sie Platz und Datenbankplatz wieder her und sortieren Sie die Zeilen in allen Tabellen neu (basierend auf dem standardmäßigen Vacuum-Schwellenwert von 95 %).

```
vacuum;
```

Stellen Sie Platz und Datenbankplatz in der SALES-Tabelle wieder her und sortieren Sie die Zeilen in allen Tabellen neu (basierend auf dem standardmäßigen Vacuum-Schwellenwert von 95 %).

```
vacuum sales;
```

Rufen Sie immer verfügbaren Platz in der Tabelle SALES ab und sortieren Sie die Zeilen neu.

```
vacuum sales to 100 percent;
```

Sortieren Sie Zeilen in der Tabelle SALES nur dann neu, wenn weniger als 75 % der Zeilen bereits sortiert sind.

```
 vacuum sort only sales to 75 percent;
```

Gewinnt Festplattenplatz in der Tabelle SALES zurück, wenn mindestens 75 Prozent der verbleibenden Zeilen nach der Bereinigung nicht zur Löschung markiert sind.

```
vacuum delete only sales to 75 percent;
```

Indiziert die Tabelle LISTING neu und bereinigt sie anschließend.

```
vacuum reindex listing;
```

Der folgende Befehl gibt einen Fehler zurück.

```
vacuum reindex listing to 75 percent;
```

Ordnet die Cluster neu an und bereinigt die LISTING-Tabelle anschließend.

```
vacuum recluster listing;
```

Ordnet die Cluster neu an und bereinigt die LISTING-Tabelle anschließend mit der BOOST-Option.

```
vacuum recluster listing boost;
```

SQL-Funktionsreferenz

Themen

- Systemadministratorfunktionen
- Funktionen für Systeminformationen

Amazon Redshift unterstützt eine Reihe von Funktionen, bei denen es sich um Erweiterungen des SQL-Standards handelt, sowie Standardaggregationsfunktionen, skalare Funktionen und Fensterfunktionen.

> **Note**
>
> Amazon Redshift basiert auf PostgreSQL. Zwischen Amazon Redshift und PostgreSQL gibt es eine Reihe wichtiger Unterschiede, die Sie berücksichtigen müssen, wenn Sie Ihre Data-Warehouse-Anwendungen entwerfen und entwickeln. Weitere Informationen zu den Unterschieden zwischen Amazon-Redshift-SQL und PostgreSQL finden Sie unter Amazon Redshift und PostgreSQL.

Exklusive Führungsknotenfunktionen

Einige Amazon-Redshift-Abfragen werden auf den Rechenknoten verteilt und ausgeführt. Andere Abfragen werden ausschließlich auf dem Führungsknoten ausgeführt.

Der Führungsknoten verteilt SQL auf den Datenverarbeitungsknoten, wenn eine Abfrage benutzererstellte Tabellen oder Systemtabellen (Tabellen mit einem STL- oder STV-Präfix und Systemansichten mit einem SVL- oder SVV-Präfix) referenziert. Eine Abfrage, die nur Katalogtabellen (Tabellen mit einem PG-Präfix wie PG_TABLE_DEF) oder keine Tabellen referenziert, wird ausschließlich auf dem Führungsknoten ausgeführt.

Einige SQL-Funktionen von Amazon Redshift werden nur auf dem Führungsknoten und nicht auf den Rechenknoten unterstützt. Eine Abfrage, die eine Führungsknotenfunktion verwendet, darf nur auf dem Führungsknoten und nicht auf den Rechenknoten ausgeführt werden. Andernfalls wird ein Fehler zurückgegeben.

Die Dokumentation für die einzelnen exklusiven Führungsknotenfunktionen enthält einen Hinweis darauf, dass die Funktion einen Fehler zurückgibt, wenn sie auf benutzerdefinierte Tabellen oder Amazon-Redshift-Systemtabellen verweist.

Weitere Informationen finden Sie unter SQL-Funktionen, die auf dem Führungsknoten unterstützt werden.

Bei den folgenden SQL-Funktionen handelt es sich um exklusive Führungsknotenfunktionen, die nicht auf den Datenverarbeitungsknoten unterstützt werden:

Funktionen für Systeminformationen

- CURRENT_SCHEMA
- CURRENT_SCHEMAS
- HAS_DATABASE_PRIVILEGE
- HAS_SCHEMA_PRIVILEGE
- HAS_TABLE_PRIVILEGE

Zeichenfolgenfunktionen

- SUBSTR

Mathematische Funktionen

- FAKTORIELL ()

Die folgenden exklusiven Führungsknotenfunktionen sind veraltet und werden nicht mehr unterstützt:

Datumsfunktionen

- AGE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- LOCALTIME
- ISFINITE
- NOW

Zeichenfolgefunktionen

- GETBIT
- GET_BYTE
- SET_BIT

- SET_BYTE
- TO_ASCII

Exklusive Rechenknotenfunktionen

Einige Amazon-Redshift-Abfragen können nur in den Rechenknoten ausgeführt werden. Wenn eine Abfrage eine vom Benutzer erstellte Tabelle referenziert, wird SQL auf den Datenverarbeitungsknoten ausgeführt.

Eine Abfrage, die nur Katalogtabellen (Tabellen mit einem PG-Präfix wie PG_TABLE_DEF) oder keine Tabellen referenziert, wird ausschließlich auf dem Führungsknoten ausgeführt.

Wenn eine Abfrage, die eine Datenverarbeitungsknotenfunktion verwendet, auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist, wird der folgende Fehler zurückgegeben.

```
[Amazon](500310) Invalid operation: One or more of the used functions must be applied
 on at least one user created table.
```

Die Dokumentation für die einzelnen exklusiven Datenverarbeitungsknotenfunktionen enthält einen Hinweis darauf, dass die Funktion einen Fehler zurückgibt, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Die folgenden SQL-Funktionen sind reine Datenverarbeitungsknoten-Funktionen:

- LISTAGG
- MEDIAN
- PERCENTILE_CONT
- PERCENTILE_DISC und APPROXIMATE PERCENTILE_DISC

Aggregationsfunktionen

Themen

- Die Funktion COUNT
- Die Funktion LISTAGG
- Die Funktion MAX
- Die Funktion MEDIAN
- Die Funktion MIN
- Die Funktion PERCENTILE_CONT
- Die Funktionen STDDEV_SAMP und STDDEV_POP
- Die Funktion SUM
- Die Funktionen VAR_SAMP und VAR_POP

Aggregationsfunktionen verarbeiten einen einzelnen Ergebniswert aus einem Satz von Eingabewerten.

SELECT-Anweisungen, die Aggregationsfunktionen verwenden, können zwei optionale Klauseln enthalten: GROUP BY und HAVING. Die Syntax für diese Klauseln ist wie folgt (mit Verwendung der Funktion COUNT als Beispiel):

```
SELECT count (*) expression FROM table_reference
WHERE condition [GROUP BY expression ] [ HAVING condition]
```

Die GROUP BY-Klausel aggregiert und gruppiert Ergebnisse anhand der spezifischen Werte in einer oder mehreren angegebenen Spalten. Die HAVING-Klausel schränkt die Ergebnisse ein, die an Zeilen zurückgegeben werden, wenn eine bestimmte Aggregationsbedingung wahr ist, wie beispielsweise (*) > 1. Die HAVING-Klausel wird auf die gleiche Weise wie WHERE verwendet, um Zeilen auf der Basis des Werts einer Spalte einzuschränken. Ein Beispiel für diese zusätzlichen Klauseln finden Sie unter COUNT.

Aggregationsfunktionen akzeptieren keine verschachtelten Aggregationsfunktionen oder Fensterfunktionen als Argumente.

Funktion ANY_VALUE

Die Funktion ANY_VALUE gibt einen beliebigen Wert aus den Eingabeausdruckswerten nicht deterministisch zurück. Diese Funktion gibt NULL zurück, wenn der Eingabeausdruck nicht dazu führt, dass Zeilen zurückgegeben werden. Die Funktion kann auch NULL zurückgeben, wenn der Eingabeausdruck NULL-Werte enthält.

Syntax

```
ANY_VALUE( [ DISTINCT | ALL ] expression )
```

Argumente

DISTINCT | ALL

Geben Sie entweder DISTINCT oder ALL an, um einen beliebigen Wert aus den Eingabeausdruckswerten zurückzugeben. Das Argument DISTINCT hat keine Auswirkung und wird ignoriert.

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Der Ausdruck ist einer der folgenden Datentypen:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- BOOLEAN
- CHAR
- VARCHAR
- DATUM
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- INTERVAL YEAR TO MONTH
- INTERVAL DAY TO SECOND
- VARBYTE
- SUPER

- HLLSKETCH
- GEOMETRY
- GEOGRAPHY

Rückgabewert

Gibt denselben Datentyp wie expression zurück.

Nutzungshinweise

Wenn eine Anweisung, die die Funktion ANY_VALUE für eine Spalte angibt, auch einen Verweis auf eine zweite Spalte enthält, muss die zweite Spalte in einer GROUP-BY-Klausel oder in einer Aggregationsfunktion enthalten sein.

Beispiele

Die Beispiele verwenden die Ereignistabelle, die in Schritt 4: Laden von Beispieldaten aus Amazon S3 im Handbuch Erste Schritte mit Amazon Redshift erstellt wurde. Das folgende Beispiel gibt eine Instance für jede dateid mit dem eventname Eagles zurück.

```
select any_value(dateid) as dateid, eventname from event where eventname ='Eagles'
 group by eventname;
```

Die Ergebnisse sehen wie folgt aus.

```
dateid | eventname
-------+---------------
 1878  | Eagles
```

Das folgende Beispiel gibt eine Instance für jede dateid mit dem eventname Eagles oder Cold War Kids zurück.

```
select any_value(dateid) as dateid, eventname from event where eventname in('Eagles',
 'Cold War Kids') group by eventname;
```

Die Ergebnisse sehen wie folgt aus.

```
dateid | eventname
```

```
-------+---------------
 1922  | Cold War Kids
 1878  | Eagles
```

Die Funktion APPROXIMATE PERCENTILE_DISC

APPROXIMATE PERCENTILE_DISC ist eine Funktion für die inverse Verteilung, die ein diskretes Verteilungsmodell annimmt. Sie empfängt einen Perzentilwert und eine Sortierspezifikation und gibt ein Element aus dem angegebenen Satz zurück. Die Annäherung ermöglicht eine sehr viel schnellere Ausführung der Funktion bei einer niedrigen relativen Fehlerquote von ungefähr 0,5 Prozent.

APPROXIMATE PERCENTILE_DISC verwendet für einen bestimmten Perzentilwert einen zusammenfassenden Quantil-Algorithmus, um das diskrete Perzentil des Ausdrucks in der ORDER BY-Klausel anzunähern. APPROXIMATE PERCENTILE_DISC gibt den Wert mit dem kleinsten kumulativen Verteilungswert (in Bezug auf dieselbe Sortierspezifikation) zurück, der größer als oder gleich Perzentil ist.

APPROXIMATE PERCENTILE_DISC ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Syntax

```
APPROXIMATE  PERCENTILE_DISC ( percentile )
WITHIN GROUP (ORDER BY expr)
```

Argumente

percentile

Numerische Konstante zwischen 0 und 1. Null-Werte werden bei der Berechnung ignoriert.

WITHIN GROUP (ORDER BY expr)

Klausel, die numerische oder Datum-/Zeitwerte angibt, um das Perzentil zu sortieren und zu verarbeiten.

Rückgabewert

Derselbe Datentyp wie der ORDER BY-Ausdruck in der WITHIN GROUP-Klausel.

Nutzungshinweise

Wenn die Anweisung APPROXIMATE PERCENTILE_DISC eine GROUP BY-Klausel enthält, ist der Ergebnissatz begrenzt. Das Limit ist vom Knotentyp und der Anzahl der Knoten abhängig. Wenn das Limit überschritten wird, schlägt die Funktion fehl und gibt den folgenden Fehler zurück.

```
GROUP BY limit for approximate percentile_disc exceeded.
```

Wenn Sie mehr Gruppen auswerten müssen, als das Limit zulässt, sollten Sie die Verwendung von in Betracht ziehen Die Funktion PERCENTILE_CONT.

Beispiele

Im folgenden Beispiel werden die Anzahl der Verkäufe, der Gesamtumsatz und der fünfzigste Perzentilwert für die 10 Topdaten zurückgegeben.

```
select top 10 date.caldate,
count(totalprice), sum(totalprice),
approximate percentile_disc(0.5)
within group (order by totalprice)
from listing
join date on listing.dateid = date.dateid
group by date.caldate
order by 3 desc;

caldate    | count | sum        | percentile_disc
-----------+-------+------------+----------------
2008-01-07 |   658 | 2081400.00 |         2020.00
2008-01-02 |   614 | 2064840.00 |         2178.00
2008-07-22 |   593 | 1994256.00 |         2214.00
2008-01-26 |   595 | 1993188.00 |         2272.00
2008-02-24 |   655 | 1975345.00 |         2070.00
2008-02-04 |   616 | 1972491.00 |         1995.00
2008-02-14 |   628 | 1971759.00 |         2184.00
2008-09-01 |   600 | 1944976.00 |         2100.00
2008-07-29 |   597 | 1944488.00 |         2106.00
2008-07-23 |   592 | 1943265.00 |         1974.00
```

Die Funktion AVG

Die AVG-Funktion gibt den Durchschnitt (das arithmetische Mittel) der Eingabeausdruckwerte zurück. Die Funktion AVG ist mit numerischen Werten kompatibel und ignoriert NULL-Werte.

Syntax

```
AVG ( [ DISTINCT | ALL ] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Der Ausdruck ist einer der folgenden Datentypen:

- SMALLINT
- INTEGER
- BIGINT
- NUMERIC
- DECIMAL
- REAL
- DOUBLE PRECISION
- SUPER

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, bevor der Durchschnitt berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, um den Durchschnitt zu berechnen. ALL ist das Standardargument.

Datentypen

Die von der Funktion AVG unterstützten Argumenttypen sind SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL, DOUBLE PRECISION und SUPER.

Die von der Funktion AVG unterstützten Rückgabetypen sind:

- BIGINT für ein Ganzzahl-Argument
- DOUBLE PRECISION für ein Gleitkomma-Argument
- Gibt denselben Datentyp wie „expression“ für jeden anderen Argumenttyp zurück.

Die Standardpräzision für ein Ergebnis der AVG-Funktion des Typs NUMERIC oder DECIMAL ist 38. Die Ergebnisskala ist die gleiche wie die Skala des Arguments. Beispielsweise gibt eine AVG-Funktion für eine DEC(5,2)-Spalte einen DEC(38,2)-Datentyp zurück.

Beispiele

Suche der durchschnittlich verkauften Menge pro Transaktion in der Tabelle SALES:

```
select avg(qtysold)from sales;

avg
-----
2
(1 row)
```

Suche des durchschnittlichen aufgelisteten Gesamtpreises für alle Einträge:

```
select avg(numtickets*priceperticket) as avg_total_price from listing;

avg_total_price
-----------------
3034.41
(1 row)
```

Suche des durchschnittlich gezahlten Preises nach Monat in absteigender Reihenfolge:

```
select avg(pricepaid) as avg_price, month
from sales, date
where sales.dateid = date.dateid
group by month
order by avg_price desc;

avg_price | month
-----------+-------
659.34 | MAR
655.06 | APR
645.82 | JAN
643.10 | MAY
642.72 | JUN
642.37 | SEP
640.72 | OCT
640.57 | DEC
635.34 | JUL
```

```
635.24 | FEB
634.24 | NOV
632.78 | AUG
(12 rows)
```

Die Funktion COUNT

Die Funktion COUNT zählt die durch den Ausdruck definierten Zeilen.

Zu der Funktion COUNT gibt es folgende Varianten.

- COUNT (*) zählt alle Zeilen in der Zieltabelle, unabhängig davon, ob sie Null-Werte enthalten oder nicht.
- COUNT (expression) berechnet die Zahl der Zeilen mit Nicht-NULL-Werten in einer spezifischen Spalte oder einem spezifischen Ausdruck.
- COUNT (DISTINCT expression) berechnet die Zahl der unterschiedlichen Nicht-NULL-Werte in einer Spalte oder einem Ausdruck.
- APPROXIMATE COUNT DISTINCT ermittelt die ungefähre Anzahl der unterschiedlichen Nicht-NULL-Werte in einer Spalte oder einem Ausdruck.

Syntax

```
COUNT( * | expression )
```

```
COUNT ( [ DISTINCT | ALL ] expression )
```

```
APPROXIMATE COUNT ( DISTINCT expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Die Funktion COUNT unterstützt alle Argumentdatentypen.

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, bevor die Zählung ausgeführt wird. Mit dem Argument ALL behält die Funktion alle

duplizierten Werte aus dem angegebenen Ausdruck, um die Zählung auszuführen. ALL ist das Standardargument.

APPROXIMATE

Bei Verwendung mit APPROXIMATE verwendet eine COUNT DISTINCT-Funktion einen HyperLogLog Algorithmus, um die Anzahl der unterschiedlichen Werte, die nicht NULL sind, in einer Spalte oder einem Ausdruck zu approximieren. Abfragen mit dem Schlüsselwort APPROXIMATE werden sehr viel schneller bei einer niedrigen relativen Fehlerquote von etwa 2 % ausgeführt. Annäherungen sollten bei Abfragen verwendet werden, die eine große Zahl unterschiedlicher Werte zurückgeben, beispielsweise mehrere Millionen oder mehr pro Abfrage (bzw. pro Gruppe, wenn es eine GROUP BY-Klausel gibt). Bei kleineren Sätzen unterschiedlicher Werte, d. h. mehreren tausend, werden Annäherungen möglicherweise langsamer ausgeführt als präzise Zählungen. APPROXIMATE kann nur mit COUNT DISTINCT verwendet werden.

Rückgabetyp

Die Funktion COUNT gibt BIGINT zurück.

Beispiele

Zählung aller Benutzer aus dem Bundesstaat Florida:

```
select count(*) from users where state='FL';

count
-------
510
```

Zählung aller Ereignisnamen aus der EVENT-Tabelle:

```
select count(eventname) from event;

count
-------
8798
```

Zählung aller Ereignisnamen aus der EVENT-Tabelle:

```
select count(all eventname) from event;
```

```
count
-------
8798
```

Zählung aller eindeutigen Veranstaltungs-IDs aus der Tabelle EVENT:

```
select count(distinct venueid) as venues from event;

venues
--------
204
```

Zählung der Häufigkeit, mit der die einzelnen Verkäufer Batches von mehr als vier Tickets zum Verkauf aufgelistet haben; Gruppierung der Ergebnisse nach Verkäufer-ID:

```
select count(*), sellerid from listing
where numtickets > 4
group by sellerid
order by 1 desc, 2;

count | sellerid
------+----------
12    |    6386
11    |    17304
11    |    20123
11    |    25428
...
```

In den folgenden Beispielen werden die Rückgabewerte und Ausführungszeichen für COUNT und APPROXIMATE COUNT verglichen.

```
select  count(distinct pricepaid) from sales;

count
-------
  4528


Time: 48.048 ms
```

```
select approximate count(distinct pricepaid) from sales;

count
-------
  4553


Time: 21.728 ms
```

Die Funktion LISTAGG

Die Aggregationsfunktion „LISTAGG“ ordnet die Zeilen der Gruppe in einer Abfrage nach dem ORDER BY-Ausdruck an. Anschließend werden die Werte zu einer einzigen Zeichenfolge verkettet.

LISTAGG ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist. Weitere Informationen finden Sie unter Abfragen der Katalogtabellen.

Syntax

```
LISTAGG( [DISTINCT] aggregate_expression [, 'delimiter' ] )
[ WITHIN GROUP (ORDER BY order_list) ]
```

Argumente

DISTINCT

Eine Klausel, die duplizierte Werte in dem angegebenen Ausdruck beseitigt, bevor die Verkettung vorgenommen wird. Nachfolgende Leerzeichen werden ignoriert. Beispielsweise werden die Zeichenfolgen `'a'` und als `'a  '` Duplikate behandelt. „LISTAGG“ verwendet den ersten registrierten Wert. Weitere Informationen finden Sie unter Die Bedeutung von Leerzeichen am Ende.

aggregate_expression

Ein gültiger Ausdruck, wie etwa ein Spaltenname, der die Werte bereitstellt, die aggregiert werden sollen. NULL-Werte und leere Zeichenfolgen werden ignoriert.

delimiter

Die Zeichenfolgenkonstante, die die verketteten Werte trennt. Der Standardwert ist „NULL“.

WITHIN GROUP (ORDER BY order_list)

Eine Klausel, die die Sortierreihenfolge der aggregierten Werte angibt.

Rückgabewert

VARCHAR(MAX). Wenn der Ergebnissatz größer als die maximal zulässige Größe von VARCHAR ist, gibt LISTAGG den folgenden Fehler zurück:

```
Invalid operation: Result size exceeds LISTAGG limit
```

Nutzungshinweise

- Wenn eine Anweisung mehrere LISTAGG-Funktionen enthält, die WITHIN GROUP-Klauseln verwenden, muss jede WITHIN GROUP-Klausel dieselben ORDER BY-Werte verwenden.

  Die folgende Anweisung gibt beispielsweise einen Fehler zurück.

  ```
  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid)
  WITHIN GROUP (ORDER BY sellerid) AS dates
  FROM sales;
  ```

  Die folgende Anweisung wird erfolgreich ausgeführt.

  ```
  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid)
  WITHIN GROUP (ORDER BY dateid) AS dates
  FROM sales;

  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid) AS dates
  FROM sales;
  ```

Beispiele

Im folgenden Beispiel werden Verkäufer-IDs aggregiert, geordnet nach Verkäufer-ID.

```
SELECT LISTAGG(sellerid, ', ')
WITHIN GROUP (ORDER BY sellerid)
FROM sales
WHERE eventid = 4337;

listagg

-----------------------------------------------------------------------------------------------
380, 380, 1178, 1178, 1178, 2731, 8117, 12905, 32043, 32043, 32043, 32432, 32432,
 38669, 38750, 41498, 45676, 46324, 47188, 47188, 48294
```

Im folgenden Beispiel wird mit „DISTINCT“ eine Liste von einzigartigen Verkäufer-IDs zurückgegeben.

```
SELECT LISTAGG(DISTINCT sellerid, ', ')
WITHIN GROUP (ORDER BY sellerid)
FROM sales
WHERE eventid = 4337;

listagg

-----------------------------------------------------------------------------------------
380, 1178, 2731, 8117, 12905, 32043, 32432, 38669, 38750, 41498, 45676, 46324, 47188,
 48294
```

Im folgenden Beispiel werden Verkäufer-IDs aggregiert, geordnet nach Datum.

```
SELECT LISTAGG(sellerid, ', ')
WITHIN GROUP (ORDER BY dateid)
FROM sales
WHERE eventid = 4337;

   listagg
-----------------------------------------------------------------------------------------------
 41498, 47188, 47188, 1178, 1178, 1178, 380, 45676, 46324, 48294, 32043, 32043, 32432,
 12905, 8117, 38750, 2731, 32432, 32043, 380, 38669
```

Im folgenden Beispiel wird eine durch Pipe-Zeichen getrennte Liste von Verkaufsterminen für den Käufer mit der ID 660 zurückgegeben.

```
SELECT LISTAGG(
```

```
    (SELECT caldate FROM date WHERE date.dateid=sales.dateid), ' | '
)
WITHIN GROUP (ORDER BY sellerid DESC, salesid ASC)
FROM sales
WHERE buyerid = 660;

             listagg
-------------------------------------------------
2008-07-16 | 2008-07-09 | 2008-01-01 | 2008-10-26
```

Im folgenden Beispiel wird eine durch Kommata getrennte Liste von Verkaufs-IDs für die Käufer-IDs 660, 661 und 662 zurückgegeben.

```
SELECT buyerid,
LISTAGG(salesid,', ')
WITHIN GROUP (ORDER BY salesid) AS sales_id
FROM sales
WHERE buyerid BETWEEN 660 AND 662
GROUP BY buyerid
ORDER BY buyerid;

buyerid |               sales_id
--------+-------------------------------------------------------
660     | 32872, 33095, 33514, 34548
661     | 19951, 20517, 21695, 21931
662     | 3318, 3823, 4215, 51980, 53202, 55908, 57832, 171603
```

Die Funktion MAX

Die Funktion MAX gibt den maximal zulässigen Wert in einem Satz von Zeilen zurück. DISTINCT oder ALL könnten zwar verwendet werden, wirken sich jedoch nicht auf das Ergebnis aus.

Syntax

```
MAX ( [ DISTINCT | ALL ] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Der Ausdruck ist einer der folgenden Datentypen:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- CHAR
- VARCHAR
- DATUM
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- VARBYTE
- SUPER

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, bevor der maximal zulässige Wert berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, um den maximal zulässigen Wert zu berechnen. ALL ist das Standardargument.

Datentypen

Gibt denselben Datentyp wie expression zurück. Das boolesche Äquivalent der Funktion MIN ist Die Funktion BOOL_AND und das boolesche Äquivalent von MAX ist Die Funktion BOOL_OR.

Beispiele

Suche des höchsten Preises, der in allen Verkäufen gezahlt wurde:

```
select max(pricepaid) from sales;

max
```

```
----------
12624.00
(1 row)
```

Suche des höchsten Preises pro Ticket, der in allen Verkäufen gezahlt wurde:

```
select max(pricepaid/qtysold) as max_ticket_price
from sales;

max_ticket_price
-----------------
2500.00000000
(1 row)
```

Die Funktion MEDIAN

Berechnet den Medianwert für den Wertebereich. NULL-Werte im Bereich werden ignoriert.

MEDIAN ist eine Funktion für die inverse Verteilung, die ein kontinuierliches Verteilungsmodell annimmt.

MEDIAN ist ein Spezialfall von PERCENTILE_CONT.

MEDIAN ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Syntax

```
MEDIAN(median_expression)
```

Argumente

median_expression

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

Datentypen

Der Rückgabetyp wird durch den Datentyp von median_expression festgelegt. Die folgende Tabelle zeigt den Rückgabetyp für jeden median_expression-Datentyp an.

Input type	Rückgabetyp
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE
TIMESTAMP	TIMESTAMP
TIMESTAMPTZ	TIMESTAMPTZ

Nutzungshinweise

Wenn das Argument median_expression den Datentyp DECIMAL hat und mit der maximal zulässigen Präzision von 38 Stellen definiert ist, gibt MEDIAN möglicherweise ein falsches Ergebnis oder einen Fehler zurück. Wenn der Rückgabewert der Funktion MEDIAN 38 Stellen überschreitet, wird das Ergebnis entsprechend abgekürzt. Dies führt zu einem Genauigkeitsverlust. Wenn während der Interpolierung ein Zwischenergebnis die maximal zulässige Genauigkeit überschreitet, erfolgt ein numerischer Überlauf und die Funktion gibt einen Fehler zurück. Um diese Bedingungen zu vermeiden, werden die Verwendung eines Datentyps mit einer niedrigeren Genauigkeit oder die Umwandlung des Arguments median_expression in ein Argument mit niedrigerer Genauigkeit empfohlen.

Wenn eine Anweisung mehrere Aufrufe von sortierbasierten Aggregationsfunktionen enthält (LISTAGG, PERCENTILE_CONT oder MEDIAN), müssen alle dieselben ORDER BY-Werte verwenden. Beachten Sie, dass MEDIAN implizit eine Reihenfolge nach dem Wert des Ausdrucks anwendet.

Die folgende Anweisung gibt beispielsweise einen Fehler zurück.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

An error occurred when executing the SQL command:
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
```

```
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

ERROR: within group ORDER BY clauses for aggregate functions must be the same
```

Die folgende Anweisung wird erfolgreich ausgeführt.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(salesid)
FROM sales
GROUP BY salesid, pricepaid;
```

Beispiele

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Das folgende Beispiel zeigt, dass MEDIAN dieselben Ergebnisse wie PERCENTILE_CONT(0.5) produziert.

```
SELECT TOP 10 DISTINCT sellerid, qtysold,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold),
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid, qtysold;

+----------+---------+-----------------+--------+
| sellerid | qtysold | percentile_cont | median |
+----------+---------+-----------------+--------+
|        2 |       2 |               2 |      2 |
|       26 |       1 |               1 |      1 |
|       33 |       1 |               1 |      1 |
|       38 |       1 |               1 |      1 |
|       43 |       1 |               1 |      1 |
|       48 |       2 |               2 |      2 |
|       48 |       3 |               3 |      3 |
|       77 |       4 |               4 |      4 |
|       85 |       4 |               4 |      4 |
|       95 |       2 |               2 |      2 |
+----------+---------+-----------------+--------+
```

Im folgenden Beispiel wird die durchschnittliche Verkaufsmenge für jede sellerid ermittelt.

```
SELECT sellerid,
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid
ORDER BY sellerid
LIMIT 10;

+----------+--------+
| sellerid | median |
+----------+--------+
|        1 |    1.5 |
|        2 |      2 |
|        3 |      2 |
|        4 |      2 |
|        5 |      1 |
|        6 |      1 |
|        7 |    1.5 |
|        8 |      1 |
|        9 |      4 |
|       12 |      2 |
+----------+--------+
```

Verwenden Sie das folgende Beispiel, um die Ergebnisse der vorherigen Abfrage für die erste sellerid zu überprüfen.

```
SELECT qtysold
FROM sales
WHERE sellerid=1;

+---------+
| qtysold |
+---------+
|       2 |
|       1 |
+---------+
```

Die Funktion MIN

Die Funktion MIN gibt den Mindestwert in einem Satz von Zeilen zurück. DISTINCT oder ALL könnten zwar verwendet werden, wirken sich jedoch nicht auf das Ergebnis aus.

Syntax

```
MIN ( [ DISTINCT | ALL ] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Der Ausdruck ist einer der folgenden Datentypen:

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- CHAR
- VARCHAR
- DATUM
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- VARBYTE
- SUPER

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, bevor der Mindestwert berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, um den Mindestwert zu berechnen. ALL ist das Standardargument.

Datentypen

Gibt denselben Datentyp wie expression zurück. Das boolesche Äquivalent der Funktion MIN ist Die Funktion BOOL_AND und das boolesche Äquivalent von MAX ist Die Funktion BOOL_OR.

Beispiele

Suche des niedrigsten Preises, der in allen Verkäufen gezahlt wurde:

```
select min(pricepaid) from sales;

min
-------
20.00
(1 row)
```

Suche des niedrigsten Preises pro Ticket, der in allen Verkäufen gezahlt wurde:

```
select min(pricepaid/qtysold)as min_ticket_price
from sales;

min_ticket_price
------------------
20.00000000
(1 row)
```

Die Funktion PERCENTILE_CONT

PERCENTILE_CONT ist eine Funktion für die inverse Verteilung, die ein kontinuierliches Verteilungsmodell annimmt. Sie empfängt einen Perzentilwert und eine Sortierspezifikation und gibt einen interpolierten Wert zurück, der in Bezug auf die Sortierspezifikation in den angegebenen Perzentilwert fällt.

PERCENTILE_CONT berechnet eine lineare Interpolierung zwischen Werten, nachdem diese der Reihenfolge entsprechend angeordnet wurden. Mithilfe des Perzentilwerts `(P)` und der Anzahl der Nicht-Null-Zeilen `(N)` in der Aggregationsgruppe berechnet die Funktion die Anzahl der Zeilen, nachdem die Zeilen entsprechend der Sortierspezifikation angeordnet wurden. Die Anzahl von Zeilen `(RN)` wird mit der Formel `RN = (1+ (P*(N-1))` berechnet. Das Endergebnis der Aggregationsfunktion wird durch lineare Interpolierung zwischen den Werten aus Zeilen zwischen `CRN = CEILING(RN)` und `FRN = FLOOR(RN)` berechnet.

Das Ergebnis wird wie folgt aussehen.

Wenn `(CRN = FRN = RN)`, ist das Ergebnis `(value of expression from row at RN)`

Andernfalls sieht das Ergebnis wie folgt aus:

`(CRN - RN) * (value of expression for row at FRN) + (RN - FRN) * (value of expression for row at CRN)`.

PERCENTILE_CONT ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Syntax

```
PERCENTILE_CONT(percentile)
WITHIN GROUP(ORDER BY expr)
```

Argumente

percentile

Numerische Konstante zwischen 0 und 1. NULL-Werte werden bei der Berechnung ignoriert.

expr

Gibt numerische oder Datum-/Zeitwerte an, nach denen das Perzentil sortiert und berechnet werden soll.

Rückgabewert

Der Rückgabetyp wird durch den Datentyp des ORDER BY-Ausdrucks in der WITHIN GROUP-Klausel festgelegt. Die folgende Tabelle zeigt den Rückgabetyp für jeden ORDER BY-Datentyp an.

Input type	Rückgabetyp
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE

Input type	Rückgabetyp
TIMESTAMP	TIMESTAMP
TIMESTAMPTZ	TIMESTAMPTZ

Nutzungshinweise

Wenn das Argument ORDER BY den Datentyp DECIMAL hat und mit der maximal zulässigen Präzision von 38 Stellen definiert ist, gibt PERCENTILE_CONT möglicherweise ein falsches Ergebnis oder einen Fehler zurück. Wenn der Rückgabewert der Funktion PERCENTILE_CONT 38 Stellen überschreitet, wird das Ergebnis entsprechend abgekürzt. Dies führt zu einem Genauigkeitsverlust. Wenn während der Interpolierung ein Zwischenergebnis die maximal zulässige Genauigkeit überschreitet, erfolgt ein numerischer Überlauf und die Funktion gibt einen Fehler zurück. Um diese Bedingungen zu vermeiden, werden die Verwendung eines Datentyps mit einer niedrigeren Genauigkeit oder die Umwandlung des Ausdrucks ORDER BY in einen Ausdruck mit niedrigerer Genauigkeit empfohlen.

Wenn eine Anweisung mehrere Aufrufe von sortierbasierten Aggregationsfunktionen enthält (LISTAGG, PERCENTILE_CONT oder MEDIAN), müssen alle dieselben ORDER BY-Werte verwenden. Beachten Sie, dass MEDIAN implizit eine Reihenfolge nach dem Wert des Ausdrucks anwendet.

Die folgende Anweisung gibt beispielsweise einen Fehler zurück.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

An error occurred when executing the SQL command:
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

ERROR: within group ORDER BY clauses for aggregate functions must be the same
```

Die folgende Anweisung wird erfolgreich ausgeführt.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(salesid)
FROM sales
GROUP BY salesid, pricepaid;
```

Beispiele

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Das folgende Beispiel zeigt, dass PERCENTILE_CONT(0.5) dieselben Ergebnisse wie MEDIAN produziert.

```
SELECT TOP 10 DISTINCT sellerid, qtysold,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold),
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid, qtysold;

+----------+---------+-----------------+--------+
| sellerid | qtysold | percentile_cont | median |
+----------+---------+-----------------+--------+
|        2 |       2 |               2 |      2 |
|       26 |       1 |               1 |      1 |
|       33 |       1 |               1 |      1 |
|       38 |       1 |               1 |      1 |
|       43 |       1 |               1 |      1 |
|       48 |       2 |               2 |      2 |
|       48 |       3 |               3 |      3 |
|       77 |       4 |               4 |      4 |
|       85 |       4 |               4 |      4 |
|       95 |       2 |               2 |      2 |
+----------+---------+-----------------+--------+
```

Im folgenden Beispiel werden PERCENTILE_CONT (0.5) und PERCENTILE_CONT (0.75) für die Verkaufsmenge für jede sellerid in der Tabelle SALES ermittelt.

```
SELECT sellerid,
```

```
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold) as pct_05,
PERCENTILE_CONT(0.75) WITHIN GROUP(ORDER BY qtysold) as pct_075
FROM sales
GROUP BY sellerid
ORDER BY sellerid
LIMIT 10;

+----------+--------+---------+
| sellerid | pct_05 | pct_075 |
+----------+--------+---------+
|        1 |    1.5 |    1.75 |
|        2 |      2 |    2.25 |
|        3 |      2 |       3 |
|        4 |      2 |       2 |
|        5 |      1 |     1.5 |
|        6 |      1 |       1 |
|        7 |    1.5 |    1.75 |
|        8 |      1 |       1 |
|        9 |      4 |       4 |
|       12 |      2 |    3.25 |
+----------+--------+---------+
```

Verwenden Sie das folgende Beispiel, um die Ergebnisse der vorherigen Abfrage für die erste sellerid zu überprüfen.

```
SELECT qtysold
FROM sales
WHERE sellerid=1;

+---------+
| qtysold |
+---------+
|       2 |
|       1 |
+---------+
```

Die Funktionen STDDEV_SAMP und STDDEV_POP

Die Funktionen STDDEV_SAMP und STDDEV_POP geben die Stichproben- und Populationsstandardabweichungen eines Satzes numerischer Werte (integer, decimal oder floating-point) zurück. Das Ergebnis der Funktion STDDEV_SAMP entspricht der Quadratwurzel der Stichprobenabweichung desselben Satzes von Werten.

STDDEV_SAMP und STDDEV sind Synonyme für dieselbe Funktion.

Syntax

```
STDDEV_SAMP | STDDEV ( [ DISTINCT | ALL ] expression)
STDDEV_POP ( [ DISTINCT | ALL ] expression)
```

Der Ausdruck muss einen Ganzzahl-, Dezimal- oder Gleitkommadatentyp haben. Unabhängig vom Datentyp des Ausdrucks ist der Rückgabewert dieser Funktion eine DOUBLE PRECISION-Zahl.

> **Note**
>
> Die Standardabweichung wird mittels Gleitkommaarithmetik berechnet. Dies kann zu einer leichten Ungenauigkeit führen.

Nutzungshinweise

Wenn die Stichprobenstandardabweichung (STDDEV oder STDDEV_SAMP) für einen Ausdruck berechnet wird, der aus einem einzigen Wert besteht, ist das Ergebnis der Funktion NULL und nicht 0.

Beispiele

Die folgende Abfrage gibt den Durchschnitt der Werte in der Spalte VENUESEATS der Tabelle VENUE zurück, gefolgt von der Stichprobenstandardabweichung und der Populationsstandardabweichung desselben Satzes von Werten. VENUESEATS ist eine INTEGER-Spalte. Die Ergebnisskala ist auf 2 Ziffern reduziert.

```
select avg(venueseats),
cast(stddev_samp(venueseats) as dec(14,2)) stddevsamp,
cast(stddev_pop(venueseats) as dec(14,2)) stddevpop
from venue;

avg  | stddevsamp | stddevpop
-------+------------+-----------
17503 |   27847.76 |  27773.20
(1 row)
```

Die folgende Abfrage gibt die Stichprobenstandardabweichung für die Spalte COMMISSION in der Tabelle SALES zurück. COMMISSION ist eine DECIMAL-Spalte. Die Ergebnisskala ist auf 10 Ziffern reduziert.

```
select cast(stddev(commission) as dec(18,10))
from sales;

stddev
----------------
130.3912659086
(1 row)
```

Die folgende Abfrage gibt die Stichprobenstandardabweichung für die Spalte COMMISSION als Ganzzahl aus.

```
select cast(stddev(commission) as integer)
from sales;

stddev
--------
130
(1 row)
```

Die folgende Abfrage gibt sowohl die Stichprobenstandardabweichung als auch die Quadratwurzel der Stichprobenabweichung für die Spalte COMMISSION zurück. Die Ergebnisse dieser Berechnungen sind identisch.

```
select
cast(stddev_samp(commission) as dec(18,10)) stddevsamp,
cast(sqrt(var_samp(commission)) as dec(18,10)) sqrtvarsamp
from sales;

stddevsamp   |  sqrtvarsamp
---------------+----------------
130.3912659086 | 130.3912659086
(1 row)
```

Die Funktion SUM

Die Funktion SUM gibt die Summe der Eingabespalten- oder Ausdruckswerte zurück. Die Funktion SUM ist mit numerischen Werten kompatibel und ignoriert NULL-Werte.

Syntax

```
SUM ( [ DISTINCT | ALL ] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Der Ausdruck ist einer der folgenden Datentypen:

- SMALLINT
- INTEGER
- BIGINT
- NUMERIC
- DECIMAL
- REAL
- DOUBLE PRECISION
- SUPER

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, bevor die Summe berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, um die Summe zu berechnen. ALL ist das Standardargument.

Datentypen

Die von der Funktion SUM unterstützten Argumenttypen sind SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL, DOUBLE PRECISION und SUPER.

Die von der Funktion SUM unterstützten Rückgabetypen sind

- BIGINT für BIGINT-, SMALLINT- und INTEGER-Argumente
- NUMERIC für NUMERIC-Argumente
- DOUBLE PRECISION für Gleitkomma-Argumente
- Gibt denselben Datentyp wie „expression“ für jeden anderen Argumenttyp zurück.

Die Standardpräzision für ein Ergebnis der SUM-Funktion des Typs NUMERIC oder DECIMAL ist 38. Die Ergebnisskala ist die gleiche wie die Skala des Arguments. Beispielsweise gibt eine SUM-Funktion für eine DEC(5,2)-Spalte einen DEC(38,2)-Datentyp zurück.

Beispiele

Suche der Summe aller gezahlten Provisionen in der Tabelle SALES:

```
select sum(commission) from sales;

sum
-------------
16614814.65
(1 row)
```

Suche der Anzahl der Plätze an allen Veranstaltungsorten im Bundesstaat Florida:

```
select sum(venueseats) from venue
where venuestate = 'FL';

sum
--------
250411
(1 row)
```

Suche der Anzahl der im Mai verkauften Plätze:

```
select sum(qtysold) from sales, date
where sales.dateid = date.dateid and date.month = 'MAY';

sum
-------
32291
(1 row)
```

Die Funktionen VAR_SAMP und VAR_POP

Die Funktionen VAR_SAMP und VAR_POP geben die Stichproben- und Populationsabweichung eines Satzes numerischer Werte (integer, decimal oder floating-point) zurück. Das Ergebnis der Funktion VAR_SAMP entspricht der Quadratwurzel der Stichprobenstandardabweichung desselben Satzes von Werten.

VAR_SAMP und VARIANCE sind Synonyme für dieselbe Funktion.

Syntax

```
VAR_SAMP | VARIANCE ( [ DISTINCT | ALL ] expression)
VAR_POP ( [ DISTINCT | ALL ] expression)
```

Der Ausdruck muss einen Ganzzahl-, Dezimal- oder Gleitkommadatentyp haben. Unabhängig vom Datentyp des Ausdrucks ist der Rückgabewert dieser Funktion eine DOUBLE PRECISION-Zahl.

> **Note**
>
> Die Ergebnisse dieser Funktionen sind je nach Data Warehouse-Cluster verschieden, abhängig von der Konfiguration des jeweiligen Clusters.

Nutzungshinweise

Wenn die Stichprobenabweichung (VARIANCE oder VAR_SAMP) für einen Ausdruck berechnet wird, der aus einem einzigen Wert besteht, ist das Ergebnis der Funktion NULL und nicht 0.

Beispiele

Die folgende Abfrage gibt die gerundete Stichproben- und Populationsabweichung für die Spalte NUMTICKETS in der Tabelle LISTING zurück.

```
select avg(numtickets),
round(var_samp(numtickets)) varsamp,
round(var_pop(numtickets)) varpop
from listing;

avg | varsamp | varpop
-----+---------+--------
10 |      54 |     54
(1 row)
```

Die folgende Abfrage führt dieselben Berechnungen aus, gibt die Ergebnisse jedoch als Dezimalwerte aus.

```
select avg(numtickets),
```

```
cast(var_samp(numtickets) as dec(10,4)) varsamp,
cast(var_pop(numtickets) as dec(10,4)) varpop
from listing;

avg | varsamp | varpop
-----+---------+---------
10 | 53.6291 | 53.6288
(1 row)
```

Array-Funktionen

Im Folgenden finden Sie eine Beschreibung der Array-Funktionen für SQL, die Amazon Redshift für den Zugriff und die Bearbeitung von Arrays unterstützt.

Themen

- array-Funktion
- array_concat-Funktion
- array_flatten-Funktion
- get_array_length-Funktion
- split_to_array-Funktion
- subarray-Funktion

array-Funktion

Erstellt ein Array des SUPER-Datentyps.

Syntax

```
ARRAY( [ expr1 ] [ , expr2 [ , ... ] ] )
```

Argument

expr1, expr2

Ausdrücke eines Amazon Redshift-Datentyps außer Datums- und Uhrzeittypen, da Amazon Redshift die Datums- und Uhrzeittypen nicht in den SUPER-Datentyp umwandelt. Die Argumente müssen nicht denselben Datentyp haben.

Rückgabetyp

Die array-Funktion gibt den Datentyp SUPER zurück.

Beispiel

Die folgenden Beispiele zeigen ein Array numerischer Werte und ein Array verschiedener Datentypen.

```
--an array of numeric values
select array(1,50,null,100);
      array
------------------
 [1,50,null,100]
(1 row)

--an array of different data types
select array(1,'abc',true,3.14);
        array
----------------------
 [1,"abc",true,3.14]
(1 row)
```

array_concat-Funktion

Die array_concat-Funktion verkettet zwei Arrays, um ein Array zu erstellen, das alle Elemente im ersten Array enthält, gefolgt von allen Elementen im zweiten Array. Beide Argumente müssen gültige Arrays sein.

Syntax

```
array_concat( super_expr1,  super_expr2 )
```

Argumente

super_expr1

Der Wert, der das erste der beiden zu verkettenden Arrays angibt.

super_expr2

Der Wert, der das zweite der beiden zu verkettenden Arrays angibt.

Rückgabetyp

Die array_concat-Funktion gibt einen SUPER-Datenwert zurück.

Beispiel

Die folgenden Beispiele zeigen die Verkettung von zwei Arrays desselben Typs und die Verkettung von zwei Arrays unterschiedlichen Typs.

```
-- concatenating two arrays
SELECT ARRAY_CONCAT(ARRAY(10001,10002),ARRAY(10003,10004));
              array_concat
-----------------------------------
 [10001,10002,10003,10004]
(1 row)

-- concatenating two arrays of different types
SELECT ARRAY_CONCAT(ARRAY(10001,10002),ARRAY('ab','cd'));
          array_concat
------------------------------
 [10001,10002,"ab","cd"]
(1 row)
```

array_flatten-Funktion

Führt mehrere Arrays in einem einzelnen Array vom SUPER-Typ zusammen.

Syntax

```
array_flatten( super_expr1,super_expr2,.. )
```

Argumente

super_expr1,super_expr2

Ein gültiger SUPER-Ausdruck der Array-Form.

Rückgabetyp

Die array_flatten-Funktion gibt einen SUPER-Datenwert zurück.

Beispiel

Das folgende Beispiel zeigt eine array_flatten-Funktion.

```
SELECT ARRAY_FLATTEN(ARRAY(ARRAY(1,2,3,4),ARRAY(5,6,7,8),ARRAY(9,10)));
     array_flatten
-----------------------
 [1,2,3,4,5,6,7,8,9,10]
(1 row)
```

get_array_length-Funktion

Gibt die Länge des angegebenen Arrays zurück. Die Funktion GET_ARRAY_LENGTH gibt die Länge eines an ein bestimmtes Objekt übergebenen SUPER-Arrays oder eines Array-Pfads an.

Syntax

```
get_array_length( super_expr )
```

Argumente

super_expr

Ein gültiger SUPER-Ausdruck der Array-Form.

Rückgabetyp

Die get_array_length-Funktion gibt einen BIGINT zurück.

Beispiel

Das folgende Beispiel zeigt eine get_array_length-Funktion.

```
SELECT GET_ARRAY_LENGTH(ARRAY(1,2,3,4,5,6,7,8,9,10));
 get_array_length
----------------------
            10
(1 row)
```

split_to_array-Funktion

Verwendet ein Trennzeichen als optionalen Parameter. Wenn kein Trennzeichen vorhanden ist, ist der Standardwert ein Komma.

Syntax

```
split_to_array( string,delimiter )
```

Argumente

string

Die Eingabezeichenfolge, die geteilt werden soll.

delimiter

Ein optionaler Wert nach dem die Eingabezeichenfolge getrennt wird. Standardmäßig wird ein Komma verwendet.

Rückgabetyp

Die split_to_array-Funktion gibt einen SUPER-Datenwert zurück.

Beispiel

Das folgende Beispiel zeigt eine split_to_array-Funktion.

```
SELECT SPLIT_TO_ARRAY('12|345|6789', '|');
     split_to_array
------------------------
 ["12","345","6789"]
(1 row)
```

subarray-Funktion

Manipuliert Arrays, um eine Teilmenge der Eingabe-Arrays zurückzugeben.

Syntax

```
SUBARRAY( super_expr, start_position, length )
```

Argumente

super_expr

Ein gültiger SUPER-Ausdruck in Array-Form.

start_position

Die Position innerhalb des Arrays, an der die Extrahierung gestartet werden soll, beginnend mit der Indexposition 0. Eine negative Position zählt vom Ende des Arrays rückwärts.

length

Die Anzahl der Element, die extrahiert werden soll (die Länge der Unterzeichenfolge).

Rückgabetyp

Die subarray-Funktion gibt einen SUPER-Datenwert zurück.

Beispiele

Das folgende Beispiel zeigt eine subarray-Funktion.

```
 SELECT SUBARRAY(ARRAY('a', 'b', 'c', 'd', 'e', 'f'), 2, 3);
   subarray
---------------
 ["c","d","e"]
(1 row)
```

Bitweise Aggregationsfunktionen

Bitweise Aggregatfunktionen berechnen Bitoperationen, um die Aggregation von Ganzzahlspalten und Spalten, die auf ganzzahlige Werte konvertiert oder gerundet werden können, durchzuführen.

Themen

- Verwendung von NULL-Werten in bitweisen Aggregationen
- DISTINCT-Unterstützung für bitweise Aggregationen
- Übersicht: Beispiele für bitweise Funktionen
- Die Funktion BIT_AND

- Die Funktion BIT_OR
- Die Funktion BOOL_AND
- Die Funktion BOOL_OR

Verwendung von NULL-Werten in bitweisen Aggregationen

Wenn Sie eine bitweise Funktion auf eine Spalte anwenden, die nullwertfähig ist, werden alle NULL-Werte entfernt, bevor das Funktionsergebnis berechnet wird. Wenn keine Zeilen für eine Aggregation qualifiziert sind, gibt die bitweise Funktion NULL zurück. Das gleiche Verhalten gilt für reguläre Aggregationsfunktionen. Im Folgenden sehen Sie ein Beispiel.

```
select sum(venueseats), bit_and(venueseats) from venue
where venueseats is null;

sum  | bit_and
------+---------
null |    null
(1 row)
```

DISTINCT-Unterstützung für bitweise Aggregationen

Wie andere Aggregationsfunktionen unterstützen auch bitweise Funktionen das Schlüsselwort DISTINCT.

Die Verwendung von DISTINCT mit diesen Funktionen wirkt sich jedoch nicht auf die Ergebnisse aus. Die erste Instanz eines Werts ist für bitweise AND- oder OR-Operationen ausreichend. Das Vorhandensein duplizierter Werte im ausgewerteten Ausdruck hat keine Auswirkungen.

Wir empfehlen, DISTINCT nicht mit bitweisen Funktionen zu verwenden, da die DISTINCT-Verarbeitung wahrscheinlich zu einem Overhead bei der Abfrageausführung führen wird.

Übersicht: Beispiele für bitweise Funktionen

Unten sehen Sie eine Übersicht mit einigen Beispielen für Einsatzmöglichkeiten von bitweisen Funktionen. Zu jeder Funktionsbeschreibung gehören spezifische Codebeispiele.

Die Beispiele für die bitweisen Funktionen basieren auf der TICKIT-Beispieldatenbank. Die Tabelle USERS in der Beispieldatenbank TICKIT enthält mehrere boolesche Spalten, die angeben, ob für die

einzelnen Benutzer bekannt ist, dass ihnen bestimmte Arten von Veranstaltungen gefallen, wie Sport, Theater, Oper usw. Ein Beispiel folgt.

```
select userid, username, lastname, city, state,
likesports, liketheatre
from users limit 10;

userid | username | lastname  |     city     | state | likesports | liketheatre
-------+----------+-----------+--------------+-------+------------+-------------
1 | JSG99FHE | Taylor    | Kent         | WA    | t          | t
9 | MSD36KVR | Watkins   | Port Orford  | MD    | t          | f
```

Nehmen wir an, dass die Tabelle USERS auf eine andere Weise erstellt wird. In der neuen Version ist eine einzelne Ganzzahlspalte enthalten, die (in binärer Form) acht Arten von Veranstaltungen definiert, die den einzelnen Benutzern gefallen oder nicht gefallen. In diesem Entwurf stellt jede Bitposition einen Veranstaltungstyp dar. Für einen Benutzer, dem alle acht Veranstaltungen gefallen, sind alle acht Bits auf 1 festgelegt (wie in der ersten Zeile der folgenden Tabelle gezeigt). Für einen Benutzer, dem keine dieser Veranstaltungen gefallen, sind alle acht Bits auf 0 festgelegt (wie in der zweiten Zeile gezeigt). Ein Benutzer, dem nur Sport und Jazz gefallen, wird in der dritten Zeile gezeigt.

	SPORT	THEATEF	JAZZ	OPER	ROCK	VEGAS	BROADW	KLASSISCH E MUSIK
Benutzer	1	1	1	1	1	1	1	1
Benutzer	0	0	0	0	0	0	0	0
Benutzer	1	0	1	0	0	0	0	0

Diese Binärwerte könnten in der Datenbanktabelle in einer einzelnen Spalte LIKES als Ganzzahlen gespeichert werden. Dies würde wie folgt aussehen.

Benutzer	Binärwert	Gespeicherter Wert (Ganzzahl)
Benutzer 1	11111111	255

Benutzer	Binärwert	Gespeicherter Wert (Ganzzahl)
Benutzer 2	00000000	0
Benutzer 3	10100000	160

Die Funktion BIT_AND

Die BIT_AND-Funktion führt bitweise OR-Operationen für alle Werte in einer einzigen Ganzzahlspalte bzw. in einem einzigen Ganzzahlausdruck aus. Diese Funktion aggregiert jedes Bit jedes Binärwerts, das jedem Ganzzahlwert im Ausdruck entspricht.

Die Funktion BIT_AND gibt das Ergebnis `0` zurück, wenn für keinen der Werte ein Bit auf 1 festgelegt ist. Wenn für einen der Werte mindestens ein Bit auf 1 festgelegt ist, gibt die Funktion einen Ganzzahlwert zurück. Diese Ganzzahl ist die Nummer, die dem Binärwert für diese Bits entspricht.

Beispielsweise enthält eine Tabelle vier Ganzzahlwerte in einer Spalte: 3, 7, 10 und 22. Diese Ganzzahlen werden wie folgt in binärer Form dargestellt:

Ganzzahl	Binärwert
3	11
7	111
10	1010
22	10110

Eine BIT_AND-Operation für diesen Datensatz stellt fest, dass alle Bits nur auf die second-to-last Position 1 gesetzt sind. Das Ergebnis ist der Binärwert `00000010`, der für den Ganzzahlwert 2 steht. Daher gibt die Funktion BIT_AND zurück 2.

Syntax

```
BIT_AND ( [DISTINCT | ALL] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Dieser Ausdruck muss den Datentyp INT, INT2 oder INT8 haben. Die Funktion gibt einen entsprechenden INT-, INT2- oder INT8-Datentyp zurück.

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte für den angegebenen Ausdruck, bevor das Ergebnis berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte. ALL ist das Standardargument. Weitere Informationen finden Sie unter DISTINCT-Unterstützung für bitweise Aggregationen.

Beispiele

Da relevante geschäftliche Informationen in Ganzzahlspalten gespeichert werden, können Sie bitweise Funktionen verwenden, um diese Informationen zu extrahieren und zu aggregieren. Die folgende Abfrage wendet die Funktion BIT_AND auf die Spalte LIKES in einer Tabelle namens USERLIKES an und gruppiert die Ergebnisse anhand der Spalte CITY.

```
select city, bit_and(likes) from userlikes group by city
order by city;
city          | bit_and
--------------+---------
Los Angeles   |       0
Sacramento    |       0
San Francisco |       0
San Jose      |      64
Santa Barbara |     192
(5 rows)
```

Diese Ergebnisse können wie folgt interpretiert werden:

- Der Ganzzahlwert 192 für Santa Barbara wird in den Binärwert `11000000` übersetzt. Mit anderen Worten, allen Benutzern in dieser Stadt gefallen Sport und Theater. Nicht allen Benutzern gefallen jedoch auch andere Arten von Veranstaltungen.
- Die Ganzzahl 64 entspricht `01000000`. Es gibt also nur eine Art von Veranstaltung, die allen Benutzern in San Jose gefällt: Theater.

- Der Wert `0` für die anderen drei Städte zeigt an, dass es keine Veranstaltungen gibt, die allen Benutzern in diesen Städten gefallen.

Die Funktion BIT_OR

Die BIT_OR-Funktion führt bitweise OR-Operationen für alle Werte in einer einzigen Ganzzahlspalte bzw. in einem einzigen Ganzzahlausdruck aus. Diese Funktion aggregiert jedes Bit jedes Binärwerts, das jedem Ganzzahlwert im Ausdruck entspricht.

Nehmen wir an, dass eine Tabelle vier Ganzzahlwerte in einer Spalte enthält: 3, 7, 10 und 22. Diese Ganzzahlen werden wie folgt in binärer Form dargestellt.

Ganzzahl	Binärwert
3	11
7	111
10	1010
22	10110

Wenn Sie auf den Satz von Ganzzahlwerten die Funktion BIT_OR anwenden, sucht die Operation nach jedem Wert, der in jeder Position eine 1 enthält. In diesem Fall ist 1 in den letzten fünf Positionen mindestens eines der Werte enthalten, was zu einem Binärergebnis von `00011111` führt. Daher gibt die Funktion `31` (oder `16 + 8 + 4 + 2 + 1`) zurück.

Syntax

```
BIT_OR ( [DISTINCT | ALL] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Dieser Ausdruck muss den Datentyp INT, INT2 oder INT8 haben. Die Funktion gibt einen entsprechenden INT-, INT2- oder INT8-Datentyp zurück.

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte für den angegebenen Ausdruck, bevor das Ergebnis berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte. ALL ist das Standardargument. Weitere Informationen finden Sie unter DISTINCT-Unterstützung für bitweise Aggregationen.

Beispiel

Die folgende Abfrage wendet die Funktion BIT_OR auf die Spalte LIKES in einer Tabelle namens USERLIKES an und gruppiert die Ergebnisse anhand der Spalte CITY.

```
select city, bit_or(likes) from userlikes group by city
order by city;
city          | bit_or
--------------+--------
Los Angeles   |    127
Sacramento    |    255
San Francisco |    255
San Jose      |    255
Santa Barbara |    255
(5 rows)
```

Mindestens einem Benutzer in den vier aufgelisteten Städten gefallen alle Arten von Veranstaltungen (255=11111111). Mindestens einem Benutzer in Los Angeles gefallen alle Arten von Veranstaltungen außer Sport (127=01111111).

Die Funktion BOOL_AND

Die Funktion BOOL_AND wird für eine einzige boolesche oder Ganzzahlspalte bzw. einen einzigen booleschen oder Ganzzahlausdruck ausgeführt. Diese Funktion wendet ähnliche Logik auf die Funktionen BIT_AND und BIT_OR an. Für diese Funktion ist der Rückgabetyp ein boolescher Wert (`true` oder `false`).

Wenn alle Werte in einem Satz „true“ sind, gibt die Funktion BOOL_AND `true` (`t`) zurück. Wenn ein Wert „false“ ist, gibt die Funktion `false` (`f`) zurück.

Syntax

```
BOOL_AND ( [DISTINCT | ALL] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Dieser Ausdruck muss einen booleschen oder Ganzzahl-Datentyp haben. Der Rückgabewert der Funktion ist BOOLEAN.

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte für den angegebenen Ausdruck, bevor das Ergebnis berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte. ALL ist das Standardargument. Weitere Informationen finden Sie unter DISTINCT-Unterstützung für bitweise Aggregationen.

Beispiele

Sie können die booleschen Funktionen auf boolesche Ausdrücke oder Ganzzahlausdrücke anwenden. Beispielsweise gibt die folgende Abfrage Ergebnisse aus der Standardtabelle USERS in der Datenbank TICKIT zurück, die mehrere boolesche Spalten besitzt.

Die Funktion BOOL_AND gibt für alle fünf Zeilen `false` zurück. Nicht allen Benutzern in diesen Bundesstaaten gefällt Sport.

```
select state, bool_and(likesports) from users
group by state order by state limit 5;

state | bool_and
------+---------
AB    | f
AK    | f
AL    | f
AZ    | f
BC    | f
(5 rows)
```

Die Funktion BOOL_OR

Die Funktion BOOL_OR wird für eine einzige boolesche oder Ganzzahlspalte bzw. einen einzigen booleschen oder Ganzzahlausdruck ausgeführt. Diese Funktion wendet ähnliche Logik auf die Funktionen BIT_AND und BIT_OR an. Für diese Funktion ist der Rückgabetyp ein boolescher Wert (`true`, `false` oder NULL).

Wenn ein oder mehrere Werte in einer Menge gleich sind`true`, gibt die Funktion BOOL_OR den Wert () zurück. `true` `t` Wenn alle Werte in einer Menge gleich sind`false`, gibt die Funktion () zurück`false`. `f` NULL kann zurückgegeben werden, wenn der Wert unbekannt ist.

Syntax

```
BOOL_OR ( [DISTINCT | ALL] expression )
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird. Dieser Ausdruck muss einen booleschen oder Ganzzahl-Datentyp haben. Der Rückgabewert der Funktion ist BOOLEAN.

DISTINCT | ALL

Mit dem Argument DISTINCT beseitigt die Funktion alle duplizierten Werte für den angegebenen Ausdruck, bevor das Ergebnis berechnet wird. Mit dem Argument ALL behält die Funktion alle duplizierten Werte. ALL ist das Standardargument. Siehe DISTINCT-Unterstützung für bitweise Aggregationen.

Beispiele

Sie können die booleschen Funktionen mit booleschen Ausdrücken oder Ganzzahlausdrücken verwenden. Beispielsweise gibt die folgende Abfrage Ergebnisse aus der Standardtabelle USERS in der Datenbank TICKIT zurück, die mehrere boolesche Spalten besitzt.

Die Funktion BOOL_OR gibt für alle fünf Zeilen `true` zurück. Mindestens einem Benutzer in diesen Bundesstaaten gefällt Sport.

```
select state, bool_or(likesports) from users
group by state order by state limit 5;

state | bool_or
------+--------
AB    | t
AK    | t
AL    | t
AZ    | t
BC    | t
```

```
(5 rows)
```

Im folgenden Beispiel wird NULL zurückgegeben.

```
SELECT BOOL_OR(NULL = '123')
               bool_or
------
NULL
```

Bedingte Ausdrücke

Themen

- Der bedingte Ausdruck CASE
- DECODE-Funktion
- Funktionen GREATEST und LEAST
- NVL- und COALESCE-Funktionen
- Funktion NVL2
- NULLIF-Funktion

Amazon Redshift unterstützt einige bedingte Ausdrücke, die Erweiterungen des SQL-Standards darstellen.

Der bedingte Ausdruck CASE

Der CASE-Ausdruck ist ein bedingter Ausdruck, vergleichbar den IF-/THEN-/ELSE-Anweisungen anderer Sprachen. CASE wird verwendet, um ein Ergebnis anzugeben, wenn es mehrere Bedingungen gibt. Verwenden Sie CASE, wenn ein SQL-Ausdruck gilt, z. B. in einem SELECT-Befehl.

Es gibt zwei Arten von CASE-Ausdrücken: einfach und gesucht.

- In einfachen CASE-Ausdrücken wird ein Ausdruck mit einem Wert verglichen. Wenn keine Übereinstimmung gefunden wird, wird die in der THEN-Klausel angegebene Aktion angewendet. Wenn keine Übereinstimmung gefunden wird, wird die in der ELSE-Klausel angegebene Aktion angewendet.
- In gesuchten CASE-Ausdrücken wird jeder CASE-Ausdruck auf der Basis eines booleschen Ausdrucks evaluiert und die CASE-Anweisung gibt den ersten übereinstimmenden CASE-Ausdruck

zurück. Wenn in den WHEN-Klauseln kein übereinstimmender Ausdruck gefunden wird, wird die Aktion in der ELSE-Klausel zurückgegeben.

Syntax

Einfache CASE-Anweisung, um übereinstimmende Bedingungen zu finden:

```
CASE expression
  WHEN value THEN result
  [WHEN...]
  [ELSE result]
END
```

Gesuchte CASE-Anweisung, um jede Bedingung auszuwerten:

```
CASE
  WHEN condition THEN result
  [WHEN ...]
  [ELSE result]
END
```

Argumente

expression

Ein Spaltenname oder ein gültiger Ausdruck.

Wert

Wert, mit dem der Ausdruck verglichen wird, wie eine numerische Konstante oder eine Zeichenfolge.

Ergebnis

Der Zielwert oder -ausdruck, der zurückgegeben wird, wenn ein Ausdruck oder eine boolesche Bedingung ausgewertet werden. Die Datentypen aller Ergebnisausdrücke müssen in einen einzigen Ausgabetyp konvertierbar sein.

condition

Ein boolescher Ausdruck, der mit true oder false ausgewertet wird. Wenn die Bedingung mit true ausgewertet wird, ist der Wert des CASE-Ausdrucks das Ergebnis, das auf die Bedingung

folgt, und der Rest des CASE-Ausdrucks wird nicht verarbeitet. Wenn die Bedingung mit false ausgewertet wird, werden alle nachfolgenden WHEN-Klauseln ausgewertet. Wenn keine Ergebnisse der WHEN-Bedingung mit true ausgewertet werden, ist der Wert des CASE-Ausdrucks das Ergebnis der ELSE-Klausel. Wenn die ELSE-Klausel ausgelassen wurde und keine Bedingung mit true ausgewertet wird, ist das Ergebnis null.

Beispiele

In den folgenden Beispielen werden die Tabellen VENUE und SALES aus den TICKIT-Beispieldaten verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie einen einfachen CASE-Ausdruck, um `New York City` durch `Big Apple` in einer für die Tabelle VENUE ausgeführten Abfrage zu ersetzen. Alle anderen Städtenamen werden durch ersetzt `other`.

```
select venuecity,
  case venuecity
    when 'New York City'
    then 'Big Apple' else 'other'
  end
from venue
order by venueid desc;

venuecity        |   case
-----------------+-----------
Los Angeles      | other
New York City    | Big Apple
San Francisco    | other
Baltimore        | other
...
```

Verwendet einen gesuchten CASE-Ausdruck, um Gruppennummern basierend auf dem PRICEPAID-Wert für einzelne Ticketverkäufe zuzuweisen:

```
select pricepaid,
  case when pricepaid <10000 then 'group 1'
    when pricepaid >10000 then 'group 2'
    else 'group 3'
  end
from sales
order by 1 desc;
```

```
pricepaid |  case
----------+---------
12624     | group 2
10000     | group 3
10000     | group 3
9996      | group 1
9988      | group 1
...
```

DECODE-Funktion

Ein DECODE-Ausdruck ersetzt einen spezifischen Wert entweder durch einen anderen spezifischen Wert oder einen Standardwert, abhängig vom Ergebnis einer Gleichheitsbedingung. Diese Operation ist der Operation eines einfachen CASE-Ausdrucks oder einer IF-THEN-ELSE-Anweisung gleichwertig.

Syntax

```
DECODE ( expression, search, result [, search, result ]... [ ,default ] )
```

Diese Art von Ausdruck ist nützlich, um Abkürzungen oder Codes zu ersetzen, die in Tabellen mit relevanten Geschäftswerten gespeichert werden, die für Berichte benötigt werden.

Parameter

expression

Die Quelle des Werts, den Sie vergleichen möchten, beispielsweise eine Spalte in einer Tabelle.

search

Der Zielwert, der anhand des Quellausdrucks verglichen wird, beispielsweise ein numerischer Wert oder eine Zeichenfolge. Der Suchausdruck muss zu einem einzelnen festen Wert ausgewertet werden. Sie können keinen Ausdruck angeben, der zu einem Bereich von Werten ausgewertet wird, wie `age between 20 and 29`. Sie müssen für jeden Wert, den Sie ersetzen möchten, eigene Such-/Ergebnispaare angeben.

Der Datentyp aller Instances des Suchausdrucks muss derselbe oder kompatibel sein. Die Parameter expression und search müssen ebenfalls kompatibel sein.

Ergebnis

Der Ersetzungswert, den die Abfrage zurückgibt, wenn der Ausdruck mit dem Suchwert übereinstimmt. Sie müssen mindestens ein Such-/Ergebnispaar in den DECODE-Ausdruck einschließen.

Die Datentypen aller Instances des Ergebnisausdrucks müssen dieselben oder kompatibel sein. Die Parameter result und default müssen ebenfalls kompatibel sein.

default

Ein optionaler Standardwert, der verwendet wird, wenn die Suchbedingung fehlschlägt. Wenn Sie keinen Standardwert angeben, gibt der DECODE-Ausdruck NULL zurück.

Nutzungshinweise

Wenn die Werte für expression und search NULL sind, ist das DECODE-Ergebnis der entsprechende Wert für result. Im Beispielabschnitt wird diese Verwendung der Funktion veranschaulicht.

Wenn DECODE auf diese Weise verwendet wird, ist sie mit Funktion NVL2 vergleichbar. Es gibt jedoch einige Unterschiede. Eine Beschreibung dieser Unterschiede finden Sie in den Nutzungshinweisen für NVL2.

Beispiele

Wenn der Wert `2008-06-01` in der Spalte CALDATE von DATETABLE vorhanden ist, wird er im folgenden Beispiel durch `June 1st, 2008` ersetzt. Im Beispiel werden alle weiteren CALDATE-Werte durch NULL ersetzt.

```
select decode(caldate, '2008-06-01', 'June 1st, 2008')
from datetable where month='JUN' order by caldate;

case
---------------
June 1st, 2008

...
(30 rows)
```

Im folgenden Beispiel wird ein DECODE-Ausdruck verwendet, um die fünf abgekürzten CATNAME-Spalten in der Tabelle CATEGORY in die vollständigen Namen zu konvertieren und andere Werte in der Spalte in zu konvertieren `Unknown`.

```
select catid, decode(catname,
'NHL', 'National Hockey League',
'MLB', 'Major League Baseball',
'MLS', 'Major League Soccer',
'NFL', 'National Football League',
'NBA', 'National Basketball Association',
'Unknown')
from category
order by catid;

catid  | case
-------+---------------------------------
1      | Major League Baseball
2      | National Hockey League
3      | National Football League
4      | National Basketball Association
5      | Major League Soccer
6      | Unknown
7      | Unknown
8      | Unknown
9      | Unknown
10     | Unknown
11     | Unknown
(11 rows)
```

Verwenden Sie einen DECODE-Ausdruck, um Veranstaltungsorte in Colorado und Nevada mit NULL-Werten in der Spalte VENUESEATS zu finden: konvertieren Sie die NULL-Werte in Nullen. Wenn die Spalte VENUESEATS nicht NULL ist, wird 1 als Ergebnis zurückgegeben.

```
select venuename, venuestate, decode(venueseats,null,0,1)
from venue
where venuestate in('NV','CO')
order by 2,3,1;

venuename               | venuestate     | case
------------------------------+----------------+-----------
Coors Field                   | CO        |   1
Dick's Sporting Goods Park    | CO        |   1
Ellie Caulkins Opera House    | CO        |   1
INVESCO Field        | CO        |   1
Pepsi Center        | CO        |   1
Ballys Hotel        | NV        |   0
```

```
Bellagio Hotel                | NV        |   0
Caesars Palace                | NV        |   0
Harrahs Hotel                 | NV        |   0
Hilton Hotel                  | NV        |   0
...
(20 rows)
```

Funktionen GREATEST und LEAST

Gibt den größten oder kleinsten Wert aus einer Liste einer beliebigen Zahl von Ausdrücken zurück.

Syntax

```
GREATEST (value [, ...])
LEAST (value [, ...])
```

Parameter

expression_list

Eine durch Komma getrennte Liste von Ausdrücken, wie beispielsweise Spaltennamen. Die Ausdrücke müssen alle in einen gemeinsamen Datentyp konvertierbar sein. NULL-Werte in der Liste werden ignoriert. Wenn alle Ausdrücke zu NULL ausgewertet werden, ist das Ergebnis NULL.

Rückgabewert

Gibt den größten Wert (bei GREATEST) oder den kleinsten Wert (bei LEAST) aus der angegebenen Liste von Ausdrücken zurück.

Beispiel

Im folgenden Beispiel wird der höchste Wert alphabetisch für `firstname` oder `lastname` zurückgegeben.

```
select firstname, lastname, greatest(firstname,lastname) from users
where userid < 10
order by 3;

 firstname | lastname  | greatest
-----------+-----------+-----------
```

```
 Lars      | Ratliff   | Ratliff
 Reagan    | Hodge     | Reagan
 Colton    | Roy       | Roy
 Barry     | Roy       | Roy
 Tamekah   | Juarez    | Tamekah
 Rafael    | Taylor    | Taylor
 Victor    | Hernandez | Victor
 Vladimir  | Humphrey  | Vladimir
 Mufutau   | Watkins   | Watkins
(9 rows)
```

NVL- und COALESCE-Funktionen

Gibt den Wert des ersten Ausdrucks in einer Reihe von Ausdrücken zurück, der nicht null ist. Wenn ein Nicht-Null-Wert gefunden wird, werden die verbleibenden Ausdrücke in der Liste nicht ausgewertet.

NVL ist identisch mit COALESCE. Es sind Synonyme. Unter diesem Thema finden Sie eine Erläuterung der Syntax sowie Beispiele für beide.

Syntax

```
NVL( expression, expression, ... )
```

Die Syntax für COALESCE ist identisch:

```
COALESCE( expression, expression, ... )
```

Wenn alle Ausdrücke null sind, ist das Ergebnis null.

Diese Funktionen sind hilfreich, wenn Sie einen Sekundärwert zurückgeben möchten, falls ein Primärwert fehlt oder null ist. Eine Abfrage könnte beispielsweise die erste von drei verfügbaren Telefonnummern zurückgeben: Mobiltelefonnummer, private oder geschäftliche Telefonnummer. Die Reihenfolge der Ausdrücke in der Funktion bestimmt die Reihenfolge der Auswertung.

Argumente

expression

Ein Ausdruck (beispielsweise ein Spaltenname), der hinsichtlich des Null-Status ausgewertet werden soll.

Rückgabetyp

Amazon Redshift bestimmt den Datentyp des zurückgegebenen Werts anhand der Eingabeausdrücke. Wenn die Datentypen der Eingabeausdrücke keinen gemeinsamen Typ haben, wird ein Fehler zurückgegeben.

Beispiele

Wenn die Liste Ausdrücke mit Ganzzahlen enthält, gibt die Funktion eine Ganzzahl zurück.

```
SELECT COALESCE(NULL, 12, NULL);

coalesce
--------------
12
```

Dieses Beispiel, das im Gegensatz zum vorherigen Beispiel NVL verwendet, gibt dasselbe Ergebnis zurück.

```
SELECT NVL(NULL, 12, NULL);

coalesce
--------------
12
```

Im folgenden Beispiel wird einen Zeichenfolgetyp zurückgegeben.

```
SELECT COALESCE(NULL, 'Amazon Redshift', NULL);

coalesce
--------------
Amazon Redshift
```

Das folgende Beispiel führt zu einem Fehler, da die Datentypen in der Ausdrucksliste unterschiedlich sind. In diesem Fall enthält die Liste sowohl einen Zeichenfolgetyp als auch einen Zahlentyp.

```
SELECT COALESCE(NULL, 'Amazon Redshift', 12);
ERROR: invalid input syntax for integer: "Amazon Redshift"
```

Für dieses Beispiel erstellen Sie eine Tabelle mit den Spalten START_DATE und END_DATE, fügen Zeilen ein, die Null-Werte enthalten, und wenden anschließend einen NVL-Ausdruck auf die beiden Spalten an.

```
create table datetable (start_date date, end_date date);
insert into datetable values ('2008-06-01','2008-12-31');
insert into datetable values (null,'2008-12-31');
insert into datetable values ('2008-12-31',null);
```

```
select nvl(start_date, end_date)
from datetable
order by 1;

coalesce
------------
2008-06-01
2008-12-31
2008-12-31
```

Der Standardspaltenname für einen NVL-Ausdruck ist COALESCE. Die folgende Abfrage gibt dieselben Ergebnisse zurück:

```
select coalesce(start_date, end_date)
from datetable
order by 1;
```

Für die folgenden Beispielabfragen erstellen Sie eine Tabelle mit Beispielinformationen für Hotelbuchungen und fügen mehrere Zeilen ein. Einige Datensätze enthalten Nullwerte.

```
create table booking_info (booking_id int, booking_code character(8), check_in date,
 check_out date, funds_collected numeric(12,2));
```

Fügen Sie die folgenden Beispieldaten ein. Manche Datensätze weisen unter `check_out` kein Datum und unter `funds_collected` keinen Betrag auf.

```
insert into booking_info values (1, 'OCEAN_WV', '2023-02-01','2023-02-03',100.00);
insert into booking_info values (2, 'OCEAN_WV', '2023-04-22','2023-04-26',120.00);
insert into booking_info values (3, 'DSRT_SUN', '2023-03-13','2023-03-16',125.00);
insert into booking_info values (4, 'DSRT_SUN', '2023-06-01','2023-06-03',140.00);
insert into booking_info values (5, 'DSRT_SUN', '2023-07-10',null,null);
```

```
insert into booking_info values (6, 'OCEAN_WV', '2023-08-15',null,null);
```

Die folgende Abfrage gibt eine Liste von Daten zurück. Wenn das `check_out`-Datum nicht verfügbar ist, wird das `check_in`-Datum aufgeführt.

```
select coalesce(check_out, check_in)
from booking_info
order by booking_id;
```

Die Ergebnisse sehen wie folgt aus. Beachten Sie, dass die letzten beiden Datensätze das `check_in`-Datum anzeigen.

```
coalesce
------------
2023-02-03
2023-04-26
2023-03-16
2023-06-03
2023-07-10
2023-08-15
```

Wenn Sie erwarten, dass eine Abfrage für bestimmte Funktionen oder Spalten Null-Werte zurückgibt, können Sie einen NVL-Ausdruck verwenden, um die Null-Werte mit einem anderen Wert zu ersetzen. Beispielsweise geben Aggregationsfunktionen wie SUM Null-Werte anstelle von Nullen zurück, wenn es keine Zeilen gibt, die ausgewertet werden können. Sie können einen NVL-Ausdruck verwenden, um diese Null-Werte durch `700.0` zu ersetzen. Anstelle von `485` lautet das Ergebnis der Addition der Werte unter `funds_collected` `1885`, da zwei Zeilen, die den Wert null aufweisen, durch `700` ersetzt wurden.

```
select sum(nvl(funds_collected, 700.0)) as sumresult from booking_info;

sumresult
------
 1885
```

Funktion NVL2

Gibt einen von zwei Werten aus, je nachdem, ob ein angegebener Ausdruck zu NULL oder zu NOT NULL aufgelöst wird.

Syntax

```
NVL2 ( expression, not_null_return_value, null_return_value )
```

Argumente

expression

Ein Ausdruck (beispielsweise ein Spaltenname), der hinsichtlich des Null-Status ausgewertet werden soll.

not_null_return_value

Der Wert, der zurückgegeben wird, wenn expression zu NOT NULL ausgewertet wird. Der Wert not_null_return_value muss entweder denselben Datentyp wie expression haben oder implizit in diesen Datentyp konvertiert werden können.

null_return_value

Der Wert, der zurückgegeben wird, wenn expression zu NULL ausgewertet wird. Der Wert null_return_value muss entweder denselben Datentyp wie expression haben oder implizit in diesen Datentyp konvertiert werden können.

Rückgabetyp

Der NVL2-Rückgabetyp wird wie folgt festgelegt:

- Wenn not_null_return_value oder null_return_value null ist, wird der Datentyp des Nicht-Null-Ausdrucks zurückgegeben.

Wenn sowohl not_null_return_value als auch null_return_value nicht null sind:

- Wenn not_null_return_value und null_return_value denselben Datentyp haben, wird dieser Datentyp zurückgegeben.
- Wenn not_null_return_value und null_return_value unterschiedliche numerische Datentypen haben, wird der kleinste kompatible numerische Datentyp zurückgegeben.
- Wenn not_null_return_value und null_return_value unterschiedliche Datum-/Uhrzeit-Datentypen haben, wird ein Zeitstempeldatentyp zurückgegeben.
- Wenn not_null_return_value und null_return_value unterschiedliche Zeichendatentypen haben, wird der Datentyp von not_null_return_value zurückgegeben.

- Wenn not_null_return_value und null_return_value gemischte numerische und nicht numerische Datentypen haben, wird der Datentyp von not_null_return_value zurückgegeben.

> **Important**
>
> In den letzten beiden Fällen, in denen der Datentyp von not_null_return_value zurückgegeben wird, wird null_return_value implizit in diesen Datentyp umgewandelt. Wenn die Datentypen nicht kompatibel sind, schlägt die Funktion fehl.

Nutzungshinweise

DECODE-Funktion kann ähnlich wie NVL2 verwendet werden, wenn die Parameter expression und search beide null sind. Der Unterschied besteht darin, dass die Rückgabe für DECODE sowohl den Wert als auch den Datentyp des Parameters result aufweist. Die Rückgabe für NVL2 weist hingegen entweder den Wert des Parameters not_null_return_value oder des Parameters null_return_value auf, je nachdem, welcher Parameter von der Funktion ausgewählt wird. Der Datentyp ist jedoch not_null_return_value.

Wenn beispielsweise column1 NULL ist, geben die folgenden Abfragen denselben Wert zurück. Der Datentyp des DECODE-Rückgabewerts ist jedoch INTEGER und der Datentyp des NVL2-Rückgabewerts ist VARCHAR.

```
select decode(column1, null, 1234, '2345');
select nvl2(column1, '2345', 1234);
```

Beispiel

Im folgenden Beispiel werden einige Beispieldaten modifiziert und anschließend zwei Felder ausgewertet, um die richtigen Kontaktinformationen für Benutzer bereitzustellen:

```
update users set email = null where firstname = 'Aphrodite' and lastname = 'Acevedo';

select (firstname + ' ' + lastname) as name,
nvl2(email, email, phone) AS contact_info
from users
where state = 'WA'
and lastname  like 'A%'
order by lastname, firstname;
```

```
name        contact_info
--------------------+-------------------------------------------
Aphrodite Acevedo (906) 632-4407
Caldwell Acevedo  Nunc.sollicitudin@Duisac.ca
Quinn Adams    vel@adipiscingligulaAenean.com
Kamal Aguilar  quis@vulputaterisusa.com
Samson Alexander  hendrerit.neque@indolorFusce.ca
Hall Alford    ac.mattis@vitaediamProin.edu
Lane Allen     et.netus@risusDonec.org
Xander Allison    ac.facilisis.facilisis@Infaucibus.com
Amaya Alvarado    dui.nec.tempus@eudui.edu
Vera Alvarez   at.arcu.Vestibulum@pellentesque.edu
Yetta Anthony  enim.sit@risus.org
Violet Arnold  ad.litora@at.com
August Ashley  consectetuer.euismod@Phasellus.com
Karyn Austin   ipsum.primis.in@Maurisblanditenim.org
Lucas Ayers    at@elitpretiumet.com
```

NULLIF-Funktion

Syntax

Der NULLIF-Ausdruck vergleicht zwei Argumente und gibt null zurück, wenn die Argumente gleich sind. Wenn sie nicht gleich sind, wird das erste Argument zurückgegeben. Dieser Ausdruck ist die Umkehrung des NVL- oder COALESCE-Ausdrucks.

```
NULLIF ( expression1, expression2 )
```

Argumente

expression1, expression2

Die Zielspalten oder -ausdrücke, die verglichen werden. Der Rückgabetyp ist mit dem Typ des ersten Ausdrucks identisch. Der Standardspaltenname des NULLIF-Ergebnisses ist der Spaltenname des ersten Ausdrucks.

Beispiele

Im folgenden Beispiel gibt die Abfrage die Zeichenfolge `first` zurück, da die Argumente nicht identisch sind.

```
SELECT NULLIF('first', 'second');
```

```
case
-------
first
```

Im folgenden Beispiel gibt die Abfrage NULL zurück, da die Argumente des Zeichenfolgeliterals identisch sind.

```
SELECT NULLIF('first', 'first');

case
-------
NULL
```

Im folgenden Beispiel gibt die Abfrage 1 zurück, da die Ganzzahlargumente nicht identisch sind.

```
SELECT NULLIF(1, 2);

case
-------
1
```

Im folgenden Beispiel gibt die Abfrage NULL zurück, da die Ganzzahlargumente identisch sind.

```
SELECT NULLIF(1, 1);

case
-------
NULL
```

Im folgenden Beispiel gibt die Abfrage null zurück, wenn die LISTID- und SALESID-Werte übereinstimmen:

```
select nullif(listid,salesid), salesid
from sales where salesid<10 order by 1, 2 desc;

listid  | salesid
--------+---------
     4  |       2
     5  |       4
     5  |       3
     6  |       5
```

```
     10 |       9
     10 |       8
     10 |       7
     10 |       6
        |       1
(9 rows)
```

Sie können NULLIF verwenden, um sicherzustellen, dass leere Zeichenfolgen stets als Null-Werte zurückgegeben werden. Im folgenden Beispiel gibt der NULLIF-Ausdruck entweder einen Null-Wert oder eine Zeichenfolge zurück, die mindestens ein Zeichen enthalten.

```
insert into category
values(0,'','Special','Special');

select nullif(catgroup,'') from category
where catdesc='Special';

catgroup
----------
null
(1 row)
```

NULLIF ignoriert am Ende stehende Leerzeichen. Wenn eine Zeichenfolge nicht leer ist, aber Leerzeichen enthält, gibt NULLIF ebenfalls null zurück:

```
create table nulliftest(c1 char(2), c2 char(2));

insert into nulliftest values ('a','a ');

insert into nulliftest values ('b','b');


select nullif(c1,c2) from nulliftest;
c1
------
null
null
(2 rows)
```

Funktionen für die Datentypformatierung

Themen

- CAST-Funktion
- CONVERT-Funktion
- TO_CHAR
- TO_DATE-Funktion
- TO_NUMBER
- TEXT_TO_INT_ALT
- TEXT_TO_NUMERIC_ALT
- Datum-/Uhrzeit-Formatzeichenfolgen
- Numerische Formatzeichenfolgen
- Formatierungszeichen im Teradata-Stil für numerische Daten

Funktionen für die Datentypformatierung bieten eine einfache Möglichkeit, Werte von einem Datentyp in einen anderen zu konvertieren. Bei jeder dieser Funktionen ist das erste Argument immer der zu formatierende Wert, und das zweite Argument enthält die Vorlage für das neue Format. Amazon Redshift unterstützt verschiedene Funktionen für die Datentypformatierung.

CAST-Funktion

Die CAST-Funktion konvertiert einen Datentyp in einen anderen kompatiblen Datentyp. Sie können beispielsweise eine Zeichenfolge in ein Datum oder einen numerischen Typ in eine Zeichenfolge konvertieren. CAST führt eine Laufzeitkonvertierung durch, was bedeutet, dass die Konvertierung den Datentyp eines Werts in einer Quelltabelle nicht ändert. Dieser wird nur im Kontext der Abfrage geändert.

Die CAST-Funktion ist the section called “CONVERT” insofern sehr ähnlich, als beide Funktionen einen Datentyp in einen anderen konvertieren. Die beiden Funktionen werden jedoch unterschiedlich aufgerufen.

Bestimmte Datentypen erfordern eine explizite Konvertierung in andere Datentypen unter Verwendung der Funktionen CAST oder CONVERT. Andere Datentypen können implizit als Teil eines anderen Befehls konvertiert werden, ohne CAST oder CONVERT zu verwenden. Siehe Kompatibilität von Typen und Umwandlung zwischen Typen.

Syntax

Verwenden Sie eine dieser beiden gleichwertigen Syntaxformate, um Ausdrücke von einem Datentyp in einen anderen umzuwandeln.

```
CAST ( expression AS type )
expression :: type
```

Argumente

expression

Ein Ausdruck, der einen oder mehrere Werte auswertet, beispielsweise ein Spaltenname oder ein Literal. Die Konvertierung von Null-Werten gibt Null-Werte zurück. Der Ausdruck darf keine leeren Zeichenfolgen enthalten.

Typ

Einer der unterstützten Datentypen.

Rückgabetyp

CAST gibt den Datentyp zurück, der durch das Argument type angegeben ist.

> **Note**
>
> Amazon Redshift gibt einen Fehler zurück, wenn Sie versuchen, eine problematische Konvertierung durchzuführen, beispielsweise die folgende DECIMAL-Konvertierung, die Präzision verliert:
>
> ```
> select 123.456::decimal(2,1);
> ```
>
> oder eine INTEGER-Konvertierung, die einen Overflow verursacht:
>
> ```
> select 12345678::smallint;
> ```

Beispiele

Einige der Beispiele verwenden die Beispieldatenbank TICKIT. Weitere Informationen zum Einrichten von Beispieldaten finden Sie unter Daten laden.

Die folgenden beiden Abfragen sind gleichwertig. Beide wandeln einen Dezimalwert in eine Ganzzahl um:

```
select cast(pricepaid as integer)
from sales where salesid=100;

pricepaid
-----------
162
(1 row)
```

```
select pricepaid::integer
from sales where salesid=100;

pricepaid
-----------
162
(1 row)
```

Das Folgende führt zu einem ähnlichen Ergebnis. Für die Ausführung sind keine Beispieldaten erforderlich:

```
select cast(162.00 as integer) as pricepaid;

pricepaid
-----------
162
(1 row)
```

In diesem Beispiel werden die Werte in einer Zeitstempelspalte in Datumsangaben umgewandelt, was dazu führt, dass die Uhrzeit aus jedem Ergebnis entfernt wird:

```
select cast(saletime as date), salesid
from sales order by salesid limit 10;

 saletime  | salesid
-----------+---------
2008-02-18 |       1
2008-06-06 |       2
2008-06-06 |       3
2008-06-09 |       4
2008-08-31 |       5
2008-07-16 |       6
2008-06-26 |       7
```

```
2008-07-10 |       8
2008-07-22 |       9
2008-08-06 |      10
(10 rows)
```

Wenn Sie CAST nicht wie im vorherigen Beispiel dargestellt verwendet haben, würden die Ergebnisse die Uhrzeit umfassen: 2008-02-18 02:36:48.

Die folgende Abfrage wandelt variable Zeichendaten in ein Datum um. Für die Ausführung sind keine Beispieldaten erforderlich.

```
select cast('2008-02-18 02:36:48' as date) as mysaletime;

mysaletime
--------------------
2008-02-18
(1 row)
```

In diesem Beispiel werden die Werte in einer Datumsspalte in Zeitstempel umgewandelt:

```
select cast(caldate as timestamp), dateid
from date order by dateid limit 10;

      caldate       | dateid
--------------------+--------
2008-01-01 00:00:00 |   1827
2008-01-02 00:00:00 |   1828
2008-01-03 00:00:00 |   1829
2008-01-04 00:00:00 |   1830
2008-01-05 00:00:00 |   1831
2008-01-06 00:00:00 |   1832
2008-01-07 00:00:00 |   1833
2008-01-08 00:00:00 |   1834
2008-01-09 00:00:00 |   1835
2008-01-10 00:00:00 |   1836
(10 rows)
```

In einem Fall wie im vorherigen Beispiel können Sie mithilfe von TO_CHAR zusätzliche Kontrolle über die Ausgabeformatierung erhalten.

In diesem Beispiel wird eine Ganzzahl in eine Zeichenfolge umgewandelt:

```
select cast(2008 as char(4));

bpchar
--------
2008
```

In diesem Beispiel wird ein DECIMAL(6,3)-Wert in einen DECIMAL(4,1)-Wert umgewandelt:

```
select cast(109.652 as decimal(4,1));

numeric
---------
109.7
```

Dieses Beispiel zeigt einen komplexeren Ausdruck. Die Spalte PRICEPAID (eine DECIMAL(8,2)-Spalte) in der Tabelle SALES wird in eine DECIMAL(38,2)-Spalte umgewandelt und die Werte werden mit 100000000000000000000 multipliziert:

```
select salesid, pricepaid::decimal(38,2)*100000000000000000000
as value from sales where salesid<10 order by salesid;


 salesid |           value
---------+----------------------------
       1 | 728000000000000000000000.00
       2 |  760000000000000000000000.00
       3 | 350000000000000000000000.00
       4 | 175000000000000000000000.00
       5 | 154000000000000000000000.00
       6 | 394000000000000000000000.00
       7 | 788000000000000000000000.00
       8 | 197000000000000000000000.00
       9 | 591000000000000000000000.00
(9 rows)
```

> **Note**
>
> Sie können keine CAST- oder CONVERT-Operation für den `GEOMETRY`-Datentyp durchführen, um ihn in einen anderen Datentyp zu ändern. Sie können jedoch eine hexadezimale Darstellung eines String-Literals im EWKB-Format (Extended Well-Known

Binary) als Parameter für Funktionen bereitstellen, die ein GEOMETRY-Argument akzeptieren. Beispielsweise erwartet die folgende ST_AsText-Funktion den Datentyp GEOMETRY.

```
SELECT ST_AsText('010100000000000000000001C40000000000002040');
```

```
st_astext
------------
 POINT(7 8)
```

Sie können außerdem explizit den Datentyp GEOMETRY angeben.

```
SELECT ST_AsText('010100000000000000000014400000000000001840'::geometry);
```

```
st_astext
------------
 POINT(5 6)
```

CONVERT-Funktion

Wie die CAST-Funktion konvertiert die CONVERT-Funktion einen Datentyp in einen anderen kompatiblen Datentyp. Sie können beispielsweise eine Zeichenfolge in ein Datum oder einen numerischen Typ in eine Zeichenfolge konvertieren. CONVERT führt eine Laufzeitkonvertierung durch, was bedeutet, dass die Konvertierung den Datentyp eines Werts in einer Quelltabelle nicht ändert. Dieser wird nur im Kontext der Abfrage geändert.

Bestimmte Datentypen erfordern eine explizite Konvertierung in andere Datentypen unter Verwendung der CONVERT-Funktion. Andere Datentypen können implizit als Teil eines anderen Befehls konvertiert werden, ohne CAST oder CONVERT zu verwenden. Siehe Kompatibilität von Typen und Umwandlung zwischen Typen.

Syntax

```
CONVERT ( type, expression )
```

Argumente

Typ

Einer der unterstützten Datentypen.

expression

Ein Ausdruck, der einen oder mehrere Werte auswertet, beispielsweise ein Spaltenname oder ein Literal. Die Konvertierung von Null-Werten gibt Null-Werte zurück. Der Ausdruck darf keine leeren Zeichenfolgen enthalten.

Rückgabetyp

CONVERT gibt den Datentyp zurück, der durch das Argument type angegeben ist.

> **Note**
>
> Amazon Redshift gibt einen Fehler zurück, wenn Sie versuchen, eine problematische Konvertierung durchzuführen, beispielsweise die folgende DECIMAL-Konvertierung, die Präzision verliert:
>
> ```
> SELECT CONVERT(decimal(2,1), 123.456);
> ```
>
> oder eine INTEGER-Konvertierung, die einen Overflow verursacht:
>
> ```
> SELECT CONVERT(smallint, 12345678);
> ```

Beispiele

Einige der Beispiele verwenden die Beispieldatenbank TICKIT. Weitere Informationen zum Einrichten von Beispieldaten finden Sie unter Daten laden.

Die folgende Abfrage verwendet die CONVERT-Funktion, um eine Spalte mit Dezimalzahlen in Ganzzahlen zu konvertieren.

```
SELECT CONVERT(integer, pricepaid)
FROM sales WHERE salesid=100;
```

In diesem Beispiel wird eine Ganzzahl in eine Zeichenfolge konvertiert.

```
SELECT CONVERT(char(4), 2008);
```

In diesem Beispiel werden das aktuelle Datum und die aktuelle Uhrzeit in einen variablen Zeichendatentyp konvertiert:

```
SELECT CONVERT(VARCHAR(30), GETDATE());

getdate
---------
2023-02-02 04:31:16
```

In diesem Beispiel wird die Saletime-Spalte in eine reine Zeitspalte konvertiert, wobei die Datumsangaben aus jeder Zeile entfernt werden.

```
SELECT CONVERT(time, saletime), salesid
FROM sales order by salesid limit 10;
```

Informationen zum Konvertieren eines Zeitstempels von einer Zeitzone in eine andere finden Sie unter Funktion CONVERT_TIMEZONE. Weitere Datums- und Uhrzeitfunktionen finden Sie unter Datums- und Zeitfunktionen.

Im folgenden Beispiel werden variable Zeichendaten in ein Datetime-Objekt konvertiert.

```
SELECT CONVERT(datetime, '2008-02-18 02:36:48') as mysaletime;
```

> **Note**
>
> Sie können keine CAST- oder CONVERT-Operation für den `GEOMETRY`-Datentyp durchführen, um ihn in einen anderen Datentyp zu ändern. Sie können jedoch eine hexadezimale Darstellung eines String-Literals im EWKB-Format (Extended Well-Known Binary) als Parameter für Funktionen bereitstellen, die ein `GEOMETRY`-Argument akzeptieren. Beispielsweise erwartet die folgende `ST_AsText`-Funktion den Datentyp `GEOMETRY`.
>
> ```
> SELECT ST_AsText('010100000000000000000001C400000000000002040');
> ```
>
> ```
> st_astext
> ```

```
------------
 POINT(7 8)
```

Sie können außerdem explizit den Datentyp GEOMETRY angeben.

```
SELECT ST_AsText('010100000000000000000014400000000000001840'::geometry);
```

```
st_astext
------------
 POINT(5 6)
```

TO_CHAR

TO_CHAR konvertiert einen Zeitstempel oder numerischen Ausdruck in ein Zeichenfolgendatenformat.

Syntax

```
TO_CHAR (timestamp_expression | numeric_expression , 'format')
```

Argumente

timestamp_expression

Ein Ausdruck, der einen TIMESTAMP- oder TIMESTAMPTZ-Typwert als Ergebnis hat oder einen Wert, der implizit zu einem Zeitstempel gezwungen werden kann.

numeric_expression

Ein Ausdruck, der einen numerischen Datentypwert als Ergebnis hat oder einen Wert, der implizit zu einem numerischen Typ gezwungen werden kann. Weitere Informationen finden Sie unter Numerische Typen. „TO_CHAR“ fügt links von der Zahlenfolge ein Leerzeichen ein.

> **Note**
>
> TO_CHAR unterstützt keine 128-Bit-Dezimalwerte.

format

Das Format für den neuen Wert. Informationen zu gültigen Formaten finden Sie unter Datum-/Uhrzeit-Formatzeichenfolgen und Numerische Formatzeichenfolgen.

Rückgabetyp

VARCHAR

Beispiele

Im folgenden Beispiel wird ein Zeitstempel in einen Wert mit Datum und Uhrzeit konvertiert, dessen Format den Namen des Monats auf neun Zeichen aufgefüllt, den Namen des Wochentages und die Tagesnummer des Monats enthält.

```
select to_char(timestamp '2009-12-31 23:15:59', 'MONTH-DY-DD-YYYY HH12:MIPM');

to_char
-------------------------
DECEMBER -THU-31-2009 11:15PM
```

Im folgenden Beispiel wird ein Zeitstempel in einen Wert mit Tageszahl des Jahres konvertiert.

```
select to_char(timestamp '2009-12-31 23:15:59', 'DDD');

to_char
-------------------------
365
```

Im folgenden Beispiel wird ein Zeitstempel in einen Wert mit ISO-Tageszahl der Woche konvertiert.

```
select to_char(timestamp '2022-05-16 23:15:59', 'ID');

to_char
-------------------------
1
```

Im folgenden Beispiel wird der Monat aus einem Datumswert extrahiert.

```
select to_char(date '2009-12-31', 'MONTH');
```

```
to_char
-------------------------
DECEMBER
```

Im folgenden Beispiel wird jeder STARTTIME-Wert in der Tabelle EVENT in eine Zeichenfolge konvertiert, die aus Stunden, Minuten und Sekunden besteht.

```
select to_char(starttime, 'HH12:MI:SS')
from event where eventid between 1 and 5
order by eventid;

to_char
----------
02:30:00
08:00:00
02:30:00
02:30:00
07:00:00
```

Im folgenden Beispiel wird ein ganzer Zeitstempelwert in ein anderes Format konvertiert.

```
select starttime, to_char(starttime, 'MON-DD-YYYY HH12:MIPM')
from event where eventid=1;

      starttime      |       to_char
---------------------+---------------------
 2008-01-25 14:30:00 | JAN-25-2008 02:30PM
```

Im folgenden Beispiel wird ein Zeitstempelliteral in eine Zeichenfolge konvertiert.

```
select to_char(timestamp '2009-12-31 23:15:59','HH24:MI:SS');

to_char
----------
23:15:59
```

Im folgenden Beispiel wird eine Dezimalzahl in eine Zeichenfolge konvertiert.

```
select to_char(125.8, '999.99');
```

```
to_char
---------
125.80
```

Im folgenden Beispiel wird eine Dezimalzahl in eine Zeichenfolge konvertiert.

```
select to_char(125.8, '999D99');

to_char
---------
125.80
```

Das folgende Beispiel konvertiert eine Zahl in eine Zeichenfolge mit einer führenden Null.

```
select to_char(125.8, '0999D99');

to_char
---------
0125.80
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge mit dem Minuszeichen am Ende konvertiert.

```
select to_char(-125.8, '999D99S');

to_char
---------
125.80-
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge mit dem positiven oder negativen Vorzeichen an der angegebenen Position konvertiert.

```
select to_char(125.8, '999D99SG');

to_char
---------
125.80+
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge mit dem positiven Vorzeichen an der angegebenen Position konvertiert.

```
select to_char(125.8, 'PL999D99');

to_char
---------
+ 125.80
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge mit dem Währungssymbol konvertiert.

```
select to_char(-125.88, '$S999D99');

to_char
---------
$-125.88
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge mit dem Währungssymbol an der angegebenen Position konvertiert.

```
select to_char(-125.88, 'S999D99L');

to_char
---------
-125.88$
```

Im folgenden Beispiel wird eine Zahl mithilfe eines Tausendertrennzeichens (Komma) in eine Zeichenfolge konvertiert.

```
select to_char(1125.8, '9,999.99');

to_char
---------
1,125.80
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge konvertiert, bei dem Eckige Klammern als negative Zahlen verwendet werden.

```
select to_char(-125.88, '$999D99PR');

to_char
---------
$<125.88>
```

Im folgenden Beispiel wird eine Zahl in eine Zeichenfolge römischer Zahlen konvertiert.

```
select to_char(125, 'RN');

to_char
---------
   CXXV
```

Im folgenden Beispiel wird ein Datum in einen Jahrhundertcode konvertiert.

```
select to_char(date '2020-12-31', 'CC');

to_char
---------
21
```

Im folgenden Beispiel wird der Wochentag angezeigt.

```
SELECT to_char(current_timestamp, 'FMDay, FMDD HH12:MI:SS');

to_char
-----------------------
Wednesday, 31 09:34:26
```

Im folgenden Beispiel wird das Ordnungszahlsuffix für eine Zahl angezeigt.

```
SELECT to_char(482, '999th');

to_char
-----------------------
 482nd
```

Im folgenden Beispiel wird in der Tabelle SALES die Provision vom gezahlten Preis abgezogen. Die Differenz wird anschließend nach oben gerundet und in eine römische Ziffer konvertiert, die in der Spalte TO_CHAR angezeigt wird:

```
select salesid, pricepaid, commission, (pricepaid - commission)
as difference, to_char(pricepaid - commission, 'rn') from sales
group by sales.pricepaid, sales.commission, salesid
order by salesid limit 10;
```

```
 salesid | pricepaid | commission | difference |     to_char
---------+-----------+------------+------------+-----------------
       1 |    728.00 |     109.20 |     618.80 |           dcxix
       2 |     76.00 |      11.40 |      64.60 |             lxv
       3 |    350.00 |      52.50 |     297.50 |        ccxcviii
       4 |    175.00 |      26.25 |     148.75 |           cxlix
       5 |    154.00 |      23.10 |     130.90 |           cxxxi
       6 |    394.00 |      59.10 |     334.90 |         cccxxxv
       7 |    788.00 |     118.20 |     669.80 |           dclxx
       8 |    197.00 |      29.55 |     167.45 |          clxvii
       9 |    591.00 |      88.65 |     502.35 |             dii
      10 |     65.00 |       9.75 |      55.25 |              lv
```

Im folgenden Beispiel wird den Differenzwerten, die in der Spalte TO_CHAR angezeigt werden, das Währungssymbol hinzugefügt:

```
select salesid, pricepaid, commission, (pricepaid - commission)
as difference, to_char(pricepaid - commission, 'l99999D99') from sales
group by sales.pricepaid, sales.commission, salesid
order by salesid limit 10;

salesid | pricepaid | commission | difference |  to_char
--------+-----------+------------+------------+------------
      1 |    728.00 |     109.20 |     618.80 | $   618.80
      2 |     76.00 |      11.40 |      64.60 | $    64.60
      3 |    350.00 |      52.50 |     297.50 | $   297.50
      4 |    175.00 |      26.25 |     148.75 | $   148.75
      5 |    154.00 |      23.10 |     130.90 | $   130.90
      6 |    394.00 |      59.10 |     334.90 | $   334.90
      7 |    788.00 |     118.20 |     669.80 | $   669.80
      8 |    197.00 |      29.55 |     167.45 | $   167.45
      9 |    591.00 |      88.65 |     502.35 | $   502.35
     10 |     65.00 |       9.75 |      55.25 | $    55.25
```

Im folgenden Beispiel wird das Jahrhundert aufgelistet, in dem die einzelnen Verkäufe ausgeführt wurden.

```
select salesid, saletime, to_char(saletime, 'cc') from sales
order by salesid limit 10;

 salesid |      saletime       | to_char
```

```
---------+---------------------+---------
       1 | 2008-02-18 02:36:48 | 21
       2 | 2008-06-06 05:00:16 | 21
       3 | 2008-06-06 08:26:17 | 21
       4 | 2008-06-09 08:38:52 | 21
       5 | 2008-08-31 09:17:02 | 21
       6 | 2008-07-16 11:59:24 | 21
       7 | 2008-06-26 12:56:06 | 21
       8 | 2008-07-10 02:12:36 | 21
       9 | 2008-07-22 02:23:17 | 21
      10 | 2008-08-06 02:51:55 | 21
```

Im folgenden Beispiel wird jeder STARTTIME-Wert in der Tabelle EVENT in eine Zeichenfolge konvertiert, die aus Stunden, Minuten, Sekunden und Zeitzone besteht.

```
select to_char(starttime, 'HH12:MI:SS TZ')
from event where eventid between 1 and 5
order by eventid;

to_char
----------
02:30:00 UTC
08:00:00 UTC
02:30:00 UTC
02:30:00 UTC
07:00:00 UTC
```

Im folgenden Beispiel wird die Formatierung für Sekunden, Millisekunden und Mikrosekunden gezeigt.

```
select sysdate,
to_char(sysdate, 'HH24:MI:SS') as seconds,
to_char(sysdate, 'HH24:MI:SS.MS') as milliseconds,
to_char(sysdate, 'HH24:MI:SS:US') as microseconds;

timestamp           | seconds  | milliseconds | microseconds
--------------------+----------+--------------+----------------
2015-04-10 18:45:09 | 18:45:09 | 18:45:09.325 | 18:45:09:325143
```

TO_DATE-Funktion

TO_DATE konvertiert ein Datum in einer Zeichenfolge in den Datentyp DATE.

Syntax

```
TO_DATE(string, format)
```

```
TO_DATE(string, format, is_strict)
```

Argumente

string

Eine Zeichenfolge, die konvertiert werden soll.

format

Ein Zeichenfolgeliteral, das das Format der Zeichenfolge in der Eingabezeichenfolge in Bezug auf die Datumsabschnitte definiert. Eine Liste der gültigen Formate für Tag, Monat und Jahr finden Sie unter Datum-/Uhrzeit-Formatzeichenfolgen.

is_strict

Ein optionaler boolescher Wert, der angibt, ob ein Fehler zurückgegeben wird, wenn ein Eingabedatumswert außerhalb des zulässigen Bereichs liegt. Wenn is_strict auf TRUE gesetzt wird, wird ein Fehler zurückgegeben, wenn ein Wert außerhalb des zulässigen Bereichs liegt. Wenn is_strict auf FALSE gesetzt wird, was die Standardeinstellung ist, sind Überlaufwerte zulässig.

Rückgabetyp

TO_DATE gibt ein DATE zurück, abhängig vom Formatwert.

Wenn die Konvertierung in das Format fehlschlägt, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung konvertiert das Datum `02 Oct 2001` in einem Datumsdatentyp.

```
select to_date('02 Oct 2001', 'DD Mon YYYY');

to_date
------------
2001-10-02
(1 row)
```

Die folgende SQL-Anweisung konvertiert die Zeichenfolge 20010631 in ein Datum.

```
select to_date('20010631', 'YYYYMMDD', FALSE);
```

Das Ergebnis ist der 1. Juli 2001, da der Juni nur 30 Tage hat.

```
to_date
------------
2001-07-01
```

Die folgende SQL-Anweisung konvertiert die Zeichenfolge 20010631 in ein Datum:

```
to_date('20010631', 'YYYYMMDD', TRUE);
```

Das Ergebnis ist ein Fehler, da der Juni nur 30 Tage hat.

```
ERROR:  date/time field date value out of range: 2001-6-31
```

TO_NUMBER

TO_NUMBER konvertiert eine Zeichenfolge in einen numerischen Wert (Dezimalwert).

Syntax

```
to_number(string, format)
```

Argumente

string

Die Zeichenfolge, die konvertiert werden soll. Das Format muss ein Literalwert sein.

format

Das zweite Argument ist eine Formatzeichenfolge, die anzeigt, wie die Zeichenfolge analysiert werden muss, um den numerischen Wert zu generieren. Beispielsweise gibt das Format '99D999' an, dass die Zeichenfolge, die konvertiert werden soll, aus fünf Ziffern mit dem Dezimalzeichen an dritter Position besteht. Beispielsweise gibt to_number('12.345','99D999') 12.345 als einen numerischen Wert zurück. Die Liste der gültigen Formate finden Sie unter Numerische Formatzeichenfolgen.

Rückgabetyp

TO_NUMBER gibt eine Dezimalzahl zurück.

Wenn die Konvertierung in das Format fehlschlägt, wird ein Fehler zurückgegeben.

Beispiele

Im folgenden Beispiel wird die Zeichenfolge `12,454.8-` in eine Zahl konvertiert:

```
select to_number('12,454.8-', '99G999D9S');

to_number
-----------
-12454.8
```

Im folgenden Beispiel wird die Zeichenfolge `$ 12,454.88` in eine Zahl konvertiert:

```
select to_number('$ 12,454.88', 'L 99G999D99');

to_number
-----------
12454.88
```

Im folgenden Beispiel wird die Zeichenfolge `$ 2,012,454.88` in eine Zahl konvertiert:

```
select to_number('$ 2,012,454.88', 'L 9,999,999.99');

to_number
-----------
2012454.88
```

TEXT_TO_INT_ALT

TEXT_TO_INT_ALT konvertiert mit der Formatierung im Teradata-Stil Zeichenfolgen in Ganzzahlen. Nachkommastellen im Ergebnis werden abgeschnitten.

Syntax

```
TEXT_TO_INT_ALT (expression [ , 'format'])
```

Argumente

expression

Ein Ausdruck, der einen oder mehrere CHAR- oder VARCHAR-Werte als Ergebnis hat, beispielsweise ein Spaltenname oder eine Literalzeichenfolge. Die Konvertierung von Null-Werten gibt Null-Werte zurück. Die Funktion wandelt leere Zeichenfolgen in 0 um.

format

Ein Zeichenfolgeliteral, das das Format des Eingabeausdrucks definiert. Weitere Informationen zu den Formatierungszeichen, die Sie angeben können, finden Sie unter Formatierungszeichen im Teradata-Stil für numerische Daten.

Rückgabetyp

TEXT_TO_INT_ALT gibt einen INTEGER-Wert zurück.

Nachkommastellen des Umwandlungsergebnisses werden abgeschnitten.

Amazon Redshift gibt einen Fehler zurück, wenn die Umwandlung der von ihnen angegebenen format-Phrase fehlschlägt.

Beispiele

Im folgenden Beispiel wird die expression-Eingabezeichenfolge '123-' in die Ganzzahl -123 umgewandelt.

```
select text_to_int_alt('123-');
```

```
text_to_int_alt
----------
      -123
```

Im folgenden Beispiel wird die expression-Eingabezeichenfolge '2147483647+' in die Ganzzahl 2147483647 umgewandelt.

```
select text_to_int_alt('2147483647+');
```

```
text_to_int_alt
----------
```

```
2147483647
```

Im folgenden Beispiel wird die exponentielle expression-Eingabezeichenfolge '-123E-2' in die Ganzzahl -1 umgewandelt.

```
select text_to_int_alt('-123E-2');
```

```
text_to_int_alt
----------
        -1
```

Im folgenden Beispiel wird die expression-Eingabezeichenfolge '2147483647+' in die Ganzzahl 2147483647 umgewandelt.

```
select text_to_int_alt('2147483647+');
```

```
text_to_int_alt
----------
2147483647
```

Im folgenden Beispiel wird die expression-Eingabezeichenfolge '123{' mit der format-Phrase '999S' in die Ganzzahl 1230 umgewandelt. Das S-Zeichen gibt ein Signed Zoned Decimal an. Weitere Informationen finden Sie unter Formatierungszeichen im Teradata-Stil für numerische Daten.

```
text_to_int_alt('123{', '999S');
```

```
text_to_int_alt
----------
      1230
```

Im folgenden Beispiel wird die expression-Eingabezeichenfolge 'USD123' mit der format-Phrase 'C9(I)' in die Ganzzahl 123 umgewandelt. Siehe Formatierungszeichen im Teradata-Stil für numerische Daten.

```
text_to_int_alt('USD123', 'C9(I)');
```

```
text_to_int_alt
----------
```

```
       123
```

Das folgende Beispiel gibt eine Tabellenspalte als Eingabe-expression an.

```
select text_to_int_alt(a), text_to_int_alt(b) from t_text2int order by 1;
```

```
 text_to_int_alt | text_to_int_alt
-----------------+-----------------
            -123 |            -123
            -123 |            -123
             123 |             123
             123 |             123
```

Im Folgenden sehen Sie die Tabellendefinition und die insert-Anweisung für dieses Beispiel.

```
create table t_text2int (a varchar(200), b char(200));
```

```
insert into t_text2int VALUES('123', '123'),('123.123', '123.123'), ('-123', '-123'),
 ('123-', '123-');
```

TEXT_TO_NUMERIC_ALT

TEXT_TO_NUMERIC_ALT führt einen Umwandlungsvorgang im Teradata-Stil aus, um eine Zeichenfolge in ein numerisches Datenformat zu konvertieren.

Syntax

```
TEXT_TO_NUMERIC_ALT (expression [, 'format'] [, precision, scale])
```

Argumente

expression

Ein Ausdruck, der einen oder mehrere CHAR- oder VARCHAR-Werte auswertet, beispielsweise ein Spaltenname oder ein Literal. Die Konvertierung von Null-Werten gibt Null-Werte zurück. Leere Zeichenfolgen werden in 0 umgewandelt.

format

Ein Zeichenfolgeliteral, das das Format des Eingabeausdrucks definiert. Weitere Informationen finden Sie unter Formatierungszeichen im Teradata-Stil für numerische Daten.

precision

Die Anzahl der Ziffern im numerischen Ergebnis. Der Standardwert ist 38.

scale

Die Anzahl der Ziffern rechts vom Dezimaltrennzeichen im numerischen Ergebnis. Der Standardwert ist 0.

Rückgabetyp

TEXT_TO_NUMERIC_ALT gibt eine DECIMAL-Zahl zurück.

Amazon Redshift gibt einen Fehler zurück, wenn die Umwandlung der von Ihnen angegebenen format-Phrase fehlschlägt.

Amazon Redshift wandelt die expression-Eingabezeichenfolge in den numerischen Typ mit der höchsten Genauigkeit um, die Sie für diesen Typ in der precision-Option angeben. Wenn die Länge des numerischen Wertes den Wert übersteigt, den Sie unter precision angegeben haben, rundet Amazon Redshift den numerischen Wert gemäß folgender Regeln:

- Wenn die Länge des Umwandlungsergebnisses die Länge übersteigt, die Sie in der format-Phrase angegeben haben, gibt Amazon Redshift einen Fehler zurück.
- Wenn das Ergebnis in einen numerischen Wert umgewandelt wird, wird das Ergebnis auf den nächsten Wert gerundet. Wenn die Nachkommastelle genau zwischen dem höheren und niedrigeren Umwandlungsergebnis liegt, wird das Ergebnis auf den nächsten geraden Wert gerundet.

Beispiele

Im folgenden Beispiel wird die expression-Eingabezeichenfolge '1.5' in den numerischen Wert '2' umgewandelt. Da die Anweisung keinen scale angibt, wird der scale standardmäßig auf 0 festgelegt und das Umwandlungsergebnis enthält keine Nachkommastellen. Da .5 zwischen 1 und 2 liegt, wird das Umwandlungsergebnis auf den geraden Wert 2 gerundet.

```
select text_to_numeric_alt('1.5');
```

```
 text_to_numeric_alt
---------------------
```

```
                   2
```

Im folgenden Beispiel wird die expression-Eingabezeichenfolge '2.51' in den numerischen Wert 3 umgewandelt. Da die Anweisung keinen scale-Wert angibt, wird der scale standardmäßig auf 0 festgelegt und das Umwandlungsergebnis enthält keine Nachkommastellen. Da .51 näher an 3 als 2 liegt, wird das Ergebnis auf den Wert 3 gerundet.

```
select text_to_numeric_alt('2.51');
```

```
 text_to_numeric_alt
---------------------
                   3
```

Im folgenden Beispiel wird die expression-Eingabezeichenfolge 123.52501 mit einer precision von 10 und einem scale von 2 in den numerischen Wert 123.53 umgewandelt.

```
select text_to_numeric_alt('123.52501', 10, 2);
```

```
 text_to_numeric_alt
---------------------
               123.53
```

Das folgende Beispiel gibt eine Tabellenspalte als Eingabe-expression an.

```
select text_to_numeric_alt(a), text_to_numeric_alt(b) from t_text2numeric order by 1;
```

```
           text_to_numeric_alt           |           text_to_numeric_alt
-----------------------------------------+-----------------------------------------
 -99999999999999999999999999999999999999 | -99999999999999999999999999999999999999
                                  -12300 |                                  -12300
                                     123 |                                     123
                                     123 |                                     123
  99999999999999999999999999999999999999 |  99999999999999999999999999999999999999
```

Im Folgenden sehen Sie die Tabellendefinition und die insert-Anweisung für dieses Beispiel.

```
create table  t_text2numeric (a varchar(200), b char(200));
```

```
insert into  t_text2numeric values
```

```
('123', '123'),
('+123.456', '+123.456'),
('-' || repeat('9', 38), '-' || repeat('9', 38)),
(repeat('9', 38) || '+', repeat('9', 38) || '+'),
('-123E2', '-123E2');
```

Datum-/Uhrzeit-Formatzeichenfolgen

Im Folgenden finden Sie eine Referenz für Datum-/Uhrzeit-Formatzeichenfolgen.

Die folgenden Formatzeichenfolgen gelten für Funktionen wie TO_CHAR. Diese Zeichenfolgen können Datum-/Uhrzeittrennzeichen (wie `-`, `/` oder `:`) sowie die folgenden „Datumsteile“ oder „Zeitteile“ enthalten.

Datumsteil oder Zeitteil	Bedeutung
BC oder B.C., AD oder A.D., b.c. oder bc, ad oder a.d.	Anzeigen für Zeitalter in Groß- und Kleinbuch staben
CC	Jahrhundertzahl mit zwei Ziffern
YYYY, YYY, YY, Y	Jahreszahl mit 4, 3, 2 oder 1 Ziffer
Y,YYY	Jahreszahl mit 4 Ziffern und Komma
IYYY, IYY, IY, I	Jahreszahl der internationalen Organisation für Normung (International Organization for Standardization, ISO) mit 4, 3, 2 oder 1 Ziffer
Q	Quartalszahl (1 bis 4)
MONAT, Monat, monat	Name des Monats (Großbuchstaben, Groß- und Kleinbuchstaben, mit leeren Stellen auf 9 Zeichen aufgefüllt)
MON, Mon, mon	Abgekürzter Name des Monates (Großbuch staben, Groß- und Kleinbuchstaben, mit leeren Stellen auf 3 Zeichen aufgefüllt)
MM	Monatszahl (01 bis 12)

Datumsteil oder Zeitteil	Bedeutung
RM, rm	Monat in römischen Ziffern (I–XII, wobei I Januar ist, Groß- oder Kleinbuchstaben)
W	Woche des Monats (1–5; die erste Woche beginnt am ersten Tag des Monats)
WW	Woche des Jahres (1–53; die erste Woche beginnt am ersten Tag des Jahres)
IW	Woche des Jahres nach ISO (der erste Donnerstag des neuen Jahres liegt in Woche 1)
DAY, Day, day	Name des Tages (Großbuchstaben, Groß- und Kleinbuchstaben, mit leeren Stellen auf 9 Zeichen aufgefüllt)
DY, Dy, dy	Abgekürzter Name des Tages (Großbuch staben, Groß- und Kleinbuchstaben, mit leeren Stellen auf 3 Zeichen aufgefüllt)
DDD	Tag des Jahres (001–366)
IDDD	Tag der Woche des Jahres nach ISO 8601 (001–371; Tag 1 des Jahres ist Montag der ersten ISO-Woche)
DD	Tag des Monats als Zahl (01–31)

Datumsteil oder Zeitteil	Bedeutung
D	Wochentag (1–7; wobei Sonntag 1 ist) **Note** Der Datumsteil D verhält sich anders als der Datumsteil Wochentag (DOW), der für die Datum-/Uhrzeitfunktionen DATE_PART and EXTRACT verwendet wird. DOW basiert auf den Ganzzahlen 0–6, wobei die 0 für den Sonntag steht. Weitere Informationen finden Sie unter Datumsteile für Datums- oder Zeitstempelfunktionen.
ID	Tag der Woche nach ISO 8601, Montag (1) bis Sonntag (7)
J	Julianischer Tag (Tage seit dem 1. Januar 4712 BC)
HH24	Stunde (24-Stunden-Uhr, 00–23)
HH oder HH12	Stunde (12-Stunden-Uhr, 01–12)
MI	Minuten (00–59)
SS	Sekunden (00–59)
MS	Millisekunden (,000)
US	Mikrosekunden (,000000)
AM oder PM, A.M. oder P.M., a.m. oder p.m., am oder pm	Meridiananzeigen für Zeitalter in Groß- und Kleinbuchstaben (für die 12-Stunden-Uhr)
TZ, tz	Zeitzonenabkürzung in Groß- und Kleinbuchstaben, nur für TIMESTAMPTZ gültig

Datumsteil oder Zeitteil	Bedeutung
OF	Unterschied zu UTC; nur für TIMESTAMPTZ gültig

> **Note**
>
> Sie müssen datetime-Separatoren (wie etwa '-', '/' oder ':') mit einzelnen Anführungszeichen umgeben, die in der vorherigen Tabelle aufgeführten "dateparts" und "timeparts" müssen allerdings in doppelten Anführungszeichen stehen.

Beispiele

Beispiele für das Formatieren von Datumsangaben als Zeichenfolgen finden Sie unter TO_CHAR.

Numerische Formatzeichenfolgen

Im Folgenden finden Sie eine Referenz für numerische Formatzeichenfolgen.

Die folgenden Formatzeichenfolgen gelten für Funktionen wie TO_NUMBER und TO_CHAR.

- Beispiele für das Formatieren von Zeichenfolgen als Zahlen finden Sie unter TO_NUMBER.
- Beispiele für das Formatieren von Zahlen als Zeichenfolgen finden Sie unterTO_CHAR.

Format	Beschreibung
9	Numerischer Wert mit der angegebenen Anzahl von Stellen.
0	Numerischer Wert mit Nullen zu Beginn.
. (Punkt), D	Dezimalpunkt.
, (Komma)	Tausendertrennzeichen.

Format	Beschreibung
CC	Jahrhundertcode. Das 21. Jahrhundert begann beispielsweise am 01.01.2001 (wird nur für TO_CHAR unterstützt).
FM	Füllmodus. Unterdrückt ausfüllende Leerzeichen und Nullen.
PR	Negativer Wert in Winkelklammern.
S	Vorzeichen, das mit einer Zahl fest verbunden ist.
L	Währungssymbol an der angegebenen Position.
G	Gruppentrennzeichen.
MI	Minuszeichen an der angegebenen Position für Zahlen kleiner als 0.
PL	Pluszeichen an der angegebenen Position für Zahlen größer als 0.
SG	Plus- oder Minuszeichen an der angegebenen Position.
RN	Römische Zahl zwischen 1 und 3999 (wird nur für TO_CHAR unterstützt).
TH oder th	Ordnungszahlsuffix. Konvertiert keine Bruchzahlen oder Werte kleiner als null.

Formatierungszeichen im Teradata-Stil für numerische Daten

Im Folgenden erfahren Sie, wie Sie die Funktionen TEXT_TO_INT_ALT und TEXT_TO_NUMERIC_ALT die Zeichen in der expression-Eingabezeichenfolge interpretieren. Außerdem erhalten Sie eine Auflistung der Zeichen, die Sie in der format-Phrase angeben können.

Darüber hinaus finden Sie eine Beschreibung der Unterschiede zwischen der Formatierung im Teradata-Stil und Amazon Redshift für die format-Option.

Format	Beschreibung
G	Wir nicht als Gruppentrennzeichen für die expression-Eingabezeichenfolge unterstützt. Sie können dieses Zeichen nicht in der format-Phrase angeben.
D	Dezimaltrennzeichen. Sie können dieses Zeichen in der format-Phrase angeben. Dieses Zeichen entspricht dem . (Punkt). Das Dezimaltrennzeichen kann nicht in einer format-Phrase vorkommen, die eines der folgenden Zeichen enthält: • . (Punkt) • S (großgeschriebenes S) • V (großgeschriebenes V)
/ , : %	Einfügungszeichen / (Schrägstrich); , (Komma); : (Doppelpunkt) und % (Prozentz eichen). Sie können diese Zeichen nicht in der format-Phrase angeben. Amazon Redshift ignoriert diese Zeichen in der expression-Eingabezeichenfolge.
.	Punkt als Dezimaltrennzeichen. Dieses Zeichen kann nicht in einer format-Phrase vorkommen, die eines der folgenden Zeichen enthält: • D (großgeschriebenes D)

Format	Beschreibung
	• S (großgeschriebenes S) • V (großgeschriebenes V)
B	Sie können kein Leerraumzeichen (B) in der format-Phrase angeben. In der expression-Zeichenfolge werden Leerzeichen am Anfang und Ende ignoriert und Leerzeichen zwischen Ziffern sind nicht zulässig.
+ -	Sie können kein Plus- oder Minuszeichen (+ oder -) in der format-Phrase angeben. Plus- und Minuszeichen werden jedoch implizit als Teil des numerischen Wertes geparst, wenn sie in der expression-Eingabezeichenfolge auftauchen.
V	Indikator für die Position des Dezimaltr ennzeichens. Dieses Zeichen kann nicht in einer format-Phrase vorkommen, die eines der folgenden Zeichen enthält: • D (großgeschriebenes D) • . (Punkt)
Z	Nullunterdrückte Dezimalstelle. Amazon Redshift kürzt führende Nullen. Das Z-Zeichen darf keiner 9 folgen. Das Z-Zeichen muss links vom Dezimaltrennzeichen stehen, wenn die Nachkommastelle eine 9 enthält.
9	Dezimalstelle.

Format	Beschreibung
CHAR(n)	Für dieses Format können Sie für Folgendes angeben: • CHAR besteht aus Z- oder 9-Zeichen. Amazon Redshift unterstützt kein + (plus) oder - (minus) im CHAR-Wert. • n ist eine Ganzzahlkonstante, I oder F. Bei I ist dies die Anzahl der Zeichen, die erforderlich sind, um den Ganzzahlteil numerischer oder ganzzahliger Daten anzuzeigen. Bei F ist dies die Anzahl der Zeichen, die erforderlich sind, um die Nachkommastellen numerischer Daten anzuzeigen.
-	Bindestrich-Zeichen (-). Sie dieses Zeichen nicht in der format-Phrase angeben. Amazon Redshift ignoriert dieses Zeichen in der expression-Eingabezeichenfolge.

Format	Beschreibung
S	Signed Zone Decimal. Das S-Zeichen muss auf die letzte Dezimalstelle in der format-Phrase folgen. Das letzte Zeichen der expression-Eingabezeichenfolge und die entsprechende numerische Konvertierung finden Sie unter Datenformatierungszeichen für Signed Zone Decimal, numerische Datenformatierung im Teradata-Stil . Das S-Zeichen kann nicht in einer format-Phrase vorkommen, die eines der folgenden Zeichen enthält: • + (Pluszeichen) • . (Punkt) • D (großgeschriebenes D) • Z (großgeschriebenes Z) • F (großgeschriebenes F) • E (großgeschriebenes E)
E	Exponentialnotation. Die expression-Eingabez eichenfolge kann ein Exponentenzeichen enthalten. E kann nicht als Exponentenzeichen in einer format-Phrase angegeben werden.
FN9	Wird in Amazon Redshift nicht unterstützt.
FNE	Wird in Amazon Redshift nicht unterstützt.

Format	Beschreibung
$, USD, US-Dollar	Dollarzeichen ($), ISO-Währungssymbol (USD) und der Währungsname US-Dollar. Beim ISO-Währungssymbol USD und dem Währungsnamen US-Dollar wird zwischen Groß- und Kleinschreibung unterschieden. Amazon Redshift unterstützt nur die Währung USD. Die expression-Eingabezeichenfolge kann Leerzeichen zwischen dem Dollar-Wä hrungssymbol und dem numerischen Wert umfassen, zum Beispiel „$ 123E2“ oder „123E2 $“.
L	Währungssymbol. Dieses Währungss ymbol kann in der format-Phrase nur einmal vorhanden sein. Es ist nicht möglich, mehrere Währungssymbole anzugeben.
C	ISO-Währungssymbol. Dieses Währungss ymbol kann in der format-Phrase nur einmal vorhanden sein. Es ist nicht möglich, mehrere Währungssymbole anzugeben.
N	Vollständiger Währungsname. Dieses Währungssymbol kann in der format-Phrase nur einmal vorhanden sein. Es ist nicht möglich, mehrere Währungssymbole anzugeben.
O	Doppeltes Währungssymbol. Sie können dieses Zeichen nicht in der format-Phrase angeben.
U	Doppeltes ISO-Währungssymbol. Sie können dieses Zeichen nicht in der format-Phrase angeben.

Format	Beschreibung
A	Vollständiger doppelter Währungsname. Sie können dieses Zeichen nicht in der format-Phrase angeben.

Datenformatierungszeichen für Signed Zone Decimal, numerische Datenformatierung im Teradata-Stil

Sie können die folgenden Zeichen in der format-Phrase der TEXT_TO_INT_ALT- und TEXT_TO_NUMERIC_ALT-Funktionen für einen Signed-Zoned-Decimal-Wert verwenden.

Letztes Zeichen der Eingabezeichenfolge	Numerische Konvertierung
{ oder 0	n … 0
A oder 1	n … 1
B oder 2	n … 2
C oder 3	n … 3
D oder 4	n … 4
E oder 5	n … 5
F oder 6	n … 6
G oder 7	n … 7
H oder 8	n … 8
I oder 9	n … 9
}	-n … 0
J	-n … 1
K	-n … 2

Letztes Zeichen der Eingabezeichenfolge	Numerische Konvertierung
L	-n … 3
M	-n … 4
N	-n … 5
O	-n … 6
P	-n … 7
Q	-n … 8
R	-n … 9

Datums- und Zeitfunktionen

In diesem Abschnitt finden Sie Informationen zu den skalaren Datums- und Zeitfunktionen, die Amazon Redshift unterstützt.

Themen

- Funktion DATE_PART
- Funktion DATE_PART_YEAR
- Funktion DATE_TRUNC
- Funktion EXTRACT
- Funktion GETDATE
- Funktion INTERVAL_CMP
- Funktion LAST_DAY
- Funktion MONTHS_BETWEEN
- Funktion NEXT_DAY
- Funktion SYSDATE
- Funktion TIMEOFDAY
- Funktion TIMESTAMP_CMP
- Funktion TIMESTAMP_CMP_DATE
- Funktion TIMESTAMP_CMP_TIMESTAMPTZ
- Funktion TIMESTAMPTZ_CMP
- Funktion TIMESTAMPTZ_CMP_DATE
- Funktion TIMESTAMPTZ_CMP_TIMESTAMP
- Funktion TIMEZONE
- Funktion TO_TIMESTAMP
- Die Funktion TRUNC
- Datumsteile für Datums- oder Zeitstempelfunktionen

Zusammenfassung der Datums- und Zeitfunktionen

Funktion	Syntax	Rückgabewert
Operator + (Verkettung) Verkettet ein Datum mit einer Uhrzeit auf beiden Seiten des Pluszeichens (+) und gibt	date + time	`TIMESTAMP` oder `TIMESTAMPZ`

Funktion	Syntax	Rückgabewert
einen TIMESTAMP oder TIMESTAMPTZ zurück.		
ADD_MONTHS Fügt die angegebene Anzahl von Monaten zu einem Datums- oder Zeitstempel hinzu.	ADD_MONTHS ({date|timestamp}, integer)	TIMESTAMP
AT TIME ZONE Gibt an, welche Zeitzone mit dem Ausdruck TIMESTAMP oder TIMESTAMPTZ zu verwenden ist.	AT TIME ZONE 'timezone'	TIMESTAMP oder TIMESTAMPTZ
CONVERT_TIMEZONE Konvertiert einen Zeitstempel von einer Zeitzone zu einer anderen.	CONVERT_TIMEZONE (['timezone',] 'timezone', timestamp)	TIMESTAMP
CURRENT_DATE Gibt ein Datum in der Zeitzone der aktuellen Sitzung (standardmäßig UTC) für den Beginn der aktuellen Transaktion aus.	CURRENT_DATE	DATE
DATE_CMP Vergleicht zwei Datumsangaben und gibt 0 aus, wenn beide identisch sind, sowie 1, wenn date1 größer ist, und -1, wenn date2 größer ist.	DATE_CMP (date1, date2)	INTEGER
DATE_CMP_TIMESTAMP Vergleicht ein Datum mit einer Uhrzeit und gibt 0 aus, wenn die Werte identisch sind, sowie, 1 wenn date größer ist, und -1, wenn timestamp größer ist.	DATE_CMP_TIMESTAMP (date, timestamp)	INTEGER

Funktion	Syntax	Rückgabewert
DATE_CMP_TIMESTAMPTZ Vergleicht ein Datum und einen Zeitstempel mit der Zeitzone und gibt `0` aus, wenn die Werte identisch sind, sowie 1, wenn date größer ist, und `-1`, wenn timestamptz größer ist.	DATE_CMP_TIMESTAMPTZ (date, timestamptz)	`INTEGER`
DATE_PART_YEAR Extrahiert das Jahr aus einem Datum.	DATE_PART_YEAR (date)	`INTEGER`
DATEADD Erhöht ein Datum oder eine Uhrzeit um ein bestimmtes Intervall.	DATEADD (datepart, interval, {date|time|timetz|timestamp})	`TIMESTAMP` oder `TIME` oder `TIMETZ`
DATEDIFF Gibt die Differenz zwischen zwei Datumsangaben oder Uhrzeiten für einen bestimmten Datumsteil, etwa einen Tag oder einen Monat, aus.	DATEDIFF (datepart, {date|time|timetz|timestamp}, {date|time|timetz|timestamp})	`BIGINT`
DATE_PART Extrahiert einen Datumsteilwert aus einem Datum oder einer Uhrzeit.	DATE_PART (datepart, {date|timestamp})	`DOUBLE`
DATE_TRUNC Verkürzt einen Zeitstempel auf der Grundlage eines Datumsteils.	DATE_TRUNC ('datepart', timestamp)	`TIMESTAMP`

Funktion	Syntax	Rückgabewert
EXTRACT Extrahiert einen Datums- oder Uhrzeitteil von einem Timestamp-, Timestamptz-, Time- oder Timetz-Wert.	EXTRACT (datepart AUS source)	`INTEGER or DOUBLE`
GETDATE Gibt das aktuelle Datum und die aktuelle Uhrzeit in der Zeitzone der aktuellen Sitzung (standardmäßig UTC) aus. Die Klammern sind erforderlich.	GETDATE()	`TIMESTAMP`
INTERVAL_CMP Vergleicht zwei Intervalle und gibt `0` aus, wenn beide identisch sind, sowie `1`, wenn interval1 größer ist, und `-1`, wenn interval2 größer ist.	INTERVAL_CMP (interval1, interval2)	`INTEGER`
LAST_DAY Gibt das Datum des letzten Tages des Monats aus, der date enthält.	LAST_DAY(date)	`DATE`
MONTHS_BETWEEN Gibt die Anzahl der Monate zwischen zwei Daten aus.	MONTHS_BETWEEN (date, date)	`FLOAT8`
NEXT_DAY Gibt das Datum der ersten Instanz von day aus, die zeitlich nach date liegt.	NEXT_DAY (date, day)	`DATE`

Funktion	Syntax	Rückgabewert
SYSDATE Gibt das Datum und die Uhrzeit nach UTC für den Beginn der aktuellen Transaktion aus.	SYSDATE	TIMESTAMP
TIMEOFDAY Gibt den aktuellen Wochentag, das aktuelle Datum und die aktuelle Uhrzeit in der Zeitzone der aktuellen Sitzung (standardmäßig UTC) als Zeichenfolgenwert aus.	TIMEOFDAY()	VARCHAR
TIMESTAMP_CMP Vergleicht zwei Zeitstempel und gibt 0 aus, wenn beide identisch sind, sowie 1, wenn timestamp1 größer ist, und -1, wenn timestamp 2 größer ist.	TIMESTAMP_CMP (timestamp1, timestamp2)	INTEGER
TIMESTAMP_CMP_DATE Vergleicht einen Zeitstempel mit einem Datum und gibt 0 aus, wenn beide identisch sind, sowie,1 wenn timestamp größer ist, und, -1 wenn date größer ist.	TIMESTAMP_CMP_DATE (timestamp, date)	INTEGER
TIMESTAMP_CMP_TIMESTAMPTZ Vergleicht einen Zeitstempel mit einem Zeitstempel mit Zeitzone und gibt 0 aus, wenn beide identisch sind, sowie, 1 wenn timestamp größer ist, und, -1 wenn timestamptz größer ist.	TIMESTAMP_CMP_TIMESTAMPTZ (timestamp, timestamptz)	INTEGER

Funktion	Syntax	Rückgabewert
TIMESTAMPTZ_CMP Vergleicht zwei Zeitstempelwerte mit Zeitzone und gibt 0 aus, wenn beide identisch sind, sowie, 1 wenn timestamptz1 größer ist, und, -1 wenn timestamptz2 größer ist.	TIMESTAMPTZ_CMP (timestamptz1, timestamptz2)	INTEGER
TIMESTAMPTZ_CMP_DATE Vergleicht den Wert eines Zeitstempels mit Zeitzone mit einem Datum und gibt 0 aus, wenn beide identisch sind, sowie 1, wenn timestamptz größer ist, und -1, wenn date größer ist.	TIMESTAMPTZ_CMP_DATE (timestamptz, date)	INTEGER
TIMESTAMPTZ_CMP_TIMESTAMP Vergleicht einen Zeitstempel mit Zeitzone mit einem Zeitstempel und gibt 0 aus, wenn beide identisch sind, sowie 1, wenn timestamptz größer ist, und -1, wenn timestamp größer ist.	TIMESTAMPTZ_CMP_TIMESTAMP (timestamptz, timestamp)	INTEGER
TIMEZONE Gibt einen Zeitstempel für den angegebenen Wert eines Zeitstempels mit Zeitzone aus.	TIMEZONE ('timezone' { timestamp | timestamptz)	TIMESTAMP oder TIMESTAMPTZ
TO_TIMESTAMP Gibt einen Zeitstempel mit Zeitzone für das angegebene Zeitstempelformat mit Zeitzone aus.	TO_TIMESTAMP ('timestamp', 'format')	TIMESTAMPTZ
TRUNC Verkürzt einen Zeitstempel und gibt ein Datum aus.	TRUNC(timestamp)	DATE

> **Note**
>
> Sprungsekunden werden bei Berechnungen der verstrichenen Zeit nicht berücksichtigt.

Datums- und Zeitfunktionen in Transaktionen

Wenn Sie die folgenden Funktionen mit einem Transaktionsblock (BEGINN … END) ausführen, gibt die Funktion das Startdatum bzw. die Startzeit der aktuellen Transaktion aus, nicht den Beginn der aktuellen Anweisung.

- SYSDATE
- TIMESTAMP
- CURRENT_DATE

Die folgenden Funktionen geben immer das Startdatum oder die Startzeit der aktuellen Anweisung aus, selbst wenn sie sich innerhalb eines Transaktionsblocks befinden.

- GETDATE
- TIMEOFDAY

Veraltete Funktionen, die ausschließlich für Führungsknoten gelten

Die folgenden Datumsfunktionen sind veraltet, da sie nur auf dem Führungsknoten ausgeführt werden. Weitere Informationen finden Sie unter Exklusive Führungsknotenfunktionen.

- AGE. Verwenden Sie stattdessen Funktion DATEDIFF.
- CURRENT_TIME. Verwenden Sie stattdessen Funktion GETDATE oder SYSDATE.
- CURRENT_TIMESTAMP. Verwenden Sie stattdessen Funktion GETDATE oder SYSDATE.
- LOCALTIME. Verwenden Sie stattdessen Funktion GETDATE oder SYSDATE.
- LOCALTIMESTAMP. Verwenden Sie stattdessen Funktion GETDATE oder SYSDATE.
- ISFINITE
- NOW. Verwenden Sie stattdessen Funktion GETDATE oder SYSDATE.

Operator + (Verkettung)

Verkettet ein DATE mit einer TIME oder TIMETZ auf beiden Seiten des Pluszeichens (+) und gibt einen TIMESTAMP oder TIMESTAMPTZ zurück.

Syntax

```
date + {time | timetz}
```

Die Reihenfolge der Argumente kann umgekehrt werden. Beispiel: time + date.

Argumente

date

Eine Spalte vom Datentyp `DATE` oder ein Ausdruck, der implizit zu einem `DATE`-Typ ausgewertet wird.

variieren

Eine Spalte vom Datentyp `TIME` oder ein Ausdruck, der implizit zu einem `TIME`-Typ ausgewertet wird.

timetz

Eine Spalte vom Datentyp `TIMETZ` oder ein Ausdruck, der implizit zu einem `TIMETZ`-Typ ausgewertet wird.

Rückgabetyp

TIMESTAMP, wenn die Eingabe date + time ist.

TIMESTAMPTZ, wenn die Eingabe date + timetz ist.

Beispiele

Beispieleinrichtung

Verwenden Sie den folgenden Befehl, um die in den Beispielen verwendeten Tabellen TIME_TEST und TIMETZ_TEST einzurichten.

```
create table time_test(time_val time);
```

```
insert into time_test values
('20:00:00'),
('00:00:00.5550'),
('00:58:00');

create table timetz_test(timetz_val timetz);

insert into timetz_test values
('04:00:00+00'),
('00:00:00.5550+00'),
('05:58:00+00');
```

Beispiele mit einer Zeitspalte

Die folgende Beispieltabelle TIME_TEST enthält eine Spalte TIME_VAL (Typ TIME) mit drei eingefügten Werten.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

Im folgenden Beispiel werden ein Datumsliteral und eine TIME_VAL-Spalte verkettet.

```
select date '2000-01-02' + time_val as ts from time_test;

ts
----------------------
2000-01-02 20:00:00
2000-01-02 00:00:00.5550
2000-01-02 00:58:00
```

Im folgenden Beispiel werden ein Datumsliteral und eine Zeitspalte verkettet.

```
select date '2000-01-01' + time '20:00:00' as ts;

         ts
---------------------
```

```
 2000-01-01 20:00:00
```

Im folgenden Beispiel werden ein Zeit- und ein Datumsliteral verkettet.

```
select time '20:00:00' + date '2000-01-01' as ts;

         ts
---------------------
 2000-01-01 20:00:00
```

Beispiele mit einer TIMETZ-Spalte

Die folgende Beispieltabelle TIMETZ_TEST enthält eine Spalte TIMETZ_VAL (Typ TIMETZ) mit drei eingefügten Werten.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

Im folgenden Beispiel werden ein Datumsliteral und eine TIMETZ_VAL-Spalte verkettet.

```
select date '2000-01-01' + timetz_val as ts from timetz_test;
ts
---------------------
2000-01-01 04:00:00+00
2000-01-01 00:00:00.5550+00
2000-01-01 05:58:00+00
```

Im folgenden Beispiel werden eine TIMETZ_VAL-Spalte und ein Datumsliteral verkettet.

```
select timetz_val + date '2000-01-01' as ts from timetz_test;
ts
---------------------
2000-01-01 04:00:00+00
2000-01-01 00:00:00.5550+00
2000-01-01 05:58:00+00
```

Im folgenden Beispiel werden ein DATE-Literal und TIMETZ-Literal verkettet. Das Beispiel gibt einen TIMESTAMPTZ zurück, der standardmäßig in der Zeitzone UTC liegt. UTC liegt 8 Stunden vor PST, das Ergebnis liegt also 8 Stunden vor der Eingabezeit.

```
select date '2000-01-01' + timetz '20:00:00 PST' as ts;

          ts
------------------------
 2000-01-02 04:00:00+00
```

Funktion ADD_MONTHS

ADD_MONTHS fügt die angegebene Zahl von Monaten zu einem Datums- oder Zeitstempelwert bzw. -ausdruck hinzu. Die Funktion DATEADD bietet eine ähnliche Funktionalität.

Syntax

```
ADD_MONTHS( {date | timestamp}, integer)
```

Argumente

date | timestamp

Eine Spalte vom Datentyp `DATE` oder `TIMESTAMP` bzw. ein Ausdruck, der implizit zu einem `DATE`- oder `TIMESTAMP`-Typ ausgewertet wird. Wenn das Datum der letzte Tag des Monats ist, oder wenn der resultierende Monat kürzer ist, gibt die Funktion im Ergebnis den letzten Tag des Monats aus. Für andere Datumsangaben enthält das Ergebnis die gleiche Tagesnummer wie der Datumsausdruck.

integer

Ein Wert vom Datentyp `INTEGER`. Verwenden Sie eine negative Zahl, um Monate von Datumsangaben abzuziehen.

Rückgabetyp

TIMESTAMP

Beispiele

Die folgende Abfrage verwendet die Funktion ADD_MONTHS innerhalb einer TRUNC-Funktion. Die TRUNC-Funktion entfernt die Tageszeit aus dem Ergebnis von ADD_MONTHS. Die Funktion

ADD_MONTHS fügt jedem Wert aus der Spalte CALDATE 12 Monate hinzu. Die Werte in der Spalte CALDATE sind Datumsangaben.

```
select distinct trunc(add_months(caldate, 12)) as calplus12,
trunc(caldate) as cal
from date
order by 1 asc;

 calplus12  |    cal
------------+------------
 2009-01-01 | 2008-01-01
 2009-01-02 | 2008-01-02
 2009-01-03 | 2008-01-03
...
(365 rows)
```

Im folgenden Beispiel wird die Funktion ADD_MONTHS verwendet, um einem Zeitstempel einen Monat hinzuzufügen.

```
select add_months('2008-01-01 05:07:30', 1);

add_months
---------------------
2008-02-01 05:07:30
```

Die folgenden Beispiele illustrieren die Verhaltensweise, wenn die Funktion ADD_MONTHS für Datumsangaben verwendet wird, die Monate mit unterschiedlichen Anzahlen von Tagen enthalten. Dieses Beispiel zeigt, wie die Funktion das Hinzufügen eines Monats zum 31. März und das Hinzufügen eines Monats zum 30. April handhabt. Der April hat 30 Tage. Wenn Sie also zum 31. März einen Monat hinzufügen, ergibt sich der 30. April. Der Mai hat 31 Tage. Wenn Sie also zum 30. April einen Monat hinzufügen, ergibt sich der 31. Mai.

```
select add_months('2008-03-31',1);

add_months
---------------------
2008-04-30 00:00:00

select add_months('2008-04-30',1);

add_months
```

```
---------------------
2008-05-31 00:00:00
```

Funktion AT TIME ZONE

AT TIME ZONE gibt an, welche Zeitzone mit dem Ausdruck TIMESTAMP oder TIMESTAMPTZ zu verwenden ist.

Syntax

```
AT TIME ZONE 'timezone'
```

Argumente

Zeitzone

Die TIMEZONE für den Rückgabewert. Die Zeitzone kann als Zeitzonenname (beispielsweise **'Africa/Kampala'** oder **'Singapore'**) oder als Zeitzonenabkürzung (beispielsweise **'UTC'** oder **'PDT'**) angegeben werden.

Führen Sie den folgenden Befehl aus, um eine Liste der unterstützten Zeitzonennamen anzuzeigen.

```
select pg_timezone_names();
```

Führen Sie den folgenden Befehl aus, um eine Liste der unterstützten Zeitzonenabkürzungen anzuzeigen.

```
select pg_timezone_abbrevs();
```

Weitere Informationen und Beispiele finden Sie unter Nutzungshinweise zu Zeitstempeln.

Rückgabetyp

TIMESTAMPTZ bei Verwendung mit einem TIMESTAMP-Ausdruck. TIMESTAMP bei Verwendung mit einem TIMESTAMPTZ-Ausdruck.

Beispiele

Im folgenden Beispiel wird ein Zeitstempelwert ohne Zeitzone konvertiert und als MST-Zeit (UTC +7 in POSIX) interpretiert. Das Beispiel gibt einen Wert des Datentyps TIMESTAMPTZ für die UTC-

Zeitzone zurück. Wenn Sie Ihre Standardzeitzone auf eine andere Zeitzone als UTC konfigurieren, wird möglicherweise ein anderes Ergebnis angezeigt.

```
SELECT TIMESTAMP '2001-02-16 20:38:40' AT TIME ZONE 'MST';

timezone
------------------------
2001-02-17 03:38:40+00
```

Im folgenden Beispiel wird ein Eingabezeitstempel mit einem Zeitzonenwert verwendet, bei dem die angegebene Zeitzone EST (UTC+5 in POSIX) lautet, und in MST (UTC+7 in POSIX) konvertiert. Das Beispiel gibt einen Wert des Datentyps TIMESTAMP zurück.

```
SELECT TIMESTAMPTZ '2001-02-16 20:38:40-05' AT TIME ZONE 'MST';

timezone
------------------------
2001-02-16 18:38:40
```

Funktion CONVERT_TIMEZONE

CONVERT_TIMEZONE konvertiert einen Zeitstempel von einer Zeitzone zu einer anderen. Die Funktion passt sich automatisch an die Sommerzeit an.

Syntax

```
CONVERT_TIMEZONE( ['source_timezone',] 'target_timezone', 'timestamp')
```

Argumente

source_timezone

(Optional) Die Zeitzone des aktuellen Zeitstempels. Der Standardwert ist UTC. Weitere Informationen finden Sie unter Nutzungshinweise zu Zeitstempeln.

target_timezone

Die Zeitzone für den neuen Zeitstempel. Weitere Informationen finden Sie unter Nutzungshinweise zu Zeitstempeln.

timestamp

Eine Zeitstempelspalte bzw. ein entsprechender Ausdruck, die/der implizit zu einem Zeitstempel konvertiert wird.

Rückgabetyp

TIMESTAMP

Nutzungshinweise zu Zeitstempeln

source_timezone oder target_timezone können als Zeitzonenname (wie 'Africa/Kampala' oder 'Singapur') oder als Abkürzung für die Zeitzone (wie 'UTC' oder 'PDT') angegeben werden. Sie müssen Zeitzonennamen nicht in Namen oder Abkürzungen in Abkürzungen konvertieren. Sie können beispielsweise einen Zeitstempel aus dem Quellzeitzonennamen „Singapur“ auswählen und ihn in einen Zeitstempel mit der Zeitzonenabkürzung „PDT“ konvertieren.

> **Note**
>
> Die Verwendung eines Zeitzonennamens oder einer Zeitzonenabkürzung kann aufgrund der lokalen Jahreszeit, z. B. der Sommerzeit, zu unterschiedlichen Ergebnissen führen.

Verwendung eines Zeitzonennamens

Führen Sie den folgenden Befehl aus, um eine aktuelle und vollständige Liste der Zeitzonennamen anzuzeigen.

```
select pg_timezone_names();
```

Jede Zeile enthält eine durch Kommas getrennte Zeichenfolge mit dem Namen der Zeitzone, der Abkürzung, dem UTC-Offset und der Angabe, ob in der Zeitzone die Sommerzeit gilt (`t` oder `f`). Das folgende Snippet zeigt beispielsweise zwei Ergebniszeilen. Die erste Zeile enthält die Zeitzone`Europe/Paris`, die Abkürzung`CET`, mit einem `01:00:00` Offset von UTC. Damit `f` wird angegeben, dass die Sommerzeit nicht eingehalten wird. Die zweite Zeile enthält die Zeitzone`Israel`, Abkürzung`IST`, mit einem `02:00:00` Offset von UTC, `f` um anzuzeigen, dass die Sommerzeit nicht eingehalten wird.

```
pg_timezone_names
```

```
-----------------
(Europe/Paris,CET,01:00:00,f)
(Israel,IST,02:00:00,f)
```

Führen Sie die SQL-Anweisung aus, um die gesamte Liste abzurufen und einen Zeitzonennamen zu finden. Es werden ungefähr 600 Zeilen zurückgegeben. Auch wenn es sich bei einigen der zurückgegebenen Zeitzonennamen um Akronyme handelt (etwa GB, PRC, ROK), werden sie von der Funktion CONVERT_TIMEZONE als Zeitzonennamen und nicht als Zeitzonenabkürzungen behandelt.

Wenn Sie eine Zeitzone mithilfe eines Zeitzonennamens angeben, passt sich CONVERT_TIMEZONE automatisch an die Sommerzeit (DST) oder ein anderes lokales saisonales Protokoll an, wie Sommerzeit, Standardzeit oder Winterzeit, das für diese Zeitzone zu dem mit 'timestamp' angegebenen Datum und Uhrzeit gilt. So steht beispielsweise „Europe/London“ für UTC im Winter, im Sommer kommt eine Stunde dazu.

Verwendung einer Zeitzonenabkürzung

Führen Sie den folgenden Befehl aus, um eine aktuelle und vollständige Liste der Zeitzonenabkürzungen anzuzeigen.

```
select pg_timezone_abbrevs();
```

Die Ergebnisse enthalten eine durch Kommas getrennte Zeichenfolge mit der Abkürzung der Zeitzone, dem UTC-Offset und der Angabe, ob in der Zeitzone die Sommerzeit gilt (`t` oder `f`). Das folgende Snippet zeigt beispielsweise zwei Ergebniszeilen. Die erste Zeile enthält die Abkürzung für Pacific Daylight Time `PDT` mit einem UTC-Offset von `-07:00:00` sowie die Angabe `t`, um anzuzeigen, dass die Sommerzeit gilt. Die zweite Zeile enthält die Abkürzung für Pacific Standard Time `PST` mit einem `-08:00:00` Offset von UTC, `f` um anzuzeigen, dass dort keine Sommerzeit eingehalten wird.

```
pg_timezone_abbrevs
--------------------
(PDT,-07:00:00,t)
(PST,-08:00:00,f)
```

Führen Sie die SQL-Anweisung aus, um die gesamte Liste abzurufen und anhand des Offsets und der Angabe zur Sommerzeit nach einer Abkürzung zu suchen. Es werden ungefähr 200 Zeilen zurückgegeben.

Zeitzonenabkürzungen stehen für eine feste Verschiebung von der UTC. Wenn Sie eine Zeitzone mit einer Abkürzung für eine Zeitzone angeben, verwendet CONVERT_TIMEZONE den festen Offset von UTC und passt sich nicht an lokale saisonale Protokolle an.

Verwendung des POSIX-Stil-Formats

Eine Zeitzonenangabe im POSIX-Stil hat die Form STDoffset oder STDoffsetDST, wobei STD eine Zeitzonenabkürzung ist, offset die numerische Verschiebung in Stunden westlich von UTC und DST eine optionale Abkürzung für die Sommerzeit. Für die Sommerzeit wird angenommen, dass Sie eine Stunde vor der angegebenen Verschiebung liegt.

Zeitzonenformate im POSIX-Stil verwenden positive Verschiebungen westlich von Greenwich; im Gegensatz dazu verwendet die ISO-8601-Konvention östlich von Greenwich positive Werte.

Es folgen einige Beispiele für Zeitzonen im POSIX-Stil:

- PST8
- PST8PDT
- EST5
- EST5EDT

> **Note**
>
> Amazon Redshift validiert Zeitzonenangaben im POSIX-Stil-Format nicht, es kann daher sein, dass die Zeitzone auf einen ungültigen Wert gesetzt wird. Beispielsweise führt der folgende Befehl nicht zu einem Fehler, obwohl dadurch die Zeitzone auf einen ungültigen Wert gesetzt wird.
>
> ```
> set timezone to ‘xxx36’;
> ```

Beispiele

Viele der Beispiele verwenden den TICKIT-Beispieldatensatz. Weitere Informationen finden Sie unter Beispieldatenbank.

Das folgende Beispiel konvertiert den Zeitstempelwert von der Standardzeitzone UTC zu PST.

```
select convert_timezone('PST', '2008-08-21 07:23:54');
```

```
 convert_timezone
------------------------
2008-08-20 23:23:54
```

Das folgende Beispiel konvertiert den Zeitstempelwert in der Spalte LISTTIME von der Standardzeitzone UTC zu PST. Obwohl der Zeitstempel in der Sommerzeitzone liegt, wird er zur Standardzeit konvertiert, da die Zielzeitzone als Abkürzung (PST) angegeben ist.

```
select listtime, convert_timezone('PST', listtime) from listing
where listid = 16;

     listtime       |   convert_timezone
--------------------+-------------------
2008-08-24 09:36:12     2008-08-24 01:36:12
```

Das folgende Beispiel konvertiert eine Zeitstempel-LISTTIME-Spalte von der Standardzeitzone UTC zur Zeitzone US/Pacific time. Die Zielzeitzone verwendet einen Zeitzonennamen, und der Zeitstempel liegt im Sommerzeitzeitraum, weshalb die Funktion die Sommerzeit ausgibt.

```
select listtime, convert_timezone('US/Pacific', listtime) from listing
where listid = 16;

     listtime       |   convert_timezone
--------------------+---------------------
2008-08-24 09:36:12 | 2008-08-24 02:36:12
```

Das folgende Beispiel konvertiert eine Zeitstempelzeichenfolge von EST zu PST:

```
select convert_timezone('EST', 'PST', '20080305 12:25:29');

 convert_timezone
-------------------
2008-03-05 09:25:29
```

Das folgende Beispiel konvertiert einen Zeitstempel zu US Eastern Standard Time, da die Zielzeitzone einen Zeitzonennamen (America/New York) verwendet und der Zeitstempel im Standardzeitzeitraum liegt.

```
select convert_timezone('America/New_York', '2013-02-01 08:00:00');
```

```
 convert_timezone
---------------------
2013-02-01 03:00:00
(1 row)
```

Das folgende Beispiel konvertiert einen Zeitstempel zu US Eastern Daylight Time, da die Zielzeitzone einen Zeitzonennamen (America/New York) verwendet und der Zeitstempel im Sommerzeitzeitraum liegt.

```
select convert_timezone('America/New_York', '2013-06-01 08:00:00');

 convert_timezone
---------------------
2013-06-01 04:00:00
(1 row)
```

Das folgende Beispiel illustriert die Verwendung von Verschiebungen.

```
SELECT CONVERT_TIMEZONE('GMT','NEWZONE +2','2014-05-17 12:00:00') as newzone_plus_2,
CONVERT_TIMEZONE('GMT','NEWZONE-2:15','2014-05-17 12:00:00') as newzone_minus_2_15,
CONVERT_TIMEZONE('GMT','America/Los_Angeles+2','2014-05-17 12:00:00') as la_plus_2,
CONVERT_TIMEZONE('GMT','GMT+2','2014-05-17 12:00:00') as gmt_plus_2;

   newzone_plus_2    | newzone_minus_2_15  |      la_plus_2      |     gmt_plus_2
---------------------+---------------------+---------------------+---------------------
2014-05-17 10:00:00 | 2014-05-17 14:15:00 | 2014-05-17 10:00:00 | 2014-05-17 10:00:00
(1 row)
```

Funktion CURRENT_DATE

CURRENT_DATE gibt ein Datum in der Zeitzone der aktuellen Sitzung (standardmäßig UTC) im Standardformat aus: JJJJ-MM-TT.

> **Note**
>
> CURRENT_DATE gibt das Startdatum für die aktuelle Transaktion aus, nicht für den Start der aktuellen Anweisung. Angenommen, Sie starten eine mehrere Anweisungen umfassende Transaktion am 01.10.08 um 23:59 Uhr und die Anweisung mit CURRENT_DATE wird am 02.10.08 um 00:00 Uhr ausgeführt. CURRENT_DATE gibt dann `10/01/08` zurück, nicht `10/02/08`.

Syntax

```
CURRENT_DATE
```

Rückgabetyp

DATUM

Beispiele

Das folgende Beispiel gibt das aktuelle Datum zurück (in AWS-Region dem die Funktion ausgeführt wird).

```
select current_date;

   date
------------
2008-10-01
```

Mit dem folgenden Beispiel werden eine Tabelle erstellt, eine Zeile eingefügt, deren Standardwert für die Spalte `todays_date` CURRENT_DATE lautet, und dann alle Zeilen in der Tabelle ausgewählt.

```
CREATE TABLE insert_dates(
    label varchar(128) NOT NULL,
    todays_date DATE DEFAULT CURRENT_DATE);

INSERT INTO insert_dates(label)
VALUES('Date row inserted');

SELECT * FROM insert_dates;


 label            | todays_date
------------------+-------------
Date row inserted | 2023-05-10
```

Funktion DATE_CMP

DATE_CMP vergleicht zwei Datumsangaben. Die Funktion gibt `0` aus, wenn beide identisch sind, sowie 1, wenn date1 größer ist, und `-1`, wenn date2 größer ist.

Syntax

```
DATE_CMP(date1, date2)
```

Argumente

date1

Eine Spalte vom Datentyp DATE oder ein Ausdruck, der zu einem DATE-Typ ausgewertet wird.

date2

Eine Spalte vom Datentyp DATE oder ein Ausdruck, der zu einem DATE-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Die folgende Abfrage vergleicht die DATE-Werte der Spalte CALDATE mit dem Datum 4. Januar 2008 und gibt aus, ob der Wert in CALDATE vor (-1), am (0) oder nach dem (1) 4. Januar 2008 liegt:

```
select caldate, '2008-01-04',
date_cmp(caldate,'2008-01-04')
from date
order by dateid
limit 10;

 caldate   |  ?column?  | date_cmp
-----------+------------+----------
2008-01-01 | 2008-01-04 |       -1
2008-01-02 | 2008-01-04 |       -1
2008-01-03 | 2008-01-04 |       -1
2008-01-04 | 2008-01-04 |        0
2008-01-05 | 2008-01-04 |        1
2008-01-06 | 2008-01-04 |        1
2008-01-07 | 2008-01-04 |        1
2008-01-08 | 2008-01-04 |        1
2008-01-09 | 2008-01-04 |        1
2008-01-10 | 2008-01-04 |        1
(10 rows)
```

Funktion DATE_CMP_TIMESTAMP

DATE_CMP_TIMESTAMP vergleicht ein Datum mit einem Zeitstempel und gibt `0` aus, wenn die Werte identisch sind, sowie 1, wenn date chronologisch größer ist, und `-1`, wenn timestamp größer ist.

Syntax

```
DATE_CMP_TIMESTAMP(date, timestamp)
```

Argumente

date

Eine Spalte vom Datentyp `DATE` oder ein Ausdruck, der zu einem `DATE`-Typ ausgewertet wird.

timestamp

Eine Spalte vom Datentyp `TIMESTAMP` oder ein Ausdruck, der zu einem `TIMESTAMP`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht das Datum `2008-06-18` mit LISTTIME. Die Werte der Spalte LISTTIME sind Zeitstempel. Vor diesem Datum erstellte Auflistungen geben 1 aus, danach erstellte Auflistungen `-1`.

```
select listid, '2008-06-18', listtime,
date_cmp_timestamp('2008-06-18', listtime)
from listing
order by 1, 2, 3, 4
limit 10;

 listid |  ?column?  |      listtime       | date_cmp_timestamp
--------+------------+---------------------+--------------------
      1 | 2008-06-18 | 2008-01-24 06:43:29 |                  1
      2 | 2008-06-18 | 2008-03-05 12:25:29 |                  1
      3 | 2008-06-18 | 2008-11-01 07:35:33 |                 -1
```

```
      4 | 2008-06-18 | 2008-05-24 01:18:37 |                  1
      5 | 2008-06-18 | 2008-05-17 02:29:11 |                  1
      6 | 2008-06-18 | 2008-08-15 02:08:13 |                 -1
      7 | 2008-06-18 | 2008-11-15 09:38:15 |                 -1
      8 | 2008-06-18 | 2008-11-09 05:07:30 |                 -1
      9 | 2008-06-18 | 2008-09-09 08:03:36 |                 -1
     10 | 2008-06-18 | 2008-06-17 09:44:54 |                  1
(10 rows)
```

Funktion DATE_CMP_TIMESTAMPTZ

DATE_CMP_TIMESTAMPTZ vergleicht ein Datum mit einem Zeitstempel mit Zeitzone und gibt `0` aus, wenn die Werte identisch sind, sowie 1, wenn date chronologisch größer ist, und `-1`, wenn timestamptz größer ist.

Syntax

```
DATE_CMP_TIMESTAMPTZ(date, timestamptz)
```

Argumente

date

Eine Spalte vom Datentyp `DATE` oder ein Ausdruck, der implizit zu einem `DATE`-Typ ausgewertet wird.

timestamptz

Eine Spalte vom Datentyp `TIMESTAMPTZ` oder ein Ausdruck, der implizit zu einem `TIMESTAMPTZ`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht das Datum `2008-06-18` mit LISTTIME. Vor diesem Datum erstellte Auflistungen geben 1 aus, danach erstellte Auflistungen `-1`.

```
select listid, '2008-06-18', CAST(listtime AS timestamptz),
date_cmp_timestamptz('2008-06-18', CAST(listtime AS timestamptz))
```

```
from listing
order by 1, 2, 3, 4
limit 10;

 listid |  ?column?  |      timestamptz       | date_cmp_timestamptz
--------+------------+------------------------+----------------------
      1 | 2008-06-18 | 2008-01-24 06:43:29+00 |                    1
      2 | 2008-06-18 | 2008-03-05 12:25:29+00 |                    1
      3 | 2008-06-18 | 2008-11-01 07:35:33+00 |                   -1
      4 | 2008-06-18 | 2008-05-24 01:18:37+00 |                    1
      5 | 2008-06-18 | 2008-05-17 02:29:11+00 |                    1
      6 | 2008-06-18 | 2008-08-15 02:08:13+00 |                   -1
      7 | 2008-06-18 | 2008-11-15 09:38:15+00 |                   -1
      8 | 2008-06-18 | 2008-11-09 05:07:30+00 |                   -1
      9 | 2008-06-18 | 2008-09-09 08:03:36+00 |                   -1
     10 | 2008-06-18 | 2008-06-17 09:44:54+00 |                    1
(10 rows)
```

Funktion DATEADD

Erhöht einen DATE-, TIME-, TIMETZ- oder TIMESTAMP-Wert um ein bestimmtes Intervall.

Syntax

```
DATEADD( datepart, interval, {date|time|timetz|timestamp} )
```

Argumente

datepart

Der Datumsteil (z. B. Jahr, Monat, Tag oder Stunde), für den die Funktion gilt. Weitere Informationen finden Sie unter Datumsteile für Datums- oder Zeitstempelfunktionen.

Intervall

Eine Ganzzahl, di das Intervall angibt (z. B. eine Anzahl von Tagen), das dem Zielausdruck hinzugefügt werden soll. Bei einer negativen Ganzzahl wird das Intervall subtrahiert.

date|time|timetz|timestamp

Eine DATE-, TIME-, TIMETZ- oder TIMESTAMP-Spalte bzw. ein entsprechender Ausdruck, die/der implizit zu einem DATE, TIME, TIMETZ oder TIMESTAMP konvertiert wird. Der DATE-, TIME-, TIMETZ- oder TIMESTAMP-Ausdruck muss den angegebenen Datumsteil enthalten.

Rückgabetyp

TIMESTAMP oder TIME oder TIMEZ abhängig vom Eingabedatendatentyp.

Beispiele mit einer DATE-Spalte

Im folgenden Beispiel werden 30 Tage zu jedem Datum hinzugefügt, das in der DATE-Tabelle vorhanden ist.

```
select dateadd(day,30,caldate) as novplus30
from date
where month='NOV'
order by dateid;

novplus30
---------------------
2008-12-01 00:00:00
2008-12-02 00:00:00
2008-12-03 00:00:00
...
(30 rows)
```

Im folgenden Beispiel werden 18 Monate zu einem Literal-Datumswert hinzugefügt.

```
select dateadd(month,18,'2008-02-28');

date_add
---------------------
2009-08-28 00:00:00
(1 row)
```

Der Standard-Spaltenname für eine DATEADD-Funktion ist DATE_ADD. Der Standard-Zeitstempel für einen Datumswert ist `00:00:00`.

Im folgenden Beispiel werden 30 Minuten zu einem Datumswert hinzugefügt, der keinen Zeitstempel angibt.

```
select dateadd(m,30,'2008-02-28');

date_add
---------------------
```

```
2008-02-28 00:30:00
(1 row)
```

Sie können Datumsteile ausschreiben oder abkürzen. In diesem Fall steht das m für Minuten, nicht für Monate.

Beispiele mit einer TIME-Spalte

Die folgende Beispieltabelle TIME_TEST enthält eine Spalte TIME_VAL (Typ TIME) mit drei eingefügten Werten.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

Im folgenden Beispiel werden jedem TIME_VAL in der TIME_TEST-Tabelle 5 Minuten hinzugefügt.

```
select dateadd(minute,5,time_val) as minplus5 from time_test;

minplus5
---------------
20:05:00
00:05:00.5550
01:03:00
```

Im folgenden Beispiel werden 8 Stunden zu einem Literal-Zeitwert hinzugefügt.

```
select dateadd(hour, 8, time '13:24:55');

date_add
---------------
21:24:55
```

Das folgende Beispiel wird angezeigt, wenn eine Zeit über 24:00:00 oder unter 00:00:00 liegt.

```
select dateadd(hour, 12, time '13:24:55');
```

```
date_add
---------------
01:24:55
```

Beispiele mit einer TIMETZ-Spalte

Die Ausgabewerte in diesen Beispielen sind in der Standardzeitzone UTC angegeben.

Die folgende Beispieltabelle TIMETZ_TEST enthält eine Spalte TIMETZ_VAL (Typ TIMETZ) mit drei eingefügten Werten.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

Im folgenden Beispiel werden jedem TIMETZ_VAL in der TIMETZ_TEST-Tabelle 5 Minuten hinzugefügt.

```
select dateadd(minute,5,timetz_val) as minplus5_tz from timetz_test;

minplus5_tz
---------------
04:05:00+00
00:05:00.5550+00
06:03:00+00
```

Im folgenden Beispiel werden 2 Stunden zu einem Literal-Timetz-Wert hinzugefügt.

```
select dateadd(hour, 2, timetz '13:24:55 PST');

date_add
---------------
23:24:55+00
```

Beispiele mit einer TIMESTAMP-Spalte

Die Ausgabewerte in diesen Beispielen sind in der Standardzeitzone UTC angegeben.

Die folgende Beispieltabelle TIMESTAMP_TEST enthält eine Spalte TIMESTAMP_VAL (Typ TIMESTAMP) mit drei eingefügten Werten.

```
SELECT timestamp_val FROM timestamp_test;

timestamp_val
------------------
1988-05-15 10:23:31
2021-03-18 17:20:41
2023-06-02 18:11:12
```

Im folgenden Beispiel werden nur den TIMESTAMP_VAL-Werten in TIMESTAMP_TEST aus der Zeit vor dem Jahr 2000 20 Jahre hinzugefügt.

```
SELECT dateadd(year,20,timestamp_val)
FROM timestamp_test
WHERE timestamp_val < to_timestamp('2000-01-01 00:00:00', 'YYYY-MM-DD HH:MI:SS');

date_add
---------------
2008-05-15 10:23:31
```

Im folgenden Beispiel werden einem literalen Zeitstempelwert, der ohne Sekundenanzeige geschrieben wurde, 5 Sekunden hinzugefügt.

```
SELECT dateadd(second, 5, timestamp '2001-06-06');

date_add
---------------
2001-06-06 00:00:05
```

Nutzungshinweise

Die Funktionen DATEADD(month, ...) und ADD_MONTHS behandeln Daten am Ende von Monaten in unterschiedlicher Weise:

- ADD_MONTHS: Wenn das Datum, das Sie hinzufügen, der letzte Tag des Monats ist, ist das Ergebnis immer der letzte Tag des Ergebnismonats, unabhängig von der Länge des Monats. Zum Beispiel: 30. April + 1 Monat = 31. Mai.

  ```
  select add_months('2008-04-30',1);
  ```

```
add_months
---------------------
2008-05-31 00:00:00
(1 row)
```

- DATEADD: Wenn das Datum, zu dem Sie hinzufügen, weniger Tage enthält als der Ergebnismonat, entspricht das Ergebnis dem Tag des Ergebnismonats, nicht dem letzten Tag des Monats. Zum Beispiel: 30. April + 1 Monat = 30. Mai.

```
select dateadd(month,1,'2008-04-30');

date_add
---------------------
2008-05-30 00:00:00
(1 row)
```

Die Funktion DATEADD behandelt das Schaltjahrdatum 02-29 anders als DATEADD(month, 12, ...) oder DATEADD(year, 1, ...).

```
select dateadd(month,12,'2016-02-29');

date_add
---------------------
2017-02-28 00:00:00

select dateadd(year, 1, '2016-02-29');

date_add
---------------------
2017-03-01 00:00:00
```

Funktion DATEDIFF

DATEDIFF gibt die Differenz zwischen den Datumsteilen zweier Datums- oder Uhrzeitausdrücke aus.

Syntax

```
DATEDIFF( datepart, {date|time|timetz|timestamp}, {date|time|timetz|timestamp} )
```

Argumente

datepart

Der spezifische Teil des Datums- oder Zeitwerts (Jahr, Monat oder Tag, Stunde, Minute, Sekunde, Millisekunde oder Mikrosekunde) auf den die Funktion angewendet wird. Weitere Informationen finden Sie unter Datumsteile für Datums- oder Zeitstempelfunktionen.

Insbesondere bestimmt DATEDIFF die Anzahl von Datumsteilgrenzen, die zwischen zwei Ausdrücken überschritten werden. Nehmen wir an, dass Sie den Jahresdifferenz zwischen zwei Daten berechnen möchten, `12-31-2008` und `01-01-2009`. In diesem Fall gibt die Funktion 1 Jahr zurück, obwohl die Daten nur einen Tag auseinanderliegen. Wenn Sie die Differenz in Stunden zwischen zwei Zeitstempeln, `01-01-2009 8:30:00` und `01-01-2009 10:00:00` ermitteln, ist das Ergebnis 2 Stunden. Wenn Sie die Differenz in Stunden zwischen zwei Zeitstempeln, `8:30:00` und `10:00:00` ermitteln, ist das Ergebnis 2 Stunden.

date|time|timetz|timestamp

Eine DATE-, TIME-, TIMETZ- oder TIMESTAMP-Spalte bzw. ein entsprechender Ausdruck, die/der implizit zu einem DATE, TIME, TIMETZ oder TIMESTAMP konvertiert wird. Beide Ausdrücke müssen den angegebenen Datums- oder Zeitteil enthalten. Wenn das zweite Datum bzw. die zweite Uhrzeit größer als das/die erste ist, ist das Ergebnis positiv. Wenn das zweite Datum bzw. die zweite Uhrzeit vor dem/der ersten liegt, ist das Ergebnis negativ.

Rückgabetyp

BIGINT

Beispiele mit einer DATE-Spalte

Im folgenden Beispiel wird die Differenz als Anzahl von Wochen zwischen zwei Literal-Datumswerten berechnet.

```
select datediff(week,'2009-01-01','2009-12-31') as numweeks;

numweeks
----------
52
(1 row)
```

Im folgenden Beispiel wird die Differenz in Stunden zwischen zwei Literal-Datumswerten ermittelt. Wenn Sie den Zeitwert für ein Datum nicht angeben, wird standardmäßig 00:00:00 verwendet.

```
select datediff(hour, '2023-01-01', '2023-01-03 05:04:03');

date_diff
----------
53
(1 row)
```

Im folgenden Beispiel wird die Differenz in Tagen zwischen zwei TIMESTAMETZ-Literalwerten ermittelt.

```
Select datediff(days, 'Jun 1,2008  09:59:59 EST', 'Jul 4,2008  09:59:59 EST')

date_diff
----------
33
```

Im folgenden Beispiel wird die Differenz in Tagen zwischen zwei Daten in derselben Zeile einer Tabelle ermittelt.

```
select * from date_table;

start_date |   end_date
-----------+-----------
2009-01-01 | 2009-03-23
2023-01-04 | 2024-05-04
(2 rows)

select datediff(day, start_date, end_date) as duration from date_table;

duration
---------
      81
     486
(2 rows)
```

Im folgenden Beispiel wird die Differenz als Anzahl von Quartalen zwischen einem in der Vergangenheit liegenden Literalwert und dem heutigen Datum berechnet. Bei diesem Beispiel

wird davon ausgegangen, dass das aktuelle Datum der 5. Juni 2008 ist. Sie können Datumsteile ausschreiben oder abkürzen. Der Standard-Spaltenname für die DATEDIFF-Funktion ist DATE_DIFF.

```
select datediff(qtr, '1998-07-01', current_date);

date_diff
-----------
40
(1 row)
```

Das folgende Beispiel verbindet die Tabellen SALES und LISTING zur Berechnung, wie viel Tage nach ihrer Auflistung Tickets für die Auflistungen 1000 bis 1005 verkauft wurden. Die längste Wartezeit für den Verkauf dieser Auflistungen betrug 15 Tage, und die kürzeste lag unter einem Tag (0 Tage).

```
select priceperticket,
datediff(day, listtime, saletime) as wait
from sales, listing where sales.listid = listing.listid
and sales.listid between 1000 and 1005
order by wait desc, priceperticket desc;

priceperticket | wait
---------------+------
 96.00         |   15
 123.00        |   11
 131.00        |    9
 123.00        |    6
 129.00        |    4
 96.00         |    4
 96.00         |    0
(7 rows)
```

Dieses Beispiel berechnet die durchschnittliche Zahl von Stunden, für die Verkäufer auf alle Ticketverkäufe warteten.

```
select avg(datediff(hours, listtime, saletime)) as avgwait
from sales, listing
where sales.listid = listing.listid;

avgwait
---------
```

```
465
(1 row)
```

Beispiele mit einer TIME-Spalte

Die folgende Beispieltabelle TIME_TEST enthält eine Spalte TIME_VAL (Typ TIME) mit drei eingefügten Werten.

```
select time_val from time_test;

time_val
--------------------
20:00:00
00:00:00.5550
00:58:00
```

Im folgenden Beispiel wird die Differenz als Anzahl von Stunden zwischen der TIME_VAL-Spalte und einem Zeitliteral berechnet.

```
select datediff(hour, time_val, time '15:24:45') from time_test;

 date_diff
-----------
        -5
        15
        15
```

Im folgenden Beispiel wird die Differenz als Anzahl von Minuten zwischen zwei Literal-Zeitwerten berechnet.

```
select datediff(minute, time '20:00:00', time '21:00:00') as nummins;

nummins
----------
60
```

Beispiele mit einer TIMETZ-Spalte

Die folgende Beispieltabelle TIMETZ_TEST enthält eine Spalte TIMETZ_VAL (Typ TIMETZ) mit drei eingefügten Werten.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

Im folgenden Beispiel werden die Differenzen als Anzahl von Stunden zwischen dem TIMETZ-Literal und timetz_val berechnet.

```
select datediff(hours, timetz '20:00:00 PST', timetz_val) as numhours from timetz_test;

numhours
----------
0
-4
1
```

Im folgenden Beispiel wird die Differenz als Anzahl von Stunden zwischen zwei Literal-TIMETZ-Werten berechnet.

```
select datediff(hours, timetz '20:00:00 PST', timetz '00:58:00 EST') as numhours;

numhours
----------
1
```

Funktion DATE_PART

DATE_PART extrahiert Datumsteilwerte aus einem Ausdruck. DATE_PART ist synonym mit der Funktion PGDATE_PART.

Syntax

```
DATE_PART(datepart, {date|timestamp})
```

Argumente

datepart

Ein Bezeichnerliteral oder eine Zeichenfolge des spezifischen Teils des Datumswertes (z. B. Jahr, Monat oder Tag), für den die Funktion gilt. Weitere Informationen finden Sie unter Datumsteile für Datums- oder Zeitstempelfunktionen.

{date|timestamp}

Eine Datums- oder Zeitstempelspalte bzw. ein entsprechender Ausdruck, die/der implizit zu einem Datum oder Zeitstempel konvertiert wird. Die Spalte bzw. der Ausdruck unter date oder timestamp muss den in datepart angegebenen Datumsteil enthalten.

Rückgabetyp

DOUBLE

Beispiele

Der Standard-Spaltenname für die DATE_PART-Funktion ist `pgdate_part`.

Weitere Hinweise zu den Daten, die in einigen dieser Beispiele verwendet werden, finden Sie unter Beispieldatenbank.

Im folgenden Beispiel wird der Minutenwert aus einem Zeitstempelliteral ermittelt.

```
SELECT DATE_PART(minute, timestamp '20230104 04:05:06.789');

pgdate_part
-----------
          5
```

Im folgenden Beispiel wird der Wochenwert aus einem Zeitstempelliteral ermittelt. Die Berechnung der Wochenzahl erfolgt gemäß ISO-Standard 8601. Weitere Informationen finden Sie unter ISO 8601 in Wikipedia.

```
SELECT DATE_PART(week, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         18
```

Im folgenden Beispiel wird der Tag des Monats aus einem Zeitstempelliteral ermittelt.

```
SELECT DATE_PART(day, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
          2
```

Im folgenden Beispiel wird der Wochentag aus einem Zeitstempelliteral ermittelt. Die Datumsberechnung „Wochentag“ ist eine Ganzzahl zwischen 0 und 6, beginnend mit Sonntag.

```
SELECT DATE_PART(dayofweek, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
          1
```

Das folgende Beispiel ermittelt das Jahrhundert aus einem Zeitstempelliteral. Die Berechnung des Jahrhunderts erfolgt gemäß ISO-Standard 8601. Weitere Informationen finden Sie unter ISO 8601 in Wikipedia.

```
SELECT DATE_PART(century, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         21
```

Im folgenden Beispiel wird das Jahrtausend aus einem Zeitstempelliteral ermittelt. Die Berechnung des Jahrtausends erfolgt gemäß ISO-Standard 8601. Weitere Informationen finden Sie unter ISO 8601 in Wikipedia.

```
SELECT DATE_PART(millennium, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
          3
```

Im folgenden Beispiel werden die Mikrosekunden aus einem Zeitstempelliteral ermittelt. Die Berechnung der Mikrosekunden erfolgt gemäß ISO-Standard 8601. Weitere Informationen finden Sie unter ISO 8601 in Wikipedia.

```
SELECT DATE_PART(microsecond, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
    789000
```

Im folgenden Beispiel wird der Monat aus einem Datumsliteral ermittelt.

```
SELECT DATE_PART(month, date '20220502');

pgdate_part
-----------
          5
```

Im folgenden Beispiel wird die Funktion DATE_PART auf eine Spalte in einer Tabelle angewendet.

```
SELECT date_part(w, listtime) AS weeks, listtime
FROM listing
WHERE listid=10


weeks |      listtime
------+---------------------
 25   | 2008-06-17 09:44:54
(1 row)
```

Sie können Datumsteile ausschreiben oder abkürzen; in diesem Fall steht w für Wochen.

Der Datumsteil „Wochentag“ gibt eine Ganzzahl zwischen 0 und 6 aus, beginnend mit Sonntag. Verwenden Sie DATE_PART mit dow (DAYOFWEEK) zur Anzeige von Ereignissen an einem Samstag.

```
SELECT date_part(dow, starttime) AS dow, starttime
FROM event
WHERE date_part(dow, starttime)=6
ORDER BY 2,1;

 dow |      starttime
-----+---------------------
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
```

```
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
...
(1147 rows)
```

Funktion DATE_PART_YEAR

Die Funktion DATE_PART_YEAR extrahiert das Jahr aus einem Datum.

Syntax

```
DATE_PART_YEAR(date)
```

Argument

date

Eine Spalte vom Datentyp DATE oder ein Ausdruck, der implizit zu einem DATE-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Im folgenden Beispiel wird das Jahr aus einem Datumsliteral ermittelt.

```
SELECT DATE_PART_YEAR(date '20220502 04:05:06.789');

date_part_year
---------------
2022
```

Das folgende Beispiel extrahiert das Jahr aus der Spalte CALDATE. Die Werte in der Spalte CALDATE sind Datumsangaben. Weitere Informationen zu den hier verwendeten Daten finden Sie unter Beispieldatenbank.

```
select caldate, date_part_year(caldate)
from date
```

```
order by
dateid limit 10;

 caldate   | date_part_year
-----------+----------------
2008-01-01 |           2008
2008-01-02 |           2008
2008-01-03 |           2008
2008-01-04 |           2008
2008-01-05 |           2008
2008-01-06 |           2008
2008-01-07 |           2008
2008-01-08 |           2008
2008-01-09 |           2008
2008-01-10 |           2008
(10 rows)
```

Funktion DATE_TRUNC

Die Funktion DATE_TRUNC verkürzt alle Zeitstempelausdrücke oder Literale auf der Grundlage des angegebenen Datumsteils, beispielsweise Stunde, Tag oder Monat.

Syntax

```
DATE_TRUNC('datepart', timestamp)
```

Argumente

datepart

Der Datumsteil, auf den der Zeitstempelwert verkürzt werden soll. Die Eingabe timestamp wird entsprechend der Genauigkeit der Eingabe datepart verkürzt. Bei Verwendung von `month` beispielsweise wird auf den ersten Tag des Monats verkürzt. Gültige Formate sind folgende:

- Mikrosekunde, Mikrosekunden
- Millisekunde, Millisekunden
- Sekunde, Sekunden
- Minute, Minuten
- Stunde, Stunden
- Tag, Tage

- Woche, Wochen
- Monat, Monate
- Quartal, Quartale
- Jahr, Jahre
- Dekade, Dekaden
- Jahrhundert, Jahrhunderte
- Jahrtausend, Jahrtausende

Weitere Informationen zu Abkürzungen einiger Formate finden Sie unter Datumsteile für Datums- oder Zeitstempelfunktionen

timestamp

Eine Zeitstempelspalte bzw. ein entsprechender Ausdruck, die/der implizit zu einem Zeitstempel konvertiert wird.

Rückgabetyp

TIMESTAMP

Beispiele

Verkürzen des Eingabezeitstempels auf die Sekunde

```
SELECT DATE_TRUNC('second', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:05:06
```

Verkürzen des Eingabezeitstempels auf die Minute

```
SELECT DATE_TRUNC('minute', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:05:00
```

Verkürzen des Eingabezeitstempels auf die Stunde

```
SELECT DATE_TRUNC('hour', TIMESTAMP '20200430 04:05:06.789');
date_trunc
```

```
2020-04-30 04:00:00
```

Verkürzen des Eingabezeitstempels auf den Tag

```
SELECT DATE_TRUNC('day', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 00:00:00
```

Verkürzen des Eingabezeitstempels auf den ersten Tag eines Monats

```
SELECT DATE_TRUNC('month', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-01 00:00:00
```

Verkürzen des Eingabezeitstempels auf den ersten Tag eines Quartals

```
SELECT DATE_TRUNC('quarter', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-01 00:00:00
```

Verkürzen des Eingabezeitstempels auf den ersten Tag eines Jahres

```
SELECT DATE_TRUNC('year', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-01-01 00:00:00
```

Verkürzen des Eingabezeitstempels auf den ersten Tag eines Jahrhunderts

```
SELECT DATE_TRUNC('millennium', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2001-01-01 00:00:00
```

Verkürzen des Eingabezeitstempels auf den Montag der Woche.

```
select date_trunc('week', TIMESTAMP '20220430 04:05:06.789');
date_trunc
2022-04-25 00:00:00
```

Im folgenden Beispiel verwendet die Funktion DATE_TRUNC den Datumsteil „Woche“ zur Rückgabe des Datums des Montags jeder Woche.

```
select date_trunc('week', saletime), sum(pricepaid) from sales where
saletime like '2008-09%' group by date_trunc('week', saletime) order by 1;

date_trunc  |    sum
------------+-------------
2008-09-01  | 2474899
2008-09-08  | 2412354
2008-09-15  | 2364707
2008-09-22  | 2359351
2008-09-29  |  705249
```

Funktion EXTRACT

Die EXTRACT-Funktion gibt einen Datums- oder Uhrzeitanteil aus einem TIMESTAMP-, TIMESTAMPTZ-, TIME-, TIMETZ-, INTERVAL YEAR TO MONTH- oder INTERVAL DAY TO SECOND-Wert zurück. Beispiele hierfür sind ein Tag, Monat, Jahr, eine Stunde, Minute, Sekunde, Millisekunde oder Mikrosekunde aus einem Zeitstempel.

Syntax

```
EXTRACT(datepart FROM source)
```

Argumente

datepart

Das zu extrahierende Unterfeld eines Datums- oder Uhrzeitwerts, z. B. Tag, Monat, Jahr, Stunde, Minute, Sekunde, Millisekunde oder Mikrosekunde. Für mögliche Werte vgl. Datumsteile für Datums- oder Zeitstempelfunktionen.

source

Eine Spalte oder ein Ausdruck, der den Datentyp TIMESTAMP, TIMESTAMPTZ, TIME, TIMETZ, INTERVAL YEAR TO MONTH oder INTERVAL DAY TO SECOND ergibt.

Rückgabetyp

INTEGER, wenn der Quellwert den Datentyp TIMESTAMP, TIME, TIMETZ, INTERVAL YEAR TO MONTH oder INTERVAL DAY TO SECOND ergibt.

DOUBLE PRECISION, wenn der Wert source zum Datentyp TIMESTAMPTZ ausgewertet wird.

Beispiele mit TIMESTAMP

Im folgenden Beispiel werden die Wochennummern für Verkäufe bestimmt, bei denen der gezahlte Preis 10 000 USD oder mehr betrug. In diesem Beispiel werden die TICKET-Daten verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

```
select salesid, extract(week from saletime) as weeknum
from sales
where pricepaid > 9999
order by 2;

salesid | weeknum
--------+---------
 159073 |       6
 160318 |       8
 161723 |      26
```

Im folgenden Beispiel wird der Minutenwert aus einem Literal-Zeitstempel-Wert zurückgegeben.

```
select extract(minute from timestamp '2009-09-09 12:08:43');

date_part
-----------
8
```

Im folgenden Beispiel wird der Millisekundenwert aus einem Literal-Timestamp-Wert zurückgegeben.

```
select extract(ms from timestamp '2009-09-09 12:08:43.101');

date_part
-----------
101
```

Beispiele mit TIMESTAMPTZ

Im folgenden Beispiel wird der Jahreswert aus einem Literal-Timestamptz-Wert zurückgegeben.

```
select extract(year from timestamptz '1.12.1997 07:37:16.00 PST');

date_part
-----------
```

```
1997
```

Beispiele mit TIME

Die folgende Beispieltabelle TIME_TEST enthält eine Spalte TIME_VAL (Typ TIME) mit drei eingefügten Werten.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

Im folgenden Beispiel werden die Minuten aus jedem time_val extrahiert.

```
select extract(minute from time_val) as minutes from time_test;

minutes
-----------
         0
         0
         58
```

Im folgenden Beispiel werden die Stunden aus jedem time_val extrahiert.

```
select extract(hour from time_val) as hours from time_test;

hours
-----------
         20
         0
         0
```

Im folgenden Beispiel wird Millisekunden aus einem Literalwert extrahiert.

```
select extract(ms from time '18:25:33.123456');

 date_part
-----------
```

```
     123
```

Beispiele mit TIMETZ

Die folgende Beispieltabelle TIMETZ_TEST enthält eine Spalte TIMETZ_VAL (Typ TIMETZ) mit drei eingefügten Werten.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

Im folgenden Beispiel werden die Stunden aus jedem timetz_val extrahiert.

```
select extract(hour from timetz_val) as hours from time_test;

hours
-----------
         4
         0
         5
```

Im folgenden Beispiel wird Millisekunden aus einem Literalwert extrahiert. Literale werden nicht in UTC konvertiert, bevor die Extraktion verarbeitet wurde.

```
select extract(ms from timetz '18:25:33.123456 EST');

 date_part
-----------
     123
```

Im folgenden Beispiel wird die Stunde der Zeitzonenabweichung von UTC aus einem Literal-Timetz-Wert zurückgegeben.

```
select extract(timezone_hour from timetz '1.12.1997 07:37:16.00 PDT');

date_part
```

```
-----------
-7
```

Beispiele mit INTERVALL YEAR TO MONTH und INTERVAL DAY TO SECOND

Im folgenden Beispiel wird der Tagesteil 1 von INTERVAL DAY TO SECOND extrahiert, der 36 Stunden definiert, was 1 Tag 12 Stunden entspricht.

```
select EXTRACT('days' from INTERVAL '36 hours' DAY TO SECOND)

 date_part
------------------
 1
```

Im folgenden Beispiel wird der Monatsteil 3 von YEAR TO MONTH extrahiert, der 15 Monate definiert, was 1 Jahr 3 Monaten entspricht.

```
select EXTRACT('month' from INTERVAL '15 months' YEAR TO MONTH)

 date_part
------------------
 3
```

Im folgenden Beispiel wird der Monatsteil 6 von 30 Monaten extrahiert, was 2 Jahren 6 Monaten entspricht.

```
select EXTRACT('month' from INTERVAL '30' MONTH)

 date_part
------------------
 6
```

Im folgenden Beispiel wird der Stundenteil 2 von 50 Stunden extrahiert, was 2 Tagen 2 Stunden entspricht.

```
select EXTRACT('hours' from INTERVAL '50' HOUR)

 date_part
------------------
```

```
 2
```

Im folgenden Beispiel wird der Minutenteil 11 von 1 Stunde 11 Minuten 11,123 Sekunden extrahiert.

```
select EXTRACT('minute' from INTERVAL '70 minutes 70.123 seconds' MINUTE TO SECOND)

 date_part
------------------
 11
```

Im folgenden Beispiel wird der Sekundenanteil `1.11` von 1 Tag 1 Stunde 1 Minute 1,11 Sekunden extrahiert.

```
select EXTRACT('seconds' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND)

 date_part
------------------
 1.11
```

Im folgenden Beispiel wird die Gesamtzahl der Stunden in einem INTERVALL extrahiert. Jeder Teil wird extrahiert und zu einer Gesamtsumme hinzugefügt.

```
select EXTRACT('days' from INTERVAL '50' HOUR) * 24 + EXTRACT('hours' from INTERVAL
 '50' HOUR)

 ?column?
------------------
 50
```

Im folgenden Beispiel wird die Gesamtzahl der Sekunden in einem INTERVALL extrahiert. Jeder Teil wird extrahiert und zu einer Gesamtsumme hinzugefügt.

```
select EXTRACT('days' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 86400 +
       EXTRACT('hours' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 3600 +
       EXTRACT('minutes' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 60 +
       EXTRACT('seconds' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND)

 ?column?
------------------
 90061.11
```

Funktion GETDATE

GETDATE gibt das aktuelle Datum und die aktuelle Uhrzeit in der Zeitzone der aktuellen Sitzung (standardmäßig UTC) aus. Es gibt das Startdatum oder die Uhrzeit der aktuellen Anweisung zurück, auch wenn es sich innerhalb eines Transaktionsblocks befindet.

Syntax

```
GETDATE()
```

Die Klammern sind erforderlich.

Rückgabetyp

TIMESTAMP

Beispiele

Das folgende Beispiel verwendet die Funktion GETDATE zur Rückgabe des vollständigen Zeitstempels für das aktuelle Datum.

```
select getdate();

timestamp
---------------------
2008-12-04 16:10:43
```

Das folgende Beispiel verwendet die Funktion GETDATE innerhalb der Funktion TRUNC zur Rückgabe des aktuellen Datums ohne die Uhrzeit.

```
select trunc(getdate());

trunc
------------
2008-12-04
```

Funktion INTERVAL_CMP

INTERVAL_COMP vergleicht zwei Intervalle und gibt 1 aus, wenn das erste Intervall größer ist, -1, wenn das zweite Intervall größer ist, und 0, wenn die Intervalle identisch sind. Weitere Informationen finden Sie unter Beispiele für Intervallliterale ohne Qualifier-Syntax.

Syntax

```
INTERVAL_CMP(interval1, interval2)
```

Argumente

interval1

Ein Intervallliteralwert.

interval2

Ein Intervallliteralwert.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht den Wert `3 days` mit `1 year`.

```
select interval_cmp('3 days','1 year');

interval_cmp
--------------
-1
```

In diesem Beispiel wird der Wert `7 days` mit `1 week` verglichen.

```
select interval_cmp('7 days','1 week');

interval_cmp
--------------
0
```

Das folgende Beispiel vergleicht den Wert `1 year` mit `3 days`.

```
select interval_cmp('1 year','3 days');

interval_cmp
--------------
```

```
1
```

Funktion LAST_DAY

LAST_DAY gibt das Datum des letzten Tages des Monats aus, der date enthält. Der Typ der Rückgabe ist immer DATE, unabhängig vom Typ des date-Arguments.

Weitere Informationen zum Abrufen spezifischer Datumsteile finden Sie unter Funktion DATE_TRUNC.

Syntax

```
LAST_DAY( { date | timestamp } )
```

Argumente

date | timestamp

Eine Spalte vom Datentyp DATE oder TIMESTAMP bzw. ein Ausdruck, der implizit zu einem DATE- oder TIMESTAMP-Typ ausgewertet wird.

Rückgabetyp

DATUM

Beispiele

Das folgende Beispiel gibt das Datum des letzten Tages des aktuellen Monats zurück.

```
select last_day(sysdate);

  last_day
------------
 2014-01-31
```

Das folgende Beispiel gibt die Anzahl der an jedem der letzten 7 Tage des Monats verkauften Tickets zurück. Die Werte in der Spalte SALETIME sind Zeitstempel.

```
select datediff(day, saletime, last_day(saletime)) as "Days Remaining", sum(qtysold)
from sales
where datediff(day, saletime, last_day(saletime)) < 7
```

```
group by 1
order by 1;

days remaining |  sum
---------------+-------
             0 | 10140
             1 | 11187
             2 | 11515
             3 | 11217
             4 | 11446
             5 | 11708
             6 | 10988
(7 rows)
```

Funktion MONTHS_BETWEEN

MONTHS_BETWEEN bestimmt die Anzahl der Monate zwischen zwei Daten.

Wenn das erste Datum nach dem zweiten Datum liegt, ist das Ergebnis positiv, andernfalls ist es negativ.

Wenn eines der Argumente Null ist, ist das Ergebnis NULL.

Syntax

```
MONTHS_BETWEEN( date1, date2 )
```

Argumente

date1

Eine Spalte vom Datentyp DATE oder ein Ausdruck, der implizit zu einem DATE-Typ ausgewertet wird.

date2

Eine Spalte vom Datentyp DATE oder ein Ausdruck, der implizit zu einem DATE-Typ ausgewertet wird.

Rückgabetyp

FLOAT8

Der ganzzahlige Teil des Ergebnisses basiert auf der Differenz zwischen den Jahr- und Monat-Werten der Datumsangaben. Der Bruchteil des Ergebnisses wird aus den Tag- und Zeitstempelwerten der Datumsangabe unter Zugrundelegung eines Monats mit 31 Tagen berechnet.

Wenn date1 und date2 beide das gleiche Datum in einem Monat (zum Beispiel 1/15/14 und 2/15/14) oder den letzten Tag des Monats (zum Beispiel 8/31/14 und 9/30/14) enthalten, ist das Ergebnis eine Ganzzahl auf der Grundlage der Jahr- und Monat-Werte der Daten, unabhängig davon, ob der Zeitstempelteil, falls vorhanden, übereinstimmt.

Beispiele

Das folgende Beispiel gibt die Monate zwischen 1/18/1969 und 3/18/1969 zurück.

```
select months_between('1969-01-18', '1969-03-18')
as months;

months
----------
-2
```

Das folgende Beispiel gibt die Monate zwischen 18.01.1969 und 18.03.1969 zurück.

```
select months_between('1969-01-18', '1969-01-18')
as months;

months
----------
0
```

Das folgende Beispiel gibt die Monate zwischen dem ersten und der letzten Vorkommen eines Ereignisses zurück.

```
select eventname,
min(starttime) as first_show,
max(starttime) as last_show,
months_between(max(starttime),min(starttime)) as month_diff
from event
group by eventname
order by eventname
limit 5;

eventname         first_show             last_show              month_diff
```

```
------------------------------------------------------------------------
.38 Special       2008-01-21 19:30:00.0  2008-12-25 15:00:00.0  11.12
3 Doors Down      2008-01-03 15:00:00.0  2008-12-01 19:30:00.0  10.94
70s Soul Jam      2008-01-16 19:30:00.0  2008-12-07 14:00:00.0  10.7
A Bronx Tale      2008-01-21 19:00:00.0  2008-12-15 15:00:00.0  10.8
A Catered Affair  2008-01-08 19:30:00.0  2008-12-19 19:00:00.0  11.35
```

Funktion NEXT_DAY

NEXT_DAY gibt das Datum der ersten Instanz des bestimmten Tages aus, der nach dem angegebenen Datum liegt.

Wenn der day-Wert derselbe Wochentag wie das angegebene Datum ist, wird die nächste Wiederkehr des Tages ausgegeben.

Syntax

```
NEXT_DAY( { date | timestamp }, day )
```

Argumente

date | timestamp

Eine Spalte vom Datentyp `DATE` oder `TIMESTAMP` bzw. ein Ausdruck, der implizit zu einem `DATE`- oder `TIMESTAMP`-Typ ausgewertet wird.

Tag

Eine Zeichenfolge, die den Namen eines Tages enthält. Die Groß- und Kleinschreibung spielt keine Rolle.

Gültige Werte sind:

Tag	Werte
Sonntag	Su, Sun, Sunday
Montag	M, Mo, Mon, Monday
Dienstag	Di, Die, Dien, Dienstag
Mittwoch	M, Mi, Mit, Mittwoch

Tag	Werte
Donnerstag	Th, Thu, Thurs, Thursday
Freitag	F, Fr, Fri, Friday
Samstag	Sa, Sat, Saturday

Rückgabetyp

DATUM

Beispiele

Das folgende Beispiel gibt das Datum des ersten Dienstags nach 8/20/2014 aus.

```
select next_day('2014-08-20','Tuesday');

next_day
-----------
2014-08-26
```

Das folgende Beispiel gibt das Datum des ersten Dienstags nach dem 01.01.2008 um 5:54:44 zurück.

```
select listtime, next_day(listtime, 'Tue') from listing limit 1;

listtime            | next_day
--------------------+-----------
2008-01-01 05:54:44 | 2008-01-08
```

Das folgende Datum ruft die Ziel-Marketingdaten für das dritte Quartal ab.

```
select username, (firstname ||' '|| lastname) as name,
eventname, caldate, next_day (caldate, 'Monday') as marketing_target
from sales, date, users, event
where sales.buyerid = users.userid
and sales.eventid = event.eventid
and event.dateid = date.dateid
and date.qtr = 3
order by marketing_target, eventname, name;
```

```
username  |     name          |     eventname        |    caldate   |
 marketing_target
----------+-------------------+----------------------+--------------
+------------------
MBO26QSG  |   Callum Atkinson | .38 Special          |  2008-07-06  | 2008-07-07
WCR50YIU  |   Erasmus Alvarez | A Doll's House       |  2008-07-03  | 2008-07-07
CKT70OIE  |   Hadassah Adkins | Ana Gabriel          |  2008-07-06  | 2008-07-07
VVG07OUO  |   Nathan Abbott   | Armando Manzanero    |  2008-07-04  | 2008-07-07
GEW77SII  |   Scarlet Avila   | August: Osage County |  2008-07-06  | 2008-07-07
ECR71CVS  |   Caryn Adkins    | Ben Folds            |  2008-07-03  | 2008-07-07
KUW82CYU  |   Kaden Aguilar   | Bette Midler         |  2008-07-01  | 2008-07-07
WZE78DJZ  |   Kay Avila       | Bette Midler         |  2008-07-01  | 2008-07-07
HXY04NVE  |   Dante Austin    | Britney Spears       |  2008-07-02  | 2008-07-07
URY81YWF  |   Wilma Anthony   | Britney Spears       |  2008-07-02  | 2008-07-07
```

Funktion SYSDATE

SYSDATE gibt das aktuelle Datum und die aktuelle Uhrzeit in der Zeitzone der aktuellen Sitzung (standardmäßig UTC) aus.

> **Note**
>
> SYSDATE gibt das Startdatum und die Uhrzeit für die aktuelle Transaktion aus, nicht für den Start der aktuellen Anweisung.

Syntax

```
SYSDATE
```

Für diese Funktion sind keine Argumente erforderlich.

Rückgabetyp

TIMESTAMP

Beispiele

Das folgende Beispiel verwendet die Funktion SYSDATE zur Rückgabe des vollständigen Zeitstempels für das aktuelle Datum.

```
select sysdate;

timestamp
----------------------------
2008-12-04 16:10:43.976353
```

Das folgende Beispiel verwendet die Funktion SYSDATE innerhalb der Funktion TRUNC zur Rückgabe des aktuellen Datums ohne die Uhrzeit.

```
select trunc(sysdate);

trunc
------------
2008-12-04
```

Die folgende Abfrage gibt Vertriebsinformationen für Daten zurück, die zwischen dem Datum der Ausgabe der Abfrage und dem 120 Tage davor liegenden Datum liegen.

```
select salesid, pricepaid, trunc(saletime) as saletime, trunc(sysdate) as now
from sales
where saletime between trunc(sysdate)-120 and trunc(sysdate)
order by saletime asc;

 salesid | pricepaid |  saletime  |    now
---------+-----------+------------+------------
91535    |    670.00 | 2008-08-07 | 2008-12-05
91635    |    365.00 | 2008-08-07 | 2008-12-05
91901    |   1002.00 | 2008-08-07 | 2008-12-05
...
```

Funktion TIMEOFDAY

TIMEOFDAY ist ein spezielles Alias zur Ausgabe von Wochentag, Datum und Uhrzeit als Zeichenfolgenwert. Es gibt die Tageszeitzeichenfolge für die aktuelle Anweisung zurück, auch wenn es sich innerhalb eines Transaktionsblocks befindet.

Syntax

```
TIMEOFDAY()
```

Rückgabetyp

VARCHAR

Beispiele

Das folgende Beispiel zeugt die Ausgabe des aktuellen Datums und der Uhrzeit mit der Funktion TIMEOFDAY.

```
select timeofday();

timeofday
------------
Thu Sep 19 22:53:50.333525 2013 UTC
```

Funktion TIMESTAMP_CMP

Vergleicht die Werte zweier Zeitstempel und gibt eine Ganzzahl aus. Wenn die Zeitstempel identisch sind, gibt die Funktion `0` zurück. Ist der erste Zeitstempel größer, gibt die Funktion `1` zurück. Ist der zweite Zeitstempel größer, gibt die Funktion `-1` zurück.

Syntax

```
TIMESTAMP_CMP(timestamp1, timestamp2)
```

Argumente

timestamp1

Eine Spalte vom Datentyp `TIMESTAMP` oder ein Ausdruck, der implizit zu einem `TIMESTAMP`-Typ ausgewertet wird.

timestamp2

Eine Spalte vom Datentyp `TIMESTAMP` oder ein Ausdruck, der implizit zu einem `TIMESTAMP`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht Zeitstempel und zeigt die Ergebnisse des Vergleichs.

```
SELECT TIMESTAMP_CMP('2008-01-24 06:43:29', '2008-01-24 06:43:29'),
 TIMESTAMP_CMP('2008-01-24 06:43:29', '2008-02-18 02:36:48'), TIMESTAMP_CMP('2008-02-18
 02:36:48', '2008-01-24 06:43:29');

timestamp_cmp  | timestamp_cmp | timestamp_cmp
---------------+---------------+---------------
             0 |            -1 |             1
```

Das folgende Beispiel vergleicht LISTTIME und SALETIME für eine Auflistung. Der Wert für TIMESTAMP_CMP ist -1 für alle Angebote, da der Zeitstempel für den Verkauf nach dem Zeitstempel für das Angebot liegt.

```
select listing.listid, listing.listtime,
sales.saletime, timestamp_cmp(listing.listtime, sales.saletime)
from listing, sales
where listing.listid=sales.listid
order by 1, 2, 3, 4
limit 10;

 listid |      listtime       |      saletime       | timestamp_cmp
--------+---------------------+---------------------+---------------
      1 | 2008-01-24 06:43:29 | 2008-02-18 02:36:48 |            -1
      4 | 2008-05-24 01:18:37 | 2008-06-06 05:00:16 |            -1
      5 | 2008-05-17 02:29:11 | 2008-06-06 08:26:17 |            -1
      5 | 2008-05-17 02:29:11 | 2008-06-09 08:38:52 |            -1
      6 | 2008-08-15 02:08:13 | 2008-08-31 09:17:02 |            -1
     10 | 2008-06-17 09:44:54 | 2008-06-26 12:56:06 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-10 02:12:36 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-16 11:59:24 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-22 02:23:17 |            -1
     12 | 2008-07-25 01:45:49 | 2008-08-04 03:06:36 |            -1
(10 rows)
```

Dieses Beispiel zeigt, dass TIMESTAMP_CMP für identische Zeitstempel den Wert 0 zurückgibt:

```
select listid, timestamp_cmp(listtime, listtime)
from listing
order by 1 , 2
```

```
limit 10;

 listid | timestamp_cmp
--------+---------------
      1 |             0
      2 |             0
      3 |             0
      4 |             0
      5 |             0
      6 |             0
      7 |             0
      8 |             0
      9 |             0
     10 |             0
(10 rows)
```

Funktion TIMESTAMP_CMP_DATE

TIMESTAMP_CMP_DATE vergleicht den Wert eines Zeitstempels mit einem Datum. Wenn die Werte von Zeitstempel und Datum identisch sind, gibt die Funktion `0` zurück. Ist der Zeitstempel chronologisch größer, gibt die Funktion `1` zurück. Ist das Datum größer, gibt die Funktion `-1` zurück.

Syntax

```
TIMESTAMP_CMP_DATE(timestamp, date)
```

Argumente

timestamp

Eine Spalte vom Datentyp `TIMESTAMP` oder ein Ausdruck, der implizit zu einem `TIMESTAMP`-Typ ausgewertet wird.

date

Eine Spalte vom Datentyp `DATE` oder ein Ausdruck, der implizit zu einem `DATE`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht LISTTIME mit dem Datum `2008-06-18`. Nach diesem Datum erstellte Auflistungen geben 1 aus, davor erstellte Auflistungen `-1`. LISTTIME-Werte sind Zeitstempel.

```
select listid, listtime,
timestamp_cmp_date(listtime, '2008-06-18')
from listing
order by 1, 2, 3
limit 10;


 listid |      listtime       | timestamp_cmp_date
--------+---------------------+--------------------
      1 | 2008-01-24 06:43:29 |                 -1
      2 | 2008-03-05 12:25:29 |                 -1
      3 | 2008-11-01 07:35:33 |                  1
      4 | 2008-05-24 01:18:37 |                 -1
      5 | 2008-05-17 02:29:11 |                 -1
      6 | 2008-08-15 02:08:13 |                  1
      7 | 2008-11-15 09:38:15 |                  1
      8 | 2008-11-09 05:07:30 |                  1
      9 | 2008-09-09 08:03:36 |                  1
     10 | 2008-06-17 09:44:54 |                 -1
(10 rows)
```

Funktion TIMESTAMP_CMP_TIMESTAMPTZ

TIMESTAMP_CMP_TIMESTAMPTZ vergleicht den Wert eines Zeitstempelausdrucks mit einem Zeitstempelausdruck mit Zeitzone. Wenn die Werte des Zeitstempels und des Zeitstempels mit Zeitzone identisch sind, gibt die Funktion `0` zurück. Ist der Zeitstempel chronologisch größer, gibt die Funktion 1 zurück. Ist der Zeitstempel mit Zeitzone größer, gibt die Funktion −1 zurück.

Syntax

```
TIMESTAMP_CMP_TIMESTAMPTZ(timestamp, timestamptz)
```

Argumente

timestamp

Eine Spalte vom Datentyp TIMESTAMP oder ein Ausdruck, der implizit zu einem TIMESTAMP-Typ ausgewertet wird.

timestamptz

Eine Spalte vom Datentyp TIMESTAMPTZ oder ein Ausdruck, der implizit zu einem TIMESTAMPTZ-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht Zeitstempel mit Zeitstempel mit Zeitzonen und zeigt die Ergebnisse des Vergleichs an.

```
SELECT TIMESTAMP_CMP_TIMESTAMPTZ('2008-01-24 06:43:29', '2008-01-24 06:43:29+00'),
 TIMESTAMP_CMP_TIMESTAMPTZ('2008-01-24 06:43:29', '2008-02-18 02:36:48+00'),
 TIMESTAMP_CMP_TIMESTAMPTZ('2008-02-18 02:36:48', '2008-01-24 06:43:29+00');

timestamp_cmp_timestamptz  | timestamp_cmp_timestamptz | timestamp_cmp_timestamptz
---------------------------+---------------------------+--------------------------
             0             |            -1             |             1
```

Funktion TIMESTAMPTZ_CMP

TIMESTAMPTZ_CMP vergleicht den Wert zweier Werte von Zeitstempeln mit Zeitzone und gibt eine Ganzzahl aus. Wenn die Zeitstempel identisch sind, gibt die Funktion 0 zurück. Ist der erste Zeitstempel chronologisch größer, gibt die Funktion 1 zurück. Ist der zweite Zeitstempel größer, gibt die Funktion −1 zurück.

Syntax

```
TIMESTAMPTZ_CMP(timestamptz1, timestamptz2)
```

Argumente

timestamptz1

Eine Spalte vom Datentyp TIMESTAMPTZ oder ein Ausdruck, der implizit zu einem TIMESTAMPTZ-Typ ausgewertet wird.

timestamptz2

Eine Spalte vom Datentyp TIMESTAMPTZ oder ein Ausdruck, der implizit zu einem TIMESTAMPTZ-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht Zeitstempel mit Zeitzonen und zeigt die Ergebnisse des Vergleichs an.

```
SELECT TIMESTAMPTZ_CMP('2008-01-24 06:43:29+00', '2008-01-24 06:43:29+00'),
 TIMESTAMPTZ_CMP('2008-01-24 06:43:29+00', '2008-02-18 02:36:48+00'),
 TIMESTAMPTZ_CMP('2008-02-18 02:36:48+00', '2008-01-24 06:43:29+00');

timestamptz_cmp  | timestamptz_cmp | timestamptz_cmp
-----------------+-----------------+----------------
        0        |       -1        |       1
```

Funktion TIMESTAMPTZ_CMP_DATE

TIMESTAMPTZ_CMP_DATE vergleicht den Wert eines Zeitstempels mit einem Datum. Wenn die Werte von Zeitstempel und Datum identisch sind, gibt die Funktion 0 zurück. Ist der Zeitstempel chronologisch größer, gibt die Funktion 1 zurück. Ist das Datum größer, gibt die Funktion −1 zurück.

Syntax

```
TIMESTAMPTZ_CMP_DATE(timestamptz, date)
```

Argumente

timestamptz

Eine Spalte vom Datentyp `TIMESTAMPTZ` oder ein Ausdruck, der implizit zu einem `TIMESTAMPTZ`-Typ ausgewertet wird.

date

Eine Spalte vom Datentyp `DATE` oder ein Ausdruck, der implizit zu einem `DATE`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Im folgenden Beispiel wird LISTTIME als Zeitstempel mit Zeitzone mit dem Datum `2008-06-18` verglichen. Nach diesem Datum erstellte Auflistungen geben 1 aus, davor erstellte Auflistungen `-1`.

```
select listid, CAST(listtime as timestamptz) as tstz,
timestamp_cmp_date(tstz, '2008-06-18')
from listing
order by 1, 2, 3
limit 10;


 listid |          tstz          | timestamptz_cmp_date
--------+------------------------+----------------------
      1 | 2008-01-24 06:43:29+00 |                   -1
      2 | 2008-03-05 12:25:29+00 |                   -1
      3 | 2008-11-01 07:35:33+00 |                    1
      4 | 2008-05-24 01:18:37+00 |                   -1
      5 | 2008-05-17 02:29:11+00 |                   -1
      6 | 2008-08-15 02:08:13+00 |                    1
      7 | 2008-11-15 09:38:15+00 |                    1
      8 | 2008-11-09 05:07:30+00 |                    1
      9 | 2008-09-09 08:03:36+00 |                    1
     10 | 2008-06-17 09:44:54+00 |                   -1
(10 rows)
```

Funktion TIMESTAMPTZ_CMP_TIMESTAMP

TIMESTAMPTZ_CMP_TIMESTAMP vergleicht den Wert eines Zeitstempelausdrucks mit Zeitzone mit einem Zeitstempelausdruck. Wenn die Werte von Zeitstempel mit Zeitzone und Zeitstempel identisch sind, gibt die Funktion `0` zurück. Ist der Zeitstempel mit Zeitzone chronologisch größer, gibt die Funktion 1 zurück. Ist der Zeitstempel größer, gibt die Funktion −1 zurück.

Syntax

```
TIMESTAMPTZ_CMP_TIMESTAMP(timestamptz, timestamp)
```

Argumente

timestamptz

Eine Spalte vom Datentyp `TIMESTAMPTZ` oder ein Ausdruck, der implizit zu einem `TIMESTAMPTZ`-Typ ausgewertet wird.

timestamp

Eine Spalte vom Datentyp `TIMESTAMP` oder ein Ausdruck, der implizit zu einem `TIMESTAMP`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Beispiele

Das folgende Beispiel vergleicht Zeitstempel mit Zeitzonen mit Zeitstempel und zeigt die Ergebnisse des Vergleichs an.

```
SELECT TIMESTAMPTZ_CMP_TIMESTAMP('2008-01-24 06:43:29+00', '2008-01-24 06:43:29'),
 TIMESTAMPTZ_CMP_TIMESTAMP('2008-01-24 06:43:29+00', '2008-02-18 02:36:48'),
 TIMESTAMPTZ_CMP_TIMESTAMP('2008-02-18 02:36:48+00', '2008-01-24 06:43:29');

timestamptz_cmp_timestamp  | timestamptz_cmp_timestamp | timestamptz_cmp_timestamp
---------------------------+---------------------------+--------------------------
             0             |            -1             |             1
```

Funktion TIMEZONE

TIMEZONE gibt einen Zeitstempel für den angegebenen Wert eines Zeitstempels mit Zeitzone aus.

Informationen und Beispiele zum Festlegen der Zeitzone finden Sie unter Zeitzone.

Informationen und Beispiele zum Konvertieren der Zeitzone finden Sie unter CONVERT_TIMEZONE.

Syntax

```
TIMEZONE('timezone', { timestamp | timestamptz })
```

Argumente

Zeitzone

Die Zeitzone für den Rückgabewert. Die Zeitzone kann als Zeitzonenname (beispielsweise **'Africa/Kampala'** oder **'Singapore'**) oder als Zeitzonenabkürzung (beispielsweise **'UTC'** oder **'PDT'**) angegeben werden. Führen Sie den folgenden Befehl aus, um eine Liste der unterstützten Zeitzonennamen anzuzeigen.

```
select pg_timezone_names();
```

Führen Sie den folgenden Befehl aus, um eine Liste der unterstützten Zeitzonenabkürzungen anzuzeigen.

```
select pg_timezone_abbrevs();
```

Weitere Informationen und Beispiele finden Sie unter Nutzungshinweise zu Zeitstempeln.

timestamp | timestamptz

Ein Ausdruck, der in einem TIMESTAMP- oder TIMESTAMPTZ-Typ oder einem Wert resultiert, der implizit zu einem Zeitstempel oder einem Zeitstempel mit Zeitzone gezwungen werden kann.

Rückgabetyp

TIMESTAMPTZ bei Verwendung mit einem TIMESTAMP-Ausdruck.

TIMESTAMP bei Verwendung mit einem TIMESTAMPTZ-Ausdruck.

Beispiele

Im Folgenden wird ein Zeitstempel für die UTC-Zeitzone unter Verwendung des Zeitstempels `2008-06-17 09:44:54` der PST-Zeitzone zurückgegeben.

```
SELECT TIMEZONE('PST', '2008-06-17 09:44:54');

timezone
----------------------
2008-06-17 17:44:54+00
```

Im Folgenden wird ein Zeitstempel für die PST-Zeitzone unter Verwendung des Zeitstempels der UTC-Zeitzone `2008-06-17 09:44:54+00` zurückgegeben.

```
SELECT TIMEZONE('PST', timestamptz('2008-06-17 09:44:54+00'));

timezone
----------------------
2008-06-17 01:44:54
```

Funktion TO_TIMESTAMP

TO_TIMESTAMP konvertiert eine TIMESTAMP-Zeichenfolge zu TIMESTAMPTZ. Eine Liste mit zusätzlichen Datums- und Uhrzeitfunktionen für Amazon Redshift finden Sie unter Datums- und Zeitfunktionen.

Syntax

```
to_timestamp(timestamp, format)
```

```
to_timestamp (timestamp, format, is_strict)
```

Argumente

timestamp

Eine Zeichenfolge, die für einen Zeitstempelwert in dem von format angegebenen Format steht. Wenn dieses Argument leer gelassen wird, wird der Zeitstempelwert standardmäßig auf `0001-01-01 00:00:00` gesetzt.

format

Ein Zeichenfolgeliteral, das das Format des timestamp-Werts definiert. Formate, die eine Zeitzone (**TZ**, **tz** oder **OF**) enthalten, werden als Eingabe nicht unterstützt. Für gültige Zeitstempelformate vgl. Datum-/Uhrzeit-Formatzeichenfolgen.

is_strict

Ein optionaler boolescher Wert, der angibt, ob ein Fehler zurückgegeben wird, wenn ein Eingabezeitstempelwert außerhalb des zulässigen Bereichs liegt. Wenn is_strict auf TRUE gesetzt wird, wird ein Fehler zurückgegeben, wenn ein Wert außerhalb des zulässigen Bereichs liegt. Wenn is_strict auf FALSE gesetzt wird, was die Standardeinstellung ist, sind Überlaufwerte zulässig.

Rückgabetyp

TIMESTAMPTZ

Beispiele

Das folgende Beispiel zeigt die Verwendung der Funktion TO_TIMESTAMP zur Konvertierung einer TIMESTAMP-Zeichenfolge in einen TIMESTAMPTZ.

```
select sysdate, to_timestamp(sysdate, 'YYYY-MM-DD HH24:MI:SS') as second;

timestamp                  | second
---------------------------   ---------------------
2021-04-05 19:27:53.281812 | 2021-04-05 19:27:53+00
```

Es ist möglich, den TO_TIMESTAMP-Teil eines Datums zu übergeben. Die übrigen Datumsteile werden auf die Standardwerte gesetzt. Die Uhrzeit ist in der Ausgabe enthalten:

```
SELECT TO_TIMESTAMP('2017','YYYY');

to_timestamp
---------------------------
2017-01-01 00:00:00+00
```

Die folgende SQL-Anweisung konvertiert die Zeichenfolge '2011-12-18 24:38:15' in einen TIMESTAMPTZ-Wert. Das Ergebnis ist ein TIMESTAMPTZ-Wert, der auf den nächsten Tag fällt, da die Anzahl der Stunden 24 übersteigt:

```
SELECT TO_TIMESTAMP('2011-12-18 24:38:15', 'YYYY-MM-DD HH24:MI:SS');

to_timestamp
----------------------
2011-12-19 00:38:15+00
```

Die folgende SQL-Anweisung konvertiert die Zeichenfolge '2011-12-18 24:38:15' in einen TIMESTAMPTZ-Wert. Das Ergebnis ist ein Fehler, da der Zeitwert im Zeitstempel 24 Stunden übersteigt:

```
SELECT TO_TIMESTAMP('2011-12-18 24:38:15', 'YYYY-MM-DD HH24:MI:SS', TRUE);

ERROR:  date/time field time value out of range: 24:38:15.0
```

Die Funktion TRUNC

Kürzt einen `TIMESTAMP` und gibt ein `DATE` zurück.

Diese Funktion kann auch eine Zahl kürzen. Weitere Informationen finden Sie unter Die Funktion TRUNC.

Syntax

```
TRUNC(timestamp)
```

Argumente

timestamp

Eine Spalte vom Datentyp `TIMESTAMP` oder ein Ausdruck, der implizit zu einem `TIMESTAMP`-Typ ausgewertet wird.

Wenn ein Zeitstempelwert mit `00:00:00` als Uhrzeit zurückgegeben werden soll, wandeln Sie das Funktionsergebnis in einen `TIMESTAMP` um.

Rückgabetyp

DATUM

Beispiele

Im folgenden Beispiel wird der Datumsabschnitt aus dem Ergebnis der SYSDATE-Funktion zurückgegeben (die einen Zeitstempel zurückgibt).

```
SELECT SYSDATE;

+----------------------------+
|         timestamp          |
+----------------------------+
| 2011-07-21 10:32:38.248109 |
+----------------------------+

SELECT TRUNC(SYSDATE);

+------------+
|   trunc    |
+------------+
| 2011-07-21 |
+------------+
```

Im folgenden Beispiel wird die TRUNC-Funktion auf eine `TIMESTAMP`-Spalte angewendet. Der Rückgabetyp ist ein Datum.

```
SELECT TRUNC(starttime) FROM event
ORDER BY eventid LIMIT 1;

+------------+
|   trunc    |
+------------+
| 2008-01-25 |
+------------+
```

Das folgende Beispiel gibt einen Zeitstempelwert mit `00:00:00` als Uhrzeit zurück, indem das Ergebnis der TRUNC-Funktion in einen `TIMESTAMP` umgewandelt wird.

```
SELECT CAST((TRUNC(SYSDATE)) AS TIMESTAMP);

+---------------------+
|        trunc        |
+---------------------+
| 2011-07-21 00:00:00 |
```

```
+---------------------+
```

Datumsteile für Datums- oder Zeitstempelfunktionen

Die folgende Tabelle identifiziert die Namen und Abkürzungen von Datumsteilen und Uhrzeitteilen, die als Argumente für die folgenden Funktionen verwendet werden können:

- DATEADD
- DATEDIFF
- DATE_PART
- EXTRACT

Datumsteil oder Uhrzeitteil	Abkürzungen
millennium, millennia	mil, mils
century, centuries	c, cent, cents
decade, decades	dec, decs
Epoche	epoch (unterstützt von EXTRACT)
year, years	y, yr, yrs
quarter, quarters	qtr, qtrs
month, months	mon, mons
week, weeks	w
Tag der Woche	dayofweek, dow, dw, weekday (unterstützt von DATE_PART und Funktion EXTRACT) Gibt eine Ganzzahl von 0–6 aus, beginnend mit Sonntag. **Note** Der Datumsteil DOW verhält sich anders als der Datumsteil „Wochentag (D)“ für Datumsteilformatze

Datumsteil oder Uhrzeitteil	Abkürzungen
	ichenfolgen. D basiert auf den Ganzzahlen 1–7, wobei die 1 für den Sonntag steht. Weitere Informationen finden Sie unter Datum-/Uhrzeit-Formatzeichenfolgen.
Tag des Jahres	dayofyear, doy, dy, yearday (unterstützt von EXTRACT)
day, days	d
hour, hours	h, hr, hrs
minute, minutes	m, min, mins
second, seconds	s, sec, secs
millisecond, milliseconds	ms, msec, msecs, msecond, mseconds, millisec, millisecs, millisecon
microsecond, microseconds	microsec, microsecs, microsecond, usecond, useconds, us, usec, usecs
timezone, timezone_hour, timezone_minute	Unterstützt von EXTRACT nur für Zeitstempel mit Zeitzone (TIMESTAMPTZ).

Abweichungen bei den Ergebnissen mit Sekunden, Millisekunden und Mikrosekunden

Kleinere Differenzen treten auf, wenn verschiedene Datumsfunktionen Sekunden, Millisekunden oder Mikrosekunden als Datumsteile angeben:

- Die Funktion EXTRACT gibt nur für den angegebenen Datumsteilen Ganzzahlen aus, wobei Datumsteile auf höheren und niedrigeren Ebenen ignoriert werden. Wenn der angegebene Datumsteil „Sekunden“ ist, werden Millisekunden und Mikrosekunden in dem Ergebnis nicht berücksichtigt. Wenn der angegebene Datumsteil „Millisekunden“ ist, werden Sekunden und Mikrosekunden in dem Ergebnis nicht berücksichtigt. Wenn der angegebene Datumsteil „Mikrosekunden“ ist, werden Sekunden und Millisekunden in dem Ergebnis nicht berücksichtigt.

- Die Funktion DATE_PART gibt den vollständigen Sekundenteil des Zeitstempels aus, unabhängig davon, welcher Datumsteil angegeben wurde; dabei wird je nach Bedarf entweder eine Dezimal- oder eine Ganzzahl ausgegeben.

Vergleichen Sie beispielsweise die Ergebnisse der folgenden Abfragen:

```
create table seconds(micro timestamp);

insert into seconds values('2009-09-21 11:10:03.189717');

select extract(sec from micro) from seconds;

date_part
-----------
3

select date_part(sec, micro) from seconds;

pgdate_part
-------------
3.189717
```

Anmerkungen zu CENTURY, EPOCH, DECADE und MIL

CENTURY oder CENTURIES

Amazon Redshift lässt ein CENTURY mit dem Jahr ###1 beginnen und mit dem Jahr ###0 enden:

```
select extract (century from timestamp '2000-12-16 12:21:13');
date_part
-----------
20

select extract (century from timestamp '2001-12-16 12:21:13');
date_part
-----------
21
```

EPOCH

Die Amazon-Redshift-Implementierung von EPOCH ist relativ zu 1970-01-01 00:00:00.000000, unabhängig von der Zeitzone, in der sich das Cluster befindet. Möglicherweise müssen Sie die Ergebnisse um die Differenz in Stunden verschieben, je nach der Zeitzone, in der sich das Cluster befindet.

Das folgende Beispiel veranschaulicht die folgenden Schritte:

1. Erstellt eine Tabelle mit der Bezeichnung EVENT_EXAMPLE auf der Grundlage der Tabelle EVENT. Dieser CREATE AS-Befehl verwendet die Funktion DATE_PART zur Erstellung einer Datumsspalte (mit der standardmäßigen Bezeichnung PGDATE_PART) zur Speicherung des Epochenwerts für jedes Ereignis.
2. Wählt die Spalte und den Datentyp von EVENT_EXAMPLE aus PG_TABLE_DEF aus.
3. Wählt EVENTNAME, STARTTIME und PGDATE_PART aus der Tabelle EVENT_EXAMPLE aus, um die verschiedenen Datums- und Uhrzeitformate anzuzeigen.
4. Wählt EVENTNAME und STARTTIME aus EVENT EXAMPLE aus. Konvertiert Epochenwerte in PGDATE_PART unter Verwendung eines Intervall von 1 Sekunde zu einem Zeitstempel ohne Zeitzone und gibt die Ergebnisse in einer Spalte mit der Bezeichnung CONVERTED_TIMESTAMP aus.

```
create table event_example
as select eventname, starttime, date_part(epoch, starttime) from event;

select "column", type from pg_table_def where tablename='event_example';

     column    |            type
---------------+-----------------------------
 eventname     | character varying(200)
 starttime     | timestamp without time zone
 pgdate_part   | double precision
(3 rows)
```

```
select eventname, starttime, pgdate_part from event_example;

   eventname          |      starttime      | pgdate_part
----------------------+---------------------+-------------
 Mamma Mia!           | 2008-01-01 20:00:00 |  1199217600
 Spring Awakening     | 2008-01-01 15:00:00 |  1199199600
 Nas                  | 2008-01-01 14:30:00 |  1199197800
 Hannah Montana       | 2008-01-01 19:30:00 |  1199215800
```

```
 K.D. Lang            | 2008-01-01 15:00:00 |  1199199600
 Spamalot             | 2008-01-02 20:00:00 |  1199304000
 Macbeth              | 2008-01-02 15:00:00 |  1199286000
 The Cherry Orchard   | 2008-01-02 14:30:00 |  1199284200
 Macbeth              | 2008-01-02 19:30:00 |  1199302200
 Demi Lovato          | 2008-01-02 19:30:00 |  1199302200


select eventname,
starttime,
timestamp with time zone 'epoch' + pgdate_part * interval '1 second' AS
 converted_timestamp
from event_example;

       eventname      |      starttime      | converted_timestamp
----------------------+---------------------+---------------------
 Mamma Mia!           | 2008-01-01 20:00:00 | 2008-01-01 20:00:00
 Spring Awakening     | 2008-01-01 15:00:00 | 2008-01-01 15:00:00
 Nas                  | 2008-01-01 14:30:00 | 2008-01-01 14:30:00
 Hannah Montana       | 2008-01-01 19:30:00 | 2008-01-01 19:30:00
 K.D. Lang            | 2008-01-01 15:00:00 | 2008-01-01 15:00:00
 Spamalot             | 2008-01-02 20:00:00 | 2008-01-02 20:00:00
 Macbeth              | 2008-01-02 15:00:00 | 2008-01-02 15:00:00
 The Cherry Orchard   | 2008-01-02 14:30:00 | 2008-01-02 14:30:00
 Macbeth              | 2008-01-02 19:30:00 | 2008-01-02 19:30:00
 Demi Lovato          | 2008-01-02 19:30:00 | 2008-01-02 19:30:00
 ...
```

DECADE oder DECADES

Amazon Redshift interpretiert DECADE oder DECADES DATEPART auf der Grundlage des gewöhnlichen Kalenders. Zum Beispiel: Da der gewöhnliche Kalender mit dem Jahr 1 beginnt, ist die erste Dekade (Dekade 1) 0001-01-01 bis 0009-12-31, und die zweite Dekade (Dekade 2) ist 0010-01-01 bis 0019-12-31. Beispielsweise reicht Dekade 201 von 2000-01-01 bis 2009-12-31:

```
select extract(decade from timestamp '1999-02-16 20:38:40');
date_part
-----------
200

select extract(decade from timestamp '2000-02-16 20:38:40');
date_part
-----------
```

```
201

select extract(decade from timestamp '2010-02-16 20:38:40');
date_part
-----------
202
```

MIL oder MILS

Amazon Redshift interpretiert ein MIL mit dem Beginn am ersten Tag des Jahres #001 und dem Ende am letzten Tag des Jahres #000:

```
select extract (mil from timestamp '2000-12-16 12:21:13');
date_part
-----------
2

select extract (mil from timestamp '2001-12-16 12:21:13');
date_part
-----------
3
```

Hash-Funktionen

Themen

- Die Funktion CHECKSUM
- Funktion farmFingerPrint64
- Funktion FUNC_SHA1
- FNV_HASH-Funktion
- Die Funktion MD5
- Die Funktion SHA
- Die Funktion SHA1
- Die Funktion SHA2
- MURMUR3_32_HASH

Eine Hash-Funktion ist eine mathematische Funktion, mit der ein numerischer Eingabewert in einen anderen Wert umgewandelt wird.

Die Funktion CHECKSUM

Berechnet einen Prüfsummenwert zur Erstellung eines Hash-Index.

Syntax

```
CHECKSUM(expression)
```

Argument

expression

Der Eingabeausdruck muss den Datentyp VARCHAR, INTEGER oder DECIMAL haben.

Rückgabetyp

Die CHECKSUM-Funktion gibt eine Ganzzahl zurück.

Beispiel

Im folgenden Beispiel wird ein Prüfsummenwert für die Spalte COMMISSION berechnet:

```
select checksum(commission)
from sales
order by salesid
limit 10;

checksum
----------
10920
1140
5250
2625
2310
5910
11820
2955
8865
975
(10 rows)
```

Funktion farmFingerPrint64

Berechnet den Farmhash-Wert des Eingabearguments unter Verwendung der Funktion `Fingerprint64`.

Syntax

```
farmFingerprint64(expression)
```

Argument

expression

Der Eingabeausdruck muss den Datentyp `VARCHAR` oder `VARBYTE` aufweisen.

Rückgabetyp

Die Funktion `farmFingerprint64` gibt einen `BIGINT`-Wert zurück.

Beispiel

Im folgenden Beispiel wird der `farmFingerprint64`-Wert von `Amazon Redshift` zurückgegeben, der als Datentyp `VARCHAR` eingegeben wird.

```
SELECT farmFingerprint64('Amazon Redshift');
```

```
  farmfingerprint64
---------------------
 8085098817162212970
```

Im folgenden Beispiel wird der `farmFingerprint64`-Wert von `Amazon Redshift` zurückgegeben, der als Datentyp `VARBYTE` eingegeben wird.

```
SELECT farmFingerprint64('Amazon Redshift'::varbyte);
```

```
  farmfingerprint64
---------------------
```

```
8085098817162212970
```

Funktion FUNC_SHA1

Synonym der SHA1-Funktion.

Siehe Die Funktion SHA1.

FNV_HASH-Funktion

Berechnet die nicht-kryptographische 64-Bit-Hash-Funktion FNV-1a für alle grundlegenden Datentypen.

Syntax

```
FNV_HASH(value [, seed])
```

Argumente

Wert

Der Eingabewert, der gehasht werden soll. Amazon Redshift verwendet die binäre Darstellung des Wertes, um den Eingabewert zu hashen. Beispielsweise werden INTEGER-Werte mit 4 Bytes gehasht und BIGINT-Werte mit 8 Bytes. Außerdem werden beim Hashing von CHAR- und VARCHAR-Eingaben keine nachstehenden Leerzeichen ignoriert.

Seed

Der BIGINT-Seed der Hash-Funktion ist optional. Wenn er nicht angegeben wird, verwendet Amazon Redshift den Standard-FNV-Seed. Dies ermöglicht eine Kombination des Hashs mehrerer Spalten ohne Konvertierungen oder Verkettungen.

Rückgabetyp

BIGINT

Beispiel

Die folgenden Beispiele geben den FNV-Hash einer Zahl, der Zeichenfolge „Amazon Redshift“ und die Verkettung beider zurück.

```
select fnv_hash(1);
```

```
        fnv_hash
----------------------
 -5968735742475085980
(1 row)
```

```
select fnv_hash('Amazon Redshift');
      fnv_hash
--------------------
 7783490368944507294
(1 row)
```

```
select fnv_hash('Amazon Redshift', fnv_hash(1));
       fnv_hash
----------------------
 -2202602717770968555
(1 row)
```

Nutzungshinweise

- Um den Hash einer Tabelle mit mehreren Spalten zu berechnen, können Sie den FNV-Hash der ersten Spalte berechnen und ihn als Seed an den Hash der zweiten Spalte übergeben. Dann wird der FNV-Hash der zweiten Spalte als Seed an den Hash der dritten Spalte übergeben.

  Im folgenden Beispiel werden Seeds erstellt, um eine Tabelle mit mehreren Spalten zu hashen.

  ```
  select fnv_hash(column_3, fnv_hash(column_2, fnv_hash(column_1))) from sample_table;
  ```

- Mit derselben Eigenschaft kann der Hash einer Verkettung von Zeichenfolgen berechnet werden.

  ```
  select fnv_hash('abcd');
         fnv_hash
  ----------------------
   -281581062704388899
  (1 row)
  ```

  ```
  select fnv_hash('cd', fnv_hash('ab'));
        fnv_hash
  ----------------------
   -281581062704388899
  (1 row)
  ```

- Die Hash-Funktion verwendet den Typ der Eingabe, um die Anzahl der zu hashenden Bytes zu bestimmen. Verwenden Sie Übertragen, um einen bestimmten Typ zu erzwingen, falls erforderlich.

  In den folgenden Beispielen werden verschiedene Eingabetypen verwendet, um unterschiedliche Ergebnisse zu erzielen.

```
select fnv_hash(1::smallint);
      fnv_hash
--------------------
 589727492704079044
(1 row)
```

```
select fnv_hash(1);
       fnv_hash
----------------------
 -5968735742475085980
(1 row)
```

```
select fnv_hash(1::bigint);
       fnv_hash
----------------------
 -8517097267634966620
(1 row)
```

Die Funktion MD5

Verwendet die kryptografische Hash-Funktion MD5, um eine Zeichenfolge mit variabler Länge in eine Zeichenfolge mit 32 Zeichen zu konvertieren, die eine Textdarstellung des hexadezimalen Werts einer 128-Bit-Prüfsumme ist.

Syntax

```
MD5(string)
```

Argumente

string

Eine Zeichenfolge mit variabler Länge.

Rückgabetyp

Die MD5-Funktion gibt eine Zeichenfolge mit 32 Zeichen zurück, die eine Textdarstellung des hexadezimalen Werts einer 128-Bit-Prüfsumme ist.

Beispiele

Im folgenden Beispiel wird der 128-Bit-Wert für die Zeichenfolge „Amazon Redshift“ gezeigt:

```
select md5('Amazon Redshift');
md5
----------------------------------
f7415e33f972c03abd4f3fed36748f7a
(1 row)
```

Die Funktion SHA

Synonym der SHA1-Funktion.

Siehe Die Funktion SHA1.

Die Funktion SHA1

Die SHA1-Funktion verwendet die kryptografische Hash-Funktion SHA1, um eine Zeichenfolge mit variabler Länge in eine Zeichenfolge mit 40 Zeichen zu konvertieren, die eine Textdarstellung des hexadezimalen Werts einer 160-Bit-Prüfsumme ist.

Syntax

SHA1 ist synonym mit Die Funktion SHA und Funktion FUNC_SHA1.

```
SHA1(string)
```

Argumente

string

Eine Zeichenfolge mit variabler Länge.

Rückgabetyp

Die SHA1-Funktion gibt eine Zeichenfolge mit 40 Zeichen zurück, die eine Textdarstellung des hexadezimalen Werts einer 160-Bit-Prüfsumme ist.

Beispiel

Im folgenden Beispiel wird der 160-Bit-Wert für das Wort „Amazon Redshift“ zurückgegeben:

```
select sha1('Amazon Redshift');
```

Die Funktion SHA2

Die SHA2-Funktion verwendet die kryptografische Hast-Funktion SHA2, um eine Zeichenfolge mit variabler Länge in eine Zeichenkette zu konvertieren. Die Zeichenkette ist eine Textdarstellung des hexadezimalen Wertes der Prüfsumme mit der angegebenen Anzahl von Bits.

Syntax

```
SHA2(string, bits)
```

Argumente

string

Eine Zeichenfolge mit variabler Länge.

integer

Die Anzahl der Bits in den Hash-Funktionen. Gültige Werte sind 0 (identisch mit 256), 224, 256, 384 und 512.

Rückgabetyp

Die SHA2-Funktion gibt eine Zeichenkette zurück, die eine Textdarstellung des Hexadezimalwerts der Prüfsumme oder eine leere Zeichenfolge ist, wenn die Anzahl der Bits ungültig ist.

Beispiel

Im folgenden Beispiel wird der 256-Bit-Wert für das Wort „Amazon Redshift“ zurückgegeben:

```
select sha2('Amazon Redshift', 256);
```

MURMUR3_32_HASH

Die Funktion MURMUR3_32_HASH berechnet den nicht kryptografischen 32-Bit-Murmur3A-Hash für alle gängigen Datentypen, einschließlich numerischer Datentypen und Zeichenfolgentypen.

Syntax

```
MURMUR3_32_HASH(value [, seed])
```

Argumente

Wert

Der Eingabewert, der gehasht werden soll. Amazon Redshift hasht die binäre Darstellung des Eingabewerts. Dieses Verhalten ähnelt FNV_HASH-Funktion, der Wert wird jedoch in die binäre Darstellung umgewandelt, die in der 32-Bit-Murmur3-Hash-Spezifikation von Apache Iceberg angegeben ist.

Seed

Der INT-Seed der Hash-Funktion. Dieses Argument ist optional. Wenn er nicht angegeben wird, verwendet Amazon Redshift den Standard-Seed 0. Dies ermöglicht eine Kombination des Hashs mehrerer Spalten ohne Konvertierungen oder Verkettungen.

Rückgabetyp

Die Funktion gibt einen INT-Wert zurück.

Beispiel

Die folgenden Beispiele geben den Murmur3-Hash einer Zahl, die Zeichenfolge „Amazon Redshift“ und die Verkettung beider zurück.

```
select MURMUR3_32_HASH(1);

    MURMUR3_32_HASH
----------------------
```

```
 -5968735742475085980
(1 row)
```

```
select MURMUR3_32_HASH('Amazon Redshift');

    MURMUR3_32_HASH
----------------------
 7783490368944507294
(1 row)
```

```
select MURMUR3_32_HASH('Amazon Redshift', MURMUR3_32_HASH(1));

    MURMUR3_32_HASH
----------------------
 -2202602717770968555
(1 row)
```

Nutzungshinweise

Um den Hash einer Tabelle mit mehreren Spalten zu berechnen, können Sie den Murmur3-Hash der ersten Spalte berechnen und ihn als Seed an den Hash der zweiten Spalte übergeben. Dann wird der Murmur3-Hash der zweiten Spalte als Seed an den Hash der dritten Spalte übergeben.

Im folgenden Beispiel werden Seeds erstellt, um eine Tabelle mit mehreren Spalten zu hashen.

```
select MURMUR3_32_HASH(column_3, MURMUR3_32_HASH(column_2, MURMUR3_32_HASH(column_1)))
 from sample_table;
```

Mit derselben Eigenschaft kann der Hash einer Verkettung von Zeichenfolgen berechnet werden.

```
select MURMUR3_32_HASH('abcd');

   MURMUR3_32_HASH
----------------------
 -281581062704388899
(1 row)
```

```
select MURMUR3_32_HASH('cd', MURMUR3_32_HASH('ab'));
```

```
   MURMUR3_32_HASH
---------------------
 -281581062704388899
(1 row)
```

Die Hash-Funktion verwendet den Typ der Eingabe, um die Anzahl der zu hashenden Bytes zu bestimmen. Verwenden Sie Übertragen, um einen bestimmten Typ zu erzwingen, falls erforderlich.

In den folgenden Beispielen werden verschiedene Eingabetypen verwendet, um unterschiedliche Ergebnisse zu erzielen.

```
select MURMUR3_32_HASH(1::smallint);

   MURMUR3_32_HASH
--------------------
 589727492704079044
(1 row)
```

```
select MURMUR3_32_HASH(1);

   MURMUR3_32_HASH
----------------------
 -5968735742475085980
(1 row)
```

```
select MURMUR3_32_HASH(1::bigint);

   MURMUR3_32_HASH
----------------------
 -8517097267634966620
(1 row)
```

HyperLogLog Funktionen

Im Folgenden finden Sie Beschreibungen der HyperLogLog Funktionen für SQL, die Amazon Redshift unterstützt.

Themen

- Funktion HLL

- Funktion HLL_CREATE_SKETCH
- Funktion HLL_CARDINALITY
- Funktion HLL_COMBINE
- HLL_COMBINE_SKETCHES-Funktion

Funktion HLL

Die HLL-Funktion gibt die HyperLogLog Kardinalität der Werte der Eingabeausdrücke zurück. Die HLL-Funktion funktioniert mit allen Datentypen mit Ausnahme des Datentyps HLLSKETCH. Die HLL-Funktion ignoriert NULL-Werte. Wenn keine Zeilen in einer Tabelle vorhanden sind oder alle Zeilen NULL sind, ist die resultierende Kardinalität 0.

Syntax

```
HLL (aggregate_expression)
```

Argument

aggregate_expression

Jeder gültige Ausdruck, der den Wert bereitstellt, der aggregiert werden soll, zum Beispiel einen Spaltennamen. Diese Funktion unterstützt jeden Datentyp als Eingabe mit Ausnahme von HLLSKETCH, GEOMETRY, GEOGRAPHY und VARBYTE.

Rückgabetyp

Die HLL-Funktion gibt einen BIGINT oder INT8-Wert zurück.

Beispiele

Im folgenden Beispiel wird die Kardinalität der Spalte `an_int` in der Tabelle `a_table` zurückgegeben.

```
CREATE TABLE a_table(an_int INT);
INSERT INTO a_table VALUES (1), (2), (3), (4);

SELECT hll(an_int) AS cardinality FROM a_table;
cardinality
```

```
-------------
4
```

Funktion HLL_CREATE_SKETCH

Die HLL_CREATE_SKETCH-Funktion gibt einen HLLSKETCH-Datentyp zurück, der die Eingabeausdruckswerte kapselt. Die HLL_CREATE_SKETCH-Funktion funktioniert mit jedem Datentyp und ignoriert NULL-Werte. Wenn keine Zeilen in einer Tabelle vorhanden sind oder alle Zeilen NULL sind, enthält die resultierende Skizze keine Index-Wert-Paare wie zum Beispiel `{"version":1,"logm":15,"sparse":{"indices":[],"values":[]}}`.

Syntax

```
HLL_CREATE_SKETCH (aggregate_expression)
```

Argument

aggregate_expression

Jeder gültige Ausdruck, der den Wert bereitstellt, der aggregiert werden soll, zum Beispiel einen Spaltennamen. NULL-Werte werden ignoriert. Diese Funktion unterstützt jeden Datentyp als Eingabe mit Ausnahme von HLLSKETCH, GEOMETRY, GEOGRAPHY und VARBYTE.

Rückgabetyp

Die HLL_CREATE_SKETCH-Funktion gibt einen HLLSKETCH-Wert zurück.

Beispiele

Im folgenden Beispiel wird der Typ HLLSKETCH für die Spalte `an_int` in der Tabelle `a_table` zurückgegeben. Ein JSON-Objekt wird verwendet, um beim Importieren, Exportieren oder Drucken von HyperLogLog Skizzen eine dünne Skizze darzustellen. Eine Zeichenkettendarstellung (im Base64-Format) wird verwendet, um eine dichte Skizze darzustellen. HyperLogLog

```
CREATE TABLE a_table(an_int INT);
INSERT INTO a_table VALUES (1), (2), (3), (4);

SELECT hll_create_sketch(an_int) AS sketch FROM a_table;
sketch
```

```
----------------------------------------------------------------------------------------------------
{"version":1,"logm":15,"sparse":{"indices":
[20812342,20850007,22362299,47158030],"values":[1,2,1,1]}}
(1 row)
```

Funktion HLL_CARDINALITY

Die HLL_CARDINALITY-Funktion gibt die Kardinalität des Eingabe-HLLSKETCH-Datentyps zurück.

Syntax

```
HLL_CARDINALITY (hllsketch_expression)
```

Argument

hllsketch_expression

Jeder gültige Ausdruck, der einen HLLSKETCH-Typ auswertet, zum Beispiel ein Spaltenname. Der Eingabewert ist der HLLSKETCH-Datentyp.

Rückgabetyp

Die HLL_CARDINALITY-Funktion gibt einen BIGINT oder INT8 Wert zurück.

Beispiele

Im folgenden Beispiel wird die Kardinalität der Spalte `sketch` in der Tabelle `hll_table` zurückgegeben.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

SELECT hll_cardinality(sketch) AS cardinality FROM hll_table;
cardinality
-------------
6
```

```
4
(2 rows)
```

Funktion HLL_COMBINE

Die HLL_COMBINE-Aggregationsfunktion gibt einen HLLSKETCH-Datentyp zurück, der alle Eingabe-HLLSKETCH-Werte kombiniert.

Die Kombination von zwei oder mehr HyperLogLog Skizzen ist ein neuer HLLSKETCH, der Informationen über die Vereinigung der unterschiedlichen Werte enthält, die jede Eingabeskizze darstellt. Nach dem Kombinieren von Skizzen extrahiert Amazon Redshift die Kardinalität der Vereinigung von zwei oder mehr Datensätzen. Weitere Informationen zum Kombinieren mehrerer Skizzen finden Sie unter Beispiel: Rückgabe einer HyperLogLog Skizze aus der Kombination mehrerer Skizzen.

Syntax

```
HLL_COMBINE (hllsketch_expression)
```

Argument

hllsketch_expression

Jeder gültige Ausdruck, der einen HLLSKETCH-Typ auswertet, zum Beispiel ein Spaltenname. Der Eingabewert ist der HLLSKETCH-Datentyp.

Rückgabetyp

Die HLL_COMBINE-Funktion gibt einen HLLSKETCH-Typ zurück.

Beispiele

Im folgenden Beispiel werden die kombinierten HLLSKETCH-Werte in der Tabelle `hll_table` zurückgegeben.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);
```

```
CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

SELECT hll_combine(sketch) AS sketches FROM hll_table;
sketches
--------------------------------------------------------------------------------------------------------
{"version":1,"logm":15,"sparse":{"indices":
[20812342,20850007,22362299,40314817,42650774,47158030],"values":[1,2,1,3,2,1]}}
(1 row)
```

HLL_COMBINE_SKETCHES-Funktion

Die HLL_COMBINE_SKETCHES ist eine skalare Funktion, die zwei HLLLSKETCH-Werte als Eingabe verwendet und zu einem einzigen HLSKETCH kombiniert.

Die Kombination von zwei oder mehr HyperLogLog Skizzen ist ein neuer HLLSKETCH, der Informationen über die Vereinigung der unterschiedlichen Werte enthält, die jede Eingabeskizze darstellt.

Syntax

```
HLL_COMBINE_SKETCHES (hllsketch_expression1, hllsketch_expression2)
```

Argument

hllsketch_expression1 und hllsketch_expression2

Jeder gültige Ausdruck, der einen HLLSKETCH-Typ auswertet, zum Beispiel ein Spaltenname.

Rückgabetyp

Die HLL_COMBINE_SKETCHES-Funktion gibt einen HLLSKETCH-Typ zurück.

Beispiele

Im folgenden Beispiel werden die kombinierten HLLSKETCH-Werte in der Tabelle zurückgegeben `hll_table`.

```
WITH tbl1(x, y)
     AS (SELECT Hll_create_sketch(1),
```

```
                Hll_create_sketch(2)
        UNION ALL
        SELECT Hll_create_sketch(3),
                Hll_create_sketch(4)
        UNION ALL
        SELECT Hll_create_sketch(5),
                Hll_create_sketch(6)
        UNION ALL
        SELECT Hll_create_sketch(7),
                Hll_create_sketch(8)),
    tbl2(x, y)
    AS (SELECT Hll_create_sketch(9),
                Hll_create_sketch(10)
        UNION ALL
        SELECT Hll_create_sketch(11),
                Hll_create_sketch(12)
        UNION ALL
        SELECT Hll_create_sketch(13),
                Hll_create_sketch(14)
        UNION ALL
        SELECT Hll_create_sketch(15),
                Hll_create_sketch(16)
        UNION ALL
        SELECT Hll_create_sketch(NULL),
                Hll_create_sketch(NULL)),
    tbl3(x, y)
    AS (SELECT *
        FROM   tbl1
        UNION ALL
        SELECT *
        FROM   tbl2)
SELECT Hll_combine_sketches(x, y)
FROM   tbl3;
```

JSON-Funktionen

Themen

- Die Funktion IS_VALID_JSON
- Die Funktion IS_VALID_JSON_ARRAY
- Die Funktion JSON_ARRAY_LENGTH
- Die Funktion „JSON_EXTRACT_ARRAY_ELEMENT_TEXT“

- Die Funktion JSON_EXTRACT_PATH_TEXT
- Funktion JSON_PARSE
- Funktion CAN_JSON_PARSE
- Funktion JSON_SERIALISE
- Funktion JSON_SERIALIZE_TO_VARBYTE

Wenn Sie einen vergleichsweise kleinen Satz von Schlüssel-Wert-Paaren speichern müssen, können Sie vielleicht Platz sparen, indem Sie die Daten im JSON-Format speichern. Da JSON-Zeichenfolgen in einer einzigen Spalte gespeichert werden können, kann die Verwendung von JSON effizienter als das Speichern Ihrer Daten im Tabellenformat sein. Angenommen, Sie haben eine Sparse-Tabelle, in der zahlreiche Spalten alle möglichen Attribute vollständig darstellen müssen, die meisten Spaltenwerte für eine bestimmte Zeile oder Spalte jedoch NULL sind. Wenn Sie zum Speichern JSON verwenden, können Sie die Daten für eine Zeile in Schlüssel:Wert-Paaren ein einer einzelnen JSON-Zeichenfolge speichern und die kaum ausgefüllten Tabellenspalten beseitigen.

Zusätzlich können Sie JSON-Zeichenfolgen leicht ändern, sodass diese weitere Schlüssel:Wert-Paare speichern, ohne einer Tabelle Spalten hinzufügen zu müssen.

Sie sollten JSON nur in bestimmten Fällen verwenden. JSON ist keine gute Wahl, wenn es um das Speichern größerer Datensätze geht, da JSON aufgrund der Tatsache, dass disparate Daten in einer einzigen Spalte gespeichert werden, die Spaltenspeicherarchitektur von Amazon Redshift nicht nutzt. Obwohl Amazon Redshift JSON-Funktionen über CHAR- und VARCHAR-Spalten unterstützt, empfehlen wir, SUPER für die Verarbeitung von Daten im JSON-Serialisierungsformat zu verwenden. SUPER verwendet eine schemalose Post-Parse-Darstellung, die hierarchische Daten effizient abfragen kann. Weitere Informationen zum SUPER-Datentyp finden Sie unter Erfassen und Abfragen von halbstrukturierten Daten in Amazon Redshift.

JSON verwendet UTF-8-kodierte Textzeichenfolgen. Daher können JSON-Zeichenfolgen als CHAR- oder VARCHAR-Datentypen gespeichert werden. Sie verwenden VARCHAR, wenn die Zeichenfolgen Multibyte-Zeichen enthalten.

JSON-Zeichenfolgen müssen ein korrektes JSON-Format aufweisen, das den folgenden Regeln entspricht:

- Der JSON-Wert kann auf Stammverzeichnisebene ein JSON-Objekt oder ein JSON-Array sein. Ein JSON-Objekt ist ein nicht geordneter Satz von durch Komma getrennten Schlüssel:Wert-Paaren, eingeschlossen in geschweiften Klammern.

Beispiel: `{"one":1, "two":2}`

- Ein JSON-Array ist ein geordneter Satz von durch Komma getrennten Werten, eingeschlossen in eckigen Klammern.

  Ein Beispiel ist folgendes: `["first", {"one":1}, "second", 3, null]`
- JSON-Arrays verwenden einen nullbasierten Index. Das erste Element in einem Array befindet sich an Position 0. In einem Schlüssel:Wert-Paar in JSON ist der Schlüssel eine Zeichenfolge in doppelten Anführungszeichen.
- Ein JSON-Wert kann jeder der folgenden Werte sein:
  - JSON-Objekt
  - JSON-Array
  - Zeichenfolge in doppelten Anführungszeichen
  - Zahl (Ganzzahl und Gleitkommazahl)
  - Boolesch
  - Null
- Leere Objekte und leere Arrays sind gültige JSON-Werte.
- JSON-Felder unterscheiden zwischen Groß- und Kleinschreibung.
- Leerzeichen zwischen JSON-Strukturelementen (wie `{ }, [ ]`) werden ignoriert.

Die Amazon-Redshift-JSON-Funktionen und der Amazon-Redshift-COPY-Befehl verwenden dieselben Methoden, um mit Daten im JSON-Format zu arbeiten. Weitere Informationen zur Arbeit mit JSON finden Sie unter . COPY von JSON-Format

Die Funktion IS_VALID_JSON

Die IS_VALID_JSON-Funktion validiert eine JSON-Zeichenfolge. Die Funktion gibt den Booleschen Wert `true` zurück, wenn die Zeichenfolge eine korrekte JSON-Formatierung aufweist, oder `false`, wenn die Formatierung falsch ist. Verwenden Sie , um ein JSON-Array zu validieren. Die Funktion IS_VALID_JSON_ARRAY

Weitere Informationen finden Sie unter JSON-Funktionen.

Syntax

```
IS_VALID_JSON('json_string')
```

Argumente

json_string

Eine Zeichenfolge oder ein Ausdruck, die/der zu einer JSON-Zeichenfolge ausgewertet wird.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um eine Tabelle zu erstellen und JSON-Zeichenfolgen zum Testen einzufügen.

```
CREATE TABLE test_json(id int IDENTITY(0,1), json_strings VARCHAR);

-- Insert valid JSON strings --
INSERT INTO test_json(json_strings) VALUES
('{"a":2}'),
('{"a":{"b":{"c":1}}}'),
('{"a": [1,2,"b"]}');

-- Insert invalid JSON strings --
INSERT INTO test_json(json_strings) VALUES
('{{}}'),
('{1:"a"}'),
('[1,2,3]');
```

Verwenden Sie das folgende Beispiel, um die Zeichenfolgen des vorherigen Beispiels zu validieren.

```
SELECT id, json_strings, IS_VALID_JSON(json_strings)
FROM test_json
ORDER BY id;

+----+--------------------+---------------+
| id |    json_strings    | is_valid_json |
+----+--------------------+---------------+
|  0 | {"a":2}            | true          |
|  4 | {"a":{"b":{"c":1}}} | true          |
|  8 | {"a": [1,2,"b"]}   | true          |
| 12 | {{}}               | false         |
| 16 | {1:"a"}            | false         |
```

```
| 20 | [1,2,3]              | false         |
+----+----------------------+---------------+
```

Die Funktion IS_VALID_JSON_ARRAY

Die IS_VALID_JSON_ARRAY-Funktion validiert ein JSON-Array. Die Funktion gibt den Booleschen Wert `true` zurück, wenn das Array eine korrekte JSON-Formatierung aufweist, oder `false`, wenn die Formatierung falsch ist. Verwenden Sie , um eine JSON-Zeichenfolge zu validieren. Die Funktion IS_VALID_JSON

Weitere Informationen finden Sie unter JSON-Funktionen.

Syntax

```
IS_VALID_JSON_ARRAY('json_array')
```

Argumente

json_array

Eine Zeichenfolge oder ein Ausdruck, die/der zu einem JSON-Array ausgewertet wird.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um eine Tabelle zu erstellen und JSON-Zeichenfolgen zum Testen einzufügen.

```
CREATE TABLE test_json_arrays(id int IDENTITY(0,1), json_arrays VARCHAR);

-- Insert valid JSON array strings --
INSERT INTO test_json_arrays(json_arrays)
VALUES('[]'),
('["a","b"]'),
('["a",["b",1,["c",2,3,null]]]');

-- Insert invalid JSON array strings --
INSERT INTO test_json_arrays(json_arrays)
VALUES('{"a":1}'),
```

```
('a'),
('[1,2,]');
```

Verwenden Sie das folgende Beispiel, um die Zeichenfolgen des vorherigen Beispiels zu validieren.

```
SELECT json_arrays, IS_VALID_JSON_ARRAY(json_arrays)
FROM test_json_arrays ORDER BY id;

+------------------------------+---------------------+
|         json_arrays          | is_valid_json_array |
+------------------------------+---------------------+
| []                           | true                |
| ["a","b"]                    | true                |
| ["a",["b",1,["c",2,3,null]]] | true                |
| {"a":1}                      | false               |
| a                            | false               |
| [1,2,]                       | false               |
+------------------------------+---------------------+
```

Die Funktion JSON_ARRAY_LENGTH

Die Funktion JSON_ARRAY_LENGTH gibt die Anzahl von Elementen im äußeren Array einer JSON-Zeichenfolge zurück. Wenn das Argument null_if_invalid auf `true` gesetzt und die JSON-Zeichenfolge ungültig ist, gibt die Funktion anstatt eines Fehlers `NULL` zurück.

Weitere Informationen finden Sie unter JSON-Funktionen.

Syntax

```
JSON_ARRAY_LENGTH('json_array' [, null_if_invalid ] )
```

Argumente

json_array

Ein korrekt formatiertes JSON-Array.

null_if_invalid

(Optional) Ein `BOOLEAN`-Wert, der angibt, ob anstatt eines Fehlers `NULL` zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Geben Sie `true` (`t`) an, damit `NULL` zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Geben Sie `false` (`f`)

an, damit ein Fehler zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Der Standardwert ist `false`.

Rückgabetyp

`INTEGER`

Beispiele

Verwenden Sie das folgende Beispiel, um die Anzahl der Elemente im Array zurückzugeben.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14]');

+------------------+
| json_array_length |
+------------------+
|                5 |
+------------------+
```

Verwenden Sie das folgende Beispiel, um einen Fehler zurückzugeben, weil die JSON-Eingabezeichenfolge ungültig ist.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14');

ERROR: invalid json array object [11,12,13,{"f1":21,"f2":[25,26]},14
```

Verwenden Sie das folgende Beispiel, um null_if_invalid auf true zu setzen, sodass die Anweisung anstatt eines Fehlers `NULL` zurückgibt, wenn die JSON-Eingabezeichenfolge ungültig ist.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14',true);

+------------------+
| json_array_length |
+------------------+
| NULL             |
+------------------+
```

Die Funktion „JSON_EXTRACT_ARRAY_ELEMENT_TEXT“

Die Funktion JSON_EXTRACT_ARRAY_ELEMENT_TEXT gibt ein JSON-Array-Element im äußersten Array einer JSON-Zeichenfolge unter Verwendung eines nullbasierten Index zurück.

Das erste Element in einem Array befindet sich an Position 0. Wenn der Index negativ ist oder sich außerhalb des Bereichs befindet, gibt JSON_EXTRACT_ARRAY_ELEMENT_TEXT eine leere Zeichenfolge zurück. Wenn das Argument null_if_invalid auf `true` gesetzt und die JSON-Zeichenfolge ungültig ist, gibt die Funktion anstatt eines Fehlers `NULL` zurück.

Weitere Informationen finden Sie unter JSON-Funktionen.

Syntax

```
JSON_EXTRACT_ARRAY_ELEMENT_TEXT('json string', pos [, null_if_invalid ] )
```

Argumente

json_string

Eine korrekt formatierte JSON-Zeichenfolge.

pos

Eine `INTEGER`, die unter Verwendung eines nullbasierten Array-Index den Index des Array-Elements darstellt, das zurückgegeben werden soll.

null_if_invalid

(Optional) Ein `BOOLEAN`-Wert, der angibt, ob anstatt eines Fehlers `NULL` zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Geben Sie `true` (`t`) an, damit `NULL` zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Geben Sie `false` (`f`) an, damit ein Fehler zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Der Standardwert ist `false`.

Rückgabetyp

`VARCHAR`

Eine `VARCHAR`-Zeichenfolge, die das JSON-Array-Element darstellt, das von pos referenziert wird.

Beispiele

Verwenden Sie das folgende Beispiel, um ein Array-Element an Position 2 zurückzugeben, das das dritte Element eines null-basierten Array-Index ist.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('[111,112,113]', 2);

+--------------------------------+
| json_extract_array_element_text |
+--------------------------------+
|                            113 |
+--------------------------------+
```

Verwenden Sie das folgende Beispiel, um einen Fehler zurückzugeben, weil die JSON-Eingabezeichenfolge ungültig ist.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('["a",["b",1,["c",2,3,null,]]]',1);

ERROR: invalid json array object ["a",["b",1,["c",2,3,null,]]]
```

Verwenden Sie das folgende Beispiel, um null_if_invalid auf true zu setzen, sodass die Anweisung anstatt eines Fehlers NULL zurückgibt, wenn die JSON-Eingabezeichenfolge ungültig ist.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('["a",["b",1,["c",2,3,null,]]]',1,true);

+--------------------------------+
| json_extract_array_element_text |
+--------------------------------+
| NULL                           |
+--------------------------------+
```

Die Funktion JSON_EXTRACT_PATH_TEXT

Die Funktion JSON_EXTRACT_PATH_TEXT gibt den Wert für das Schlüssel-Wert-Paar zurück, auf das in einer Reihe von Pfadelementen in einer JSON-Zeichenfolge verwiesen wird. Der JSON-Pfad kann bis zu einer Tiefe von fünf Ebenen verschachtelt sein. Pfadelemente unterscheiden zwischen Groß- und Kleinschreibung. Wenn in der JSON-Zeichenfolge ein Pfadelement nicht vorhanden ist, gibt JSON_EXTRACT_PATH_TEXT NULL zurück.

Wenn das Argument null_if_invalid auf `true` gesetzt und die JSON-Zeichenfolge ungültig ist, gibt die Funktion anstatt eines Fehlers NULL zurück.

Informationen zu zusätzlichen JSON-Funktionen finden Sie unter JSON-Funktionen. Weitere Informationen zur Arbeit mit JSON finden Sie unter COPY von JSON-Format.

Syntax

```
JSON_EXTRACT_PATH_TEXT('json_string', 'path_elem' [,'path_elem'[, …] ]
 [, null_if_invalid ] )
```

Argumente

json_string

Eine korrekt formatierte JSON-Zeichenfolge.

path_elem

Ein Pfadelement in einer JSON-Zeichenfolge. Es ist mindestens ein Pfadelement erforderlich. Es können zusätzliche Pfadelemente angegeben werden, bis zu einer Tiefe von fünf Ebenen.

null_if_invalid

(Optional) Ein BOOLEAN-Wert, der angibt, ob anstatt eines Fehlers NULL zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Geben Sie `true` (`t`) an, damit NULL zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Geben Sie `false` (`f`) an, damit ein Fehler zurückgegeben wird, wenn die JSON-Eingabezeichenfolge ungültig ist. Der Standardwert ist `false`.

Amazon Redshift erkennt in einer JSON-Zeichenfolge \n als Zeichen für neue Zeilen und `\t` als Tabulatorzeichen. Um einen Backslash zu laden, muss ein Backslash als Escape-Zeichen verwendet werden (\\). Weitere Informationen finden Sie unter Escape-Zeichen in JSON.

Rückgabetyp

VARCHAR

Eine VARCHAR-Zeichenfolge, die den JSON-Wert darstellt, der von den Pfadelementen referenziert wird.

Beispiele

Verwenden Sie das folgende Beispiel, um den Wert für den Pfad `'f4', 'f6'` zurückzugeben.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}','f4', 'f6');
```

```
+------------------------+
| json_extract_path_text |
+------------------------+
| star                   |
+------------------------+
```

Verwenden Sie das folgende Beispiel, um einen Fehler zurückzugeben, weil die JSON-Eingabezeichenfolge ungültig ist.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}','f4', 'f6');

ERROR: invalid json object {"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}
```

Verwenden Sie das folgende Beispiel, um null_if_invalid auf true zu setzen, sodass die Anweisung bei einem ungültigen JSON anstatt eines Fehlers NULL zurückgibt.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}','f4',
 'f6',true);

+------------------------+
| json_extract_path_text |
+------------------------+
| NULL                   |
+------------------------+
```

Verwenden Sie das folgende Beispiel, um den Wert für den Pfad `'farm', 'barn', 'color'` zurückgegeben, wobei sich der abgerufene Wert auf der dritten Ebene befindet. Dieses Beispiel ist mit einem JSON-Lint-Tool formatiert, um das Lesen zu vereinfachen.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
    "farm": {
        "barn": {
            "color": "red",
            "feed stocked": true
        }
    }
}', 'farm', 'barn', 'color');
+------------------------+
| json_extract_path_text |
+------------------------+
```

```
| red                    |
+------------------------+
```

Verwenden Sie das folgende Beispiel, um NULL zurückzugeben, da das `'color'`-Element fehlt. Dieses Beispiel ist mit einem JSON-Lint-Tool formatiert.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
    "farm": {
        "barn": {}
    }
}', 'farm', 'barn', 'color');

+------------------------+
| json_extract_path_text |
+------------------------+
| NULL                   |
+------------------------+
```

Wenn das JSON-Format gültig ist, wird beim Versuch, ein fehlendes Element zu extrahieren, NULL zurückgegeben.

Verwenden Sie das folgende Beispiel, um den Wert für den Pfad `'house', 'appliances', 'washing machine', 'brand'` zurückzugeben.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
  "house": {
    "address": {
      "street": "123 Any St.",
      "city": "Any Town",
      "state": "FL",
      "zip": "32830"
    },
    "bathroom": {
      "color": "green",
      "shower": true
    },
    "appliances": {
      "washing machine": {
        "brand": "Any Brand",
        "color": "beige"
      },
      "dryer": {
```

```
        "brand": "Any Brand",
        "color": "white"
      }
    }
  }
}', 'house', 'appliances', 'washing machine', 'brand');

+------------------------+
| json_extract_path_text |
+------------------------+
| Any Brand              |
+------------------------+
```

Im folgenden Beispiel wird eine Beispieltabelle erstellt und sie mit SUPER-Werten aufgefüllt. Anschließend wird der Wert für den Pfad für beide Zeilen zurückgegeben. `'f2'`

```
CREATE TABLE json_example(id INT, json_text SUPER);

INSERT INTO json_example VALUES
(1, JSON_PARSE('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}')),
(2, JSON_PARSE('{
    "farm": {
        "barn": {
            "color": "red",
            "feed stocked": true
        }
    }
}'));

SELECT * FROM json_example;
id          | json_text
------------+-------------------------------------------
1           | {"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}
2           | {"farm":{"barn":{"color":"red","feed stocked":true}}}


SELECT id, JSON_EXTRACT_PATH_TEXT(JSON_SERIALIZE(json_text), 'f2') FROM json_example;

id          | json_text
------------+-------------------------------------------
1           | {"f3":1}
2           |
```

Funktion JSON_PARSE

Die Funktion JSON_PARSE analysiert Daten im JSON-Format und konvertiert sie in die SUPER-Darstellung.

Verwenden Sie die JSON_PARSE-Funktion, um mit dem Befehl INSERT oder UPDATE in den SUPER-Datentyp aufzunehmen. Wenn Sie JSON_PARSE() zum Parsing von JSON-Zeichenfolgen in SUPER-Werte verwenden, gelten bestimmte Einschränkungen. Weitere Informationen finden Sie unter Parsing-Optionen für SUPER.

Syntax

```
JSON_PARSE( {json_string | binary_value} )
```

Argumente

json_string

Ein Ausdruck, der eine serialisierte JSON-Zeichenfolge als Datentyp VARBYTE oder VARCHAR zurückgibt.

binary_value

Ein Binärwert des Datentyps VARBYTE.

Rückgabetyp

SUPER

Beispiele

Verwenden Sie das folgende Beispiel, um das JSON-Array `[10001,10002,"abc"]` in den Datentyp SUPER zu konvertieren.

```
SELECT JSON_PARSE('[10001,10002,"abc"]');

+---------------------+
|     json_parse      |
+---------------------+
| [10001,10002,"abc"] |
+---------------------+
```

Verwenden Sie das folgende Beispiel, um sicherzustellen, dass die Funktion das JSON-Array in den Datentyp SUPER konvertiert hat. Weitere Informationen finden Sie unter Die Funktion JSON_TYPEOF.

```
SELECT JSON_TYPEOF(JSON_PARSE('[10001,10002,"abc"]'));

+-------------+
| json_typeof |
+-------------+
| array       |
+-------------+
```

Funktion CAN_JSON_PARSE

Die Funktion CAN_JSON_PARSE analysiert Daten im JSON-Format und gibt `true` zurück, wenn das Ergebnis mithilfe der Funktion JSON_PARSE in einen SUPER-Wert konvertiert werden kann.

Syntax

```
CAN_JSON_PARSE( {json_string | binary_value} )
```

Argumente

json_string

Ein Ausdruck, der serialisierte JSON-Datentypen im VARBYTE- oder VARCHAR-Formular zurückgibt.

binary_value

Ein Binärwert des Datentyps VARBYTE.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um festzustellen, ob das JSON-Array `[10001,10002,"abc"]` in den Datentyp SUPER konvertiert werden kann.

```
SELECT CAN_JSON_PARSE('[10001,10002,"abc"]');

+----------------+
| can_json_parse |
+----------------+
| true           |
+----------------+
```

Funktion JSON_SERIALISE

Die Funktion JSON_SERIALISE serialisiert einen SUPER-Ausdruck in eine textbasierte JSON-Darstellung gemäß RFC 8259. Weitere Informationen zu diesem RFC finden Sie unter The JavaScript Object Notation (JSON) Data Interchange Format.

Das SUPER-Größenlimit entspricht ungefähr dem Blocklimit, und das VARCHAR-Limit ist kleiner als das SUPER-Größenlimit. Daher gibt die Funktion JSON_SERIALISE einen Fehler zurück, wenn das JSON-Format das varchar-Limit des Systems überschreitet. Wenn Sie die Größe eines SUPER-Ausdrucks überprüfen möchten, sehen Sie sich die Funktion JSON_SIZE an.

Syntax

```
JSON_SERIALIZE(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um den SUPER-Wert einer Zeichenfolge zu serialisieren.

```
SELECT JSON_SERIALIZE(JSON_PARSE('[10001,10002,"abc"]'));

+---------------------+
```

```
|   json_serialize    |
+---------------------+
| [10001,10002,"abc"] |
+---------------------+
```

Funktion JSON_SERIALIZE_TO_VARBYTE

Die Funktion JSON_SERIALIZE_TO_VARBYTE konvertiert einen SUPER-Wert in eine ähnliche JSON-Zeichenfolge wie bei JSON_SERIALIZE(), jedoch in einem VARBYTE-Wert gespeichert.

Syntax

```
JSON_SERIALIZE_TO_VARBYTE(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

VARBYTE

Beispiele

Verwenden Sie das folgende Beispiel, um einen SUPER-Wert zu serialisieren und das Ergebnis im VARBYTE-Format zurückzugeben.

```
SELECT JSON_SERIALIZE_TO_VARBYTE(JSON_PARSE('[10001,10002,"abc"]'));

+----------------------------------------+
|       json_serialize_to_varbyte        |
+----------------------------------------+
| 5b31303030312c31303030322c22616263225d |
+----------------------------------------+
```

Verwenden Sie das folgende Beispiel, um einen SUPER-Wert zu serialisieren und das Ergebnis im VARCHAR-Format zu übertragen. Weitere Informationen finden Sie unter CAST-Funktion.

```
SELECT CAST((JSON_SERIALIZE_TO_VARBYTE(JSON_PARSE('[10001,10002,"abc"]'))) AS VARCHAR);
```

```
+--------------------------+
| json_serialize_to_varbyte |
+--------------------------+
| [10001,10002,"abc"]       |
+--------------------------+
```

Machine-Learning-Funktionen

Mithilfe von Amazon-Redshift-Machine-Learning (ML) können Sie ML-Modelle mithilfe von SQL-Anweisungen trainieren und sie in SQL-Abfragen für Prognosen aufrufen. Die Modellerklärbarkeit Amazon Redshift umfasst Werte zur Feature-Bedeutung, die Ihnen helfen, zu verstehen, wie jedes Attribut in Ihren Trainingsdaten zum prognostizierten Ergebnis beiträgt.

Im Folgenden finden Sie Beschreibungen zu den Machine-Learning-Funktionen für SQL, die Amazon Redshift unterstützt.

Themen

Funktion EXPLAIN_MODEL

Die Funktion EXPLAIN_MODEL gibt einen SUPER-Datentyp zurück, der einen Bericht zur Erklärung des Modells im JSON-Format enthält. Der Erklärbarkeitsbericht enthält Informationen zum Shapley-Wert für alle Modell-Features.

Die Funktion EXPLAIN_MODEL unterstützt derzeit nur die XGBoost-Modelle AUTO ON oder AUTO OFF.

Wenn der Erklärbarkeitsbericht nicht verfügbar ist, gibt die Funktion Status zurück, die den Fortschritt des Modells anzeigen. Dies sind beispielsweise `Waiting for training job to complete`, `Waiting for processing job to complete` und `Processing job failed`.

Wenn Sie die Anweisung CREATE MODEL ausführen, wird der Erklärungsstatus zu `Waiting for training job to complete` geändert. Wenn das Modell trainiert wurde und eine Erklärungsanforderung gesendet wurde, wird der der Erklärungsstatus zu `Waiting for processing job to complete` geändert. Wenn die Modellerklärung erfolgreich abgeschlossen wurde, steht der gesamte Erklärbarkeitsbericht zur Verfügung. Andernfalls wird der Status zu `Processing job failed` geändert.

Wenn Sie die CREATE MODEL-Anweisung ausführen, können Sie den optionalen MAX_RUNTIME-Parameter zur Angabe der maximalen Zeit verwenden, die das Training in Anspruch nehmen soll. Sobald die Modellerstellung diesen Zeitraum erreicht hat, beendet Amazon Redshift die Erstellung des Modells. Wenn Sie dieses Zeitlimit bei der Erstellung eines Autopilot-Modells erreichen, gibt Amazon Redshift das bis dahin beste Modell zurück. Die Erklärbarkeit des Modells wird verfügbar, sobald das Modelltraining abgeschlossen ist. Wenn MAX_RUNTIME auf einen niedrigen Zeitraum eingestellt wurde, ist der Erklärbarkeitsbericht möglicherweise nicht verfügbar. Die Trainingszeit variiert und hängt von der Modellkomplexität, der Datengröße und anderen Faktoren ab.

Syntax

```
EXPLAIN_MODEL ('schema_name.model_name')
```

Argument

schema_name

Der Name des Schemas. Wenn kein schema_name angegeben wird, wird das aktuelle Schema ausgewählt.

model_name

Der Name des Modells Der Modellname in einem Schema muss eindeutig sein.

Rückgabetyp

Die Funktion EXPLAIN_MODEL gibt einen SUPER-Datentyp zurück, wie nachstehend gezeigt.

```
{"version":"1.0","explanations":{"kernel_shap":{"label0":{"global_shap_values":
{"x0":0.05,"x1":0.10,"x2":0.30,"x3":0.15},"expected_value":0.50}}}}
```

Beispiele

Im folgenden Beispiel wird der Erklärungsstatus `waiting for training job to complete` zurückgegeben.

```
select explain_model('customer_churn_auto_model');
                 explain_model
-------------------------------------------------------
{"explanations":"waiting for training job to complete"}
```

```
(1 row)
```

Wenn die Modellerklärung erfolgreich abgeschlossen wurde, steht der gesamte Erklärbarkeitsbericht zur Verfügung. Dies sieht wie folgt aus.

```
select explain_model('customer_churn_auto_model');
                                        explain_model
-------------------------------------------------------------------------------------------------
{"version":"1.0","explanations":{"kernel_shap":{"label0":{"global_shap_values":
{"x0":0.05386043365892927,"x1":0.10801289723274592,"x2":0.23227865827017378,"x3":0.067668513394
(1 row)
```

Da die Funktion EXPLAIN_MODEL den Datentyp SUPER zurückgibt, können Sie den Erklärbarkeitsbericht abfragen. So können Sie `global_shap_values`, `expected_value` oder Feature-spezifische Shapley-Werte extrahieren.

Im folgenden Beispiel wird `global_shap_values` für das Modell extrahiert.

```
select json_table.report.explanations.kernel_shap.label0.global_shap_values from
 (select explain_model('customer_churn_auto_model') as report) as json_table;
                                                    global_shap_values
-------------------------------------------------------------------------------------------------
{"state":0.10983770427197151,"account_length":0.1772441398408543,"area_code":0.0862682396863959
(1 row)
```

Im folgenden Beispiel wird `global_shap_values` für das Feature x0 extrahiert.

```
select json_table.report.explanations.kernel_shap.label0.global_shap_values.x0 from
 (select explain_model('customer_churn_auto_model') as report) as json_table;
          x0
------------------------
  0.05386043365892927
(1 row)
```

Wenn das Modell in einem bestimmten Schema erstellt wird und Sie Zugriff auf das erstellte Modell haben, können Sie die Modellerklärung wie folgt abfragen.

```
-- Check the current schema
SHOW search_path;
   search_path
-----------------
```

```
  $user, public
(1 row)
-- If you have the privilege to access the model explanation
-- in `test_schema`
SELECT explain_model('test_schema.test_model_name');
                        explain_model
-------------------------------------------------------------
{"explanations":"waiting for training job to complete"}
(1 row)
```

Mathematische Funktionen

Themen

- Die Funktion RADIANS
- Die Funktion RANDOM
- Die Funktion ROUND
- Die Funktion SIN
- Die Funktion SIGN
- Die Funktion SQRT
- Die Funktion TAN
- Die Funktion TRUNC

In diesem Abschnitt werden die mathematischen Operatoren und Funktionen beschrieben, die in Amazon Redshift unterstützt werden.

Symbole für mathematische Operatoren

In der folgenden Tabelle werden die unterstützten mathematischen Operatoren aufgeführt.

Unterstützte Operatoren

Operator	Beschreibung	Beispiel	Ergebnis
+	Addition	2 + 3	5
-	Subtraktion	2 - 3	-1
*	Multiplikation	2 * 3	6
/	Division	4 / 2	2
%	Modulo	5 % 4	1
^	Potenzierung	2,0 ^ 3,0	8
|/	Quadratwurzel	| / 25,0	5

Operator	Beschreibung	Beispiel	Ergebnis
||/	Kubikwurzel	|| / 27,0	3
@	Absoluter Wert	@ -5,0	5
<<	bitweise Verschiebung nach links	1 << 4	16
>>	bitweise Verschiebung nach rechts	8 >> 2	2
&	bitweises „und“	8 & 2	0

Beispiele

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die gezahlte Provision zuzüglich einer Bearbeitungsgebühr in Höhe von 2,00 USD für eine bestimmte Transaktion zu berechnen.

```
SELECT
    commission,
    (commission + 2.00) AS comm
FROM
    sales
WHERE
    salesid = 10000;

+------------+-------+
| commission | comm  |
+------------+-------+
```

```
|      28.05 | 30.05 |
+------------+-------+
```

Verwenden Sie das folgende Beispiel, um 20 Prozent des Verkaufspreises für eine bestimmte Transaktion zu berechnen.

```
SELECT pricepaid, (pricepaid * .20) as twentypct
FROM sales
WHERE salesid=10000;

+-----------+-----------+
| pricepaid | twentypct |
+-----------+-----------+
|       187 |      37.4 |
+-----------+-----------+
```

Verwenden Sie das folgende Beispiel, um Ticketverkäufe auf der Basis eines kontinuierlichen Wachstumsmusters zu prognostizieren. In diesem Beispiel gibt die Unterabfrage die Anzahl der Tickets zurück, die 2008 verkauft wurden. Dieses Ergebnis wird exponentiell mit einer kontinuierlichen Wachstumsrate von 5 % über 10 Jahre multipliziert.

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid AND year=2008)^((5::float/100)*10) AS qty10years;

+------------------+
|    qty10years    |
+------------------+
| 587.664019657491 |
+------------------+
```

Verwenden Sie das folgende Beispiel, um den gezahlten Gesamtpreis und die Provision für Verkäufe mit einer Datums-ID zu suchen, die gleich oder größer als 2000 ist. Anschließend wird die Gesamtprovision vom gezahlten Gesamtpreis abgezogen.

```
SELECT SUM(pricepaid) AS sum_price, dateid,
SUM(commission) AS sum_comm, (SUM(pricepaid) - SUM(commission)) AS value
FROM sales
WHERE dateid >= 2000
GROUP BY dateid
ORDER BY dateid
LIMIT 10;
```

```
+-----------+--------+----------+-----------+
| sum_price | dateid | sum_comm |   value   |
+-----------+--------+----------+-----------+
|    305885 |   2000 | 45882.75 | 260002.25 |
|    316037 |   2001 | 47405.55 | 268631.45 |
|    358571 |   2002 | 53785.65 | 304785.35 |
|    366033 |   2003 | 54904.95 | 311128.05 |
|    307592 |   2004 |  46138.8 |  261453.2 |
|    333484 |   2005 |  50022.6 |  283461.4 |
|    317670 |   2006 |  47650.5 |  270019.5 |
|    351031 |   2007 | 52654.65 | 298376.35 |
|    313359 |   2008 | 47003.85 | 266355.15 |
|    323675 |   2009 | 48551.25 | 275123.75 |
+-----------+--------+----------+-----------+
```

Funktion ABS

ABS berechnet den absoluten Wert einer Zahl, wobei diese Zahl ein Literal oder ein Ausdruck sein kann, der zu einer Zahl ausgewertet wird.

Syntax

```
ABS(number)
```

Argumente

number (Zahl

Zahl oder Ausdruck, der zu einer Zahl ausgewertet wird. Es kann sich um den Typ `SMALLINT`, `INTEGER`, `BIGINT`, `DECIMAL`, `FLOAT4`, `FLOAT8` oder `SUPER` handeln.

Rückgabetyp

ABS gibt denselben Datentyp wie sein Argument zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um den absoluten Wert von `-38` zu nutzen.

```
SELECT ABS(-38);
```

```
+-----+
| abs |
+-----+
|  38 |
+-----+
```

Verwenden Sie das folgende Beispiel, um den absoluten Wert von `(14-76)` zu nutzen.

```
SELECT ABS(14-76);

+-----+
| abs |
+-----+
|  62 |
+-----+
```

Die Funktion ACOS

ACOS ist eine trigonometrische Funktion, die den Arcuscosinus einer Zahl zurückgibt. Der Rückgabewert wird in Radianten ausgedrückt und liegt zwischen `0` und `PI`.

Syntax

```
ACOS(number)
```

Argumente

number (Zahl

Der Eingabeparameter ist eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Arcuscosinus von `-1` zurückzugeben.

```
SELECT ACOS(-1);

+-------------------+
```

```
|       acos        |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Verwenden Sie das folgende Beispiel, um den Arcuscosinus von `.5` in die entsprechende Zahl von Grad zu konvertieren.

```
SELECT (ACOS(.5) * 180/(SELECT PI())) AS degrees;

+-------------------+
|      degrees      |
+-------------------+
| 60.00000000000001 |
+-------------------+
```

Die Funktion ASIN

ASIN ist eine trigonometrische Funktion, die den Arcussinus einer Zahl zurückgibt. Der Rückgabewert wird in Radianten ausgedrückt und liegt zwischen `PI/2` und `-PI/2`.

Syntax

```
ASIN(number)
```

Argumente

number (Zahl

Der Eingabeparameter ist eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Arcussinus von 1 zurückzugeben.

```
SELECT ASIN(1) AS halfpi;

+-------------------+
```

```
|       halfpi       |
+--------------------+
| 1.5707963267948966 |
+--------------------+
```

Verwenden Sie das folgende Beispiel, um den Arcussinus von `.5` in die entsprechende Zahl von Grad zu konvertieren.

```
SELECT (ASIN(.5) * 180/(SELECT PI())) AS degrees;

+--------------------+
|      degrees       |
+--------------------+
| 30.000000000000004 |
+--------------------+
```

Die Funktion ATAN

ATAN ist eine trigonometrische Funktion, die den Arcustangens einer Zahl zurückgibt. Der Rückgabewert wird in Radianten ausgedrückt und liegt zwischen `-PI` und `PI`.

Syntax

```
ATAN(number)
```

Argumente

number (Zahl

Der Eingabeparameter ist eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Arcustangens von 1 zurückzugeben und mit 4 multipliziert.

```
SELECT ATAN(1) * 4 AS pi;
```

```
+-------------------+
|        pi         |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Verwenden Sie das folgende Beispiel, um den Arcustangens von 1 in die entsprechende Zahl von Grad zu konvertieren.

```
SELECT (ATAN(1) * 180/(SELECT PI())) AS degrees;

+---------+
| degrees |
+---------+
|      45 |
+---------+
```

Die Funktion ATAN2

ATAN2 ist eine trigonometrische Funktion, die den Arcustangens einer Zahl dividiert durch eine andere Zahl zurückgibt. Der Rückgabewert wird in Radianten ausgedrückt und liegt zwischen `PI/2` und `-PI/2`.

Syntax

```
ATAN2(number1, number2)
```

Argumente

number1

Eine `DOUBLE PRECISION`-Zahl.

number2

Eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Arcustangens von 2/2 zurückzugeben und mit 4 multipliziert.

```
SELECT ATAN2(2,2) * 4 AS PI;

+-------------------+
|        pi         |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Verwenden Sie das folgende Beispiel, um den Arcustangens von 1/0 (der mit 0 ausgewertet wird) in die entsprechende Zahl von Grad zu konvertieren.

```
SELECT (ATAN2(1,0) * 180/(SELECT PI())) AS degrees;

+---------+
| degrees |
+---------+
|      90 |
+---------+
```

Die Funktion CBRT

Die CBRT-Funktion ist eine mathematische Funktion, die die Kubikwurzel einer gegebenen Zahl berechnet.

Syntax

```
CBRT(number)
```

Argumente

CBRT übernimmt eine DOUBLE PRECISION-Zahl als ein Argument.

Rückgabetyp

DOUBLE PRECISION

Beispiele

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Kubikwurzel der Provision, die für eine bestimmte Verkaufstransaktion gezahlt wird, zu berechnen.

```
SELECT CBRT(commission) FROM sales WHERE salesid=10000;

+--------------------+
|        cbrt        |
+--------------------+
| 3.0383953904884344 |
+--------------------+
```

Die Funktion CEILING (oder CEIL)

Die CEILING- oder CEIL-Funktion wird verwendet, um eine Zahl auf die nächste ganze Zahl aufzurunden. (Die Die Funktion FLOOR rundet eine Zahl auf die nächste ganze Zahl ab.)

Syntax

```
{CEIL | CEILING}(number)
```

Argumente

number (Zahl

Die Zahl oder der Ausdruck, der zu einer Zahl ausgewertet wird. Es kann sich um den Typ `SMALLINT`, `INTEGER`, `BIGINT`, `DECIMAL`, `FLOAT4`, `FLOAT8` oder `SUPER` handeln.

Rückgabetyp

CEILING und CEIL geben denselben Datentyp wie ihr Argument zurück.

Wenn die Eingabe den Typ `SUPER` hat, behält die Ausgabe den gleichen dynamischen Typ wie die Eingabe bei, während der statische Typ weiterhin den Typ SUPER hat. Wenn der dynamische Typ von `SUPER` keine Zahl ist, gibt Amazon Redshift eine Null zurück.

Beispiele

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Decke der Provision, die für eine bestimmte Verkaufstransaktion gezahlt wird, zu berechnen.

```
SELECT CEILING(commission) FROM sales
WHERE salesid=10000;

+---------+
| ceiling |
+---------+
|      29 |
+---------+
```

Die Funktion COS

COS ist eine trigonometrische Funktion, die den Cosinus einer Zahl zurückgibt. Der Rückgabewert wird in Radianten ausgedrückt und liegt zwischen `-1` und `1`, jeweils einschließlich.

Syntax

```
COS(double_precision)
```

Argumente

number (Zahl

Der Eingabeparameter ist eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

Die COS-Funktion gibt eine `DOUBLE PRECISION`-Zahl zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um den Kosinus von `0` zurückzugeben.

```
SELECT COS(0);
```

```
+-----+
| cos |
+-----+
|   1 |
+-----+
```

Verwenden Sie das folgende Beispiel, um den Kosinus von `pi` zurückzugeben.

```
SELECT COS(PI());

+-----+
| cos |
+-----+
|  -1 |
+-----+
```

Die Funktion COT

COT ist eine trigonometrische Funktion, die den Kotangens einer Zahl zurückgibt. Der Eingabeparameter darf nicht null sein.

Syntax

```
COT(number)
```

Argument

number (Zahl

Der Eingabeparameter ist eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Kotangens von 1 zurückzugeben.

```
SELECT COT(1);
```

```
+--------------------+
|        cot         |
+--------------------+
| 0.6420926159343306 |
+--------------------+
```

Die Funktion DEGREES

Konvertiert einen Winkel in Radianten in die Entsprechung in Grad.

Syntax

```
DEGREES(number)
```

Argument

number (Zahl

Der Eingabeparameter ist eine DOUBLE PRECISION-Zahl.

Rückgabetyp

DOUBLE PRECISION

Beispiele

Verwenden Sie das folgende Beispiel, um die Entsprechung in Grad des Radianten 0,5 zurückzugeben.

```
SELECT DEGREES(.5);

+-------------------+
|      degrees      |
+-------------------+
| 28.64788975654116 |
+-------------------+
```

Verwenden Sie das folgende Beispiel, um PI-Radianten in Grad zu konvertieren.

```
SELECT DEGREES(pi());

+---------+
```

```
| degrees |
+---------+
|     180 |
+---------+
```

Die Funktion DEXP

Die DEXP-Funktion gibt den exponentiellen SPLI-Wert in wissenschaftlicher Notierung für eine Doppelpräszisionsnummer zurück. Der einzige Unterschied zwischen den Funktionen DEXP und EXP besteht darin, dass es sich beim Parameter für DEXP um eine `DOUBLE PRECISION` handeln muss.

Syntax

```
DEXP(number)
```

Argument

number (Zahl

Der Eingabeparameter ist eine `DOUBLE PRECISION`-Zahl.

Rückgabetyp

`DOUBLE PRECISION`

Beispiel

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Die DEXP-Funktion wird verwendet, um Ticketverkäufe auf der Basis eines kontinuierlichen Wachstumsmusters zu prognostizieren. In diesem Beispiel gibt die Unterabfrage die Anzahl der Tickets zurück, die 2008 verkauft wurden. Dieses Ergebnis wird mit dem Ergebnis der DEXP-Funktion multipliziert, das eine kontinuierliche Wachstumsrate von 7 % über 10 Jahre angibt.

```
SELECT (SELECT SUM(qtysold)
FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * DEXP((7::FLOAT/100)*10) qty2010;
```

```
+-------------------+
|      qty2010      |
+-------------------+
| 695447.4837722216 |
+-------------------+
```

Die Funktion DLOG1

Die DLOG1-Funktion gibt den natürlichen Logarithmus des Eingabeparameters zurück. Synonym von Die Funktion LN.

Die Funktion DLOG10

Die DLOG10-Funktion gibt den Logarithmus des Eingabeparameters zur Basis 10 zurück.

Synonym von Die Funktion LOG.

Syntax

```
DLOG10(number)
```

Argument

number (Zahl

Der Eingabeparameter ist eine DOUBLE PRECISION-Zahl.

Rückgabetyp

DOUBLE PRECISION

Beispiel

Verwenden Sie das folgende Beispiel, um den Logarithmus der Zahl 100 zur Basis 10 zurückzugeben.

```
SELECT DLOG10(100);

+--------+
| dlog10 |
+--------+
|      2 |
```

```
+--------+
```

Die Funktion EXP

Die EXP-Funktion implementiert die Exponentialfunktion für einen numerischen Ausdruck, oder die Basis des natürlichen Logarithmus, e, potenziert mit dem Ausdruck. Die EXP-Funktion ist die Umkehrung von Die Funktion LN.

Syntax

```
EXP(expression)
```

Argument

expression

Der Ausdruck muss den Datentyp `INTEGER`, `DECIMAL` oder `DOUBLE PRECISION` aufweisen.

Rückgabetyp

`DOUBLE PRECISION`

Beispiel

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Die EXP-Funktion wird verwendet, um Ticketverkäufe auf der Basis eines kontinuierlichen Wachstumsmusters zu prognostizieren. In diesem Beispiel gibt die Unterabfrage die Anzahl der Tickets zurück, die 2008 verkauft wurden. Dieses Ergebnis wird mit dem Ergebnis der EXP-Funktion multipliziert, das eine kontinuierliche Wachstumsrate von 7 % über 10 Jahre angibt.

```
SELECT (SELECT SUM(qtysold)
FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * EXP((7::FLOAT/100)*10) qty2018;

+-------------------+
|      qty2018      |
+-------------------+
| 695447.4837722216 |
```

```
+-------------------+
```

Die Funktion FLOOR

Die FLOOR-Funktion rundet eine Zahl auf die nächste ganze Zahl ab.

Syntax

```
FLOOR(number)
```

Argument

number (Zahl

Die Zahl oder der Ausdruck, der zu einer Zahl ausgewertet wird. Es kann sich um den Typ SMALLINT, INTEGER, BIGINT, DECIMAL, FLOAT4, FLOAT8 oder SUPER handeln.

Rückgabetyp

FLOOR gibt denselben Datentyp wie sein Argument zurück.

Wenn die Eingabe den Typ SUPER hat, behält die Ausgabe den gleichen dynamischen Typ wie die Eingabe bei, während der statische Typ weiterhin den Typ SUPER hat. Wenn der dynamische Typ von SUPER keine Zahl ist, gibt Amazon Redshift NULL zurück.

Beispiele

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um den Wert der Provision zu zeigen, die für eine bestimmte Verkaufstransaktion vor und nach Verwendung der FLOOR-Funktion bezahlt wurde.

```
SELECT commission
FROM sales
WHERE salesid=10000;

+------------+
| commission |
+------------+
|      28.05 |
+------------+
```

```
SELECT FLOOR(commission)
FROM sales
WHERE salesid=10000;

+-------+
| floor |
+-------+
|    28 |
+-------+
```

Die Funktion LN

Gibt den natürlichen Logarithmus des Eingabeparameters zurück.

Synonym von Die Funktion DLOG1.

Syntax

```
LN(expression)
```

Argument

expression

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

> **Note**
>
> Diese Funktion gibt für einige Datentypen einen Fehler zurück, wenn der Ausdruck auf eine benutzererstellte Amazon-Redshift-Tabelle oder eine Amazon-Redshift-STL- oder -SRV-Systemtabelle verweist.

Ausdrücke mit den folgenden Datentypen führen zu einem Fehler, wenn sie eine benutzererstellte oder eine Systemtabelle referenzieren. Ausdrücke mit den folgenden Datentypen werden ausschließlich auf dem Führungsknoten ausgeführt:

- BOOLEAN
- CHAR
- DATE

- DECIMAL oder NUMERIC
- TIMESTAMP
- VARCHAR

Ausdrücke mit den folgenden Datentypen werden für benutzererstellte und STL- oder STV-Systemtabellen erfolgreich ausgeführt:

- BIGINT
- DOUBLE PRECISION
- INTEGER
- REAL
- SMALLINT

Rückgabetyp

Die LN-Funktion gibt denselben Typ wie der Eingabeausdruck zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um den natürliche Logarithmus bzw. Logarithmus zur Basis e der Zahl 2,718281828 zurückzugeben.

```
SELECT LN(2.718281828);

+--------------------+
|         ln         |
+--------------------+
| 0.9999999998311267 |
+--------------------+
```

Beachten Sie, dass die Antwort beinahe gleich 1 ist.

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um den natürlichen Logarithmus der Werte in der Spalte „userid“ in der Tabelle USERS zurückgegeben:

```
SELECT username, LN(userid) FROM users ORDER BY userid LIMIT 10;
```

```
+----------+--------------------+
| username |         ln         |
+----------+--------------------+
| JSG99FHE |                  0 |
| PGL08LJI | 0.6931471805599453 |
| IFT66TXU | 1.0986122886681098 |
| XDZ38RDD | 1.3862943611198906 |
| AEB55QTM | 1.6094379124341003 |
| NDQ15VBM |  1.791759469228055 |
| OWY35QYB | 1.9459101490553132 |
| AZG78YIP | 2.0794415416798357 |
| MSD36KVR | 2.1972245773362196 |
| WKW41AIW |  2.302585092994046 |
+----------+--------------------+
```

Die Funktion LOG

Gibt den Logarithmus einer Zahl zurück.

Wenn Sie diese Funktion verwenden, um den Logarithmus zur Basis 10 zu berechnen, können Sie auch Die Funktion DLOG10 verwenden.

Syntax

```
LOG([base, ]argument)
```

Parameter

base

(Optional) Die Basis der Logarithmusfunktion. Diese Zahl muss positiv sein und darf nicht gleich 1 sein. Wenn dieser Parameter weggelassen wird, berechnet Amazon Redshift den Logarithmus zur Basis 10 des Arguments.

argument

Das Argument der Logarithmusfunktion. Diese Zahl muss positiv sein. Gibt `0` zurück, wenn das Argument den Wert 1 hat.

Rückgabetyp

Die LOG-Funktion gibt eine `DOUBLE PRECISION`-Zahl zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um den Logarithmus der Zahl 100 zur Basis 2 zurückzugeben.

```
SELECT LOG(2, 100);
+-------------------+
|        log        |
+-------------------+
| 6.643856189774725 |
+-------------------+
```

Verwenden Sie das folgende Beispiel, um den Logarithmus der Zahl 100 zur Basis 10 zurückzugeben. Beachten Sie, dass Amazon Redshift eine Basis von 10 voraussetzt, wenn Sie den Basisparameter weglassen.

```
SELECT LOG(100);

+-----+
| log |
+-----+
|   2 |
+-----+
```

Die Funktion MOD

Gibt den Rest von zwei Zahlen zurück, auch bekannt als Modulo-Operation. Um das Ergebnis zu berechnen, wird der erste Parameter durch den zweiten geteilt.

Syntax

```
MOD(number1, number2)
```

Argumente

number1

Der erste Eingabeparameter ist eine Zahl vom Typ `INTEGER`, `SMALLINT`, `BIGINT` oder `DECIMAL`. Wenn es sich bei einem der beiden Parameter um einen Parameter des Typs `DECIMAL` handelt, muss es sich beim anderen Parameter ebenfalls um einen Parameter des Typs `DECIMAL` handeln. Wenn es sich bei einem der beiden Parameter um einen Parameter des Typs `INTEGER` handelt, kann es sich beim anderen Parameter um einen Parameter des Typs `INTEGER`,

SMALLINT oder BIGINT handeln. Beide Parameter können auch vom Typ SMALLINT oder BIGINT sein, aber ein Parameter kann nicht vom Typ SMALLINT sein, wenn der andere ein BIGINT ist.

number2

Der zweite Parameter ist eine Zahl vom Typ INTEGER, SMALLINT, BIGINT oder DECIMAL. Die gleichen Datentypregeln gelten für number2 wie für number1.

Rückgabetyp

Der Rückgabetyp der MOD-Funktion ist der gleiche numerische Typ wie die Eingabeparameter, wenn beide Eingabeparameter denselben Datentyp haben. Wenn es sich bei einem der Eingabeparameter um einen INTEGER handelt, ist der Rückgabetyp auch ein INTEGER. Gültige Rückgabearten sind DECIMAL, INT, SMALLINT und BIGINT.

Nutzungshinweise

Sie können % als Modulo-Operator verwenden.

Beispiele

Verwenden Sie das folgende Beispiel, um den Rest einer Division von zwei Zahlen zurückgegeben:

```
SELECT MOD(10, 4);

+-----+
| mod |
+-----+
|   2 |
+-----+
```

Verwenden Sie das folgende Beispiel, um ein DECIMAL-Ergebnis zurückzugeben, wenn Sie die MOD-Funktion verwenden.

```
SELECT MOD(10.5, 4);

+-----+
| mod |
+-----+
| 2.5 |
+-----+
```

Verwenden Sie das folgende Beispiel, um vor dem Ausführen der MOD-Funktion eine Zahl umzuwandeln. Weitere Informationen finden Sie unter CAST-Funktion.

```
SELECT MOD(CAST(16.4 AS INTEGER), 5);

+-----+
| mod |
+-----+
|   1 |
+-----+
```

Verwenden Sie das folgende Beispiel, um zu überprüfen, ob der erste Parameter gerade ist, indem Sie ihn durch 2 teilen.

```
SELECT mod(5,2) = 0 AS is_even;

+---------+
| is_even |
+---------+
| false   |
+---------+
```

Verwenden Sie das folgende Beispiel, um % als Modulo-Operator zu verwenden.

```
SELECT 11 % 4 as remainder;

 +-----------+
| remainder |
+-----------+
|         3 |
+-----------+
```

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um Informationen zu Kategorien mit ungeraden Nummern in der Tabelle CATEGORY zurückzugeben.

```
SELECT catid, catname
FROM category
WHERE MOD(catid,2)=1
```

```
ORDER BY 1,2;

+-------+-----------+
| catid |  catname  |
+-------+-----------+
|     1 | MLB       |
|     3 | NFL       |
|     5 | MLS       |
|     7 | Plays     |
|     9 | Pop       |
|    11 | Classical |
+-------+-----------+
```

Die Funktion PI

Die PI-Funktion gibt den Wert von Pi auf 14 Dezimalstellen zurück.

Syntax

```
PI()
```

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Wert von Pi zurückzugeben.

```
SELECT PI();

+------------------+
|        pi        |
+------------------+
| 3.141592653589793 |
+------------------+
```

Die Funktion POWER

Die POWER-Funktion ist eine Exponentialfunktion, die einen numerischen Ausdruck mit der Potenz eines zweiten numerischen Ausdrucks potenziert. Beispielsweise wird 2 in der dritten Potenz als `POWER(2,3)` berechnet. Das Ergebnis ist 8.

Syntax

```
{POW | POWER}(expression1, expression2)
```

Argumente

expression1

Der numerische Ausdruck, der potenziert werden soll. Muss ein `INTEGER`-, `DECIMAL`- oder `FLOAT`-Datentyp sein.

expression2

Potenz, mit der expression1potenziert werden soll. Muss ein `INTEGER`-, `DECIMAL`- oder `FLOAT`-Datentyp sein.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Im folgenden Beispiel wird die POWER-Funktion verwendet, um die Ticketverkäufe in den nächsten 10 Jahren vorherzusagen, basierend auf der Zahl der im Jahr 2008 verkauften Tickets (das Ergebnis der Unterabfrage). Die Wachstumsrate wird in diesem Beispiel auf 7 % festgelegt.

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * POW((1+7::FLOAT/100),10) qty2010;

+-------------------+
|      qty2010      |
+-------------------+
| 679353.7540885945 |
+-------------------+
```

Das folgende Beispiel ist eine Variante des vorherigen Beispiels. Die Wachstumsrate wird auch hier auf 7 % pro Jahr festgelegt. Das Intervall ist jedoch als Monat festgelegt (120 Monate über 10 Jahre).

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * POW((1+7::FLOAT/100/12),120) qty2010;

+-----------------+
|     qty2010     |
+-----------------+
| 694034.54678046 |
+-----------------+
```

Die Funktion RADIANS

Die RADIANS-Funktion konvertiert einen Winkel in Grad in die Entsprechung im Bogenmaß.

Syntax

```
RADIANS(number)
```

Argument

number (Zahl

Der Eingabeparameter ist eine DOUBLE PRECISION-Zahl.

Rückgabetyp

DOUBLE PRECISION

Beispiele

Verwenden Sie das folgende Beispiel, um die Entsprechung in 180 Grad des Radianten zurückzugeben.

```
SELECT RADIANS(180);

+-------------------+
|      radians      |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Die Funktion RANDOM

Die RANDOM-Funktion generiert einen zufälligen Wert zwischen 0,0 (einschließlich) und 1,0 (ausschließlich).

Syntax

```
RANDOM()
```

Rückgabetyp

`DOUBLE PRECISION`

Nutzungshinweise

Rufen Sie RANDOM auf, nachdem für den Befehl SET ein SEED-Wert festgelegt wurde, damit RANDOM Zahlen in einer vorhersagbaren Folge generiert.

Beispiele

Verwenden Sie das folgende Beispiel, um einen Zufallswert zwischen 0 und 99 zu berechnen. Wenn die zufällige Zahl 0 bis 1 ist, produziert diese Abfrage eine zufällige Zahl zwischen 0 und 100.

```
SELECT CAST(RANDOM() * 100 AS INT);

+------+
| int4 |
+------+
|   59 |
+------+
```

In diesem Beispiel wird der Befehl SET verwendet, um einen SEED-Wert festzulegen, sodass RANDOM eine vorhersagbare Folge von Zahlen generiert.

Verwenden Sie das folgende Beispiel, um drei zufällige Ganzzahlen zurückzugeben, ohne den Wert SEED festzulegen.

```
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|    6 |
```

```
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   68 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   56 |
+------+
```

Verwenden Sie das folgende Beispiel, um den SEED-Wert auf `.25` festzulegen und drei weitere RANDOM-Zahlen zurückzugeben.

```
SET SEED TO .25;
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   21 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   79 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   12 |
+------+
```

Verwenden Sie das folgende Beispiel, um zum Schluss den SEED-Wert auf `.25` zurückzusetzen und zu überprüfen, ob RANDOM dieselben Ergebnisse wie in den vorherigen drei Aufrufen zurückgibt.

```
SET SEED TO .25;
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   21 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   79 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   12 |
+------+
```

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um eine einheitliche Zufallsstichprobe von 10 Elementen aus der Tabelle SALES abzurufen.

```
SELECT *
FROM sales
ORDER BY RANDOM()
LIMIT 10;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|   45422 |  51114 |     5983 |   24482 |    4369 |   2118 |       1 |       195 |
 29.25 | 2008-10-19 05:20:07 |
|   42481 |  47638 |     4573 |    6198 |    6479 |   1987 |       4 |      1140 |
   171 | 2008-06-10 09:39:19 |
```

```
|   31494 |  34759 |    18895 |    4719 |    7753 |   2090 |       4 |      1024 |
 153.6 | 2008-09-21 03:44:26 |
|  119388 | 136685 |    21815 |   41905 |    2071 |   1884 |       1 |       359 |
 53.85 | 2008-02-27 10:43:10 |
|  166990 | 225037 |    18529 |    7628 |     746 |   2113 |       1 |      2009 |
 301.35 | 2008-10-14 10:07:44 |
|   11146 |  12096 |    42685 |    6619 |    1876 |   2123 |       1 |        29 |
  4.35 | 2008-10-24 06:23:54 |
|  148537 | 172056 |    15102 |   11787 |    6122 |   1923 |       2 |       480 |
    72 | 2008-04-07 03:58:23 |
|   68945 |  78387 |     7359 |   18323 |    6636 |   1910 |       1 |       457 |
 68.55 | 2008-03-25 08:31:03 |
|   52796 |  59576 |     9909 |   15102 |    7958 |   1951 |       1 |       479 |
 71.85 | 2008-05-05 02:25:08 |
|   90684 | 103522 |    38052 |   21549 |    7384 |   2117 |       1 |       313 |
 46.95 | 2008-10-18 05:43:11 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Verwenden Sie das folgende Beispiel, um jetzt eine zufällige Stichprobe von 10 Elementen abzurufen, wählen Sie die Elemente jedoch im Verhältnis zu deren Preis aus. Beispiel: Ein Element, das doppelt so teuer wie ein anderes Element ist, wird doppelt so wahrscheinlich in den Abfrageergebnissen angezeigt.

```
SELECT *
FROM sales
ORDER BY -LOG(RANDOM()) / pricepaid
LIMIT 10;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|  158340 | 208208 |    17082 |   42018 |    1211 |   2160 |       4 |      6852 |
 1027.8 | 2008-11-30 12:21:43 |
|   53250 |  60069 |    12644 |    7066 |    7942 |   1838 |       4 |      1528 |
 229.2 | 2008-01-12 11:24:56 |
|   22929 |  24938 |    47314 |    6503 |     179 |   2000 |       3 |       741 |
 111.15 | 2008-06-23 08:04:50 |
|  164980 | 221181 |     1949 |   19670 |    1471 |   1906 |       1 |      1330 |
 199.5 | 2008-03-21 07:59:51 |
```

```
|  159641 | 211179 |    44897 |   16652 |    7458 |   2128 |       1 |      1019 |
 152.85 | 2008-10-29 02:02:15 |
|   73143 |  83439 |     5716 |    5727 |    7314 |   1903 |       1 |       248 |
  37.2 | 2008-03-18 11:07:42 |
|   84778 |  96749 |    46608 |   32980 |    3883 |   1999 |       2 |       958 |
 143.7 | 2008-06-22 12:13:31 |
|  171096 | 232929 |    43683 |    8536 |    8353 |   1870 |       1 |       929 |
 139.35 | 2008-02-13 01:36:36 |
|   74212 |  84697 |    39809 |   15569 |    5525 |   2105 |       2 |       896 |
 134.4 | 2008-10-06 11:47:50 |
|  158011 | 207556 |    25399 |   16881 |     232 |   2088 |       2 |      2526 |
 378.9 | 2008-09-19 06:00:26 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Die Funktion ROUND

Die ROUND-Funktion rundet Zahlen auf den nächsten Ganzzahl- oder Dezimalwert auf.

Die ROUND-Funktion kann optional ein zweites Argument als `INTEGER` umfassen, um die Anzahl der Dezimalstellen für die Rundung in beide Richtungen anzugeben. Wenn Sie das zweite Argument nicht angeben, wird die Funktion auf die nächste ganze Zahl gerundet. Wenn das zweite Argument Ganzzahl angegeben wurde, wird die Funktion auf die nächste Zahl mit einer Genauigkeit von Ganzzahl Dezimalstellen gerundet.

Syntax

```
ROUND(number [ , integer ] )
```

Argumente

number (Zahl

Eine Zahl oder ein Ausdruck, der zu einer Zahl ausgewertet wird. Es kann sich um den Typ `DECIMAL`, `FLOAT8` oder `SUPER` handeln. Amazon Redshift kann implizit andere numerische Datentypen konvertieren.

integer

(Optional) Eine `INTEGER`, die die Zahl der Dezimalstellen für das Runden in beide Richtungen angibt. Der `SUPER`-Datentyp wird für dieses Argument nicht unterstützt.

Rückgabetyp

ROUND gibt denselben numerischen Datentyp wie die Eingabezahlzurück.

Wenn die Eingabe den Typ SUPER hat, behält die Ausgabe den gleichen dynamischen Typ wie die Eingabe bei, während der statische Typ weiterhin den Typ SUPER hat. Wenn der dynamische Typ von SUPER keine Zahl ist, gibt Amazon Redshift NULL zurück.

Beispiele

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die für eine bestimmte Transaktion gezahlte Vergütung auf die nächste ganze Zahl zu runden.

```
SELECT commission, ROUND(commission)
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |    28 |
+------------+-------+
```

Verwenden Sie das folgende Beispiel, um die für eine bestimmte Transaktion gezahlte Vergütung auf die erste Dezimalstelle zu runden.

```
SELECT commission, ROUND(commission, 1)
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |  28.1 |
+------------+-------+
```

Verwenden Sie das folgende Beispiel, um die Genauigkeit wie im vorherigen Beispiel in die entgegengesetzte Richtung zu erweitern.

```
SELECT commission, ROUND(commission, -1)
```

```
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |    30 |
+------------+-------+
```

Die Funktion SIN

SIN ist eine trigonometrische Funktion, die den Sinus einer Zahl zurückgibt. Der zurückgegebene Wert liegt zwischen `-1` und `1`.

Syntax

```
SIN(number)
```

Argument

number (Zahl

Eine `DOUBLE PRECISION`-Zahl im Bogenmaß.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um den Sinus von `-PI` zurückzugeben.

```
SELECT SIN(-PI());

+-------------------------+
|           sin           |
+-------------------------+
| -0.00000000000000012246 |
+-------------------------+
```

Die Funktion SIGN

Die SIGN-Funktion gibt das Vorzeichen (positiv oder negativ) einer Zahl zurück. Das Ergebnis der SIGN-Funktion ist 1, wenn das Argument positiv ist, -1, wenn das Argument negativ ist, oder 0, wenn folgendes Argument0.

Syntax

```
SIGN(number)
```

Argument

number (Zahl

Zahl oder Ausdruck, der zu einer Zahl ausgewertet wird. Es kann sich um den Typ DECIMAL, FLOAT8 oder SUPER handeln. Amazon Redshift kann weitere Datentypen gemäß den impliziten Konvertierungsregeln konvertieren.

Rückgabetyp

SIGN gibt denselben numerischen Datentyp wie das/die Eingabeargument(e) zurück. Bei einer Eingabe von DECIMAL ist die Ausgabe DECIMAL(1,0).

Wenn die Eingabe den Typ SUPER hat, behält die Ausgabe den gleichen dynamischen Typ wie die Eingabe bei, während der statische Typ weiterhin den Typ SUPER hat. Wenn der dynamische Typ von SUPER keine Zahl ist, gibt Amazon Redshift eine NULL zurück.

Beispiele

Das folgende Beispiel zeigt, dass Spalte d in Tabelle t2 DOUBLE PRECISION als Typ hat, da die Eingabe DOUBLE PRECISION ist, und dass Spalte n in Tabelle t2 NUMERIC(1,0) als Ausgabe hat, da die Eingabe NUMERIC ist.

```
CREATE TABLE t1(d DOUBLE PRECISION, n NUMERIC(12, 2));
INSERT INTO t1 VALUES (4.25, 4.25), (-4.25, -4.25);
CREATE TABLE t2 AS SELECT SIGN(d) AS d, SIGN(n) AS n FROM t1;
SELECT table_name, column_name, data_type FROM SVV_REDSHIFT_COLUMNS WHERE
 table_name='t1' OR table_name='t2';
```

```
+------------+-------------+----------------------+
| table_name | column_name |       data_type      |
+------------+-------------+----------------------+
| t1         | d           | double precision     |
| t1         | n           | numeric(12,2)        |
| t2         | d           | double precision     |
| t2         | n           | numeric(1,0)         |
| t1         | col1        | character varying(20) |
+------------+-------------+----------------------+
```

In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um das Vorzeichen der Decke der Provision zu bestimmten, die für eine bestimmte Verkaufstransaktion aus der Tabelle SALES gezahlt wird.

```
SELECT commission, SIGN(commission)
FROM sales WHERE salesid=10000;

+------------+------+
| commission | sign |
+------------+------+
|      28.05 |    1 |
+------------+------+
```

Die Funktion SQRT

Die SQRT-Funktion gibt die Quadratwurzel eines `NUMERIC`-Werts zurück. Die Quadratwurzel ist eine Zahl, die mit sich selbst multipliziert den angegebenen Wert ergibt.

Syntax

```
SQRT(expression)
```

Argument

expression

Der Ausdruck muss den Datentyp `INTEGER`, `DECIMAL` oder `FLOAT` oder einen Datentyp aufweisen, der implizit in diese Datentypen konvertiert. Der Ausdruck kann Funktionen enthalten.

Rückgabetyp

`DOUBLE PRECISION`

Beispiele

Verwenden Sie das folgende Beispiel, um die Quadratwurzel von 16 zurückzugeben.

```
SELECT SQRT(16);

+------+
| sqrt |
+------+
|    4 |
+------+
```

Verwenden Sie das folgende Beispiel, um die Quadratwurzel der Zeichenfolge 16 zurückzugeben.

```
SELECT SQRT('16');

+------+
| sqrt |
+------+
|    4 |
+------+
```

Verwenden Sie das folgende Beispiel, um die Quadratwurzel von 16,4 zurückzugeben, nachdem Sie die ROUND-Funktion verwendet haben.

```
SELECT SQRT(ROUND(16.4));

+------+
| sqrt |
+------+
|    4 |
+------+
```

Verwenden Sie das folgende Beispiel, um die Länge des Radius zurückzugeben, wenn die Fläche eines Kreises gegeben ist. Der Radius wird beispielsweise in Zoll berechnet, wenn die Fläche in Quadratzoll angegeben ist. Die Fläche in dem Beispiel beträgt 20.

```
SELECT SQRT(20/PI()) AS radius;

+--------------------+
|      radius        |
+--------------------+
| 2.5231325220201604 |
+--------------------+
```

In den folgenden Beispielen werden Daten aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Quadratwurzel für COMMISSION-Werte aus der Tabelle SALES zurückgegeben. Die COMMISSION-Spalte ist eine DECIMAL-Spalte. Dieses Beispiel zeigt, wie Sie die Funktion in einer Abfrage mit komplexerer bedingter Logik verwenden können.

```
SELECT SQRT(commission)
FROM sales WHERE salesid < 10 ORDER BY salesid;

+--------------------+
|        sqrt        |
+--------------------+
| 10.449880382090505 |
| 3.3763886032268267 |
|  7.245688373094719 |
|  5.123475382979799 |
|  4.806245936279167 |
|  7.687652437513028 |
| 10.871982339941507 |
| 5.4359911699707535 |
|   9.41541289588513 |
+--------------------+
```

Verwenden Sie das folgende Beispiel, um die gerundete Quadratwurzel für denselben Satz von COMMISSION-Werten zurückzugeben.

```
SELECT ROUND(SQRT(commission))
FROM sales WHERE salesid < 10 ORDER BY salesid;

+-------+
| round |
+-------+
```

```
|    10 |
|     3 |
|     7 |
|     5 |
|     5 |
|     8 |
|    11 |
|     5 |
|     9 |
+-------+
```

Die Funktion TAN

TAN ist eine trigonometrische Funktion, die den Tangens einer Zahl zurückgibt. Das Eingabeargument ist eine Zahl (im Bogenmaß).

Syntax

```
TAN(number)
```

Argument

number (Zahl

Eine DOUBLE PRECISION-Zahl.

Rückgabetyp

DOUBLE PRECISION

Beispiele

Verwenden Sie das folgende Beispiel, um den Tangens von null zurückzugeben.

```
SELECT TAN(0);

+-----+
| tan |
+-----+
|   0 |
+-----+
```

Die Funktion TRUNC

Die TRUNC-Funktion verkürzt Zahlen auf die vorherige Ganz- oder Dezimalzahlen.

Die TRUNC-Funktion kann optional ein zweites Argument als `INTEGER` umfassen, um die Anzahl der Dezimalstellen für die Rundung in beide Richtungen anzugeben. Wenn Sie das zweite Argument nicht angeben, wird die Funktion auf die nächste ganze Zahl gerundet. Wenn das zweite Argument Ganzzahl angegeben wurde, wird die Funktion auf die nächste Zahl mit einer Genauigkeit von Ganzzahl Dezimalstellen gerundet.

Diese Funktion kann auch einen `TIMESTAMP` kürzen und ein `DATE` zurückgeben. Weitere Informationen finden Sie unter Die Funktion TRUNC.

Syntax

```
TRUNC(number [ , integer ])
```

Argumente

number (Zahl

Eine Zahl oder ein Ausdruck, der zu einer Zahl ausgewertet wird. Es kann sich um den Typ `DECIMAL`, `FLOAT8` oder `SUPER` handeln. Amazon Redshift kann weitere Datentypen gemäß den impliziten Konvertierungsregeln konvertieren.

integer

(Optional) Eine `INTEGER`, die die Zahl der Dezimalstellen der Präzision in beide Richtungen anzeigt. Wenn keine Ganzzahl angegeben wird, wird die Zahl zu einer ganzen Zahl abgeschnitten. Wenn eine Ganzzahl angegeben wird, wird die Zahl an der angegebenen Dezimalstelle abgeschnitten. Dies wird für den `SUPER`-Datentyp nicht unterstützt.

Rückgabetyp

TRUNC gibt denselben numerischen Datentyp wie die Eingabezahl zurück.

Wenn die Eingabe den Typ `SUPER` hat, behält die Ausgabe den gleichen dynamischen Typ wie die Eingabe bei, während der statische Typ weiterhin den Typ `SUPER` hat. Wenn der dynamische Typ von `SUPER` keine Zahl ist, gibt Amazon Redshift `NULL` zurück.

Beispiele

Einige der folgenden Beispiele verwenden die Beispieldatenbank TICKIT. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Decke der Provision, die für eine bestimmte Verkaufstransaktion gezahlt wird, zu kürzen.

```
SELECT commission, TRUNC(commission)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 |   111 |
+------------+-------+
```

Verwenden Sie das folgende Beispiel, um denselben Provisionswert an der ersten Dezimalstelle zu kürzen.

```
SELECT commission, TRUNC(commission,1)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 | 111.1 |
+------------+-------+
```

Verwenden Sie das folgende Beispiel, um die Provision mit einem negativen Wert für das zweite Argument zu kürzen. Beachten Sie, dass `111.15` auf `110` abgerundet wird.

```
SELECT commission, TRUNC(commission,-1)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 |   110 |
+------------+-------+
```

Objektfunktionen

Im Folgenden sind die SQL-Objektfunktionen aufgeführt, die Amazon Redshift zum Erstellen von Objekten des SUPER–Typs unterstützt:

Themen

Funktion LOWER_ATTRIBUTE_NAMES

Konvertiert alle zutreffenden Attributnamen in einem SUPER-Wert in Kleinbuchstaben und verwendet dabei dieselbe Konvertierungsroutine wie bei. Die Funktion LOWER LOWER_ATTRIBUTE_NAMES unterstützt UTF-8-Multibyte-Zeichen, bis zu einem Maximum von vier Byte pro Zeichen.

Um SUPER-Attributnamen in Großbuchstaben umzuwandeln, verwenden Sie den. Funktion UPPER_ATTRIBUTE_NAMES

Syntax

```
LOWER_ATTRIBUTE_NAMES(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck.

Rückgabetyp

SUPER

Nutzungshinweise

In Amazon Redshift wird bei Spaltenbezeichnern traditionell nicht zwischen Groß- und Kleinschreibung unterschieden und sie werden in Kleinbuchstaben umgewandelt. Wenn Sie Daten

aus Datenformaten wie JSON aufnehmen, bei denen die Groß- und Kleinschreibung beachtet wird, können die Daten Attributnamen mit gemischter Groß- und Kleinschreibung enthalten.

Betrachten Sie das folgende Beispiel.

```
CREATE TABLE t1 (s) AS SELECT JSON_PARSE('{"AttributeName": "Value"}');


SELECT s.AttributeName FROM t1;

attributename
-------------
NULL


SELECT s."AttributeName" FROM t1;

attributename
-------------
NULL
```

Amazon Redshift gibt für beide Abfragen NULL zurück. Verwenden Sie für Abfragen LOWER_ATTRIBUTE_NAMES`AttributeName`, um die Attributnamen der Daten in Kleinbuchstaben umzuwandeln. Betrachten Sie das folgende Beispiel.

```
CREATE TABLE t2 (s) AS SELECT LOWER_ATTRIBUTE_NAMES(s) FROM t1;


SELECT s.attributename FROM t2;

attributename
-------------
"Value"


SELECT s.AttributeName FROM t2;

attributename
-------------
"Value"


SELECT s."attributename" FROM t2;
```

```
attributename
-------------
"Value"


SELECT s."AttributeName" FROM t2;

attributename
-------------
"Value"
```

Eine verwandte Option für die Arbeit mit Objektattributnamen mit gemischter Groß- und Kleinschreibung ist die `enable_case_sensitive_super_attribute` Konfigurationsoption, mit der Amazon Redshift Groß- und Kleinschreibung in SUPER-Attributnamen erkennt. Dies kann eine alternative Lösung zur Verwendung von LOWER_ATTRIBUTE_NAMES sein. Weitere Informationen zu finden Sie unter. `enable_case_sensitive_super_attribute` enable_case_sensitive_super_attribute

Beispiele

SUPER-Attributnamen in Kleinbuchstaben konvertieren

Im folgenden Beispiel werden LOWER_ATTRIBUTE_NAMES verwendet, um die Attributnamen aller SUPER-Werte in einer Tabelle zu konvertieren.

```
-- Create a table and insert several SUPER values.
CREATE TABLE t (i INT, s SUPER);

INSERT INTO t VALUES
  (1, NULL),
  (2, 'A'::SUPER),
  (3, JSON_PARSE('{"AttributeName": "B"}')),
  (4, JSON_PARSE(
     '[{"Subobject": {"C": "C"},
       "Subarray": [{"D": "D"}, "E"]
      }]'));

-- Convert all attribute names to lowercase.
UPDATE t SET s = LOWER_ATTRIBUTE_NAMES(s);

SELECT i, s FROM t ORDER BY i;
```

```
 i |                         s
---+-------------------------------------------------
 1 | NULL
 2 | "A"
 3 | {"attributename":"B"}
 4 | [{"subobject":{"c":"C"},"subarray":[{"d":"D"}, "E"]}]
```

Beobachten Sie, wie LOWER_ATTRIBUTE_NAMES funktioniert.

- NULL-Werte und skalare SUPER-Werte wie sind unverändert. `"A"`
- In einem SUPER-Objekt werden alle Attributnamen in Kleinbuchstaben geändert, Attributwerte wie z. B. `"B"` bleiben jedoch unverändert.
- LOWER_ATTRIBUTE_NAMES gilt rekursiv für jedes SUPER-Objekt, das in einem SUPER-Array oder in einem anderen Objekt verschachtelt ist.

Verwendung von LOWER_ATTRIBUTE_NAMES für ein SUPER-Objekt mit doppelten Attributnamen

Wenn ein SUPER-Objekt Attribute enthält, deren Namen sich nur in ihrer Groß- und Kleinschreibung unterscheiden, gibt LOWER_ATTRIBUTE_NAMES einen Fehler aus. Betrachten Sie das folgende Beispiel.

```
SELECT LOWER_ATTRIBUTE_NAMES(JSON_PARSE('{"A": "A", "a": "a"}'));

error:   Invalid input
code:    8001
context: SUPER value has duplicate attributes after case conversion.
```

OBJECT-Funktion

Erstellt ein Array des SUPER-Datentyps.

Syntax

```
OBJECT ( [ key1, value1 ], [ key2, value2 ...] )
```

Argumente

key1, key2

Ausdrücke, die zu VARCHAR-Typzeichenfolgen ausgewertet werden.

value1, value2

Ausdrücke eines Amazon-Redshift-Datentyps außer Datums- und Uhrzeittypen, da Amazon Redshift die Datums- und Uhrzeittypen nicht in den SUPER-Datentyp umwandelt. Weitere Informationen zu Datums- und Uhrzeittypen finden Sie unter Datum-/Uhrzeittypen.

`value`-Ausdrücke in einem Objekt müssen nicht denselben Datentyp haben.

Rückgabetyp

SUPER

Beispiel

```
-- Creates an empty object.
select object();

object
--------
{}
(1 row)

-- Creates objects with different keys and values.
select object('a', 1, 'b', true, 'c', 3.14);

object
---------------------------
{"a":1,"b":true,"c":3.14}
(1 row)

select object('a', object('aa', 1), 'b', array(2,3), 'c', json_parse('{}'));

object
---------------------------------
{"a":{"aa":1},"b":[2,3],"c":{}}
(1 row)

-- Creates objects using columns from a table.
create table bar (k varchar, v super);
insert into bar values ('k1', json_parse('[1]')), ('k2', json_parse('{}'));
select object(k, v) from bar;

object
```

```
------------
{"k1":[1]}
{"k2":{}}
(2 rows)

-- Errors out because DATE type values can't be converted to SUPER type.
select object('k', '2008-12-31'::date);

ERROR:  OBJECT could not convert type date to super
```

OBJECT_TRANSFORM-Funktion

Transformiert ein SUPER-Objekt.

Syntax

```
OBJECT_TRANSFORM(
  input
  [KEEP path1, ...]
  [SET
    path1, value1,
    ...,  ...
  ]
)
```

Argumente

input

Ein Ausdruck, der in ein Objekt vom Typ SUPER aufgelöst wird.

KEEP

Alle in dieser Klausel angegebenen Pfad-Werte werden beibehalten und auf das Ausgabeobjekt übertragen.

Diese Klausel ist optional.

path1, path2, ...

Konstante Zeichenfolgenliterale im Format von Pfadkomponenten in doppelten Anführungszeichen, die durch Punkte getrennt sind. Beispielsweise ist `'"a"."b"."c"'` ein gültiger Pfad-Wert. Dies gilt für den Pfadparameter sowohl in der KEEP- als auch in der SET-Klausel.

SET

Pfad- und Wert-Paare, um einen bestehenden Pfad zu ändern oder einen neuen Pfad hinzuzufügen und den Wert dieses Pfads im Ausgabeobjekt festzulegen.

Diese Klausel ist optional.

value1, value2, ...

Ausdrücke, die in Werte vom Typ SUPER aufgelöst werden. Beachten Sie, dass numerische, Text- und boolesche Typen in SUPER aufgelöst werden können.

Rückgabetyp

SUPER

Nutzungshinweise

OBJECT_TRANSFORM gibt ein Objekt vom Typ SUPER zurück, das die Pfadwerte aus der Eingabe enthält, die in KEEP angegeben wurden, und die Pfad-Wert-Paare, die in SET angegeben wurden.

Wenn sowohl KEEP als auch SET leer sind, gibt OBJECT_TRANSFORM Eingabe zurück.

Wenn es sich bei Eingabe nicht um ein Objekt vom Typ SUPER handelt, gibt OBJECT_TRANSFORM unabhängig von KEEP- oder SET-Werten Eingabe zurück.

Beispiel

Im folgenden Beispiel wird ein SUPER-Objekt in ein anderes SUPER-Objekt umgewandelt.

```
CREATE TABLE employees (
    col_person SUPER
);

INSERT INTO employees
VALUES
    (
        json_parse('
            {
                "name": {
                    "first": "John",
                    "last": "Doe"
                },
                "age": 25,
```

```
                "ssn": "111-22-3333",
                "company": "Company Inc.",
                "country": "U.S."
            }
        ')
    ),
    (
        json_parse('
            {
                "name": {
                    "first": "Jane",
                    "last": "Appleseed"
                },
                "age": 34,
                "ssn": "444-55-7777",
                "company": "Organization Org.",
                "country": "Ukraine"
            }
        ')
    )
;

SELECT
    OBJECT_TRANSFORM(
        col_person
        KEEP
            '"name"."first"',
            '"age"',
            '"company"',
            '"country"'
        SET
            '"name"."first"', UPPER(col_person.name.first::TEXT),
            '"age"', col_person.age + 5,
            '"company"', 'Amazon'
    ) AS col_person_transformed
FROM employees;

--This result is formatted for ease of reading.
                  col_person_transformed
-------------------------------------------------------------
{
    "name": {
        "first": "JOHN"
    },
```

```
    "age": 30,
    "company": "Amazon",
    "country": "U.S."
}
{
    "name": {
        "first": "JANE"
    },
    "age": 39,
    "company": "Amazon",
    "country": "Ukraine"
}
```

Funktion UPPER_ATTRIBUTE_NAMES

Konvertiert alle zutreffenden Attributnamen in einem SUPER-Wert in Großbuchstaben und verwendet dabei dieselbe Routine zur Konvertierung von Groß- und Kleinschreibung wie bei. Die Funktion UPPER UPPER_ATTRIBUTE_NAMES unterstützt UTF-8-Multibyte-Zeichen, bis zu einem Maximum von vier Byte pro Zeichen.

Um SUPER-Attributnamen in Kleinbuchstaben umzuwandeln, verwenden Sie den. Funktion LOWER_ATTRIBUTE_NAMES

Syntax

```
UPPER_ATTRIBUTE_NAMES(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck.

Rückgabetyp

SUPER

Beispiele

SUPER-Attributnamen in Großbuchstaben konvertieren

Im folgenden Beispiel werden UPPER_ATTRIBUTE_NAMES verwendet, um die Attributnamen aller SUPER-Werte in einer Tabelle zu konvertieren.

```
-- Create a table and insert several SUPER values.
CREATE TABLE t (i INT, s SUPER);

INSERT INTO t VALUES
  (1, NULL),
  (2, 'a'::SUPER),
  (3, JSON_PARSE('{"AttributeName": "b"}')),
  (4, JSON_PARSE(
     '[{"Subobject": {"c": "c"},
       "Subarray": [{"d": "d"}, "e"]
      }]'));

-- Convert all attribute names to uppercase.
UPDATE t SET s = UPPER_ATTRIBUTE_NAMES(s);

SELECT i, s FROM t ORDER BY i;

 i |                         s
---+---------------------------------------------------
 1 | NULL
 2 | "a"
 3 | {"ATTRIBUTENAME":"B"}
 4 | [{"SUBOBJECT":{"C":"c"},"SUBARRAY":[{"D":"d"}, "e"]}]
```

Beobachten Sie, wie UPPER_ATTRIBUTE_NAMES funktioniert.

- NULL-Werte und skalare SUPER-Werte wie sind unverändert. `"a"`
- In einem SUPER-Objekt werden alle Attributnamen in Großbuchstaben geändert, Attributwerte wie z. B. `"b"` bleiben jedoch unverändert.
- UPPER_ATTRIBUTE_NAMES gilt rekursiv für jedes SUPER-Objekt, das in einem SUPER-Array oder in einem anderen Objekt verschachtelt ist.

Verwendung von UPPER_ATTRIBUTE_NAMES für ein SUPER-Objekt mit doppelten Attributnamen

Wenn ein SUPER-Objekt Attribute enthält, deren Namen sich nur in ihrer Groß- und Kleinschreibung unterscheiden, gibt UPPER_ATTRIBUTE_NAMES einen Fehler aus. Betrachten Sie das folgende Beispiel.

```
SELECT UPPER_ATTRIBUTE_NAMES(JSON_PARSE('{"A": "A", "a": "a"}'));

error:   Invalid input
code:    8001
context: SUPER value has duplicate attributes after case conversion.
```

Geofunktionen

Die Beziehungen zwischen Geometrieobjekten basieren auf dem Dimensionally Extended nine-Intersection Model (DE-9IM). Dieses Modell definiert Prädikate wie Equals, Contains und Covers. Weitere Informationen zur Definition von räumlichen -Beziehungen finden Sie unter DE-9IM in Wikipedia.

Weitere Informationen zur Verwendung raumbezogener Daten mit Amazon Redshift finden Sie unter Abfrage von Geodaten in Amazon Redshift.

Amazon Redshift bietet Geofunktionen für die Datentypen `GEOMETRY` und `GEOGRAPHY`. Im Folgenden sind die Funktionen aufgeführt, die den `GEOGRAPHY`-Datentyp unterstützen:

- ST_Area
- ST_AsEWKT
- ST_ JSON AsGeo
- ST_ EWKB AsHex
- ST_ WKB AsHex
- ST_ AsText
- ST_Distance
- ST_ GeogFromText
- GeogFromST_ WKB
- ST_Length
- ST_NPoints
- ST_Perimeter

Im Folgenden sind alle Geofunktionen aufgeführt, die von Amazon Redshift unterstützt werden.

Themen

- AddBBox
- DropBBox
- GeometryType
- H3_ FromLongLat
- H3_ FromPoint
- H3_Polyfill
- ST_ AddPoint
- ST_Angle
- ST_Area
- ST_ AsBinary
- ST_AsEWKB
- ST_AsEWKT
- ST_ JSON AsGeo
- ST_ WKB AsHex
- ST_ EWKB AsHex
- ST_ AsText
- ST_Azimuth
- ST_Boundary
- ST_Buffer
- ST_Centroid
- ST_Collect
- ST_Contains
- ST_ ContainsProperly
- ST_ ConvexHull
- ST_ CoveredBy
- ST_Covers
- ST_Crosses
- ST_Dimension
- ST_Disjoint
- ST_Distance

- ST_ DistanceSphere
- ST_DWithin
- ST_ EndPoint
- ST_Envelope
- ST_Equals
- ST_ ExteriorRing
- ST_Force2D
- ST_Force3D
- ST_Force3DM
- ST_Force3DZ
- ST_Force4D
- ST_ GeoHash
- ST_ GeogFromText
- GeogFromST_ WKB
- ST_GeometryN
- ST_ GeometryType
- ST_ EWKB GeomFrom
- ST_ GeomFrom EWKT
- ST_ GeomFromGeoHash
- GeomFromGeoST_ JSON
- ST_ GeomFromGeoSquare
- ST_ GeomFromText
- GeomFromST_ WKB
- ST_ GeoSquare
- ST_ N InteriorRing
- ST_Intersects
- ST_Intersection
- ST_ IsPolygon CCW
- IsPolygonST_ CW
- ST_ IsClosed

- ST_ IsCollection
- ST_ IsEmpty
- ST_ IsRing
- ST_ IsSimple
- ST_ IsValid
- ST_Length
- ST_ LengthSphere
- ST_Length2D
- ST_ LineFromMultiPoint
- ST_ LineInterpolatePoint
- ST_M
- ST_ MakeEnvelope
- ST_ MakeLine
- ST_ MakePoint
- ST_ MakePolygon
- ST_ MemSize
- ST_MMax
- ST_MMin
- ST_Multi
- ST_NDims
- ST_NPoints
- ST_NRings
- ST_ NumGeometries
- ST_ NumInteriorRings
- ST_ NumPoints
- ST_Perimeter
- ST_Perimeter2D
- ST_Point
- ST_PointN
- ST_Points

- ST_Polygon
- ST_ RemovePoint
- ST_Reverse
- ST_ SetPoint
- ST_SetSRID
- ST_Simplify
- ST_SRID
- ST_ StartPoint
- ST_Touches
- ST_Transform
- ST_Union
- ST_Within
- ST_X
- ST_XMax
- ST_XMin
- ST_Y
- ST_YMax
- ST_YMin
- ST_Z
- ST_ZMax
- ST_ZMin
- SupportsBBox

AddBBox

AddBBox gibt eine Kopie der Eingabegeometrie zurück, die die Kodierung mit einem vorberechneten Begrenzungsrahmen unterstützt. Weitere Informationen zur Unterstützung von Bounding Boxes finden Sie unter Begrenzungsrahmen.

Syntax

```
AddBBox(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt eine Kopie einer Polygon-Eingabegeometrie zurück, die die Kodierung mit einem Begrenzungsrahmen unterstützt.

```
SELECT ST_AsText(AddBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
 st_astext
----------
 POLYGON((0 0,1 0,0 1,0 0))
```

DropBBox

DropBBox gibt eine Kopie der Eingabegeometrie zurück, die keine Kodierung mit einem vorberechneten Begrenzungsrahmen unterstützt. Weitere Informationen zur Unterstützung von Bounding Boxes finden Sie unter Begrenzungsrahmen.

Syntax

```
DropBBox(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt eine Kopie einer Polygon-Eingabegeometrie zurück, die keine Kodierung mit einem Begrenzungsrahmen unterstützt.

```
SELECT ST_AsText(DropBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
 st_astext
----------
 POLYGON((0 0,1 0,0 1,0 0))
```

GeometryType

GeometryType gibt den Subtyp einer Eingabegeometrie als Zeichenfolge zurück.

Syntax

```
GeometryType(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

VARCHAR stellt den Subtyp von geom dar.

Wenn geom null ist, wird null zurückgegeben.

Die zurückgegebenen Werte lauten wie folgt.

Zurückgegebener Zeichenfolgenwert für 2D-, 3DZ-, 4D-Geometrien	Zurückgegebener Zeichenfolgenwert für 3DM-Geometrien	Geometrie-Subtyp
POINT	POINTM	Wird zurückgegeben, wenn geom ein POINT-Subtyp ist.
LINESTRING	LINESTRINGM	Wird zurückgegeben, wenn geom ein LINESTRING -Subtyp ist.
POLYGON	POLYGONM	Wird zurückgegeben, wenn geom ein POLYGON-Subtyp ist.
MULTIPOINT	MULTIPOINTM	Wird zurückgegeben, wenn geom ein MULTIPOINT -Subtyp ist.
MULTILINESTRING	MULTILINESTRINGM	Wird zurückgegeben, wenn geom ein MULTILINESTRING -Subtyp ist.
MULTIPOLYGON	MULTIPOLYGONM	Wird zurückgegeben, wenn geom ein MULTIPOLYGON -Subtyp ist.
GEOMETRYCOLLECTION	GEOMETRYCOLLECTIONM	Wird zurückgegeben, wenn geom ein GEOMETRYCOLLECTION -Subtyp ist.

Beispiele

Das folgende SQL konvertiert eine WKT-Darstellung eines Polygons und gibt den Subtyp GEOMETRY als Linestring zurück.

```
SELECT GeometryType(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
geometrytype
-------------
 POLYGON
```

H3_ FromLongLat

H3_ FromLongLat gibt die entsprechende H3-Zellen-ID aus einem eingegebenen Längengrad, Breitengrad und Auflösung zurück. Weitere Informationen zur H3-Indizierung finden Sie in H3.

Syntax

```
H3_FromLongLat(longitude, lattitude, resolution)
```

Argumente

Längengrad

Ein Wert vom Datentyp `DOUBLE PRECISION` oder ein Ausdruck, der zu einem `DOUBLE PRECISION`-Typ ausgewertet wird.

Breitengrand

Ein Wert vom Datentyp `DOUBLE PRECISION` oder ein Ausdruck, der zu einem `DOUBLE PRECISION`-Typ ausgewertet wird.

Auflösung

Ein Wert vom Datentyp `INTEGER` oder ein Ausdruck, der zu einem `INTEGER`-Typ ausgewertet wird. Der Wert steht für die Auflösung des H3-Rastersystems. Der Wert muss eine ganze Zahl zwischen 0 und einschließlich 15 sein. Dabei ist `0` die größte und `15` die feinste Auflösung.

Rückgabetyp

`BIGINT` – steht für die H3-Zellen-ID.

Wenn die Auflösung außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die H3-Zellen-ID anhand von Längengrad `0`, Breitengrad `0` und Auflösung `10` zurück.

```
SELECT H3_FromLongLat(0, 0, 10);
```

```
 h3_fromlonglat
-------------------
 623560421467684863
```

H3_ FromPoint

H3_ FromPoint gibt die entsprechende H3-Zellen-ID aus einem Eingabegeometriepunkt und einer Auflösung zurück. Weitere Informationen zur H3-Indizierung finden Sie in H3.

Syntax

```
H3_FromPoint(geom, resolution)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. geom muss ein POINT sein.

Auflösung

Ein Wert vom Datentyp INTEGER oder ein Ausdruck, der zu einem INTEGER-Typ ausgewertet wird. Der Wert steht für die Auflösung des H3-Rastersystems. Der Wert muss eine ganze Zahl zwischen 0 und einschließlich 15 sein. Dabei ist 0 die größte und 15 die feinste Auflösung.

Rückgabetyp

BIGINT – steht für die H3-Zellen-ID.

Wenn geom kein POINT ist, wird ein Fehler zurückgegeben.

Wenn die Auflösung außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben.

Wenn geom leer ist, wird NULL zurückgegeben.

Beispiele

Das folgende SQL gibt die H3-Zellen-ID von Punkt 0,0 und Auflösung 10 zurück.

```
SELECT H3_FromPoint(ST_GeomFromText('POINT(0 0)'), 10);
```

```
 h3_frompoint
--------------------
 623560421467684863
```

H3_Polyfill

H3_Polyfill gibt die entsprechenden H3-Zellen-IDs zurück, die den Sechsecken und Fünfecken entsprechen, die im Eingabepolygon der angegebenen Auflösung enthalten sind. Weitere Informationen zur H3-Indizierung finden Sie in H3.

Syntax

```
H3_Polyfill(geom, resolution)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. geom muss ein POLYGON sein.

Auflösung

Ein Wert vom Datentyp INTEGER oder ein Ausdruck, der zu einem INTEGER-Typ ausgewertet wird. Der Wert steht für die Auflösung des H3-Rastersystems. Der Wert muss eine ganze Zahl zwischen 0 und einschließlich 15 sein. Dabei ist 0 die größte und 15 die feinste Auflösung.

Rückgabetyp

SUPER – steht für eine Liste von H3-Zellen-IDs.

Wenn geom kein POLYGON ist, wird ein Fehler zurückgegeben.

Wenn die Auflösung außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben.

Wenn geom leer ist, wird NULL zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt ein Array vom Datentyp SUPER mit H3-Zellen-IDs aus einem Polygon und einer Auflösung 4 zurück.

```
SELECT H3_Polyfill(ST_GeomFromText('POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))'), 4);
```

```
 h3_polyfill
-------------------------------------------------------------------------------------------------
 [596538848238895103,596538805289222143,596538856828829695,596538813879156735,5965379205259591
```

ST_ AddPoint

ST_ AddPoint gibt eine Linienkettengeometrie zurück, die mit der Eingabegeometrie identisch ist, wobei ein Punkt hinzugefügt wurde. Wenn ein Index angegeben wird, wird der Punkt an der Indexposition hinzugefügt. Wenn der Index -1 ist oder nicht angegeben wird, wird der Punkt an den Linestring angehängt.

Der Index ist null-basiert. Der räumliche Referenzsystem-Bezeichner (SRID) des Ergebnisses entspricht dem der Eingabegeometrie.

Die Dimension der zurückgegebenen Geometrie entspricht der des geom1-Werts. Wenn geom1 und geom2 unterschiedliche Dimensionen haben, wird geom2 auf die Dimension von geom1 projiziert.

Syntax

```
ST_AddPoint(geom1, geom2)
```

```
ST_AddPoint(geom1, geom2, index)
```

Argumente

geom1

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `LINESTRING`.

geom2

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `POINT`. Der Punkt kann der leere Punkt sein.

index

Ein Wert des Datentyps `INTEGER`. der die Position eines null-basierten Indexes darstellt.

Rückgabetyp

`GEOMETRY`

Wenn geom1, geom2 oder index null ist, wird null zurückgegeben.

Wenn geom2 der leere Punkt ist, wird eine Kopie von geom1 zurückgegeben.

Wenn geom1 kein `LINESTRING` ist, wird ein Fehler zurückgegeben.

Wenn geom2 kein `POINT` ist, wird ein Fehler zurückgegeben.

Wenn der Index außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben. Gültige Werte für die Indexposition sind -1 oder ein Wert zwischen 0 und `ST_NumPoints(geom1)`.

Beispiele

Die folgende SQL-Anweisung fügt einem Linestring einen Punkt hinzu, um ihn zu einem geschlossenen Linestring zu machen.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_AddPoint(g, ST_StartPoint(g))) FROM tmp;
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5,0 5,0 0)
```

Die folgende SQL-Anweisung fügt einen Punkt zu einer bestimmten Position in einem Linestring hinzu.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_AddPoint(g, ST_SetSRID(ST_Point(5, 10), 4326), 3)) FROM tmp;
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 10,5 5,0 5)
```

ST_Angle

ST_Angle gibt den Winkel im Bogenmaß zwischen Punkten zurück, gemessen im Uhrzeigersinn. Und zwar wie folgt:

- Wenn drei Punkte eingegeben werden, wird der zurückgegebene Winkel P1-P2-P3 so gemessen, als ob der Winkel durch Drehen von P1 nach P3 um P2 im Uhrzeigersinn ermittelt wurde.
- Wenn vier Punkte eingegeben werden, wird der zurückgegebene Winkel im Uhrzeigersinn zurückgegeben, der durch die gerichteten Linien P1-P2 und P3-P4 gebildet wird. Wenn es sich bei der Eingabe um einen fehlerhaften Fall handelt (d. h. P1 entspricht P2 oder P3 entspricht P4), wird null zurückgegeben.

Der Rückgabewert wird in Radianten ausgedrückt und liegt im Bereich (0, 2π).

ST_Boundary wird auf den 2D-Projektionen der Eingabegeometrien ausgeführt.

Syntax

```
ST_Angle(geom1, geom2, geom3)
```

```
ST_Angle(geom1, geom2, geom3, geom4)
```

Argumente

geom1

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `POINT`.

geom2

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `POINT`.

geom3

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `POINT`.

geom4

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Der Subtyp muss sein POINT.

Rückgabetyp

DOUBLE PRECISION.

Wenn geom1 geom2, entspricht oder geom2 geom3, entspricht, wird null zurückgegeben.

Wenn geom1, geom2, geom3 oder geom4 null ist, wird null zurückgegeben.

Wenn geom1, geom2, geom3 oder geom4 der leere Punkt ist, wird ein Fehler zurückgegeben.

Wenn geom1, geom2, geom3 und geom4 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt den Winkel zurück, der in Grad von drei Eingabepunkten konvertiert wurde.

```
SELECT ST_Angle(ST_Point(1,1), ST_Point(0,0), ST_Point(1,0)) / Pi() * 180.0 AS angle;
```

```
 angle
---------------
    45
```

Die folgende SQL-Anweisung gibt den Winkel zurück, der in Grad von vier Eingabepunkten konvertiert wurde.

```
SELECT ST_Angle(ST_Point(1,1), ST_Point(0,0), ST_Point(1,0), ST_Point(2,0)) / Pi() *
 180.0 AS angle;
```

```
 angle
---------------
   225
```

ST_Area

Für eine Eingabegeometrie gibt ST_Area die kartesischen Fläche der 2D-Projektion zurück. Die Flächeneinheiten entsprechen den Einheiten, in denen die Koordinaten der Eingabegeometrie ausgedrückt werden. Bei Punkten, Linestrings, Multipoints und Multi-Linestrings gibt die Funktion 0 zurück. Bei Geometriesammlungen wird die Flächensumme der Geometrien in der Sammlung zurückgegeben.

Für eine Eingabegeografie gibt ST_Area die geodätische Fläche der 2D-Projektion einer Eingabeflächengeografie zurück, die auf dem vom SRID festgelegten Sphäroiden berechnet wurde. Die Länge wird in Quadratmeter angeben. Die Funktion gibt Null (0) für Punkte, Multipoints und lineare Geografien zurück. Wenn es sich bei der Eingabe um eine Geometriesammlung handelt, gibt die Funktion die Flächensumme der Flächengeografien in der Sammlung zurück.

Syntax

```
ST_Area(geo)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION

Wenn geo null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt den kartesischen Bereich eines Multipolygons zurück.

```
SELECT ST_Area(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20 10,10
 0)))'));
```

```
 st_area
---------
```

```
     100
```

Die folgende SQL-Anweisung gibt die Fläche eines Polygons in einer Geografie zurück.

```
SELECT ST_Area(ST_GeogFromText('polygon((34 35, 28 30, 25 34, 34 35))'));
```

```
     st_area
------------------
 201824655743.383
```

Die folgende SQL-Anweisung gibt für eine lineare Geografie null zurück.

```
SELECT ST_Area(ST_GeogFromText('multipoint(0 0, 1 1, -21.32 121.2)'));
```

```
 st_area
---------
       0
```

ST_ AsBinary

ST_ AsBinary gibt die hexadezimale, wohlbekannte Binärdarstellung (WKB) einer Eingabegeometrie zurück. Für 3DZ-, 3DM- und 4D-Geometrien AsBinary verwendet ST_ den Standardwert des Open Geospatial Consortium (OGC) für den Geometrietyp.

Syntax

```
ST_AsBinary(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`VARBYTE`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die hexadezimale WKB-Darstellung eines Polygons zurück.

```
SELECT ST_AsBinary(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_asbinary
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

ST_AsEWKB

ST_AsEWKB gibt die EWKB-Darstellung (Extended Well-Known Binary) einer Eingangsgeometrie zurück. Für 3DZ-, 3DM- und 4D-Geometrien verwendet ST_AsEWKB den OGC-Standardwert (Open Geospatial Consortium) für den Geometrietyp.

Syntax

```
ST_AsEWKB(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`VARBYTE`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die hexadezimale EWKB-Darstellung eines Polygons zurück.

```
SELECT ST_AsEWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_asewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F0
```

ST_AsEWKT

ST_AsEWKT gibt die EWKT-Darstellung (Extended Well-Known Binary) einer Eingabegeometrie oder -geografie zurück. Bei 3DZ-, 3DM- und 4D-Geometrien fügt ST_ASEwKT Z, M oder ZM an den WKT-Wert für den Geometrietyp an.

Syntax

```
ST_AsEWKT(geo)
```

```
ST_AsEWKT(geo, precision)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

precision

Ein Wert vom Datentyp INTEGER. Für Geometrien werden die Koordinaten von geo mit der angegebenen Genauigkeit 1–20 angezeigt. Wenn precision nicht angegeben ist, ist der Standardwert 15. Für Geografien werden die Koordinaten von geo mit der angegebenen Genauigkeit angezeigt. Wenn precision nicht angegeben ist, ist der Standardwert 15.

Rückgabetyp

VARCHAR

Wenn geo null ist, wird null zurückgegeben.

Wenn Präzision null ist, wird null zurückgegeben.

Wenn das Ergebnis größer als 64-KB VARCHAR ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die EWKT-Darstellung eines Linestrings zurück.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326));
```

```
st_asewkt
-------------------------------
 SRID=4326;LINESTRING(3.14159265358979 -6.28318530717959,2.71828182845905
 -1.41421356237309)
```

Das folgende SQL gibt die EWKT-Darstellung eines Linestrings zurück. Die Koordinaten der Geometrien werden mit sechs Stellen Genauigkeit angezeigt.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326), 6);
```

```
st_asewkt
-------------------------------
 SRID=4326;LINESTRING(3.14159 -6.28319,2.71828 -1.41421)
```

Das folgende SQL gibt die EWKT-Darstellung einer Geografie zurück.

```
SELECT ST_AsEWKT(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
                 st_asewkt
---------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

ST_ JSON AsGeo

ST_ AsGeo JSON gibt die GeoJSON-Repräsentation einer Eingabegeometrie oder -geografie zurück. Weitere Informationen zu GeoJSON finden Sie unter GeoJSON bei Wikipedia.

Bei 3DZ- und 4D-Geometrien ist die Ausgabegeometrie eine 3DZ-Projektion der 3DZ- oder 4D-Eingabegeometrie. Das heißt, dass in der Ausgabe die Koordinaten x, y und z vorhanden sind. Bei 3DM-Geometrien ist die Ausgabegeometrie eine 2D-Projektion der 3DM-Eingabegeometrie. Das heißt, dass in der Ausgabe nur die Koordinaten x und y vorhanden sind.

Für Eingabegeografien gibt ST_ AsGeo JSON die GeoJSON-Repräsentation einer Eingabegeografie zurück. Die Koordinaten der Geografie werden mit der angegebenen Genauigkeit angezeigt.

Syntax

```
ST_AsGeoJSON(geo)
```

```
ST_AsGeoJSON(geo, precision)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

precision

Ein Wert vom Datentyp INTEGER. Für Geometrien werden die Koordinaten von geo mit der angegebenen Genauigkeit 1–20 angezeigt. Wenn precision nicht angegeben ist, ist der Standardwert 15. Für Geografien werden die Koordinaten von geo mit der angegebenen Genauigkeit angezeigt. Wenn precision nicht angegeben ist, ist der Standardwert 15.

Rückgabetyp

VARCHAR

Wenn geo null ist, wird null zurückgegeben.

Wenn Präzision null ist, wird null zurückgegeben.

Wenn das Ergebnis größer als 64-KB VARCHAR ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die GeoJSON-Darstellung eines Linestrings zurück.

```
SELECT ST_AsGeoJSON(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)'));
```

```
st_asgeojson
-----------------------------------------------------------------------------------------------
 {"type":"LineString","coordinates":[[3.14159265358979,-6.28318530717959],
[2.71828182845905,-1.414213562373095]]}
```

Das folgende SQL gibt die GeoJSON-Darstellung eines Linestrings zurück. Die Koordinaten der Geometrien werden mit sechs Stellen Genauigkeit angezeigt.

```
SELECT ST_AsGeoJSON(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)'), 6);
```

```
st_asgeojson
-------------------------------------------------------------------------
  {"type":"LineString","coordinates":[[3.14159,-6.28319],[2.71828,-1.41421]]}
```

Das folgende SQL gibt die GeoJSON-Darstellung einer Geografie zurück.

```
SELECT ST_AsGeoJSON(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
                            st_asgeojson
-------------------------------------------------------------------
 {"type":"LineString","coordinates":[[110,40],[2,3],[-10,80],[-7,9]]}
```

ST_ WKB AsHex

ST_ AsHex WKB gibt die hexadezimale, wohlbekannte Binärdarstellung (WKB) einer Eingabegeometrie oder -geografie mithilfe von ASCII-Hexadezimalzeichen (0—9, A—F) zurück. Für 3DZ-, 3DM- und 4D-Geometrien oder Geografien verwendet ST_ WKB den Standardwert des Open Geospatial Consortium (OGC) für den Geometrie- oder Geographietyp. AsHex

Syntax

```
ST_AsHexWKB(geo)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

Rückgabetyp

VARCHAR

Wenn geo null ist, wird null zurückgegeben.

Wenn das Ergebnis größer als 64-KB VARCHAR ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die hexadezimale WKB-Darstellung eines Polygons in einer Geometrie zurück.

```
SELECT ST_AsHexWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_ashexwkb
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

Die folgende SQL-Anweisung gibt die hexadezimale WKB-Darstellung eines Polygons in einer Geografie zurück.

```
SELECT ST_AsHexWKB(ST_GeogFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))'));
```

```
st_ashexwkb
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

ST_ EWKB AsHex

ST_ AsHex EWKB gibt die erweiterte bekannte Binärdarstellung (EWKB) einer Eingabegeometrie oder -geografie unter Verwendung von ASCII-Hexadezimalzeichen (0—9, A—F) zurück. Für 3DZ-,

3DM- und 4D-Geometrien oder Geografien verwendet ST_ AsHex EWKB den erweiterten PostGIS-WKB-Wert für den Geometrie- oder Geographietyp.

Syntax

```
ST_AsHexEWKB(geo)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

Rückgabetyp

VARCHAR

Wenn geo null ist, wird null zurückgegeben.

Wenn das Ergebnis größer als 64-KB VARCHAR ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die hexadezimale EWKB-Darstellung eines Polygons in einer Geometrie zurück.

```
SELECT ST_AsHexEWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_ashexewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F0
```

Das folgende SQL gibt die hexadezimale EWKB-Darstellung eines Polygons in einer Geografie zurück.

```
SELECT ST_AsHexEWKB(ST_GeogFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))'));
```

```
st_ashexewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000000000000000000000000000000000F
```

ST_ AsText

ST_ AsText gibt die WKT-Darstellung (Wellknown Text) einer Eingabegeometrie oder -geografie zurück. Bei 3DZ-, 3DM- und 4D-Geometrien oder -Geografien fügt ST_AsEWKT Z, M oder ZM an den WKT-Wert für den Geometrie- oder Geografietyp an.

Syntax

```
ST_AsText(geo)
```

```
ST_AsText(geo, precision)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

precision

Ein Wert vom Datentyp INTEGER. Für Geometrien werden die Koordinaten von geo mit der angegebenen Genauigkeit 1–20 angezeigt. Wenn precision nicht angegeben ist, ist der Standardwert 15. Für Geografien werden die Koordinaten von geo mit der angegebenen Genauigkeit angezeigt. Wenn precision nicht angegeben ist, ist der Standardwert 15.

Rückgabetyp

VARCHAR

Wenn geo null ist, wird null zurückgegeben.

Wenn Präzision null ist, wird null zurückgegeben.

Wenn das Ergebnis größer als 64-KB VARCHAR ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die WKT-Darstellung eines Linestrings zurück.

```
SELECT ST_AsText(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326));
```

```
st_astext
-------------------------------
LINESTRING(3.14159265358979 -6.28318530717959,2.71828182845905 -1.41421356237309)
```

Das folgende SQL gibt die WKT-Darstellung eines Linestrings zurück. Die Koordinaten der Geometrien werden mit sechs Stellen Genauigkeit angezeigt.

```
SELECT ST_AsText(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326), 6);
```

```
st_astext
---------------------------------------------
 LINESTRING(3.14159 -6.28319,2.71828 -1.41421)
```

Das folgende SQL gibt die WKT-Darstellung einer Geografie zurück.

```
SELECT ST_AsText(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
             st_astext
-----------------------------------
 LINESTRING(110 40,2 3,-10 80,-7 9)
```

ST_Azimuth

ST_Azimuth gibt den nordbasierten kartesischen Azimut unter Verwendung von 2D-Projektionen der zwei Eingabepunkte zurück.

Syntax

```
ST_Azimuth(point1, point2)
```

Argumente

point1

Ein POINT-Wert vom Datentyp GEOMETRY. Der SRID von point1 muss mit der SRID von point2 übereinstimmen.

point2

Ein POINT-Wert vom Datentyp GEOMETRY. Die SRID von point2 muss mit der SRID von point1 übereinstimmen.

Rückgabetyp

Eine Zahl, die ein Bogenmaß-Winkel vom Datentyp DOUBLE PRECISION ist. Die Werte reichen von 0 (inklusive) bis 2 pi (exklusiv).

Wenn point1 oder point2 der leere Punkt ist, wird ein Fehler zurückgegeben.

Wenn entweder point1 oder point2 null ist, wird null zurückgegeben.

Wenn point1 und point2 gleich sind, dann wird null zurückgegeben.

Wenn point1 oder point2 kein Punkt ist, wird ein Fehler zurückgegeben.

Wenn point1 und point2 nicht den Wert für den SRID (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt den Azimut der Eingangspunkte zurück.

```
SELECT ST_Azimuth(ST_Point(1,2), ST_Point(5,6));
```

```
st_azimuth
-------------------
 0.7853981633974483
```

ST_Boundary

ST_Boundary gibt die Begrenzung einer Eingabegeometrie zurück, und zwar wie folgt:

- Wenn die Eingabegeometrie leer ist (d. h. sie enthält keine Punkte), wird sie unverändert zurückgegeben.
- Wenn es sich bei der Eingabegeometrie um einen Punkt oder einen nicht leeren Multipoint handelt, wird eine leere Geometriesammlung zurückgegeben.
- Wenn es sich bei der Eingabe um einen Linestring oder einen Multilinestring handelt, wird ein Multipoint zurückgegeben, der alle Punkte auf der Grenze enthält. Der Multipoint ist möglicherweise leer).
- Wenn es sich bei der Eingabe um ein Polygon handelt, das keine inneren Ringe aufweist, wird ein geschlossener Linestring zurückgegeben, der seine Grenze darstellt.
- Wenn es sich bei der Eingabe um ein Polygon mit inneren Ringen oder um ein Multipolygon handelt, wird eine Multilinestring zurückgegeben. Der Multilinestring enthält die Begrenzungen aller Ringe in der Flächengeometrie als geschlossene Linestrings.

Um die Punktgleichheit zu bestimmen, wird ST_Boundary auf der 2D-Projektion der Eingabegeometrie ausgeführt. Wenn die Eingabegeometrie leer ist, wird eine Kopie dieser Geometrie in derselben Dimension wie die Eingabe zurückgegeben. `m`-Koordinaten von nicht leeren 3DM- oder 4D-Geometrien werden gelöscht. Im speziellen Fall von 3DZ- und 4D-Multilinestrings werden die z-Koordinaten der Begrenzungspunkte des Multilinestrings so berechnet wie der Durchschnitt der jeweiligen z-Werte auf den Linestring-Begrenzungspunkten mit der gleichen 2D-Projektion.

Syntax

```
ST_Boundary(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`GEOMETRY`

Wenn geom null ist, wird null zurückgegeben.

Wenn geom eine `GEOMETRYCOLLECTION` ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die Begrenzung des Eingabe-Polygons als Multilinestring zurück.

```
SELECT ST_AsEWKT(ST_Boundary(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1
 2,2 1,1 1))')));
```

```
st_asewkt
--------------------
 MULTILINESTRING((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 1,1 1))
```

ST_Buffer

ST_Buffer gibt 2D-Geometrie zurück, die alle Punkte darstellt, deren Abstand von der auf die XY-kartesische Ebene projizierten Eingabegeometrie kleiner oder gleich dem Eingabeabstand ist.

Syntax

```
ST_Buffer(geom, distance)
```

```
ST_Buffer(geom, distance, number_of_segments_per_quarter_circle)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

distance

Ein Wert vom Datentyp `DOUBLE PRECISION`, der die Entfernung (oder den Radius) des Puffers darstellt.

number_of_segments_per_quarter_circle

Ein Wert vom Datentyp `INTEGER`. Dieser Wert bestimmt die Anzahl der Punkte, die einem Viertelkreis um jeden Scheitelpunkt der Eingabegeometrie annähern sollen. Negative Werte sind standardmäßig auf null. Der Standardwert ist 8.

Rückgabetyp

GEOMETRY

Die Funktion ST_Buffer gibt zweidimensionale (2D) Geometrie in der xy-kartesischen Ebene zurück.

Wenn geom eine GEOMETRYCOLLECTION ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL gibt den Puffer von Linestring zurück.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('LINESTRING(1 2,5 2,5 8)'), 2));
```

```
               st_asewkt
POLYGON((-1 2,-0.96157056080646 2.39018064403226,-0.847759065022573
 2.76536686473018,-0.662939224605089 3.11114046603921,-0.414213562373093
 3.4142135623731,-0.111140466039201 3.66293922460509,0.234633135269824
 3.84775906502257,0.609819355967748 3.96157056080646,1 4,3 4,3 8,3.03842943919354
 8.39018064403226,3.15224093497743 8.76536686473018,3.33706077539491
 9.11114046603921,3.58578643762691 9.4142135623731,3.8888595339608
 9.66293922460509,4.23463313526982 9.84775906502257,4.60981935596775
 9.96157056080646,5 10,5.39018064403226 9.96157056080646,5.76536686473018
 9.84775906502257,6.11114046603921 9.66293922460509,6.4142135623731
 9.41421356237309,6.66293922460509 9.1111404660392,6.84775906502258
 8.76536686473017,6.96157056080646 8.39018064403225,7 8,7 2,6.96157056080646
 1.60981935596774,6.84775906502257 1.23463313526982,6.66293922460509
 0.888859533960796,6.41421356237309 0.585786437626905,6.1111404660392
 0.33706077539491,5.76536686473018 0.152240934977427,5.39018064403226
 0.0384294391935391,5 0,1 0,0.609819355967744 0.0384294391935391,0.234633135269821
 0.152240934977427,-0.111140466039204 0.337060775394909,-0.414213562373095
 0.585786437626905,-0.662939224605091 0.888859533960796,-0.847759065022574
 1.23463313526982,-0.961570560806461 1.60981935596774,-1 2))
```

Das folgende SQL gibt den Puffer der Eingangspunktgeometrie zurück, der einem Kreis annähert. Da der Befehl die Anzahl der Segmente pro Viertelkreis nicht angibt, verwendet die Funktion den Standardwert von acht Segmenten, um den Viertelkreis anzunähern.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('POINT(3 4)'), 2));
```

```
               st_asewkt
```

```
POLYGON((1 4,1.03842943919354 4.39018064403226,1.15224093497743
 4.76536686473018,1.33706077539491 5.11114046603921,1.58578643762691
 5.4142135623731,1.8888595339608 5.66293922460509,2.23463313526982
 5.84775906502257,2.60981935596775 5.96157056080646,3 6,3.39018064403226
 5.96157056080646,3.76536686473019 5.84775906502257,4.11114046603921
 5.66293922460509,4.4142135623731 5.41421356237309,4.66293922460509
 5.1111404660392,4.84775906502258 4.76536686473017,4.96157056080646 4.39018064403225,5
 4,4.96157056080646 3.60981935596774,4.84775906502257 3.23463313526982,4.66293922460509
 2.8888595339608,4.41421356237309 2.58578643762691,4.1111404660392
 2.33706077539491,3.76536686473018 2.15224093497743,3.39018064403226 2.03842943919354,3
 2,2.60981935596774 2.03842943919354,2.23463313526982 2.15224093497743,1.8888595339608
 2.33706077539491,1.58578643762691 2.58578643762691,1.33706077539491
 2.8888595339608,1.15224093497743 3.23463313526982,1.03842943919354 3.60981935596774,1
 4))
```

Das folgende SQL gibt den Puffer der Eingangspunktgeometrie zurück, der einem Kreis annähert. Da der Befehl 3 als Anzahl der Segmente pro Viertelkreis angibt, verwendet die Funktion drei Segmente, um den Viertelkreis anzunähern.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('POINT(3 4)'), 2, 3));
```

```
               st_asewkt
POLYGON((1 4,1.26794919243112 5,2 5.73205080756888,3 6,4
 5.73205080756888,4.73205080756888 5,5 4,4.73205080756888 3,4 2.26794919243112,3 2,2
 2.26794919243112,1.26794919243112 3,1 4))
```

ST_Centroid

ST_Centroid gibt einen Punkt zurück, der einen Schwerpunkt einer Geometrie darstellt, und zwar wie folgt:

- Für `POINT`-Geometrien gibt ST_Centroid den Punkt zurück, dessen Koordinaten die durchschnittlichen Koordinaten der Punkte in der Geometrie sind.
- Für `LINESTRING`-Geometrien gibt ST_Centroid den Punkt zurück, dessen Koordinaten der gewichtete Durchschnitt der Mittelpunkte der Segmente der Geometrie sind, wobei die Gewichte die Längen der Segmente in der Geometrie sind.
- Für `POLYGON`-Geometrien gibt ST_Centroid den Punkt zurück, dessen Koordinaten der gewichtete Durchschnitt der Schwerpunkte einer Triangulation der Flächengeometrie sind, wobei die Gewichte die Flächen der Dreiecke in der Triangulation sind.

- Für Geometriesammlungen gibt ST_Centroid den gewichteten Durchschnitt der Schwerpunkte der Geometrien mit maximaler topologischer Dimension in der Geometriesammlung zurück.

Syntax

```
ST_Centroid(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt einen zentralen Punkt aus einem Eingabe-Linestring zurück.

```
SELECT ST_AsEWKT(ST_Centroid(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9, -22
 -33)', 4326)))
```

```
                    st_asewkt
-----------------------------------------------------
 SRID=4326;POINT(15.6965103455214 27.0206782881905)
```

ST_Collect

ST_Collect hat zwei Varianten. Eine akzeptiert zwei Geometrien, die andere einen Aggregatausdruck.

Die erste Variante von ST_Collect erstellt eine Geometrie aus den Eingabegeometrien. Die Reihenfolge der Eingabegeometrien bleibt erhalten. Diese Variante funktioniert wie folgt:

- Wenn beide Eingabegeometrien Punkte sind, wird ein `MULTIPOINT` mit zwei Punkten zurückgegeben.
- Wenn beide Eingabegeometrien Linestrings sind, wird ein `MULTILINESTRING` mit zwei Linestrings zurückgegeben.
- Wenn beide Eingabegeometrien Polygone sind, wird ein `MULTIPOLYGON` mit zwei Polygonen zurückgegeben.
- Andernfalls wird eine `GEOMETRYCOLLECTION` mit zwei Eingabegeometrien zurückgegeben.

Die zweite Variante von ST_Collect erstellt eine Geometrie aus Geometrien in einer Geometrie-Spalte. Es gibt keine festgelegte Rückgabereihenfolge der Geometrien. Geben Sie die Klausel WITHIN GROUP (ORDER BY …) an, um die Reihenfolge der zurückgegebenen Geometrien festzulegen. Diese Variante funktioniert wie folgt:

- Wenn alle Nicht-NULL-Zeilen im Eingabeaggregatsausdruck Punkte sind, wird ein Multipoint zurückgegeben, der alle Punkte im Aggregatsausdruck enthält.
- Wenn alle Nicht-NULL-Zeilen im Aggregatsausdruck Linestrings sind, wird ein Multilinestring zurückgegeben, der alle Linestrings im Aggregatsausdruck enthält.
- Wenn alle Nicht-NULL-Zeilen im Aggregatsausdruck Polygone sind, wird ein Multipolygon zurückgegeben, das alle Polygone im Aggregatsausdruck enthält.
- Andernfalls wird eine `GEOMETRYCOLLECTION` zurückgegeben, die alle Geometrien im Aggregatausdruck enthält.

ST_Collect gibt die Geometrie der gleichen Dimension wie die der Eingabegeometrien zurück. Alle Eingabegeometrien müssen die gleiche Dimension haben.

Syntax

```
ST_Collect(geom1, geom2)
```

```
ST_Collect(aggregate_expression)  [WITHIN GROUP (ORDER BY sort_expression1 [ASC | DESC]
 [, sort_expression2 [ASC | DESC] ...])]
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

aggregate_expression

Eine Spalte vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

[WITHIN GROUP (ORDER BY sort_expression1 [ASC | DESC] [, sort_expression2 [ASC | DESC] ...])]

Eine optionale Klausel, die die Sortierreihenfolge der aggregierten Werte angibt. Die Klausel ORDER BY enthält eine Liste von Sortierungsausdrücken. Sortierungsausdrücke sind Ausdrücke, die den gültigen Sortierungsausdrücken in einer Abfrageauswahlliste ähneln, z. B. einem Spaltennamen. Sie können aufsteigend (ASC) oder absteigend (DESC) sortieren. Der Standardwert ist ASC.

Rückgabetyp

GEOMETRY des Subtyps MULTIPOINT, MULTILINESTRING, MULTIPOLYGON oder GEOMETRYCOLLECTION.

Der SRID-Wert der zurückgegebenen Geometrie ist der SRID-Wert der Eingangsgeometrien.

Wenn sowohl geom1 oder geom2 null sind, wird null zurückgegeben.

Wenn alle Zeilen von aggregate_expression null sind, wird null zurückgegeben.

Wenn geom1 null ist, wird eine Kopie von geom2 zurückgegeben. Wenn geom2 null ist, wird eine Kopie von geom1 zurückgegeben.

Wenn geom1 und geom2 unterschiedliche SRID-Werte haben, wird ein Fehler zurückgegeben.

Wenn zwei Geometrien in aggregate_expression unterschiedliche SRID-Werte haben, wird ein Fehler zurückgegeben.

Wenn die zurückgegebene Geometrie größer ist als die maximale Größe einer GEOMETRY, wird ein Fehler zurückgeben.

Wenn geom1 und geom2 unterschiedliche Dimensionen haben, wird ein Fehler zurückgegeben.

Wenn zwei Geometrien in aggregate_expression unterschiedliche Dimensionen haben, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung eine Geometriesammlung zurück, die zwei Eingabegeometrien enthält.

```
SELECT ST_AsText(ST_Collect(ST_GeomFromText('LINESTRING(0 0,1 1)'),
 ST_GeomFromText('POLYGON((10 10,20 10,10 20,10 10))')));
```

```
st_astext
-----------
 GEOMETRYCOLLECTION(LINESTRING(0 0,1 1),POLYGON((10 10,20 10,10 20,10 10)))
```

Die folgende SQL-Anweisung erfasst alle Geometrien aus einer Tabelle in einer Geometriesammlung.

```
WITH tbl(g) AS (SELECT ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT NULL::geometry UNION ALL
SELECT ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326))
SELECT ST_AsEWKT(ST_Collect(g)) FROM tbl;
```

```
st_astext
-----------
 SRID=4326;GEOMETRYCOLLECTION(POINT(1 2),LINESTRING(0 0,10 0),MULTIPOINT((13 4),(8 5),
(4 4)),POLYGON((0 0,10 0,0 10,0 0)))
```

Die folgende SQL-Anweisung sammelt alle Geometrien in der Tabelle, gruppiert nach der ID-Spalte und geordnet nach dieser ID. In diesem Beispiel werden resultierende Geometrien wie folgt nach ID gruppiert:

- id 1 – Punkte in einem Multipoint.

- id 2 – Linestrings in einem Multilinestring.
- id 3 – gemischte Subtypen in einer Geometriesammlung.
- id 4 – Polygone in einem Multipolygon.
- id 5 – null und das Ergebnis ist null.

```
WITH tbl(id, g) AS (SELECT 1, ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT 1, ST_GeomFromText('POINT(4 5)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(10 0,20 -5)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5))', 4326) UNION
 ALL
SELECT 4, ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326) UNION ALL
SELECT 4, ST_GeomFromText('POLYGON((20 20,20 30,30 20,20 20))', 4326) UNION ALL
SELECT 1, NULL::geometry UNION ALL SELECT 2, NULL::geometry UNION ALL
SELECT 5, NULL::geometry UNION ALL SELECT 5, NULL::geometry)
SELECT id, ST_AsEWKT(ST_Collect(g)) FROM tbl GROUP BY id ORDER BY id;
```

```
 id |                                                st_asewkt

----
+-----------------------------------------------------------------------------------------------
  1 | SRID=4326;MULTIPOINT((1 2),(4 5))
  2 | SRID=4326;MULTILINESTRING((0 0,10 0),(10 0,20 -5))
  3 | SRID=4326;GEOMETRYCOLLECTION(MULTIPOINT((13 4),(8 5),(4 4)),MULTILINESTRING((-1
 -1,-2 -2),(-3 -3,-5 -5)))
  4 | SRID=4326;MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((20 20,20 30,30 20,20 20)))
  5 |
```

Die folgende SQL-Anweisung erfasst alle Geometrien aus einer Tabelle in einer Geometriesammlung. Die Ergebnisse werden in absteigender Reihenfolge nach `id` angeordnet und dann lexikografisch auf Grundlage ihrer minimalen und maximalen x-Koordinaten.

```
WITH tbl(id, g) AS (
SELECT 1, ST_GeomFromText('POINT(4 5)', 4326) UNION ALL
SELECT 1, ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(10 0,20 -5)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
```

```
SELECT 3, ST_GeomFromText('MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5))', 4326) UNION
 ALL
SELECT 4, ST_GeomFromText('POLYGON((20 20,20 30,30 20,20 20))', 4326) UNION ALL
SELECT 4, ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326) UNION ALL
SELECT 1, NULL::geometry UNION ALL SELECT 2, NULL::geometry UNION ALL
SELECT 5, NULL::geometry UNION ALL SELECT 5, NULL::geometry)
SELECT ST_AsEWKT(ST_Collect(g) WITH GROUP (ORDER BY id DESC, ST_XMin(g), ST_XMax(g)))
 FROM tbl;
```

```
                          st_asewkt

-----------------------------------------------------------------------------------------------
 SRID=4326;GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0 0)),POLYGON((20 20,20 30,30
 20,20 20)),MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5)),MULTIPOINT((13 4),(8 5),(4
 4)),LINESTRING(0 0,10 0),LINESTRING(10 0,20 -5),POINT(1 2),POINT(4 5)
```

ST_Contains

ST_Contains gibt true zurück, wenn die 2D-Projektion der ersten Eingabegeometrie die 2D-Projektion der zweiten Eingabegeometrie enthält. Die Geometrie A enthält die Geometrie B, wenn jeder Punkt in B ein Punkt in A ist, und ihre Innenräume einen nicht leeren Schnittpunkt haben.

ST_Contains(A, B) entspricht ST_Within(B, A).

Syntax

```
ST_Contains(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Dieser Wert wird mit geom1 verglichen, um festzustellen, ob er in geom1 enthalten ist.

Rückgabetyp

`BOOLEAN`

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das erste Polygon das zweite Polygon enthält.

```
SELECT ST_Contains(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_contains
-----------
 false
```

ST_ ContainsProperly

ST_ ContainsProperly gibt true zurück, wenn beide Eingabegeometrien nicht leer sind und alle Punkte der 2D-Projektion der zweiten Geometrie Innenpunkte der 2D-Projektion der ersten Geometrie sind.

Syntax

```
ST_ContainsProperly(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp darf nicht sein `GEOMETRYCOLLECTION`.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Der Subtyp darf nicht sein GEOMETRYCOLLECTION. Dieser Wert wird mit geom1 verglichen, um festzustellen, ob alle seine Punkte innere Punkte von geom1 sind.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die Werte von ST_Contains und ST_ zurück, ContainsProperly wobei die Eingabelinienfolge das Innere und die Grenze des Eingabepolygons (aber nicht dessen Außenseite) schneidet. Das Polygon enthält den Linestring, enthält den Linestring aber nicht richtig.

```
WITH tmp(g1, g2)
AS (SELECT ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('LINESTRING(5 5,10 5,10 6,5 5)')) SELECT ST_Contains(g1, g2),
 ST_ContainsProperly(g1, g2)
FROM tmp;
```

```
 st_contains | st_containsproperly
-------------+---------------------
 t           | f
```

ST_ ConvexHull

ST_ ConvexHull gibt eine Geometrie zurück, die die konvexe Hülle der nicht leeren Punkte darstellt, die in der Eingabegeometrie enthalten sind.

Bei einer leeren Eingabe ist die resultierende Geometrie die gleiche wie die Eingabegeometrie. Bei allen nicht leeren Eingaben wird die Funktion auf der 2D-Projektion der Eingabegeometrie ausgeführt.

Die Dimension der Ausgabegeometrie hängt jedoch von der Dimension der Eingabegeometrie ab. Genauer gesagt: Wenn es sich bei der Eingabegeometrie um eine nicht leere 3DM- oder 3D-Geometrie handelt, werden die m-Koordinaten gelöscht. Das heißt, dass die Dimension der zurückgegebenen Geometrie jeweils 2D oder 3DZ ist. Wenn es sich bei der Eingabe um eine nicht leere 2D- oder 3DZ-Geometrie handelt, hat die resultierende Geometrie dieselbe Dimension.

Syntax

```
ST_ConvexHull(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Die zurückgegebenen Werte lauten wie folgt.

Anzahl der Punkte auf der konvexen Hülle	Geometrie-Subtyp
0	Eine Kopie von geom wird zurückgegeben.
1	Ein POINT-Subtyp wird zurückgegeben.
2	Ein LINESTRING -Subtyp wird zurückgeg eben. Die beiden Punkte des zurückgegebenen Linestrings sind lexikografisch geordnet.
3 oder höher	Ein POLYGON-Subtyp ohne innere Ringe wird zurückgegeben. Das Polygon ist im Uhrzeiger

Anzahl der Punkte auf der konvexen Hülle	Geometrie-Subtyp
	sinn ausgerichtet und der erste Punkt des äußeren Rings ist der lexikografisch kleinste Punkt des Rings.

Beispiele

Das folgende SQL-Anweisung gibt die EWKT-Darstellung eines Linestrings zurück. In diesem Fall ist die zurückgegebene konvexe Hülle ein Polygon.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('LINESTRING(0 0,1 0,0 1,1 1,0.5 0.5)')))
 as output;
```

```
output
-------------
POLYGON((0 0,0 1,1 1,1 0,0 0))
```

Das folgende SQL gibt die EWKT-Darstellung eines Linestrings zurück. In diesem Fall ist die zurückgegebene konvexe Hülle ein Linestring.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('LINESTRING(0 0,1 1,0.2 0.2,0.6 0.6,0.5
 0.5)'))) as output;
```

```
output
-------------
LINESTRING(0 0,1 1)
```

Die folgende SQL-Anweisung gibt die EWKT-Darstellung eines Multipoints zurück. In diesem Fall ist die zurückgegebene konvexe Hülle ein Punkt.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('MULTIPOINT(0 0,0 0,0 0)'))) as output;
```

```
output
-------------
```

```
POINT(0 0)
```

ST_ CoveredBy

ST_ CoveredBy gibt true zurück, wenn die 2D-Projektion der ersten Eingabegeometrie durch die 2D-Projektion der zweiten Eingabegeometrie verdeckt wird. Die Geometrie A wird durch die Geometrie B gedeckt, wenn beide nicht leer sind und jeder Punkt in A ein Punkt in B ist.

ST_ CoveredBy (A,B) entspricht ST_Covers (,). B A

Syntax

```
ST_CoveredBy(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Dieser Wert wird mit geom2 verglichen, um festzustellen, ob er durch geom2 abgedeckt ist.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das erste Polygon durch das zweite Polygon abgedeckt ist.

```
SELECT ST_CoveredBy(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_coveredby
-----------
 true
```

ST_Covers

ST_Covers gibt true zurück, wenn die 2D-Projektion der ersten Eingabegeometrie die 2D-Projektion der zweiten Eingabegeometrie abgedeckt. Die Geometrie A deckt die Geometrie B, wenn beide nicht leer sind und jeder Punkt in B ein Punkt in A ist.

ST_Covers (A,B) entspricht ST_ (,). CoveredBy B A

Syntax

```
ST_Covers(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Dieser Wert wird mit geom1 verglichen, um festzustellen, ob er geom1 abdeckt.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das erste Polygon das zweite Polygon abdeckt.

```
SELECT ST_Covers(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_covers
-----------
 false
```

ST_Crosses

ST_Crosses gibt true zurück, wenn sich die 2D-Projektionen der beiden Eingabegeometrien kreuzen.

Syntax

```
ST_Crosses(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung prüft, ob das erste Polygon das zweite Polygon schneidet. In diesem Beispiel schneidet der Multipoint sowohl das Innere als auch das Äußere des Polygons. Daher gibt ST_Crosses den Wert true zurück.

```
SELECT ST_Crosses (ST_GeomFromText('polygon((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('multipoint(5 5,0 0,-1 -1)'));
```

```
st_crosses
-------------
 true
```

Die folgende SQL-Anweisung prüft, ob das erste Polygon das zweite Polygon schneidet. In diesem Beispiel schneidet der Multipoint das Äußere, nicht jedoch das Innere, des Polygons. Daher gibt ST_Crosses den Wert false zurück.

```
SELECT ST_Crosses (ST_GeomFromText('polygon((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('multipoint(0 0,-1 -1)'));
```

```
st_crosses
-------------
 false
```

ST_Dimension

ST_Dimension gibt die inhärente Dimension einer Eingangsgeometrie zurück. Die Inhärentdimension ist der Dimensionswert des Subtyps, der in der Geometrie definiert ist.

Bei 3DM-, 3DZ- und 4D-Geometrieeingaben gibt ST_Dimension dasselbe Ergebnis zurück wie bei 2D-Geometrieeingaben.

Syntax

```
ST_Dimension(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`INTEGER` stellt die inhärente Dimension von geom dar.

Wenn geom null ist, wird null zurückgegeben.

Die zurückgegebenen Werte lauten wie folgt.

Zurückgegebener Wert	Geometrie-Subtyp
0	Wird zurückgegeben, wenn geom ein `POINT`- oder `MULTIPOINT` -Subtyp ist.
1	Wird zurückgegeben, wenn geom ein `LINESTRING` - oder `MULTILINESTRING` -Subtyp ist.
2	Wird zurückgegeben, wenn geom ein `POLYGON`- oder `MULTIPOLYGON` -Subtyp ist.
0	Wird zurückgegeben, wenn geom ein leerer `GEOMETRYCOLLECTION` -Subtyp ist.
Größte Dimension der Komponenten der Collection	Wird zurückgegeben, wenn geom ein `GEOMETRYCOLLECTION` -Subtyp ist.

Beispiele

Das folgende SQL konvertiert eine WKT-Darstellung eines Vierpunkt-LINESTRINGs in ein GEOMETRY-Objekt und gibt die Dimension des Linienstrings zurück.

```
SELECT ST_Dimension(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_dimension
-------------
1
```

ST_Disjoint

ST_Disjoint gibt true zurück, wenn die 2D-Projektionen der beiden Eingabegeometrien keine gemeinsamen Punkte haben.

Syntax

```
ST_Disjoint(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das erste Polygon vom zweiten Polygon getrennt ist.

```
SELECT ST_Disjoint(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(2 2,2 5,5 5,5
 2,2 2))'), ST_Point(4, 4));
```

```
st_disjoint
-----------
 true
```

ST_Distance

Bei Eingabegeometrien gibt ST_Distance den minimalen euklidischen Abstand zwischen den 2D-Projektionen der beiden Eingabegeometriewerte zurück.

Bei 3DM-, 3DZ- und 4D-Geometrien gibt ST_Distance den euklidischen Abstand zwischen den 2D-Projektionen beider Eingabegeometriewerte zurück.

Bei Eingabegeografien gibt ST_Distance den geodätische Abstand von zwei 2D-Punkten an. Die Abstände werden in Meter angegeben. Bei Geografien, bei denen es sich nicht um Punkte oder leere Punkte handelt, wird ein Fehler zurückgegeben.

Syntax

```
ST_Distance(geo1, geo2)
```

Argumente

geo1

Ein Wert vom Datentyp `GEOMETRY` oder `GEOGRAPHY` oder ein Ausdruck, der zu einem `GEOMETRY`- oder `GEOGRAPHY`-Typ ausgewertet wird. Der Datentyp von geo1 muss dem Datentyp von geo2 entsprechen.

geo2

Ein Wert vom Datentyp `GEOMETRY` oder `GEOGRAPHY` oder ein Ausdruck, der zu einem `GEOMETRY`- oder `GEOGRAPHY`-Typ ausgewertet wird. Der Datentyp von geo2 muss dem Datentyp von geo1 entsprechen.

Rückgabetyp

`DOUBLE PRECISION` in der gleichen Einheit wie die Eingangsgeometrien oder -geografien.

Wenn geo1 oder geo2 null oder leer ist, wird null zurückgegeben.

Wenn geo1 und geo2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geo1 oder geo2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt den Abstand zwischen zwei Polygonen zurück.

```
SELECT ST_Distance(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 -3,-2 -1,0 -3,-1 -3))'));
```

```
  st_distance
-----------
1.4142135623731
```

Die folgende SQL gibt die Entfernung (in Metern) zwischen dem Brandenburger Tor und dem Reichstagsgebäude in Berlin unter Verwendung eines GEOGRAPHY-Datentyps zurück.

```
SELECT ST_Distance(ST_GeogFromText('POINT(13.37761826722198 52.516411678282445)'),
 ST_GeogFromText('POINT(13.377950831464005 52.51705102546893)'));
```

```
   st_distance
------------------
 74.64129172609631
```

ST_ DistanceSphere

ST_ DistanceSphere gibt den Abstand zwischen zwei Punktgeometrien zurück, die auf einer Kugel liegen.

Syntax

```
ST_DistanceSphere(geom1, geom2)
```

```
ST_DistanceSphere(geom1, geom2, radius)
```

Argumente

geom1

Ein Punktwert in Grad vom Datentyp GEOMETRY auf einer Kugel. Die erste Koordinate des Punktes ist der Längengrad. Die zweite Koordinate des Punktes ist der Breitenwert. Für 3DZ-, 3DM- oder 4D-Geometrien werden nur die ersten beiden Koordinaten verwendet.

geom2

Ein Punktwert in Grad vom Datentyp GEOMETRY auf einer Kugel. Die erste Koordinate des Punktes ist der Längengrad. Die zweite Koordinate des Punktes ist der Breitenwert. Für 3DZ-, 3DM- oder 4D-Geometrien werden nur die ersten beiden Koordinaten verwendet.

radius

Der Radius einer Kugel vom Datentyp DOUBLE PRECISION. Wenn kein Radius angegeben wird, ist die Kugel standardmäßig die Erde und der Radius wird aus der WGS 84-Darstellung (World Geodetic System) des Ellipsoids berechnet.

Rückgabetyp

DOUBLE PRECISION in der gleichen Einheit wie der Radius. Wenn kein Radius vorgegeben ist, wird die Entfernung in Metern angegeben.

Wenn geom1 oder geom2 null oder leer ist, wird null zurückgegeben.

Wenn kein Radius angegeben wird, wird das Ergebnis in Metern entlang der Erdoberfläche angegeben.

Wenn radius eine negative Zahl ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 kein Punkt ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende Beispiel-SQL berechnet die Entfernung in Kilometern zwischen zwei Punkten auf der Erde.

```
SELECT ROUND(ST_DistanceSphere(ST_Point(-122, 47), ST_Point(-122.1, 47.1))/ 1000, 0);
```

```
  round
-----------
 13
```

Das folgende Beispiel SQL berechnet die Kilometerabstände zwischen drei Flughafenstandorten in Deutschland: Berlin Tegel (TXL), Munich International (MUC) und Frankfurt International (FRA).

```
WITH airports_raw(code,lon,lat) AS (
(SELECT 'MUC', 11.786111, 48.353889) UNION
(SELECT 'FRA', 8.570556, 50.033333) UNION
(SELECT 'TXL', 13.287778, 52.559722)),
airports1(code,location) AS (SELECT code, ST_Point(lon, lat) FROM airports_raw),
airports2(code,location) AS (SELECT * from airports1)
SELECT (airports1.code || ' <-> ' || airports2.code) AS airports,
round(ST_DistanceSphere(airports1.location, airports2.location) / 1000, 0) AS
 distance_in_km
FROM airports1, airports2 WHERE airports1.code < airports2.code ORDER BY 1;
```

```
  airports   | distance_in_km
-------------+----------------
 FRA <-> MUC |            299
 FRA <-> TXL |            432
 MUC <-> TXL |            480
```

ST_DWithin

ST_DWithin gibt true zurück, wenn der euklidische Abstand zwischen den 2D-Projektionen der beiden Eingabegeometriewerte nicht größer als ein Schwellenwert ist.

Syntax

```
ST_DWithin(geom1, geom2, threshold)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

threshold

Ein Wert vom Datentyp DOUBLE PRECISION. Dieser Wert wird in Einheiten der Eingangsargumente angegeben.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn threshold1 negativ ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob der Abstand zwischen zwei Polygonen innerhalb von fünf Einheiten liegt.

```
SELECT ST_DWithin(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'),5);
```

```
st_dwithin
-----------
 true
```

ST_ EndPoint

ST_ EndPoint gibt den letzten Punkt einer Eingabelinienfolge zurück. Der Spatial Reference System Identifier (SRID)-Wert des Ergebnisses entspricht dem der Eingabegeometrie. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_EndPoint(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `LINESTRING`.

Rückgabetyp

`GEOMETRY`

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom nicht `LINESTRING` ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine EWKT-Darstellung (Extended Well-known Text) eines Vierpunkt-`LINESTRING` zu einem `GEOMETRY`-Objekt und den Endpunkt des Linestrings zurück.

```
SELECT ST_AsEWKT(ST_EndPoint(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 5)
```

ST_Envelope

ST_Envelope gibt den minimalen Begrenzungsrahmen der Eingabegeometrie wie folgt zurück:

- Wenn die Eingabegeometrie leer ist, ist die zurückgegebene Geometrie eine Kopie der Eingabegeometrie.
- Wenn der minimale Begrenzungsrahmen der Eingabegeometrie zu einem Punkt degeneriert wird, ist die zurückgegebene Geometrie ein Punkt.
- Wenn der minimale Begrenzungsrahmen der Eingabegeometrie eindimensional ist, wird ein Zwei-Punkt-Linestring zurückgegeben.
- Wenn keiner der vorhergehenden Werte „true“ ist, gibt die Funktion ein im Uhrzeigersinn orientiertes Polygon zurück, dessen Eckpunkte die Ecken des minimalen Begrenzungsrahmens sind.

Der Spatial Reference System Identifier (SRID)-Wert der zurückgegebenen Geometrie entspricht dem der Eingabegeometrie.

Bei allen nicht leeren Eingaben wird die Funktion auf der 2D-Projektion der Eingabegeometrie ausgeführt.

Syntax

```
ST_Envelope(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`GEOMETRY`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung konvertiert eine WKT-Darstellung (Well-known text) eines Vierpunkt-LINESTRING in ein GEOMETRY-Objekt und gibt ein Polygon zurück, dessen Eckpunkte das minimale Begrenzungsfeld sind.

```
SELECT ST_AsText(ST_Envelope(ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0
 10,0 0)),LINESTRING(20 10,20 0,10 0))')));
```

```
    st_astext
-----------------------------------
  POLYGON((0 0,0 10,20 10,20 0,0 0))
```

ST_Equals

ST_Equals gibt true zurück, wenn die 2D-Projektionen der Eingabegeometrien geometrisch gleich sind. Geometrien gelten als geometrisch gleich, wenn sie gleiche Punktmengen haben und ihre Innenräume einen nicht leeren Schnittpunkt haben.

Syntax

```
ST_Equals(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Dieser Wert wird mit geom1 verglichen, um festzustellen, ob er gleich geom1 ist.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob die beiden Polygone geometrisch gleich sind.

```
SELECT ST_Equals(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_equals
-----------
 false
```

Das folgende SQL prüft, ob die beiden Linestrings geometrisch gleich sind.

```
SELECT ST_Equals(ST_GeomFromText('LINESTRING(1 0,10 0)'), ST_GeomFromText('LINESTRING(1
 0,5 0,10 0)'));
```

```
st_equals
-----------
 true
```

ST_ ExteriorRing

ST_ ExteriorRing gibt eine geschlossene Linienfolge zurück, die den äußeren Ring eines Eingabepolygons darstellt. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_ExteriorRing(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY des Subtyps LINESTRING.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom kein Polygon ist, wird null zurückgegeben.

Wenn geom leer ist, wird ein leeres Polygon zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt den äußeren Ring eines Polygons als geschlossenen Linestring zurück.

```
SELECT ST_AsEWKT(ST_ExteriorRing(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17
 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12
 14,15 14,13 11,12 14))')));
```

```
st_asewkt
-----------
 LINESTRING(7 9,8 7,11 6,15 8,16 6,17 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9)
```

ST_Force2D

ST_Force2D gibt eine 2D-Geometrie der Eingabegeometrie zurück. Bei 2D-Geometrien wird eine Kopie der Eingabe zurückgegeben. Bei 3DZ-, 3DM- und 4D-Geometrien projiziert ST_Force2D

die Geometrie auf die kartesische XY-Ebene. Leere Punkte in der Eingabegeometrie bleiben leere Punkte in der Ausgabegeometrie.

Syntax

```
ST_Force2D(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird eine leere Geometrie zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine 2D-Geometrie aus einer 3DZ-Geometrie zurück.

```
SELECT ST_AsEWKT(ST_Force2D(ST_GeomFromText('MULTIPOINT Z(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT((0 1),EMPTY,(2 3),(5 6))
```

ST_Force3D

ST_Force3D ist ein Alias für ST_Force3DZ. Weitere Informationen finden Sie unter ST_Force3DZ.

ST_Force3DM

ST_Force3DM gibt eine 3DM-Geometrie der Eingabegeometrie zurück. Bei 2D-Geometrien werden alle `m`-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie auf `0` festgelegt. Bei 3DM-Geometrien wird eine Kopie der Eingabegeometrie zurückgegeben. Bei 3DZ-Geometrien wird die Geometrie auf die kartesische xy-Ebene projiziert und alle `m`-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie werden auf `0` festgelegt. Bei 4D-Geometrien wird die Geometrie auf den kartesischen XYM-Raum projiziert. Leere Punkte in der Eingabegeometrie bleiben leere Punkte in der Ausgabegeometrie.

Syntax

```
ST_Force3DM(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`GEOMETRY`.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird eine leere Geometrie zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine 3DM-Geometrie aus einer 3DZ-Geometrie zurück.

```
SELECT ST_AsEWKT(ST_Force3DM(ST_GeomFromText('MULTIPOINT Z(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT M ((0 1 0),EMPTY,(2 3 0),(5 6 0))
```

ST_Force3DZ

ST_Force3DZ gibt eine 3DZ-Geometrie von der Eingabegeometrie zurück. Bei 2D-Geometrien werden alle z-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie auf `0` festgelegt. Bei 3DM-Geometrien wird die Geometrie auf die kartesische xy-Ebene projiziert und alle z-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie werden auf `0` festgelegt. Bei 3DZ-Geometrien wird eine Kopie der Eingabegeometrie zurückgegeben. Bei 4D-Geometrien wird die Geometrie auf den kartesischen XYZ-Raum projiziert. Leere Punkte in der Eingabegeometrie bleiben leere Punkte in der Ausgabegeometrie.

Syntax

```
ST_Force3DZ(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`GEOMETRY`.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird eine leere Geometrie zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine 3DZ-Geometrie aus einer 3DM-Geometrie zurück.

```
SELECT ST_AsEWKT(ST_Force3DZ(ST_GeomFromText('MULTIPOINT M(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT Z ((0 1 0),EMPTY,(2 3 0),(5 6 0))
```

ST_Force4D

ST_Force4D gibt eine 4D-Geometrie der Eingabegeometrie zurück. Bei 2D-Geometrien werden alle z- und m-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie auf 0 festgelegt. Bei 3DM-Geometrien werden alle z-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie auf 0 festgelegt. Bei 3DZ-Geometrien werden alle m-Koordinaten der nicht leeren Punkte in der Ausgabegeometrie auf 0 festgelegt. Bei 4D-Geometrien wird eine Kopie der Eingabegeometrie zurückgegeben. Leere Punkte in der Eingabegeometrie bleiben leere Punkte in der Ausgabegeometrie.

Syntax

```
ST_Force4D(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird eine leere Geometrie zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine 4D-Geometrie aus einer 3DM-Geometrie zurück.

```
SELECT ST_AsEWKT(ST_Force4D(ST_GeomFromText('MULTIPOINT M(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT ZM ((0 1 0 2),EMPTY,(2 3 0 4),(5 6 0 7))
```

ST_ GeoHash

ST_ GeoHash gibt die geohash Darstellung des Eingabepunkts mit der angegebenen Genauigkeit zurück. Der Standardgenauigkeitswert ist 20. Weitere Informationen zur Definition von Geohash finden Sie unter Geohash in Wikipedia.

Syntax

```
ST_GeoHash(geom)
```

```
ST_GeoHash(geom, precision)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

precision

Ein Wert vom Datentyp INTEGER. Der Standardwert ist 20.

Rückgabetyp

GEOMETRY

Die Funktion gibt die geohash-Darstellung des Eingabepunkts zurück.

Ist der Eingangspunkt leer, gibt die Funktion null zurück.

Wenn die Eingabegeometrie kein Punkt ist, gibt die Funktion einen Fehler zurück.

Beispiele

Die folgende SQL gibt die Geohash-Darstellung des Eingabepunkts zurück.

```
SELECT ST_GeoHash(ST_GeomFromText('POINT(45 -45)'), 25) AS geohash;
```

```
          geohash
---------------------------
 m0000000000000000000gzz
```

Das folgende SQL gibt null zurück, da der Eingangspunkt leer ist.

```
SELECT ST_GeoHash(ST_GeomFromText('POINT EMPTY'), 10) IS NULL AS result;
```

```
 result
---------
 true
```

ST_ GeogFromText

ST_ GeogFromText konstruiert ein Geografieobjekt aus einer Repräsentation einer Eingabegeografie in bekanntem Text (WKT) oder erweitertem bekanntem Text (EWKT).

Syntax

```
ST_GeogFromText(wkt_string)
```

Argumente

wkt_string

Ein Wert vom Datentyp VARCHAR, der eine WKT- oder EWKT-Darstellung einer Geografie ist.

Rückgabetyp

GEOGRAPHY

Wenn der SRID-Wert auf den angegebenen Wert in der Eingabe festgelegt ist. Wenn der SRID-Wert nicht angegeben ist, wird er auf 4326 gesetzt.

Wenn wkt_string null ist, wird null zurückgegeben.

Wenn wkt_string nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert ein Polygon aus einem Geografieobjekt mit einem SRID-Wert.

```
SELECT ST_AsEWKT(ST_GeogFromText('SRID=4324;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))'));
```

```
  st_asewkt
-----------------------------------------------
 SRID=4324;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))
```

Das folgende SQL konstruiert ein Polygon aus einem Geografieobjekt. Der SRID-Wert wurde auf 4326 gesetzt.

```
SELECT ST_AsEWKT(ST_GeogFromText('POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))'));
```

```
 st_asewkt
-----------------------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))
```

GeogFromST_ WKB

ST_ GeogFrom WKB konstruiert ein Geografieobjekt aus einer hexadezimalen WKB-Darstellung (Wellknown Binary) einer Eingabegeografie.

Syntax

```
ST_GeogFromWKB(wkb_string)
```

Argumente

wkb_string

Ein Wert vom Datentyp VARCHAR, der eine hexadezimale WKB-Darstellung einer Geografie ist.

Rückgabetyp

GEOGRAPHY

Wenn der SRID-Wert angegeben ist, wird er auf den angegeben Wert gesetzt. Wenn der SRID-Wert nicht angegeben ist, wird er auf 4326 gesetzt.

Wenn wkb_string null ist, dann wird null zurückgegeben.

Wenn wkb_string nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert eine Geografie aus einem hexadezimalen WKB-Wert.

```
SELECT
 ST_AsEWKT(ST_GeogFromWKB('0103000000010000000500000000000000000000000000000000000000000000000000F03F000000000000F03F000000000000F03F000000000000000000000000000000000000000000000000'));
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_GeometryN

ST_GeometryN gibt eine Geometrie zurück, auf die der Eingabeindex der Eingabegeometrie zeigt, wie folgt:

- Wenn es sich bei der Eingabe um einen Punkt, Linestring oder ein Polygon handelt, wird die Geometrie unverändert zurückgegeben, wenn der Index gleich eins ist (1) und als null, wenn der Index ein anderer als eins (1) ist.
- Wenn es sich bei der Eingabe um eine Multipoint-, Multilinestring-, Multipolygon- oder Geometriesammlung handelt, wird eine Punkt-, Linestring-, Polygon- oder Geometriesammlung zurückgegeben, auf die ein Eingabeindex zeigt.

Der Index ist eins-basiert. Der räumliche Referenzsystem-Bezeichner (SRID) des Ergebnisses entspricht dem der Eingabegeometrie. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_GeometryN(geom, index)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

index

Ein Wert des Datentyps INTEGER, der die Position eines eins-basierten Indexes darstellt.

Rückgabetyp

GEOMETRY

Wenn geom oder index null ist, wird null zurückgegeben.

Wenn der Index außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die Geometrien in einer Geometriesammlung zurück.

```
WITH tmp1(idx) AS (SELECT 1 UNION SELECT 2),
tmp2(g) AS (SELECT ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0
 0)),LINESTRING(20 10,20 0,10 0))'))
SELECT idx, ST_AsEWKT(ST_GeometryN(g, idx)) FROM tmp1, tmp2 ORDER BY idx;
```

```
 idx |          st_asewkt
-----+----------------------------
   1 | POLYGON((0 0,10 0,0 10,0 0))
   2 | LINESTRING(20 10,20 0,10 0)
```

ST_ GeometryType

ST_ GeometryType gibt den Untertyp einer Eingabegeometrie als Zeichenfolge zurück.

Für 3DM-, 3DZ- und 4D-Geometrieeingaben gibt ST_ dasselbe Ergebnis GeometryType zurück wie für 2D-Geometrieeingaben.

Syntax

```
ST_GeometryType(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

VARCHAR stellt den Subtyp von geom dar.

Wenn geom null ist, wird null zurückgegeben.

Die zurückgegebenen Werte lauten wie folgt.

Zurückgegebener String-Wert	Geometrie-Subtyp
ST_Point	Wird zurückgegeben, wenn geom ein POINT-Subtyp ist.
ST_LineString	Wird zurückgegeben, wenn geom ein LINESTRING -Subtyp ist.
ST_Polygon	Wird zurückgegeben, wenn geom ein POLYGON-Subtyp ist.
ST_MultiPoint	Wird zurückgegeben, wenn geom ein MULTIPOINT -Subtyp ist.

Zurückgegebener String-Wert	Geometrie-Subtyp
ST_MultiLineString	Wird zurückgegeben, wenn geom ein MULTILINESTRING -Subtyp ist.
ST_MultiPolygon	Wird zurückgegeben, wenn geom ein MULTIPOLYGON -Subtyp ist.
ST_GeometryCollection	Wird zurückgegeben, wenn geom ein GEOMETRYCOLLECTION -Subtyp ist.

Beispiele

Das folgende SQL gibt den Subtyp der Eingangs-Linestring-Geometrie zurück.

```
SELECT ST_GeometryType(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_geometrytype
-------------
 ST_LineString
```

ST_ EWKB GeomFrom

ST_ GeomFrom EWKB konstruiert ein Geometrieobjekt aus der erweiterten, bekannten Binärdarstellung (EWKB) einer Eingabegeometrie.

ST_ GeomFrom EWKB akzeptiert 3DZ-, 3DM- und 4D-Geometrien, die im Hexadezimalformat WKB und EWKB geschrieben sind.

Syntax

```
ST_GeomFromEWKB(ewkb_string)
```

Argumente

ewkb_string

Ein Wert vom Datentyp VARCHAR, der eine hexadezimale EWKB-Darstellung einer Geometrie ist.

Rückgabetyp

GEOMETRY

Wenn ewkb_string null ist, dann wird null zurückgegeben.

Wenn ewkb_string nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert ein Polygon aus einem EWKB-Wert und gibt die EWKT-Darstellung eines Polygons zurück.

```
SELECT
 ST_AsEWKT(ST_GeomFromEWKB('0103000020E6100000010000000500000000000000000000000000000000000000000
```

```
 st_asewkt
-------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_ GeomFrom EWKT

ST_ GeomFrom EWKT konstruiert ein Geometrieobjekt aus der erweiterten EWKT-Darstellung (Extended Well-known Text) einer Eingabegeometrie.

ST_ GeomFrom EWKT akzeptiert 3DZ, 3DM und 4D, wobei dem Geometrietyp jeweils Z, M oder ZM vorangestellt ist.

Syntax

```
ST_GeomFromEWKT(ewkt_string)
```

Argumente

ewkt_string

Ein Wert vom Datentyp VARCHAR oder ein Ausdruck, der zu einem VARCHAR-Typ ausgewertet wird, der eine EWKT-Darstellung einer Geometrie ist.

Sie können das WKT-Schlüsselwort EMPTY verwenden, um einen leeren Punkt, einen Multipoint mit einem leeren Punkt oder eine Geometriesammlung mit einem leeren Punkt festzulegen. Im folgenden Beispiel wird ein leerer Punkt erstellt.

```
ST_GeomFromEWKT('SRID=4326;POINT EMPTY');
```

Rückgabetyp

GEOMETRY

Wenn ewkt_string null ist, wird null zurückgegeben.

Wenn ewkt_string nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung konstruiert einen Multilinestring aus einem EWKT-Wert und gibt eine Geometrie zurück. Außerdem wird das ST_AsEWKT-Ergebnis der Geometrie zurückgegeben.

```
SELECT ST_GeomFromEWKT('SRID=4326;MULTILINESTRING((1 0,1 0),(2 0,3 0),(4 0,5 0,6 0))')
 as geom, ST_AsEWKT(geom);
```

```
                                                    geom

                                        |                        st_asewkt

-------------------------------------------------------------------------------------------
+----------------------------------------------------------
 0105000020E6100000030000000102000000020000000000000000F03F0000000000000000000000000000F03F00
 | SRID=4326;MULTILINESTRING((1 0,1 0),(2 0,3 0),(4 0,5 0,6 0))
```

ST_ GeomFromGeoHash

ST_ GeomFromGeoHash konstruiert ein Geometrieobjekt aus der Geohash-Darstellung einer Eingabegeometrie. ST_ GeomFromGeoHash gibt eine zweidimensionale (2D) Geometrie mit dem Raumbezugsidentifikator (SRID) von Null (0) zurück. Weitere Informationen zum Geohash-Format finden Sie unter Geohash bei Wikipedia.

Syntax

```
ST_GeomFromGeoHash(geohash_string)
```

```
ST_GeomFromGeoHash(geohash_string, precision)
```

Argumente

geohash_string

Ein Wert vom Datentyp VARCHAR oder ein Ausdruck, der in einen Typ VARCHAR ausgewertet wird, der eine Geohash-Darstellung einer Geometrie ist.

precision

Ein Wert vom Datentyp INTEGER, der die Präzision des Geohashs darstellt. Der Wert ist die Anzahl der Zeichen des Geohashs, die als Präzision verwendet werden sollen. Wenn der Wert nicht angegeben ist, kleiner als Null oder größer als die Länge von geohash_string ist, wird die Länge von geohash_string verwendet.

Rückgabetyp

GEOMETRY

Wenn geohash_string null ist, wird null zurückgegeben.

Wenn geohash_string nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL gibt ein Polygon mit hoher Präzision zurück.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz0'));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.172816 36.114646,-115.172816 36.114646,-115.172816 36.114646,-115.172816
 36.114646,-115.172816 36.114646))
```

Die folgende SQL gibt einen Punkt mit hoher Präzision zurück.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz00'));
```

```
 st_asewkt
-----------------------
 POINT(-115.172816 36.114646)
```

Die folgende SQL gibt ein Polygon mit geringer Präzision zurück.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qq'));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.3125 35.15625,-115.3125 36.5625,-113.90625 36.5625,-113.90625
 35.15625,-115.3125 35.15625))
```

Die folgende SQL gibt ein Polygon mit Präzision 3 zurück.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz0', 3));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.3125 35.15625,-115.3125 36.5625,-113.90625 36.5625,-113.90625
 35.15625,-115.3125 35.15625))
```

GeomFromGeoST_ JSON

ST_ GeomFromGeo JSON konstruiert ein Geometrieobjekt aus der GeoJSON-Darstellung einer Eingabegeometrie. Weitere Informationen zum GeoJSON-Format finden Sie unter GeoJSON bei Wikipedia.

Wenn es mindestens einen Punkt mit drei oder mehr Koordinaten gibt, ist die resultierende Geometrie 3DZ, wobei die Z-Komponente für die Punkte null ist, die nur zwei Koordinaten haben. Wenn alle

Punkte in der Eingabe-GeoJSON zwei Koordinaten enthalten oder leer sind, gibt ST_ GeomFromGeo JSON eine 2D-Geometrie zurück. Der Spatial-Reference-Identifier-Wert (SRID) der zurückgegebenen Geometrie ist immer 4326.

Syntax

```
ST_GeomFromGeoJSON(geojson_string)
```

Argumente

geojson_string

Ein Wert vom Datentyp VARCHAR oder ein Ausdruck, der zu einem VARCHAR-Typ ausgewertet wird, der eine GeoJSON-Darstellung einer Geometrie ist.

Rückgabetyp

GEOMETRY

Wenn geojson_string null ist, wird null zurückgegeben.

Wenn geojson_string nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine 2D-Geometrie zurück, die in der GeoJSON-Eingabegeometrie dargestellt wird.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"Point","coordinates":[1,2]}'));
```

```
 st_asewkt
-----------------------
 SRID=4326;POINT(1 2)
```

Die folgende SQL-Anweisung gibt eine 3DZ-Geometrie zurück, die in der GeoJSON-Eingabegeometrie dargestellt wird.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"LineString","coordinates":[[1,2,3],
[4,5,6],[7,8,9]]}'));
```

```
 st_asewkt
-----------------------------------------
 SRID=4326;LINESTRING Z (1 2 3,4 5 6,7 8 9)
```

Die folgende SQL gibt eine 3DZ-Geometrie zurück, wenn nur ein Punkt drei Koordinaten hat, während alle anderen Punkte zwei Koordinaten in der GeoJSON-Eingabegeometrie haben.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"Polygon","coordinates":[[[0, 0],[0, 1,
 8],[1, 0],[0, 0]]]}'));
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;POLYGON Z ((0 0 0,0 1 8,1 0 0,0 0 0))
```

ST_ GeomFromGeoSquare

ST_ GeomFromGeoSquare gibt eine Geometrie zurück, die den Bereich abdeckt, der durch einen Geoquadrat-Eingabewert repräsentiert wird. Die zurückgegebene Geometrie ist immer zweidimensional. Informationen zur Berechnung eines GeoSquare-Werts finden Sie unter ST_ GeoSquare.

Syntax

```
ST_GeomFromGeoSquare(geosquare)
```

```
ST_GeomFromGeoSquare(geosquare, max_depth)
```

Argumente

geosquare

Ein Wert des Datentyps `BIGINT` oder ein Ausdruck, der zu einem `BIGINT`-Typ ausgewertet wird, bei dem es sich um einen GeoSquare-Wert handelt. Dieser Wert beschreibt die Reihenfolge der Unterteilungen, die an der ursprünglichen Domain vorgenommen wurden, um das gewünschte Rechteck zu erreichen. Dieser Wert wird berechnet durch ST_ GeoSquare.

max_depth

Ein Wert des Datentyps `INTEGER`, der die maximale Anzahl von Domain-Unterteilungen darstellt, die an der ursprünglichen Domain vorgenommen wurden. Der Wert muss gleich oder größer als 1 sein.

Rückgabetyp

`GEOMETRY`

Wenn geosquare nicht gültig ist, gibt die Funktion einen Fehler zurück.

Wenn die Eingabe max_depth nicht innerhalb des zulässigen Bereichs liegt, gibt die Funktion einen Fehler zurück.

Beispiele

Die folgende SQL-Anweisung gibt eine Geometrie aus einem GeoSquare-Wert zurück.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(797852));
```

```
 st_astext
-----------------------------------------------------------------------------------------------
 POLYGON((13.359375 52.3828125,13.359375 52.734375,13.7109375 52.734375,13.7109375
 52.3828125,13.359375 52.3828125))
```

Die folgende SQL-Anweisung gibt eine Geometrie aus einem GeoSquare-Wert und einer maximalen Tiefe von 3 zurück.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(797852, 3));
```

```
 st_astext
-------------------------------------
 POLYGON((0 45,0 90,45 90,45 45,0 45))
```

Die folgende SQL-Anweisung berechnet zunächst den GeoSquare-Wert für Seattle, indem die X-Koordinate als Längengrad und die Y-Koordinate als Breitengrad (-122,3; 47,6) angegeben wird.

Dann gibt sie das Polygon für das GeoSquare zurück. Obwohl die Ausgabe eine zweidimensionale Geometrie ist, kann sie zur Berechnung raumbezogener Daten in Bezug auf Längen- und Breitengrad verwendet werden.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(ST_GeoSquare(ST_Point(-122.3, 47.6))));
```

```
 st_astext
-----------------------------------------------------------------------------------------------------------
POLYGON((-122.335167014971 47.6080129947513,-122.335167014971
 47.6080130785704,-122.335166931152 47.6080130785704,-122.335166931152
 47.6080129947513,-122.335167014971 47.6080129947513))
```

ST_ GeomFromText

ST_ GeomFromText konstruiert ein Geometrieobjekt aus einer WKT-Darstellung (Wellknown Text) einer Eingabegeometrie.

ST_ GeomFromText akzeptiert 3DZ, 3DM und 4D, wobei dem Geometrietyp jeweils Z, M oder ZM vorangestellt ist.

Syntax

```
ST_GeomFromText(wkt_string)
```

```
ST_GeomFromText(wkt_string, srid)
```

Argumente

wkt_string

Ein Wert vom Datentyp VARCHAR, der eine WKT-Darstellung einer Geometrie ist.

Sie können das WKT-Schlüsselwort EMPTY verwenden, um einen leeren Punkt, einen Multipoint mit einem leeren Punkt oder eine Geometriesammlung mit einem leeren Punkt festzulegen. Im folgenden Beispiel wird ein Multipoint mit einem leeren und einem nicht leeren Punkt erstellt.

```
ST_GeomFromEWKT('MULTIPOINT(1 0,EMPTY)');
```

srid

Ein Wert vom Datentyp `INTEGER`, der eine SRID (Spatial Reference Identifier) ist. Wenn ein SRID-Wert angegeben wird, hat die zurückgegebene Geometrie diesen SRID-Wert. Andernfalls wird der Spatial Reference System Identifier (SRID)-Wert der zurückgegebenen Geometrie auf null (0) festgelegt.

Rückgabetyp

`GEOMETRY`

Wenn wkt_string oder srid null ist, wird null zurückgegeben.

Wenn srid negativ ist, wird null zurückgegeben.

Wenn wkt_string nicht gültig ist, wird ein Fehler zurückgegeben.

Wenn srid nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert ein Geometrieobjekt aus der WKT-Darstellung und dem SRID-Wert.

```
SELECT ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326);
```

```
st_geomfromtext
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F
```

GeomFromST_ WKB

ST_ GeomFrom WKB konstruiert ein Geometrieobjekt aus einer hexadezimalen, wohlbekannten Binärdarstellung (WKB) einer Eingabegeometrie.

ST_ GeomFrom WKB akzeptiert 3DZ-, 3DM- und 4D-Geometrien, die im Hexadezimalformat WKB geschrieben sind.

Syntax

```
ST_GeomFromWKB(wkb_string)
```

```
ST_GeomFromWKB(wkb_string, srid)
```

Argumente

wkb_string

Ein Wert vom Datentyp VARCHAR, der eine hexadezimale WKB-Darstellung einer Geometrie ist.

srid

Ein Wert vom Datentyp INTEGER, der eine SRID (Spatial Reference Identifier) ist. Wenn ein SRID-Wert angegeben wird, hat die zurückgegebene Geometrie diesen SRID-Wert. Andernfalls wird der SRID-Wert der zurückgegebenen Geometrie auf 0 gesetzt.

Rückgabetyp

GEOMETRY

Wenn wkb_string oder srid null ist, wird null zurückgegeben.

Wenn srid negativ ist, wird null zurückgegeben.

Wenn wkb_string nicht gültig ist, wird ein Fehler zurückgegeben.

Wenn srid nicht gültig ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert ein Polygon aus einem WKB-Wert und gibt die WKT-Darstellung eines Polygons zurück.

```
SELECT
 ST_AsText(ST_GeomFromWKB('010300000001000000050000000000000000000000000000000000000000000000
```

```
 st_astext
-------------------------------
 POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_ GeoSquare

ST_ unterteilt die Domäne ([-180, 180], [-90, 90]) GeoSquare rekursiv bis zu einer bestimmten Tiefe in gleiche quadratische Bereiche, die als Geosquare bezeichnet werden. Die Unterteilung basiert auf der Position eines angegebenen Punkts. Eines der GeoSquares, das den Punkt enthält, wird bei jedem Schritt unterteilt, bis die maximale Tiefe erreicht ist. Die Auswahl dieses GeoSquares ist stabil, das heißt, das Funktionsergebnis hängt nur von den Eingabeargumenten ab. Die Funktion gibt einen eindeutigen Wert zurück, der das endgültige GeoSquare identifiziert, in dem sich der Punkt befindet.

ST_ GeoSquare akzeptiert einen PUNKT, wobei die X-Koordinate den Längengrad und die Y-Koordinate den Breitengrad darstellt. Der Längen- und der Breitengrad sind auf [-180, 180] bzw. [-90, 90] begrenzt. Die Ausgabe von ST_ GeoSquare kann als Eingabe für die ST_ GeomFromGeoSquare Funktion verwendet werden.

Der Bogen des äquatorialen Umfangs der Erde beträgt 360° und ist in zwei Hemisphären (Ost- und Westhalbkugel) unterteilt, von denen jede vom 0°-Meridian aus 180°-Längslinien (Meridiane) aufweist. Konventionell sind die östlichen Längengrade „+“-Koordinaten (positiv), wenn sie auf der kartesischen Ebene auf eine X-Achse projiziert werden, und die westlichen Längengrade sind „-“-Koordinaten (negativ), wenn sie auf der kartesischen Ebene auf eine X-Achse projiziert werden. Es gibt 90°-Breitengrade nördlich und südlich des 0°-Äquatorialumfangs der Erde, die jeweils parallel zum 0°-Äquatorumfang der Erde verlaufen. Konventionell kreuzen die nördlichen Breitengrade die (positive) „+“-Y-Achse, wenn sie auf die kartesische Ebene projiziert werden, und die südlichen Breitengrade kreuzen die (negative) „-“-Y-Achse, wenn sie auf die kartesische Ebene projiziert werden. Das Kugelraster, das durch die Kreuzung von Längs- und Breitengraden entsteht, wird in ein auf die kartesische Ebene projiziertes Raster mit positiven und negativen X-Koordinaten sowie positiven und negativen Y-Koordinaten auf der kartesischen Ebene umgewandelt.

Der Zweck von ST_ besteht GeoSquare darin, nahe gelegene Punkte mit gleichen Codewerten zu kennzeichnen oder zu markieren. Punkte, die sich auf demselben GeoSquare befinden, erhalten den gleichen Codewert. Ein GeoSquare wird verwendet, um geografische Koordinaten (Breitengrad und Längengrad) in eine Ganzzahl zu codieren. Eine größere Region wird in Raster unterteilt, um ein Gebiet auf einer Karte mit unterschiedlichen Auflösungen abzugrenzen. Ein GeoSquare kann für die räumliche Indexierung, die räumliche Aufteilung, die Umkreissuche, Standortsuche und die Erstellung eindeutiger Ortskennungen verwendet werden. Die Funktion ST_ GeoHash folgt einem ähnlichen Prozess der Aufteilung einer Region in Raster, hat jedoch eine andere Codierung.

Syntax

```
ST_GeoSquare(geom)
```

```
ST_GeoSquare(geom, max_depth)
```

Argumente

geom

Ein PUNKT-Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem PUNKT-Untertyp ausgewertet wird. Die X-Koordinate (Längengrad) des Punkts muss innerhalb dieses Bereichs liegen: `-180`-`180`. Die Y-Koordinate (Breitengrad) des Punkts muss innerhalb dieses Bereichs liegen: `-90`-`90`.

max_depth

Ein Wert vom Datentyp `INTEGER`. Gibt an, wie oft die Domäne, die den Punkt enthält, maximal rekursiv unterteilt wird. Der Wert muss eine Ganzzahl zwischen 1 und 32 sein. Der Standardwert ist 32. Die tatsächliche endgültige Anzahl der Unterteilungen ist kleiner oder gleich des festgelegten Werts für max_depth.

Rückgabetyp

`BIGINT`

Die Funktion gibt einen eindeutigen Wert zurück, der das endgültige GeoSquare identifiziert, in dem sich der Eingabepunkt befindet.

Wenn die Eingabegeometrie kein Punkt ist, gibt die Funktion einen Fehler zurück.

Wenn der Eingabepunkt leer ist, ist der Rückgabewert keine gültige Eingabe für die ST_ GeomFromGeoSquare-Funktion. Verwenden Sie die ST_ IsEmpty Funktion, um Aufrufe von ST_ GeoSquare mit einem leeren Punkt zu verhindern.

Wenn der Eingabepunkt nicht innerhalb des zulässigen Bereichs liegt, gibt die Funktion einen Fehler zurück.

Wenn die Eingabe max_depth außerhalb des zulässigen Bereichs liegt, gibt die Funktion einen Fehler zurück.

Beispiele

Die folgende SQL-Anweisung gibt ein GeoSquare aus einem Eingabepunkt zurück.

```
SELECT ST_GeoSquare(ST_Point(13.5, 52.5));
```

```
  st_geosquare
-----------------------
 -4410772491521635895
```

Die folgende SQL-Anweisung gibt ein GeoSquare aus einem Eingabepunkt mit einer maximalen Tiefe von 10 zurück.

```
SELECT ST_GeoSquare(ST_Point(13.5, 52.5), 10);
```

```
 st_geosquare
--------------
 797852
```

ST_ N InteriorRing

ST_ InteriorRing N gibt eine geschlossene Linienfolge zurück, die dem inneren Ring eines Eingabepolygons an der Indexposition entspricht. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_InteriorRingN(geom, index)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

index

Ein Wert des Datentyps `INTEGER`, der die Position eines Rings eines eins-basierten Indexes darstellt.

Rückgabetyp

`GEOMETRY` des Subtyps `LINESTRING`.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn geom oder index null ist, wird null zurückgegeben.

Wenn der Index außerhalb des Bereichs liegt, wird null zurückgegeben.

Wenn geom kein Polygon ist, wird null zurückgegeben.

Wenn geom ein leeres Polygon ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt den zweiten Ring eines Polygons als geschlossenen Linestring zurück.

```
SELECT ST_AsEWKT(ST_InteriorRingN(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17
 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12
 14,15 14,13 11,12 14))'),2));
```

```
st_asewkt
-----------
 LINESTRING(12 14,15 14,13 11,12 14)
```

ST_Intersects

ST_Intersects gibt true zurück, wenn die 2D-Projektionen der beiden Eingabegeometrien mindestens einen gemeinsamen Punkt haben.

Syntax

```
ST_Intersects(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das erste Polygon das zweite Polygon schneidet.

```
SELECT ST_Intersects(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(2 2,2 5,5 5,5
 2,2 2))'), ST_GeomFromText('MULTIPOINT((4 4),(6 6))'));
```

```
st_intersects
-------------
 true
```

ST_Intersection

ST_Intersection gibt eine Geometrie zurück, die die Punktmengenschnittmenge zweier Geometrien darstellt. ST_Intersection gibt also den Teil der beiden Eingabegeometrien zurück, der zwischen ihnen geteilt wird.

Syntax

```
ST_Intersection(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Wenn geom1 und geom2 keinen Bereich teilen (sie also getrennt sind), wird eine leere Geometrie zurückgegeben.

Wenn geom1 und geom2 leer sind, wird eine leere Geometrie zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 keine zweidimensionale Geometrie (2D) ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt eine nicht leere Geometrie zurück, die die Schnittmenge von zwei Eingabegeometrien darstellt.

```
SELECT ST_AsEWKT(ST_Intersection(ST_GeomFromText('polygon((0 0,100 100,0 200,0 0))'),
 ST_GeomFromText('polygon((0 0,10 0,0 10,0 0))')));
```

```
        st_asewkt
-------------------------
 POLYGON((0 0,0 10,5 5,0 0))
```

Das folgende SQL gibt eine leere Geometrie zurück, wenn getrennte (sich nicht überschneidende) Eingabegeometrien übergeben wurden.

```
SELECT ST_AsEWKT(ST_Intersection(ST_GeomFromText('linestring(0 100,0 0)'),
 ST_GeomFromText('polygon((1 0,10 0,1 10,1 0))')));
```

```
    st_asewkt
------------------
 LINESTRING EMPTY
```

ST_ IsPolygon CCW

ST_ IsPolygon CCW gibt true zurück, wenn die 2D-Projektion des Eingabepolygons oder Multipolygons gegen den Uhrzeigersinn erfolgt. Wenn es sich bei der Eingabegeometrie um einen Punkt, Linestring, Multipoint oder Multilinestring handelt, wird true zurückgegeben. Bei Geometriesammlungen gibt ST_ IsPolygon CCW den Wert true zurück, wenn alle Geometrien in der Sammlung gegen den Uhrzeigersinn verlaufen.

Syntax

```
ST_IsPolygonCCW(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`BOOLEAN`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung prüft, ob das Polygon gegen den Uhrzeigersinn ausgerichtet ist.

```
SELECT ST_IsPolygonCCW(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17 7,17 10,18
 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12 14,15 14,13
 11,12 14))'));
```

```
 st_ispolygonccw
----------
 true
```

IsPolygonST_ CW

ST_ IsPolygon CW gibt true zurück, wenn die 2D-Projektion des Eingabepolygons oder Multipolygons im Uhrzeigersinn erfolgt. Wenn es sich bei der Eingabegeometrie um einen Punkt, Linestring, Multipoint oder Multilinestring handelt, wird true zurückgegeben. Bei Geometriesammlungen gibt ST_ IsPolygon CW true zurück, wenn sich alle Geometrien in der Sammlung im Uhrzeigersinn befinden.

Syntax

```
ST_IsPolygonCW(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`BOOLEAN`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung prüft, ob das Polygon mit dem Uhrzeigersinn ausgerichtet ist.

```
SELECT ST_IsPolygonCW(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17 7,17 10,18
 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12 14,15 14,13
 11,12 14))'));
```

```
 st_ispolygonccw
----------
 true
```

ST_ IsClosed

ST_ IsClosed gibt true zurück, wenn die 2D-Projektion der Eingabegeometrie geschlossen ist. Die folgenden Regeln definieren eine geschlossene Geometrie:

- Die Eingangsgeometrie ist ein Punkt oder ein Multipunkt.
- Die Eingangsgeometrie ist ein Linestring, und der Anfangs- und Endpunkt des Linestrings stimmen überein.
- Die Eingangsgeometrie ist ein nicht leerer Multilinestring und alle seine Linestrings sind geschlossen.
- Die Eingangsgeometrie ist ein nicht leeres Polygon, alle Ringe des Polygons sind nicht leer, und der Anfangs- und Endpunkt aller seiner Ringe stimmen überein.
- Die Eingangsgeometrie ist ein nicht leeres Multipolygon und alle seine Polygone sind geschlossen.
- Die Eingangsgeometrie ist eine nicht leere Geometrie-Collection und alle ihre Komponenten sind geschlossen.

Syntax

```
ST_IsClosed(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom ein leerer Punkt ist, wird false zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das Polygon geschlossen ist.

```
SELECT ST_IsClosed(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_isclosed
-----------
 true
```

ST_ IsCollection

ST_ IsCollection gibt true zurück, wenn es sich bei der Eingabegeometrie um einen der folgenden Subtypen handelt:GEOMETRYCOLLECTION,, MULTIPOINT oder. MULTILINESTRING MULTIPOLYGON

Syntax

```
ST_IsCollection(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das Polygon eine Collections ist.

```
SELECT ST_IsCollection(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_iscollection
-----------
 false
```

ST_ IsEmpty

ST_ IsEmpty gibt true zurück, wenn die Eingabegeometrie leer ist. Eine Geometrie ist nicht leer, wenn sie mindestens einen nicht leeren Punkt enthält.

ST_ IsEmpty gibt true zurück, wenn die Eingabegeometrie mindestens einen nicht leeren Punkt enthält.

Syntax

```
ST_IsEmpty(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`BOOLEAN`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das angegebene Polygon leer ist.

```
SELECT ST_IsEmpty(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_isempty
-----------
 false
```

ST_ IsRing

ST_ IsRing gibt true zurück, wenn es sich bei der eingegebenen Linienfolge um einen Ring handelt. Ein Linestring ist ein Ring, wenn er geschlossen und einfach ist.

Syntax

```
ST_IsRing(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Die Geometrie muss ein `LINESTRING` sein.

Rückgabetyp

`BOOLEAN`

Wenn geom kein `LINESTRING` ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung prüft, ob der angegebene Linestring ein Ring ist.

```
SELECT ST_IsRing(ST_GeomFromText('linestring(0 0, 1 1, 1 2, 0 0)'));
```

```
st_isring
-----------
```

```
 true
```

ST_ IsSimple

ST_ IsSimple gibt true zurück, wenn die 2D-Projektion der Eingabegeometrie einfach ist. Weitere Informationen zur Definition einer einfachen Geometrie finden Sie unter Geometrische Einfachheit.

Syntax

```
ST_IsSimple(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung prüft, ob der angegebene Linestring einfach ist. In diesem Beispiel ist er nicht einfach, da eine Selbstüberschneidung vorliegt.

```
SELECT ST_IsSimple(ST_GeomFromText('LINESTRING(0 0,10 0,5 5,5 -5)'));
```

```
 st_issimple
-----------
 false
```

ST_ IsValid

ST_ IsValid gibt true zurück, wenn die 2D-Projektion der Eingabegeometrie gültig ist. Weitere Informationen zur Definition einer gültigen Geometrie finden Sie unter Geometrische Gültigkeit.

Syntax

```
ST_IsValid(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung prüft, ob das angegebene Polygon leer ist. In diesem Beispiel ist das Polygon ungültig, da das Innere des Polygons nicht einfach verbunden ist.

```
SELECT ST_IsValid(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(5 0,10 5,5 10,0
 5,5 0))'));
```

```
 st_isvalid
------------
 false
```

ST_Length

Für eine lineare Geometrie gibt ST_Length die kartesische Länge der 2D-Projektion zurück. Die Längeneinheiten entsprechen den Einheiten, in denen die Koordinaten der Eingabegeometrie ausgedrückt werden. Die Funktion gibt Null (0) für Punkte, Multipoints und Flächengeometrien zurück. Wenn es sich bei der Eingabe um eine Geometriesammlung handelt, gibt die Funktion die Summe der Längen der Geometrien in der Sammlung zurück.

Für eine Geografie gibt ST_Length die geodätische Länge der 2D-Projektion einer linearen Eingabegeografie zurück, die auf dem vom SRID festgelegten Sphäroiden berechnet wurde.

Die Länge wird in Meter angegeben. Die Funktion gibt Null (0) für Punkte, Multipoints und Flächengeografien zurück. Wenn es sich bei der Eingabe um eine Geometriesammlung handelt, gibt die Funktion die Summe der Längen der Geografien in der Sammlung zurück.

Syntax

```
ST_Length(geo)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION

Wenn geo null ist, wird null zurückgegeben.

Wenn der SRID-Wert nicht gefunden werden kann, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die kartesische Länge eines Multi-Linestrings zurück.

```
SELECT ST_Length(ST_GeomFromText('MULTILINESTRING((0 0,10 0,0 10),(10 0,20 0,20
 10))'));
```

```
st_length
--------------------------------
  44.142135623731
```

Die folgende SQL-Anweisung gibt die Länge eines Linestrings in einer Geografie zurück.

```
SELECT ST_Length(ST_GeogFromText('SRID=4326;LINESTRING(5 0,6 0,4 0)'));
```

```
 st_length
------------------
 333958.472379804
```

Die folgende SQL-Anweisung gibt die Länge eines Punkts in einer Geografie zurück.

```
SELECT ST_Length(ST_GeogFromText('SRID=4326;POINT(4 5)'));
```

```
 st_length
-----------
 0
```

ST_ LengthSphere

ST_ LengthSphere gibt die Länge einer linearen Geometrie in Metern zurück. Für Punkt-, Mehrpunkt- und Flächengeometrien gibt ST_ 0 zurück. LengthSphere Für Geometriesammlungen gibt ST_ die Gesamtlänge der linearen Geometrien in der Sammlung in Metern LengthSphere zurück.

ST_ LengthSphere interpretiert die Koordinaten der einzelnen Punkte der Eingabegeometrie als Längen- und Breitengrad in Grad. Für 3DZ-, 3DM- oder 4D-Geometrien werden nur die ersten beiden Koordinaten verwendet.

Syntax

```
ST_LengthSphere(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`DOUBLE PRECISION`-Länge in Metern. Die Länge der Berechnung basiert auf dem Kugel-Modell der Erde, wobei der Radius dem Radius-Mittelwert der Erde aus der WGS 84-Darstellung (World Geodetic System) des Ellipsoid-Modells der Erde entspricht.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Im folgenden Beispiel berechnet SQL die Länge des Linestrings in Metern.

```
SELECT ST_LengthSphere(ST_GeomFromText('LINESTRING(10 10,45 45)'));
```

```
 st_lengthsphere
------------------
 5127736.08292556
```

ST_Length2D

ST_Length2d ist ein Alias für ST_Length. Weitere Informationen finden Sie unter ST_Length.

ST_ LineFromMultiPoint

ST_ LineFromMultiPoint gibt eine Linienfolge aus einer Eingabe-Multipoint-Geometrie zurück. Die Reihenfolge der Punkte bleibt erhalten. Der Spatial Reference System Identifier (SRID)-Wert der zurückgegebenen Geometrie entspricht dem der Eingabegeometrie. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_LineFromMultiPoint(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `MULTIPOINT`.

Rückgabetyp

`GEOMETRY`

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird ein leerer LINESTRING zurückgegeben.

Wenngeom leere Punkte enthält, werden diese leeren Punkte ignoriert.

Wenn geom kein MULTIPOINT ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung erstellt einen Linestring aus einem Multipoint.

```
SELECT ST_AsEWKT(ST_LineFromMultiPoint(ST_GeomFromText('MULTIPOINT(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
 st_asewkt
--------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5,0 5)
```

ST_ LineInterpolatePoint

ST_ LineInterpolatePoint gibt einen Punkt auf einer Linie zurück, der sich in einem Bruchteil der Entfernung vom Anfang der Linie befindet.

Um die Punktgleichheit zu bestimmen, verwendet LineInterpolatePoint ST_ die 2D-Projektion der Eingabegeometrie. Wenn die Eingabegeometrie leer ist, wird eine Kopie dieser Geometrie in derselben Dimension wie die Eingabe zurückgegeben. Bei 3DZ-, 3DM- und 4D-Geometrien ist die z- oder m-Koordinate der Durchschnitt der z- oder m-Koordinaten des Segments in dem sich der Punkt befindet.

Syntax

```
ST_LineInterpolatePoint(geom, fraction)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Der Subtyp ist LINESTRING.

fraction

Ein Wert des Datentyps `DOUBLE PRECISION`, der die Position eines Punktes entlang des Linestrings für die Linie darstellt. Der Wert ist ein Bruchteil im Bereich 0–1.

Rückgabetyp

`GEOMETRY` des Subtyps `POINT`.

Wenn geom oder fraction null ist, wird null zurückgegeben.

Wenn geom leer ist, wird der leere Punkt zurückgegeben.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Wenn fraction außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben.

Wenn geom kein Linestring ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt einen Punkt auf halber Strecke entlang eines Linestrings zurück.

```
SELECT ST_AsEWKT(ST_LineInterpolatePoint(ST_GeomFromText('LINESTRING(0 0, 5 5, 7 7, 10
 10)'), 0.50));
```

```
st_asewkt
-----------
 POINT(5 5)
```

Die folgende SQL-Anweisung gibt einen Punkt auf 90 % der Strecke entlang eines Linestrings zurück.

```
SELECT ST_AsEWKT(ST_LineInterpolatePoint(ST_GeomFromText('LINESTRING(0 0, 5 5, 7 7, 10
 10)'), 0.90));
```

```
st_asewkt
-----------
 POINT(9 9)
```

ST_M

ST_M gibt die m-Koordinate eines Eingabepunktes zurück.

Syntax

```
ST_M(point)
```

Argumente

point

Ein POINT-Wert vom Datentyp GEOMETRY.

Rückgabetyp

DOUBLE PRECISION-Wert der m-Koordinate.

Wenn Punkt null ist, wird null zurückgegeben.

Wenn point ein 2D- oder 3DZ-Punkt ist, wird null zurückgegeben.

Wenn point ein leerer Punkt ist, wird null zurückgegeben.

Wenn point kein POINT ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die m-Koordinate eines Punktes in einer 3DM-Geometrie zurück.

```
SELECT ST_M(ST_GeomFromEWKT('POINT M (1 2 3)'));
```

```
st_m
-----------
 3
```

Die folgende SQL-Anweisung gibt die m-Koordinate eines Punktes in einer 4D-Geometrie zurück.

```
SELECT ST_M(ST_GeomFromEWKT('POINT ZM (1 2 3 4)'));
```

```
st_m
-----------
 4
```

ST_ MakeEnvelope

ST_ MakeEnvelope gibt eine Geometrie wie folgt zurück:

- Wenn die Eingabekoordinaten einen Punkt angeben, ist die zurückgegebene Geometrie ein Punkt.
- Wenn die Eingabekoordinaten eine Linie angeben, ist die zurückgegebene Geometrie ein Linestring.
- Andernfalls ist die zurückgegebene Geometrie ein Polygon, bei dem die Eingabekoordinaten die untere linke und obere rechte Ecke eines Quaders angeben.

Sofern angegeben, wird der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie auf den eingegeben SRID-Wert gesetzt.

Syntax

```
ST_MakeEnvelope(xmin, ymin, xmax, ymax)
```

```
ST_MakeEnvelope(xmin, ymin, xmax, ymax, srid)
```

Argumente

xmin

Ein Wert vom Datentyp `DOUBLE PRECISION`. Dieser Wert entspricht der ersten Koordinate in der unteren linken Ecke eines Quaders.

ymin

Ein Wert vom Datentyp `DOUBLE PRECISION`. Dieser Wert entspricht der zweiten Koordinate in der unteren linken Ecke eines Quaders.

xmax

Ein Wert vom Datentyp `DOUBLE PRECISION`. Dieser Wert entspricht der ersten Koordinate in der oberen rechten Ecke eines Quaders.

ymax

Ein Wert vom Datentyp `DOUBLE PRECISION`. Dieser Wert entspricht der zweiten Koordinate in der oberen rechten Ecke eines Quaders.

srid

Ein Wert vom Datentyp `INTEGER`, der einen Spatial Reference System Identifier (SRID) darstellt. Wenn der SRID-Wert nicht angegeben wird, wird er auf null gesetzt.

Rückgabetyp

`GEOMETRY` des Subtyps `POINT`, `LINESTRING` oder `POLYGON`.

Die SRID der zurückgegebenen Geometrie wird auf `srid` festgelegt, bzw. null, wenn `srid` nicht festgelegt wurde.

Wenn xmin, ymin, xmax, ymax oder srid null ist, wird null zurückgegeben.

Wenn srid negativ ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt ein Polygon zurück, das ein Envelope darstellt, das von den vier eingegebenen Koordinatenwerten definiert wird.

```
SELECT ST_AsEWKT(ST_MakeEnvelope(2,4,5,7));
```

```
 st_astext
---------------
 POLYGON((2 4,2 7,5 7,5 4,2 4))
```

Die folgende SQL-Anweisung gibt ein Polygon zurück, das ein Envelope darstellt, das von den vier eingegebenen Koordinatenwerten und einem SRID-Wert definiert wird.

```
SELECT ST_AsEWKT(ST_MakeEnvelope(2,4,5,7,4326));
```

```
 st_astext
---------------------------------
```

```
SRID=4326;POLYGON((2 4,2 7,5 7,5 4,2 4))
```

ST_ MakeLine

ST_ MakeLine erstellt eine Linienfolge aus den Eingabegeometrien.

Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrien. Beide Eingabegeometrien müssen die gleiche Dimension haben.

Syntax

```
ST_MakeLine(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss `POINT`, `LINESTRING` oder `MULTIPOINT` sein.

geom2

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss `POINT`, `LINESTRING` oder `MULTIPOINT` sein.

Rückgabetyp

`GEOMETRY` des Subtyps `LINESTRING`.

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 der leere Punkt ist oder leere Punkte enthält, werden diese leeren Punkte ignoriert.

Wenn geom1 und geom2 leer sind, wird ein leerer `LINESTRING` zurückgegeben.

Der SRID-Wert der zurückgegebenen Geometrie ist der SRID-Wert der Eingangsgeometrien.

Wenn geom1 und geom2 unterschiedliche SRID-Werte haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 kein `POINT`, `LINESTRING`, oder `MULTIPOINT` ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 unterschiedliche Dimensionen haben, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert einen Linestring aus zwei Eingangs-Linestrings.

```
SELECT ST_MakeLine(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'), ST_GeomFromText('LINESTRING(88.29 39.07,88.42 39.26,88.27
 39.31,88.29 39.07)'));
```

```
st_makeline
-----------
 0102000000080000000C3F5285C8F52534052B81E85EB113D407B14AE47E15A5340C3F5285C8F423D40E17A14AE4751
```

ST_ MakePoint

ST_ MakePoint gibt eine Punktgeometrie zurück, deren Koordinatenwerte die Eingabewerte sind.

Syntax

```
ST_MakePoint(x, y)
```

```
ST_MakePoint(x, y, z)
```

```
ST_MakePoint(x, y, z, m)
```

Argumente

x

Ein Wert vom Datentyp DOUBLE PRECISION, der die erste Koordinate darstellt.

y

Ein Wert vom Datentyp DOUBLE PRECISION, der die zweite Koordinate darstellt.

z

Ein Wert vom Datentyp DOUBLE PRECISION, der die dritte Koordinate darstellt.

i

Ein Wert vom Datentyp DOUBLE PRECISION, der die vierte Koordinate darstellt.

Rückgabetyp

`GEOMETRY` des Subtyps `POINT`.

Der SRID-Wert der zurückgegebenen Geometrie wird auf 0 gesetzt.

Wenn x, y, z oder m null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt einen `GEOMETRY`-Typ des Subtyps `POINT` mit den angegebenen Koordinaten zurück.

```
SELECT ST_AsText(ST_MakePoint(1,3));
```

```
st_astext
-----------
 POINT(1 3)
```

Das folgende SQL gibt einen `GEOMETRY`-Typ des Subtyps `POINT` mit den angegebenen Koordinaten zurück.

```
SELECT ST_AsEWKT(ST_MakePoint(1, 2, 3));
```

```
st_asewkt
----------------
 POINT Z (1 2 3)
```

Das folgende SQL gibt einen `GEOMETRY`-Typ des Subtyps `POINT` mit den angegebenen Koordinaten zurück.

```
SELECT ST_AsEWKT(ST_MakePoint(1, 2, 3, 4));
```

```
st_asewkt
-------------------
```

```
POINT ZM (1 2 3 4)
```

ST_ MakePolygon

ST_ MakePolygon hat zwei Varianten, die ein Polygon zurückgeben. Eine nimmt eine einzelne Geometrie an und eine andere nimmt zwei Geometrien an.

- Die Eingabe der ersten Variante ist eine Linestring, der den äußeren Ring des Ausgabepolygons definiert.
- Die Eingabe der zweiten Variante ist ein Linestring und ein Multilinestring. Beide sind leer oder geschlossen.

  Die Begrenzung des äußeren Rings des Ausgabepolygons ist der Eingabe-Linestring. Die Begrenzungen der inneren Ringe des Polygons sind die Linestrings im Eingabe-Multilinestring. Wenn die Eingabe-Linestring leer ist, wird ein leeres Polygon zurückgegeben. Leere Linestrings im Multilinestring werden ignoriert. Der SRID-Wert (Spatial Reference System Identifier) der resultierenden Geometrie ist der gemeinsame SRID-Wert der beiden Eingabegeometrien.

Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrien. Der Außenring und die Innenringe müssen die gleiche Dimension haben.

Syntax

```
ST_MakePolygon(geom1)
```

```
ST_MakePolygon(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `LINESTRING`. Der Wert linestring muss geschlossen werden oder leer sein.

geom2

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `MULTILINESTRING`.

Rückgabetyp

GEOMETRY des Subtyps POLYGON.

Der SRID (Spatial Reference System Identifier) der zurückgegebenen Geometrie entspricht der SRID der Eingaben.

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 kein Linestring ist, wird ein Fehler zurückgegeben.

Wenn geom2 kein Multilinestring ist, wird ein Fehler zurückgegeben.

Wenn geom1 nicht geschlossen ist, wird ein Fehler zurückgegeben.

Wenn geom1 ein einzelner Punkt oder nicht geschlossen ist, wird ein Fehler zurückgegeben.

Wenn geom2 mindestens einen Linestring hat, der einen einzelnen Punkt hat oder nicht geschlossen ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 unterschiedliche SRID-Werte haben, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 unterschiedliche Dimensionen haben, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt ein Polygon aus einem Eingangs-Linestring zurück.

```
SELECT ST_AsText(ST_MakePolygon(ST_GeomFromText('LINESTRING(77.29 29.07,77.42
 29.26,77.27 29.31,77.29 29.07)')));
```

```
 st_astext
---------------
POLYGON((77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07))
```

Die folgende SQL-Anweisung erstellt ein Polygon aus einem geschlossenen Linestring und einem geschlossenen Multilinestring. Der Linestring wird für den äußeren Ring des Polygons verwendet. Die Linestrings in den Multilinestrings werden für die Innenringe des Polygons verwendet.

```
SELECT ST_AsEWKT(ST_MakePolygon(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,0 10,0 0)'),
 ST_GeomFromText('MULTILINESTRING((1 1,1 2,2 1,1 1),(3 3,3 4,4 3,3 3))')));
```

```
 st_astext
----------------------------------
POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 1,1 1),(3 3,3 4,4 3,3 3))
```

ST_ MemSize

ST_ MemSize gibt die Menge an Speicherplatz (in Byte) zurück, die von der Eingabegeometrie verwendet wird. Diese Größe hängt von der Amazon-Redshift-internen Darstellung der Geometrie ab und kann sich somit ändern, wenn sich die interne Darstellung ändert. Sie können diese Größe als Angabe der relativen Größe von Geometrieobjekten in Amazon Redshift verwenden.

Syntax

```
ST_MemSize(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`INTEGER` stellt die inhärente Dimension von geom dar.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die Speichergröße einer Geometrie-Auflistung zurück.

```
SELECT ST_MemSize(ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0
 0)),LINESTRING(20 10,20 0,10 0))')::varchar + ' bytes';
```

```
 ?column?
-----------
```

```
 172 bytes
```

ST_MMax

ST_MMax gibt die maximale m-Koordinate einer Eingabegeometrie zurück.

Syntax

```
ST_MMax(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der maximalen m-Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom eine 2D- oder 3DZ-Geometrie ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die größte m-Koordinate eines Linestrings in einer 3DM-Geometrie zurück.

```
SELECT ST_MMax(ST_GeomFromEWKT('LINESTRING M (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_mmax
-----------
  8
```

Die folgende SQL-Anweisung gibt die größte m-Koordinate eines Linestrings in einer 4D-Geometrie zurück.

```
SELECT ST_MMax(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_mmax
-----------
  11
```

ST_MMin

ST_MMin gibt die minimale m-Koordinate einer Eingabegeometrie zurück.

Syntax

```
ST_MMin(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der minimalen m-Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom eine 2D- oder 3DZ-Geometrie ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die kleinste m-Koordinate eines Linestrings in einer 3DM-Geometrie zurück.

```
SELECT ST_MMin(ST_GeomFromEWKT('LINESTRING M (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_mmin
-----------
  2
```

Die folgende SQL-Anweisung gibt die kleinste m-Koordinate eines Linestrings in einer 4D-Geometrie zurück.

```
SELECT ST_MMin(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_mmin
-----------
  3
```

ST_Multi

ST_Multi konvertiert eine Geometrie in den entsprechenden Multityp. Wenn es sich bei der Eingabegeometrie bereits um einen Multityp oder eine Geometriesammlung handelt, wird eine Kopie zurückgegeben. Wenn es sich bei der Eingabegeometrie um einen Punkt, einen Linestring oder ein Polygon handelt, wird dementsprechend ein Multipoint, ein Multilinestring oder ein Multipolygon zurückgegeben, das/der die Eingabegeometrie enthält.

Syntax

```
ST_Multi(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY mit Subtyp MULTIPOINT, MULTILINESTRING, MULTIPOLYGON oder GEOMETRYCOLLECTION.

Der Spatial Reference System Identifier (SRID)-Wert der zurückgegebenen Geometrie entspricht dem der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt einen Multipoint aus einem eingegebenen Multipoint zurück.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('MULTIPOINT((1 2),(3 4))', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;MULTIPOINT((1 2),(3 4))
```

Die folgende SQL-Anweisung gibt einen Multipoint aus einem eingegebenen Punkt zurück.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('POINT(1 2)', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;MULTIPOINT((1 2))
```

Die folgende SQL-Anweisung gibt eine Geometriesammlung aus einer eingegebenen Geometriesammlung zurück.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('GEOMETRYCOLLECTION(POINT(1 2),MULTIPOINT((1
 2),(3 4)))', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;GEOMETRYCOLLECTION(POINT(1 2),MULTIPOINT((1 2),(3 4)))
```

ST_NDims

ST_NDIMS gibt die Koordinatendimension einer Geometrie zurück. ST_NDIMS berücksichtigt die topologische Dimension einer Geometrie nicht. Stattdessen wird ein konstanter Wert zurückgegeben, der von der Dimension der Geometrie abhängt.

Syntax

```
ST_NDims(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

INTEGER stellt die inhärente Dimension von geom dar.

Wenn geom null ist, wird null zurückgegeben.

Die zurückgegebenen Werte lauten wie folgt.

Zurückgegebener Wert	Dimension der Eingabegeometrie
2	2D
3	3DZ oder 3DM
4	4D

Beispiele

Die folgende SQL-Anweisung gibt die Anzahl der Dimensionen eines 2D-Linestrings zurück.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING(0 0,1 1,2 2,0 0)'));
```

```
st_ndims
-------------
 2
```

Die folgende SQL-Anweisung gibt die Anzahl der Dimensionen eines 3DZ-Linestrings zurück.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING Z(0 0 3,1 1 3,2 2 3,0 0 3)'));
```

```
st_ndims
-------------
 3
```

Die folgende SQL-Anweisung gibt die Anzahl der Dimensionen eines 3DM-Linestrings zurück.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING M(0 0 4,1 1 4,2 2 4,0 0 4)'));
```

```
st_ndims
-------------
 3
```

Die folgende SQL-Anweisung gibt die Anzahl der Dimensionen eines 4D-Linestrings zurück.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING ZM(0 0 3 4,1 1 3 4,2 2 3 4,0 0 3 4)'));
```

```
st_ndims
-------------
 4
```

ST_NPoints

ST_NPoints gibt die Anzahl der nicht leeren Punkte in einer Eingabegeometrie oder -geografie zurück.

Syntax

```
ST_NPoints(geo)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Wenn geo ein leerer Punkt ist, wird 0 zurückgegeben.

Wenn geo null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die Anzahl der Punkte in einem Linestring zurück.

```
SELECT ST_NPoints(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_npoints
-------------
 4
```

Das folgende SQL gibt die Anzahl der Punkte in einem Linestring in einer Geografie zurück.

```
SELECT ST_NPoints(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
st_npoints
-------------
 4
```

ST_NRings

ST_NRings gibt die Anzahl der Ringe in einer Eingangsgeometrie zurück.

Syntax

```
ST_NRings(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Wenn geom null ist, wird null zurückgegeben.

Die zurückgegebenen Werte lauten wie folgt.

Zurückgegebener Wert	Geometrie-Subtyp
0	Wird zurückgegeben, wenn geom ein POINT-, LINESTRING -, MULTIPOINT - oder MULTILINESTRING -Subtyp ist.
Die Anzahl der Ringe.	Wird zurückgegeben, wenn geom ein POLYGON- oder MULTIPOLYGON -Subtyp ist.
Die Anzahl der Ringe in allen Komponenten	Wird zurückgegeben, wenn geom ein GEOMETRYCOLLECTION -Subtyp ist.

Beispiele

Das folgende SQL gibt die Anzahl der Ringe in einem Multipolygon zurück.

```
SELECT ST_NRings(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((0 0,-10 0,0
 -10,0 0)))'));
```

```
 st_nrings
-------------
 2
```

ST_ NumGeometries

ST_ NumGeometries gibt die Anzahl der Geometrien in einer Eingabegeometrie zurück.

Syntax

```
ST_NumGeometries(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

INTEGER stellt die Anzahl der Geometrien in geom dar.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom eine einzelne leere Geometrie ist, wird 0 zurückgegeben.

Wenn geom eine einzelne nicht leere Geometrie ist, wird 1 zurückgegeben.

Wenn geom ein GEOMETRYCOLLECTION- oder MULTI-Subtyp ist, wird die Anzahl der Geometrien zurückgegeben.

Beispiele

Das folgende SQL gibt die Anzahl der Geometrien im Eingangs-Multilinestring zurück.

```
SELECT ST_NumGeometries(ST_GeomFromText('MULTILINESTRING((0 0,1 0,0 5),(3 4,13 26))'));
```

```
st_numgeometries
-------------
```

```
 2
```

ST_ NumInteriorRings

ST_ NumInteriorRings gibt die Anzahl der Ringe in einer Eingabe-Polygongeometrie zurück.

Syntax

```
ST_NumInteriorRings(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Wenn geom null ist, wird null zurückgegeben.

Wenn geom kein Polygon ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die Anzahl der Innenringe im Eingangspolygon zurück.

```
SELECT ST_NumInteriorRings(ST_GeomFromText('POLYGON((0 0,100 0,100 100,0 100,0 0),(1
 1,1 5,5 1,1 1),(7 7,7 8,8 7,7 7))'));
```

```
 st_numinteriorrings
-------------
 2
```

ST_ NumPoints

ST_ NumPoints gibt die Anzahl der Punkte in einer Eingabegeometrie zurück.

Syntax

```
ST_NumPoints(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Wenn geom null ist, wird null zurückgegeben.

Wenn geom nicht vom Subtyp LINESTRING ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die Anzahl der Punkte in der Eingangs-Linestring zurück.

```
SELECT ST_NumPoints(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_numpoints
-------------
4
```

Die folgende SQL gibt null zurück, da das Eingabe-geom nicht vom Subtyp LINESTRING ist.

```
SELECT ST_NumPoints(ST_GeomFromText('MULTIPOINT(1 2,3 4)'));
```

```
st_numpoints
-------------
```

ST_Perimeter

Für eine Eingabeflächengeometrie gibt ST_Perimeter den kartesischen Umfang (Länge der Grenze) der 2D-Projektion zurück. Die Umfangseinheiten entsprechen den Einheiten, in denen die Koordinaten der Eingabegeometrie ausgedrückt werden. Die Funktion gibt Null (0) für Punkte, Multipoints und lineare Geometrien zurück. Wenn es sich bei der Eingabe um eine Geometriesammlung handelt, gibt die Funktion die Summe der Umfänge der Geometrien in der Sammlung zurück.

Für eine Eingabegeografie gibt ST_Perimeter den geodätische Umfang (Länge der Grenze) der 2D-Projektion einer Eingabeflächengeografie zurück, die auf dem vom SRID festgelegten Sphäroiden berechnet wurde. Der Umfang wird in Meter angegeben. Die Funktion gibt Null (0) für Punkte, Multipoints und lineare Geografien zurück. Wenn es sich bei der Eingabe um eine Geometriesammlung handelt, gibt die Funktion die Summe der Umfänge der Geografien in der Sammlung zurück.

Syntax

```
ST_Perimeter(geo)
```

Argumente

geo

Ein Wert vom Datentyp GEOMETRY oder GEOGRAPHY oder ein Ausdruck, der zu einem GEOMETRY- oder GEOGRAPHY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION

Wenn geo null ist, wird null zurückgegeben.

Wenn der SRID-Wert nicht gefunden werden kann, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt den kartesischen Umfang eines Multipolygons zurück.

```
SELECT ST_Perimeter(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20
 10,10 0)))'));
```

```
 st_perimeter
------------------------------
    68.2842712474619
```

Die folgende SQL-Anweisung gibt den kartesischen Umfang eines Multipolygons zurück.

```
SELECT ST_Perimeter(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20
 10,10 0)))'));
```

```
 st_perimeter
------------------------------
    68.2842712474619
```

Die folgende SQL-Anweisung gibt den Umfang eines Polygons in einer Geografie zurück.

```
SELECT ST_Perimeter(ST_GeogFromText('SRID=4326;POLYGON((0 0,1 0,0 1,0 0))'));
```

```
 st_perimeter
------------------
 378790.428393693
```

Die folgende SQL-Anweisung gibt den Umfang eines Linestrings in einer Geografie zurück.

```
SELECT ST_Perimeter(ST_GeogFromText('SRID=4326;LINESTRING(5 0,10 0)'));
```

```
 st_perimeter
--------------
 0
```

ST_Perimeter2D

ST_Perimeter2d ist ein Alias für ST_Perimeter. Weitere Informationen finden Sie unter ST_Perimeter.

ST_Point

ST_Point gibt eine Punktgeometrie aus den eingegebenen Koordinatenwerten zurück.

Syntax

```
ST_Point(x, y)
```

Argumente

x

Ein Wert vom Datentyp DOUBLE PRECISION, der eine erste Koordinate darstellt.

y

Ein Wert vom Datentyp DOUBLE PRECISION, der eine zweite Koordinate darstellt.

Rückgabetyp

GEOMETRY des Subtyps POINT.

Der SRID-Wert der zurückgegebenen Geometrie wird auf 0 gesetzt.

Wenn x oder y null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL konstruiert aus den Eingangskoordinaten eine Punktgeometrie.

```
SELECT ST_AsText(ST_Point(5.0, 7.0));
```

```
st_astext
-------------
POINT(5 7)
```

ST_PointN

ST_PointN gibt einen Punkt in einem Linestring zurück, wie durch einen Indexwert angegeben. Negative Indexwerte werden vom Ende des Linestrings rückwärts gezählt, so dass -1 der letzte Punkt ist.

Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_PointN(geom, index)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Der Subtyp muss sein LINESTRING.

index

Ein Wert des Datentyps INTEGER, der den Index eines Punktes in einem Linestring darstellt.

Rückgabetyp

GEOMETRY des Subtyps POINT.

Der SRID-Wert der zurückgegebenen Geometrie wird auf 0 gesetzt.

Wenn geom oder index null ist, wird null zurückgegeben.

Wenn der Index außerhalb des Bereichs liegt, wird null zurückgegeben.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom kein LINESTRING ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine erweiterte Extended Well-known text (EWKT)-Repräsentation eines Sechspunkt-LINESTRING zu einem GEOMETRY-Objekt zurück und gibt den Punkt bei Index 5 des Linestrings zurück.

```
SELECT ST_AsEWKT(ST_PointN(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5,0
 0)',4326), 5));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 5)
```

ST_Points

ST_Points gibt eine Multipoint-Geometrie zurück, die alle nicht leeren Punkte in der Eingabegeometrie enthält. ST_Points entfernt keine Punkte, die in der Eingabe dupliziert sind, einschließlich der Start- und Endpunkte von Ringgeometrien.

Syntax

```
ST_Points(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`GEOMETRY` des Subtyps `MULTIPOINT`.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie entspricht geom.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird der leere Multipoint zurückgegeben.

Beispiele

Die folgenden SQL-Beispiele konstruieren eine Multipoint-Geometrie aus der Eingabegeometrie. Das Ergebnis ist eine Multipoint-Geometrie, die die nicht leeren Punkte in der Eingabegeometrie enthält.

```
SELECT ST_AsEWKT(ST_Points(ST_SetSRID(ST_GeomFromText('LINESTRING(1 0,2 0,3 0)'),
 4326)));
```

```
st_asewkt
-------------
SRID=4326;MULTIPOINT((1 0),(2 0),(3 0))
```

```
SELECT ST_AsEWKT(ST_Points(ST_SetSRID(ST_GeomFromText('MULTIPOLYGON(((0 0,1 0,0 1,0
 0)))'), 4326)));
```

```
st_asewkt
-------------
SRID=4326;MULTIPOINT((0 0),(1 0),(0 1),(0 0))
```

ST_Polygon

ST_Polygon gibt eine Polygongeometrie zurück, deren äußerer Ring der Eingangs-Linestring mit dem Wert ist, der für die SRID eingegeben wurde.

Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_Polygon(linestring, srid)
```

Argumente

linestring

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss `LINESTRING` sein, der einen Linestring darstellt. Der Wert linestring muss geschlossen werden.

srid

Ein Wert von Datentyp `INTEGER`, der eine SRID darstellt.

Rückgabetyp

`GEOMETRY` des Subtyps `POLYGON`.

Der SRID-Wert der zurückgegebenen Geometrie wird auf srid gesetzt.

Wenn linestring oder srid null ist, wird null zurückgegeben.

Wenn linestring kein Linestring ist, wird ein Fehler zurückgegeben.

Wenn linestring nicht geschlossen ist, wird ein Fehler zurückgegeben.

Wenn srid negativ ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL konstruiert ein Polygon mit einem SRID-Wert.

```
SELECT ST_AsEWKT(ST_Polygon(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'),4356));
```

```
st_asewkt
-------------
 SRID=4356;POLYGON((77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07))
```

ST_ RemovePoint

ST_ RemovePoint gibt eine Linienkettengeometrie zurück, bei der der Punkt der Eingabegeometrie an einer Indexposition entfernt wurde.

Der Index ist null-basiert. Die SRID des Ergebnisses entspricht der Eingabegeometrie. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_RemovePoint(geom, index)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `LINESTRING`.

index

Ein Wert des Datentyps `INTEGER`. der die Position eines null-basierten Indexes darstellt.

Rückgabetyp

`GEOMETRY`

Wenn geom oder index null ist, wird null zurückgegeben.

Wenn geom kein Subtyp LINESTRING ist, wird ein Fehler zurückgegeben.

Wenn der Index außerhalb des Bereichs liegt, wird ein Fehler zurückgegeben. Gültige Werte für die Indexposition liegen zwischen 0 und ST_NumPoints(geom) minus 1.

Beispiele

Die folgende SQL-Anweisung entfernt den letzten Punkt in einem Linestring.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_RemovePoint(g, ST_NumPoints(g) - 1)) FROM tmp;
```

```
   st_asewkt
----------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5)
```

ST_Reverse

ST_Reverse kehrt die Reihenfolge der Eckpunkte für lineare und Flächengeometrien um. Bei Punkt- oder Multipoint-Geometrien wird eine Kopie der ursprünglichen Geometrie zurückgegeben. ST_Reverse kehrt bei Geometriesammlungen die Reihenfolge der Eckpunkte für jede der Geometrien in der Sammlung um.

Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_Reverse(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Der Spatial Reference System Identifier (SRID)-Wert der zurückgegebenen Geometrie entspricht dem der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die Anzahl der Punkte in einem Linestring zurück.

```
SELECT ST_AsEWKT(ST_Reverse(ST_GeomFromText('LINESTRING(1 0,2 0,3 0,4 0)', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;LINESTRING(4 0,3 0,2 0,1 0)
```

ST_ SetPoint

ST_ SetPoint gibt eine Linienfolge mit aktualisierten Koordinaten in Bezug auf die Position der Eingabelinienfolge zurück, wie sie durch den Index angegeben ist. Die neuen Koordinaten sind die Koordinaten des Eingabepunkts.

Die Dimension der zurückgegebenen Geometrie entspricht der des geom1-Werts. Wenn geom1 und geom2 unterschiedliche Dimensionen haben, wird geom2 auf die Dimension von geom1 projiziert.

Syntax

```
ST_SetPoint(geom1, index, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `LINESTRING`.

index

Ein Wert des Datentyps `INTEGER`, der die Position eines Index darstellt. Eine `0` bezieht sich auf den ersten Punkt eines Linestrings von links, 1 bezieht sich auf den zweiten Punkt und so

weiter. Der Index kann ein negativer Wert sein. Eine `-1` bezieht sich auf den ersten Punkt eines Linestrings von rechts, `-2` bezieht sich auf den zweiten Punkt des Linestrings von rechts und so weiter.

geom2

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `POINT`.

Rückgabetyp

`GEOMETRY`

Wenn geom2 der leere Punkt ist, wird geom1 zurückgegeben.

Wenn geom1, geom2 oder index null ist, wird null zurückgegeben.

Wenn geom1 kein Linestring ist, wird ein Fehler zurückgegeben.

Wenn index nicht in einem gültigen Indexbereich ist, wird ein Fehler zurückgegeben.

Wenn geom2 kein Punkt ist, wird ein Fehler zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt einen neuen Linestring zurück, bei dem wir den zweiten Punkt des Eingabe-Linestrings mit dem angegebenen Punkt festgelegt haben.

```
SELECT ST_AsText(ST_SetPoint(ST_GeomFromText('LINESTRING(1 2, 3 2, 5 2, 1 2)'), 2,
 ST_GeomFromText('POINT(7 9)')));
```

```
st_astext
-------------
 LINESTRING(1 2,3 2,7 9,1 2)
```

Das folgende SQL-Beispiel gibt einen neuen Linestring zurück, bei dem wir den dritten Punkt von rechts (der Index ist negativ) des Linestrings mit dem angegebenen Punkt festgelegt haben.

```
SELECT ST_AsText(ST_SetPoint(ST_GeomFromText('LINESTRING(1 2, 3 2, 5 2, 1 2)'), -3,
 ST_GeomFromText('POINT(7 9)')));
```

```
st_astext
-------------
 LINESTRING(1 2,7 9,5 2,1 2)
```

ST_SetSRID

ST_SetSRID gibt eine Geometrie zurück, die der Eingangsgeometrie entspricht und mit der Werteingabe für die SRID aktualisiert ist.

Syntax

```
ST_SetSRID(geom, srid)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

srid

Ein Wert von Datentyp `INTEGER`, der eine SRID darstellt.

Rückgabetyp

`GEOMETRY`

Der SRID-Wert der zurückgegebenen Geometrie wird auf srid gesetzt.

Wenn geom oder srid null ist, wird null zurückgegeben.

Wenn srid negativ ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL legt den SRID-Wert eines Linestrings fest.

```
SELECT ST_AsEWKT(ST_SetSRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'),50));
```

```
st_asewkt
-------------
 SRID=50;LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07)
```

ST_Simplify

ST_Simplify gibt eine vereinfachte Kopie der Eingabegeometrie mit dem Ramer-Douglas-Peucker-Algorithmus mit der angegebenen Toleranz zurück. Die Topologie der Eingabegeometrie wird möglicherweise nicht beibehalten. Weitere Informationen finden Sie im Wikipedia-Artikel zum Douglas-Peucker-Algorithmus.

Wenn ST_Simplify Entfernungen berechnet, um eine Geometrie zu vereinfachen, arbeitet ST_Simplify mit der 2D-Projektion der Eingabegeometrie.

Syntax

```
ST_Simplify(geom, tolerance)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

tolerance

Ein Wert vom Datentyp `DOUBLE PRECISION`, der das Toleranzniveau eines Ramer-Douglas-Peucker-Algorithmus darstellt. Wenn tolerance eine negative Zahl ist, wird null verwendet.

Rückgabetyp

`GEOMETRY`.

Der SRID-Wert (Spatial Reference System Identifier) der zurückgegebenen Geometrie ist der SRID-Wert der Eingabegeometrie.

Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung vereinfacht den Eingabe-Linestring, indem für den euklidischen Abstand eine Toleranz von 1 mit dem Ramer-Douglas-Peucker-Algorithmus verwendet wird. Die Abstandseinheiten entsprechen denen der Koordinaten der Geometrie.

```
SELECT ST_AsEWKT(ST_Simplify(ST_GeomFromText('LINESTRING(0 0,1 2,1 1,2 2,2 1)'), 1));
```

```
 st_asewkt
-----------
LINESTRING(0 0,1 2,2 1)
```

ST_SRID

ST_SRID liefert die SRID einer Eingangsgeometrie. Weitere Informationen zum Erstellen einer SRID finden Sie unter Abfrage von Geodaten in Amazon Redshift.

Syntax

```
ST_SRID(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`INTEGER` stellt den SRID-Wert von geom dar.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt den SRID-Wert eines Linestrings zurück, der auf SRID `4326` festgelegt ist.

```
SELECT ST_SRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)',4326));
```

```
st_srid
-------------
 4326
```

Das folgende SQL gibt den SRID-Wert eines Linestrings zurück, der bei der Konstruktion nicht festgelegt ist. Das führt beim SRID-Wert zu `0`.

```
SELECT ST_SRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_srid
-------------
 0
```

ST_ StartPoint

ST_ StartPoint gibt den ersten Punkt einer Eingabelinienfolge zurück. Der Spatial Reference System Identifier (SRID)-Wert des Ergebnisses entspricht dem der Eingabegeometrie. Die Dimension der zurückgegebenen Geometrie entspricht der der Eingabegeometrie.

Syntax

```
ST_StartPoint(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird. Der Subtyp muss sein `LINESTRING`.

Rückgabetyp

`GEOMETRY`

Wenn geom null ist, wird null zurückgegeben.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom nicht LINESTRING ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt eine EWKT-Darstellung (Extended Well-known Text) eines Vierpunkt-LINESTRING zu einem GEOMETRY-Objekt und den Startpunkt des Linestrings zurück.

```
SELECT ST_AsEWKT(ST_StartPoint(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 0)
```

ST_Touches

ST_Touches gibt true zurück, wenn sich die 2D-Projektionen der beiden Eingabegeometrien berühren. Die beiden Geometrien berühren sich, wenn sie nicht leer sind, sich schneiden und keine inneren Punkte gemeinsam haben.

Syntax

```
ST_Touches(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

`BOOLEAN`

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-.Anweisung prüft, ob ein Polygon einen Linestring berührt.

```
SELECT ST_Touches(ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))'),
 ST_GeomFromText('LINESTRING(20 10,20 0,10 0)'));
```

```
 st_touches
-------------
 t
```

ST_Transform

ST_Transform gibt eine neue Geometrie mit Koordinaten zurück, die in einem vom SRID (Spatial Reference System Identifier) definierten Koordinatenreferenzsystem transformiert werden.

Syntax

```
ST_Transform(geom, srid)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

srid

Ein Wert von Datentyp `INTEGER`, der eine SRID darstellt.

Rückgabetyp

GEOMETRY.

Der SRID-Wert der zurückgegebenen Geometrie wird auf srid gesetzt.

Wenn geom oder srid null ist, wird null zurückgegeben.

Wenn der mit der Eingabe-geom verbundene SRID-Wert nicht existiert, wird ein Fehler zurückgegeben.

Wenn srid nicht existiert, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung transformiert den SRID-Wert einer leeren Geometriesammlung.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('GEOMETRYCOLLECTION EMPTY', 3857),
 4326));
```

```
            st_asewkt
----------------------------------
 SRID=4326;GEOMETRYCOLLECTION EMPTY
```

Die folgende SQL-Anweisung transformiert den SRID-Wert eines Linestrings.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9,
 -22 -33)', 4326), 26918));
```

```
    st_asewkt
-----------------------------------------------------------------------------------------
 SRID=26918;LINESTRING(73106.6977300955 15556182.9688576,14347201.5059964
 1545178.32934967,1515090.41262989 9522193.25115316,10491250.83295
 2575457.28410878,5672303.72135968 -5233682.61176205)
```

Die folgende SQL-Anweisung transformiert den SRID-Wert eines Polygons.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('POLYGON Z ((-10 10 -7, -65 10 -6, -10 64
 -5, -10 10 -7), (-11 11 5, -11 12 6, -12 11 7, -11 11 5))', 6989), 6317));
```

```
                                        st_asewkt
-------------------------------------------------------------------------------------------
 SRID=6317;POLYGON Z ((6186430.2771091 -1090834.57212608
 1100247.33216237,2654831.67853801 -5693304.90741276 1100247.50581055,2760987.41750022
 -486836.575101877 5709710.44137268,6186430.2771091 -1090834.57212608
 1100247.33216237),(6146675.25029258 -1194792.63532103 1209007.1115113,6125027.87562215
 -1190584.81194058 1317403.77865723,6124888.99555252 -1301885.3455052
 1209007.49312929,6146675.25029258 -1194792.63532103 1209007.1115113))
```

ST_Union

ST_Union gibt eine Geometrie zurück, die die Vereinigung zweier Geometrien darstellt. Die Eingabegeometrien werden also zusammengeführt, um eine daraus resultierende Geometrie ohne Überlappungen zu erzeugen.

Syntax

```
ST_Union(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

GEOMETRY

Der SRID-Wert der zurückgegebenen Geometrie ist der SRID-Wert der Eingangsgeometrien.

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 leer sind, wird eine leere Geometrie zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection, ein Linestring oder ein Multilinestring ist, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 keine zweidimensionale Geometrie (2D) ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL gibt eine nicht leere Geometrie zurück, die die Vereinigung von zwei Eingabegeometrien darstellt.

```
SELECT ST_AsEWKT(ST_Union(ST_GeomFromText('POLYGON((0 0,100 100,0 200,0 0))'),
 ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))')));
```

```
        st_asewkt
-------------------------
 POLYGON((0 0,0 200,100 100,5 5,10 0,0 0))
```

ST_Within

ST_Within gibt true zurück, wenn die 2D-Projektion der ersten Eingabegeometrie innerhalb der 2D-Projektion der zweiten Eingabegeometrie liegt.

Die Geometrie A liegt beispielsweise innerhalb der Geometrie B, wenn jeder Punkt in A ein Punkt in B ist und ihre Innenräume einen nicht leeren Schnittpunkt haben.

ST_Within(A, B) entspricht ST_Contains(B, A).

Syntax

```
ST_Within(geom1, geom2)
```

Argumente

geom1

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird. Dieser Wert wird mit geom2 verglichen, um festzustellen, ob er innerhalb von geom2 liegt.

geom2

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

BOOLEAN

Wenn geom1 oder geom2 null ist, wird null zurückgegeben.

Wenn geom1 und geom2 nicht den gleichen SRID-Wert (Spatial Reference System Identifier) haben, wird ein Fehler zurückgegeben.

Wenn geom1 oder geom2 eine Geometrie-Collection ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL prüft, ob das erste Polygon innerhalb des zweiten Polygons liegt.

```
SELECT ST_Within(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_within
-----------
 true
```

ST_X

ST_X gibt die erste Koordinate eines Eingangspunktes zurück.

Syntax

```
ST_X(point)
```

Argumente

point

Ein POINT-Wert vom Datentyp GEOMETRY.

Rückgabetyp

DOUBLE PRECISION-Wert der ersten Koordinate.

Wenn Punkt null ist, wird null zurückgegeben.

Wenn point ein leerer Punkt ist, wird null zurückgegeben.

Wenn point kein POINT ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die erste Koordinate eines Punktes zurück.

```
SELECT ST_X(ST_Point(1,2));
```

```
st_x
-----------
 1.0
```

ST_XMax

ST_XMax gibt die maximale erste Koordinate einer Eingangsgeometrie zurück.

Syntax

```
ST_XMax(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der maximalen ersten Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die größte erste Koordinate eines Linestrings zurück.

```
SELECT ST_XMax(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_xmax
-----------
 77.42
```

ST_XMin

ST_XMin gibt die minimale erste Koordinate einer Eingangsgeometrie zurück.

Syntax

```
ST_XMin(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der ersten minimalen Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die kleinste erste Koordinate eines Linestrings zurück.

```
SELECT ST_XMin(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_xmin
-----------
 77.27
```

ST_Y

ST_Y gibt die zweite Koordinate eines Eingangspunktes zurück.

Syntax

```
ST_Y(point)
```

Argumente

point

Ein `POINT`-Wert vom Datentyp `GEOMETRY`.

Rückgabetyp

`DOUBLE PRECISION`-Wert der zweiten Koordinate.

Wenn Punkt null ist, wird null zurückgegeben.

Wenn point ein leerer Punkt ist, wird null zurückgegeben.

Wenn point kein `POINT` ist, wird ein Fehler zurückgegeben.

Beispiele

Das folgende SQL gibt die zweite Koordinate eines Punktes zurück.

```
SELECT ST_Y(ST_Point(1,2));
```

```
st_y
-----------
 2.0
```

ST_YMax

ST_YMax gibt die maximale zweite Koordinate einer Eingangsgeometrie zurück.

Syntax

```
ST_YMax(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der maximalen zweiten Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die größte zweite Koordinate eines Linestrings zurück.

```
SELECT ST_YMax(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_ymax
-----------
```

```
 29.31
```

ST_YMin

ST_YMin gibt die minimale zweite Koordinate einer Eingangsgeometrie zurück.

Syntax

```
ST_YMin(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der minimalen zweiten Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Das folgende SQL gibt die kleinste zweite Koordinate eines Linestrings zurück.

```
SELECT ST_YMin(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_ymin
-----------
 29.07
```

ST_Z

ST_Z gibt die z-Koordinate eines Eingabepunktes zurück.

Syntax

```
ST_Z(point)
```

Argumente

point

Ein POINT-Wert vom Datentyp GEOMETRY.

Rückgabetyp

DOUBLE PRECISION-Wert der m-Koordinate.

Wenn Punkt null ist, wird null zurückgegeben.

Wenn point ein 2D- oder 3DM-Punkt ist, wird null zurückgegeben.

Wenn point ein leerer Punkt ist, wird null zurückgegeben.

Wenn point kein POINT ist, wird ein Fehler zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die z-Koordinate eines Punktes in einer 3DZ-Geometrie zurück.

```
SELECT ST_Z(ST_GeomFromEWKT('POINT Z (1 2 3)'));
```

```
st_z
-----------
 3
```

Die folgende SQL-Anweisung gibt die z-Koordinate eines Punktes in einer 4D-Geometrie zurück.

```
SELECT ST_Z(ST_GeomFromEWKT('POINT ZM (1 2 3 4)'));
```

```
st_z
-----------
 3
```

ST_ZMax

ST_ZMax gibt die maximale z-Koordinate einer Eingabegeometrie zurück.

Syntax

```
ST_ZMax(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der maximalen z-Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom eine 2D- oder 3DM-Geometrie ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die größte z-Koordinate eines Linestrings in einer 3DZ-Geometrie zurück.

```
SELECT ST_ZMax(ST_GeomFromEWKT('LINESTRING Z (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_zmax
-----------
  8
```

Die folgende SQL-Anweisung gibt die größte z-Koordinate eines Linestrings in einer 4D-Geometrie zurück.

```
SELECT ST_ZMax(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_zmax
-----------
  10
```

ST_ZMin

ST_ZMin gibt die minimale z-Koordinate einer Eingabegeometrie zurück.

Syntax

```
ST_ZMin(geom)
```

Argumente

geom

Ein Wert vom Datentyp GEOMETRY oder ein Ausdruck, der zu einem GEOMETRY-Typ ausgewertet wird.

Rückgabetyp

DOUBLE PRECISION-Wert der minimalen z-Koordinate.

Wenn geom leer ist, wird null zurückgegeben.

Wenn geom null ist, wird null zurückgegeben.

Wenn geom eine 2D- oder 3DM-Geometrie ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt die kleinste z-Koordinate eines Linestrings in einer 3DZ-Geometrie zurück.

```
SELECT ST_ZMin(ST_GeomFromEWKT('LINESTRING Z (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_zmin
-----------
  2
```

Die folgende SQL-Anweisung gibt die kleinste z-Koordinate eines Linestrings in einer 4D-Geometrie zurück.

```
SELECT ST_ZMin(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_zmin
-----------
  2
```

SupportsBBox

SupportSBBox gibt true zurück, wenn die Eingabegeometrie die Kodierung mit einem vorberechneten Begrenzungsrahmen unterstützt. Weitere Informationen zur Unterstützung von Bounding Boxes finden Sie unter Begrenzungsrahmen.

Syntax

```
SupportsBBox(geom)
```

Argumente

geom

Ein Wert vom Datentyp `GEOMETRY` oder ein Ausdruck, der zu einem `GEOMETRY`-Typ ausgewertet wird.

Rückgabetyp

`BOOLEAN`

Wenn geom null ist, wird null zurückgegeben.

Beispiele

Die folgende SQL-Anweisung gibt true zurück, da die Eingabepunktgeometrie die Kodierung mit einem Begrenzungsrahmen unterstützt.

```
SELECT SupportsBBox(AddBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
supportsbbox
--------------
t
```

Die folgende SQL-Anweisung gibt false zurück, da die Eingabepunktgeometrie die Kodierung mit einem Begrenzungsrahmen nicht unterstützt.

```
SELECT SupportsBBox(DropBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
supportsbbox
--------------
f
```

Zeichenfolgenfunktionen

Themen

Zeichenfolgefunktionen verarbeiten und bearbeiten Zeichenfolgen oder Ausdrücke, die zu Zeichenfolgen ausgewertet werden. Wenn das Argument string in diesen Funktionen ein Literalwert

ist, muss es in einfache Anführungszeichen eingeschlossen werden. Die unterstützten Datentypen sind CHAR und VARCHAR.

Im folgenden Abschnitt werden Funktionsnamen, Syntax und Beschreibungen der unterstützten Funktionen bereitgestellt. Alle Offsets in Zeichenfolgen sind eins-basiert.

Veraltete Funktionen, die ausschließlich für Führungsknoten gelten

Die folgenden Zeichenfolgefunktionen sind veraltet, da sie nur auf dem Führungsknoten ausgeführt werden. Weitere Informationen finden Sie unter Exklusive Führungsknotenfunktionen

- GET_BYTE
- SET_BIT
- SET_BYTE
- TO_ASCII

Der Operator || (Verkettung)

Verkettet zwei Ausdrücke auf beiden Seiten des Symbols `||` und gibt den verketteten Ausdruck zurück.

Ähnlich Funktion CONCAT.

> **Note**
>
> Wenn ein oder beide Ausdrücke null sind, ist das Ergebnis der Verkettung `NULL`.

Syntax

```
expression1 || expression2
```

Argumente

expression1

Eine `CHAR`-Zeichenfolge, eine `VARCHAR`-Zeichenfolge, ein binärer Ausdruck oder ein Ausdruck, der zu einem dieser Typen ausgewertet wird.

expression2

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge, ein binärer Ausdruck oder ein Ausdruck, der zu einem dieser Typen ausgewertet wird.

Rückgabetyp

Der Rückgabetyp der Zeichenfolge ist derselbe Typ wie die Eingabeargumente. Beim Verketten von zwei Zeichenfolgen vom Typ VARCHAR wird eine Zeichenfolge vom Typ VARCHAR zurückgegeben.

Beispiele

In den folgenden Beispielen werden die Tabellen USERS und VENUE aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Felder FIRSTNAME und LASTNAME aus der Tabelle USERS in der Beispieldatenbank zu verketten.

```
SELECT (firstname || ' ' || lastname) as fullname
FROM users
ORDER BY 1
LIMIT 10;

+-----------------+
|    fullname     |
+-----------------+
| Aaron Banks     |
| Aaron Booth     |
| Aaron Browning  |
| Aaron Burnett   |
| Aaron Casey     |
| Aaron Cash      |
| Aaron Castro    |
| Aaron Dickerson |
| Aaron Dixon     |
| Aaron Dotson    |
+-----------------+
```

Um Spalten zu verketten, die möglicherweise Null-Werte enthalten, verwenden Sie den Ausdruck NVL- und COALESCE-Funktionen. Im folgenden Beispiel wird NVL verwendet, um 0 zurückzugeben, wenn NULL gefunden wird.

```
SELECT (venuename || ' seats ' || NVL(venueseats, 0)) as seating
FROM venue
WHERE venuestate = 'NV' or venuestate = 'NC'
ORDER BY 1
LIMIT 10;

+-------------------------------------+
|               seating               |
+-------------------------------------+
| Ballys Hotel seats 0                |
| Bank of America Stadium seats 73298 |
| Bellagio Hotel seats 0              |
| Caesars Palace seats 0              |
| Harrahs Hotel seats 0               |
| Hilton Hotel seats 0                |
| Luxor Hotel seats 0                 |
| Mandalay Bay Hotel seats 0          |
| Mirage Hotel seats 0                |
| New York New York seats 0           |
+-------------------------------------+
```

Funktion ASCII

Die ASCII-Funktion gibt den ASCII-Code oder den Unicode-Codepunkt des ersten Zeichens in der von Ihnen angegebenen Zeichenfolge zurück. Wenn die Zeichenfolge leer ist, gibt die Funktion `0` zurück. Wenn die Zeichenfolge null ist, wird `NULL` zurückgegeben.

Syntax

```
ASCII('string')
```

Argument

string

Eine `CHAR`- oder `VARCHAR`-Zeichenfolge.

Rückgabetyp

INTEGER

Beispiele

Verwenden Sie das folgende Beispiel, um NULL zurückzugeben. Die NULLIF-Funktion gibt NULL zurück, wenn die beiden Argumente identisch sind. Daher lautet das Eingabeargument für die ASCII-Funktion NULL. Weitere Informationen finden Sie unter NULLIF-Funktion.

```
SELECT ASCII(NULLIF('',''));

+-------+
| ascii |
+-------+
|  NULL |
+-------+
```

Verwenden Sie das folgende Beispiel, um den ASCII-Code von 0 zurückzugeben.

```
SELECT ASCII('');

+-------+
| ascii |
+-------+
|     0 |
+-------+
```

Verwenden Sie das folgende Beispiel, um den ASCII-Code 97 für den ersten Buchstaben des Wortes amazon zurückzugeben.

```
SELECT ASCII('amazon');

+-------+
| ascii |
+-------+
|    97 |
+-------+
```

Verwenden Sie das folgende Beispiel, um den ASCII-Code 65 für den ersten Buchstaben des Wortes Amazon zurückzugeben.

```
SELECT ASCII('Amazon');

+-------+
```

```
| ascii |
+-------+
|    65 |
+-------+
```

Die Funktion BPCHARCMP

Vergleicht den Wert zweier Zeichenfolgen und gibt eine Ganzzahl aus. Wenn die Zeichenfolgen identisch sind, gibt die Funktion `0` zurück. Ist die erste Zeichenfolge alphabetisch größer, gibt die Funktion `1` zurück. Ist die zweite Zeichenfolge größer, gibt die Funktion `-1` zurück.

Im Fall von Multibyte-Zeichen basiert der Vergleich auf der Byte-Kodierung.

Synonym von Die Funktion BTTEXT_PATTERN_CMP.

Syntax

```
BPCHARCMP(string1, string2)
```

Argumente

string1

Eine `CHAR`- oder `VARCHAR`-Zeichenfolge.

string2

Eine `CHAR`- oder `VARCHAR`-Zeichenfolge.

Rückgabetyp

INTEGER

Beispiele

In den folgenden Beispielen wird die Tabelle USERS aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um für die ersten zehn Einträge in der USERS-Tabelle zu bestimmen, ob der Vorname eines Benutzers alphabetisch größer als der Nachname des Benutzers ist. Für Einträge, bei denen die Zeichenfolge für FIRSTNAME alphabetisch nach der Zeichenfolge für LASTNAME steht, gibt die Funktion 1 zurück. Wenn LASTNAME alphabetisch nach FIRSTNAME steht, gibt die Funktion `-1` zurück.

```
SELECT userid, firstname, lastname, BPCHARCMP(firstname, lastname)
FROM users
ORDER BY 1, 2, 3, 4
LIMIT 10;

+--------+-----------+-----------+-----------+
| userid | firstname | lastname  | bpcharcmp |
+--------+-----------+-----------+-----------+
|      1 | Rafael    | Taylor    |        -1 |
|      2 | Vladimir  | Humphrey  |         1 |
|      3 | Lars      | Ratliff   |        -1 |
|      4 | Barry     | Roy       |        -1 |
|      5 | Reagan    | Hodge     |         1 |
|      6 | Victor    | Hernandez |         1 |
|      7 | Tamekah   | Juarez    |         1 |
|      8 | Colton    | Roy       |        -1 |
|      9 | Mufutau   | Watkins   |        -1 |
|     10 | Naida     | Calderon  |         1 |
+--------+-----------+-----------+-----------+
```

Verwenden Sie das folgende Beispiel, um alle Einträge der Tabelle USERS zurückzugeben, für welche die Funktion `0` zurückgibt. Die Funktion gibt `0` zurück, wenn FIRSTNAME mit LASTNAME identisch ist.

```
SELECT userid, firstname, lastname,
BPCHARCMP(firstname, lastname)
FROM users
WHERE BPCHARCMP(firstname, lastname)=0
ORDER BY 1, 2, 3, 4;

+--------+-----------+----------+-----------+
| userid | firstname | lastname | bpcharcmp |
+--------+-----------+----------+-----------+
|     62 | Chase     | Chase    |         0 |
|   4008 | Whitney   | Whitney  |         0 |
|  12516 | Graham    | Graham   |         0 |
|  13570 | Harper    | Harper   |         0 |
|  16712 | Cooper    | Cooper   |         0 |
|  18359 | Chase     | Chase    |         0 |
|  27530 | Bradley   | Bradley  |         0 |
|  31204 | Harding   | Harding  |         0 |
+--------+-----------+----------+-----------+
```

Die Funktion BTRIM

Die BTRIM-Funktion kürzt eine Zeichenfolge durch Entfernen von Leerzeichen am Anfang und am Ende oder durch Entfernen von Zeichen am Anfang und am Ende, die mit einer optionalen angegebenen Zeichenfolge übereinstimmen.

Syntax

```
BTRIM(string [, trim_chars ] )
```

Argumente

string

Die VARCHAR-Eingabezeichenfolge, die gekürzt werden soll.

trim_chars

Die VARCHAR-Zeichenfolge, die die Zeichen für die Übereinstimmung enthält.

Rückgabetyp

Die BTRIM-Funktion gibt eine VARCHAR-Zeichenfolge zurück.

Beispiele

Im folgenden Beispiel werden Leerzeichen am Anfang und am Ende aus der Zeichenfolge entfernt `' abc '`:

```
select '     abc    ' as untrim, btrim('     abc    ') as trim;

untrim    | trim
----------+------
   abc    | abc
```

Im folgenden Beispiel werden die Zeichenfolgen `'xyz'` am Anfang und am Ende aus der Zeichenfolge `'xyzaxyzbxyzcxyz'` entfernt. Die Zeichenfolgen `'xyz'` am Anfang und am Ende werden entfernt, entsprechende Zeichenfolgen innerhalb dieser Zeichenfolge jedoch nicht.

```
select 'xyzaxyzbxyzcxyz' as untrim,
```

```
btrim('xyzaxyzbxyzcxyz', 'xyz') as trim;

     untrim      |   trim
-----------------+-----------
 xyzaxyzbxyzcxyz | axyzbxyzc
```

Im folgenden Beispiel werden die Teile am Anfang und am Ende der Zeichenfolge `'setuphistorycassettes'` entfernt, die mit einem der Zeichen in der trim_chars-Liste `'tes'` übereinstimmen. Alle `t`, `e` oder `s` am Anfang oder Ende der Eingabezeichenfolge, die vor einem anderen Zeichen stehen, das nicht in der trim_chars-Liste enthalten ist, werden entfernt.

```
SELECT btrim('setuphistorycassettes', 'tes');

     btrim
----------------
 uphistoryca
```

Die Funktion BTTEXT_PATTERN_CMP

Synonym mit der Funktion BPCHARCMP.

Details dazu finden Sie unter Die Funktion BPCHARCMP.

Die Funktion CHAR_LENGTH

Synonym mit der Funktion LEN.

Siehe Die Funktion LEN.

Die Funktion CHARACTER_LENGTH

Synonym mit der Funktion LEN.

Siehe Die Funktion LEN.

Funktion CHARINDEX

Gibt den Ort der angegebenen Unterzeichenfolge innerhalb einer Zeichenfolge zurück.

Ähnliche Funktionen finden Sie unter Die Funktion POSITION und Die Funktion STRPOS.

Syntax

```
CHARINDEX( substring, string )
```

Argumente

substring

Die Unterzeichenfolge, die innerhalb der Zeichenfolge gesucht werden soll.

string

Die Zeichenfolge oder Spalte, die durchsucht werden soll.

Rückgabetyp

INTEGER

Die CHARINDEX-Funktion gibt eine `INTEGER` zurück, die der Position der Teilzeichenfolge entspricht (eins-basiert, nicht null-basiert). Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. CHARINDEX gibt `0` zurück, wenn die Teilzeichenfolge nicht innerhalb der Zeichenfolge gefunden wird.

Beispiele

Verwenden Sie das folgende Beispiel, um die Position der Zeichenfolge `fish` innerhalb des Worts dog zu zeigen.

```
SELECT CHARINDEX('fish', 'dog');

+-----------+
| charindex |
+-----------+
|         0 |
+-----------+
```

Verwenden Sie das folgende Beispiel, um die Position der Zeichenfolge `fish` innerhalb des Worts `dogfish` zu zeigen.

```
SELECT CHARINDEX('fish', 'dogfish');
```

```
+-----------+
| charindex |
+-----------+
|         4 |
+-----------+
```

Im folgenden Beispiel wird die Tabelle SALES aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Zahl der bestimmten Verkaufstransaktionen mit einer Kommission von mehr als 999,00 aus der Tabelle SALES zurückzugeben. Dieser Befehl zählt Provisionen über 999,00, indem geprüft wird, ob die Dezimalzahl mehr als 4 Stellen vom Anfang des Provisionswerts entfernt ist.

```
SELECT DISTINCT CHARINDEX('.', commission), COUNT (CHARINDEX('.', commission))
FROM sales
WHERE CHARINDEX('.', commission) > 4
GROUP BY CHARINDEX('.', commission)
ORDER BY 1,2;

+-----------+-------+
| charindex | count |
+-----------+-------+
|         5 |   629 |
+-----------+-------+
```

Die Funktion CHR

Die CHR-Funktion gibt das Zeichen zurück, das mit dem ASCII-Codepunktzeichenwert übereinstimmt, der durch den Eingabeparameter angegeben wird.

Syntax

```
CHR(number)
```

Argument

number (Zahl

Der Eingabeparameter ist eine `INTEGER`, die einen ASCII-Codepunktwert darstellt.

Rückgabetyp

CHAR

Die CHR-Funktion gibt eine CHAR-Zeichenfolge zurück, wenn ein ASCII-Zeichen mit dem Eingabewert übereinstimmt. Wenn es für die Eingabezahl keine ASCII-Übereinstimmung gibt, gibt die Funktion NULL zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um das Zeichen zurückzugeben, das dem ASCII-Codepunkt 0 entspricht. Beachten Sie, dass die CHR-Funktion NULL für die Eingabe 0 zurückgibt.

```
SELECT CHR(0);

+-----+
| chr |
+-----+
|     |
+-----+
```

Verwenden Sie das folgende Beispiel, um das Zeichen zurückzugeben, das dem ASCII-Codepunkt 65 entspricht.

```
SELECT CHR(65);

+-----+
| chr |
+-----+
| A   |
+-----+
```

Verwenden Sie das folgende Beispiel, um Ereignisnamen zurückzugeben, die mit einem großen A beginnen (ASCII-Codepunkt 65). Verwenden Sie das folgende Beispiel, um die Tabelle EVENT aus der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

```
SELECT DISTINCT eventname FROM event
WHERE SUBSTRING(eventname, 1, 1)=CHR(65) LIMIT 5;
```

```
+----------------------+
|       eventname      |
+----------------------+
| A Catered Affair     |
| As You Like It       |
| A Man For All Seasons |
| Alan Jackson         |
| Armando Manzanero    |
+----------------------+
```

Funktion COLLATE

Die Funktion COLLATE überschreibt die Sortierung einer Zeichenfolgenspalte oder eines Ausdrucks.

Weitere Informationen zur Erstellung von Tabellen mit der Datenbanksortierung finden Sie unter CREATE TABLE.

Weitere Informationen zur Erstellung von Datenbanken mit der Datenbanksortierung finden Sie unter CREATE DATABASE.

Syntax

```
COLLATE( string, 'case_sensitive' | 'case_insensitive');
```

Argumente

string

Eine Zeichenfolgenspalte oder ein Ausdruck, die/den Sie überschreiben möchten.

‚case_sensitive' | ‚case_insensitive'

Die Zeichenfolgenkonstante eines Sortierungsnamens. Amazon Redshift unterstützt case_sensitive oder case_insensitive.

Rückgabetyp

Die COLLATE-Funktion gibt abhängig vom ersten Eingabeausdruckstyp `VARCHAR`oder `CHAR` zurück. Diese Funktion ändert nur die Sortierung des ersten Eingabearguments, nicht jedoch seinen Ausgabewert.

Beispiele

Verwenden Sie das folgende Beispiel, um Tabelle T zu erstellen und col1 in Tabelle T als `case_sensitive` zu definieren.

```
CREATE TABLE T ( col1 Varchar(20) COLLATE case_sensitive );

INSERT INTO T VALUES ('john'),('JOHN');
```

Wenn Sie die erste Abfrage ausführen, gibt Amazon Redshift nur `john` zurück. Nachdem COLLATE-Funktion auf col1 ausgeführt wurde, ändert sich die Sortierung zu `case_insensitive`. Die zweite Abfrage gibt `john` und `JOHN` zurück.

```
SELECT * FROM T WHERE col1 = 'john';

+------+
| col1 |
+------+
| john |
+------+

SELECT * FROM T WHERE COLLATE(col1, 'case_insensitive') = 'john';

+------+
| col1 |
+------+
| john |
| JOHN |
+------+
```

Verwenden Sie das folgende Beispiel, um Tabelle A zu erstellen und col1 in Tabelle A als `case_insensitive` zu definieren.

```
CREATE TABLE A ( col1 Varchar(20) COLLATE case_insensitive );

INSERT INTO A VALUES ('john'),('JOHN');
```

Wenn Sie die erste Abfrage ausführen, gibt Amazon Redshift `john` und `JOHN` zurück. Nachdem COLLATE-Funktion auf col1 ausgeführt wurde, ändert sich die Sortierung zu `case_sensitive`. Die zweite Abfrage gibt nur `john` zurück.

```
SELECT * FROM A WHERE col1 = 'john';

+------+
| col1 |
+------+
| john |
| JOHN |
+------+

SELECT * FROM A WHERE COLLATE(col1, 'case_sensitive') = 'john';

+------+
| col1 |
+------+
| john |
+------+
```

Funktion CONCAT

Die CONCAT-Funktion verkettet zwei Ausdrücke und gibt den Ergebnisausdruck zurück. Um mehr als zwei Ausdrücke zu verketten, verwenden Sie verschachtelte CONCAT-Funktionen. Der Verkettungsoperator (| |) zwischen zwei Ausdrücken generiert dieselben Ergebnisse wie die CONCAT-Funktion.

Syntax

```
CONCAT ( expression1, expression2 )
```

Argumente

expression1, expression2

Beide Argumente können eine Zeichenfolge mit fester Länge, eine Zeichenfolge variabler Länge, ein binärer Ausdruck oder ein Ausdruck sein, der für eine dieser Eingaben ausgewertet wird.

Rückgabetyp

CONCAT gibt einen Ausdruck zurück. Der Datentyp des Ausdrucks ist derselbe Typ wie die Eingabeargumente.

Wenn es sich bei den Eingabeausdrücken um verschiedene Typen handelt, versucht Amazon Redshift implizit einen der beiden Ausdrücke umzuwandeln. Wenn Werte nicht umgewandelt werden können, wird ein Fehler zurückgegeben.

Nutzungshinweise

- Für die Funktion CONCAT und den Verkettungsoperator gilt, dass das Ergebnis der Verkettung null ist, wenn einer oder beide Ausdrücke null sind.

Beispiele

Im folgenden Beispiel werden zwei Zeichenliterale verkettet:

```
SELECT CONCAT('December 25, ', '2008');

concat
-------------------
December 25, 2008
(1 row)
```

Die folgende Abfrage verwendet anstelle von | | den Operator CONCAT und generiert dasselbe Ergebnis:

```
SELECT 'December 25, '||'2008';

?column?
-------------------
December 25, 2008
(1 row)
```

Im folgenden Beispiel wird eine verschachtelte CONCAT-Funktion innerhalb einer anderen CONCAT-Funktion benutzt, um drei Zeichenfolgen zu verketten:

```
SELECT CONCAT('Thursday, ', CONCAT('December 25, ', '2008'));

concat
----------------------------
Thursday, December 25, 2008
(1 row)
```

Um Spalten zu verketten, die NULL-Werte enthalten könnten, benutzen Sie die NVL- und COALESCE-Funktionen, die einen bestimmten Wert zurückgibt, wenn sie auf NULL trifft. Im folgenden Beispiel wird NVL verwendet, um eine 0 zurückzugeben, wenn NULL gefunden wird.

```
SELECT CONCAT(venuename, CONCAT(' seats ', NVL(venueseats, 0))) AS seating
FROM venue WHERE venuestate = 'NV' OR venuestate = 'NC'
ORDER BY 1
LIMIT 5;

seating
---------------------------------
Ballys Hotel seats 0
Bank of America Stadium seats 73298
Bellagio Hotel seats 0
Caesars Palace seats 0
Harrahs Hotel seats 0
(5 rows)
```

Die folgende Abfrage verkettet CITY- und STATE-Werte aus der Tabelle VENUE:

```
SELECT CONCAT(venuecity, venuestate)
FROM venue
WHERE venueseats > 75000
ORDER BY venueseats;

concat
-------------------
DenverCO
Kansas CityMO
East RutherfordNJ
LandoverMD
(4 rows)
```

Die folgende Abfrage verwendet verschachtelte CONCAT-Funktionen. Die Abfrage verkettet CITY- und STATE-Werte aus der Tabelle, trennt die Ergebniszeichenfolge jedoch durch ein Komma und ein Leerzeichen:

```
SELECT CONCAT(CONCAT(venuecity,', '),venuestate)
FROM venue
WHERE venueseats > 75000
ORDER BY venueseats;
```

```
concat
--------------------
Denver, CO
Kansas City, MO
East Rutherford, NJ
Landover, MD
(4 rows)
```

Im folgenden Beispiel werden zwei Binärausdrücke verkettet. Hierbei ist `abc` ein Binärwert (mit einer hexadezimalen Darstellung von `616263`) und `def` ein Binärwert (mit einer hexadezimalen Darstellung von `646566`). Das Ergebnis wird automatisch als hexadezimale Darstellung des Binärwerts angezeigt.

```
SELECT CONCAT('abc'::VARBYTE, 'def'::VARBYTE);

concat
------------------
616263646566
```

Die Funktion CRC32

CRC32 ist eine Funktion zur Fehlererkennung. Die Funktion verwendet einen CRC32-Algorithmus, um Änderungen zwischen Quell- und Zieldaten zu entdecken. Die CRC32-Funktion konvertiert eine Zeichenfolge mit variabler Länge in eine Zeichenfolge mit 8 Zeichen, die eine Textdarstellung des hexadezimalen Werts einer 32-Bit-Binärfolge ist. Wenn Sie Änderungen zwischen Quell- und Zieldaten erkennen möchten, verwenden Sie die CRC32-Funktion für die Quelldaten und speichern Sie die Ausgabe. Verwenden Sie dann die CRC32-Funktion für die Zieldaten und vergleichen Sie diese Ausgabe mit der Ausgabe der Quelldaten. Die Ausgaben sind identisch, wenn die Daten nicht geändert wurden, und sie sind unterschiedlich, wenn die Daten geändert wurden.

Syntax

```
CRC32(string)
```

Argumente

string

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein CHAR- oder VARCHAR-Typ ausgewertet wird.

Rückgabetyp

Die CRC32-Funktion gibt eine Zeichenfolge mit 8 Zeichen zurück, die eine Textdarstellung des hexadezimalen Werts einer 32-Bit-Binärfolge ist. Die CRC32-Funktion von Amazon Redshift basiert auf dem CRC-32C-Polynom.

Beispiele

Um den 8-Bit-Wert für die Zeichenfolge `Amazon Redshift` anzuzeigen.

```
SELECT CRC32('Amazon Redshift');

+----------+
|  crc32   |
+----------+
| f2726906 |
+----------+
```

Funktion DIFFERENCE

Die Funktion DIFFERENCE vergleicht die amerikanischen Soundex-Codes zweier Zeichenfolgen. Die Funktion gibt `INTEGER` zurück, um die Anzahl der übereinstimmenden Zeichen zwischen den Soundex-Codes anzugeben.

Ein Soundex-Code ist eine Zeichenfolge, die vier Zeichen lang ist. Ein Soundex-Code stellt eher den Klang eines Wortes und weniger seine Schreibweise dar. Zum Beispiel haben `Smith` und `Smyth` den gleichen Soundex-Code.

Syntax

```
DIFFERENCE(string1, string2)
```

Argumente

string1

Eine `CHAR`-Zeichenfolge, eine `VARCHAR`-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein `CHAR`- oder `VARCHAR`-Typ ausgewertet wird.

string2

Eine `CHAR`-Zeichenfolge, eine `VARCHAR`-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein `CHAR`- oder `VARCHAR`-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Die Funktion DIFFERENCE gibt einen `INTEGER`-Wert von 0–4 zurück, der die Anzahl der übereinstimmenden Zeichen in den amerikanische Soundex-Codes der beiden Zeichenfolgen zählt. Ein Soundex-Code hat 4 Zeichen, daher gibt die DIFFERENCE-Funktion `4` zurück, wenn alle 4 Zeichen der Werte des amerikanischen Soundex-Codes der Zeichenfolgen identisch sind. DIFFERENCE gibt `0` zurück, wenn eine der beiden Zeichenfolgen leer ist. Die Funktion gibt 1 zurück, wenn keine Zeichenfolge gültige Zeichen enthält. Die Funktion DIFFERENCE konvertiert nur englische alphabetische ASCII-Zeichen in Klein- oder Großbuchstaben, einschließlich a–z und A–Z. DIFFERENCE ignoriert andere Zeichen.

Beispiele

Verwenden Sie das folgende Beispiel, um die Soundex-Werte der Zeichenfolgen `%` und `@` zu vergleichen. Die Funktion gibt 1 zurück, da keine Zeichenfolge gültige Zeichen enthält.

```
SELECT DIFFERENCE('%', '@');

+------------+
| difference |
+------------+
|          1 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Soundex-Werte von `Amazon` und einer leeren Zeichenfolge zu vergleichen. Die Funktion gibt `0` zurück, da eine der beiden Zeichenfolgen leer ist.

```
SELECT DIFFERENCE('Amazon', '');

+------------+
| difference |
+------------+
|          0 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Soundex-Werte der Zeichenfolgen `Amazon` und `Ama` zu vergleichen. Die Funktion gibt 2 zurück, weil 2 Zeichen der Soundex-Werte der Zeichenfolgen identisch sind.

```
SELECT DIFFERENCE('Amazon', 'Ama');

+------------+
| difference |
+------------+
|          2 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Soundex-Werte der Zeichenfolgen `Amazon` und `+-*/%Amazon` zu vergleichen. Die Funktion gibt `4` zurück, weil 4 Zeichen der Soundex-Werte der Zeichenfolgen identisch sind. Beachten Sie, dass die Funktion die ungültigen Zeichen `+-*/%` in der zweiten Zeichenfolge ignoriert.

```
SELECT DIFFERENCE('Amazon', '+-*/%Amazon');

+------------+
| difference |
+------------+
|          4 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Soundex-Werte der Zeichenfolgen `AC/DC` und `Ay See Dee See` zu vergleichen. Die Funktion gibt `4` zurück, weil 4 Zeichen der Soundex-Werte der Zeichenfolgen identisch sind.

```
SELECT DIFFERENCE('AC/DC', 'Ay See Dee See');

+------------+
| difference |
+------------+
|          4 |
+------------+
```

Die Funktion INITCAP

Wandelt den ersten Buchstaben jedes Worts in einer angegebenen Zeichenfolge in einen Großbuchstaben um. INITCAP unterstützt UTF-8-Multibyte-Zeichen bis zu einer maximalen Länge von vier Bytes pro Zeichen.

Syntax

```
INITCAP(string)
```

Argument

string

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein CHAR- oder VARCHAR-Typ ausgewertet wird.

Rückgabetyp

VARCHAR

Nutzungshinweise

Die INITCAP-Funktion wandelt den ersten Buchstaben jedes Worts in einer Zeichenfolge in einen Großbuchstaben um. Alle folgenden Buchstaben werden in Kleinbuchstaben umgewandelt (oder bleiben Kleinbuchstaben). Daher ist es wichtig, zu verstehen, welche Zeichen (außer Leerzeichen) als Worttrennzeichen fungieren. Ein Worttrennzeichen ist ein nicht alphanumerisches Zeichen, einschließlich Interpunktionszeichen, Symbolen und Steuerzeichen. Alle folgenden Zeichen sind Worttrennzeichen:

```
! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [ \ ] ^ _ ` { | } ~
```

Tabulatorzeichen, Zeichen für neue Zeilen, Seitenvorschübe, Zeilenvorschübe und Zeilenumbrüche sind ebenfalls Worttrennzeichen.

Beispiele

In den folgenden Beispielen werden Daten aus der Tabellen CATEGORY und USERS in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Anfangsbuchstaben jedes Worts in der Spalte CATDESC in Großbuchstaben umzuwandeln.

```
SELECT catid, catdesc, INITCAP(catdesc)
FROM category
ORDER BY 1, 2, 3;
```

```
+-------+-------------------------------------------
+-----------------------------------------+
| catid |                  catdesc                  |                  initcap
          |
+-------+-------------------------------------------
+-----------------------------------------+
|     1 | Major League Baseball                     | Major League Baseball
          |
|     2 | National Hockey League                    | National Hockey League
          |
|     3 | National Football League                  | National Football League
          |
|     4 | National Basketball Association           | National Basketball Association
          |
|     5 | Major League Soccer                       | Major League Soccer
          |
|     6 | Musical theatre                           | Musical Theatre
          |
|     7 | All non-musical theatre                   | All Non-Musical Theatre
          |
|     8 | All opera and light opera                 | All Opera And Light Opera
          |
|     9 | All rock and pop music concerts           | All Rock And Pop Music Concerts
          |
|    10 | All jazz singers and bands                | All Jazz Singers And Bands
          |
|    11 | All symphony, concerto, and choir concerts | All Symphony, Concerto, And
 Choir Concerts |
+-------+-------------------------------------------
+-----------------------------------------+
```

Verwenden Sie das folgende Beispiel, um zu zeigen, dass die INITCAP-Funktion Großbuchstaben nicht beibehält, wenn sie sich nicht am Anfang von Wörtern befinden. Zum Beispiel wird Zeichenfolge `MLB` zu `Mlb`.

```
SELECT INITCAP(catname)
FROM category
ORDER BY catname;

+-----------+
|  initcap  |
+-----------+
```

```
| Classical |
| Jazz      |
| Mlb       |
| Mls       |
| Musicals  |
| Nba       |
| Nfl       |
| Nhl       |
| Opera     |
| Plays     |
| Pop       |
+-----------+
```

Verwenden Sie das folgende Beispiel, um zu zeigen, dass nicht alphanumerische Zeichen außer Leerzeichen als Worttrennzeichen dienen. Mehrere Buchstaben in jeder Zeichenfolge werden großgeschrieben.

```
SELECT email, INITCAP(email)
FROM users
ORDER BY userid DESC LIMIT 5;

+------------------------------------+-----------------------------------+
|               email                |              initcap              |
+------------------------------------+-----------------------------------+
| urna.Ut@egetdictumplacerat.edu     | Urna.Ut@Egetdictumplacerat.Edu    |
| nibh.enim@egestas.ca               | Nibh.Enim@Egestas.Ca              |
| in@Donecat.ca                      | In@Donecat.Ca                     |
| sodales@blanditviverraDonec.ca     | Sodales@Blanditviverradonec.Ca    |
| sociis.natoque.penatibus@vitae.org | Sociis.Natoque.Penatibus@Vitae.Org |
+------------------------------------+-----------------------------------+
```

Die Funktionen LEFT und RIGHT

Diese Funktionen geben die angegebene Zahl der Zeichen am weitesten links oder am weitesten rechts in einer Zeichenfolge zurück.

Die Zahl basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt.

Syntax

```
LEFT( string,  integer )
```

```
RIGHT( string,  integer )
```

Argumente

string

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge oder ein beliebiger Ausdruck, die bzw. der implizit als eine CHAR- oder VARCHAR-Zeichenfolge ausgewertet wird.

integer

Eine positive Ganzzahl.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle EVENT in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die 5 Zeichen in Veranstaltungsnamen mit IDs zwischen 1000 und 1005 zurückzugeben, die sich am weitesten links oder am weitesten rechts befinden.

```
SELECT eventid, eventname,
LEFT(eventname,5) AS left_5,
RIGHT(eventname,5) AS right_5
FROM event
WHERE eventid BETWEEN 1000 AND 1005
ORDER BY 1;

+---------+----------------+--------+---------+
| eventid |   eventname    | left_5 | right_5 |
+---------+----------------+--------+---------+
|    1000 | Gypsy          | Gypsy  | Gypsy   |
|    1001 | Chicago        | Chica  | icago   |
|    1002 | The King and I | The K  | and I   |
|    1003 | Pal Joey       | Pal J  | Joey    |
|    1004 | Grease         | Greas  | rease   |
|    1005 | Chicago        | Chica  | icago   |
+---------+----------------+--------+---------+
```

Die Funktion LEN

Gibt die Länge der angegebenen Zeichenfolge durch die Anzahl der Zeichen an.

Syntax

LEN ist synonym mit Die Funktion LENGTH, Die Funktion CHAR_LENGTH, Die Funktion CHARACTER_LENGTH und Die Funktion TEXTLEN.

```
LEN(expression)
```

Argument

expression

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge, ein VARBYTE-Ausdruck oder ein Ausdruck, die bzw. der implizit als ein CHAR-, VARCHAR- oder VARBYTE-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Die LEN-Funktion gibt eine Ganzzahl zurück, die die Anzahl der Zeichen in der Eingabezeichenfolge anzeigt.

Wenn es sich um eine Folge von Zeichen handelt, gibt die LEN-Funktion die tatsächliche Anzahl der Zeichen in Multibyte-Zeichenfolgen zurück, nicht die Anzahl der Bytes. Beispielsweise ist eine VARCHAR(12)-Spalte erforderlich, um drei chinesische Zeichen mit vier Bytes zu speichern. Die LEN-Funktion gibt für diese Zeichenfolge 3 zurück. Verwenden Sie die Funktion OCTET_LENGTH, um die Länge einer Zeichenfolge in Bytes abzurufen.

Nutzungshinweise

Wenn Ausdruck eine CHAR-Zeichenfolge ist, werden nachfolgende Leerzeichen nicht gezählt.

Wenn Ausdruck eine VARCHAR-Zeichenfolge ist, werden nachfolgende Leerzeichen gezählt.

Beispiele

Verwenden Sie das folgende Beispiel, um die Anzahl der Bytes und die Anzahl der Zeichen der Zeichenfolge `français` zurückzugeben.

```
SELECT OCTET_LENGTH('français'),
LEN('français');

+--------------+-----+
| octet_length | len |
+--------------+-----+
|            9 |   8 |
+--------------+-----+
```

Verwenden Sie das folgende Beispiel, um die Anzahl der Bytes und die Anzahl der Zeichen der Zeichenfolge `français` zurückzugeben, ohne die Funktion OCTET_LENGTH zu nutzen. Weitere Informationen hierzu finden Sie unter CAST-Funktion.

```
SELECT LEN(CAST('français' AS VARBYTE)) as bytes, LEN('français');

+-------+-----+
| bytes | len |
+-------+-----+
|     9 |   8 |
+-------+-----+
```

Verwenden Sie das folgende Beispiel, um die Anzahl der Zeichen in folgenden Zeichenfolgen zurückzugeben: `cat` ohne abschließende Leerzeichen,`cat` mit drei abschließenden Leerzeichen, `cat` mit drei abschließenden Leerzeichen, umgewandelt in ein CHAR der Länge 6 und `cat` mit drei abschließenden Leerzeichen, umgewandelt in ein VARCHAR der Länge 6. Beachten Sie, dass die Funktion zwar nachfolgende Leerzeichen für CHAR-Zeichenfolgen nicht zählt, nachgestellte Leerzeichen für VARCHAR-Zeichenfolgen werden aber gezählt.

```
SELECT LEN('cat'), LEN('cat   '), LEN(CAST('cat   ' AS CHAR(6))) AS len_char,
 LEN(CAST('cat   ' AS VARCHAR(6))) AS len_varchar;

+-----+-----+----------+-------------+
| len | len | len_char | len_varchar |
+-----+-----+----------+-------------+
|   3 |   6 |        3 |           6 |
+-----+-----+----------+-------------+
```

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle VENUE in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die 10 längsten Ortsnamen in der Tabelle VENUE zurückzugeben.

```
SELECT venuename, LEN(venuename)
FROM venue
ORDER BY 2 DESC, 1
LIMIT 10;

+----------------------------------------+-----+
|                venuename               | len |
+----------------------------------------+-----+
| Saratoga Springs Performing Arts Center |  39 |
| Lincoln Center for the Performing Arts  |  38 |
| Nassau Veterans Memorial Coliseum       |  33 |
| Jacksonville Municipal Stadium          |  30 |
| Rangers BallPark in Arlington           |  29 |
| University of Phoenix Stadium           |  29 |
| Circle in the Square Theatre            |  28 |
| Hubert H. Humphrey Metrodome            |  28 |
| Oriole Park at Camden Yards             |  27 |
| Dick's Sporting Goods Park              |  26 |
+----------------------------------------+-----+
```

Die Funktion LENGTH

Synonym mit der Funktion LEN.

Siehe Die Funktion LEN.

Die Funktion LOWER

Konvertiert eine Zeichenfolge in Kleinbuchstaben. LOWER unterstützt UTF-8-Multibyte-Zeichen bis zu einer maximalen Länge von vier Bytes pro Zeichen.

Syntax

```
LOWER(string)
```

Argument

string

Eine VARCHAR-Zeichenfolge oder ein beliebiger Ausdruck, die/der zum Typ VARCHAR ausgewertet wird.

Rückgabetyp

Zeichenfolge

Die LOWER-Funktion gibt eine Zeichenfolge zurück, die den gleichen Datentyp wie die Zeichenfolge hat. Wenn es sich bei der Eingabe beispielsweise um eine CHAR-Zeichenfolge handelt, gibt die Funktion eine CHAR-Zeichenfolge zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle CATEGORY in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Zeichenfolge VARCHAR im Feld CATGROUP in Kleinbuchstaben zu konvertieren.

```
SELECT catname, LOWER(catname) FROM category ORDER BY 1,2;

+-----------+-----------+
|  catname  |   lower   |
+-----------+-----------+
| Classical | classical |
| Jazz      | jazz      |
| MLB       | mlb       |
| MLS       | mls       |
| Musicals  | musicals  |
| NBA       | nba       |
| NFL       | nfl       |
| NHL       | nhl       |
| Opera     | opera     |
| Plays     | plays     |
| Pop       | pop       |
+-----------+-----------+
```

Die Funktionen LPAD und RPAD

Diese Funktionen fügen vor oder nach einer Zeichenfolge Zeichen an, basierend auf einer angegebenen Länge.

Syntax

```
LPAD(string1, length, [ string2 ])
```

```
RPAD(string1, length, [ string2 ])
```

Argumente

string1

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein CHAR- oder VARCHAR-Typ ausgewertet wird.

length

Eine Ganzzahl, die die Länge des Ergebnisses der Funktion definiert. Die Länge einer Zeichenfolge basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Wenn string1 länger als die angegebene Länge ist, wird sie abgeschnitten (rechts). Wenn length null oder eine negative Zahl ist, ist das Ergebnis der Funktion eine leere Zeichenfolge.

string2

(Optional) Ein oder mehrere Zeichen, die vor oder nach string1 angefügt werden. Wenn dieses Argument nicht angegeben wird, werden Leerzeichen verwendet.

Rückgabetyp

VARCHAR

Beispiele

In den folgenden Beispielen werden Daten aus der EVENT-Tabelle in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um einen angegebenen Satz von Veranstaltungsnamen auf 20 Zeichen zu kürzen und vor den kürzeren Namen Leerzeichen einzufügen.

```
SELECT LPAD(eventname, 20) FROM event
WHERE eventid BETWEEN 1 AND 5 ORDER BY 1;

+----------------------+
|         lpad         |
+----------------------+
|              Salome |
|        Il Trovatore |
|       Boris Godunov |
|     Gotterdammerung |
|La Cenerentola (Cind |
+----------------------+
```

Verwenden Sie das folgende Beispiel, um denselben Satz von Veranstaltungsnamen auf 20 Zeichen zu kürzen und vor den kürzeren Namen jedoch `0123456789` anzufügen.

```
SELECT RPAD(eventname, 20,'0123456789') FROM event
WHERE eventid BETWEEN 1 AND 5 ORDER BY 1;

+----------------------+
|         rpad         |
+----------------------+
| Boris Godunov0123456 |
| Gotterdammerung01234 |
| Il Trovatore01234567 |
| La Cenerentola (Cind |
| Salome01234567890123 |
+----------------------+
```

Die Funktion LTRIM

Kürzt Zeichen ab dem Anfang einer Zeichenfolge. Entfernt die längste Zeichenfolge, die nur Zeichen aus der Liste der Trimm-Zeichen enthält. Die Kürzung ist abgeschlossen, wenn in der Eingabezeichenfolge kein Trimm-Zeichen enthalten ist.

Syntax

```
LTRIM( string [, trim_chars] )
```

Argumente

string

Eine Zeichenfolgenspalte, ein Ausdruck oder ein Zeichenfolgenliteral, die/der/das gekürzt werden soll.

trim_chars

Eine Zeichenfolgenspalte, ein Ausdruck oder ein Zeichenfolgenliteral, die/der/das die Zeichen darstellt, die ab dem Anfang von string gekürzt werden sollen. Wenn nicht angegeben, wird ein Leerzeichen als Trimm-Zeichen verwendet.

Rückgabetyp

Die LTRIM-Funktion gibt eine Zeichenfolge zurück, die denselben Datentyp wie die Eingabezeichenfolge (string) hat (CHAR oder VARCHAR).

Beispiele

Im folgenden Beispiel wird das Jahr aus der `listime`-Spalte gekürzt. Die Trimm-Zeichen im Zeichenfolgenliteral `'2008-'` geben die Zeichen an, die von links gekürzt werden sollen. Bei Verwendung der Trimm-Zeichen `'028-'` erzielen Sie dasselbe Ergebnis.

```
select listid, listtime, ltrim(listtime, '2008-')
from listing
order by 1, 2, 3
limit 10;

listid |      listtime       |     ltrim
-------+---------------------+---------------
     1 | 2008-01-24 06:43:29 | 1-24 06:43:29
     2 | 2008-03-05 12:25:29 | 3-05 12:25:29
     3 | 2008-11-01 07:35:33 | 11-01 07:35:33
     4 | 2008-05-24 01:18:37 | 5-24 01:18:37
     5 | 2008-05-17 02:29:11 | 5-17 02:29:11
     6 | 2008-08-15 02:08:13 | 15 02:08:13
     7 | 2008-11-15 09:38:15 | 11-15 09:38:15
     8 | 2008-11-09 05:07:30 | 11-09 05:07:30
     9 | 2008-09-09 08:03:36 | 9-09 08:03:36
    10 | 2008-06-17 09:44:54 | 6-17 09:44:54
```

LTRIM entfernt alle Zeichen in trim_chars, wenn sie sich am Anfang von string befinden. Im folgenden Beispiel werden die Zeichen „C“, „D“ und „G“gekürzt, wenn sie sich am Anfang von VENUENAME befinden. Dabei handelt es sich um eine VARCHAR-Spalte.

```
select venueid, venuename, ltrim(venuename, 'CDG')
from venue
where venuename like '%Park'
order by 2
limit 7;

venueid | venuename                  | btrim
--------+----------------------------+--------------------------
    121 | ATT Park                   | ATT Park
    109 | Citizens Bank Park         | itizens Bank Park
    102 | Comerica Park              | omerica Park
      9 | Dick's Sporting Goods Park | ick's Sporting Goods Park
     97 | Fenway Park                | Fenway Park
    112 | Great American Ball Park   | reat American Ball Park
    114 | Miller Park                | Miller Park
```

Im folgenden Beispiel wird das Trimm-Zeichen 2 verwendet, das aus der`venueid`-Spalte abgerufen wird.

```
select ltrim('2008-01-24 06:43:29', venueid)
from venue where venueid=2;

ltrim
------------------
008-01-24 06:43:29
```

Im folgenden Beispiel werden keine Zeichen gekürzt, da vor dem Trimm-Zeichen `'0'` eine 2 enthalten ist.

```
select ltrim('2008-01-24 06:43:29', '0');

ltrim
-------------------
2008-01-24 06:43:29
```

Im folgenden Beispiel werden standardmäßige Leerzeichen als Trimm-Zeichen verwendet und die beiden Leerzeichen zu Beginn der Zeichenfolge werden gekürzt.

```
select ltrim('  2008-01-24 06:43:29');

ltrim
-------------------
2008-01-24 06:43:29
```

Funktion OCTENDEX

Die Funktion OCTEINDEX gibt den Ort einer Teilzeichenfolge innerhalb eines Strings als Anzahl der Bytes zurück.

Syntax

```
OCTETINDEX(substring, string)
```

Argumente

substring

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein CHAR- oder VARCHAR-Typ ausgewertet wird.

string

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge oder ein Ausdruck, die bzw. der implizit als ein CHAR- oder VARCHAR-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Die Funktion OCTETINDEX gibt einen INTEGER-Wert zurück, der der Position der Teilzeichenfolge innerhalb der Zeichenfolge als Anzahl von Bytes entspricht, wobei das erste Zeichen in der Zeichenfolge als 1 gezählt wird. Wenn die Zeichenfolge keine Multibyte-Zeichen enthält, entspricht das Ergebnis dem Ergebnis der Funktion CHARINDEX. Wenn die Zeichenfolge die Teilzeichenfolge nicht enthält, gibt die Funktion 0 zurück. Wenn die Teilzeichenfolge leer ist, gibt die Funktion 1 zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um die Position der Teilzeichenfolge q innerhalb der Zeichenfolge `Amazon Redshift` zurückzugeben. Dieses Beispiel gibt `0` zurück, da die Teilzeichenfolge nicht in der Zeichenfolge enthalten ist.

```
SELECT OCTETINDEX('q', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          0 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Position einer leeren Zeichenfolge innerhalb der Zeichenfolge `Amazon Redshift` zurückzugeben. Dieses Beispiel gibt 1 zurück, da die Teilzeichenfolge leer ist.

```
SELECT OCTETINDEX('', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          1 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Position der Teilzeichenfolge `Redshift` innerhalb der Zeichenfolge `Amazon Redshift` zurückzugeben. Dieses Beispiel gibt 8 zurück, da die Teilzeichenfolge mit dem achten Byte der Zeichenfolge beginnt.

```
SELECT OCTETINDEX('Redshift', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          8 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Position der Teilzeichenfolge `Redshift` innerhalb der Zeichenfolge `Amazon Redshift` zurückzugeben. Im folgenden Beispiel wird 21 zurückgegeben, da die ersten sechs Zeichen der Zeichenfolge Doppelbyte-Zeichen sind.

```
SELECT OCTETINDEX('Redshift', 'Άμαζον Amazon Redshift');

+------------+
| octetindex |
+------------+
|         21 |
+------------+
```

Die OCTET_LENGTH-Funktion

Gibt die Länge der angegebenen Zeichenfolge durch die Anzahl der Bytes an.

Syntax

```
OCTET_LENGTH(expression)
```

Argument

expression

Eine CHAR-Zeichenfolge, eine VARCHAR-Zeichenfolge, ein VARBYTE-Ausdruck oder ein Ausdruck, die bzw. der implizit als ein CHAR-, VARCHAR- oder VARBYTE-Typ ausgewertet wird.

Rückgabetyp

INTEGER

Die Funktion „OCTET_LENGTH“ gibt eine Ganzzahl zurück, die die Anzahl der Bytes in der Eingabezeichenfolge anzeigt.

Wenn es sich um eine Folge von Zeichen handelt, gibt die LEN-Funktion die tatsächliche Anzahl der Zeichen in Multibyte-Zeichenfolgen zurück, nicht die Anzahl der Bytes. Beispielsweise ist eine VARCHAR(12)-Spalte erforderlich, um drei chinesische Zeichen mit vier Bytes zu speichern. Die Funktion OCTET_LENGTH gibt für diese Zeichenfolge 12 zurück und die LEN-Funktion gibt für dieselbe Zeichenfolge 3 zurück.

Nutzungshinweise

Wenn Ausdruck eine CHAR-Zeichenfolge ist, gibt die Funktion die Länge der CHAR-Zeichenfolge zurück. Zum Beispiel ist die Ausgabe einer CHAR(6)-Eingabe ein CHAR(6).

Wenn Ausdruck eine VARCHAR-Zeichenfolge ist, werden nachfolgende Leerzeichen gezählt.

Beispiele

Verwenden Sie das folgende Beispiel, um die Anzahl der Byte zurückzugeben, wenn die Zeichenfolge `francais` mit drei abschließenden Leerzeichen in ein CHAR und einen VARCHAR-Typ umgewandelt wird. Weitere Informationen hierzu finden Sie unter CAST-Funktion.

```
SELECT OCTET_LENGTH(CAST('francais   ' AS CHAR(15))) AS octet_length_char,
 OCTET_LENGTH(CAST('francais   ' AS VARCHAR(15))) AS octet_length_varchar;

+------------------+---------------------+
| octet_length_char | octet_length_varchar |
+------------------+---------------------+
|                15 |                  11 |
+------------------+---------------------+
```

Verwenden Sie das folgende Beispiel, um die Anzahl der Bytes und die Anzahl der Zeichen der Zeichenfolge `français` zurückzugeben.

```
SELECT OCTET_LENGTH('français'), LEN('français');

+--------------+-----+
| octet_length | len |
+--------------+-----+
|            9 |   8 |
+--------------+-----+
```

Verwenden Sie das folgende Beispiel, um die Anzahl der Byte zurückzugeben, wenn die Zeichenfolge `français` in ein VARBYTE umgewandelt wird.

```
SELECT OCTET_LENGTH(CAST('français' AS VARBYTE));

+--------------+
| octet_length |
+--------------+
|            9 |
+--------------+
```

Die Funktion POSITION

Gibt den Ort der angegebenen Unterzeichenfolge innerhalb einer Zeichenfolge zurück.

Ähnliche Funktionen finden Sie unter Funktion CHARINDEX und Die Funktion STRPOS.

Syntax

```
POSITION(substring IN string )
```

Argumente

substring

Die Unterzeichenfolge, die innerhalb der Zeichenfolge gesucht werden soll.

string

Die Zeichenfolge oder Spalte, die durchsucht werden soll.

Rückgabetyp

Die POSITION-Funktion gibt eine `INTEGER` zurück, die der Position der Teilzeichenfolge entspricht (eins-basiert, nicht null-basiert). Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. POSITION gibt `0` zurück, wenn die Teilzeichenfolge nicht innerhalb der Zeichenfolge gefunden wird.

Beispiele

Verwenden Sie das folgende Beispiel, um die Position der Zeichenfolge `fish` innerhalb des Worts dog zu zeigen.

```
SELECT POSITION('fish' IN 'dog');

+-----------+
|  position |
+-----------+
|         0 |
+-----------+
```

Verwenden Sie das folgende Beispiel, um die Position der Zeichenfolge `fish` innerhalb des Worts `dogfish` zu zeigen.

```
SELECT POSITION('fish' IN 'dogfish');

+-----------+
```

```
|  position |
+-----------+
|         4 |
+-----------+
```

Im folgenden Beispiel wird die Tabelle SALES aus der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Zahl der bestimmten Verkaufstransaktionen mit einer Kommission von mehr als 999,00 aus der Tabelle SALES zurückzugeben. Dieser Befehl zählt Provisionen über 999,00, indem geprüft wird, ob die Dezimalzahl mehr als 4 Stellen vom Anfang des Provisionswerts entfernt ist.

```
SELECT DISTINCT POSITION('.' IN commission), COUNT (POSITION('.' IN commission))
FROM sales
WHERE POSITION('.' IN commission) > 4
GROUP BY POSITION('.' IN commission)
ORDER BY 1,2;

+-----------+-------+
|  position | count |
+-----------+-------+
|         5 |   629 |
+-----------+-------+
```

Die Funktion QUOTE_IDENT

Die Funktion QUOTE_IDENT gibt die angegebene Zeichenfolge als Zeichenfolge mit einem doppelten Anführungszeichen am Anfang und einem am Ende zurück. Die Funktionsausgabe kann als Bezeichner in einer SQL-Anweisung verwendet werden. Diese Funktion verdoppelt eingebettete doppelte Anführungszeichen korrekt.

QUOTE_IDENT fügt doppelte Anführungszeichen nur dann hinzu, wenn dies zum Erstellen eines gültigen Bezeichners erforderlich ist. Dies ist bei Zeichenfolgen der Fall, die Nicht-Bezeichner-Zeichen enthalten oder bei denen Großbuchstaben wie Kleinbuchstaben behandelt werden. Wenn eine Zeichenfolge immer in einfachen Anführungszeichen zurückgegeben werden soll, verwenden Sie QUOTE_LITERAL.

Syntax

```
QUOTE_IDENT(string)
```

Argument

string

Eine CHAR- oder VARCHAR-Zeichenfolge.

Rückgabetyp

Die QUOTE_IDENT-Funktion gibt denselben Zeichenfolgetyp wie die Eingabezeichenfolge zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um die Zeichenfolge "CAT" mit doppelten Anführungszeichen zurückzugeben.

```
SELECT QUOTE_IDENT('"CAT"');

+-------------+
| quote_ident |
+-------------+
| """CAT"""   |
+-------------+
```

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle CATEGORY in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Spalte CATNAME in doppelten Anführungszeichen zurückzugeben.

```
SELECT catid, QUOTE_IDENT(catname)
FROM category
ORDER BY 1,2;

+-------+-------------+
| catid | quote_ident |
+-------+-------------+
|     1 | "MLB"       |
|     2 | "NHL"       |
|     3 | "NFL"       |
|     4 | "NBA"       |
|     5 | "MLS"       |
|     6 | "Musicals"  |
```

```
|     7 | "Plays"     |
|     8 | "Opera"     |
|     9 | "Pop"       |
|    10 | "Jazz"      |
|    11 | "Classical" |
+-------+-------------+
```

Die Funktion QUOTE_LITERAL

Die QUOTE_LITERAL-Funktion gibt die angegebene Zeichenfolge als eine Zeichenfolge in Anführungszeichen zurück, damit sie als Zeichenfolgeliteral in einer SQL-Anweisung verwendet werden kann. Wenn es sich beim Eingabeparameter um eine Zahl handelt, wird er von QUOTE_LITERAL als Zeichenfolge behandelt. Eingebettete einfache Anführungszeichen und Backslashes werden korrekt verdoppelt.

Syntax

```
QUOTE_LITERAL(string)
```

Argument

string

Eine CHAR- oder VARCHAR-Zeichenfolge.

Rückgabetyp

Die QUOTE_LITERAL-Funktion gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück, die den gleichen Datentyp wie die Eingabezeichenfolge hat.

Beispiele

Verwenden Sie das folgende Beispiel, um die Zeichenfolge `''CAT''` mit EINZELNEN Anführungszeichen zurückzugeben.

```
SELECT QUOTE_LITERAL('''CAT''');

+---------------+
| quote_literal |
+---------------+
| '''CAT'''     |
```

```
+---------------+
```

In den folgenden Beispielen werden die Daten aus der CATEGORY-Tabelle in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Spalte CATNAME in doppelten Anführungszeichen zurückzugeben.

```
SELECT catid, QUOTE_LITERAL(catname)
FROM category
ORDER BY 1,2;

+-------+---------------+
| catid | quote_literal |
+-------+---------------+
|     1 | 'MLB'         |
|     2 | 'NHL'         |
|     3 | 'NFL'         |
|     4 | 'NBA'         |
|     5 | 'MLS'         |
|     6 | 'Musicals'    |
|     7 | 'Plays'       |
|     8 | 'Opera'       |
|     9 | 'Pop'         |
|    10 | 'Jazz'        |
|    11 | 'Classical'   |
+-------+---------------+
```

Verwenden Sie das folgende Beispiel, um die Spalte CATID in doppelten Anführungszeichen zurückzugeben.

```
SELECT QUOTE_LITERAL(catid), catname
FROM category
ORDER BY 1,2;

+---------------+-----------+
| quote_literal |  catname  |
+---------------+-----------+
| '1'           | MLB       |
| '10'          | Jazz      |
| '11'          | Classical |
| '2'           | NHL       |
```

```
| '3'           | NFL       |
| '4'           | NBA       |
| '5'           | MLS       |
| '6'           | Musicals  |
| '7'           | Plays     |
| '8'           | Opera     |
| '9'           | Pop       |
+---------------+-----------+
```

Die Funktion REGEXP_COUNT

Durchsucht eine Zeichenfolge nach einem regulären Ausdrucksmuster und gibt eine Ganzzahl zurück, die die Häufigkeit angibt, mit der das angegebene Muster in der Zeichenfolge auftritt. Wenn keine Übereinstimmung gefunden wird, gibt die Funktion `0` zurück. Weitere Informationen zu regulären Ausdrücken finden Sie unter POSIX-Operatoren und Reguläre Ausdrücke in Wikipedia.

Syntax

```
REGEXP_COUNT( source_string, pattern [, position [, parameters ] ] )
```

Argumente

source_string

Eine `CHAR`- oder `VARCHAR`-Zeichenfolge.

pattern

Ein UTF-8-String-Literal, das ein Muster für reguläre Ausdrücke darstellt. Weitere Informationen finden Sie unter POSIX-Operatoren.

position

(optional) Ein positiver `INTEGER`, die die Position innerhalb von source_string angibt, an der die Suche gestartet werden soll. Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Der Standardwert ist 1. Wenn position kleiner als 1 ist, beginnt die Suche mit dem ersten Zeichen von source_string. Wenn position größer als die Anzahl der Zeichen in source_string ist, ist das Ergebnis `0`.

parameters (Parameter

(Optional) Ein oder mehrere Zeichenfolgenliterale, die angeben, wie die Funktion mit dem Muster übereinstimmt. Die folgenden Werte sind möglich:

- c – Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung durchführen. Die Standardeinstellung ist, beim Abgleich die Groß- und Kleinschreibung zu beachten.
- i – Übereinstimmung ohne Unterscheidung von Groß- und Kleinschreibung durchführen.
- p – Das Musters mit einem PCRE-Dialekt (Perl Compatible Regular Expression) interpretieren. Weitere Informationen zu PCRE finden Sie unter Perl-kompatible reguläre Ausdrücke in Wikipedia.

Rückgabetyp

INTEGER

Beispiele

Verwenden Sie das folgende Beispiel, um die Häufigkeit zu zählen, mit der eine Folge aus drei Buchstaben auftritt.

```
SELECT REGEXP_COUNT('abcdefghijklmnopqrstuvwxyz', '[a-z]{3}');

+--------------+
| regexp_count |
+--------------+
|            8 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um die Anzahl der Vorkommen der Zeichenfolge F0X zu zählen, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_COUNT('the fox', 'FOX', 1, 'i');

+--------------+
| regexp_count |
+--------------+
|            1 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um ein im PCRE-Dialekt geschriebenes Muster zu verwenden, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Dieses Beispiel verwendet den Operator ?=, der eine bestimmte Lookahead-Konnotation in PCRE hat. In diesem Beispiel wird die Anzahl der Vorkommen solcher Wörter gezählt, wobei zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_COUNT('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 'p');

+--------------+
| regexp_count |
+--------------+
|            2 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um ein im PCRE-Dialekt geschriebenes Muster zu verwenden, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator ?= verwendet, der eine bestimmte Konnotation in PCRE hat. In diesem Beispiel wird die Anzahl der Vorkommen solcher Wörter gezählt. Dies unterscheidet sich insofern vom vorherigen Beispiel, als dass nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_COUNT('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 'ip');

+--------------+
| regexp_count |
+--------------+
|            3 |
+--------------+
```

Im folgenden Beispiel werden Daten aus der Tabelle USERS in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Häufigkeit zu zählen, mit der der Name der obersten Domain entweder `org` oder `edu` ist.

```
SELECT email, REGEXP_COUNT(email,'@[^.]*\.(org|edu)') FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------+--------------+
|                     email                    | regexp_count |
+----------------------------------------------+--------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |            1 |
| Suspendisse.tristique@nonnisiAenean.edu       |            1 |
| amet.faucibus.ut@condimentumegetvolutpat.ca   |            0 |
| sed@lacusUtnec.ca                             |            0 |
+----------------------------------------------+--------------+
```

Die Funktion REGEXP_INSTR

Durchsucht eine Zeichenfolge nach einem regulären Ausdrucksmuster und gibt eine Ganzzahl zurück, die die Anfangs- oder Endposition der übereinstimmenden Unterzeichenfolge angibt. Wenn keine Übereinstimmung gefunden wird, gibt die Funktion `0` zurück. REGEXP_INSTR ist der Funktion POSITION ähnlich. Sie können damit jedoch eine Zeichenfolge nach einem regulären Ausdrucksmuster durchsuchen. Weitere Informationen zu regulären Ausdrücken finden Sie unter POSIX-Operatoren und Reguläre Ausdrücke in Wikipedia.

Syntax

```
REGEXP_INSTR( source_string, pattern [, position [, occurrence] [, option [, parameters
 ] ] ] ] )
```

Argumente

source_string

Ein Zeichenfolgenausdruck (beispielsweise ein Spaltenname), der gesucht werden soll.

pattern

Ein UTF-8-String-Literal, das ein Muster für reguläre Ausdrücke darstellt. Weitere Informationen finden Sie unter POSIX-Operatoren.

position

(optional) Ein positiver `INTEGER`, die die Position innerhalb von source_string angibt, an der die Suche gestartet werden soll. Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Der Standardwert ist 1. Wenn position kleiner als 1 ist, beginnt die Suche mit dem ersten Zeichen von source_string. Wenn position größer als die Anzahl der Zeichen in source_string ist, ist das Ergebnis `0`.

occurrence

(Optional) Eine positive `INTEGER`, die angibt, welches Vorkommen des Musters verwendet werden soll. REGEXP_INSTR überspringt die ersten *occurrence*`-1` Übereinstimmungen. Der Standardwert ist 1. Wenn occurrence kleiner als 1 oder größer als die Anzahl der Zeichen in source_string ist, wird die Suche ignoriert und das Ergebnis ist `0`.

option

(Optional) Ein Wert, der angibt, ob die Position des ersten Zeichens der Übereinstimmung (0) oder die Position des ersten Zeichens nach dem Ende der Übereinstimmung (1) zurückgegeben werden soll. Ein Wert ungleich null entspricht 1. Der Standardwert ist 0.

parameters (Parameter

(Optional) Ein oder mehrere Zeichenfolgenliterale, die angeben, wie die Funktion mit dem Muster übereinstimmt. Die folgenden Werte sind möglich:

- c – Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung durchführen. Die Standardeinstellung ist, beim Abgleich die Groß- und Kleinschreibung zu beachten.
- i – Übereinstimmung ohne Unterscheidung von Groß- und Kleinschreibung durchführen.
- e – Teilzeichenfolge mittels eines Unterausdrucks extrahieren.

  Wenn pattern einen Unterausdruck enthält, sucht REGEXP_INSTR nach einer Teilzeichenfolge, die mit dem ersten Unterausdruck in pattern übereinstimmt. REGEXP_INSTR berücksichtigt nur den ersten Unterausdruck. Zusätzliche Unterausdrücke werden ignoriert. Wenn das Muster über keinen Unterausdruck verfügt, ignoriert REGEXP_INSTR den Parameter 'e'.
- p – Das Musters mit einem PCRE-Dialekt (Perl Compatible Regular Expression) interpretieren. Weitere Informationen zu PCRE finden Sie unter Perl-kompatible reguläre Ausdrücke in Wikipedia.

Rückgabetyp

Ganzzahl

Beispiele

In den folgenden Beispielen werden Daten aus der Tabelle USERS in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um nach dem Zeichen @ zu suchen, mit dem Domain-Namen beginnen. Anschließend wird die Anfangsposition der ersten Übereinstimmung zurückgegeben.

```
SELECT email, REGEXP_INSTR(email, '@[^.]*')
FROM users
ORDER BY userid LIMIT 4;
```

```
+-----------------------------------------------+--------------+
|                     email                     | regexp_instr |
+-----------------------------------------------+--------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |           21 |
| Suspendisse.tristique@nonnisiAenean.edu       |           22 |
| amet.faucibus.ut@condimentumegetvolutpat.ca   |           17 |
| sed@lacusUtnec.ca                             |            4 |
+-----------------------------------------------+--------------+
```

Verwenden Sie das folgende Beispiel, um nach Varianten des Worts `Center` zu suchen. Anschließend wird die Anfangsposition der ersten Übereinstimmung zurückgegeben.

```
SELECT venuename, REGEXP_INSTR(venuename,'[cC]ent(er|re)$')
FROM venue
WHERE REGEXP_INSTR(venuename,'[cC]ent(er|re)$') > 0
ORDER BY venueid LIMIT 4;

+----------------------+--------------+
|       venuename      | regexp_instr |
+----------------------+--------------+
| The Home Depot Center |           16 |
| Izod Center           |            6 |
| Wachovia Center       |           10 |
| Air Canada Centre     |           12 |
+----------------------+--------------+
```

Verwenden Sie das folgende Beispiel, um die Anfangsposition des ersten Vorkommens der Zeichenfolge `FOX` zu finden, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_INSTR('the fox', 'FOX', 1, 1, 0, 'i');

+--------------+
| regexp_instr |
+--------------+
|            5 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um ein im PCRE-Dialekt geschriebenes Muster zu verwenden, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator `?=` verwendet, der eine bestimmte Lookahead-Konnotation in PCRE hat. In diesem Beispiel wird die Anfangsposition des zweiten Wortes gefunden.

```
SELECT REGEXP_INSTR('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 0, 'p');

+--------------+
| regexp_instr |
+--------------+
|           21 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um ein im PCRE-Dialekt geschriebenes Muster zu verwenden, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator ?= verwendet, der eine bestimmte Lookahead-Konnotation in PCRE hat. In diesem Beispiel wird die Anfangsposition des zweiten Worts gefunden. Dies unterscheidet sich insofern vom vorherigen Beispiel, als dass nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_INSTR('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 0, 'ip');

+--------------+
| regexp_instr |
+--------------+
|           15 |
+--------------+
```

Die Funktion REGEXP_REPLACE

Durchsucht eine Zeichenfolge nach einem regulären Ausdrucksmuster und ersetzt jedes Vorkommen des Musters durch die angegebene Zeichenfolge. REGEXP_REPLACE ist Die Funktion REPLACE ähnlich. Sie können jedoch eine Zeichenfolge nach einem regulären Ausdrucksmuster durchsuchen. Weitere Informationen zu regulären Ausdrücken finden Sie unter POSIX-Operatoren und Reguläre Ausdrücke in Wikipedia.

REGEXP_REPLACE ist Die Funktion TRANSLATE und Die Funktion REPLACE ähnlich. TRANSLATE führt jedoch mehrere Einzelzeichenersetzungen durch und REPLACE ersetzt eine ganze Zeichenfolge durch eine andere Zeichenfolge. Mit REGEXP_REPLACE können Sie dagegen eine Zeichenfolge nach einem regulären Ausdrucksmuster durchsuchen.

Syntax

```
REGEXP_REPLACE( source_string, pattern [, replace_string [ , position [, parameters
 ] ] ] )
```

Argumente

source_string

Ein CHAR- oder VARCHAR-Zeichenfolgeausdruck (beispielsweise ein Spaltenname), der gesucht werden soll.

pattern

Ein UTF-8-String-Literal, das ein Muster für reguläre Ausdrücke darstellt. Weitere Informationen finden Sie unter POSIX-Operatoren.

replace_string

(Optional) Ein CHAR- oder VARCHAR-Zeichenfolgeausdruck (beispielsweise ein Spaltenname), der jedes Vorkommen eines Musters ersetzt. Der Standardwert ist eine leere Zeichenfolge ("").

position

(Optional) Eine positive Ganzzahl, die die Position innerhalb von source_string angibt, an der die Suche gestartet werden soll. Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Der Standardwert ist 1. Wenn position kleiner als 1 ist, beginnt die Suche mit dem ersten Zeichen von source_string. Wenn position größer als die Anzahl der Zeichen in source_string ist, ist das Ergebnis source_string.

parameters (Parameter

(Optional) Ein oder mehrere Zeichenfolgenliterale, die angeben, wie die Funktion mit dem Muster übereinstimmt. Die folgenden Werte sind möglich:

- c – Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung durchführen. Die Standardeinstellung ist, beim Abgleich die Groß- und Kleinschreibung zu beachten.
- i – Übereinstimmung ohne Unterscheidung von Groß- und Kleinschreibung durchführen.
- p – Das Musters mit einem PCRE-Dialekt (Perl Compatible Regular Expression) interpretieren. Weitere Informationen zu PCRE finden Sie unter Perl-kompatible reguläre Ausdrücke in Wikipedia.

Rückgabetyp

VARCHAR

Wenn pattern oder replace_string `NULL` sind, ist der Rückgabewert `NULL`.

Beispiele

Verwenden Sie das folgende Beispiel, um alle Vorkommen der Zeichenfolge `FOX` innerhalb des Werts `quick brown fox` zu ersetzen, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_REPLACE('the fox', 'FOX', 'quick brown fox', 1, 'i');

+---------------------+
|   regexp_replace    |
+---------------------+
| the quick brown fox |
+---------------------+
```

Im folgenden Beispiel wird ein im PCRE-Dialekt geschriebenes Muster verwendet, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator `?=` verwendet, der eine bestimmte Lookahead-Konnotation in PCRE hat. Verwenden Sie das folgende Beispiel, um alle Vorkommen eines solchen Worts durch den Wert `[hidden]` zu ersetzen.

```
SELECT REGEXP_REPLACE('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 '[hidden]', 1, 'p');

+-------------------------------+
|        regexp_replace         |
+-------------------------------+
| [hidden] plain A1234 [hidden] |
+-------------------------------+
```

Im folgenden Beispiel wird ein im PCRE-Dialekt geschriebenes Muster verwendet, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator `?=` verwendet, der eine bestimmte Lookahead-Konnotation in PCRE hat. Verwenden Sie das folgende Beispiel, um alle Vorkommen eines solchen Worts mit dem Wert `[hidden]` zu ersetzen. Dies unterscheidet sich insofern vom vorherigen Beispiel, als dass nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT REGEXP_REPLACE('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 '[hidden]', 1, 'ip');

+----------------------------------+
|          regexp_replace          |
+----------------------------------+
| [hidden] plain [hidden] [hidden] |
+----------------------------------+
```

In den folgenden Beispielen werden Daten aus der Tabelle USERS in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um @ und den Domain-Name aus E-Mail-Adressen zu löschen.

```
SELECT email, REGEXP_REPLACE(email, '@.*\\.(org|gov|com|edu|ca)$')
FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------+-----------------------+
|                    email                     |    regexp_replace     |
+----------------------------------------------+-----------------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu | Etiam.laoreet.libero  |
| Suspendisse.tristique@nonnisiAenean.edu       | Suspendisse.tristique |
| amet.faucibus.ut@condimentumegetvolutpat.ca   | amet.faucibus.ut      |
| sed@lacusUtnec.ca                             | sed                   |
+----------------------------------------------+-----------------------+
```

Verwenden Sie das folgende Beispiel, um die Domain-Namen von E-Mail-Adressen durch `internal.company.com` zu ersetzen.

```
SELECT email, REGEXP_REPLACE(email, '@.*\\.[[:alpha:]]{2,3}','@internal.company.com')
FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------
+-------------------------------------------+
|                    email                  |              regexp_replace
      |
+----------------------------------------------
+-------------------------------------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |
 Etiam.laoreet.libero@internal.company.com  |
```

```
| Suspendisse.tristique@nonnisiAenean.edu       |
 Suspendisse.tristique@internal.company.com |
| amet.faucibus.ut@condimentumegetvolutpat.ca   | amet.faucibus.ut@internal.company.com
      |
| sed@lacusUtnec.ca                             | sed@internal.company.com
      |
+-----------------------------------------------
+-----------------------------------------+
```

Die Funktion REGEXP_SUBSTR

Gibt Zeichen aus einer Zeichenfolge zurück, indem diese nach einem regulären Ausdrucksmuster durchsucht wird. REGEXP_SUBSTR ist der Funktion Die Funktion SUBSTRING ähnlich. Sie können jedoch eine Zeichenfolge nach einem regulären Ausdrucksmuster durchsuchen. Wenn die Funktion den regulären Ausdruck keinem Zeichen in der Zeichenfolge zuordnen kann, wird eine leere Zeichenfolge zurückgegeben. Weitere Informationen zu regulären Ausdrücken finden Sie unter POSIX-Operatoren und Reguläre Ausdrücke in Wikipedia.

Syntax

```
REGEXP_SUBSTR( source_string, pattern [, position [, occurrence [, parameters ] ] ] )
```

Argumente

source_string

Ein Zeichenfolgeausdruck, der durchsucht werden soll.

pattern

Ein UTF-8-String-Literal, das ein Muster für reguläre Ausdrücke darstellt. Weitere Informationen finden Sie unter POSIX-Operatoren.

position

Eine positive Ganzzahl, die die Position innerhalb von source_string angibt, an der die Suche gestartet werden soll. Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Der Standardwert ist 1. Wenn position kleiner als 1 ist, beginnt die Suche mit dem ersten Zeichen von source_string. Wenn position größer als die Anzahl der Zeichen in source_string ist, ist das Ergebnis eine leere Zeichenfolge ("").

occurrence

Eine positive Ganzzahl, die angibt, welches Vorkommen des Musters verwendet werden soll. REGEXP_SUBSTR überspringt die erste occurrence -1 Übereinstimmungen. Der Standardwert ist 1. Wenn occurrence kleiner als 1 oder größer als die Anzahl der Zeichen in source_string ist, wird die Suche ignoriert und das Ergebnis ist NULL.

parameters (Parameter

Ein oder mehrere Zeichenfolgenliterale, die angeben, wie die Funktion mit dem Muster übereinstimmt. Die folgenden Werte sind möglich:

- c – Übereinstimmung mit Unterscheidung von Groß- und Kleinschreibung durchführen. Die Standardeinstellung ist, beim Abgleich die Groß- und Kleinschreibung zu beachten.
- i – Übereinstimmung ohne Unterscheidung von Groß- und Kleinschreibung durchführen.
- e – Teilzeichenfolge mittels eines Unterausdrucks extrahieren.

  Wenn pattern einen Unterausdruck enthält, sucht REGEXP_SUBSTR nach einer Teilzeichenfolge, die mit dem ersten Unterausdruck in pattern übereinstimmt. Ein Unterausdruck ist ein Ausdruck innerhalb des Musters, der in Klammern gesetzt ist. Bei dem Muster `'This is a (\\w+)'` beispielsweise wird der erste Ausdruck mit der Zeichenfolge `'This is a '`, gefolgt von einem Wort abgeglichen. Anstatt ein Muster zurückzugeben, gibt REGEXP_SUBSTR mit dem Parameter e nur die Zeichenfolge innerhalb des Unterausdrucks zurück.

  REGEXP_SUBSTR berücksichtigt nur den ersten Unterausdruck. Zusätzliche Unterausdrücke werden ignoriert. Wenn das Muster über keinen Unterausdruck verfügt, ignoriert REGEXP_SUBSTR den Parameter 'e'.
- p – Das Musters mit einem PCRE-Dialekt (Perl Compatible Regular Expression) interpretieren. Weitere Informationen zu PCRE finden Sie unter Perl-kompatible reguläre Ausdrücke in Wikipedia.

Rückgabetyp

VARCHAR

Beispiele

Im folgenden Beispiel wird der E-Mail-Adresse-Abschnitt zwischen dem Zeichen @ und der Domänenerweiterung zurückgegeben. Die abgefragten `users`-Daten stammen aus den Amazon-Redshift-Beispieldaten. Weitere Informationen finden Sie unter Beispieldatenbank.

```
SELECT email, regexp_substr(email,'@[^.]*')
FROM users
ORDER BY userid LIMIT 4;

                     email                      |      regexp_substr
------------------------------------------------+--------------------------
 Suspendisse.tristique@nonnisiAenean.edu        | @nonnisiAenean
 amet.faucibus.ut@condimentumegetvolutpat.ca    | @condimentumegetvolutpat
 sed@lacusUtnec.ca                              | @lacusUtnec
 Cum@accumsan.com                               | @accumsan
```

Im folgenden Beispiel wird der Teil der Eingabe zurückgegeben, der dem ersten Vorkommen der Zeichenfolge `FOX` entspricht, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT regexp_substr('the fox', 'FOX', 1, 1, 'i');

 regexp_substr
---------------
 fox
```

Im folgenden Beispiel wird der Teil der Eingabe zurückgegeben, der dem zweiten Vorkommen der Zeichenfolge `FOX` entspricht, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird. Das Ergebnis ist `NULL` (leer), weil es kein zweites Vorkommen gibt.

```
SELECT regexp_substr('the fox', 'FOX', 1, 2, 'i');

 regexp_substr
---------------

```

Das folgende Beispiel gibt den ersten Teil der Eingabe zurück, der mit Kleinbuchstaben beginnt. Dies ist funktionell identisch mit derselben SELECT-Anweisung ohne den `c`-Parameter.

```
SELECT regexp_substr('THE SECRET CODE IS THE LOWERCASE PART OF 1931abc0EZ.', '[a-z]+',
 1, 1, 'c');
```

```
 regexp_substr
---------------
 abc
```

Im folgenden Beispiel wird ein im PCRE-Dialekt geschriebenes Muster verwendet, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator ?= verwendet, der eine bestimmte Lookahead-Konnotation in PCRE hat. In diesem Beispiel wird der Teil der Eingabe zurückgegeben, der dem zweiten Wort entspricht.

```
SELECT regexp_substr('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 'p');

 regexp_substr
---------------
 a1234
```

Im folgenden Beispiel wird ein im PCRE-Dialekt geschriebenes Muster verwendet, um Wörter mit mindestens einer Zahl und einem Kleinbuchstaben zu finden. Hierfür wird der Operator ?= verwendet, der eine bestimmte Lookahead-Konnotation in PCRE hat. In diesem Beispiel wird der der Teil der Eingabe zurückgegeben, der dem zweiten Wort entspricht. Dies unterscheidet sich insofern vom vorherigen Beispiel, als dass nicht zwischen Groß- und Kleinschreibung unterschieden wird.

```
SELECT regexp_substr('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 'ip');

 regexp_substr
---------------
 A1234
```

Im folgenden Beispiel wird ein Unterausdruck verwendet, um die zweite Zeichenfolge zu finden, die dem Muster `'this is a (\\w+)'` entspricht, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird. Der Unterausdruck in Klammern wird zurückgegeben.

```
SELECT regexp_substr(
               'This is a cat, this is a dog. This is a mouse.',
               'this is a (\\w+)', 1, 2, 'ie');

 regexp_substr
---------------
```

```
dog
```

Die Funktion REPEAT

Wiederholt eine Zeichenfolge mit der angegebenen Häufigkeit. Wenn der Eingabeparameter numerisch ist, wird er von REPEAT als Zeichenfolge behandelt.

Synonym mit Die Funktion REPLICATE.

Syntax

```
REPEAT(string, integer)
```

Argumente

string

Der erste Eingabeparameter ist die Zeichenfolge, die wiederholt werden soll.

integer

Der zweite Parameter ist eine INTEGER, die die Häufigkeit angibt, mit der die Zeichenfolge wiederholt werden soll.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle CATEGORY in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um den Wert der Spalte CATID in der Tabelle CATEGORY dreimal zu wiederholen.

```
SELECT catid, REPEAT(catid,3)
FROM category
ORDER BY 1,2;

+-------+--------+
```

```
| catid | repeat |
+-------+--------+
|     1 |    111 |
|     2 |    222 |
|     3 |    333 |
|     4 |    444 |
|     5 |    555 |
|     6 |    666 |
|     7 |    777 |
|     8 |    888 |
|     9 |    999 |
|    10 | 101010 |
|    11 | 111111 |
+-------+--------+
```

Die Funktion REPLACE

Ersetzt alle Vorkommen eines Satzes von Zeichen innerhalb einer vorhandenen Zeichenfolge durch andere angegebene Zeichen.

REPLACE ist Die Funktion TRANSLATE und Die Funktion REGEXP_REPLACE ähnlich. TRANSLATE führt jedoch mehrere Einzelzeichenersetzungen durch und REPLACE ersetzt eine ganze Zeichenfolge durch eine andere Zeichenfolge. REPLACE ersetzt dagegen eine ganze Zeichenfolge durch eine andere Zeichenfolge.

Syntax

```
REPLACE(string, old_chars, new_chars)
```

Argumente

string

Die CHAR- oder VARCHAR-Zeichenfolge, die durchsucht werden soll.

old_chars

Die CHAR- oder VARCHAR-Zeichenfolge, die ersetzt werden soll.

new_chars

Die neue CHAR- oder VARCHAR-Zeichenfolge, die old_string ersetzt.

Rückgabetyp

VARCHAR

Wenn old_chars oder new_chars NULL sind, ist der Rückgabewert NULL.

Beispiele

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle CATEGORY in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Zeichenfolge Shows in Theatre im Feld CATGROUP zu konvertieren.

```
SELECT catid, catgroup, REPLACE(catgroup, 'Shows', 'Theatre')
FROM category
ORDER BY 1,2,3;

+-------+----------+----------+
| catid | catgroup | replace  |
+-------+----------+----------+
|     1 | Sports   | Sports   |
|     2 | Sports   | Sports   |
|     3 | Sports   | Sports   |
|     4 | Sports   | Sports   |
|     5 | Sports   | Sports   |
|     6 | Shows    | Theatre  |
|     7 | Shows    | Theatre  |
|     8 | Shows    | Theatre  |
|     9 | Concerts | Concerts |
|    10 | Concerts | Concerts |
|    11 | Concerts | Concerts |
+-------+----------+----------+
```

Die Funktion REPLICATE

Synonym mit der Funktion REPEAT.

Siehe Die Funktion REPEAT.

Die Funktion REVERSE

Die REVERSE-Funktion wird für eine Zeichenfolge ausgeführt und gibt die Zeichen in umgekehrter Reihenfolge wieder. Beispielsweise gibt `reverse('abcde')` `edcba` zurück. Diese Funktion kann auf numerische und Datumsdatentypen sowie Zeichendatentypen angewendet werden. In den meisten Fällen hat sie jedoch für Zeichenfolgen mit Zeichen praktischen Nutzen.

Syntax

```
REVERSE( expression )
```

Argument

expression

Ein Ausdruck mit einem Zeichen-, Datums-, Zeitstempel- oder numerischen Datentyp, der das Ziel der Zeichenumkehrung darstellt. Alle Ausdrücke werden implizit in VARCHAR-Zeichenfolgen konvertiert. Nachfolgende Leerzeichen in CHAR-Zeichenfolgen werden ignoriert.

Rückgabetyp

VARCHAR

Beispiele

In den folgenden Beispielen werden Daten aus der Tabellen USERS und SALES in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um fünf verschiedene Namen von Städten und die entsprechenden Umkehrungen der Namen aus der Tabelle USERS auszuwählen.

```
SELECT DISTINCT city AS cityname, REVERSE(cityname)
FROM users
ORDER BY city LIMIT 5;

+----------+----------+
| cityname | reverse  |
+----------+----------+
| Aberdeen | needrebA |
| Abilene  | enelibA  |
| Ada      | adA      |
| Agat     | tagA     |
```

```
| Agawam   | mawagA   |
+----------+----------+
```

Verwenden Sie das folgende Beispiel, um fünf Vertriebs-IDs und die entsprechenden umgekehrten IDs-Umwandlungen als Zeichenfolgen auszuwählen.

```
SELECT salesid, REVERSE(salesid)
FROM sales
ORDER BY salesid DESC LIMIT 5;

+---------+---------+
| salesid | reverse |
+---------+---------+
|  172456 |  654271 |
|  172455 |  554271 |
|  172454 |  454271 |
|  172453 |  354271 |
|  172452 |  254271 |
+---------+---------+
```

Die Funktion RTRIM

Die RTRIM-Funktion kürzt einen angegebenen Satz von Zeichen ab dem Ende einer Zeichenfolge. Entfernt die längste Zeichenfolge, die nur Zeichen aus der Liste der Trimm-Zeichen enthält. Die Kürzung ist abgeschlossen, wenn in der Eingabezeichenfolge kein Trimm-Zeichen enthalten ist.

Syntax

```
RTRIM( string, trim_chars )
```

Argumente

string

Eine Zeichenfolgenspalte, ein Ausdruck oder ein Zeichenfolgenliteral, die/der/das gekürzt werden soll.

trim_chars

Eine Zeichenfolgenspalte, ein Ausdruck oder ein Zeichenfolgenliteral, die/der/das die Zeichen darstellt, die am Ende von string gekürzt werden sollen. Wenn nicht angegeben, wird ein Leerzeichen als Trimm-Zeichen verwendet.

Rückgabetyp

Eine Zeichenfolge mit demselben Datentyp wie das string-Argument.

Beispiel

Im folgenden Beispiel werden Leerzeichen am Anfang und am Ende aus der Zeichenfolge entfernt `' abc '`:

```
select '     abc    ' as untrim, rtrim('     abc    ') as trim;

untrim    | trim
----------+------
   abc    |    abc
```

Im folgenden Beispiel werden die Zeichenfolgen `'xyz'` am Ende der Zeichenfolge `'xyzaxyzbxyzcxyz'` entfernt. Die Zeichenfolgen `'xyz'` am Ende werden entfernt, entsprechende Zeichenfolgen innerhalb dieser Zeichenfolge jedoch nicht.

```
select 'xyzaxyzbxyzcxyz' as untrim,
rtrim('xyzaxyzbxyzcxyz', 'xyz') as trim;

     untrim      |   trim
-----------------+-----------
 xyzaxyzbxyzcxyz | xyzaxyzbxyzc
```

Im folgenden Beispiel werden die Teile am Ende der Zeichenfolge `'setuphistorycassettes'` entfernt, die mit einem der Zeichen in der trim_chars-Liste `'tes'` übereinstimmen. Alle `t`, `e` oder `s` am Ende der Eingabezeichenfolge, die vor einem anderen Zeichen stehen, das nicht in der trim_chars-Liste enthalten ist, werden entfernt.

```
SELECT rtrim('setuphistorycassettes', 'tes');

     rtrim
-----------------
 setuphistoryca
```

Im folgenden Beispiel werden die Zeichen „Park“ ab dem Ende von VENUENAME gekürzt, wenn vorhanden:

```
select venueid, venuename, rtrim(venuename, 'Park')
```

```
from venue
order by 1, 2, 3
limit 10;

venueid |         venuename          |          rtrim
--------+----------------------------+-------------------------
      1 | Toyota Park                | Toyota
      2 | Columbus Crew Stadium      | Columbus Crew Stadium
      3 | RFK Stadium                | RFK Stadium
      4 | CommunityAmerica Ballpark  | CommunityAmerica Ballp
      5 | Gillette Stadium           | Gillette Stadium
      6 | New York Giants Stadium    | New York Giants Stadium
      7 | BMO Field                  | BMO Field
      8 | The Home Depot Center      | The Home Depot Cente
      9 | Dick's Sporting Goods Park | Dick's Sporting Goods
     10 | Pizza Hut Park             | Pizza Hut
```

Beachten Sie, dass RTRIM alle `P`, `a`, `r` oder `k` entfernt, wenn sie sich am Ende eines VENUENAME befinden.

Funktion SOUNDEX

Die Funktion SOUNDEX gibt den amerikanischen Soundex-Wert zurück, der aus dem ersten Buchstaben der Eingabezeichenfolge gefolgt von einer 3-stelligen Kodierung der Laute besteht, die die englische Aussprache der angegebenen Zeichenfolge repräsentieren. Zum Beispiel haben `Smith` und `Smyth` den gleichen Soundex-Wert.

Syntax

```
SOUNDEX(string)
```

Argumente

string

Sie geben eine CHAR- oder VARCHAR-Zeichenfolge an, die Sie in einen amerikanischen Soundex-Codewert konvertieren möchten.

Rückgabetyp

VARCHAR(4)

Nutzungshinweise

Die Funktion SOUNDEX konvertiert nur englische alphabetische ASCII-Zeichen in Klein- und Großbuchstaben, einschließlich a–z und A–Z. SOUNDEX ignoriert andere Zeichen. SOUNDEX gibt einen einzelnen Soundex-Wert für eine Zeichenfolge aus mehreren Wörtern zurück, die durch Leerzeichen getrennt sind.

```
SELECT SOUNDEX('AWS Amazon');

+---------+
| soundex |
+---------+
| A252    |
+---------+
```

SOUNDEX gibt eine leere Zeichenfolge zurück, wenn die Eingabezeichenfolge keine englischen Buchstaben enthält.

```
SELECT SOUNDEX('+-*/%');

+---------+
| soundex |
+---------+
|         |
+---------+
```

Beispiele

Verwenden Sie das folgende Beispiel, um den Soundex-Wert für `Amazon` zurückzugeben.

```
SELECT SOUNDEX('Amazon');

+---------+
| soundex |
+---------+
| A525    |
+---------+
```

Verwenden Sie das folgende Beispiel, um den Soundex-Wert für `smith` und `smyth` zurückzugeben. Beachten Sie, dass die Soundex-Werte identisch sind.

```
SELECT SOUNDEX('smith'), SOUNDEX('smyth');

+-------+-------+
| smith | smyth |
+-------+-------+
| S530  | S530  |
+-------+-------+
```

Die Funktion SPLIT_PART

Teilt eine Zeichenfolge am angegebenen Trennzeichen und gibt den Teil an der angegebenen Position zurück.

Syntax

```
SPLIT_PART(string, delimiter, position)
```

Argumente

string

Eine Zeichenfolgenspalte, ein Ausdruck oder ein Zeichenfolgenliteral, die/der/das geteilt werden soll. Die Zeichenfolge kann CHAR oder VARCHAR sein.

delimiter

Die Trennzeichen-Zeichenfolge, die Abschnitte des Eingabe-string angibt.

Wenn delimiter ein Literal ist, schließen Sie es in einfache Anführungszeichen ein.

position

Position des string-Abschnitts, der zurückgegeben werden soll (gezählt ab 1). Es muss sich um eine Ganzzahl größer als 0 handeln. Wenn position größer als die Anzahl der Zeichenfolgenabschnitte ist, gibt SPLIT_PART eine leere Zeichenfolge zurück. Wenn delimiter nicht in string gefunden wird, enthält der zurückgegebene Wert den Inhalt des angegebenen Teils. Dabei kann es sich um die gesamte Zeichenfolge oder einen leeren Wert handeln.

Rückgabetyp

Eine CHAR- oder VARCHAR-Zeichenfolge, identisch mit dem Parameter string.

Beispiele

Im folgenden Beispiel wird ein Zeichenfolgenliteral mithilfe des Trennzeichens $ in Teile aufgeteilt und der zweite Teil zurückgegeben.

```
select split_part('abc$def$ghi','$',2)

split_part
----------
def
```

Im folgenden Beispiel wird ein Zeichenfolgenliteral mithilfe des Trennzeichens $ in Teile aufgeteilt. Es wird eine leere Zeichenfolge zurückgegeben, da der Teil 4 nicht gefunden wurde.

```
select split_part('abc$def$ghi','$',4)

split_part
----------

```

Im folgenden Beispiel wird ein Zeichenfolgenliteral mithilfe des Trennzeichens # in Teile aufgeteilt. Da das Trennzeichen nicht gefunden wurde, wird die gesamte Zeichenfolge zurückgegeben, wobei es sich um den ersten Teil handelt.

```
select split_part('abc$def$ghi','#',1)

split_part
------------
abc$def$ghi
```

Im folgenden Beispiel wird das Zeitstempelfeld LISTTIME in die Komponenten Jahr, Monat und Datum aufgeteilt.

```
select listtime, split_part(listtime,'-',1) as year,
split_part(listtime,'-',2) as month,
split_part(split_part(listtime,'-',3),' ',1) as day
from listing limit 5;

      listtime       | year | month | day
---------------------+------+-------+------
```

```
 2008-03-05 12:25:29 | 2008 | 03    | 05
 2008-09-09 08:03:36 | 2008 | 09    | 09
 2008-09-26 05:43:12 | 2008 | 09    | 26
 2008-10-04 02:00:30 | 2008 | 10    | 04
 2008-01-06 08:33:11 | 2008 | 01    | 06
```

Im folgenden Beispiel wird das Zeitstempelfeld LISTTIME ausgewählt und am Zeichen `'-'` getrennt, um den Monat zu erhalten (den zweiten Teil der Zeichenfolge LISTTIME). Anschließend wird die Zahl der Einträge für jeden Monat gezählt:

```
select split_part(listtime,'-',2) as month, count(*)
from listing
group by split_part(listtime,'-',2)
order by 1, 2;

 month | count
-------+-------
    01 | 18543
    02 | 16620
    03 | 17594
    04 | 16822
    05 | 17618
    06 | 17158
    07 | 17626
    08 | 17881
    09 | 17378
    10 | 17756
    11 | 12912
    12 | 4589
```

Die Funktion STRPOS

Gibt die Position einer Unterzeichenfolge innerhalb einer angegebenen Zeichenfolge zurück.

Ähnliche Funktionen finden Sie unter Funktion CHARINDEX und Die Funktion POSITION.

Syntax

```
STRPOS(string, substring )
```

Argumente

string

Der erste Eingabeparameter ist die CHAR- oder VARCHAR-Zeichenfolge, die durchsucht werden soll.

substring

Der zweite Parameter ist die Unterzeichenfolge, nach der innerhalb der Zeichenfolge gesucht werden soll.

Rückgabetyp

INTEGER

Die STRPOS-Funktion gibt eine INTEGER zurück, die der Position der Teilzeichenfolge entspricht (eins-basiert, nicht null-basiert). Die Position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt.

Nutzungshinweise

STRPOS gibt 0 zurück, wenn die Teilzeichenfolge nicht innerhalb der Zeichenfolge gefunden wird.

```
SELECT STRPOS('dogfish', 'fist');

+--------+
| strpos |
+--------+
|      0 |
+--------+
```

Beispiele

Verwenden Sie das folgende Beispiel, um die Position von fish innerhalb vondogfish anzuzeigen.

```
SELECT STRPOS('dogfish', 'fish');

+--------+
| strpos |
+--------+
|      4 |
```

```
+--------+
```

Verwenden Sie das folgende Beispiel, um die Tabelle SALES aus der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um die Zahl der Verkaufstransaktionen mit einer COMMISSION von mehr als 999,00 aus der Tabelle SALES zurückzugeben.

```
SELECT DISTINCT STRPOS(commission, '.'),
COUNT (STRPOS(commission, '.'))
FROM sales
WHERE STRPOS(commission, '.') > 4
GROUP BY STRPOS(commission, '.')
ORDER BY 1, 2;

+--------+-------+
| strpos | count |
+--------+-------+
|      5 |   629 |
+--------+-------+
```

Die Funktion STRTOL

Konvertiert einen Zeichenfolgenausdruck einer Nummer der angegebenen Basis in den entsprechenden Ganzzahlwert. Der konvertierte Wert muss innerhalb des signierten 64-Bit-Bereichs liegen.

Syntax

```
STRTOL(num_string, base)
```

Argumente

num_string

Zeichenfolgeausdruck einer Zahl, der konvertiert werden soll. Wenn num_string leer ist (`''`) oder mit dem Null-Zeichen (`'\0'`) beginnt, ist der konvertierte Wert `0`. Wenn num_string eine Spalte ist, die einen NULL-Wert enthält, gibt STRTOL `NULL` zurück. Die Zeichenfolge kann mit einer beliebigen Zahl von Leerzeichen beginnen, optional gefolgt von einem einzelnen Plus (+)- oder Minus (-)-Zeichen, um einen positiven oder negativen Wert anzugeben. Der Standardwert ist '+'. Wenn base 16 ist, kann die Zeichenfolge optional mit `0x` beginnen.

base

INTEGER zwischen 2 und 36.

Rückgabetyp

BIGINT

Wenn num_string null ist, gibt die Funktion NULL zurück.

Beispiele

Verwenden Sie die folgenden Beispiele, um Zeichenfolgen- und Basiswertpaare in Ganzzahlen zu konvertieren.

```
SELECT STRTOL('0xf',16);

+--------+
| strtol |
+--------+
|     15 |
+--------+

SELECT STRTOL('abcd1234',16);

+------------+
|   strtol   |
+------------+
| 2882343476 |
+------------+

SELECT STRTOL('1234567', 10);

+---------+
| strtol  |
+---------+
| 1234567 |
+---------+

SELECT STRTOL('1234567', 8);

+--------+
```

```
| strtol |
+--------+
| 342391 |
+--------+

SELECT STRTOL('110101', 2);

+--------+
| strtol |
+--------+
|     53 |
+--------+

SELECT STRTOL('\0', 2);

+--------+
| strtol |
+--------+
|      0 |
+--------+
```

Die Funktion SUBSTRING

Gibt die Teilmenge einer Zeichenfolge basierend auf der angegebenen Startposition zurück.

Wenn es sich bei der Eingabe um eine Zeichenfolge handelt, basieren die Startposition und die Anzahl der extrahierten Zeichen auf Zeichen, nicht auf Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Wenn es sich bei der Eingabe um einen binären Ausdruck handelt, basieren die Startposition und die extrahierte Teilzeichenfolge auf Bytes. Sie können keine negative Länge angeben. Sie können jedoch eine negative Startposition angeben.

Syntax

```
SUBSTRING(character_string FROM start_position [ FOR number_characters ] )
```

```
SUBSTRING(character_string, start_position, number_characters )
```

```
SUBSTRING(binary_expression, start_byte, number_bytes )
```

```
SUBSTRING(binary_expression, start_byte )
```

Argumente

character_string

Die Zeichenfolge, die durchsucht werden soll. Datentypen, die keine Zeichen sind, werden als Zeichenfolge behandelt.

start_position

Die Position innerhalb der Zeichenfolge, an der die Extrahierung gestartet werden soll, beginnend mit 1. Die start_position basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Diese Zahl kann negativ sein.

number_characters

Die Anzahl der Zeichen, die extrahiert werden soll (die Länge der Unterzeichenfolge). Die number_characters basiert auf der Anzahl der Zeichen, nicht der Bytes. Daher werden Zeichen mit mehreren Bytes als einzelne Zeichen gezählt. Diese Zahl darf nicht negativ sein.

binary_expression

Der binary_expression des zu durchsuchenden Datentyps VARBYTE.

start_byte

Die Position innerhalb des Binärausdrucks, an der die Extrahierung gestartet werden soll, beginnend mit 1. Diese Zahl kann negativ sein.

number_bytes

Die Anzahl der Bytes, die extrahiert werden sollen, also die Länge der Unterzeichenfolge. Diese Zahl darf nicht negativ sein.

Rückgabetyp

Je nach Eingabe VARCHAR oder VARBYTE.

Nutzungshinweise

Im Folgenden finden Sie einige Beispiele dafür, wie Sie start_position und number_characters verwenden können, um Teilzeichenfolgen aus verschiedenen Positionen in einer Zeichenfolge zu extrahieren.

Im folgenden Beispiel wird eine Zeichenfolge mit vier Zeichen zurückgegeben, beginnend mit dem sechsten Zeichen.

```
select substring('caterpillar',6,4);
substring
-----------
pill
(1 row)
```

Wenn die Startposition + number_characters die Länge der Zeichenfolge überschreiten, gibt SUBSTRING eine Unterzeichenfolge ab start_position bis zum Ende der Zeichenfolge zurück. Beispiel:

```
select substring('caterpillar',6,8);
substring
-----------
pillar
(1 row)
```

Wenn `start_position` negativ oder 0 ist, gibt die Funktion SUBSTRING eine Unterzeichenfolge ab dem ersten Zeichen der Zeichenfolge mit der Länge `start_position` + `number_characters` -1 zurück. Beispiel:

```
select substring('caterpillar',-2,6);
substring
-----------
cat
(1 row)
```

Wenn `start_position` + `number_characters` -1 gleich oder kleiner als null ist, gibt SUBSTRING eine leere Zeichenfolge zurück. Beispiel:

```
select substring('caterpillar',-5,4);
substring
-----------

(1 row)
```

Beispiele

Im folgenden Beispiel wird der Monat aus der Zeichenfolge LISTTIME in der Tabelle LISTING zurückgegeben:

```
select listid, listtime,
substring(listtime, 6, 2) as month
from listing
order by 1, 2, 3
limit 10;

 listid |      listtime       | month
--------+---------------------+-------
      1 | 2008-01-24 06:43:29 | 01
      2 | 2008-03-05 12:25:29 | 03
      3 | 2008-11-01 07:35:33 | 11
      4 | 2008-05-24 01:18:37 | 05
      5 | 2008-05-17 02:29:11 | 05
      6 | 2008-08-15 02:08:13 | 08
      7 | 2008-11-15 09:38:15 | 11
      8 | 2008-11-09 05:07:30 | 11
      9 | 2008-09-09 08:03:36 | 09
     10 | 2008-06-17 09:44:54 | 06
(10 rows)
```

Im folgenden Beispiel wird das Gleiche wie oben gezeigt, jedoch mit der Option FROM...FOR:

```
select listid, listtime,
substring(listtime from 6 for 2) as month
from listing
order by 1, 2, 3
limit 10;

 listid |      listtime       | month
--------+---------------------+-------
      1 | 2008-01-24 06:43:29 | 01
      2 | 2008-03-05 12:25:29 | 03
      3 | 2008-11-01 07:35:33 | 11
      4 | 2008-05-24 01:18:37 | 05
      5 | 2008-05-17 02:29:11 | 05
      6 | 2008-08-15 02:08:13 | 08
      7 | 2008-11-15 09:38:15 | 11
      8 | 2008-11-09 05:07:30 | 11
      9 | 2008-09-09 08:03:36 | 09
     10 | 2008-06-17 09:44:54 | 06
(10 rows)
```

Sie können SUBSTRING nicht verwenden, um das Präfix einer Zeichenfolge, die möglicherweise Multibyte-Zeichen enthält, auf vorhersehbare Weise zu extrahieren, da Sie die Länge einer Multibyte-Zeichenfolge anhand der Anzahl der Bytes und nicht anhand der Anzahl der Zeichen angeben müssen. Um das Anfangssegment einer Zeichenfolge auf der Basis der Länge in Bytes zu extrahieren, können Sie die Zeichenfolge in (byte_length) umwandeln, um die Zeichenfolge abzuschneiden, wobei byte_length die erforderliche Länge ist. Im folgenden Beispiel werden die ersten 5 Bytes aus der Zeichenfolge extrahiert `'Fourscore and seven'`.

```
select cast('Fourscore and seven' as varchar(5));

varchar
-------
Fours
```

Das folgende Beispiel zeigt eine negative Startposition eines Binärwerts `abc`. Da die Startpositon -3 ist, wird die Unterzeichenfolge ab dem Anfang des Binärwerts extrahiert. Das Ergebnis wird automatisch als hexadezimale Darstellung der binären Unterzeichenfolgen angezeigt.

```
select substring('abc'::varbyte, -3);

 substring
-----------
 616263
```

Das folgende Beispiel zeigt eine 1 für die Startposition eines Binärwerts `abc`. Da keine Länge angegeben ist, wird die Zeichenfolge von der Startposition bis zum Ende der Zeichenfolge extrahiert. Das Ergebnis wird automatisch als hexadezimale Darstellung der binären Unterzeichenfolgen angezeigt.

```
select substring('abc'::varbyte, 1);

 substring
-----------
 616263
```

Das folgende Beispiel zeigt eine 3 für die Startposition eines Binärwerts `abc`. Da keine Länge angegeben ist, wird die Zeichenfolge von der Startposition bis zum Ende der Zeichenfolge extrahiert. Das Ergebnis wird automatisch als hexadezimale Darstellung der binären Unterzeichenfolgen angezeigt.

```
select substring('abc'::varbyte, 3);

 substring
-----------
 63
```

Das folgende Beispiel zeigt eine 2 für die Startposition eines Binärwerts `abc`. Die Zeichenfolge wird von der Startposition auf Position 10 extrahiert, aber das Ende der Zeichenfolge befindet sich an Position 3. Das Ergebnis wird automatisch als hexadezimale Darstellung der binären Unterzeichenfolgen angezeigt.

```
select substring('abc'::varbyte, 2, 10);

 substring
-----------
 6263
```

Das folgende Beispiel zeigt eine 2 für die Startposition eines Binärwerts `abc`. Die Zeichenfolge wird für 1 Byte aus der Startposition extrahiert. Das Ergebnis wird automatisch als hexadezimale Darstellung der binären Unterzeichenfolgen angezeigt.

```
select substring('abc'::varbyte, 2, 1);

 substring
-----------
 62
```

Das folgende Beispiel gibt den Vornamen `Ana` zurück, der nach dem letzten Leerzeichen in der Eingabezeichenfolge `Silva, Ana` erscheint.

```
select reverse(substring(reverse('Silva, Ana'), 1, position(' ' IN reverse('Silva,
 Ana'))))

 reverse
-----------
 Ana
```

Die Funktion TEXTLEN

Synonym mit der Funktion LEN.

Siehe Die Funktion LEN.

Die Funktion TRANSLATE

Ersetzt für einen bestimmten Ausdruck alle Vorkommen von angegebenen Zeichen durch angegebene Ersatzzeichen. Vorhandene Zeichen werden aufgrund Ihrer Positionen in den Argumenten characters_to_replace und characters_to_substitute zu Ersatzzeichen zugeordnet. Wenn im Argument characters_to_replace mehr Zeichen als im Argument characters_to_substitute angegeben sind, werden die zusätzlichen Zeichen aus dem Argument characters_to_replace im Rückgabewert ausgelassen.

TRANSLATE ist Die Funktion REPLACE und Die Funktion REGEXP_REPLACE ähnlich. Während REPLACE jedoch eine ganze Zeichenfolge durch eine andere Zeichenfolge ersetzt und REGEXP_REPLACE eine Zeichenfolge nach einem regulären Ausdrucksmuster durchsucht, führt TRANSLATE mehrere Einzelzeichenersetzungen aus.

Wenn ein Argument null ist, ist der Rückgabewert NULL.

Syntax

```
TRANSLATE( expression, characters_to_replace, characters_to_substitute )
```

Argumente

expression

Der Ausdruck, der übersetzt werden soll.

characters_to_replace

Eine Zeichenfolge, die die Zeichen enthält, die ersetzt werden sollen.

characters_to_substitute

Eine Zeichenfolge, die die Zeichen enthält, die ersetzt werden sollen.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um mehrere Zeichen in einer Zeichenfolge zu ersetzen.

```
SELECT TRANSLATE('mint tea', 'inea', 'osin');

+-----------+
| translate |
+-----------+
| most tin  |
+-----------+
```

In den folgenden Beispielen werden Daten aus der Tabelle USERS in der TICKIT-Beispieldatenbank verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie das folgende Beispiel, um für alle Werte in einer Spalte das Zeichen @ durch einen Punkt zu ersetzen.

```
SELECT email, TRANSLATE(email, '@', '.') as obfuscated_email
FROM users LIMIT 10;

+--------------------------------------+--------------------------------------+
|                email                 |           obfuscated_email           |
+--------------------------------------+--------------------------------------+
| Cum@accumsan.com                     | Cum.accumsan.com                     |
| lorem.ipsum@Vestibulumante.com       | lorem.ipsum.Vestibulumante.com       |
| non.justo.Proin@ametconsectetuer.edu | non.justo.Proin.ametconsectetuer.edu |
| non.ante.bibendum@porttitortellus.org | non.ante.bibendum.porttitortellus.org |
| eros@blanditatnisi.org               | eros.blanditatnisi.org               |
| augue@Donec.ca                       | augue.Donec.ca                       |
| cursus@pedeacurna.edu                | cursus.pedeacurna.edu                |
| at@Duis.com                          | at.Duis.com                          |
| quam@facilisisvitaeorci.ca           | quam.facilisisvitaeorci.ca           |
| mi.lorem@nunc.edu                    | mi.lorem.nunc.edu                    |
+--------------------------------------+--------------------------------------+
```

Verwenden Sie das folgende Beispiel, um für alle Werte in einer Spalte Leerzeichen durch Unterstriche zu ersetzen und Punkte zu entfernen.

```
SELECT city, TRANSLATE(city, ' .', '_')
FROM users
WHERE city LIKE 'Sain%' OR city LIKE 'St%'
GROUP BY city
ORDER BY city;
```

```
+----------------+---------------+
|      city      |   translate   |
+----------------+---------------+
| Saint Albans   | Saint_Albans  |
| Saint Cloud    | Saint_Cloud   |
| Saint Joseph   | Saint_Joseph  |
| Saint Louis    | Saint_Louis   |
| Saint Paul     | Saint_Paul    |
| St. George     | St_George     |
| St. Marys      | St_Marys      |
| St. Petersburg | St_Petersburg |
| Stafford       | Stafford      |
| Stamford       | Stamford      |
| Stanton        | Stanton       |
| Starkville     | Starkville    |
| Statesboro     | Statesboro    |
| Staunton       | Staunton      |
| Steubenville   | Steubenville  |
| Stevens Point  | Stevens_Point |
| Stillwater     | Stillwater    |
| Stockton       | Stockton      |
| Sturgis        | Sturgis       |
+----------------+---------------+
```

Die Funktion TRIM

Kürzt die Leerzeichen oder angegebenen Zeichen einer Zeichenfolge.

Syntax

```
TRIM( [ BOTH | LEADING | TRAILING ] [trim_chars FROM ] string )
```

Argumente

BOTH | LEADING | TRAILING

(Optional) Gibt an, von wo Zeichen gekürzt werden sollen. Verwenden Sie BOTH, um führende und nachgestellte Zeichen zu entfernen, verwenden Sie LEADING, um nur führende Zeichen zu entfernen, und verwenden Sie TRAILING, um nur nachfolgende Zeichen zu entfernen. Wenn dieser Parameter weggelassen wird, werden sowohl führende als auch nachfolgende Zeichen gekürzt.

trim_chars

(Optional) Die Zeichen, die aus der Zeichenfolge gekürzt werden sollen. Wenn dieser Parameter ausgelassen wird, werden Leerzeichen ausgeschnitten.

string

Die Zeichenfolge, die gekürzt werden soll.

Rückgabetyp

Die TRIM-Funktion gibt eine VARCHAR- oder CHAR-Zeichenfolge zurück. Wenn Sie die TRIM-Funktion mit einem SQL-Befehl verwenden, konvertiert Amazon Redshift die Ergebnisse implizit in VARCHAR. Wenn Sie die TRIM-Funktion in der SELECT-Liste für eine SQL-Funktion verwenden, konvertiert Amazon Redshift die Ergebnisse nicht implizit und Sie müssen möglicherweise eine explizite Konvertierung ausführen, um einen Fehler wegen fehlender Datentypübereinstimmung zu vermeiden. Weitere Informationen zu expliziten Konvertierungen finden Sie in den Abschnitten zu den Funktionen CAST-Funktion und CONVERT-Funktion.

Beispiele

Verwenden Sie das folgende Beispiel, um Leerzeichen am Anfang und am Ende aus der Zeichenfolge dog zu entfernen.

```
SELECT TRIM('    dog    ');

+-------+
| btrim |
+-------+
| dog   |
+-------+
```

Verwenden Sie das folgende Beispiel, um Leerzeichen am Anfang und am Ende aus der Zeichenfolge dog zu entfernen.

```
SELECT TRIM(BOTH FROM '    dog    ');

+-------+
| btrim |
+-------+
| dog   |
```

```
+-------+
```

Verwenden Sie das folgende Beispiel, um die führenden doppelten Anführungszeichen aus der Zeichenfolge ''dog'' zu entfernen.

```
SELECT TRIM(LEADING '"' FROM'"dog"');

+-------+
| ltrim |
+-------+
| dog"  |
+-------+
```

Verwenden Sie das folgende Beispiel, um die nachstehenden doppelten Anführungszeichen aus der Zeichenfolge ''dog'' zu entfernen.

```
SELECT TRIM(TRAILING '"' FROM'"dog"');

+-------+
| rtrim |
+-------+
| "dog  |
+-------+
```

TRIM entfernt alle Zeichen in trim_chars, wenn sie sich am Anfang oder am Ende von string befinden. Im folgenden Beispiel werden die Zeichen „C“, „D“ und „G“gekürzt, wenn sie sich am Anfang von VENUENAME befinden. Dabei handelt es sich um eine VARCHAR-Spalte. Weitere Informationen finden Sie unter Tabelle VENUE.

```
SELECT venueid, venuename, TRIM('CDG' FROM venuename)
FROM venue
WHERE venuename LIKE '%Park'
ORDER BY 2
LIMIT 7;

+---------+---------------------------+---------------------------+
| venueid |         venuename         |           btrim           |
+---------+---------------------------+---------------------------+
|     121 | AT&T Park                 | AT&T Park                 |
|     109 | Citizens Bank Park        | itizens Bank Park         |
|     102 | Comerica Park             | omerica Park              |
```

```
|       9 | Dick's Sporting Goods Park | ick's Sporting Goods Park |
|      97 | Fenway Park                | Fenway Park               |
|     112 | Great American Ball Park   | reat American Ball Park   |
|     114 | Miller Park                | Miller Park               |
+---------+----------------------------+---------------------------+
```

Die Funktion UPPER

Konvertiert eine Zeichenfolge in Großbuchstaben. UPPER unterstützt UTF-8-Multibyte-Zeichen bis zu einer maximalen Länge von vier Bytes pro Zeichen.

Syntax

```
UPPER(string)
```

Argumente

string

Der Eingabeparameter ist eine VARCHAR-Zeichenfolge oder ein anderer Datentyp wie CHAR, der implizit in VARCHAR konvertiert werden kann.

Rückgabetyp

Die UPPER-Funktion gibt eine Zeichenfolge zurück, die den gleichen Datentyp wie die Eingabezeichenfolge hat. Wenn es sich bei der Eingabe beispielsweise um eine VARCHAR-Zeichenfolge handelt, gibt die Funktion eine VARCHAR-Zeichenfolge zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um Daten aus die Tabelle CATEGORY in der TICKIT-Beispieldatenbank zu verwenden. Weitere Informationen finden Sie unter Beispieldatenbank.

Verwenden Sie Folgendes, um das Feld CATNAME-Feld in Großbuchstaben zu konvertieren.

```
SELECT catname, UPPER(catname)
FROM category
ORDER BY 1,2;

+-----------+-----------+
|  catname  |   upper   |
```

```
+-----------+-----------+
| Classical | CLASSICAL |
| Jazz      | JAZZ      |
| MLB       | MLB       |
| MLS       | MLS       |
| Musicals  | MUSICALS  |
| NBA       | NBA       |
| NFL       | NFL       |
| NHL       | NHL       |
| Opera     | OPERA     |
| Plays     | PLAYS     |
| Pop       | POP       |
+-----------+-----------+
```

Funktionen für SUPER-Typinformationen

Im folgenden finden Sie eine Beschreibung der Funktionen für Typinformationen für SQL, die Amazon Redshift unterstützt, um dynamische Informationen aus Eingaben des SUPER-Datentyps abzuleiten.

Themen

Die Funktion DECIMAL_PRECISION

Überprüft die Genauigkeit der maximalen Gesamtzahl der zu speichernden Dezimalstellen. Diese Zahl enthält sowohl die Vor- als auch Nachkommastellen. Die Genauigkeit liegt zwischen 1 und 38, wobei der Standardwert 38 ist.

Syntax

```
DECIMAL_PRECISION(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

INTEGER

Beispiele

Verwenden Sie das folgende Beispiel, um die Funktion DECIMAL_PRECISION auf die Tabelle t anzuwenden.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (3.14159);

SELECT DECIMAL_PRECISION(s) FROM t;

+-------------------+
| decimal_precision |
+-------------------+
|                 6 |
+-------------------+
```

Die Funktion DECIMAL_SCALE

Überprüft die Anzahl der zu speichernden Dezimalstellen nach dem Dezimaltrennzeichen. Die Skalierung liegt zwischen 0 und dem Genauigkeitspunkt, wobei der Standardwert 0 ist.

Syntax

```
DECIMAL_SCALE(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

INTEGER

Beispiele

Verwenden Sie das folgende Beispiel, um die Funktion DECIMAL_SCALE auf die Tabelle t anzuwenden.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (3.14159);

SELECT DECIMAL_SCALE(s) FROM t;

+---------------+
| decimal_scale |
+---------------+
|             5 |
+---------------+
```

Die Funktion IS_ARRAY

Überprüft, ob eine Variable ein Array ist. Die Funktion gibt `true` zurück, wenn die Variable ein Array ist. Die Funktion umfasst auch leere Arrays. Andernfalls gibt die Funktion `false` für alle anderen Werte zurück, einschließlich null.

Syntax

```
IS_ARRAY(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_ARRAY-Funktion zu überprüfen, ob `[1,2]` ein Array ist.

```
SELECT IS_ARRAY(JSON_PARSE('[1,2]'));

+----------+
| is_array |
+----------+
| true     |
+----------+
```

Die Funktion IS_BIGINT

Überprüft, ob ein Wert ein `BIGINT` ist. Die Funktion IS_BIGINT gibt `true` für Zahlen der Skala 0 im 64-Bit-Bereich zurück. Andernfalls gibt die Funktion `false` für alle anderen Werte zurück, einschließlich null und Gleitkommazahlen.

Die Funktion IS_BIGINT ist eine Obermenge von IS_INTEGER.

Syntax

```
IS_BIGINT(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_BIGINT-Funktion zu überprüfen, ob 5 ein BIGINT ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_BIGINT(s) FROM t;

+---+-----------+
| s | is_bigint |
+---+-----------+
| 5 | true      |
+---+-----------+
```

Die Funktion IS_BOOLEAN

Überprüft, ob ein Wert ein BOOLEAN ist. Die Funktion IS_BOOLEAN gibt `true` für konstante boolesche JSON-Werte zurück. Für alle anderen Werte, einschließlich null, gibt die Funktion `false` zurück.

Syntax

```
IS_BOOLEAN(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_BOOLEAN-Funktion zu überprüfen, ob TRUE ein BOOLEAN ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (TRUE);

SELECT s, IS_BOOLEAN(s) FROM t;

+------+------------+
|  s   | is_boolean |
+------+------------+
| true | true       |
+------+------------+
```

Die Funktion IS_CHAR

Überprüft, ob ein Wert ein CHAR ist. Die Funktion IS_CHAR gibt `true` für Zeichenfolgen zurück, die nur ASCII-Zeichen enthalten, da der CHAR-Typ nur Zeichen im ASCII-Format speichern kann. Für alle anderen Werte gibt die Funktion `false` zurück.

Syntax

```
IS_CHAR(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_CHAR-Funktion zu überprüfen, ob `t` ein CHAR ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES ('t');

SELECT s, IS_CHAR(s) FROM t;

+-----+---------+
|  s  | is_char |
+-----+---------+
| "t" | true    |
+-----+---------+
```

Die Funktion IS_DECIMAL

Überprüft, ob ein Wert ein `DECIMAL` ist. Die Funktion IS_DECIMAL gibt `true` für Zahlen zurück, die keine Gleitkommazahlen sind. Für alle anderen Werte, einschließlich null, gibt die Funktion `false` zurück.

Die Funktion IS_DECIMAL ist eine Obermenge von IS_BIGINT.

Syntax

```
IS_DECIMAL(super_expression)
```

Argumente

super_expression

Ein `SUPER`-Ausdruck oder eine Spalte.

Rückgabetyp

`BOOLEAN`

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_DECIMAL-Funktion zu überprüfen, ob `1.22` ein `DECIMAL` ist.

```
CREATE TABLE t(s SUPER);
```

```
INSERT INTO t VALUES (1.22);

SELECT s, IS_DECIMAL(s) FROM t;

+------+------------+
|  s   | is_decimal |
+------+------------+
| 1.22 | true       |
+------+------------+
```

Die Funktion IS_FLOAT

Überprüft, ob ein Wert eine Gleitkommazahl ist. Die Funktion IS_FLOAT gibt `true` für Gleitkommazahlen (`FLOAT4` und `FLOAT8`) zurück. Für alle anderen Werte gibt die Funktion `false` zurück.

Die Mengen IS_DECIMAL und IS_FLOAT sind voneinander getrennt.

Syntax

```
IS_FLOAT(super_expression)
```

Argumente

super_expression

Ein `SUPER`-Ausdruck oder eine Spalte.

Rückgabetyp

`BOOLEAN`

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_FLOAT-Funktion zu überprüfen, ob `2.22::FLOAT` ein `FLOAT` ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES(2.22::FLOAT);
```

```
SELECT s, IS_FLOAT(s) FROM t;

+---------+----------+
|    s    | is_float |
+---------+----------+
| 2.22e+0 | true     |
+---------+----------+
```

Die Funktion IS_INTEGER

Gibt `true` für Zahlen der Skala 0 im 32-Bit-Bereich zurück und `false` für alles andere (einschließlich Null- und Gleitkommazahlen).

Die Funktion IS_INTEGER ist eine Obermenge der Funktion IS_SMALLINT.

Syntax

```
IS_INTEGER(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_INTEGER-Funktion zu überprüfen, ob 5 ein INTEGER ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_INTEGER(s) FROM t;
```

```
+---+------------+
| s | is_integer |
+---+------------+
| 5 | true       |
+---+------------+
```

Die Funktion IS_OBJECT

Überprüft, ob eine Variable ein Objekt ist. Die Funktion IS_OBJECT gibt `true` für Objekte zurück, einschließlich leerer Objekte. Für alle anderen Werte, einschließlich null, gibt die Funktion `false` zurück.

Syntax

```
IS_OBJECT(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_OBJECT-Funktion zu überprüfen, ob `{"name": "Joe"}` ein Objekt ist.

```
CREATE TABLE t(s super);

INSERT INTO t VALUES (JSON_PARSE('{"name": "Joe"}'));

SELECT s, IS_OBJECT(s) FROM t;

+----------------+-----------+
|       s        | is_object |
+----------------+-----------+
| {"name":"Joe"} | true      |
```

```
+----------------+-----------+
```

Die Funktion IS_SCALAR

Überprüft, ob eine Variable ein Skalar ist. Die Funktion IS_SCALAR gibt `true` für jeden Wert zurück, der kein Array oder Objekt ist. Für alle anderen Werte, einschließlich null, gibt die Funktion `false` zurück.

Die Menge von IS_ARRAY, IS_OBJECT und IS_SCALAR deckt alle Werte mit Ausnahme von null ab.

Syntax

```
IS_SCALAR(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_SCLALR-Funktion zu überprüfen, ob `{"name": "Joe"}` ein Skalar ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (JSON_PARSE('{"name": "Joe"}'));

SELECT s, IS_SCALAR(s.name) FROM t;

+----------------+-----------+
|       s        | is_scalar |
+----------------+-----------+
| {"name":"Joe"} | true      |
+----------------+-----------+
```

Die Funktion IS_SMALLINT

Überprüft, ob eine Variable ein SMALLINT ist. Die Funktion IS_SMALLINT gibt `true` für Zahlen der Skala 0 im 16-Bit-Bereich zurück. Für alle anderen Werte, einschließlich null und Gleitkommazahlen, gibt die Funktion `false` zurück.

Syntax

```
IS_SMALLINT(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Ergebnis

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_SMALLINT-Funktion zu überprüfen, ob 5 ein SMALLINT ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_SMALLINT(s) FROM t;

+---+-------------+
| s | is_smallint |
+---+-------------+
| 5 | true        |
+---+-------------+
```

Die Funktion IS_VARCHAR

Überprüft, ob eine Variable ein VARCHAR ist. Die Funktion IS_VARCHAR gibt für alle Zeichenfolgen `true` zurück. Für alle anderen Werte gibt die Funktion `false` zurück.

Die Funktion IS_VARCHAR ist eine Obermenge der Funktion IS_CHAR.

Syntax

```
IS_VARCHAR(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

BOOLEAN

Beispiele

Verwenden Sie das folgende Beispiel, um mit der IS_VARCHAR-Funktion zu überprüfen, ob abc ein VARCHAR ist.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES ('abc');

SELECT s, IS_VARCHAR(s) FROM t;

+-------+------------+
|   s   | is_varchar |
+-------+------------+
| "abc" | true       |
+-------+------------+
```

Die Funktion JSON_SIZE

Die Funktion JSON_SIZE gibt die Anzahl der Bytes in dem gegebenen SUPER-Ausdruck zurück, wenn in eine Zeichenfolge serialisiert.

Syntax

```
JSON_SIZE(super_expression)
```

Argumente

super_expression

Eine SUPER-Konstante oder ein Ausdruck.

Rückgabetyp

INTEGER

Die Funktion JSON_SIZE gibt eine INTEGER zurück, die die Anzahl der Bytes in der Eingabezeichenfolge anzeigt. Dieser Wert unterscheidet sich von der Anzahl der Zeichen. Zum Beispiel ist das UTF-8-Zeichen # (ein schwarzer Punkt) 3 Bytes groß, obwohl es nur 1 Zeichen ist.

Nutzungshinweise

JSON_SIZE(x) ist funktionell identisch mit OCTET_LENGTH(JSON_SERIALIZE). Beachten Sie jedoch, dass JSON_SERIALIZE einen Fehler zurückgibt, wenn der angegebene SUPER-Ausdruck bei der Serialisierung die VARCHAR-Grenze des Systems überschreiten würde. Bei JSON_SIZE gibt es diese Einschränkung nicht.

Beispiele

Verwenden Sie das folgende Beispiel, um die Länge eines SUPER-Werts zurückzugeben, der zu einer Zeichenfolge serialisiert wurde.

```
SELECT JSON_SIZE(JSON_PARSE('[10001,10002,"#"]'));

+-----------+
| json_size |
+-----------+
|        19 |
+-----------+
```

Beachten Sie, dass der angegebene SUPER-Ausdruck 17 Zeichen umfasst. # ist jedoch ein 3-Byte-Zeichen, sodass JSON_SIZE 19 zurückgibt.

Die Funktion JSON_TYPEOF

Die Skalarfunktion JSON_TYPEOF gibt einen VARCHAR mit den Werten „boolean“, „number“, „string“, „object“, „array“ oder „null“ zurück, abhängig vom dynamischen Typ des SUPER-Wertes.

Syntax

```
JSON_TYPEOF(super_expression)
```

Argumente

super_expression

Ein SUPER-Ausdruck oder eine Spalte.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um den JSON-Typ für das Array `[1,2]` zu überprüfen.

```
SELECT JSON_TYPEOF(ARRAY(1,2));

+-------------+
| json_typeof |
+-------------+
| array       |
+-------------+
```

Verwenden Sie das folgende Beispiel, um den JSON-Typ für das Objekt `{"name":"Joe"}` zu überprüfen.

```
SELECT JSON_TYPEOF(JSON_PARSE('{"name":"Joe"}'));

+-------------+
| json_typeof |
+-------------+
| object      |
+-------------+
```

SIZE

Gibt die binäre In-Memory-Größe einer Konstante oder eines Ausdrucks vom Typ SUPER als INTEGER zurück.

Syntax

```
SIZE(super_expression)
```

Argumente

super_expression

Eine Konstante oder ein Ausdruck vom Typ SUPER.

Rückgabetyp

INTEGER

Beispiele

Verwenden Sie das folgende Beispiel, um die In-Memory-Größe mehrerer Ausdrücke vom Typ SUPER mit SIZE abzurufen.

```
CREATE TABLE test_super_size(a SUPER);

INSERT INTO test_super_size
VALUES
  (null),
  (TRUE),
  (JSON_PARSE('[0,1,2,3]')),
  (JSON_PARSE('{"a":0,"b":1,"c":2,"d":3}'))
;

SELECT a, SIZE(a)
FROM test_super_size
ORDER BY 2, 1;

+---------------------------+------+
|             a             | size |
+---------------------------+------+
| true                      |    4 |
| NULL                      |    4 |
| [0,1,2,3]                 |   23 |
| {"a":0,"b":1,"c":2,"d":3} |   52 |
+---------------------------+------+
```

VARBYTE-Funktionen und -Operatoren

Zu den Amazon-Redshift-Funktionen und -Operatoren, die den VARBYTE-Datentyp unterstützen, gehören:

- VARBYTE-Operatoren
- FROM_HEX
- FROM_VARBYTE
- GETBIT
- TO_HEX
- TO_VARBYTE
- CONCAT
- LEN
- Die Funktion LENGTH
- OCTET_LENGTH
- Die Funktion SUBSTRING

VARBYTE-Operatoren

In der folgende Tabelle sind die VARBYTE-Operatoren aufgelistet. Der Operator arbeitet mit dem Binärwert des Datentyps VARBYTE. Wenn eine oder beide Eingaben null sind, ist das Ergebnis null.

Unterstützte Operatoren

Operator	Beschreibung	Rückgabetyp
<	kleiner als	BOOLEAN
<=	Kleiner als oder gleich	BOOLEAN
=	Gleich	BOOLEAN
>	größer als	BOOLEAN

Operator	Beschreibung	Rückgabetyp
>=	Größer als oder gleich	BOOLEAN
!= oder <>	Ungleich	BOOLEAN
||	Verkettung	VARBYTE
+	Verkettung	VARBYTE
~	Bitweises „nicht“	VARBYTE
&	Bitweises „und“	VARBYTE
|	Bitweises „oder“	VARBYTE
#	Bitweises „exklusives oder“	VARBYTE

Beispiele

In den folgenden Beispiel ist der Wert von `'a'::VARBYTE` 61 und der Wert von `'b'::VARBYTE` 62. Das `::` wandelt die Zeichenfolgen in den Datentyp VARBYTE um. Weitere Informationen zum Umwandeln von Datentypen finden Sie unter CAST.

Verwenden Sie das folgende Beispiel, um mit dem <-Operator zu vergleichen, ob `'a'` kleiner ist als `'b'`.

```
SELECT 'a'::VARBYTE < 'b'::VARBYTE AS less_than;

+-----------+
| less_than |
+-----------+
```

```
| true      |
+-----------+
```

Verwenden Sie das folgende Beispiel, um mit dem =-Operator zu vergleichen, ob `'a'` gleich `'b'` ist.

```
SELECT 'a'::VARBYTE = 'b'::VARBYTE AS equal;

+-------+
| equal |
+-------+
| false |
+-------+
```

Verwenden Sie das folgende Beispiel, um mit dem `||`-Operator zwei Binärwerte zu verketten.

```
SELECT 'a'::VARBYTE || 'b'::VARBYTE AS concat;

+--------+
| concat |
+--------+
|   6162 |
+--------+
```

Verwenden Sie das folgende Beispiel, um mit dem +-Operator zwei Binärwerte zu verketten.

```
SELECT 'a'::VARBYTE + 'b'::VARBYTE AS concat;

+--------+
| concat |
+--------+
|   6162 |
+--------+
```

Verwenden Sie das folgende Beispiel, um jedes Bit des eingegebenen Binärwerts mithilfe der Funktion FROM_VARBYTE zu negieren. Die Zeichenfolge `'a'` wird zu `01100001` ausgewertet. Weitere Informationen finden Sie unter FROM_VARBYTE.

```
SELECT FROM_VARBYTE(~'a'::VARBYTE, 'binary');

+--------------+
| from_varbyte |
```

```
+--------------+
|     10011110 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um den &-Operator auf die beiden Eingabebinärwerte anzuwenden. Die Zeichenfolge `'a'` wird zu `01100001` und `'b'` wird zu `01100010` ausgewertet.

```
SELECT FROM_VARBYTE('a'::VARBYTE & 'b'::VARBYTE, 'binary');

+--------------+
| from_varbyte |
+--------------+
|     01100000 |
+--------------+
```

Funktion FROM_HEX

FROM_HEX konvertiert einen Hexadezimalwert in einen Binärwert.

Syntax

```
FROM_HEX(hex_string)
```

Argumente

hex_string

Die hexadezimale Zeichenfolge des Datentyps `VARCHAR` oder `TEXT`, die konvertiert werden soll. Das Format muss ein Literalwert sein.

Rückgabetyp

`VARBYTE`

Beispiele

Verwenden Sie das folgende Beispiel, um die hexadezimale Darstellung von `'6162'` in einen Binärwert zu konvertieren. Das Ergebnis wird automatisch als hexadezimale Darstellung des Binärwerts angezeigt.

```
SELECT FROM_HEX('6162');
```

```
+----------+
| from_hex |
+----------+
|     6162 |
+----------+
```

Funktion FROM_VARBYTE

FROM_VARBYTE konvertiert einen Binärwert in eine Zeichenfolge im angegebenen Format.

Syntax

```
FROM_VARBYTE(binary_value, format)
```

Argumente

binary_value

Ein Binärwert des Datentyps VARBYTE.

format

Das Format der zurückgegebenen Zeichenfolge. Gültige Werte, bei denen die Groß-/Kleinschreibung nicht beachtet wird, sind `hex`, `binary`, `utf8` (auch `utf-8` und `utf_8`) und `base64`.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um den Binärwert `'ab'` in einen Hexadezimalwert zu konvertieren.

```
SELECT FROM_VARBYTE('ab', 'hex');

+--------------+
| from_varbyte |
+--------------+
|         6162 |
+--------------+
```

Verwenden Sie das folgende Beispiel, um die binäre Darstellung von `'4d'` zurückzugeben. Die binäre Darstellung von `'4d'` ist die Zeichenfolge `01001101`.

```
SELECT FROM_VARBYTE(FROM_HEX('4d'), 'binary');

+--------------+
| from_varbyte |
+--------------+
|     01001101 |
+--------------+
```

Funktion GETBIT

GETBIT gibt den Bitwert eines Binärwerts als den angegebenen Index zurück.

Syntax

```
GETBIT(binary_value, index)
```

Argumente

binary_value

Ein Binärwert des Datentyps VARBYTE.

index

Eine Indexnummer des Bits im zurückgegebenen Binärwert. Der Binärwert ist ein 0-basiertes Bit-Array, das vom am weitesten rechts stehenden Bit (unwichtigstes Bit) bis zum am weitesten links stehenden Bit (wichtigstes Bit) indiziert wird.

Rückgabetyp

`INTEGER`

Beispiele

Verwenden Sie das folgende Beispiel, um ein Bit bei Index 2 des Binärwerts `from_hex('4d')` zurückzugeben. Die binäre Darstellung von `'4d'` ist `01001101`.

```
SELECT GETBIT(FROM_HEX('4d'), 2);
```

```
+--------+
| getbit |
+--------+
|      1 |
+--------+
```

Verwenden Sie das folgende Beispiel, um das Bit an acht Indexstellen des von `from_hex('4d')` zurückgegebenen Binärwerts zurückzugeben. Die binäre Darstellung von `'4d'` ist `01001101`.

```
SELECT GETBIT(FROM_HEX('4d'), 7), GETBIT(FROM_HEX('4d'), 6),
  GETBIT(FROM_HEX('4d'), 5), GETBIT(FROM_HEX('4d'), 4),
  GETBIT(FROM_HEX('4d'), 3), GETBIT(FROM_HEX('4d'), 2),
  GETBIT(FROM_HEX('4d'), 1), GETBIT(FROM_HEX('4d'), 0);

+--------+--------+--------+--------+--------+--------+--------+--------+
| getbit | getbit | getbit | getbit | getbit | getbit | getbit | getbit |
+--------+--------+--------+--------+--------+--------+--------+--------+
|      0 |      1 |      0 |      0 |      1 |      1 |      0 |      1 |
+--------+--------+--------+--------+--------+--------+--------+--------+
```

Funktion TO_HEX

TO_HEX konvertiert eine Zahl oder einen Binärwert in eine hexadezimale Darstellung.

Syntax

```
TO_HEX(value)
```

Argumente

Wert

Entweder eine Zahl oder ein Binärwert (VARBYTE), die/der konvertiert werden soll.

Rückgabetyp

VARCHAR

Beispiele

Verwenden Sie das folgende Beispiel, um eine Zahl in ihre hexadezimale Darstellung zu konvertieren.

```
SELECT TO_HEX(2147676847);

+----------+
|  to_hex  |
+----------+
| 8002f2af |
+----------+
```

Verwenden Sie das folgende Beispiel, um die VARBYTE-Darstellung von `'abc'` in eine hexadezimale Zahl zu konvertieren.

```
SELECT TO_HEX('abc'::VARBYTE);

+--------+
| to_hex |
+--------+
| 616263 |
+--------+
```

Verwenden Sie das folgende Beispiel, um eine Tabelle zu erstellen, die VARBYTE-Darstellung von `'abc'` in eine Hexadezimalzahl einzufügen und anschließend die Spalte mit dem Wert auszuwählen.

```
CREATE TABLE t (vc VARCHAR);
INSERT INTO t SELECT TO_HEX('abc'::VARBYTE);
SELECT vc FROM t;

+--------+
|   vc   |
+--------+
| 616263 |
+--------+
```

Verwenden Sie das folgende Beispiel, um zu zeigen, dass beim Umwandeln eines VARBYTE-Werts in VARCHAR das Format UTF-8 ist.

```
CREATE TABLE t (vc VARCHAR);
INSERT INTO t SELECT 'abc'::VARBYTE::VARCHAR;

SELECT vc FROM t;

+-----+
```

```
| vc  |
+-----+
| abc |
+-----+
```

Funktion TO_VARBYTE

TO_VARBYTE konvertiert eine Zeichenfolge in einem angegebenen Format in einen Binärwert.

Syntax

```
TO_VARBYTE(string, format)
```

Argumente

string

Eine CHAR- oder VARCHAR-Zeichenfolge.

format

Das Format der Eingabezeichenfolge. Gültige Werte, bei denen die Groß-/Kleinschreibung nicht beachtet wird, sind `hex`, `binary`, `utf8` (auch `utf-8` und `utf_8`) und `base64`.

Rückgabetyp

VARBYTE

Beispiele

Verwenden Sie das folgende Beispiel, um den Hexadezimalwert 6162 in einen Binärwert zu konvertieren. Das Ergebnis wird automatisch als hexadezimale Darstellung des Binärwerts angezeigt.

```
SELECT TO_VARBYTE('6162', 'hex');

+------------+
| to_varbyte |
+------------+
|       6162 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die binäre Darstellung von 4d zurückzugeben. Die binäre Darstellung von „4d“ ist 01001101.

```
SELECT TO_VARBYTE('01001101', 'binary');

+------------+
| to_varbyte |
+------------+
|         4d |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Zeichenfolge 'a' in UTF-8 in einen Binärwert zu konvertieren. Das Ergebnis wird automatisch als hexadezimale Darstellung des Binärwerts angezeigt.

```
SELECT TO_VARBYTE('a', 'utf8');

+------------+
| to_varbyte |
+------------+
|         61 |
+------------+
```

Verwenden Sie das folgende Beispiel, um die Zeichenfolge '4' in einen Hexadezimalwert zu konvertieren. Wenn es sich bei der Länge der hexadezimalen Zeichenfolge um eine ungerade Zahl handelt, wird 0 vorangestellt, um eine gültige hexadezimale Zahl zu erhalten.

```
SELECT TO_VARBYTE('4', 'hex');

+------------+
| to_varbyte |
+------------+
|         04 |
+------------+
```

Fensterfunktionen

Mit Fensterfunktionen können Sie analytische geschäftliche Abfragen effizienter erstellen. Fensterfunktionen werden für eine Partition bzw. ein „Fenster“ eines Ergebnissatzes ausgeführt und geben für jede Zeile in diesem Fenster einen Wert zurück. Funktionen ohne Fenster führen ihre

Berechnungen dagegen für alle Zeilen des Ergebnissatzes aus. Im Gegensatz zu Gruppenfunktionen, die die Ergebniszeilen aggregieren, behalten Fensterfunktionen alle Zeilen im Tabellenausdruck bei.

Die zurückgegebenen Werte werden mithilfe von Werten aus den Sätzen von Zeilen in diesem Fenster berechnet. Das Fenster definiert für jede Zeile in der Tabelle einen Satz von Zeilen, der für die Verarbeitung zusätzlicher Attribute verwendet wird. Ein Fenster wird mithilfe einer Fensterspezifikation (der OVER-Klausel) definiert und basiert auf drei Hauptkonzepten:

- Fensterpartitionierung, die Gruppen von Zeilen bildet (PARTITION-Klausel)
- Fensteranordnung, die eine Reihenfolge oder Sequenz von Zeilen innerhalb der einzelnen Partitionen definiert (ORDER BY-Klausel)
- Fensterrahmen, die in Bezug auf die einzelnen Zeilen definiert werden, um den Satz von Zeilen weiter einzuschränken (ROWS-Spezifikation)

Fensterfunktionen sind der letzte Satz von Operationen, die in einer Abfrage ausgeführt werden, abgesehen von der abschließenden ORDER BY-Klausel. Alle Joins und alle -, - und - Klauseln werden abgeschlossen, bevor die Fensterfunktionen verarbeitet werden. Daher können Fensterfunktionen nur in der Auswahlliste oder in der ORDER BY-Klausen enthalten sein. Innerhalb einer einzelnen Abfrage können mehrere Fensterfunktionen mit unterschiedlichen Rahmenklauseln verwendet werden. Außerdem können Sie Fensterfunktionen in anderen skalaren Ausdrücken verwenden, beispielsweise CASE.

Übersicht über die Syntax von Fensterfunktionen

Fensterfunktionen folgen einer Standardsyntax, die wie folgt lautet.

```
function (expression) OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list [ frame_clause ] ] )
```

Hier ist function eine der in diesem Abschnitt beschriebenen Funktionen.

Die expr_list lautet wie folgt.

```
expression | column_name [, expr_list ]
```

Die order_list lautet wie folgt.

```
expression | column_name [ ASC | DESC ]
[ NULLS FIRST | NULLS LAST ]
[, order_list ]
```

Die frame_clause lautet wie folgt.

```
ROWS
{ UNBOUNDED PRECEDING | unsigned_value PRECEDING | CURRENT ROW } |

{ BETWEEN
{ UNBOUNDED PRECEDING | unsigned_value { PRECEDING | FOLLOWING } | CURRENT ROW}
AND
{ UNBOUNDED FOLLOWING | unsigned_value { PRECEDING | FOLLOWING } | CURRENT ROW }}
```

Argumente

Funktion

Die Fensterfunktion. Details finden Sie in den Beschreibungen der einzelnen Funktionen.

OVER

Die Klausel, die die Fensterspezifikation definiert. Die OVER-Klausel ist für Fensterfunktionen obligatorisch und differenziert Fensterfunktionen von anderen SQL-Funktionen.

PARTITION BY expr_list

(Optional) Die PARTITION-BY-Klausel unterteilt den Ergebnissatz in Partitionen, ähnlich wie die GROUP-BY-Klausel. Wenn eine Partitionsklausel vorhanden ist, wird die Funktion für die Zeilen in den einzelnen Partitionen berechnet. Wenn keine Partitionsklausel angegeben ist, enthält eine einzige Partition die gesamte Tabelle und die Funktion wird für die gesamte Tabelle berechnet.

Die Rangfestlegungsfunktionen DENSE_RANK, NTILE, RANK und ROW_NUMBER erfordern einen globalen Vergleich aller Zeilen im Ergebnissatz. Wenn eine PARTITION BY-Klausel verwendet wird, kann die Abfrageoptimierung die einzelnen Aggregationen parallel ausführen, indem der Workload entsprechend den Partitionen über mehrere Slices verteilt wird. Wenn die PARTITION BY-Klausel nicht vorhanden ist, muss der Aggregationsschritt seriell für einen einzelnen Slice ausgeführt werden. Dies kann erhebliche negative Auswirkungen auf die Leistung haben, besonders für größere Cluster.

Amazon Redshift unterstützt keine Zeichenfolgeliterale in PARTITION-BY-Klauseln.

ORDER BY order_list

(Optional) Die Fensterfunktion wird auf die Zeilen innerhalb der einzelnen Partitionen angewendet, sortiert entsprechend der Reihenfolgenspezifikation in ORDER BY. Diese ORDER BY-Klausel unterscheidet sich von der ORDER BY-Klausel in der frame_clauseund ist mit dieser in keiner Weise verwandt. Die ORDER BY-Klausel kann ohne die PARTITION BY-Klausel verwendet werden.

Für Rangfestlegungsfunktionen identifiziert die ORDER BY-Klausel die Messwerte für die Rangfestlegungswerte. Für Aggregationsfunktionen müssen die partitionierten Zeilen angeordnet werden, bevor die jeweilige Aggregationsfunktion für die einzelnen Rahmen berechnet wird. Weitere Informationen zu den Arten von Windowsfunktionen finden Sie unter Fensterfunktionen.

In der Reihenfolgenliste werden Spaltenbezeichner oder Ausdrücke, die zu Spaltenbezeichnern ausgewertet werden, benötigt. Konstanten oder Konstantenausdrücke können nicht als Ersatz für Spaltennamen verwendet werden.

NULL-Werte werden als eigene Gruppe behandelt und entsprechend der Option NULLS FIRST oder NULLS LAST sortiert und angeordnet. Standardmäßig werden NULL-Werte in einer ASC-Reihenfolge an letzter Stelle sortiert und aufgeführt und in einer DESC-Reihenfolge an erster Stelle sortiert und aufgeführt.

Amazon Redshift unterstützt keine Zeichenfolgeliterale in ORDER-BY-Klauseln.

Wenn die ORDER BY-Klausel ausgelassen wird, ist die Reihenfolge der Zeilen nicht deterministisch.

> **Note**
>
> In einem parallelen System wie Amazon Redshift, in dem eine ORDER-BY-Klausel keine spezifische und globale Anordnung der Daten generiert, ist die Reihenfolge der Zeilen nicht deterministisch. Wenn der ORDER-BY-Ausdruck duplizierte Werte produziert (eine partielle Anordnung), kann sich die Rückgabereihenfolge dieser Zeilen zwischen Ausführungen von Amazon Redshift unterscheiden. In diesem Fall können Fensterfunktionen unerwartete oder inkonsistente Ergebnisse zurückgeben. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

column_name

Der Name einer Spalte, nach der die Partitionierung oder Anordnung erfolgen soll.

ASC | DESC

Eine Option, die die Sortierreihenfolge für den Ausdruck wie folgt definiert:

- ASC: aufsteigend (beispielsweise niedrig nach hoch für numerische Werte und A bis Z für Zeichenfolgen). Wenn keine Option angegeben wird, werden die Daten standardmäßig in aufsteigender Reihenfolge sortiert.
- DESC: absteigend (beispielsweise hoch nach niedrig für numerische Werte und Z bis A für Zeichenfolgen).

NULLS FIRST | NULLS LAST

Option, die angibt, ob NULL-Werte an erster Stelle vor Nicht-Null-Werten oder an letzter Stelle nach Nicht-Null-Werten aufgelistet werden sollen. Standardmäßig werden NULL-Werte in einer ASC-Reihenfolge an letzter Stelle sortiert und aufgeführt und in einer DESC-Reihenfolge an erster Stelle sortiert und aufgeführt.

frame_clause

Die Rahmenklausel gibt für Aggregationsfunktionen den Satz von Zeilen im Fenster einer Funktion bei Verwendung von ORDER BY noch genauer an. Sie ermöglicht das Ein- oder Ausschließen von Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren.

Die Rahmenklausel kann nicht auf Rangfestlegungsfunktionen angewendet werden. Außerdem ist sie nicht erforderlich, wenn in der ORDER-BY-Klausel für eine Aggregationsfunktion keine OVER-Klausel verwendet wird. Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich.

Wenn keine ORDER-BY-Klausel angegeben ist, ist der implizierte Rahmen unbegrenzt, äquivalent zu ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING.

ROWS

Diese Klausel definiert den Fensterrahmen durch Angabe eines physischen Offsets von der aktuellen Zeile.

Diese Klausel gibt die Zeilen im aktuellen Fenster oder in der aktuellen Partition an, mit denen der Wert in der aktuellen Zeile kombiniert werden soll. Sie verwendet Argumente, die die Zeilenposition angeben. Diese kann sich vor oder nach der aktuellen Zeile befinden. Der

Referenzpunkt für alle Fensterrahmen ist die aktuelle Zeile. Alle Zeilen werden nacheinander zur aktuellen Zeile, während der Fensterrahmen in der Partition vorwärts gleitet.

Beim Rahmen kann es sich um einen einfachen Satz von Zeilen bis zur und einschließlich der aktuellen Zeile handeln.

```
{UNBOUNDED PRECEDING | offset PRECEDING | CURRENT ROW}
```

Es kann sich auch um einen Satz von Zeilen zwischen zwei Grenzen handeln.

```
BETWEEN
{ UNBOUNDED PRECEDING | offset { PRECEDING | FOLLOWING } | CURRENT ROW }
AND
{ UNBOUNDED FOLLOWING | offset { PRECEDING | FOLLOWING } | CURRENT ROW }
```

UNBOUNDED PRECEDING zeigt an, dass das Fenster an der ersten Zeile der Partition beginnt; offset PRECEDING zeigt an, dass das Fenster um eine Zahl von Reihen vor der aktuellen Zeile beginnt, die dem Offset-Wert entspricht. UNBOUNDED PRECEDING ist der Standardwert.

CURRENT ROW zeigt an, dass das Fenster an der aktuellen Zeile beginnt oder endet.

UNBOUNDED FOLLOWING zeigt an, dass das Fenster an der letzten Zeile der Partition endet; offset FOLLOWING zeigt an, dass das Fenster um eine Zahl von Reihen nach der aktuellen Zeile endet, die dem Offset-Wert entspricht.

offset bezeichnet eine physische Anzahl von Zeilen vor oder nach der aktuellen Zeile. In diesem Fall muss offset eine Konstante sein, der zu einem positiven numerischen Wert ausgewertet wird. Beispielsweise wird bei 5 FOLLOWING der Rahmen fünf Zeilen nach der aktuellen Zeile beendet.

Wenn BETWEEN nicht angegeben ist, wird der Rahmen implizit von der aktuellen Zeile begrenzt. Beispielsweise ist `ROWS 5 PRECEDING` gleich `ROWS BETWEEN 5 PRECEDING AND CURRENT ROW`. Ebenso ist `ROWS UNBOUNDED FOLLOWING` gleich `ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING`.

> **Note**
>
> Sie können keinen Rahmen angeben, in dem die Startgrenze größer als die Endgrenze ist. Sie können beispielsweise keinen der folgenden Rahmen angeben.
>
> ```
> between 5 following and 5 preceding
> between current row and 2 preceding
> ```

```
between 3 following and current row
```

Spezifisches Anordnen von Daten für Fensterfunktionen

Wenn eine ORDER-BY-Klausel für eine Fensterfunktion keine spezifische und globale Anordnung der Daten generiert, ist die Reihenfolge der Zeilen nicht deterministisch. Wenn der ORDER-BY-Ausdruck duplizierte Werte generiert (eine partielle Anordnung), kann sich die Rückgabereihenfolge dieser Zeilen zwischen verschiedenen Ausführungen unterscheiden. In diesem Fall geben Fensterfunktionen möglicherweise unerwartete oder inkonsistente Ergebnisse zurück.

Beispielsweise gibt die folgende Abfrage in verschiedenen Ausführen unterschiedliche Ergebnisse zurück. Diese unterschiedlichen Ergebnisse treten auf, da `order by dateid` keine spezifische Reihenfolge der Daten für die SUM-Fensterfunktion erzeugt.

```
select dateid, pricepaid,
sum(pricepaid) over(order by dateid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid |   sumpaid
--------+-----------+-------------
1827 |   1730.00 |     1730.00
1827 |    708.00 |     2438.00
1827 |    234.00 |     2672.00
...

select dateid, pricepaid,
sum(pricepaid) over(order by dateid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid |   sumpaid
--------+-----------+-------------
1827 |    234.00 |      234.00
1827 |    472.00 |      706.00
1827 |    347.00 |     1053.00
...
```

In diesem Fall kann das Hinzufügen einer zweiten ORDER-BY-Spalte zur Fensterfunktion das Problem lösen.

```
select dateid, pricepaid,
sum(pricepaid) over(order by dateid, pricepaid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid | sumpaid
--------+-----------+---------
1827 |    234.00 |  234.00
1827 |    337.00 |  571.00
1827 |    347.00 |  918.00
...
```

Unterstützte Funktionen

Amazon Redshift unterstützt zwei Arten von Fensterfunktionen: Aggregation und Rangfestlegung.

Die folgenden Aggregationsfunktionen werden unterstützt:

- Die Fensterfunktion AVG
- Die Fensterfunktion COUNT
- CUME_DIST-Fensterfunktion
- Die Fensterfunktion DENSE_RANK
- Die Fensterfunktion FIRST_VALUE
- Die Fensterfunktion LAG
- Die Fensterfunktion LAST_VALUE
- Die Fensterfunktion LEAD
- Die Fensterfunktion LISTAGG
- Die Fensterfunktion MAX
- Die Fensterfunktion MEDIAN
- Die Fensterfunktion MIN
- Die Fensterfunktion NTH_VALUE
- Fensterfunktion PERCENTILE_CONT
- Die Fensterfunktion PERCENTILE_DISC
- Die Fensterfunktion RATIO_TO_REPORT
- Die Fensterfunktionen STDDEV_SAMP und STDDEV_POP (STDDEV_SAMP und STDDEV sind Synonyme)

- Die Fensterfunktion SUM
- Die Fensterfunktionen VAR_SAMP und VAR_POP (VAR_SAMP und VARIANCE sind Synonyme)

Die folgenden Rangfestlegungsfunktionen werden unterstützt:

- Die Fensterfunktion DENSE_RANK
- Die Fensterfunktion NTILE
- Die Fensterfunktion PERCENT_RANK
- Die Fensterfunktion RANK
- Die Fensterfunktion ROW_NUMBER

Beispieltabelle mit Beispielen von Fensterfunktionen

Zu jeder Funktionsbeschreibung gehören spezifische Fensterfunktionsbeispiele. In einigen der Beispiele wird eine Tabelle namens WINSALES verwendet, die 11 Zeilen enthält. Dies sieht wie folgt aus.

SALESID	DATEID	SELLERID	BUYERID	QTY	QTY_SHIPPED
30001	8/2/2003	3	B	10	10
10001	12/24/2003	1	C	10	10
10005	12/24/2003	1	A	30	
40001	1/9/2004	4	A	40	
10006	1/18/2004	1	C	10	
20001	2/12/2004	2	B	20	20
40005	2/12/2004	4	A	10	10
20002	2/16/2004	2	C	20	20
30003	4/18/2004	3	B	15	

SALESID	DATEID	SELLERID	BUYERID	QTY	QTY_SHIPPED
30004	4/18/2004	3	B	20	
30007	9/7/2004	3	C	30	

Im folgenden Skript wird die Beispieltabelle WINSALES erstellt und ausgefüllt.

```
CREATE TABLE winsales(
  salesid int,
  dateid date,
  sellerid int,
  buyerid char(10),
  qty int,
  qty_shipped int);

INSERT INTO winsales VALUES
  (30001, '8/2/2003', 3, 'b', 10, 10),
  (10001, '12/24/2003', 1, 'c', 10, 10),
  (10005, '12/24/2003', 1, 'a', 30, null),
  (40001, '1/9/2004', 4, 'a', 40, null),
  (10006, '1/18/2004', 1, 'c', 10, null),
  (20001, '2/12/2004', 2, 'b', 20, 20),
  (40005, '2/12/2004', 4, 'a', 10, 10),
  (20002, '2/16/2004', 2, 'c', 20, 20),
  (30003, '4/18/2004', 3, 'b', 15, null),
  (30004, '4/18/2004', 3, 'b', 20, null),
  (30007, '9/7/2004', 3, 'c', 30, null);
```

Die Fensterfunktion AVG

Die AVG-Fensterfunktion gibt den Durchschnitt (das arithmetische Mittel) der Eingabeausdruckwerte zurück. Die Funktion AVG ist mit numerischen Werten kompatibel und ignoriert NULL-Werte.

Syntax

```
AVG ( [ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
```

```
                        frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, um die Zählung auszuführen. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Gibt die Fensterklauseln für die Aggregationsfunktionen an. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die AVG-Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Die von der Funktion AVG unterstützten Argumenttypen sind SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL und DOUBLE PRECISION.

Die von der Funktion AVG unterstützten Rückgabetypen sind:

- BIGINT für SMALLINT- oder INTEGER-Argumente
- NUMERIC für BIGINT-Argumente
- DOUBLE PRECISION für Gleitkomma-Argumente

Beispiele

Im folgenden Beispiel wird ein gleitender Durchschnitt von verkauften Mengen nach Datum berechnet und die Ergebnisse nach Datums- und Verkaufs-ID geordnet:

```
select salesid, dateid, sellerid, qty,
avg(qty) over
(order by dateid, salesid rows unbounded preceding) as avg
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | avg
---------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  10
10005 | 2003-12-24 |        1 |  30 |  16
40001 | 2004-01-09 |        4 |  40 |  22
10006 | 2004-01-18 |        1 |  10 |  20
20001 | 2004-02-12 |        2 |  20 |  20
40005 | 2004-02-12 |        4 |  10 |  18
20002 | 2004-02-16 |        2 |  20 |  18
30003 | 2004-04-18 |        3 |  15 |  18
30004 | 2004-04-18 |        3 |  20 |  18
30007 | 2004-09-07 |        3 |  30 |  19
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Die Fensterfunktion COUNT

Die Fensterfunktion COUNT zählt die durch den Ausdruck definierten Zeilen.

Die Funktion COUNT hat zwei Varianten. COUNT(*) zählt alle Zeilen in der Zieltabelle, unabhängig davon, ob sie Null-Werte enthalten oder nicht. COUNT(expression) berechnet die Zahl der Zeilen mit Nicht-NULL-Werten in einer spezifischen Spalte oder einem spezifischen Ausdruck.

Syntax

```
COUNT ( * | [ ALL ] expression) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem angegebenen Ausdruck, um die Zählung auszuführen. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Gibt die Fensterklauseln für die Aggregationsfunktionen an. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die COUNT-Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Die Funktion COUNT unterstützt alle Argumentdatentypen.

Der von der Funktion COUNT unterstützte Rückgabetyp ist BIGINT.

Beispiele

Das folgende Beispiel zeigt die Verkaufs-ID, die Menge und die Zahl aller Zeilen ab dem Beginn des Datenfensters:

```
select salesid, qty,
count(*) over (order by salesid rows unbounded preceding) as count
from winsales
order by salesid;

salesid | qty | count
---------+-----+-----
10001 |  10 |   1
10005 |  30 |   2
10006 |  10 |   3
20001 |  20 |   4
20002 |  20 |   5
30001 |  10 |   6
30003 |  15 |   7
30004 |  20 |   8
30007 |  30 |   9
40001 |  40 |   10
40005 |  10 |   11
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Das folgende Beispiel zeigt die Verkaufs-ID, die Menge und die Zahl der Nicht-Null-Zeilen ab dem Beginn des Datenfensters an. (In der Tabelle WINSALES enthält die Spalte QTY_SHIPPED einige NULL-Werte.)

```
select salesid, qty, qty_shipped,
count(qty_shipped)
over (order by salesid rows unbounded preceding) as count
from winsales
```

```
order by salesid;

salesid | qty | qty_shipped | count
---------+-----+-------------+-------
10001 |  10 |          10 |   1
10005 |  30 |             |   1
10006 |  10 |             |   1
20001 |  20 |          20 |   2
20002 |  20 |          20 |   3
30001 |  10 |          10 |   4
30003 |  15 |             |   4
30004 |  20 |             |   4
30007 |  30 |             |   4
40001 |  40 |             |   4
40005 |  10 |          10 |   5
(11 rows)
```

CUME_DIST-Fensterfunktion

Berechnet die kumulative Verteilung eines Werts in einem Fenster oder einer Partition. Bei aufsteigender Anordnung wird die kumulative Verteilung anhand der folgenden Formel festgelegt:

`count of rows with values <= x / count of rows in the window or partition`

wobei x gleich dem Wert in der aktuellen Zeile der Spalte ist, die in der ORDER BY-Klausel angegeben wird. Der folgende Datensatz zeigt die Verwendung dieser Formel:

```
Row# Value   Calculation    CUME_DIST
1        2500    (1)/(5)    0.2
2        2600    (2)/(5)    0.4
3        2800    (3)/(5)    0.6
4        2900    (4)/(5)    0.8
5        3100    (5)/(5)    1.0
```

Der Rückgabewertbereich ist >0 bis 1 (einschließlich).

Syntax

```
CUME_DIST ()
OVER (
[ PARTITION BY partition_expression ]
[ ORDER BY order_list ]
```

```
)
```

Argumente

OVER

Eine Klausel, die die Fensterpartitionierung angibt. Die OVER-Klausel darf keine Fensterrahmenspezifikation enthalten.

PARTITION BY partition_expression

Optional. Ein Ausdruck, der den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

ORDER BY order_list

Der Ausdruck, anhand dessen die kumulative Verteilung berechnet wird. Der Datentyp des Ausdrucks muss entweder numerisch sein oder implizit in einen solchen konvertierbar sein. Wenn ORDER BY ausgelassen wird, ist der Rückgabewert für alle Zeilen 1.

Wenn ORDER-BY nicht zu einer spezifischen Reihenfolge führt, ist die Reihenfolge der Zeilen nicht deterministisch. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

Rückgabetyp

FLOAT8

Beispiele

Im folgenden Beispiel wird die kumulative Verteilung der Menge für die einzelnen Verkäufer berechnet:

```
select sellerid, qty, cume_dist()
over (partition by sellerid order by qty)
from winsales;

sellerid   qty    cume_dist
-------------------------------------------------
1         10.00    0.33
1         10.64    0.67
1         30.37    1
```

```
3          10.04     0.25
3          15.15     0.5
3          20.75     0.75
3          30.55     1
2          20.09     0.5
2          20.12     1
4          10.12     0.5
4          40.23     1
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Die Fensterfunktion DENSE_RANK

Die Fensterfunktion DENSE_RANK legt den Rang eines Werts in einer Gruppe von Werten fest, basierend auf dem ORDER BY-Ausdruck in der OVER-Klausel. Wenn die optionale PARTITION BY-Klausel vorhanden ist, wird die Rangfolge für jede Gruppe von Zeilen neu festgelegt. Zeilen mit gleichen Werten in Bezug auf die Rangfestlegungskriterien erhalten den gleichen Rang. Die Funktion DENSE_RANK unterscheidet sich nur in einer Hinsicht von RANK: Wenn zwei oder mehr Zeilen den gleichen Rang erhalten, entsteht in der Rangfolge der Werte keine Lücke. Wenn beispielsweise zwei Zeilen den Rang 1 erhalten, ist der nächste Rang 2.

Sie können in derselben Abfrage Rangfestlegungsfunktionen mit unterschiedlichen PARTITION BY- und ORDER BY-Klauseln verwenden.

Syntax

```
DENSE_RANK() OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list ]
)
```

Argumente

( )

Die Funktion verwendet keine Argumente. Es ist jedoch eine leere Klammer erforderlich.

OVER

Die Fensterklauseln für die Funktion DENSE_RANK.

PARTITION BY expr_list

(Optional) Ein oder mehrere Ausdrücke, der/die das Fenster definiert/definieren.

ORDER BY order_list

(Optional) Der Ausdruck, auf dem die Rangfestlegungwerte basieren. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle. Wenn ORDER BY ausgelassen wird, ist der Rückgabewert für alle Zeilen 1.

Wenn ORDER-BY nicht zu einer spezifischen Reihenfolge führt, ist die Reihenfolge der Zeilen nicht deterministisch. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

Rückgabetyp

`INTEGER`

Beispiele

Die folgenden Beispiele verwenden die Beispieltabelle für Fensterfunktionen. Weitere Informationen finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird die Tabelle nach der verkauften Menge geordnet und jeder Zeile ein DENSE_RANK-Wert und ein regulärer Rang zugewiesen. Die Ergebnisse werden sortiert, nachdem die Fensterfunktionsergebnisse angewendet wurden.

```
SELECT salesid, qty,
DENSE_RANK() OVER(ORDER BY qty DESC) AS d_rnk,
RANK() OVER(ORDER BY qty DESC) AS rnk
FROM winsales
ORDER BY 2,1;

+---------+-----+-------+-----+
| salesid | qty | d_rnk | rnk |
+---------+-----+-------+-----+
|   10001 |  10 |     5 |   8 |
|   10006 |  10 |     5 |   8 |
|   30001 |  10 |     5 |   8 |
|   40005 |  10 |     5 |   8 |
|   30003 |  15 |     4 |   7 |
|   20001 |  20 |     3 |   4 |
```

```
|   20002 |  20 |     3 |   4 |
|   30004 |  20 |     3 |   4 |
|   10005 |  30 |     2 |   2 |
|   30007 |  30 |     2 |   2 |
|   40001 |  40 |     1 |   1 |
+---------+-----+-------+-----+
```

Beachten Sie den Unterschied bei den Rängen, die demselben Satz von Zeilen zugewiesen werden, wenn die Funktionen DENSE_RANK und RANK zusammen in derselben Umfrage verwendet werden.

Im folgenden Beispiel wird die Tabelle nach sellerid partitioniert, die einzelnen Partitionen nach der Menge geordnet und jeder Zeile ein Dichtewert zugewiesen. Die Ergebnisse werden sortiert, nachdem die Fensterfunktionsergebnisse angewendet wurden.

```
SELECT salesid, sellerid, qty,
DENSE_RANK() OVER(PARTITION BY sellerid ORDER BY qty DESC) AS d_rnk
FROM winsales
ORDER BY 2,3,1;

+---------+----------+-----+-------+
| salesid | sellerid | qty | d_rnk |
+---------+----------+-----+-------+
|   10001 |        1 |  10 |     2 |
|   10006 |        1 |  10 |     2 |
|   10005 |        1 |  30 |     1 |
|   20001 |        2 |  20 |     1 |
|   20002 |        2 |  20 |     1 |
|   30001 |        3 |  10 |     4 |
|   30003 |        3 |  15 |     3 |
|   30004 |        3 |  20 |     2 |
|   30007 |        3 |  30 |     1 |
|   40005 |        4 |  10 |     2 |
|   40001 |        4 |  40 |     1 |
+---------+----------+-----+-------+
```

Wenn Sie das letzte Beispiel erfolgreich verwenden möchten, fügen Sie eine Zeile mit dem folgenden Befehl in die Tabelle WINSALES ein. Diese Zeile hat dieselbe buyerid, sellerid und qtysold wie eine andere Zeile. Dadurch werden im letzten Beispiel zwei Zeilen miteinander verknüpft, was den Unterschied zwischen den Funktionen DENSE_RANK und RANK verdeutlicht.

```
INSERT INTO winsales VALUES(30009, '2/2/2003', 3, 'b', 20, NULL);
```

Im folgenden Beispiel wird die Tabelle nach buyerid und sellerid partitioniert, die einzelnen Partitionen nach der Menge geordnet und jeder Zeile wird ein Dichtewert und ein regulärer Rang zugewiesen. Die Ergebnisse werden sortiert, nachdem die Fensterfunktion angewendet wurde.

```
SELECT salesid, sellerid, qty, buyerid,
DENSE_RANK() OVER(PARTITION BY buyerid, sellerid ORDER BY qty DESC) AS d_rnk,
RANK() OVER (PARTITION BY buyerid, sellerid ORDER BY qty DESC) AS rnk
FROM winsales
ORDER BY rnk;

+---------+----------+-----+---------+-------+-----+
| salesid | sellerid | qty | buyerid | d_rnk | rnk |
+---------+----------+-----+---------+-------+-----+
|   20001 |        2 |  20 | b       |     1 |   1 |
|   30007 |        3 |  30 | c       |     1 |   1 |
|   10006 |        1 |  10 | c       |     1 |   1 |
|   10005 |        1 |  30 | a       |     1 |   1 |
|   20002 |        2 |  20 | c       |     1 |   1 |
|   30009 |        3 |  20 | b       |     1 |   1 |
|   40001 |        4 |  40 | a       |     1 |   1 |
|   30004 |        3 |  20 | b       |     1 |   1 |
|   10001 |        1 |  10 | c       |     1 |   1 |
|   40005 |        4 |  10 | a       |     2 |   2 |
|   30003 |        3 |  15 | b       |     2 |   3 |
|   30001 |        3 |  10 | b       |     3 |   4 |
+---------+----------+-----+---------+-------+-----+
```

Die Fensterfunktion FIRST_VALUE

Bei einem geordneten Satz von Zeilen gibt FIRST_VALUE den Wert des angegebenen Ausdrucks in Bezug auf die erste Zeile im Fensterrahmen zurück.

Informationen zur Auswahl der letzten Zeile im Rahmen finden Sie unter Die Fensterfunktion LAST_VALUE.

Syntax

```
FIRST_VALUE( expression )[ IGNORE NULLS | RESPECT NULLS ]
OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Argumente

expression

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

IGNORE NULLS

Bei Verwendung dieser Option für FIRST_VALUE gibt die Funktion den ersten Wert im Rahmen zurück, der nicht NULL ist (oder NULL, wenn alle Werte NULL sind).

RESPECT NULLS

Gibt an, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte berücksichtigen soll. Wenn Sie IGNORE NULLS nicht angeben, wird RESPECT NULLS standardmäßig unterstützt.

OVER

Führt die Fensterklauseln für die Funktion ein.

PARTITION BY expr_list

Definiert das Fenster für die Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn die PARTITION BY-Klausel nicht angegeben ist, sortiert ORDER BY die gesamte Tabelle. Wenn Sie eine ORDER BY-Klausel angeben, müssen Sie auch eine frame_clause angeben.

Die Ergebnisse der Funktion FIRST_VALUE sind von der Anordnung der Daten abhängig. Die Ergebnisse sind in den folgenden Fällen nicht deterministisch:

- Wenn keine ORDER BY-Klausel angegeben ist und eine Partition zwei verschiedene Werte für einen Ausdruck enthält
- Wenn der Ausdruck zu verschiedenen Werten ausgewertet wird, die demselben Wert in der ORDER BY-Liste entsprechen

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen im geordneten

Ergebnis. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Rückgabetyp

Diese Funktionen unterstützen Ausdrücke, die einfache Amazon-Redshift-Datentypen verwenden. Der Rückgabetyp ist mit dem Datentyp von expression identisch.

Beispiele

In den folgenden Beispielen wird die Tabelle VENUE aus den TICKIT-Beispieldaten verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Im folgenden Beispiel wird die Sitzplatzkapazität für die einzelnen Veranstaltungsorte in der Tabelle VENUE zurückgegeben, wobei die Ergebnisse nach Kapazität (hoch zu niedrig) geordnet sind. Die Funktion FIRST_VALUE wird verwendet, um den Namen des Veranstaltungsorts auszuwählen, der der ersten Zeile im Rahmen entspricht, in diesem Fall der Zeile mit der größten Zahl von Sitzplätzen. Die Ergebnisse werden nach Bundesstaat partitioniert. Wenn der Wert für VENUESTATE geändert wird, wird daher ein neuer erster Wert ausgewählt. Der Fensterrahmen ist unbegrenzt. Daher wird für jede Zeile in jeder Partition derselbe erste Wert ausgewählt.

Im Fall von Kalifornien hat `Qualcomm Stadium` die größte Zahl von Sitzplätzen (`70561`). Daher ist dieser Name der erste Wert für alle Zeilen in der Partition `CA`.

```
select venuestate, venueseats, venuename,
first_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |          venuename            |         first_value
-----------+------------+-------------------------------
+-----------------------------
CA         |      70561 | Qualcomm Stadium              | Qualcomm Stadium
CA         |      69843 | Monster Park                  | Qualcomm Stadium
CA         |      63026 | McAfee Coliseum               | Qualcomm Stadium
CA         |      56000 | Dodger Stadium                | Qualcomm Stadium
CA         |      45050 | Angel Stadium of Anaheim      | Qualcomm Stadium
CA         |      42445 | PETCO Park                    | Qualcomm Stadium
```

```
CA         |      41503 | AT&T Park                       | Qualcomm Stadium
CA         |      22000 | Shoreline Amphitheatre          | Qualcomm Stadium
CO         |      76125 | INVESCO Field                   | INVESCO Field
CO         |      50445 | Coors Field                     | INVESCO Field
DC         |      41888 | Nationals Park                  | Nationals Park
FL         |      74916 | Dolphin Stadium                 | Dolphin Stadium
FL         |      73800 | Jacksonville Municipal Stadium | Dolphin Stadium
FL         |      65647 | Raymond James Stadium           | Dolphin Stadium
FL         |      36048 | Tropicana Field                 | Dolphin Stadium
...
```

Im folgenden Beispiel wird die Verwendung der Option IGNORE NULLS gezeigt, abhängig von der Hinzufügung einer neuen Zeile zur Tabelle VENUE:

```
insert into venue values(2000,null,'Stanford','CA',90000);
```

Diese neue Zeile enthält einen NULL-Wert für die Spalte VENUENAME. Anschließend wird die FIRST_VALUE-Abfrage wiederholt, die früher in diesem Abschnitt gezeigt wurde:

```
select venuestate, venueseats, venuename,
first_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |         venuename          | first_value
-----------+------------+----------------------------+-------------
CA         |      90000 | NULL                       | NULL
CA         |      70561 | Qualcomm Stadium           | NULL
CA         |      69843 | Monster Park               | NULL
...
```

Da die neue Zeile den größten VENUESEATS-Wert enthält (90000) und der VENUENAME NULL ist, gibt die Funktion FIRST_VALUE für die Partition CA NULL zurück. Um Zeilen wie diese in der Funktionsauswertung zu ignorieren, fügen Sie dem Funktionsargument die Option IGNORE NULLS hinzu:

```
select venuestate, venueseats, venuename,
first_value(venuename) ignore nulls
```

```
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venuestate='CA')
order by venuestate;

venuestate | venueseats |         venuename          |   first_value
-----------+------------+----------------------------+-----------------
CA         |      90000 | NULL                       | Qualcomm Stadium
CA         |      70561 | Qualcomm Stadium           | Qualcomm Stadium
CA         |      69843 | Monster Park               | Qualcomm Stadium
...
```

Die Fensterfunktion LAG

Die Fensterfunktion LAG gibt die Werte für eine Zeile in einem bestimmten Offset oberhalb (vor) der aktuellen Zeile in der Partition zurück.

Syntax

```
LAG (value_expr [, offset ])
[ IGNORE NULLS | RESPECT NULLS ]
OVER ( [ PARTITION BY window_partition ] ORDER BY window_ordering )
```

Argumente

value_expr

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

offset

Ein optionaler Parameter, der die Anzahl der Zeilen vor der aktuellen Zeile angibt, für die Werte zurückgegeben werden sollen. Beim Offset kann es sich um eine ganzzahlige Konstante oder um einen Ausdruck handeln, der zu einer Ganzzahl ausgewertet wird. Wenn Sie keinen Offset angeben, verwendet Amazon Redshift 1 als Standardwert. Ein Offset von `0` gibt die aktuelle Zeile an.

IGNORE NULLS

Eine optionale Spezifikation, die angibt, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte überspringen soll. Wenn IGNORE NULLS nicht angegeben wird, werden Null-Werte berücksichtigt.

> **Note**
>
> Sie können einen NVL- oder COALESCE-Ausdruck verwenden, um die Null-Werte durch einen anderen Wert zu ersetzen. Weitere Informationen finden Sie unter NVL- und COALESCE-Funktionen.

RESPECT NULLS

Gibt an, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte berücksichtigen soll. Wenn Sie IGNORE NULLS nicht angeben, wird RESPECT NULLS standardmäßig unterstützt.

OVER

Gibt die Fensterpartitionierung und -anordnung an. Die OVER-Klausel darf keine Fensterrahmenspezifikation enthalten.

PARTITION BY window_partition

Ein optionales Argument, das den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

ORDER BY window_ordering

Sortiert die Zeilen innerhalb der einzelnen Partitionen.

Die Fensterfunktion LAG unterstützt Ausdrücke, die einen der Amazon-Redshift-Datentypen verwenden. Der Rückgabetyp ist mit dem Typ von value_expr identisch.

Beispiele

Im folgenden Beispiel wird die Menge der Tickets gezeigt, die an den Käufer mit der Käufer-ID 3 verkauft wurden, sowie die Uhrzeit, zu der Käufer 3 die Tickets gekauft hat. Um jeden Verkauf mit dem vorherigen Kauf für Käufer 3 zu vergleichen, gibt die Abfrage für jeden Verkauf die vorherige Menge zurück, die verkauft wurde. Da vor dem 16.01.2008 kein Kauf stattfand, ist der erste Wert für die vorherige verkaufte Menge null:

```
select buyerid, saletime, qtysold,
lag(qtysold,1) over (order by buyerid, saletime) as prev_qtysold
from sales where buyerid = 3 order by buyerid, saletime;
```

```
buyerid |      saletime       | qtysold | prev_qtysold
--------+---------------------+---------+--------------
3 | 2008-01-16 01:06:09 |       1 |
3 | 2008-01-28 02:10:01 |       1 |            1
3 | 2008-03-12 10:39:53 |       1 |            1
3 | 2008-03-13 02:56:07 |       1 |            1
3 | 2008-03-29 08:21:39 |       2 |            1
3 | 2008-04-27 02:39:01 |       1 |            2
3 | 2008-08-16 07:04:37 |       2 |            1
3 | 2008-08-22 11:45:26 |       2 |            2
3 | 2008-09-12 09:11:25 |       1 |            2
3 | 2008-10-01 06:22:37 |       1 |            1
3 | 2008-10-20 01:55:51 |       2 |            1
3 | 2008-10-28 01:30:40 |       1 |            2
(12 rows)
```

Die Fensterfunktion LAST_VALUE

Bei einem geordneten Satz von Zeilen gibt die Funktion LAST_VALUE den Wert des Ausdrucks in Bezug auf die letzte Zeile im Rahmen zurück.

Informationen zur Auswahl der ersten Zeile im Rahmen finden Sie unter Die Fensterfunktion FIRST_VALUE.

Syntax

```
LAST_VALUE( expression )[ IGNORE NULLS | RESPECT NULLS ]
OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Argumente

expression

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

IGNORE NULLS

Die Funktion gibt den letzten Wert im Rahmen zurück, der nicht NULL ist (oder NULL, wenn alle Werte NULL sind).

RESPECT NULLS

Gibt an, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte berücksichtigen soll. Wenn Sie IGNORE NULLS nicht angeben, wird RESPECT NULLS standardmäßig unterstützt.

OVER

Führt die Fensterklauseln für die Funktion ein.

PARTITION BY expr_list

Definiert das Fenster für die Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn die PARTITION BY-Klausel nicht angegeben ist, sortiert ORDER BY die gesamte Tabelle. Wenn Sie eine ORDER BY-Klausel angeben, müssen Sie auch eine frame_clause angeben.

Die Ergebnisse sind von der Anordnung der Daten abhängig. Die Ergebnisse sind in den folgenden Fällen nicht deterministisch:

- Wenn keine ORDER BY-Klausel angegeben ist und eine Partition zwei verschiedene Werte für einen Ausdruck enthält
- Wenn der Ausdruck zu verschiedenen Werten ausgewertet wird, die demselben Wert in der ORDER BY-Liste entsprechen

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen im geordneten Ergebnis. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Rückgabetyp

Diese Funktionen unterstützen Ausdrücke, die einfache Amazon-Redshift-Datentypen verwenden. Der Rückgabetyp ist mit dem Datentyp von expression identisch.

Beispiele

In den folgenden Beispielen wird die Tabelle VENUE aus den TICKIT-Beispieldaten verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

Im folgenden Beispiel wird die Sitzplatzkapazität für die einzelnen Veranstaltungsorte in der Tabelle VENUE zurückgegeben, wobei die Ergebnisse nach Kapazität (hoch zu niedrig) geordnet sind. Die Funktion LAST_VALUE wird verwendet, um den Namen des Veranstaltungsorts auszuwählen, der der letzten Zeile im Rahmen entspricht, in diesem Fall der Zeile mit der geringsten Anzahl von Sitzplätzen. Die Ergebnisse werden nach Bundesstaat partitioniert. Wenn der Wert für VENUESTATE geändert wird, wird daher ein neuer letzter Wert ausgewählt. Der Fensterrahmen ist unbegrenzt. Daher wird für jede Zeile in jeder Partition derselbe letzte Wert ausgewählt.

Im Fall von Kalifornien wird `Shoreline Amphitheatre` für jede Zeile in der Partition zurückgegeben, da es die kleinste Zahl von Sitzplätzen hat (`22000`).

```
select venuestate, venueseats, venuename,
last_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |           venuename            |          last_value
-----------+------------+--------------------------------
+-----------------------------
CA         |      70561 | Qualcomm Stadium               | Shoreline Amphitheatre
CA         |      69843 | Monster Park                   | Shoreline Amphitheatre
CA         |      63026 | McAfee Coliseum                | Shoreline Amphitheatre
CA         |      56000 | Dodger Stadium                 | Shoreline Amphitheatre
CA         |      45050 | Angel Stadium of Anaheim       | Shoreline Amphitheatre
CA         |      42445 | PETCO Park                     | Shoreline Amphitheatre
CA         |      41503 | AT&T Park                      | Shoreline Amphitheatre
CA         |      22000 | Shoreline Amphitheatre         | Shoreline Amphitheatre
CO         |      76125 | INVESCO Field                  | Coors Field
CO         |      50445 | Coors Field                    | Coors Field
DC         |      41888 | Nationals Park                 | Nationals Park
FL         |      74916 | Dolphin Stadium                | Tropicana Field
FL         |      73800 | Jacksonville Municipal Stadium | Tropicana Field
FL         |      65647 | Raymond James Stadium          | Tropicana Field
FL         |      36048 | Tropicana Field                | Tropicana Field
```

```
...
```

Die Fensterfunktion LEAD

Die Fensterfunktion LEAD gibt die Werte für eine Zeile in einem bestimmten Offset unterhalb (nach) der aktuellen Zeile in der Partition zurück.

Syntax

```
LEAD (value_expr [, offset ])
[ IGNORE NULLS | RESPECT NULLS ]
OVER ( [ PARTITION BY window_partition ] ORDER BY window_ordering )
```

Argumente

value_expr

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

offset

Ein optionaler Parameter, der die Anzahl der Zeilen unterhalb der aktuellen Zeile angibt, für die Werte zurückgegeben werden sollen. Beim Offset kann es sich um eine ganzzahlige Konstante oder um einen Ausdruck handeln, der zu einer Ganzzahl ausgewertet wird. Wenn Sie keinen Offset angeben, verwendet Amazon Redshift 1 als Standardwert. Ein Offset von `0` gibt die aktuelle Zeile an.

IGNORE NULLS

Eine optionale Spezifikation, die angibt, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte überspringen soll. Wenn IGNORE NULLS nicht angegeben wird, werden Null-Werte berücksichtigt.

> **Note**
>
> Sie können einen NVL- oder COALESCE-Ausdruck verwenden, um die Null-Werte durch einen anderen Wert zu ersetzen. Weitere Informationen finden Sie unter NVL- und COALESCE-Funktionen.

RESPECT NULLS

Gibt an, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte berücksichtigen soll. Wenn Sie IGNORE NULLS nicht angeben, wird RESPECT NULLS standardmäßig unterstützt.

OVER

Gibt die Fensterpartitionierung und -anordnung an. Die OVER-Klausel darf keine Fensterrahmenspezifikation enthalten.

PARTITION BY window_partition

Ein optionales Argument, das den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

ORDER BY window_ordering

Sortiert die Zeilen innerhalb der einzelnen Partitionen.

Die Fensterfunktion LEAD unterstützt Ausdrücke, die einen der Amazon-Redshift-Datentypen verwenden. Der Rückgabetyp ist mit dem Typ von value_expr identisch.

Beispiele

Im folgenden Beispiel wird die Provision für Veranstaltungen in der Tabelle SALES angegeben, für die am 1. und 2. Januar 2008 Tickets verkauft wurden, sowie die Provision, die für verkaufte Tickets im anschließenden Verkauf gezahlt wurden. In diesem Beispiel wird die Musterdatenbank TICKIT verwendet. Weitere Informationen finden Sie unter Beispieldatenbank.

```
SELECT eventid, commission, saletime, LEAD(commission, 1) over ( ORDER BY saletime ) AS
 next_comm
FROM sales
WHERE saletime BETWEEN '2008-01-09 00:00:00' AND '2008-01-10 12:59:59'
LIMIT 10;

+---------+------------+---------------------+-----------+
| eventid | commission |      saletime       | next_comm |
+---------+------------+---------------------+-----------+
|    1664 |       13.2 | 2008-01-09 01:00:21 |      69.6 |
|     184 |       69.6 | 2008-01-09 01:00:36 |     116.1 |
|    6870 |      116.1 | 2008-01-09 01:02:37 |      11.1 |
|    3718 |       11.1 | 2008-01-09 01:05:19 |     205.5 |
```

```
|    6772 |      205.5 | 2008-01-09 01:14:04 |      38.4 |
|    3074 |       38.4 | 2008-01-09 01:26:50 |     209.4 |
|    5254 |      209.4 | 2008-01-09 01:29:16 |      26.4 |
|    3724 |       26.4 | 2008-01-09 01:40:09 |      57.6 |
|    5303 |       57.6 | 2008-01-09 01:40:21 |      51.6 |
|    3678 |       51.6 | 2008-01-09 01:42:54 |      43.8 |
+---------+------------+---------------------+-----------+
```

Die Fensterfunktion LISTAGG

Die Fensterfunktion „LISTAGG“ ordnet die Zeilen der Gruppe in einer Abfrage nach dem ORDER BY-Ausdruck an. Anschließend werden die Werte zu einer einzigen Zeichenfolge verkettet.

LISTAGG ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist. Weitere Informationen finden Sie unter Abfragen der Katalogtabellen.

Syntax

```
LISTAGG( [DISTINCT] expression [, 'delimiter' ] )
[ WITHIN GROUP (ORDER BY order_list) ]
OVER ( [PARTITION BY partition_expression] )
```

Argumente

DISTINCT

(Optional) Eine Klausel, die duplizierte Werte in dem angegebenen Ausdruck beseitigt, bevor die Verkettung vorgenommen wird. Leerzeichen am Ende werden ignoriert, sodass beispielsweise die Zeichenfolgen `'a'` und `'a  '` als duplizierte Werte behandelt werden würden. „LISTAGG“ verwendet den ersten registrierten Wert. Weitere Informationen finden Sie unter Die Bedeutung von Leerzeichen am Ende.

aggregate_expression

Ein gültiger Ausdruck (beispielsweise ein Spaltenname), der die Werte bereitstellt, die aggregiert werden sollen. NULL-Werte und leere Zeichenfolgen werden ignoriert.

delimiter

(Optional) Die Zeichenfolgenkonstante, die die verketteten Werte trennt. Der Standardwert ist „NULL“.

WITHIN GROUP (ORDER BY order_list)

(Optional) Eine Klausel, die die Sortierreihenfolge der aggregierten Werte angibt. Dies ist nur dann deterministisch, wenn ORDER BY eine spezifische Reihenfolge bereitstellt. Das Standardverhalten besteht darin, alle Zeilen zu aggregieren und einen einzelnen Wert zurückzugeben.

OVER

Eine Klausel, die die Fensterpartitionierung angibt. Die OVER-Klausel darf keine Spezifikation für Fensteranordnungen oder Fensterrahmen enthalten.

PARTITION BY partition_expression

(Optional) Legt den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel fest.

Rückgabewert

VARCHAR(MAX). Wenn der Ergebnissatz größer als die maximal zulässige Größe von VARCHAR ist (64.000 – 1 oder 65535), gibt LISTAGG den folgenden Fehler zurück:

```
Invalid operation: Result size exceeds LISTAGG limit
```

Beispiele

Im folgenden Beispiel wird die Tabelle WINSALES verwendet. Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird eine Liste von Verkäufer-IDs zurückgegeben, geordnet nach Verkäufer-ID.

```
select listagg(sellerid)
within group (order by sellerid)
over() from winsales;

  listagg
------------
 11122333344
...
...
 11122333344
 11122333344
   (11 rows)
```

Im folgenden Beispiel wird eine Liste von Verkäufer-IDs für Verkäufer B zurückgegeben, geordnet nach Datum.

```
select listagg(sellerid)
within group (order by dateid)
over () as seller
from winsales
where buyerid = 'b' ;

  seller
---------
    3233
    3233
    3233
    3233
```

Im folgenden Beispiel wird eine durch Komma getrennte Liste von Verkaufsterminen für Käufer B zurückgegeben.

```
select listagg(dateid,',')
within group (order by sellerid desc,salesid asc)
over () as dates
from winsales
where buyerid  = 'b';

             dates
----------------------------------------
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
```

Im folgenden Beispiel wird mit „DISTINCT“ eine Liste von einzigartigen Verkaufsterminen für Käufer B zurückgegeben.

```
select listagg(distinct dateid,',')
within group (order by sellerid desc,salesid asc)
over () as dates
from winsales
where buyerid  = 'b';

           dates
```

```
-------------------------------
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
```

Im folgenden Beispiel wird eine durch Komma getrennte Liste von Verkaufs-IDs für die einzelnen Käufer-IDs zurückgegeben.

```
select buyerid,
listagg(salesid,',')
within group (order by salesid)
over (partition by buyerid) as sales_id
from winsales
order by buyerid;

+---------+------------------------+
| buyerid |        sales_id        |
+---------+------------------------+
| a       | 10005,40001,40005      |
| a       | 10005,40001,40005      |
| a       | 10005,40001,40005      |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
+---------+------------------------+
```

Die Fensterfunktion MAX

Die Fensterfunktion MAX gibt den maximal zulässigen Wert der Eingabeausdruckswerte zurück. Die Funktion MAX ist mit numerischen Werten kompatibel und ignoriert NULL-Werte.

Syntax

```
MAX ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
```

```
[ ORDER BY order_list frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem Ausdruck bei. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Eine Klausel, die die Fensterklauseln für die Aggregationsfunktionen angibt. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die MAX-Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Akzeptiert alle Datentypen als Eingabe. Gibt denselben Datentyp wie expression zurück.

Beispiele

Das folgende Beispiel zeigt die Verkaufs-ID, die Menge und die maximale Menge ab dem Beginn des Datenfensters an:

```
select salesid, qty,
max(qty) over (order by salesid rows unbounded preceding) as max
from winsales
order by salesid;

salesid | qty | max
---------+-----+-----
10001 |  10 |  10
10005 |  30 |  30
10006 |  10 |  30
20001 |  20 |  30
20002 |  20 |  30
30001 |  10 |  30
30003 |  15 |  30
30004 |  20 |  30
30007 |  30 |  30
40001 |  40 |  40
40005 |  10 |  40
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Das folgende Beispiel zeigt die Verkaufs-ID, die Menge und die maximale Menge in einem eingeschränkten Rahmen an:

```
select salesid, qty,
max(qty) over (order by salesid rows between 2 preceding and 1 preceding) as max
from winsales
order by salesid;

salesid | qty | max
---------+-----+-----
10001 |  10 |
10005 |  30 |  10
10006 |  10 |  30
20001 |  20 |  30
20002 |  20 |  20
30001 |  10 |  20
30003 |  15 |  20
30004 |  20 |  15
30007 |  30 |  20
40001 |  40 |  30
```

```
40005 |  10 |  40
(11 rows)
```

Die Fensterfunktion MEDIAN

Berechnet den Medianwert für den Wertebereich in einem Fenster oder einer Partition. NULL-Werte im Bereich werden ignoriert.

MEDIAN ist eine Funktion für die inverse Verteilung, die ein kontinuierliches Verteilungsmodell annimmt.

MEDIAN ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Syntax

```
MEDIAN ( median_expression )
OVER ( [ PARTITION BY partition_expression ] )
```

Argumente

median_expression

Ein Ausdruck (beispielsweise ein Spaltenname), der die Werte bereitstellt, für die der Median ermittelt werden soll. Der Datentyp des Ausdrucks muss entweder numerisch oder Datum/Uhrzeit sein oder implizit in einen solchen konvertierbar sein.

OVER

Eine Klausel, die die Fensterpartitionierung angibt. Die OVER-Klausel darf keine Spezifikation für Fensteranordnungen oder Fensterrahmen enthalten.

PARTITION BY partition_expression

Optional. Ein Ausdruck, der den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

Datentypen

Der Rückgabetyp wird durch den Datentyp von median_expression festgelegt. Die folgende Tabelle zeigt den Rückgabetyp für jeden median_expression-Datentyp an.

Typ der Eingabe	Typ der Rückgabe
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATUM	DATUM

Nutzungshinweise

Wenn das Argument median_expression den Datentyp DECIMAL hat und mit der maximal zulässigen Präzision von 38 Stellen definiert ist, gibt MEDIAN möglicherweise ein falsches Ergebnis oder einen Fehler zurück. Wenn der Rückgabewert der Funktion MEDIAN 38 Stellen überschreitet, wird das Ergebnis entsprechend abgekürzt. Dies führt zu einem Genauigkeitsverlust. Wenn während der Interpolierung ein Zwischenergebnis die maximal zulässige Genauigkeit überschreitet, erfolgt ein numerischer Überlauf und die Funktion gibt einen Fehler zurück. Um diese Bedingungen zu vermeiden, werden die Verwendung eines Datentyps mit einer niedrigeren Genauigkeit oder die Umwandlung des Arguments median_expression in ein Argument mit niedrigerer Genauigkeit empfohlen.

Beispielsweise gibt eine SUM-Funktion mit einem DECIMAL-Argument standardmäßig eine Präzision von 38 Stellen zurück. Die Ergebnisskala ist die gleiche wie die Skala des Arguments. Eine SUM für eine DECIMAL(5,2)-Spalte gibt also einen DECIMAL(38,2)-Datentyp zurück.

Im folgenden Beispiel wird eine SUM-Funktion im Argument median_expression einer MEDIAN-Funktion verwendet. Der Datentyp der Spalte PRICEPAID ist DECIMAL(8,2). Daher gibt die SUM-Funktion DECIMAL(38,2) zurück.

```
select salesid, sum(pricepaid), median(sum(pricepaid))
over() from sales where salesid < 10 group by salesid;
```

Um einen möglichen Präzisionsverlust oder Overflow-Fehler zu vermeiden, wandeln Sie das Ergebnis in einen DECIMAL-Datentyp mit einer niedrigeren Präzision um, wie im folgenden Beispiel gezeigt.

```
select salesid, sum(pricepaid), median(sum(pricepaid)::decimal(30,2))
over() from sales where salesid < 10 group by salesid;
```

Beispiele

Im folgenden Beispiel wird der Median für die Verkaufsmenge für die einzelnen Verkäufer berechnet:

```
select sellerid, qty, median(qty)
over (partition by sellerid)
from winsales
order by sellerid;


sellerid qty median
---------------------------
1  10 10.0
1  10 10.0
1  30 10.0
2  20 20.0
2  20 20.0
3  10 17.5
3  15 17.5
3  20 17.5
3  30 17.5
4  10 25.0
4  40 25.0
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Die Fensterfunktion MIN

Die Fensterfunktion MIN gibt den mindestens erforderlichen Wert der Eingabeausdruckswerte zurück. Die Funktion MIN ist mit numerischen Werten kompatibel und ignoriert NULL-Werte.

Syntax

```
MIN ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem Ausdruck bei. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Gibt die Fensterklauseln für die Aggregationsfunktionen an. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die MIN-Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Akzeptiert alle Datentypen als Eingabe. Gibt denselben Datentyp wie expression zurück.

Beispiele

Das folgende Beispiel zeigt die Verkaufs-ID, die Menge und die Mindestmenge ab dem Beginn des Datenfensters an:

```
select salesid, qty,
min(qty) over
```

```
(order by salesid rows unbounded preceding)
from winsales
order by salesid;

salesid | qty | min
---------+-----+-----
10001 |  10 |  10
10005 |  30 |  10
10006 |  10 |  10
20001 |  20 |  10
20002 |  20 |  10
30001 |  10 |  10
30003 |  15 |  10
30004 |  20 |  10
30007 |  30 |  10
40001 |  40 |  10
40005 |  10 |  10
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Das folgende Beispiel zeigt die Verkaufs-ID, die Menge und die Mindestmenge in einem eingeschränkten Rahmen an:

```
select salesid, qty,
min(qty) over
(order by salesid rows between 2 preceding and 1 preceding) as min
from winsales
order by salesid;

salesid | qty | min
---------+-----+-----
10001 |  10 |
10005 |  30 |  10
10006 |  10 |  10
20001 |  20 |  10
20002 |  20 |  10
30001 |  10 |  20
30003 |  15 |  10
30004 |  20 |  10
30007 |  30 |  15
40001 |  40 |  20
```

```
40005 |  10 |  30
(11 rows)
```

Die Fensterfunktion NTH_VALUE

Die Fensterfunktion NTH_VALUE gibt den Ausdruckswert der angegebenen Zeile des Fensterrahmens in Bezug auf die erste Zeile des Fensters zurück.

Syntax

```
NTH_VALUE (expr, offset)
[ IGNORE NULLS | RESPECT NULLS ]
OVER
( [ PARTITION BY window_partition ]
[ ORDER BY window_ordering
                        frame_clause ] )
```

Argumente

expr

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

offset

Legt die Zeilenzahl in Bezug auf die erste Zeile in dem Fenster zurück, für das der Ausdruck zurückgegeben werden soll. Beim Offset kann es sich um eine Konstante oder einen Ausdruck handeln. Es muss sich um eine positive Ganzzahl größer als 0 handeln.

IGNORE NULLS

Eine optionale Spezifikation, die angibt, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte überspringen soll. Wenn IGNORE NULLS nicht angegeben wird, werden Null-Werte berücksichtigt.

RESPECT NULLS

Gibt an, dass Amazon Redshift bei der Festlegung der Zeile, die verwendet werden soll, Null-Werte berücksichtigen soll. Wenn Sie IGNORE NULLS nicht angeben, wird RESPECT NULLS standardmäßig unterstützt.

OVER

Gibt die Fensterpartitionierung und -anordnung sowie den Fensterrahmen an.

PARTITION BY window_partition

Legt den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel fest.

ORDER BY window_ordering

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn die ORDER BY-Klausel ausgelassen wird, besteht der Rahmen standardmäßig aus allen Zeilen in der Partition.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen im geordneten Ergebnis. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Die Fensterfunktion NTH_VALUE unterstützt Ausdrücke, die einen der Amazon-Redshift-Datentypen verwenden. Der Rückgabetyp ist mit dem Typ von expr identisch.

Beispiele

Im folgenden Beispiel wird die Anzahl der Sitzplätze der drittgrößten Veranstaltungsorte in Kalifornien, Florida, und New York gezeigt, verglichen mit der Anzahl der Sitzplätze der anderen Veranstaltungsorte in diesen Bundesstaaten:

```
select venuestate, venuename, venueseats,
nth_value(venueseats, 3)
ignore nulls
over(partition by venuestate order by venueseats desc
rows between unbounded preceding and unbounded following)
as third_most_seats
from (select * from venue where venueseats > 0 and
venuestate in('CA', 'FL', 'NY'))
order by venuestate;

venuestate |           venuename            | venueseats | third_most_seats
-----------+--------------------------------+------------+------------------
CA         | Qualcomm Stadium               |      70561 |            63026
CA         | Monster Park                   |      69843 |            63026
CA         | McAfee Coliseum                |      63026 |            63026
CA         | Dodger Stadium                 |      56000 |            63026
```

```
CA         | Angel Stadium of Anaheim       |      45050 |           63026
CA         | PETCO Park                     |      42445 |           63026
CA         | AT&T Park                      |      41503 |           63026
CA         | Shoreline Amphitheatre         |      22000 |           63026
FL         | Dolphin Stadium                |      74916 |           65647
FL         | Jacksonville Municipal Stadium |      73800 |           65647
FL         | Raymond James Stadium          |      65647 |           65647
FL         | Tropicana Field                |      36048 |           65647
NY         | Ralph Wilson Stadium           |      73967 |           20000
NY         | Yankee Stadium                 |      52325 |           20000
NY         | Madison Square Garden          |      20000 |           20000
(15 rows)
```

Die Fensterfunktion NTILE

Die Fensterfunktion NTILE teilt angeordnete Zeilen in der Partition so gleichmäßig wie möglich in die angegebene Zahl von Gruppen mit Rangfestlegung auf und gibt die Gruppe zurück, zu der eine bestimmte Zeile gehört.

Syntax

```
NTILE (expr)
OVER (
[ PARTITION BY expression_list ]
[ ORDER BY order_list ]
)
```

Argumente

expr

Die Anzahl der Gruppen mit Rangfestlegung; muss für jede Partition einen positiven Ganzzahlwert (größer als 0) als Ergebnis haben. Das Argument expr darf nicht nullwertfähig sein.

OVER

Eine Klausel, die die Fensterpartitionierung und -anordnung angibt. Die OVER-Klausel darf keine Fensterrahmenspezifikation enthalten.

PARTITION BY window_partition

Optional. Der Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel.

ORDER BY window_ordering

Optional. Ein Ausdruck, der die Zeilen innerhalb der einzelnen Partitionen sortiert. Wenn die ORDER BY-Klausel ausgelassen wird, bleibt das Rangfestlegungsverhalten gleich.

Wenn ORDER BY nicht zu einer spezifischen Reihenfolge führt, ist die Reihenfolge der Zeilen nicht deterministisch. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

Rückgabetyp

BIGINT

Beispiele

Im folgenden Beispiel wird der Preis, der am 26. August 2008 für Hamlet-Tickets gezahlt wurde, in vier Rangfestlegungsgruppen angezeigt. Das Ergebnis sind 17 Zeilen, die beinahe gleichmäßig auf die Rangfestlegungsgruppen 1 bis 4 aufgeteilt sind:

```
select eventname, caldate, pricepaid, ntile(4)
over(order by pricepaid desc) from sales, event, date
where sales.eventid=event.eventid and event.dateid=date.dateid and eventname='Hamlet'
and caldate='2008-08-26'
order by 4;

eventname |  caldate   | pricepaid | ntile
----------+------------+-----------+-------
Hamlet    | 2008-08-26 |   1883.00 |     1
Hamlet    | 2008-08-26 |   1065.00 |     1
Hamlet    | 2008-08-26 |    589.00 |     1
Hamlet    | 2008-08-26 |    530.00 |     1
Hamlet    | 2008-08-26 |    472.00 |     1
Hamlet    | 2008-08-26 |    460.00 |     2
Hamlet    | 2008-08-26 |    355.00 |     2
Hamlet    | 2008-08-26 |    334.00 |     2
Hamlet    | 2008-08-26 |    296.00 |     2
Hamlet    | 2008-08-26 |    230.00 |     3
Hamlet    | 2008-08-26 |    216.00 |     3
Hamlet    | 2008-08-26 |    212.00 |     3
Hamlet    | 2008-08-26 |    106.00 |     3
Hamlet    | 2008-08-26 |    100.00 |     4
Hamlet    | 2008-08-26 |     94.00 |     4
Hamlet    | 2008-08-26 |     53.00 |     4
```

```
Hamlet    | 2008-08-26 |     25.00 |     4
(17 rows)
```

Die Fensterfunktion PERCENT_RANK

Berechnet den prozentualen Rang einer bestimmten Zeile. Der prozentuale Rang wird anhand der folgenden Formel festgelegt:

`(x - 1) / (the number of rows in the window or partition - 1)`

wobei x der Rang der aktuellen Zeile ist. Der folgende Datensatz zeigt die Verwendung dieser Formel:

```
Row# Value Rank Calculation PERCENT_RANK
1 15 1 (1-1)/(7-1) 0.0000
2 20 2 (2-1)/(7-1) 0.1666
3 20 2 (2-1)/(7-1) 0.1666
4 20 2 (2-1)/(7-1) 0.1666
5 30 5 (5-1)/(7-1) 0.6666
6 30 5 (5-1)/(7-1) 0.6666
7 40 7 (7-1)/(7-1) 1.0000
```

Der Rückgabewertbereich ist 0 bis 1 (einschließlich). Die erste Zeile in jedem Satz besitzt den PERCENT_RANK 0.

Syntax

```
PERCENT_RANK ()
OVER (
[ PARTITION BY partition_expression ]
[ ORDER BY order_list ]
)
```

Argumente

( )

Die Funktion verwendet keine Argumente. Es ist jedoch eine leere Klammer erforderlich.

OVER

Eine Klausel, die die Fensterpartitionierung angibt. Die OVER-Klausel darf keine Fensterrahmenspezifikation enthalten.

PARTITION BY partition_expression

Optional. Ein Ausdruck, der den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

ORDER BY order_list

Optional. Der Ausdruck, anhand dessen der prozentuale Rang berechnet wird. Der Datentyp des Ausdrucks muss entweder numerisch sein oder implizit in einen solchen konvertierbar sein. Wenn ORDER BY ausgelassen wird, ist der Rückgabewert für alle Zeilen 0.

Wenn ORDER BY nicht zu einer spezifischen Reihenfolge führt, ist die Reihenfolge der Zeilen nicht deterministisch. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

Rückgabetyp

FLOAT8

Beispiele

Im folgenden Beispiel wird der prozentuale Rang der Verkaufsmengen für die einzelnen Verkäufer berechnet:

```
select sellerid, qty, percent_rank()
over (partition by sellerid order by qty)
from winsales;

sellerid qty  percent_rank
---------------------------------------
1  10.00  0.0
1  10.64  0.5
1  30.37  1.0
3  10.04  0.0
3  15.15  0.33
3  20.75  0.67
3  30.55  1.0
2  20.09  0.0
2  20.12  1.0
4  10.12  0.0
4  40.23  1.0
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Fensterfunktion PERCENTILE_CONT

PERCENTILE_CONT ist eine Funktion für die inverse Verteilung, die ein kontinuierliches Verteilungsmodell annimmt. Sie empfängt einen Perzentilwert und eine Sortierspezifikation und gibt einen interpolierten Wert zurück, der in Bezug auf die Sortierspezifikation in den angegebenen Perzentilwert fällt.

PERCENTILE_CONT berechnet eine lineare Interpolierung zwischen Werten, nachdem diese der Reihenfolge entsprechend angeordnet wurden. Mithilfe des Perzentilwerts `(P)` und der Anzahl der Nicht-Null-Zeilen `(N)` in der Aggregationsgruppe berechnet die Funktion die Anzahl der Zeilen, nachdem die Zeilen entsprechend der Sortierspezifikation angeordnet wurden. Die Anzahl von Zeilen `(RN)` wird mit der Formel `RN = (1+ (P*(N-1))` berechnet. Das Endergebnis der Aggregationsfunktion wird durch lineare Interpolierung zwischen den Werten aus Zeilen zwischen `CRN = CEILING(RN)` und `FRN = FLOOR(RN)` berechnet.

Das Ergebnis wird wie folgt aussehen.

Wenn `(CRN = FRN = RN)`, ist das Ergebnis `(value of expression from row at RN)`

Andernfalls sieht das Ergebnis wie folgt aus:

`(CRN - RN) * (value of expression for row at FRN) + (RN - FRN) * (value of expression for row at CRN)`.

Sie können in der OVER-Klausel nur die PARTITION-Klausel angeben. Wenn PARTITION angegeben ist, gibt PERCENTILE_CONT für jede Zeile den Wert zurück, der in einem Satz von Werten innerhalb einer bestimmten Partition in das angegebene Perzentil fallen würde.

PERCENTILE_CONT ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Syntax

```
PERCENTILE_CONT ( percentile )
WITHIN GROUP (ORDER BY expr)
OVER (  [ PARTITION BY expr_list ]  )
```

Argumente

percentile

Numerische Konstante zwischen 0 und 1. Null-Werte werden bei der Berechnung ignoriert.

WITHIN GROUP (ORDER BY expr)

Gibt numerische oder Datum-/Zeitwerte an, nach denen das Perzentil sortiert und berechnet werden soll.

OVER

Gibt die Fensterpartitionierung an. Die OVER-Klausel darf keine Spezifikation für Fensteranordnungen oder Fensterrahmen enthalten.

PARTITION BY expr

Optionales Argument, das den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

Rückgabewert

Der Rückgabetyp wird durch den Datentyp des ORDER BY-Ausdrucks in der WITHIN GROUP-Klausel festgelegt. Die folgende Tabelle zeigt den Rückgabetyp für jeden ORDER BY-Datentyp an.

Typ der Eingabe	Typ der Rückgabe
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATUM	DATUM
TIMESTAMP	TIMESTAMP

Nutzungshinweise

Wenn das Argument ORDER BY den Datentyp DECIMAL hat und mit der maximal zulässigen Präzision von 38 Stellen definiert ist, gibt PERCENTILE_CONT möglicherweise ein falsches Ergebnis oder einen Fehler zurück. Wenn der Rückgabewert der Funktion PERCENTILE_CONT 38 Stellen

überschreitet, wird das Ergebnis entsprechend abgekürzt. Dies führt zu einem Genauigkeitsverlust. Wenn während der Interpolierung ein Zwischenergebnis die maximal zulässige Genauigkeit überschreitet, erfolgt ein numerischer Überlauf und die Funktion gibt einen Fehler zurück. Um diese Bedingungen zu vermeiden, werden die Verwendung eines Datentyps mit einer niedrigeren Genauigkeit oder die Umwandlung des Ausdrucks ORDER BY in einen Ausdruck mit niedrigerer Genauigkeit empfohlen.

Beispielsweise gibt eine SUM-Funktion mit einem DECIMAL-Argument standardmäßig eine Präzision von 38 Stellen zurück. Die Ergebnisskala ist die gleiche wie die Skala des Arguments. Eine SUM für eine DECIMAL(5,2)-Spalte gibt also einen DECIMAL(38,2)-Datentyp zurück.

Im folgenden Beispiel wird in der ORDER BY-Klausel einer PERCENTILE_CONT-Funktion eine SUM-Funktion verwendet. Der Datentyp der Spalte PRICEPAID ist DECIMAL(8,2). Daher gibt die SUM-Funktion DECIMAL(38,2) zurück.

```
select salesid, sum(pricepaid), percentile_cont(0.6)
within group (order by sum(pricepaid) desc) over()
from sales where salesid < 10 group by salesid;
```

Um einen möglichen Präzisionsverlust oder Overflow-Fehler zu vermeiden, wandeln Sie das Ergebnis in einen DECIMAL-Datentyp mit einer niedrigeren Präzision um, wie im folgenden Beispiel gezeigt.

```
select salesid, sum(pricepaid), percentile_cont(0.6)
within group (order by sum(pricepaid)::decimal(30,2) desc) over()
from sales where salesid < 10 group by salesid;
```

Beispiele

Im folgenden Beispiel wird die Tabelle WINSALES verwendet. Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

```
select sellerid, qty, percentile_cont(0.5)
within group (order by qty)
over() as median from winsales;

 sellerid | qty | median
----------+-----+--------
        1 |  10 |   20.0
        1 |  10 |   20.0
        3 |  10 |   20.0
```

```
        4 |  10 |   20.0
        3 |  15 |   20.0
        2 |  20 |   20.0
        3 |  20 |   20.0
        2 |  20 |   20.0
        3 |  30 |   20.0
        1 |  30 |   20.0
        4 |  40 |   20.0
(11 rows)
```

```
select sellerid, qty, percentile_cont(0.5)
within group (order by qty)
over(partition by sellerid) as median from winsales;

 sellerid | qty | median
----------+-----+--------
        2 |  20 |   20.0
        2 |  20 |   20.0
        4 |  10 |   25.0
        4 |  40 |   25.0
        1 |  10 |   10.0
        1 |  10 |   10.0
        1 |  30 |   10.0
        3 |  10 |   17.5
        3 |  15 |   17.5
        3 |  20 |   17.5
        3 |  30 |   17.5
(11 rows)
```

Im folgenden Beispiel wird der Wert für PERCENTILE_CONT und PERCENTILE_DISC für Ticketverkäufe von Verkäufern im Bundesstaat Washington berechnet.

```
SELECT sellerid, state, sum(qtysold*pricepaid) sales,
percentile_cont(0.6) within group (order by sum(qtysold*pricepaid::decimal(14,2) )
 desc) over(),
percentile_disc(0.6) within group (order by sum(qtysold*pricepaid::decimal(14,2) )
 desc) over()
from sales s, users u
where s.sellerid = u.userid and state = 'WA' and sellerid < 1000
group by sellerid, state;

 sellerid | state |  sales  | percentile_cont | percentile_disc
----------+-------+---------+-----------------+-----------------
```

```
      127 | WA    | 6076.00 |         2044.20 |         1531.00
      787 | WA    | 6035.00 |         2044.20 |         1531.00
      381 | WA    | 5881.00 |         2044.20 |         1531.00
      777 | WA    | 2814.00 |         2044.20 |         1531.00
       33 | WA    | 1531.00 |         2044.20 |         1531.00
      800 | WA    | 1476.00 |         2044.20 |         1531.00
        1 | WA    | 1177.00 |         2044.20 |         1531.00
(7 rows)
```

Die Fensterfunktion PERCENTILE_DISC

PERCENTILE_DISC ist eine Funktion für die inverse Verteilung, die ein diskretes Verteilungsmodell annimmt. Sie empfängt einen Perzentilwert und eine Sortierspezifikation und gibt ein Element aus dem angegebenen Satz zurück.

PERCENTILE_DISC sortiert für den Perzentilwert P die Werte des Ausdrucks in der ORDER BY-Klausel und gibt den Wert mit dem kleinsten kumulativen Verteilungswert (in Bezug auf dieselbe Sortierspezifikation) zurück, der größer als oder gleich P ist.

Sie können in der OVER-Klausel nur die PARTITION-Klausel angeben.

PERCENTILE_DISC ist eine reine Datenverarbeitungsknoten-Funktion. Die Funktion gibt einen Fehler zurück, wenn die Abfrage auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist.

Syntax

```
PERCENTILE_DISC ( percentile )
WITHIN GROUP (ORDER BY expr)
OVER (  [ PARTITION BY expr_list ]  )
```

Argumente

percentile

Numerische Konstante zwischen 0 und 1. Null-Werte werden bei der Berechnung ignoriert.

WITHIN GROUP (ORDER BY expr)

Gibt numerische oder Datum-/Zeitwerte an, nach denen das Perzentil sortiert und berechnet werden soll.

OVER

Gibt die Fensterpartitionierung an. Die OVER-Klausel darf keine Spezifikation für Fensteranordnungen oder Fensterrahmen enthalten.

PARTITION BY expr

Optionales Argument, das den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

Rückgabewert

Derselbe Datentyp wie der ORDER BY-Ausdruck in der WITHIN GROUP-Klausel.

Beispiele

Im folgenden Beispiel wird die Tabelle WINSALES benutzt. Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

```
SELECT sellerid, qty, PERCENTILE_DISC(0.5)
WITHIN GROUP (ORDER BY qty)
OVER() AS MEDIAN FROM winsales;

+----------+-----+--------+
| sellerid | qty | median |
+----------+-----+--------+
| 3        | 10  | 20     |
| 1        | 10  | 20     |
| 1        | 10  | 20     |
| 4        | 10  | 20     |
| 3        | 15  | 20     |
| 2        | 20  | 20     |
| 2        | 20  | 20     |
| 3        | 20  | 20     |
| 1        | 30  | 20     |
| 3        | 30  | 20     |
| 4        | 40  | 20     |
+----------+-----+--------+

SELECT sellerid, qty, PERCENTILE_DISC(0.5)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS MEDIAN FROM winsales;
```

```
+----------+-----+--------+
| sellerid | qty | median |
+----------+-----+--------+
| 4        | 10  | 10     |
| 4        | 40  | 10     |
| 3        | 10  | 15     |
| 3        | 15  | 15     |
| 3        | 20  | 15     |
| 3        | 30  | 15     |
| 2        | 20  | 20     |
| 2        | 20  | 20     |
| 1        | 10  | 10     |
| 1        | 10  | 10     |
| 1        | 30  | 10     |
+----------+-----+--------+
```

Benutzen Sie die folgenden Beispiele, um PERCENTILE_DISC(0.25) und PERCENTILE_DISC(0.75) für die Menge zu ermitteln, wenn nach der Verkäufer-ID partitioniert wird.

```
SELECT sellerid, qty, PERCENTILE_DISC(0.25)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS quartile1 FROM winsales;

+----------+-----+-----------+
| sellerid | qty | quartile1 |
+----------+-----+-----------+
| 4        | 10  | 10        |
| 4        | 40  | 10        |
| 2        | 20  | 20        |
| 2        | 20  | 20        |
| 3        | 10  | 10        |
| 3        | 15  | 10        |
| 3        | 20  | 10        |
| 3        | 30  | 10        |
| 1        | 10  | 10        |
| 1        | 10  | 10        |
| 1        | 30  | 10        |
+----------+-----+-----------+

SELECT sellerid, qty, PERCENTILE_DISC(0.75)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS quartile3 FROM winsales;
```

```
+----------+-----+-----------+
| sellerid | qty | quartile3 |
+----------+-----+-----------+
| 3        | 10  | 20        |
| 3        | 15  | 20        |
| 3        | 20  | 20        |
| 3        | 30  | 20        |
| 4        | 10  | 40        |
| 4        | 40  | 40        |
| 2        | 20  | 20        |
| 2        | 20  | 20        |
| 1        | 10  | 30        |
| 1        | 10  | 30        |
| 1        | 30  | 30        |
+----------+-----+-----------+
```

Die Fensterfunktion RANK

Die Fensterfunktion RANK legt den Rang eines Werts in einer Gruppe von Werten fest, basierend auf dem ORDER BY-Ausdruck in der OVER-Klausel. Wenn die optionale PARTITION BY-Klausel vorhanden ist, wird die Rangfolge für jede Gruppe von Zeilen neu festgelegt. Zeilen mit gleichen Werten in Bezug auf die Rangfestlegungskriterien erhalten den gleichen Rang. Amazon Redshift fügt die Anzahl der gleichwertigen Zeilen dem gleichwertigen Rang hinzu, um den nächsten Rang zu berechnen. Daher sind die Ränge möglicherweise nicht konsekutiv. Wenn beispielsweise zwei Zeilen den Rang 1 erhalten, ist der nächste Rang 3.

RANK unterscheidet sich in einer Hinsicht von Die Fensterfunktion DENSE_RANK: Wenn zwei oder mehr Zeilen den gleichen Rang erhalten, entsteht bei DENSE_RANK in der Rangfolge der Werte keine Lücke. Wenn beispielsweise zwei Zeilen den Rang 1 erhalten, ist der nächste Rang 2.

Sie können in derselben Abfrage Rangfestlegungsfunktionen mit unterschiedlichen PARTITION BY- und ORDER BY-Klauseln verwenden.

Syntax

```
RANK () OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list ]
)
```

Argumente

( )

Die Funktion verwendet keine Argumente. Es ist jedoch eine leere Klammer erforderlich.

OVER

Die Fensterklauseln für die Funktion RANK.

PARTITION BY expr_list

Optional. Ein oder mehrere Ausdrücke, der/die das Fenster definiert/definieren.

ORDER BY order_list

Optional. Definiert die Spalten, auf denen die Rangfestlegungswerte basieren. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle. Wenn ORDER BY ausgelassen wird, ist der Rückgabewert für alle Zeilen 1.

Wenn ORDER BY nicht zu einer spezifischen Reihenfolge führt, ist die Reihenfolge der Zeilen nicht deterministisch. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

Rückgabetyp

INTEGER

Beispiele

Im folgenden Beispiel wird die Tabelle nach der verkauften Menge (standardmäßig in aufsteigender Reihenfolge) geordnet und jeder Zeile einen Rang zugewiesen. Der Rangwert 1 ist der Wert mit dem höchsten Rang. Die Ergebnisse werden sortiert, nachdem die Fensterfunktionsergebnisse angewendet wurden:

```
select salesid, qty,
rank() over (order by qty) as rnk
from winsales
order by 2,1;

salesid | qty | rnk
--------+-----+-----
```

```
10001 |  10 |  1
10006 |  10 |  1
30001 |  10 |  1
40005 |  10 |  1
30003 |  15 |  5
20001 |  20 |  6
20002 |  20 |  6
30004 |  20 |  6
10005 |  30 |  9
30007 |  30 |  9
40001 |  40 |  11
(11 rows)
```

Beachten Sie, dass die äußere ORDER-BY-Klausel in diesem Beispiel die Spalten 2 und 1 einschließt, um sicherzustellen, dass Amazon Redshift bei jeder Ausführung dieser Abfrage konsistent sortierte Ergebnisse zurückgibt. Die Zeilen mit den Verkaufs-IDs 10001 und 10006 besitzen beispielsweise identische Werte für QTY und RNK. Durch die Anordnung des endgültigen Ergebnissatzes nach Spalte 1 wird sichergestellt, dass die Zeile 10001 stets vor der Zeile 10006 angeordnet wird. Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird die Anordnung für die Fensterfunktion () umgekehrt. (`order by qty desc`). Jetzt wird der höchste Rangwert auf den größten QTY-Wert angewendet.

```
select salesid, qty,
rank() over (order by qty desc) as rank
from winsales
order by 2,1;

 salesid | qty | rank
---------+-----+-----
   10001 |  10 |   8
   10006 |  10 |   8
   30001 |  10 |   8
   40005 |  10 |   8
   30003 |  15 |   7
   20001 |  20 |   4
   20002 |  20 |   4
   30004 |  20 |   4
   10005 |  30 |   2
   30007 |  30 |   2
   40001 |  40 |   1
```

```
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird die Tabelle nach SELLERID partitioniert, die einzelnen Partitionen nach Menge (in absteigender Reihenfolge) geordnet und jeder Zeile ein Rang zugewiesen. Die Ergebnisse werden sortiert, nachdem die Fensterfunktionsergebnisse angewendet wurden.

```
select salesid, sellerid, qty, rank() over
(partition by sellerid
order by qty desc) as rank
from winsales
order by 2,3,1;

salesid | sellerid | qty | rank
--------+----------+-----+-----
  10001 |        1 |  10 |  2
  10006 |        1 |  10 |  2
  10005 |        1 |  30 |  1
  20001 |        2 |  20 |  1
  20002 |        2 |  20 |  1
  30001 |        3 |  10 |  4
  30003 |        3 |  15 |  3
  30004 |        3 |  20 |  2
  30007 |        3 |  30 |  1
  40005 |        4 |  10 |  2
  40001 |        4 |  40 |  1
(11 rows)
```

Die Fensterfunktion RATIO_TO_REPORT

Berechnet das Verhältnis eines Werts zur Summe der Werte in einem Fenster oder einer Partition. Der RATIO_TO_REPORT-Wert wird anhand der folgenden Formel festgelegt:

`value of` ratio_expression `argument for the current row / sum of` ratio_expression `argument for the window or partition`

Der folgende Datensatz zeigt die Verwendung dieser Formel:

```
Row# Value Calculation RATIO_TO_REPORT
1 2500 (2500)/(13900) 0.1798
```

```
2 2600 (2600)/(13900) 0.1870
3 2800 (2800)/(13900) 0.2014
4 2900 (2900)/(13900) 0.2086
5 3100 (3100)/(13900) 0.2230
```

Der Rückgabewertbereich ist 0 bis 1 (einschließlich). Wenn ratio_expression NULL ist, dann ist der Rückgabewert NULL. Wenn ein Wert in partition_expression eindeutig ist, gibt die Funktion für diesen Wert 1 zurück.

Syntax

```
RATIO_TO_REPORT ( ratio_expression )
OVER ( [ PARTITION BY partition_expression ] )
```

Argumente

ratio_expression

Ein Ausdruck (beispielsweise ein Spaltenname), der den Wert bereitstellt, für den das Verhältnis ermittelt werden soll. Der Datentyp des Ausdrucks muss entweder numerisch sein oder implizit in einen solchen konvertierbar sein.

Sie können in ratio_expression keine anderen analytischen Funktionen verwenden.

OVER

Eine Klausel, die die Fensterpartitionierung angibt. Die OVER-Klausel darf keine Spezifikation für Fensteranordnungen oder Fensterrahmen enthalten.

PARTITION BY partition_expression

Optional. Ein Ausdruck, der den Datensatzbereich für die einzelnen Gruppen in der OVER-Klausel festlegt.

Rückgabetyp

FLOAT8

Beispiele

Im folgenden Beispiel wird die Tabelle WINSALES benutzt. Weitere Informationen zum Erstellen der WINSALES-Tabelle finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird der ratio-to-report Wert jeder Zeile der Menge eines Verkäufers als Summe aller Mengen des Verkäufers berechnet.

```
select sellerid, qty, ratio_to_report(qty)
over()
from winsales
order by sellerid;

sellerid  qty    ratio_to_report
-------------------------------------
1         30     0.13953488372093023
1         10     0.046511627906976744
1         10     0.046511627906976744
2         20     0.09302325581395349
2         20     0.09302325581395349
3         30     0.13953488372093023
3         20     0.09302325581395349
3         15     0.06976744186046512
3         10     0.046511627906976744
4         10     0.046511627906976744
4         40     0.18604651162790697
```

Im folgenden Beispiel werden die Verhältnisse der Verkaufsmengen für die einzelnen Verkäufer nach Partition berechnet.

```
select sellerid, qty, ratio_to_report(qty)
over(partition by sellerid)
from winsales;

sellerid   qty    ratio_to_report
-----------------------------------------
2          20     0.5
2          20     0.5
4          40     0.8
4          10     0.2
1          10     0.2
1          30     0.6
1          10     0.2
3          10     0.13333333333333333
3          15     0.2
3          20     0.26666666666666666
3          30     0.4
```

Die Fensterfunktion ROW_NUMBER

Weist die Ordnungszahl der aktuellen Zeile innerhalb einer Gruppe von Zeilen zu, ab 1 zählend, basierend auf dem ORDER BY-Ausdruck in der OVER-Klausel. Wenn die optionale PARTITION BY-Klausel vorhanden ist, werden die Ordnungszahlen für jede Gruppe von Zeilen neu festgelegt. Zeilen mit gleichen Werten für die ORDER BY-Ausdrücke erhalten auf nicht deterministische Weise unterschiedliche Zeilenzahlen.

Syntax

```
ROW_NUMBER() OVER(
  [ PARTITION BY expr_list ]
  [ ORDER BY order_list ]
)
```

Argumente

( )

Die Funktion verwendet keine Argumente. Es ist jedoch eine leere Klammer erforderlich.

OVER

Die Fensterfunktionsklausel für die Funktion ROW_NUMBER.

PARTITION BY expr_list

Optional. Ein oder mehrere Spaltenausdrücke, welche die Ergebnisse in Zeilensätze unterteilen.

ORDER BY order_list

Optional. Ein oder mehrere Spaltenausdrücke, welche die Reihenfolge der Zeilen innerhalb eines Satzes definieren. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

Wenn ORDER BY nicht zu einer eindeutigen Reihenfolge führt oder ausgelassen wird, ist die Reihenfolge der Zeilen nicht deterministisch. Weitere Informationen finden Sie unter Spezifisches Anordnen von Daten für Fensterfunktionen.

Rückgabetyp

BIGINT

Beispiele

Die folgenden Beispiele verwenden die Tabelle WINSALES. Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird die Tabelle nach QTY (in aufsteigender Reihenfolge) sortiert. Anschließend wird jeder Zeile eine Zeilennummer zugewiesen. Die Ergebnisse werden sortiert, nachdem die Fensterfunktionsergebnisse angewendet wurden.

```
SELECT salesid, sellerid, qty,
ROW_NUMBER() OVER(
   ORDER BY qty ASC) AS row
FROM winsales
ORDER BY 4,1;

salesid   sellerid   qty   row
---------+----------+-----+-----
   30001 |        3 |  10 |   1
   10001 |        1 |  10 |   2
   10006 |        1 |  10 |   3
   40005 |        4 |  10 |   4
   30003 |        3 |  15 |   5
   20001 |        2 |  20 |   6
   20002 |        2 |  20 |   7
   30004 |        3 |  20 |   8
   10005 |        1 |  30 |   9
   30007 |        3 |  30 |  10
   40001 |        4 |  40 |  11
```

Im folgenden Beispiel werden die Tabelle nach SELLERID partitioniert und die einzelnen Partitionen nach QTY angeordnet (in aufsteigender Reihenfolge). Anschließend wird jeder Zeile eine Zeilennummer zugewiesen. Die Ergebnisse werden sortiert, nachdem die Fensterfunktionsergebnisse angewendet wurden.

```
SELECT salesid, sellerid, qty,
ROW_NUMBER() OVER(
  PARTITION BY sellerid
  ORDER BY qty ASC) AS row_by_seller
FROM winsales
ORDER BY 2,4;

 salesid | sellerid | qty | row_by_seller
```

```
---------+----------+-----+-----
   10001 |        1 |  10 |   1
   10006 |        1 |  10 |   2
   10005 |        1 |  30 |   3
   20001 |        2 |  20 |   1
   20002 |        2 |  20 |   2
   30001 |        3 |  10 |   1
   30003 |        3 |  15 |   2
   30004 |        3 |  20 |   3
   30007 |        3 |  30 |   4
   40005 |        4 |  10 |   1
   40001 |        4 |  40 |   2
```

Das folgende Beispiel zeigt die Ergebnisse, wenn die optionalen Klauseln nicht verwendet werden.

```
SELECT salesid, sellerid, qty, ROW_NUMBER() OVER() AS row
FROM winsales
ORDER BY 4,1;

salesid   sellerid   qty   row
---------+----------+-----+-----
   30001 |        3 |  10 |   1
   10001 |        1 |  10 |   2
   10005 |        1 |  30 |   3
   40001 |        4 |  40 |   4
   10006 |        1 |  10 |   5
   20001 |        2 |  20 |   6
   40005 |        4 |  10 |   7
   20002 |        2 |  20 |   8
   30003 |        3 |  15 |   9
   30004 |        3 |  20 |  10
   30007 |        3 |  30 |  11
```

Die Fensterfunktionen STDDEV_SAMP und STDDEV_POP

Die Fensterfunktionen STDDEV_SAMP und STDDEV_POP geben die Stichproben- und Populationsstandardabweichungen eines Satzes numerischer Werte (integer, decimal oder floating-point) zurück. Weitere Informationen finden Sie auch unter Die Funktionen STDDEV_SAMP und STDDEV_POP.

STDDEV_SAMP und STDDEV sind Synonyme für dieselbe Funktion.

Syntax

```
STDDEV_SAMP | STDDEV | STDDEV_POP
( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem Ausdruck bei. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Gibt die Fensterklauseln für die Aggregationsfunktionen an. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Die von den STDDEV-Funktion SUM unterstützten Argumenttypen sind SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL und DOUBLE PRECISION.

Unabhängig vom Datentyp des Ausdrucks ist der Rückgabewert einer STDDEV-Funktion eine DOUBLE PRECISION-Zahl.

Beispiele

Das folgende Beispiel zeigt, wie die Funktionen STDDEV_POP und VAR_POP als Fensterfunktionen verwendet werden. Die Abfrage berechnet Populationsvarianz und Populationsstandardabweichung für PRICEPAID-Werte in der Tabelle SALES.

```
select salesid, dateid, pricepaid,
round(stddev_pop(pricepaid) over
(order by dateid, salesid rows unbounded preceding)) as stddevpop,
round(var_pop(pricepaid) over
(order by dateid, salesid rows unbounded preceding)) as varpop
from sales
order by 2,1;

salesid | dateid | pricepaid | stddevpop | varpop
--------+--------+-----------+-----------+---------
  33095 |   1827 |    234.00 |         0 |       0
  65082 |   1827 |    472.00 |       119 |   14161
  88268 |   1827 |    836.00 |       248 |   61283
  97197 |   1827 |    708.00 |       230 |   53019
 110328 |   1827 |    347.00 |       223 |   49845
 110917 |   1827 |    337.00 |       215 |   46159
 150314 |   1827 |    688.00 |       211 |   44414
 157751 |   1827 |   1730.00 |       447 |  199679
 165890 |   1827 |   4192.00 |      1185 | 1403323
...
```

Die Funktionen für Stichprobenstandardabweichung und -varianz können auf die gleiche Weise verwendet werden.

Die Fensterfunktion SUM

Die Fensterfunktion SUM gibt die Summe der Eingabespalten- oder Ausdruckswerte zurück. Die Funktion SUM ist mit numerischen Werten kompatibel und ignoriert NULL-Werte.

Syntax

```
SUM ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem Ausdruck bei. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Gibt die Fensterklauseln für die Aggregationsfunktionen an. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die SUM-Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Die von der Funktion SUM unterstützten Argumenttypen sind SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL und DOUBLE PRECISION.

Die von der Funktion SUM unterstützten Rückgabetypen sind:

- BIGINT für SMALLINT- oder INTEGER-Argumente
- NUMERIC für BIGINT-Argumente
- DOUBLE PRECISION für Gleitkomma-Argumente

Beispiele

Im folgenden Beispiel wird eine kumulative (gleitende) Summe von Verkaufsmengen nach Datum und Verkaufs-ID erstellt:

```
select salesid, dateid, sellerid, qty,
sum(qty) over (order by dateid, salesid rows unbounded preceding) as sum
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | sum
---------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  20
10005 | 2003-12-24 |        1 |  30 |  50
40001 | 2004-01-09 |        4 |  40 |  90
10006 | 2004-01-18 |        1 |  10 | 100
20001 | 2004-02-12 |        2 |  20 | 120
40005 | 2004-02-12 |        4 |  10 | 130
20002 | 2004-02-16 |        2 |  20 | 150
30003 | 2004-04-18 |        3 |  15 | 165
30004 | 2004-04-18 |        3 |  20 | 185
30007 | 2004-09-07 |        3 |  30 | 215
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel wird eine kumulative (gleitende) Summe von Verkaufsmengen nach Datum erstellt, die Ergebnisse nach Verkäufer-ID partitioniert und die Ergebnisse innerhalb der Partition nach Datum und Verkaufs-ID geordnet:

```
select salesid, dateid, sellerid, qty,
sum(qty) over (partition by sellerid
order by dateid, salesid rows unbounded preceding) as sum
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | sum
---------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  10
10005 | 2003-12-24 |        1 |  30 |  40
40001 | 2004-01-09 |        4 |  40 |  40
10006 | 2004-01-18 |        1 |  10 |  50
20001 | 2004-02-12 |        2 |  20 |  20
40005 | 2004-02-12 |        4 |  10 |  50
20002 | 2004-02-16 |        2 |  20 |  40
30003 | 2004-04-18 |        3 |  15 |  25
30004 | 2004-04-18 |        3 |  20 |  45
30007 | 2004-09-07 |        3 |  30 |  75
(11 rows)
```

Im folgenden Beispiel werden alle Zeilen im Ergebnissatz sequenziell nummeriert, geordnet nach den Spalten SELLERID und SALESID:

```
select salesid, sellerid, qty,
sum(1) over (order by sellerid, salesid rows unbounded preceding) as rownum
from winsales
order by 2,1;

salesid | sellerid |  qty | rownum
--------+----------+------+--------
10001 |        1 |   10 |     1
10005 |        1 |   30 |     2
10006 |        1 |   10 |     3
20001 |        2 |   20 |     4
20002 |        2 |   20 |     5
30001 |        3 |   10 |     6
30003 |        3 |   15 |     7
30004 |        3 |   20 |     8
```

```
30007 |        3 |   30 |     9
40001 |        4 |   40 |    10
40005 |        4 |   10 |    11
(11 rows)
```

Eine Beschreibung der Tabelle WINSALES finden Sie unter Beispieltabelle mit Beispielen von Fensterfunktionen.

Im folgenden Beispiel werden alle Zeilen im Ergebnissatz sequenziell nummeriert, die Ergebnisse nach SELLERID partitioniert und die Ergebnisse innerhalb der Partition nach SELLERID und SALESID geordnet:

```
select salesid, sellerid, qty,
sum(1) over (partition by sellerid
order by sellerid, salesid rows unbounded preceding) as rownum
from winsales
order by 2,1;

salesid | sellerid | qty | rownum
---------+----------+-----+--------
10001 |        1 |  10 |      1
10005 |        1 |  30 |      2
10006 |        1 |  10 |      3
20001 |        2 |  20 |      1
20002 |        2 |  20 |      2
30001 |        3 |  10 |      1
30003 |        3 |  15 |      2
30004 |        3 |  20 |      3
30007 |        3 |  30 |      4
40001 |        4 |  40 |      1
40005 |        4 |  10 |      2
(11 rows)
```

Die Fensterfunktionen VAR_SAMP und VAR_POP

Die Fensterfunktionen VAR_SAMP und VAR_POP geben die Stichproben- und Populationsabweichung eines Satzes numerischer Werte (integer, decimal oder floating-point) zurück. Weitere Informationen finden Sie auch unter Die Funktionen VAR_SAMP und VAR_POP.

VAR_SAMP und VARIANCE sind Synonyme für dieselbe Funktion.

Syntax

```
VAR_SAMP | VARIANCE | VAR_POP
( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Argumente

Ausdruck

Die Zielspalte oder der Ausdruck, für die/den die Funktion ausgeführt wird.

ALL

Mit dem Argument ALL behält die Funktion alle duplizierten Werte aus dem Ausdruck bei. ALL ist das Standardargument. DISTINCT wird nicht unterstützt.

OVER

Gibt die Fensterklauseln für die Aggregationsfunktionen an. Die OVER-Klausel unterscheidet Fensteraggregationsfunktionen von normalen Satzaggregationsfunktionen.

PARTITION BY expr_list

Definiert das Fenster für die Funktion in Bezug auf mindestens einen Ausdruck.

ORDER BY order_list

Sortiert die Zeilen innerhalb der einzelnen Partitionen. Wenn PARTITION BY nicht angegeben ist, verwendet ORDER BY die gesamte Tabelle.

frame_clause

Wenn eine ORDER BY-Klausel für eine Aggregationsfunktion verwendet wird, ist eine explizite Rahmenklausel erforderlich. Die Rahmenklausel gibt den Satz von Zeilen im Fenster einer Funktion genauer an, einschließlich oder ausschließlich Sätzen von Zeilen innerhalb des geordneten Ergebnisses. Die Rahmenklausel besteht aus dem Schlüsselwort ROWS und verknüpften Spezifikatoren. Siehe Übersicht über die Syntax von Fensterfunktionen.

Datentypen

Die von den VARIANCE-Funktion SUM unterstützten Argumenttypen sind SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL und DOUBLE PRECISION.

Unabhängig vom Datentyp des Ausdrucks ist der Rückgabewert einer VARIANCE-Funktion eine DOUBLE PRECISION-Zahl.

Systemadministratorfunktionen

Themen

Amazon Redshift unterstützt verschiedene Systemadministratorfunktionen.

CHANGE_QUERY_PRIORITY

CHANGE_QUERY_PRIORITY ermöglicht Superusern die Änderung der Priorität einer Abfrage, die in Workload Management (WLM) wartet oder ausgeführt wird.

Diese Funktion ermöglicht Superusern, die Priorität jeder Abfrage im System sofort zu ändern. Es kann immer nur eine Abfrage, ein Benutzer oder eine Sitzung mit der Priorität ausgeführt werden `CRITICAL`.

Syntax

```
CHANGE_QUERY_PRIORITY(query_id, priority)
```

Argumente

query_id

Die Abfrage-ID der Abfrage, deren Priorität geändert wird. Benötigt einen `INTEGER`-Wert.

priority

Die neue Priorität, die der Abfrage zugewiesen werden soll. Bei diesem Argument muss es sich um eine Zeichenfolge mit dem Wert `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW` oder `LOWEST` handeln.

Typ der Rückgabe

None

Beispiele

Verwenden Sie das folgende Beispiel, um die Spalte `query_priority` in der Systemtabelle STV_WLM_QUERY_STATE anzuzeigen.

```
SELECT query, service_class, query_priority, state
FROM stv_wlm_query_state WHERE service_class = 101;

+-------+---------------+----------------+---------+
| query | service_class | query_priority |  state  |
+-------+---------------+----------------+---------+
|  1076 |           101 | Lowest         | Running |
|  1075 |           101 | Lowest         | Running |
+-------+---------------+----------------+---------+
```

Verwenden Sie das folgende Beispiel, um die Ergebnisse für einen Superuser zu zeigen, der die Funktion `change_query_priority` ausführt, um die Priorität in `CRITICAL` zu ändern.

```
SELECT CHANGE_QUERY_PRIORITY(1076, 'Critical');

+-----------------------------------------------------------------------------+
|                           change_query_priority                             |
+-----------------------------------------------------------------------------+
| Succeeded to change query priority. Priority changed from Lowest to Critical. |
+-----------------------------------------------------------------------------+
```

CHANGE_SESSION_PRIORITY

CHANGE_SESSION_PRIORITY ermöglicht Superusern, die Priorität jeder Sitzung im System sofort zu ändern. Es kann immer nur eine Sitzung, ein Benutzer oder eine Abfrage mit der Priorität ausgeführt werden `CRITICAL`.

Syntax

```
CHANGE_SESSION_PRIORITY(pid, priority)
```

Argumente

pid

Die Prozess-ID der Sitzung, deren Priorität geändert wird. Der Wert `-1` bezieht sich auf die aktuelle Sitzung. Benötigt einen `INTEGER`-Wert.

priority

Die neue Priorität, die der Sitzung zugewiesen werden soll. Bei diesem Argument muss es sich um eine Zeichenfolge mit dem Wert `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW` oder `LOWEST` handeln.

Rückgabetyp

None

Beispiele

Verwenden Sie das folgende Beispiel, um die Prozess-ID des Serverprozesses zurückzugeben, der die aktuelle Sitzung verarbeitet.

```
SELECT pg_backend_pid();

+----------------+
| pg_backend_pid |
+----------------+
|          30311 |
+----------------+
```

In diesem Beispiel wird die Priorität der aktuellen Sitzung in `LOWEST` geändert.

```
SELECT CHANGE_SESSION_PRIORITY(30311, 'Lowest');

+-------------------------------------------------------------------------------
+
|                               change_session_priority
 |
+-------------------------------------------------------------------------------
+
| Succeeded to change session priority. Changed session (pid:30311) priority to lowest.
 |
+-------------------------------------------------------------------------------
+
```

In diesem Beispiel wird die Priorität der aktuellen Sitzung in `HIGH` geändert.

```
SELECT CHANGE_SESSION_PRIORITY(-1, 'High');

+-------------------------------------------------------------------------------
+
|                                  change_session_priority
           |
+-------------------------------------------------------------------------------
+
| Succeeded to change session priority. Changed session (pid:30311) priority from
 lowest to high. |
+-------------------------------------------------------------------------------
+
```

Verwenden Sie das folgende Beispiel, um eine gespeicherte Prozedur zum Ändern der Priorität einer Sitzung zu erstellen. Die Berechtigung zum Ausführen dieser gespeicherten Prozedur wird dem Datenbankbenutzer erteilt `test_user`.

```
CREATE OR REPLACE PROCEDURE sp_priority_low(pid IN int, result OUT varchar)
AS $$
BEGIN
  SELECT CHANGE_SESSION_PRIORITY(pid, 'low') into result;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER;
GRANT EXECUTE ON PROCEDURE sp_priority_low(int) TO test_user;
```

Anschließend ruft der Datenbankbenutzer mit dem Namen `test_user` die Prozedur auf.

```
CALL sp_priority_low(pg_backend_pid());

+-----------------------------------------------------+
|                        result                       |
+-----------------------------------------------------+
| Success. Change session (pid:13155) priority to low. |
+-----------------------------------------------------+
```

CHANGE_USER_PRIORITY

CHANGE_USER_PRIORITY ermöglicht Superusern die Änderung der Priorität aller von einem Benutzer ausgegebenen Abfragen, die in Workload Management (WLM) warten oder ausgeführt werden. Es kann immer nur ein Benutzer, eine Sitzung oder eine Abfrage mit der Priorität ausgeführt werden `CRITICAL`.

Syntax

```
CHANGE_USER_PRIORITY(user_name, priority)
```

Argumente

user_name

Der Name des Datenbankbenutzers, dessen Abfragepriorität geändert wird.

priority

Die neue Priorität, die allen von ausgegebenen Abfragen zugewiesen werden soll `user_name`. Bei diesem Argument muss es sich um eine Zeichenfolge mit dem Wert `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW`, `LOWEST` oder `RESET` handeln. Nur Superuser können die Priorität in ändern `CRITICAL`. Durch das Ändern der Priorität in `RESET` wird die Prioritätseinstellung für `user_name` entfernt.

Rückgabetyp

None

Beispiele

Verwenden Sie das folgende Beispiel, um die Priorität für den Benutzer `analysis_user` in `LOWEST` zu ändern.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'lowest');

+-------------------------------------------------------------------------------------+
|                                change_user_priority                                 |
+-------------------------------------------------------------------------------------+
| Succeeded to change user priority. Changed user (analysis_user) priority to lowest. |
+-------------------------------------------------------------------------------------+
```

Verwenden Sie das folgende Beispiel, um die Priorität für den Benutzer in `LOW` zu ändern.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'low');

+-------------------------------------------------------------------------------------------------------------
+
|                                    change_user_priority
        |
+-------------------------------------------------------------------------------------------------------------
+
| Succeeded to change user priority. Changed user (analysis_user) priority from Lowest
 to low. |
+-------------------------------------------------------------------------------------------------------------
+
```

Verwenden Sie das folgende Beispiel, um die Priorität zurückzusetzen.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'reset');

+-----------------------------------------------------+
|                change_user_priority                 |
+-----------------------------------------------------+
| Succeeded to reset priority for user (analysis_user). |
+-----------------------------------------------------+
```

CURRENT_SETTING

CURRENT_SETTING gibt den aktuellen Wert des angegebenen Konfigurationsparameters zurück.

Diese Funktion entspricht dem Befehl ZEIGEN.

Syntax

```
current_setting('parameter')
```

Die folgende Anweisung gibt den aktuellen Wert der angegebenen Sitzungskontextvariablen zurück.

```
current_setting('variable_name')
current_setting('variable_name'[, error_if_undefined])
```

Argumente

Parameter

Der Parameterwert, der angezeigt werden soll. Eine Liste der Konfigurationsparameter finden Sie unter Konfigurationsreferenz

variable_name

Der Name der Variablen, die angezeigt werden soll. Dies muss eine Zeichenfolgenkonstante für Sitzungskontextvariablen sein.

error_if_undefined

(Optional) Ein optionaler boolescher Wert, der das Verhalten angibt, wenn der Variablenname nicht existiert. Wenn error_if_undefined auf TRUE festgelegt ist, was der Standardwert ist, löst Amazon Redshift einen Fehler aus. Wenn error_if_undefined auf FALSE festgelegt ist, gibt Amazon Redshift NULL zurück. Amazon Redshift unterstützt den Parameter error_if_undefined nur für Sitzungskontextvariablen. Dies kann nicht verwendet werden, wenn die Eingabe ein Konfigurationsparameter ist.

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um die aktuelle Einstellung für den Parameter query_group zurückzugeben.

```
SELECT CURRENT_SETTING('query_group');

+-----------------+
| current_setting |
+-----------------+
| unset           |
+-----------------+
```

Verwenden Sie das folgende Beispiel, um die aktuelle Einstellung für die Variable `app_context.user_id` zurückzugeben.

```
SELECT CURRENT_SETTING('app_context.user_id', FALSE);
```

PG_CANCEL_BACKEND

Bricht eine Abfrage ab. PG_CANCEL_BACKEND entspricht funktionell dem Befehl CANCEL. Sie können Aufträge abbrechen, die zurzeit von Ihrem Benutzer ausgeführt werden. Superuser können alle Abfragen abbrechen.

Syntax

```
pg_cancel_backend( pid )
```

Argumente

pid

Die Prozess-ID (PID) der Abfrage, die abgebrochen werden soll. Sie können eine Abfrage nicht durch Angabe einer Abfrage-ID abbrechen. Sie müssen die Prozess-ID der Abfrage angeben. Benötigt einen `INTEGER`-Wert.

Rückgabetyp

Keine

Nutzungshinweise

Wenn Abfragen in mehreren Sitzungen Sperren für die gleiche Tabelle bewirken, können Sie die Funktion PG_TERMINATE_BACKEND verwenden, um eine der Sitzungen zu beenden. Dadurch werden alle Transaktionen, die zurzeit in der beendeten Sitzung ausgeführt werden, gezwungen, alle Sperren aufzuheben und ein Rollback für die Transaktion auszuführen. Führen Sie eine Abfrage für die Systemtabelle PG__LOCKS aus, um die zurzeit vorhandenen Sperren anzuzeigen. Wenn Sie eine Abfrage nicht abbrechen können, weil sie sich in einem Transaktionsblock befindet (BEGIN … END), können Sie mithilfe der Funktion PG_TERMINATE_BACKEND die Sitzung beenden, in der die Abfrage ausgeführt wird.

Beispiele

Um eine Abfrage abzubrechen, die aktuell ausgeführt wird, rufen Sie zuerst die Prozess-ID für die Abfrage ab, die Sie abbrechen möchten. Um die Prozess-IDs für alle zurzeit ausgeführten Abfragen zu ermitteln, führen Sie den folgenden Befehl aus.

```
SELECT pid, TRIM(starttime) AS start,
duration, TRIM(user_name) AS user,
SUBSTRING(query,1,40) AS querytxt
FROM stv_recents
WHERE status = 'Running';

+-----+-----------------------+----------+--------+-----------------------------+
| pid |       starttime       | duration |  user  |          querytxt           |
+-----+-----------------------+----------+--------+-----------------------------+
| 802 | 2013-10-14 09:19:03.55 |      132 | dwuser | select venuename from venue |
| 834 | 2013-10-14 08:33:49.47 |  1250414 | dwuser | select * from listing;      |
| 964 | 2013-10-14 08:30:43.29 |   326179 | dwuser | select sellerid from sales  |
+-----+-----------------------+----------+--------+-----------------------------+
```

Verwenden Sie das folgende Beispiel, um die Abfrage mit der Prozess-ID 802 abzubrechen.

```
SELECT PG_CANCEL_BACKEND(802);
```

PG_TERMINATE_BACKEND

Beendet eine Sitzung. Sie können eine Sitzung beenden, deren Besitzer Ihr Benutzer ist. Superuser können jede Sitzung beenden.

Syntax

```
pg_terminate_backend( pid )
```

Argumente

pid

Die Prozess-ID der Sitzung, die beendet werden soll. Benötigt einen `INTEGER`-Wert.

Rückgabetyp

Keine

Nutzungshinweise

Wenn Sie nahe der Grenze für gleichzeitige Verbindungen sind, verwenden Sie PG_TERMINATE_BACKEND, um ungenutzte Sitzungen zu beenden und die Verbindungen freizugeben. Weitere Informationen finden Sie unter Amazon-Redshift-Limits.

Wenn Abfragen in mehreren Sitzungen Sperren für die gleiche Tabelle bewirken, können Sie die Funktion PG_TERMINATE_BACKEND verwenden, um eine der Sitzungen zu beenden. Dadurch werden alle Transaktionen, die zurzeit in der beendeten Sitzung ausgeführt werden, gezwungen, alle Sperren aufzuheben und ein Rollback für die Transaktion auszuführen. Führen Sie eine Abfrage für die Katalogtabelle PG_LOCKS aus, um die zurzeit vorhandenen Sperren anzuzeigen.

Wenn sich eine Abfrage nicht in einem Transaktionsblock befindet (BEGIN … END), können Sie mithilfe des Befehls CANCEL oder der Funktion PG_CANCEL_BACKEND die Abfrage abbrechen.

Beispiele

Verwenden Sie das folgende Beispiel, um die Tabelle SVV_TRANSACTIONS anzufragen, um alle Sperren anzuzeigen, die für aktuelle Transaktionen gültig sind.

```
SELECT * FROM svv_transactions;

+----------+--------+-------+------+--------------------+-----------------
+----------------------+----------+---------+
| txn_owner | txn_db |  xid  | pid  |      txn_start      |    lock_mode    |
 lockable_object_type | relation | granted |
+----------+--------+-------+------+--------------------+-----------------
+----------------------+----------+---------+
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |    51940 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |    52000 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |   108623 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | ExclusiveLock   |
 transactionid        |          | true    |
+----------+--------+-------+------+--------------------+-----------------
+----------------------+----------+---------+
```

Verwenden Sie das folgende Beispiel, um die Sitzung mit den Sperren zu beenden.

```
SELECT PG_TERMINATE_BACKEND(8585);
```

REBOOT_CLUSTER

Starten Sie den Amazon-Redshift-Cluster neu, ohne die Verbindungen zum Cluster zu schließen. Sie müssen Datenbank-Superuser sein, um diesen Befehl auszuführen.

Nach dem Abschluss des Soft Reboots gibt der Amazon-Redshift-Cluster einen Fehler an die Benutzeranwendung zurück und verlangt, dass die Benutzeranwendung alle Transaktionen oder Abfragen, die beim Soft Reboot unterbrochen wurden, erneut übermittelt.

Syntax

```
SELECT REBOOT_CLUSTER();
```

SET_CONFIG

Legt einen Konfigurationsparameter auf eine neue Einstellung fest.

Diese Funktion entspricht dem Befehl SET in SQL.

Syntax

```
SET_CONFIG('parameter', 'new_value' , is_local)
```

Die folgende Anweisung setzt eine Sitzungskontextvariable auf eine neue Einstellung.

```
set_config('variable_name', 'new_value' , is_local)
```

Argumente

Parameter

Der Parameter, der festgelegt werden soll.

variable_name

Der Name der Variablen, die gesetzt werden soll.

new_value

Neuer Wert des Parameters.

is_local

Wenn wahr, gilt der Parameterwert nur für die aktuelle Transaktion. Gültige Werte sind `true` oder `1` und `false` oder `0`.

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiele

Verwenden Sie das folgende Beispiel, um nur für die aktuelle Transaktion den Wert des Parameters `query_group` auf `test` festzulegen.

```
SELECT SET_CONFIG('query_group', 'test', true);

+------------+
| set_config |
+------------+
| test       |
+------------+
```

Verwenden Sie das folgende Beispiel, um Sitzungskontextvariablen festzulegen.

```
SELECT SET_CONFIG('app.username', 'cuddy', FALSE);
```

Funktionen für Systeminformationen

Amazon Redshift unterstützt zahlreiche Systeminformationsfunktionen.

Themen

- CURRENT_AWS_ACCOUNT
- CURRENT_DATABASE
- CURRENT_NAMESPACE
- CURRENT_SCHEMA

- CURRENT_SCHEMAS
- CURRENT_USER
- CURRENT_USER_ID
- DEFAULT_IAM_ROLE
- HAS_ASSUMEROLE_PRIVILEGE
- HAS_DATABASE_PRIVILEGE
- HAS_SCHEMA_PRIVILEGE
- HAS_TABLE_PRIVILEGE
- LAST_USER_QUERY_ID
- PG_BACKEND_PID
- PG_GET_COLS
- PG_GET_GRANTEE_BY_IAM_ROLE
- PG_GET_IAM_ROLE_BY_USER
- PG_GET_LATE_BINDING_VIEW_COLS
- PG_GET_SESSION_ROLES
- PG_LAST_COPY_COUNT
- PG_LAST_COPY_ID
- PG_LAST_UNLOAD_ID
- PG_LAST_QUERY_ID
- PG_LAST_UNLOAD_COUNT
- Die Funktion SLICE_NUM
- USER
- ROLE_IS_MEMBER_OF
- USER_IS_MEMBER_OF
- VERSION

CURRENT_AWS_ACCOUNT

Gibt das AWS Konto zurück, das dem Amazon Redshift Redshift-Cluster zugeordnet ist, der eine Anfrage eingereicht hat.

Syntax

```
current_aws_account
```

Rückgabetyp

Gibt eine Ganzzahl zurück.

Beispiel

Die folgende Abfrage gibt den Namen der aktuellen Datenbank zurück.

```
select user, current_aws_account;
current_user | current_account
-------------+---------------
dwuser       | 987654321

(1 row)
```

CURRENT_DATABASE

Gibt den Namen der Datenbank zurück, mit der Sie zurzeit verbunden sind.

Syntax

```
current_database()
```

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiel

Die folgende Abfrage gibt den Namen der aktuellen Datenbank zurück.

```
select current_database();

current_database
------------------
tickit
```

```
(1 row)
```

CURRENT_NAMESPACE

Gibt den Cluster-Namespace des aktuellen Amazon-Redshift-Clusters zurück. Der Amazon-Redshift-Cluster-Namespace ist die eindeutige ID des Amazon-Redshift-Clusters.

Syntax

```
current_namespace
```

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiel

Die folgende Abfrage gibt den Namen des aktuellen Namespace zurück.

```
select user, current_namespace;
current_user | current_namespace
-------------+--------------------------------------
dwuser       | 86b5169f-01dc-4a6f-9fbb-e2e24359e9a8

(1 row)
```

CURRENT_SCHEMA

Gibt den Namen des Schemas am Anfang des Suchpfads zurück. Dieses Schema wird für Tabellen oder andere benannte Objekte verwendet, die ohne Angabe eines Zielschemas erstellt werden.

Syntax

> **Note**
>
> Dies ist eine Führungsknotenfunktion. Diese Funktion gibt einen Fehler zurück, wenn sie eine benutzererstellte Tabelle, eine STL- oder STV-Systemtabelle oder eine SVV- oder SVL-Systemansicht referenziert.

```
current_schema()
```

Rückgabetyp

CURRENT_SCHEMA gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiele

Die folgende Abfrage gibt das aktuelle Schema zurück:

```
select current_schema();

current_schema
----------------
public
(1 row)
```

CURRENT_SCHEMAS

Gibt ein Array der Namen von Schemas im aktuellen Suchpfad zurück. Der aktuelle Suchpfad wird im Parameter „search_path“ definiert.

Syntax

> **Note**
>
> Dies ist eine Führungsknotenfunktion. Diese Funktion gibt einen Fehler zurück, wenn sie eine benutzererstellte Tabelle, eine STL- oder STV-Systemtabelle oder eine SVV- oder SVL-Systemansicht referenziert.

```
current_schemas(include_implicit)
```

Argument

include_implicit

Wenn wahr, gibt dieser Parameter an, dass der Suchpfad alle implizit enthaltenen Systemschemas einschließen soll. Gültige Werte sind `true` und `false`. Wenn `true`, gibt dieser Parameter in der Regel zusätzlich zum aktuellen Schema das Schema `pg_catalog` zurück.

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiele

Im folgenden Beispiel werden die Namen der Schemas im aktuellen Suchpfad ohne implizit enthaltene Systemschemas zurückgegeben:

```
select current_schemas(false);

current_schemas
----------------
{public}
(1 row)
```

Im folgenden Beispiel werden die Namen der Schemas im aktuellen Suchpfad einschließlich implizit enthaltener Systemschemas zurückgegeben:

```
select current_schemas(true);

current_schemas
--------------------
{pg_catalog,public}
(1 row)
```

CURRENT_USER

Gibt den Benutzernamen des aktuellen „effektiven“ Benutzers der Datenbank zurück, abhängig von den Überprüfungsberechtigungen. In der Regel ist dieser Benutzername mit dem Namen des Sitzungsbenutzers identisch. Dies kann jedoch gelegentlich von Superusern geändert worden sein.

> **Note**
>
> Verwenden Sie beim Aufrufen von CURRENT_USER keine Klammern am Ende.

Syntax

```
current_user
```

Rückgabetyp

CURRENT_USER gibt einen NAME-Datentyp zurück und kann als CHAR- oder VARCHAR-Zeichenfolge umgewandelt werden.

Nutzungshinweise

Wenn eine gespeicherte Prozedur mit der Option SECURITY DEFINER des Befehls CREATE_PROCEDURE erstellt wurde, gibt Amazon Redshift beim Aufrufen der CURRENT_USER-Funktion innerhalb der gespeicherten Prozedur den Benutzernamen des Besitzers der gespeicherten Prozedur zurück.

Beispiel

Die folgende Abfrage gibt den Namen des aktuellen Datenbankbenutzers zurück:

```
select current_user;

current_user
-------------
dwuser
(1 row)
```

CURRENT_USER_ID

Gibt den eindeutigen Bezeichner für den Amazon-Redshift-Benutzer zurück, der in der aktuellen Sitzung angemeldet ist.

Syntax

```
CURRENT_USER_ID
```

Rückgabetyp

Die Funktion CURRENT_USER_ID gibt einen Ganzzahlwert zurück.

Beispiele

Im folgenden Beispiel werden der Benutzername und die aktuelle Benutzer-ID für diese Sitzung zurückgegeben:

```
select user, current_user_id;
```

```
 current_user | current_user_id
--------------+-----------------
   dwuser     |               1
(1 row)
```

DEFAULT_IAM_ROLE

Gibt die Standard-IAM-Rolle zurück, die derzeit dem Amazon-Redshift-Cluster zugeordnet ist. Die Funktion gibt „none“ zurück, wenn keine Standard-IAM-Rolle zugeordnet ist.

Syntax

```
select default_iam_role();
```

Rückgabetyp

Gibt eine VARCHAR-Zeichenfolge zurück.

Beispiel

Im folgenden Beispiel wird die Standard-IAM-Rolle zurückgegeben, die derzeit dem angegebenen Amazon-Redshift-Cluster zugeordnet ist.

```
select default_iam_role();
              default_iam_role
---------------------------------------------
 arn:aws:iam::123456789012:role/myRedshiftRole
(1 row)
```

HAS_ASSUMEROLE_PRIVILEGE

Gibt den booleschen Wert `true` (`t`) zurück, wenn der angegebene Benutzer die angegebene IAM-Rolle mit dem Recht zur Ausführung des angegebenen Befehls hat. Die Funktion gibt `false` (`f`), wenn der Benutzer keine angegebene IAM-Rolle mit dem Recht zur Ausführung des angegebenen Befehls hat. Weitere Informationen zu Rechten finden Sie in GRANT.

Syntax

```
has_assumerole_privilege( [ user, ] iam_role_arn, cmd_type)
```

Argumente

user

Der Name des Benutzers, dessen IAM-Rollen-Berechtigungen überprüft werden sollen. Standardmäßig wird der aktuelle Benutzer überprüft. Diese Funktion kann sowohl von Superusern als auch Benutzern verwendet werden. Benutzer können jedoch nur ihre eigenen Berechtigungen anzeigen.

iam_role_arn

Die IAM-Rolle, der die Befehlsberechtigungen erteilt wurden.

cmd_type

Der Befehl, für den der Zugriff gewährt wurde. Gültige Werte:

- COPY
- UNLOAD
- EXTERNAL FUNCTION
- CREATE MODEL

Rückgabetyp

BOOLEAN

Beispiel

Die folgende Abfrage bestätigt, dass der Benutzer `reg_user1` das Recht für die `Redshift-S3-Read`-Rolle hat, den COPY-Befehl auszuführen.

```
select has_assumerole_privilege('reg_user1', 'arn:aws:iam::123456789012:role/Redshift-S3-Read', 'copy');
```

```
has_assumerole_privilege
------------------------
true
(1 row)
```

HAS_DATABASE_PRIVILEGE

Gibt `true` zurück, wenn der Benutzer das angegebene Recht für die angegebene Datenbank besitzt. Weitere Informationen zu Rechten finden Sie in GRANT.

Syntax

> **Note**
>
> Dies ist eine Führungsknotenfunktion. Diese Funktion gibt einen Fehler zurück, wenn sie eine benutzererstellte Tabelle, eine STL- oder STV-Systemtabelle oder eine SVV- oder SVL-Systemansicht referenziert.

```
has_database_privilege( [ user, ] database, privilege)
```

Argumente

user

Der Name des Benutzers, dessen Datenbankberechtigungen überprüft werden sollen. Standardmäßig wird der aktuelle Benutzer überprüft.

Datenbank

Die Datenbank, die mit die Berechtigung verknüpft ist.

privilege

Die Berechtigung, die überprüft werden soll. Gültige Werte:

- CREATE
- TEMPORARY
- TEMP

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiel

Die folgende Abfrage überprüft, ob der Benutzer GUEST für die Datenbank TICKIT die Berechtigung TEMP besitzt.

```
select has_database_privilege('guest', 'tickit', 'temp');

has_database_privilege
-----------------------
true
(1 row)
```

HAS_SCHEMA_PRIVILEGE

Gibt `true` zurück, wenn der Benutzer das angegebene Recht für das angegebene Schema besitzt. Weitere Informationen zu Rechten finden Sie in GRANT.

Syntax

> **Note**
>
> Dies ist eine Führungsknotenfunktion. Diese Funktion gibt einen Fehler zurück, wenn sie eine benutzererstellte Tabelle, eine STL- oder STV-Systemtabelle oder eine SVV- oder SVL-Systemansicht referenziert.

```
has_schema_privilege( [ user, ] schema, privilege)
```

Argumente

user

Der Name des Benutzers, dessen Schemaberechtigungen überprüft werden sollen. Standardmäßig wird der aktuelle Benutzer überprüft.

schema

Das Schema, das mit der Berechtigung verknüpft ist.

privilege

Die Berechtigung, die überprüft werden soll. Gültige Werte:

- CREATE
- USAGE

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiel

Die folgende Abfrage überprüft, ob der Benutzer GUEST für das Schema PUBLIC das Recht CREATE besitzt:

```
select has_schema_privilege('guest', 'public', 'create');

has_schema_privilege
----------------------
true
(1 row)
```

HAS_TABLE_PRIVILEGE

Gibt `true` zurück, wenn der Benutzer die bestimmte Berechtigung für die angegebene Tabelle besitzt. Andernfalls wird `false` zurückgegeben.

Syntax

> **Note**
>
> Dies ist eine Führungsknotenfunktion. Diese Funktion gibt einen Fehler zurück, wenn sie eine benutzererstellte Tabelle, eine STL- oder STV-Systemtabelle oder eine SVV- oder SVL-Systemansicht referenziert. Weitere Informationen zu Rechten finden Sie in GRANT.

```
has_table_privilege( [ user, ] table, privilege)
```

Argumente

user

Der Name des Benutzers, dessen Tabellenberechtigungen überprüft werden sollen. Standardmäßig wird der aktuelle Benutzer überprüft.

Tabelle

Die Tabelle, die mit der Berechtigung verknüpft ist.

privilege

Die Berechtigung, das überprüft werden soll. Gültige Werte:

- SELECT
- INSERT
- AKTUALISIERUNG
- DELETE
- DROP
- REFERENCES

Rückgabetyp

BOOLEAN

Beispiele

Die folgende Abfrage stellt fest, dass der Benutzer GUEST für die Tabelle LISTING nicht die Berechtigung SELECT besitzt.

```
select has_table_privilege('guest', 'listing', 'select');

has_table_privilege
---------------------
false
```

Die folgende Abfrage listet Tabellenberechtigungen auf, einschließlich select, insert, update und delete, wobei die Ausgabe aus den Katalogtabellen pg_tables und pg_user verwendet wird. Dies ist nur ein Beispiel. Möglicherweise müssen Sie einen Schemanamen und Tabellennamen aus Ihrer Datenbank angeben. Weitere Informationen finden Sie unter Abfragen der Katalogtabellen.

```
SELECT
     tablename
     ,usename
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'select') AS sel
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'insert') AS ins
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'update') AS upd
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'delete') AS del
FROM
(SELECT * from pg_tables
WHERE schemaname = 'public' and tablename in ('event','listing')) as tables
,(SELECT * FROM pg_user) AS users;

tablename | usename   |  sel   |  ins  |  upd  | del
----------+-----------+--------+-------+-------+-------
event     |  john     |  true  | true  | true  | true
event     |  sally    |  false | false | false | false
event     |  elsa     |  false | false | false | false
listing   |  john     |  true  | true  | true  | true
listing   |  sally    |  false | false | false | false
listing   |  elsa     |  false | false | false | false
```

Die vorherige Abfrage enthält auch einen Cross Join. Weitere Informationen finden Sie unter JOIN-Beispiele. Wenn Sie Tabellen abfragen möchten, die sich nicht im `public`-Schema befinden, entfernen Sie die `schemaname`-Bedingung aus der WHERE-Klausel und verwenden Sie das folgende Beispiel vor Ihrer Abfrage.

```
SET SEARCH_PATH to 'schema_name';
```

LAST_USER_QUERY_ID

Gibt die Abfrage-ID der in der aktuellen Sitzung zuletzt abgeschlossenen Benutzerabfrage zurück. Wenn in der aktuellen Sitzung keine Abfragen ausgeführt wurden, gibt last_user_query_id den Wert -1 zurück. Die Funktion gibt keine Abfrage-IDs für Abfragen zurück, die ausschließlich auf dem Führungsknoten ausgeführt werden. Weitere Informationen finden Sie unter Exklusive Führungsknotenfunktionen.

Syntax

```
last_user_query_id()
```

Rückgabetyp

Gibt eine Ganzzahl zurück.

Beispiel

Die folgende Abfrage gibt die ID der letzten Abfrage zurück, die von einem Benutzer in der aktuellen Sitzung ausgeführt wurde.

```
select last_user_query_id();
```

Die Ergebnisse sehen wie folgt aus.

```
last_user_query_id
-----------------------
    5437
(1 row)
```

Die folgende Abfrage gibt die Abfrage-ID und den Text der zuletzt abgeschlossenen Abfrage aus, die von einem Benutzer in der aktuellen Sitzung ausgeführt wurde.

```
select query_id, query_text from sys_query_history where query_id =
 last_user_query_id();
```

Die Ergebnisse sehen wie folgt aus.

```
 query_id, query_text
---------
+-----------------------------------------------------------------------------------------------
 5556975 | select last_user_query_id() limit 100 --RequestID=<unique request ID>;
 TraceID=<unique trace ID>
```

PG_BACKEND_PID

Gibt die Prozess-ID (PID) des Serverprozesses zurück, der die aktuelle Sitzung verarbeitet.

> **Note**
>
> Die PID ist global nicht eindeutig. Sie kann über die Zeit wiederverwendet werden.

Syntax

```
pg_backend_pid()
```

Rückgabetyp

Gibt eine Ganzzahl zurück.

Beispiel

Sie können PG_BACKEND_PID mit Protokolltabellen korrelieren, um Informationen für die aktuelle Sitzung abzurufen. Beispielsweise gibt die folgende Abfrage die Abfrage-ID und einen Teil des Abfragetexts für Abfragen aus, die in der aktuellen Sitzung abgeschlossen wurden.

```
select query, substring(text,1,40)
from stl_querytext
where pid =  PG_BACKEND_PID()
order by query desc;

 query |                substring
-------+----------------------------------------
 14831 | select query, substring(text,1,40) from
 14827 | select query, substring(path,0,80) as pa
 14826 | copy category from 's3://dw-tickit/manif
 14825 | Count rows in target table
 14824 | unload ('select * from category') to 's3
(5 rows)
```

Sie können PG_BACKEND_PID mit der pid-Spalte in den folgenden Protokolltabellen korrelieren (Ausnahmen werden in Klammern angegeben):

- STL_CONNECTION_LOG
- STL_DDLTEXT
- STL_ERROR
- STL_QUERY
- STL_QUERYTEXT
- STL_SESSIONS (Verarbeitung)
- STL_TR_CONFLICT

- STL_UTILITYTEXT
- STV_ACTIVE_CURSORS
- STV_INFLIGHT
- STV_LOCKS (lock_owner_pid)
- STV_RECENTS (process_id)

PG_GET_COLS

Gibt die Spaltenmetadaten für eine Tabellen- oder Ansichtdefinition zurück.

Syntax

```
pg_get_cols('name')
```

Argumente

Name

Der Name einer Amazon-Redshift-Tabelle oder -Ansicht. Weitere Informationen finden Sie unter Namen und Kennungen.

Rückgabetyp

VARCHAR

Nutzungshinweise

Die PG_GET_COLS-Funktion gibt für jede Spalte in der Tabelle oder Ansichtdefinition eine Zeile zurück. Die Zeile enthält eine durch Komma getrennte Liste mit dem Schemanamen, dem Beziehungsnamen, dem Spaltennamen, dem Datentyp und der Spaltennummer. Die Formatierung des SQL-Ergebnisses hängt vom verwendeten SQL-Client ab.

Beispiele

Die folgenden Beispiele geben Ergebnisse für eine im Schema benannte Ansicht `public` und eine `SALES_VW` im Schema benannte `sales` Tabellentabelle zurück`mytickit1`, die vom Benutzer in der verbundenen Datenbank erstellt wurden`dev`.

Das folgende Beispiel gibt die Spaltenmetadaten für eine Ansicht mit dem Namen zurückSALES_VW.

```
select pg_get_cols('sales_vw');

pg_get_cols
---------------------------------------------------------------
(public,sales_vw,salesid,integer,1)
(public,sales_vw,listid,integer,2)
(public,sales_vw,sellerid,integer,3)
(public,sales_vw,buyerid,integer,4)
(public,sales_vw,eventid,integer,5)
(public,sales_vw,dateid,smallint,6)
(public,sales_vw,qtysold,smallint,7)
(public,sales_vw,pricepaid,"numeric(8,2)",8)
(public,sales_vw,commission,"numeric(8,2)",9)
(public,sales_vw,saletime,"timestamp without time zone",10)
```

Im folgenden Beispiel werden die Spaltenmetadaten für die SALES_VW Ansicht im Tabellenformat zurückgegeben.

```
select * from pg_get_cols('sales_vw')
cols(view_schema name, view_name name, col_name name, col_type varchar, col_num int);

view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
public      | sales_vw  | salesid    | integer                     |       1
public      | sales_vw  | listid     | integer                     |       2
public      | sales_vw  | sellerid   | integer                     |       3
public      | sales_vw  | buyerid    | integer                     |       4
public      | sales_vw  | eventid    | integer                     |       5
public      | sales_vw  | dateid     | smallint                    |       6
public      | sales_vw  | qtysold    | smallint                    |       7
public      | sales_vw  | pricepaid  | numeric(8,2)                |       8
public      | sales_vw  | commission | numeric(8,2)                |       9
public      | sales_vw  | saletime   | timestamp without time zone |      10
```

Im folgenden Beispiel werden die Spaltenmetadaten für die SALES Tabelle im Schema mytickit1 im Tabellenformat zurückgegeben.

```
select * from pg_get_cols('"mytickit1"."sales"')
cols(view_schema name, view_name name, col_name name, col_type varchar, col_num int);
```

```
view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
mytickit1   | sales     | salesid    | integer                     |       1
mytickit1   | sales     | listid     | integer                     |       2
mytickit1   | sales     | sellerid   | integer                     |       3
mytickit1   | sales     | buyerid    | integer                     |       4
mytickit1   | sales     | eventid    | integer                     |       5
mytickit1   | sales     | dateid     | smallint                    |       6
mytickit1   | sales     | qtysold    | smallint                    |       7
mytickit1   | sales     | pricepaid  | numeric(8,2)                |       8
mytickit1   | sales     | commission | numeric(8,2)                |       9
mytickit1   | sales     | saletime   | timestamp without time zone |      10
```

PG_GET_GRANTEE_BY_IAM_ROLE

Gibt alle Benutzer und Gruppen zurück, denen eine angegebene IAM-Rolle gewährt wurde.

Syntax

```
pg_get_grantee_by_iam_role('iam_role_arn')
```

Argumente

iam_role_arn

Die IAM-Rolle, für die die Benutzer und Gruppen zurückgegeben werden sollen, denen diese Rolle erteilt wurde.

Rückgabetyp

VARCHAR

Nutzungshinweise

Die Funktion PG_GET_GRANTEE_BY_IAM_ROLE gibt für jeden Benutzer oder jede Gruppe eine Zeile zurück. Jede Zeile enthält den Berechtigungsempfängernamen, den Berechtigungsempfängertyp und die gewährte Berechtigung. Die möglichen Werte für den Berechtigungsempfängertyp sind p für public (öffentlich), u für user (Benutzer) und g für group (Gruppe).

Sie müssen Superuser sein, um diese Funktion zu verwenden.

Beispiel

Das folgende Beispiel zeigt, dass `group1` und `reg_user1` die IAM-Rolle `Redshift-S3-Write` gewährt wird. Benutzer in `group_1` können angeben, dass die Rolle nur COPY-Vorgänge durchführen soll. `reg_user1`-Benutzer können angeben, dass die Rolle nur UNLOAD-Vorgänge durchführen soll.

```
select pg_get_grantee_by_iam_role('arn:aws:iam::123456789012:role/Redshift-S3-Write');
```

```
  pg_get_grantee_by_iam_role
-----------------------------
 (group_1,g,COPY)
 (reg_user1,u,UNLOAD)
```

Im folgenden Beispiel einer PG_GET_GRANTEE_BY_IAM_ROLE-Funktion wird das Ergebnis als Tabelle formatiert.

```
select grantee, grantee_type, cmd_type FROM
 pg_get_grantee_by_iam_role('arn:aws:iam::123456789012:role/Redshift-S3-Write')
 res_grantee(grantee text, grantee_type text, cmd_type text) ORDER BY 1,2,3;
```

```
  grantee  | grantee_type | cmd_type
-----------+--------------+----------
 group_1   | g            | COPY
 reg_user1 | u            | UNLOAD
```

PG_GET_IAM_ROLE_BY_USER

Gibt alle IAM-Rollen und Befehlsberechtigungen zurück, die einem Benutzer gewährt wurden.

Syntax

```
pg_get_iam_role_by_user('name')
```

Argumente

Name

Der Name des Benutzers, für den IAM-Rollen zurückgegeben werden sollen.

Rückgabetyp

VARCHAR

Nutzungshinweise

Die Funktion PG_GET_IAM_ROLE_BY_USER gibt eine Zeile für jede Gruppe von Rollen und Befehlsberechtigungen zurück. Die Zeile enthält eine durch Kommas getrennte Liste mit dem Benutzernamen, der IAM-Rolle und dem Befehl.

Ein `default`-Wert im Ergebnis zeigt an, dass der Benutzer eine beliebige verfügbare Rolle angeben kann, um den angezeigten Befehl auszuführen.

Sie müssen Superuser sein, um diese Funktion zu verwenden.

Beispiel

Das folgende Beispiel zeigt, dass der Benutzer `reg_user1` jede verfügbare IAM-Rolle angeben kann, um COPY-Vorgänge durchzuführen. Der Benutzer kann außerdem die `Redshift-S3-Write`-Rolle für die UNLOAD-Vorgänge angeben.

```
select pg_get_iam_role_by_user('reg_user1');
```

```
                            pg_get_iam_role_by_user
-------------------------------------------------------------------------------
 (reg_user1,default,COPY)
 (reg_user1,arn:aws:iam::123456789012:role/Redshift-S3-Write,COPY|UNLOAD)
```

Im folgenden Beispiel einer PG_GET_IAM_ROLE_BY_USER-Funktion wird das Ergebnis als Tabelle formatiert.

```
select username, iam_role, cmd FROM pg_get_iam_role_by_user('reg_user1')
 res_iam_role(username text, iam_role text, cmd text);
```

```
 username  |                    iam_role                     | cmd
-----------+-------------------------------------------------+------
 reg_user1 | default                                         | None
 reg_user1 | arn:aws:iam::123456789012:role/Redshift-S3-Read | COPY
```

PG_GET_LATE_BINDING_VIEW_COLS

Gibt die Spaltenmetadaten für alle Ansichten mit später Bindung in der Datenbank zurück. Weitere Informationen finden Sie unter Ansichten mit später Bindung

Syntax

```
pg_get_late_binding_view_cols()
```

Rückgabetyp

VARCHAR

Nutzungshinweise

Die PG_GET_LATE_BINDING_VIEW_COLS-Funktion gibt eine Zeile für jede Spalte in den Ansichten mit später Bindung zurück. Die Zeile enthält eine durch Komma getrennte Liste mit dem Schemanamen, dem Beziehungsnamen, dem Spaltennamen, dem Datentyp und der Spaltennummer.

Beispiel

Im folgenden Beispiel werden die Spaltenmetadaten für alle Ansichten mit später Bindung zurückgegeben.

```
select pg_get_late_binding_view_cols();

pg_get_late_binding_view_cols
------------------------------------------------------------
(public,myevent,eventname,"character varying(200)",1)
(public,sales_lbv,salesid,integer,1)
(public,sales_lbv,listid,integer,2)
(public,sales_lbv,sellerid,integer,3)
(public,sales_lbv,buyerid,integer,4)
(public,sales_lbv,eventid,integer,5)
(public,sales_lbv,dateid,smallint,6)
(public,sales_lbv,qtysold,smallint,7)
(public,sales_lbv,pricepaid,"numeric(8,2)",8)
(public,sales_lbv,commission,"numeric(8,2)",9)
(public,sales_lbv,saletime,"timestamp without time zone",10)
(public,event_lbv,eventid,integer,1)
```

```
(public,event_lbv,venueid,smallint,2)
(public,event_lbv,catid,smallint,3)
(public,event_lbv,dateid,smallint,4)
(public,event_lbv,eventname,"character varying(200)",5)
(public,event_lbv,starttime,"timestamp without time zone",6)
```

Im folgenden Beispiel werden die Spaltenmetadaten für alle Ansichten mit später Bindung im Tabellenformat zurückgegeben.

```
select * from pg_get_late_binding_view_cols() cols(view_schema name, view_name name,
 col_name name, col_type varchar, col_num int);
view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
public      | sales_lbv | salesid    | integer                     |       1
public      | sales_lbv | listid     | integer                     |       2
public      | sales_lbv | sellerid   | integer                     |       3
public      | sales_lbv | buyerid    | integer                     |       4
public      | sales_lbv | eventid    | integer                     |       5
public      | sales_lbv | dateid     | smallint                    |       6
public      | sales_lbv | qtysold    | smallint                    |       7
public      | sales_lbv | pricepaid  | numeric(8,2)                |       8
public      | sales_lbv | commission | numeric(8,2)                |       9
public      | sales_lbv | saletime   | timestamp without time zone |      10
public      | event_lbv | eventid    | integer                     |       1
public      | event_lbv | venueid    | smallint                    |       2
public      | event_lbv | catid      | smallint                    |       3
public      | event_lbv | dateid     | smallint                    |       4
public      | event_lbv | eventname  | character varying(200)      |       5
public      | event_lbv | starttime  | timestamp without time zone |       6
```

PG_GET_SESSION_ROLES

Gibt die Sitzungsrollen des aktuell angemeldeten Benutzers zurück. Sitzungsrollen eines Benutzers sind die Gruppen, die von einem Identitätsanbieter (IdP) für den angemeldeten Benutzer definiert wurden. Zum Beispiel überprüft ein Identitätsanbieter (IdP) wie Microsoft Azure Active Directory (Azure AD) die Identität des Benutzers und gibt bei der Benutzeranmeldung alle externen Gruppen an, denen der Benutzer angehört. Diese externen Gruppen werden in Amazon-Redshift-Rollen umgewandelt und sind während der aktuellen Sitzung verfügbar. Diese Rollen werden als Sitzungsrollen bezeichnet. Ein Administrator kann einer Sitzungsrolle ähnlich wie bei anderen Amazon-Redshift-Rollen Rechte gewähren. Weitere Informationen zur Nutzung von Rollen finden Sie unter Rollenbasierte Zugriffskontrolle (RBAC). Informationen zur Verwaltung von Identitäten mit

einem Identitätsanbieter (IdP) finden Sie unter Nativer Identitätsanbieter(IdP)-Verbund für Amazon Redshift im Leitfaden zur Verwaltung von Amazon Redshift.

Wenn Sie die im Amazon-Redshift-Katalog definierten Rollen anzeigen möchten, stellen Sie als Administrator oder Superuser eine Verbindung zur Datenbank her und fragen Sie die Systemansicht SVV_ROLES ab.

Syntax

```
pg_get_session_roles()
```

Rückgabetyp

Ein Zeilensatz, der aus zwei Werten besteht. Der erste Wert besteht aus zwei Teilen, die durch einen Doppelpunkt (:) getrennt sind, der einen `idp-namespace:role-name` enthält. Der `idp-namespace` ist der Namespace des Identitätsanbieters (IdP). Der `role-name` ist der Name der externen Gruppe im Identitätsanbieter (IdP). Der zweite Wert enthält eine `role-id`. Dies ist die Rollenkennung.

Nutzungshinweise

Die `PG_GET_SESSION_ROLES`-Funktion gibt für jede zurückgegebene Sitzungsrolle eine Zeile zurück.

Beispiele

Im folgenden Beispiel wird eine Zeile für jede Rolle aus dem Azure-Active-Directory-IdP zurückgegeben. Die zurückgegebenen Spalten werden in `sess_roles` mit den Spalten `name` und `roleid` umgewandelt. Jeder `name` besteht aus dem Azure-Active-Directory-Namespace und einem Gruppennamen in Azure Active Directory.

```
SELECT * FROM pg_get_session_roles() AS sess_roles(name name, roleid integer);

name                   roleid
--------------------------------
my_aad:test_group_1   106204
my_aad:test_group_2   106205
my_aad:test_group_3   106206
my_aad:test_group_4   106207
my_aad:test_group_5   106208
```

Im folgenden Beispiel wird eine Zeile für jede IAM-Gruppe zurückgegeben, der der aktuell angemeldete IAM-Benutzer angehört. Die zurückgegebenen Spalten werden in `sess_roles` mit den Spalten `name` und `roleid` umgewandelt. Jeder `name` besteht aus dem IAM-Namespace und dem IAM-Gruppennamen.

```
SELECT * FROM pg_get_session_roles() AS sess_roles(name name, roleid integer);

name                  roleid
-------------------------------
IAM:myGroup           110332
```

PG_LAST_COPY_COUNT

Gibt die Anzahl der Zeilen zurück, die vom letzten COPY-Befehl geladen wurden, der in der aktuellen Sitzung ausgeführt wurde. PG_LAST_COPY_COUNT wird mit der letzten COPY ID aktualisiert. Dies ist die Abfrage-ID der letzten COPY-Operation, die den Ladeprozess gestartet hat, auch wenn der Ladeprozess fehlgeschlagen ist. Die Abfrage-ID und COPY ID werden aktualisiert, wenn der Befehl COPY den Ladeprozess startet.

Wenn COPY aufgrund eines Syntaxfehlers oder unzureichender Rechte fehlschlägt, wird COPY ID nicht aktualisiert und PG_LAST_COPY_COUNT gibt die Zahl für die vorherige COPY-Operation zurück. Wenn in der aktuellen Sitzung keine COPY-Befehle ausgeführt wurden oder der letzte COPY-Befehl während des Ladens fehlgeschlagen ist, gibt PG_LAST_COPY_COUNT 0 zurück. Weitere Informationen finden Sie unter PG_LAST_COPY_ID.

Syntax

```
pg_last_copy_count()
```

Rückgabetyp

Gibt BIGINT zurück.

Beispiel

Die folgende Abfrage gibt die Anzahl der Zeilen zurück, die vom letzten COPY-Befehl in der aktuellen Sitzung geladen wurden.

```
select pg_last_copy_count();
```

```
pg_last_copy_count
--------------------
             192497
(1 row)
```

PG_LAST_COPY_ID

Gibt die Abfrage-ID des in der aktuellen Sitzung zuletzt abgeschlossenen COPY-Befehls zurück. Wenn in der aktuellen Sitzung keine COPY-Befehle ausgeführt wurden, gibt PG_LAST_COPY_ID -1 zurück.

Der Wert für PG_LAST_COPY_ID wird aktualisiert, wenn der Befehl COPY den Ladeprozess startet. Wenn COPY aufgrund ungültiger Ladedaten fehlschlägt, wird die COPY ID aktualisiert, damit Sie PG_LAST_COPY_ID verwenden können, wenn Sie eine Abfrage für die Tabelle STL_LOAD_ERRORS ausführen. Wenn für die COPY-Transaktion ein Rollback ausgeführt wird, wird die COPY ID nicht aktualisiert.

Die COPY ID wird nicht aktualisiert, wenn der COPY-Befehl aufgrund eines Fehlers fehlschlägt, der vor dem Start des Ladeprozesses auftritt, beispielsweise aufgrund von Syntaxfehlern, Zugriffsfehlern, ungültiger Anmeldeinformationen oder unzureichender Rechte. Die COPY ID wird nicht aktualisiert, wenn COPY während des Analysekompressionsschritts fehlschlägt, der nach der erfolgreichen Herstellung einer Verbindung, jedoch vor dem Laden von Daten gestartet wird.

Syntax

```
pg_last_copy_id()
```

Rückgabetyp

Gibt eine Ganzzahl zurück.

Beispiel

Die folgende Abfrage gibt die Abfrage-ID des letzten COPY-Befehls in der aktuellen Sitzung zurück.

```
select pg_last_copy_id();

pg_last_copy_id
---------------
```

```
          5437
(1 row)
```

Die folgende Abfrage verbindet STL_LOAD_ERRORS mit STL_LOADERROR_DETAIL zur Anzeige der Details der Fehler, die in der aktuellen Sitzung während des letzten Ladevorgangs aufgetreten sind:

```
select d.query, substring(d.filename,14,20),
d.line_number as line,
substring(d.value,1,16) as value,
substring(le.err_reason,1,48) as err_reason
from stl_loaderror_detail d, stl_load_errors le
where d.query = le.query
and d.query = pg_last_copy_id();

 query |    substring      | line |  value   |                    err_reason
-------+-------------------+------+----------
+-----------------------------------------------
    558| allusers_pipe.txt |  251 | 251      | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | ZRU29FGR | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | Kaitlin  | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | Walter   | String contains invalid or unsupported
 UTF8 code
```

PG_LAST_UNLOAD_ID

Gibt die Abfrage-ID des in der aktuellen Sitzung zuletzt abgeschlossenen UNLOAD-Befehls zurück. Wenn in der aktuellen Sitzung keine UNLOAD-Befehle ausgeführt wurden, gibt PG_LAST_UNLOAD_ID den Wert -1 zurück.

Der Wert für PG_LAST_UNLOAD_ID wird aktualisiert, wenn der UNLOAD-Befehl den Ladeprozess startet. Wenn der UNLOAD-Befehl aufgrund ungültiger Ladedaten fehlschlägt, wird die UNLOAD ID aktualisiert, sodass Sie die UNLOAD ID für weitere Untersuchungen verwenden können. Wenn für die UNLOAD-Transaktion ein Rollback ausgeführt wird, wird die UNLOAD ID nicht aktualisiert.

Die UNLOAD ID wird nicht aktualisiert, wenn der UNLOAD-Befehl aufgrund eines Fehlers fehlschlägt, der vor dem Start des Ladeprozesses auftritt, beispielsweise aufgrund von Syntaxfehlern, Zugriffsfehlern, ungültiger Anmeldeinformationen oder unzureichender Rechte.

Syntax

```
PG_LAST_UNLOAD_ID()
```

Rückgabetyp

Gibt eine Ganzzahl zurück.

Beispiel

Die folgende Abfrage gibt die Abfrage-ID des letzten UNLOAD-Befehls in der aktuellen Sitzung zurück.

```
select PG_LAST_UNLOAD_ID();

PG_LAST_UNLOAD_ID
---------------
          5437
(1 row)
```

PG_LAST_QUERY_ID

Gibt die Abfrage-ID der in der aktuellen Sitzung zuletzt abgeschlossenen Abfrage zurück. Wenn in der aktuellen Sitzung keine Abfragen ausgeführt wurden, gibt PG_LAST_QUERY_ID -1 zurück. PG_LAST_QUERY_ID gibt keine Abfrage-IDs für Abfragen zurück, die ausschließlich auf dem Führungsknoten ausgeführt werden. Weitere Informationen finden Sie unter Exklusive Führungsknotenfunktionen.

Syntax

```
pg_last_query_id()
```

Rückgabetyp

Gibt eine Ganzzahl zurück.

Beispiel

Die folgende Abfrage gibt die ID der letzten Abfrage in der aktuellen Sitzung zurück.

```
select pg_last_query_id();
```

Die Ergebnisse sehen wie folgt aus.

```
pg_last_query_id
----------------
           5437
(1 row)
```

Die folgende Abfrage gibt die Abfrage-ID und den Text der in der aktuellen Sitzung zuletzt abgeschlossenen Abfrage aus.

```
select query, trim(querytxt) as sqlquery
from stl_query
where query = pg_last_query_id();
```

Die Ergebnisse sehen wie folgt aus.

```
query | sqlquery
------+-------------------------------------------------
 5437 | select name, loadtime from stl_file_scan where loadtime > 1000000;
(1 rows)
```

PG_LAST_UNLOAD_COUNT

Gibt die Anzahl der Zeilen zurück, die vom letzten UNLOAD-Befehl entladen wurden, der in der aktuellen Sitzung abgeschlossen wurde. PG_LAST_UNLOAD_COUNT wird mit der Abfrage-ID der letzten UNLOAD-Operation aktualisiert, auch wenn die Operation fehlgeschlagen ist. Die Abfrage-ID wird aktualisiert, wenn UNLOAD abgeschlossen wird. Wenn UNLOAD aufgrund eines Syntaxfehlers oder unzureichender Rechte fehlschlägt, gibt PG_LAST_UNLOAD_COUNT die Zahl für die vorherige UNLOAD-Operation zurück. Wenn in der aktuellen Sitzung keine UNLOAD-Befehle abgeschlossenen wurden oder der letzte UNLOAD-Befehl während der Entladeoperation fehlgeschlagen ist, gibt PG_LAST_UNLOAD_COUNT 0 zurück.

Syntax

```
pg_last_unload_count()
```

Rückgabetyp

Gibt BIGINT zurück.

Beispiel

Die folgende Abfrage gibt die Anzahl der Zeilen zurück, die vom letzten UNLOAD-Befehl in der aktuellen Sitzung entladen wurden.

```
select pg_last_unload_count();

pg_last_unload_count
--------------------
             192497
(1 row)
```

Die Funktion SLICE_NUM

Gibt eine Ganzzahl zurück, die der Slice-Nummer in dem Cluster entspricht, in dem sich die Daten für eine Zeile befinden. SLICE_NUM verwendet keine Parameter.

Syntax

```
SLICE_NUM()
```

Rückgabetyp

Die SLICE_NUM-Funktion gibt eine Ganzzahl zurück.

Beispiele

In dem folgenden Beispiel wird gezeigt, welche Slices Daten für die ersten zehn EVENT-Zeilen in der Tabelle EVENTS enthalten:

```
select distinct eventid, slice_num() from event order by eventid limit 10;

 eventid | slice_num
---------+-----------
       1 |         1
       2 |         2
       3 |         3
       4 |         0
       5 |         1
       6 |         2
       7 |         3
       8 |         0
       9 |         1
```

```
      10 |         2
(10 rows)
```

Im folgenden Beispiel wird ein Code (10000) zurückgegeben, um zu zeigen, dass auf dem Führungsknoten eine Abfrage ohne FROM-Anweisung ausgeführt wird:

```
select slice_num();
slice_num
-----------
10000
(1 row)
```

USER

Synonym mit CURRENT_USER. Siehe CURRENT_USER.

ROLE_IS_MEMBER_OF

Gibt „true“ zurück, wenn die Rolle Mitglied einer anderen Rolle ist. Superuser können die Mitgliedschaft aller Rollen überprüfen. Reguläre Benutzer mit der Berechtigung ACCESS SYSTEM TABLE können die Mitgliedschaft aller Benutzer überprüfen. Andere reguläre Benutzer können nur Rollen überprüfen, auf die sie Zugriff haben. Amazon Redshift gibt Fehler aus, wenn die bereitgestellten Rollen nicht vorhanden sind oder der aktuelle Benutzer keinen Zugriff auf die Rolle hat.

Syntax

```
role_is_member_of( role_name,  granted_role_name)
```

Argumente

rollen_name

Der Name der Rolle.

granted_role_name

Der Name der gewährten Rolle.

Rückgabetyp

Gibt einen booleschen Wert zurück.

Beispiel

Die folgende Abfrage bestätigt, dass die Rolle kein Mitglied von „role1“ oder „role2“ ist.

```
SELECT role_is_member_of('role1', 'role2');

 role_is_member_of
-------------------
             False
```

USER_IS_MEMBER_OF

Gibt „true“ zurück, wenn der Benutzer ein Mitglied einer Rolle oder Gruppe ist. Superuser können die Mitgliedschaft aller Benutzer überprüfen. Reguläre Benutzer, die Mitglieder der Rolle „sys:secadmin“ oder „sys:superuser“ sind, können die Mitgliedschaft aller Benutzer überprüfen. Andere reguläre Benutzer können nur sich selbst überprüfen. Amazon Redshift gibt Fehler aus, wenn die bereitgestellten Identitäten nicht vorhanden sind oder der aktuelle Benutzer keinen Zugriff auf die Rolle hat.

Syntax

```
user_is_member_of( user_name,  role_name | group_name)
```

Argumente

user_name

Der Name des Benutzers.

rollen_name

Der Name der Rolle.

group_name

Der Name der Gruppe.

Rückgabetyp

Gibt einen booleschen Wert zurück.

Beispiel

Die folgende Abfrage bestätigt, dass der Benutzer kein Mitglied von „Role1“ ist.

```
SELECT user_is_member_of('reguser', 'role1');

 user_is_member_of
-------------------
           False
```

VERSION

Die VERSION-Funktion gibt Details zur aktuell installierten Version zurück. Die Informationen zur spezifischen Amazon-Redshift-Version befinden sich am Ende.

> **Note**
>
> Dies ist eine Führungsknotenfunktion. Diese Funktion gibt einen Fehler zurück, wenn sie eine benutzererstellte Tabelle, eine STL- oder STV-Systemtabelle oder eine SVV- oder SVL-Systemansicht referenziert.

Syntax

```
VERSION()
```

Rückgabetyp

Gibt eine CHAR- oder VARCHAR-Zeichenfolge zurück.

Beispiele

Das folgende Beispiel zeigt die Cluster-Versionsinformationen des aktuellen Clusters:

```
select version();
```

```
 version
----------------------------------------------------------------------------------------------

PostgreSQL 8.0.2 on i686-pc-linux-gnu, compiled by GCC gcc (GCC) 3.4.2 20041017 (Red
Hat 3.4.2-6.fc3), Redshift 1.0.12103
```

Dabei ist `1.0.12103` die Cluster-Versionsnummer.

> **Note**
>
> Um Ihren Cluster zur Aktualisierung auf die neueste Clusterversion zu zwingen, passen Sie Ihr Wartungsfenster an.

Reservierte Wörter

Die folgende Liste enthält die reservierten Wörter für Amazon Redshift. Sie können die reservierten Wörter mit abgegrenzten Kennungen (doppelten Anführungszeichen) verwenden.

> **Note**
>
> START und CONNECT sind zwar keine reservierten Wörter, Sie sollten jedoch abgegrenzte Kennungen oder AS verwenden, wenn Sie START und CONNECT als Tabellenaliasse in Ihrer Abfrage verwenden, um Fehler während der Laufzeit zu vermeiden.

Weitere Informationen finden Sie unter Namen und Kennungen.

```
AES128
AES256
ALL
ALLOWOVERWRITE
ANALYSE
ANALYZE
AND
ANY
ARRAY
AS
ASC
AUTHORIZATION
AZ64
BACKUP
BETWEEN
BINARY
BLANKSASNULL
BOTH
BYTEDICT
```

```
BZIP2
CASE
CAST
CHECK
COLLATE
COLUMN
CONSTRAINT
CREATE
CREDENTIALS
CROSS
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURRENT_USER
CURRENT_USER_ID
DEFAULT
DEFERRABLE
DEFLATE
DEFRAG
DELTA
DELTA32K
DESC
DISABLE
DISTINCT
DO
ELSE
EMPTYASNULL
ENABLE
ENCODE
ENCRYPT
ENCRYPTION
END
EXCEPT
EXPLICIT
FALSE
FOR
FOREIGN
FREEZE
FROM
FULL
GLOBALDICT256
GLOBALDICT64K
GRANT
GROUP
```

```
GZIP
HAVING
IDENTITY
IGNORE
ILIKE
IN
INITIALLY
INNER
INTERSECT
INTERVAL
INTO
IS
ISNULL
JOIN
LEADING
LEFT
LIKE
LIMIT
LOCALTIME
LOCALTIMESTAMP
LUN
LUNS
LZO
LZOP
MINUS
MOSTLY16
MOSTLY32
MOSTLY8
NATURAL
NEW
NOT
NOTNULL
NULL
NULLS
OFF
OFFLINE
OFFSET
OID
OLD
ON
ONLY
OPEN
OR
ORDER
```

```
OUTER
OVERLAPS
PARALLEL
PARTITION
PERCENT
PERMISSIONS
PIVOT
PLACING
PRIMARY
RAW
READRATIO
RECOVER
REFERENCES
REJECTLOG
RESORT
RESPECT
RESTORE
RIGHT
SELECT
SESSION_USER
SIMILAR
SNAPSHOT
SOME
SYSDATE
SYSTEM
TABLE
TAG
TDES
TEXT255
TEXT32K
THEN
TIMESTAMP
TO
TOP
TRAILING
TRUE
TRUNCATECOLUMNS
UNION
UNIQUE
UNNEST
UNPIVOT
USER
USING
VERBOSE
```

```
WALLET
WHEN
WHERE
WITH
WITHOUT
```

Referenz zu Systemtabellen und Ansichten

Themen

Systemtabellen und Ansichten

Amazon Redshift verfügt über zahlreiche Systemtabellen und Ansichten, die Informationen zur Funktionsweise des Systems enthalten. Sie können diese Systemtabellen und Ansichten genauso abfragen wie andere Datenbanktabellen auch. Dieser Abschnitt illustriert einige Beispiele für Systemtabellenabfragen und erläutert:

- Wie unterschiedliche Arten von Systemtabellen und Ansichten generiert werden
- Welche Arten von Informationen Sie aus diesen Tabellen erhalten können
- Wie Sie Amazon-Redshift-Systemtabellen mit Katalogtabellen verbinden
- Wie Sie mit der Zunahme der Systemtabellenprotokolldateien umgehen

Einige Systemtabellen können nur von AWS Mitarbeitern zu Diagnosezwecken verwendet werden. Die folgenden Abschnitte behandeln die Systemtabellen, die Systemadministratoren oder andere Datenbankbenutzer abfragen können, um nützliche Informationen zu erhalten.

> **Note**
>
> Systemtabellen werden in automatisierten oder manuellen Cluster-Backups (Snapshots) nicht berücksichtigt. STL-Systemansichten speichern den Protokollverlauf von sieben Tagen. Die Aufbewahrung von Protokollen erfordert keine Maßnahmen des Kunden. Wenn Sie Protokolldaten jedoch länger als 7 Tage aufbewahren möchten, müssen Sie sie regelmäßig in andere Tabellen kopieren oder in Amazon S3 entladen.

Arten von Systemtabellen und Ansichten

Es gibt mehrere Arten von Systemtabellen und Ansichten:

- SVV-Ansichten enthalten Informationen über Datenbankobjekte mit Verweisen auf transiente STV-Tabellen.
- SYS-Ansichten werden zur Überwachung der Abfrage- und Workloadnutzung für bereitgestellte Cluster und Serverless-Arbeitsgruppen verwendet.
- STL-Ansichten werden aus Protokollen generiert, die dauerhaft auf der Festplatte gespeichert werden, um einen Systemverlauf bereitzustellen.
- STV-Tabellen sind virtuelle Systemtabellen, die Snapshots der aktuellen Systemdaten enthalten. Sie basieren auf flüchtigen Arbeitsspeicherdaten und werden nicht dauerhaft in Protokollen oder regulären Tabellen gespeichert.
- SVCS-Ansichten enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern.
- SVL-Ansichten stellen Informationen zu Abfragen auf Haupt-Clustern bereit.

Systemtabellen und Ansichten verwenden nicht dasselbe Konsistenzmodell wie reguläre Tabellen. Dies sollten sich bei ihrer Abfrage stets bewusst machen, besonders für STV-Tabellen und SVV-Ansichten. Zum Beispiel: Beim Vorliegen einer regulären Tabelle t1 mit einer Spalte c1 würden Sie erwarten, dass die folgende Abfrage keine Zeilen zurückgibt:

```
select * from t1
where c1 > (select max(c1) from t1)
```

Die folgende Abfrage gegen eine Systemtabelle kann aber sehr wohl Zeilen ausgeben:

```
select * from stv_exec_state
where currenttime > (select max(currenttime) from stv_exec_state)
```

Der Grund dafür ist der, dass currenttime ein temporärer Wert ist und die beiden Referenzen in der Abfrage bei der Evaluierung möglicherweise nicht den gleichen Wert ausgeben.

Andererseits kann die folgende Abfrage möglicherweise auch keine Zeilen ausgeben:

```
select * from stv_exec_state
where currenttime = (select max(currenttime) from stv_exec_state)
```

Sichtbarkeit der Daten in Systemtabellen und Ansichten

Es gibt zwei Klassen der Sichtbarkeit für Daten in Systemtabellen und Ansichten: für Benutzer sichtbar und für Superuser sichtbar.

Nur Benutzer mit Superuser-Berechtigungen können Daten in Tabellen der Kategorie "sichtbar für Superuser" anzeigen. Normale Benutzer sehen Nur die Daten in für Benutzer sichtbaren Tabellen. Um einem normalen Benutzer Zugriff auf Tabellen zu gewähren, die für Superuser sichtbar sind, gewähren Sie dem normalen Benutzer das SELECT-Privileg für diese Tabelle. Weitere Informationen finden Sie unter GRANT.

Standardmäßig sind in den meisten für Benutzer sichtbaren Tabellen von einem anderen Benutzer generierte Zeilen für einen normalen Benutzer nicht sichtbar. Wenn einem normalen Benutzer SYSLOG ACCESS UNRESTRICTED gewährt wird, kann dieser Benutzer alle Zeilen in für den Benutzer sichtbaren Tabellen sehen, einschließlich der Zeilen, die von einem anderen Benutzer generiert wurden. Weitere Informationen finden Sie unter ALTER USER oder CREATE USER. Alle Zeilen in SVV_TRANSACTIONS sind für alle Benutzer sichtbar. Weitere Informationen zur Sichtbarkeit von Daten finden Sie im AWS re:Post Knowledgebase-Artikel Wie kann ich regulären Benutzern der Amazon Redshift Redshift-Datenbank die Erlaubnis erteilen, Daten in Systemtabellen von anderen Benutzern für meinen Cluster anzuzeigen? .

Für Metadatenansichten gewährt Amazon Redshift Benutzern, denen SYSLOG ACCESS UNRESTRICTED gewährt wurde, keine Sichtbarkeit.

> **Note**
>
> Wenn Sie einem Benutzer uneingeschränkten Zugriff auf Systemtabellen gewähren, sieht der Benutzer auch Daten, die von anderen Benutzern generiert wurden. STL_QUERY und

STL_QUERY_TEXT enthalten beispielsweise den vollständigen Text von INSERT-, UPDATE- und DELETE-Anweisungen, die möglicherweise sensible von Benutzern generierte Daten enthalten.

Ein Superuser kann alle Zeilen in allen Tabellen sehen. Um einem normalen Benutzer Zugriff auf Tabellen zu gewähren, die nur für Superuser sichtbar sind, gewähren Sie dem normalen Benutzer die GRANT SELECT-Berechtigung für diese Tabelle.

Filterung systemgenerierter Abfragen

Die abfragebezogenen Systemtabellen und Ansichten, wie etwa SVL_QUERY_SUMMARY, SVL_QLOG und andere, enthalten normalerweise eine sehr große Zahl automatisch generierter Anweisungen, die Amazon Redshift für die Überwachung des Status der Datenbank verwenden. Diese systemgenerierten Abfragen sind für Superuser sichtbar, jedoch selten wirklich nützlich. Um diese bei der Auswahl aus einer Systemtabelle oder Systemtabelle auszufiltern, die die Spalte `userid` verwendet, fügen Sie die Bedingung `userid > 1` der WHERE-Klausel hinzu. Beispielsweise:

```
select * from svl_query_summary where userid > 1
```

Migration von Abfragen, die nur bereitgestellt wurden, zu Abfragen mit SYS-Überwachungsansichten

Migrieren von bereitgestellten Clustern zu Amazon Redshift Serverless

Wenn Sie einen bereitgestellten Cluster zu Amazon Redshift Serverless migrieren, haben Sie möglicherweise Abfragen, die die folgenden Systemansichten verwenden, in denen nur Daten aus bereitgestellten Clustern gespeichert werden.

- Alle STL-Ansichten
- Alle STV-Ansichten
- Alle SVCS-Ansichten
- Alle SVL-Ansichten
- Einige SVV-Ansichten

- Eine vollständige Liste der SVV-Ansichten, die in Amazon Redshift Serverless nicht unterstützt werden, finden Sie in der Liste am Ende von Monitoring queries and workloads with Amazon Redshift Serverless im Amazon Redshift Management Guide.

Um Ihre Abfragen weiterhin zu benutzen, passen Sie sie so an, dass sie Spalten verwenden, die in den SYS-Überwachungsansichten definiert sind und den Spalten in Ihren nur bereitgestellten Ansichten entsprechen. Um die Zuordnungsbeziehung zwischen den nur bereitgestellten Ansichten und den SYS-Überwachungsansichten zu sehen, gehen Sie zu Systemansichtszuordnung für die Migration zu SYS-Überwachungsansichten

Aktualisieren von Abfragen in einem bereitgestellten Cluster

Wenn Sie nicht zu Amazon Redshift Serverless migrieren, möchten Sie möglicherweise trotzdem Ihre vorhandenen Abfragen aktualisieren. Die SYS-Überwachungsansichten sind auf Benutzerfreundlichkeit und reduzierte Komplexität ausgelegt und bieten eine vollständige Palette von Metriken für eine effektive Überwachung und Fehlerbehebung. Mithilfe von SYS-Ansichten wie SYS_QUERY_HISTORY und SYS_QUERY_DETAIL, die die Informationen mehrerer nur bereitgestellter Ansichten konsolidieren, können Sie Ihre Abfragen optimieren.

Verbesserung der Nachverfolgung von Abfrage-IDs mithilfe der SYS-Überwachungsansichten

SYS-Überwachungsansichten wie SYS_QUERY_HISTORY und SYS_QUERY_DETAIL enthalten die Spalte „query_id“ mit der ID der Abfragen der Benutzer. In ähnlicher Weise gibt es in nur bereitgestellten Ansichten wie STL_QUERY und SVL_QLOG die Spalte „query“, die ebenfalls die Abfrage-IDs enthält. Die in den SYS-Systemansichten aufgezeichneten Abfrage-IDs unterscheiden sich jedoch von den IDs, die in den nur bereitgestellten Ansichten aufgezeichnet werden.

Zwischen den Werten der Spalte „query_id“ in den SYS-Ansichten und den Werten der Spalte „query“ in den nur bereitgestellten Ansichten besteht folgender Unterschied:

- In SYS-Ansichten werden vom Benutzer eingereichte Abfragen in der Spalte „query_id“ in ihrer ursprünglichen Form aufgezeichnet. Der Amazon-Redshift-Optimierer kann sie zur Verbesserung der Leistung in untergeordnete Abfragen aufteilen, doch für eine einzelne Abfrage, die Sie ausführen, gibt es dennoch nur eine einzige Zeile in SYS_QUERY_HISTORY. Die einzelnen untergeordneten Abfragen können Sie in SYS_QUERY_DETAIL einsehen.

- In nur bereitgestellten Ansichten werden Abfragen in der Spalte „query“ auf der Ebene der untergeordneten Abfragen aufgezeichnet. Wenn der Amazon-Redshift-Optimierer Ihre ursprüngliche Abfrage in mehrere untergeordnete Abfragen unterteilt, gibt es in STL_QUERY mehrere Zeilen mit unterschiedlichen Abfrage-ID-Werten für eine einzelne Abfrage, die Sie ausführen.

Wenn Sie Ihre Überwachungs- und Diagnoseabfragen von nur bereitgestellten Ansichten zu SYS-Ansichten migrieren, sollten Sie diesen Unterschied berücksichtigen und Ihre Abfragen entsprechend bearbeiten. Weitere Informationen zur Verarbeitung von Abfragen durch Amazon Redshift finden Sie unter Workflow der Abfrageplanung und -ausführung.

Beispiel

Ein Beispiel dafür, wie Amazon Redshift Abfragen in den Überwachungsansichten „Nur bereitgestellt“ und „SYS“ unterschiedlich aufzeichnet, finden Sie in der folgenden Beispielabfrage. Diese Abfrage ist so geschrieben, wie Sie sie in Amazon Redshift ausführen würden.

```
SELECT
  s_name
  , COUNT(*) AS numwait
FROM
  supplier,
  lineitem l1,
  orders,
  nation
WHERE    s_suppkey = l1.l_suppkey
         AND o_orderkey = l1.l_orderkey
         AND o_orderstatus = 'F'
         AND l1.l_receiptdate > l1.l_commitdate
         AND EXISTS (SELECT
                       *
                     FROM
                       lineitem l2
                     WHERE  l2.l_orderkey = l1.l_orderkey
                            AND l2.l_suppkey <> l1.l_suppkey )
         AND NOT EXISTS (SELECT
                           *
                         FROM
                           lineitem l3
                         WHERE  l3.l_orderkey = l1.l_orderkey
                                AND l3.l_suppkey <> l1.l_suppkey
```

```
                                AND l3.l_receiptdate > l3.l_commitdate )
        AND s_nationkey = n_nationkey
        AND n_name = 'UNITED STATES'
GROUP BY
  s_name
ORDER BY
  numwait DESC
  , s_name LIMIT 100;
```

Im Hintergrund schreibt der Amazon-Redshift-Abfrageoptimierer die oben aufgeführte vom Benutzer eingereichte Abfrage in fünf untergeordnete Abfragen um.

Die erste untergeordnete Abfrage erstellt eine temporäre Tabelle, um eine Unterabfrage zu materialisieren.

```
CREATE TEMP TABLE volt_tt_606590308b512(l_orderkey
                                       , l_suppkey
                                       , s_name   ) AS SELECT
                                                     l1.l_orderkey
                                                     , l1.l_suppkey
                                                     , public.supplier.s_name
                                                   FROM
                                                     public.lineitem AS l1,
                                                     public.nation,
                                                     public.orders,
                                                     public.supplier
                                                   WHERE  l1.l_commitdate <
 l1.l_receiptdate
                                                          AND l1.l_orderkey =
 public.orders.o_orderkey
                                                          AND l1.l_suppkey =
 public.supplier.s_suppkey
                                                          AND public.nation.n_name
 = 'UNITED STATES'::CHAR(8)
                                                          AND
 public.nation.n_nationkey = public.supplier.s_nationkey
                                                          AND
 public.orders.o_orderstatus = 'F'::CHAR(1);
```

Die zweite untergeordnete Abfrage sammelt Statistiken aus der temporären Tabelle.

```
padb_fetch_sample: select count(*) from volt_tt_606590308b512;
```

Die dritte untergeordnete Abfrage erstellt eine weitere temporäre Tabelle, um eine weitere Unterabfrage zu materialisieren, die auf die oben erstellte temporäre Tabelle verweist.

```
CREATE TEMP TABLE volt_tt_606590308c2ef(l_orderkey
                                       , l_suppkey) AS (SELECT

 volt_tt_606590308b512.l_orderkey
                                                          ,
 volt_tt_606590308b512.l_suppkey
                                                        FROM
                                                          public.lineitem AS l2,
                                                          volt_tt_606590308b512
                                                        WHERE  l2.l_suppkey <>
 volt_tt_606590308b512.l_suppkey
                                                               AND l2.l_orderkey =
 volt_tt_606590308b512.l_orderkey)
                                                               EXCEPT distinct (SELECT
 volt_tt_606590308b512.l_orderkey, volt_tt_606590308b512.l_suppkey
                                                               FROM public.lineitem AS
 l3, volt_tt_606590308b512
                                                               WHERE l3.l_commitdate <
 l3.l_receiptdate
                                                                 AND l3.l_suppkey <>
 volt_tt_606590308b512.l_suppkey
                                                                 AND l3.l_orderkey =
 volt_tt_606590308b512.l_orderkey);
```

Die vierte untergeordnete Abfrage sammelt wiederum Statistiken aus der temporären Tabelle.

```
padb_fetch_sample: select count(*) from volt_tt_606590308c2ef
```

Die letzte untergeordnete Abfrage verwendet die oben erstellten temporären Tabellen, um die Ausgabe zu generieren.

```
SELECT
  volt_tt_606590308b512.s_name AS s_name
  , COUNT(*) AS numwait
FROM
  volt_tt_606590308b512,
  volt_tt_606590308c2ef
WHERE    volt_tt_606590308b512.l_orderkey = volt_tt_606590308c2ef.l_orderkey
         AND volt_tt_606590308b512.l_suppkey = volt_tt_606590308c2ef.l_suppkey
```

```
GROUP BY
  1
ORDER BY
  2 DESC
  , 1 ASC LIMIT 100;
```

In der nur bereitgestellten Systemansicht STL_QUERY zeichnet Amazon Redshift fünf Zeilen auf Ebene der untergeordneten Abfragen auf, wie im Folgenden dargestellt:

```
SELECT userid, xid, pid, query, querytxt::varchar(100);
FROM stl_query
WHERE xid = 48237350
ORDER BY xid, starttime;

 userid |   xid    |    pid     |  query   |
    querytxt
--------+----------+------------+----------
+-----------------------------------------------------------------------------------------------------------
    101 | 48237350 | 1073840810 | 12058151 | CREATE TEMP TABLE
 volt_tt_606590308b512(l_orderkey, l_suppkey, s_name) AS SELECT l1.l_orderkey, l1.l
    101 | 48237350 | 1073840810 | 12058152 | padb_fetch_sample: select count(*) from
 volt_tt_606590308b512
    101 | 48237350 | 1073840810 | 12058156 | CREATE TEMP TABLE
 volt_tt_606590308c2ef(l_orderkey, l_suppkey) AS (SELECT volt_tt_606590308b512.l_or
    101 | 48237350 | 1073840810 | 12058168 | padb_fetch_sample: select count(*) from
 volt_tt_606590308c2ef
    101 | 48237350 | 1073840810 | 12058170 | SELECT s_name , COUNT(*) AS numwait FROM
 supplier, lineitem l1, orders, nation WHERE s_suppkey = l1.
(5 rows)
```

In der SYS-Überwachungsansicht SYS_QUERY_HISTORY zeichnet Amazon Redshift die Abfrage wie folgt auf:

```
SELECT user_id, transaction_id, session_id, query_id, query_text::varchar(100)
FROM sys_query_history
WHERE transaction_id = 48237350
ORDER BY start_time;

 user_id | transaction_id | session_id | query_id |
          query_text
---------+----------------+------------+----------
+-----------------------------------------------------------------------------------------------------------
```

```
     101 |       48237350 | 1073840810 | 12058149 | SELECT s_name , COUNT(*) AS numwait
 FROM supplier, lineitem l1, orders, nation WHERE s_suppkey = l1.
```

In SYS_QUERY_DETAIL können Sie unter Verwendung des Wertes für „query_id“ aus SYS_QUERY_HISTORY Details auf Ebene der untergeordneten Abfrage abrufen. Die Spalte „child_query_sequence“ zeigt die Reihenfolge, in der die untergeordneten Abfragen ausgeführt werden. Weitere Informationen zu den Spalten in SYS_QUERY_DETAIL finden Sie unter SYS_QUERY_DETAIL.

```
select user_id,
       query_id,
       child_query_sequence,
       stream_id,
       segment_id,
       step_id,
       start_time,
       end_time,
       duration,
       blocks_read,
       blocks_write,
       local_read_io,
       remote_read_io,
       data_skewness,
       time_skewness,
       is_active,
       spilled_block_local_disk,
       spilled_block_remote_disk
from sys_query_detail
where query_id = 12058149
      and step_id = -1
order by query_id,
         child_query_sequence,
         stream_id,
         segment_id,
         step_id;

 user_id | query_id | child_query_sequence | stream_id | segment_id | step_id |
       start_time         |          end_time          | duration | blocks_read |
 blocks_write | local_read_io | remote_read_io | data_skewness | time_skewness |
 is_active | spilled_block_local_disk | spilled_block_remote_disk
---------+----------+----------------------+-----------+------------+---------
+----------------------------+----------------------------+----------+-------------
```

```
+--------------+---------------+----------------+---------------+---------------
+-----------+--------------------------+--------------------------
     101 | 12058149 |                    1 |         0 |          0 |      -1 |
 2023-09-27 15:40:38.512415 | 2023-09-27 15:40:38.533333 |    20918 |           0 |
         0 |             0 |              0 |             0 |            44 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         1 |          1 |      -1 |
 2023-09-27 15:40:39.931437 | 2023-09-27 15:40:39.972826 |    41389 |          12 |
         0 |            12 |              0 |             0 |            77 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         2 |          2 |      -1 |
 2023-09-27 15:40:40.584412 | 2023-09-27 15:40:40.613982 |    29570 |          32 |
         0 |            32 |              0 |             0 |            25 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         2 |          3 |      -1 |
 2023-09-27 15:40:40.582038 | 2023-09-27 15:40:40.615758 |    33720 |           0 |
         0 |             0 |              0 |             0 |             1 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         3 |          4 |      -1 |
 2023-09-27 15:40:46.668766 | 2023-09-27 15:40:46.705456 |    36690 |          24 |
         0 |            15 |              0 |             0 |            17 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         4 |          5 |      -1 |
 2023-09-27 15:40:46.707209 | 2023-09-27 15:40:46.709176 |     1967 |           0 |
         0 |             0 |              0 |             0 |            18 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         4 |          6 |      -1 |
 2023-09-27 15:40:46.70656  | 2023-09-27 15:40:46.71289  |     6330 |           0 |
         0 |             0 |              0 |             0 |             0 | f
 |                         0 |                           0
     101 | 12058149 |                    1 |         5 |          7 |      -1 |
 2023-09-27 15:40:46.71405  | 2023-09-27 15:40:46.714343 |      293 |           0 |
         0 |             0 |              0 |             0 |             0 | f
 |                         0 |                           0
     101 | 12058149 |                    2 |         0 |          0 |      -1 |
 2023-09-27 15:40:52.083907 | 2023-09-27 15:40:52.087854 |     3947 |           0 |
         0 |             0 |              0 |             0 |            35 | f
 |                         0 |                           0
     101 | 12058149 |                    2 |         1 |          1 |      -1 |
 2023-09-27 15:40:52.089632 | 2023-09-27 15:40:52.091129 |     1497 |           0 |
         0 |             0 |              0 |             0 |            11 | f
 |                         0 |                           0
     101 | 12058149 |                    2 |         1 |          2 |      -1 |
 2023-09-27 15:40:52.089008 | 2023-09-27 15:40:52.091306 |     2298 |           0 |
```

```
        0 |             0 |              0 |             0 |             0 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         0 |          0 |      -1 |
2023-09-27 15:40:56.882013 | 2023-09-27 15:40:56.897282 |    15269 |           0 |
        0 |             0 |              0 |             0 |            29 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         1 |          1 |      -1 |
2023-09-27 15:40:59.718554 | 2023-09-27 15:40:59.722789 |     4235 |           0 |
        0 |             0 |              0 |             0 |            13 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         2 |          2 |      -1 |
2023-09-27 15:40:59.800382 | 2023-09-27 15:40:59.807388 |     7006 |           0 |
        0 |             0 |              0 |             0 |            58 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         3 |          3 |      -1 |
2023-09-27 15:41:06.488685 | 2023-09-27 15:41:06.493825 |     5140 |           0 |
        0 |             0 |              0 |             0 |            56 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         3 |          4 |      -1 |
2023-09-27 15:41:06.486206 | 2023-09-27 15:41:06.497756 |    11550 |           0 |
        0 |             0 |              0 |             0 |             2 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         4 |          5 |      -1 |
2023-09-27 15:41:06.499201 | 2023-09-27 15:41:06.500851 |     1650 |           0 |
        0 |             0 |              0 |             0 |            15 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         4 |          6 |      -1 |
2023-09-27 15:41:06.498609 | 2023-09-27 15:41:06.500949 |     2340 |           0 |
        0 |             0 |              0 |             0 |             0 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         5 |          7 |      -1 |
2023-09-27 15:41:06.502945 | 2023-09-27 15:41:06.503282 |      337 |           0 |
        0 |             0 |              0 |             0 |             0 | f
|                        0 |                           0
    101 | 12058149 |                    4 |         0 |          0 |      -1 |
2023-09-27 15:41:06.62899  | 2023-09-27 15:41:06.631452 |     2462 |           0 |
        0 |             0 |              0 |             0 |            22 | f
|                        0 |                           0
    101 | 12058149 |                    4 |         1 |          1 |      -1 |
2023-09-27 15:41:06.632313 | 2023-09-27 15:41:06.63391  |     1597 |           0 |
        0 |             0 |              0 |             0 |            20 | f
|                        0 |                           0
    101 | 12058149 |                    4 |         1 |          2 |      -1 |
2023-09-27 15:41:06.631726 | 2023-09-27 15:41:06.633813 |     2087 |           0 |
```

```
         0 |             0 |              0 |             0 |             0 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         0 |          0 |      -1 |
 2023-09-27 15:41:12.571974 | 2023-09-27 15:41:12.584234 |    12260 |           0 |
         0 |             0 |              0 |             0 |            39 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         0 |          1 |      -1 |
 2023-09-27 15:41:12.569815 | 2023-09-27 15:41:12.585391 |    15576 |           0 |
         0 |             0 |              0 |             0 |             4 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         1 |          2 |      -1 |
 2023-09-27 15:41:13.758513 | 2023-09-27 15:41:13.76401  |     5497 |           0 |
         0 |             0 |              0 |             0 |            39 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         1 |          3 |      -1 |
 2023-09-27 15:41:13.749    | 2023-09-27 15:41:13.772987 |    23987 |           0 |
         0 |             0 |              0 |             0 |            32 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         2 |          4 |      -1 |
 2023-09-27 15:41:13.799526 | 2023-09-27 15:41:13.813506 |    13980 |           0 |
         0 |             0 |              0 |             0 |            62 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         2 |          5 |      -1 |
 2023-09-27 15:41:13.798823 | 2023-09-27 15:41:13.813651 |    14828 |           0 |
         0 |             0 |              0 |             0 |             0 | f
 |                        0 |                           0
(28 rows)
```

Abfrage-, Prozess- und Sitzungs-IDs für Systemtabellen

Beachten Sie bei der Analyse von Abfrage-, Prozess- und Sitzungs-IDs, die in Systemtabellen erscheinen, Folgendes:

- Der Wert der Abfrage-ID (in Spalten wie `query_id` und`query`) kann im Laufe der Zeit wiederverwendet werden.
- Der Wert für die Prozess-ID oder die Sitzungs-ID (in Spalten wie `process_id`pid, und`session_id`) kann im Laufe der Zeit wiederverwendet werden.
- Der Wert der Transaktions-ID (in Spalten wie `transaction_id` und`xid`) ist eindeutig.

SVV-Metadatenansichten

SVV-Ansichten sind Systemansichten in Amazon Redshift, die Informationen über Datenbankobjekte enthalten.

> **Note**
>
> Amazon Redshift meldet eine WARNING, keinen ERROR wenn eine Datenbankantwort aus irgendeinem Grund fehlschlägt. Amazon Redshift sendet keine FEHLERMELDUNGEN, wenn Sie Objekte in einem Datashare abfragen.

Themen

- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_FUNCTION_PRIVILEGES
- SVV_GEOGRAPHY_COLUMNS
- SVV_GEOMETRY_COLUMNS
- SVV_IAM_PRIVILEGES
- SVV_IDENTITY_PROVIDERS
- SVV_INTEGRATION
- SVV_INTEGRATION_TABLE_STATE
- SVV_INTERLEAVED_COLUMNS
- SVV_LANGUAGE_PRIVILEGES
- SVV_MASKING_POLICY
- SVV_ML_MODEL_INFO
- SVV_ML_MODEL_PRIVILEGES
- SVV_MV_DEPENDENCY
- SVV_MV_INFO
- SVV_QUERY_INFLIGHT
- SVV_QUERY_STATE
- SVV_REDSHIFT_COLUMNS
- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMA_QUOTA
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES
- SVV_RELATION_PRIVILEGES
- SVV_RLS_APPLIED_POLICY
- SVV_RLS_ATTACHED_POLICY
- SVV_RLS_POLICY

- SVV_RLS_RELATION
- SVV_ROLE_GRANTS
- SVV_ROLES
- SVV_SCHEMA_PRIVILEGES
- SVV_SCHEMA_QUOTA_STATE
- SVV_SYSTEM_PRIVILEGES
- SVV_TABLE_INFO
- SVV_TABLES
- SVV_TRANSACTIONS
- SVV_USER_GRANTS
- SVV_USER_INFO
- SVV_VACUUM_PROGRESS
- SVV_VACUUM_SUMMARY

SVV_ACTIVE_CURSORS

STV_ACTIVE_CURSORS zeigt Details zu aktuell offenen Cursor an. Weitere Informationen finden Sie unter DECLARE.

SVV_ACTIVE_CURSORS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten. Ein Benutzer kann nur Cursors sehen, die er selbst geöffnet hat. Ein Superuser kann alle Cursors sehen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die ID des Benutzers, der den Cursor erstellt hat.
cursor_name	varchar(128)	Der Name des Cursors.
transaction_id	bigint(128)	Die ID der Transaktion.

Spaltenname	Datentyp	Beschreibung
session_id	Ganzzahl	Die ID des Prozesses mit dem aktiven Cursor.
declare_time	Zeitstempel	Zeitpunkt, zu dem der Cursor deklariert wurde.
total_bytes	bigint	Die Größe des Cursor-Ergebnissatzes in Byte.
total_rows	bigint	die Anzahl der Zeilen im Cursor-Ergebnissatz.
fetched_rows	bigint	Die Anzahl der aktuell vom Cursor-Ergebnissatz abgerufenen Zeilen.
cursor_storage_limit_used_percent	Ganzzahl	Der prozentuale Anteil des derzeit vom Cursor verwendeten Speicherplatzes.

SVV_ALL_COLUMNS

Verwenden Sie SVV_ALL_COLUMNS, um eine Vereinigung von Spalten aus Amazon-Redshift-Tabellen anzuzeigen, wie in SVV_REDSHIFT_COLUMNS und der konsolidierten Liste aller externen Spalten aus allen externen Tabellen dargestellt. Weitere Information zu Amazon-Redshift-Spalten finden Sie unter SVV_REDSHIFT_COLUMNS.

SVV_ALL_COLUMNS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Name der Datenbank.

Spaltenname	Datentyp	Beschreibung
schema_name	varchar(128)	Der Name des Schemas.
table_name	varchar(128)	Der Name der Tabelle.
column_name	varchar(128)	Der Name der Spalte.
ordinal_position	integer	Die Position der Spalten in der Tabelle.
column_default	varchar(4000)	Der Standardwert der Spalte.
is_nullable	varchar(3)	Ein Wert, der angibt, ob die Tabelle den Wert Null enthalten kann. Mögliche Werte sind ja und nein.
data_type	varchar(128)	Der Datentyp der Spalte.
character_maximum_length	integer	Die maximale Anzahl von Zeichen in der Spalte.
numeric_precision	integer	Die numerische Präzision.
numeric_scale	integer	Die numerische Skala.
remarks	varchar(256)	Anmerkungen.

Beispielabfragen

Im folgenden Beispiel wird die Ausgabe von SVV_ALL_COLUMNS zurückgegeben.

```
SELECT *
FROM svv_all_columns
WHERE database_name = 'tickit_db'
    AND TABLE_NAME = 'tickit_sales_redshift'
ORDER BY COLUMN_NAME,
    SCHEMA_NAME
LIMIT 5;
```

```
 database_name | schema_name |     table_name        | column_name | ordinal_position
 | column_default | is_nullable | data_type | character_maximum_length |
 numeric_precision | numeric_scale | remarks
 -------------+------------+----------------------+-------------+-----------------
+----------------+-------------+-----------+--------------------------
+-------------------+---------------+---------
   tickit_db   |    public   | tickit_sales_redshift |    buyerid  |        4         |
               |      NO     |  integer  |                           |         32
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift | commission  |        9         |
               |     YES     |  numeric  |                           |          8
  | 2       |
   tickit_db   |    public   | tickit_sales_redshift |    dateid   |        7         |
               |      NO     |  smallint |                           |         16
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift |   eventid   |        5         |
               |      NO     |  integer  |                           |         32
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift |    listid   |        2         |
               |      NO     |  integer  |                           |         32
  |       0       |
```

SVV_ALL_SCHEMAS

Verwenden Sie SVV_ALL_SCHEMAS, um eine Vereinigung von Amazon-Redshift-Schemata anzuzeigen, wie in SVV_REDSHIFT_SCHEMAS und der konsolidierten Liste aller externen Schemata aus allen Datenbanken dargestellt. Weitere Informationen zu Amazon-Redshift-Schemata finden Sie unter SVV_REDSHIFT_SCHEMAS.

SVV_ALL_SCHEMAS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Der Name der Datenbank, in der das Schema existiert.
schema_name	varchar(128)	Der Name des Schemas.

Spaltenname	Datentyp	Beschreibung
schema_owner	integer	Die Benutzer-ID des Schemabesitzers. Weitere Informationen zu Benutzer-IDs finden Sie unter PG_USER_INFO.
schema_type	varchar(128)	Der Typ des Schemas. Mögliche Werte sind externe, lokale und freigegebene Schemata.
schema_acl	varchar(128)	Die Zeichenfolge, die die Berechtigungen für den angegebenen Benutzer oder die Benutzergruppe für das Schema definiert.
source_database	varchar(128)	Der Name der Quelldatenbank für das externe Schema.
schema_option	varchar(256)	Die Optionen des Schemas. Dies ist ein externes Schemaattribut.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_ALL_SCHEMAS zurückgegeben.

```
SELECT *
FROM svv_all_schemas
WHERE database_name = 'tickit_db'
ORDER BY database_name,
    SCHEMA_NAME;


 database_name |    schema_name     | schema_owner | schema_type | schema_acl |
 source_database | schema_option
```

```
--------------+--------------------+-------------+-------------+-----------
+-----------------+--------------
   tickit_db   |       public       |       1      |   shared    |           |
         |
```

SVV_ALL_TABLES

Verwenden Sie SVV_ALL_TABLES, um eine Vereinigung von Amazon-Redshift-Tabellen anzuzeigen, wie in SVV_REDSHIFT_TABLES und der konsolidierten Liste aller externen Tabellen aus allen externen Schemata dargestellt. Weitere Informationen zu Amazon-Redshift-Tabellen finden Sie unter SVV_REDSHIFT_TABLES.

SVV_ALL_TABLES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Der Name der Datenbank, in der die Tabelle existiert.
schema_name	varchar(128)	Der Name des Schemas für die Tabelle.
table_name	varchar(128)	Der Name der Tabelle.
table_acl	varchar(128)	♣Die Zeichenfolge, die die Berechtigung für den angegebenen Benutzer oder die Benutzergruppe für die Tabelle definiert.
table_type	varchar(128)	Der Typ der Tabelle. Mögliche Werte sind Ansichten, Basistabellen, externe Tabellen und freigegebene Tabellen.

Spaltenname	Datentyp	Beschreibung
remarks	varchar(256)	Anmerkungen.

Beispielabfragen

Im folgenden Beispiel wird die Ausgabe von SVV_ALL_TABLES zurückgegeben.

```
SELECT *
FROM svv_all_tables
WHERE database_name = 'tickit_db'
ORDER BY TABLE_NAME,
    SCHEMA_NAME
LIMIT 5;

 database_name | schema_name |        table_name        | table_type | table_acl |
 remarks
---------------+-------------+--------------------------+------------+-----------
+---------
   tickit_db   |    public   | tickit_category_redshift |    TABLE   |           |
   tickit_db   |    public   |   tickit_date_redshift   |    TABLE   |           |
   tickit_db   |    public   |   tickit_event_redshift  |    TABLE   |           |
   tickit_db   |    public   | tickit_listing_redshift  |    TABLE   |           |
   tickit_db   |    public   |   tickit_sales_redshift  |    TABLE   |           |
```

Wenn der Wert table_acl Null ist, wurden keine Zugriffsrechte explizit für die entsprechende Tabelle gewährt.

SVV_ALTER_TABLE_RECOMMENDATIONS

Zeichnet die aktuellen Empfehlungen von Amazon Redshift Advisor für Tabellen auf. Diese Ansicht zeigt Empfehlungen für alle Tabellen, unabhängig davon, ob sie für die automatische Optimierung definiert sind oder nicht. Um anzuzeigen, ob eine Tabelle für die automatische Optimierung definiert ist, siehe SVV_TABLE_INFO. Einträge werden nur für Tabellen angezeigt, die in der Datenbank der aktuellen Sitzung sichtbar sind. Nachdem eine Empfehlung (entweder von Amazon Redshift oder von Ihnen) angewendet wurde, wird sie nicht mehr in der Ansicht angezeigt.

SVV_ALTER_TABLE_RECOMMENDATIONS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
type	character (30)	Die Art der Empfehlung. Mögliche Werte sind distkey und sortkey.
Datenbank	character (128)	Der Datenbankname.
table_id	integer	Die Tabellenkennung.
group_id	integer	Die Gruppennummer eines Satzes von Empfehlungen. Alle Empfehlungen in einer Gruppe sollten angewendet werden, um den maximalen Nutzen zu erzielen. Mögliche Werte sind -1 für eine Sortierschlüsselempfehlung und eine Zahl größer als Null für eine Verteilungsschlüsselempfehlung.
ddl	character (1024)	Die SQL-Anweisung, die ausgeführt werden muss, um die Empfehlung anzuwenden.
auto_eligible	character (1)	Der Wert gibt an, ob die Empfehlung für die automatische Ausführung von Amazon Redshift berechtigt ist. Wenn dieser Wert `t` ist, lautet die Angabe „true“, wenn er `f` ist, lautet sie „false“.

Beispielabfragen

Im folgenden Beispiel zeigen die Zeilen im Ergebnis Empfehlungen für Verteilungs- und Sortierschlüssel. Die Zeilen zeigen auch an, ob die Empfehlungen dafür berechtigt sind, automatisch von Amazon Redshift angewendet zu werden.

```
select type, database, table_id, group_id, ddl, auto_eligible
from svv_alter_table_recommendations;
```

```
type      | database | table_id | group_id | ddl

                  | auto_eligible
diststyle | db0      | 117884   | 2        | ALTER TABLE "sch"."dp21235_tbl_1" ALTER
DISTSTYLE KEY DISTKEY "c0"
                    | f
diststyle | db0      | 117892   | 2        | ALTER TABLE "sch"."dp21235_tbl_1" ALTER
DISTSTYLE KEY DISTKEY "c0"
                    | f
diststyle | db0      | 117885   | 1        | ALTER TABLE "sch"."catalog_returns"
ALTER DISTSTYLE KEY DISTKEY "cr_sold_date_sk", ALTER COMPOUND SORTKEY
("cr_sold_date_sk","cr_returned_time_sk") | t
sortkey   | db0      | 117890   | -1       | ALTER TABLE "sch"."customer_addresses"
ALTER COMPOUND SORTKEY ("ca_address_sk")
                     | t
```

SVV_ATTACHED_MASKING_POLICY

Verwenden Sie SVV_ATTACHED_MASKING_POLICY, um alle Relationen und Rollen/Benutzer mit angefügten Richtlinien in der aktuell verbundenen Datenbank anzuzeigen.

Nur Superuser und Benutzer mit der Rolle `sys:secadmin` können SVV_ATTACHED_MASKING_POLICY einsehen. Regulären Benutzern werden 0 Zeilen angezeigt.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
policy_name	text	Der Name der Maskierungsrichtlinie, die der Tabelle angefügt ist.
schema_name	text	Das Schema der Tabelle, der die Richtlinie angefügt ist.
table_name	text	Der Name der Tabelle, der die Richtlinie angefügt ist.
table_type	text	Der Typ der Tabelle, der die Richtlinie angefügt ist.

Spaltenname	Datentyp	Beschreibung
grantor	text	Der Name des Benutzers, der die Richtlinie angefügt hat.
grantee	text	Der Name des Benutzers/der Rolle, dem/der die Richtlinie angefügt wurde.
grantee_type	text	Die Art des Berechtig ungsempfängers. Dies kann eine Rolle, ein Benutzer oder Public sein.
priority	int	Die Priorität der angefügten Richtlinie.
input_columns	text	Die Eingabespaltenattribute der angefügten Richtlinie.
output_columns	text	Die Ausgabespaltenattribute der angefügten Richtlinie.

Interne Funktionen

SVV_ATTACHED_MASKING_POLICY unterstützt die folgenden internen Funktionen:

mask_get_policy_for_role_on_column

Ruft die Richtlinie mit der höchsten Priorität ab, die für ein bestimmtes Spalten-/Rollenpaar gilt.

Syntax

```
mask_get_policy_for_role_on_column
                        (relschema,
                        relname,
                        colname,
                        rolename);
```

Parameter

relschema

Der Name des Schemas, in dem sich die Richtlinie befindet.

relname

Der Name der Tabelle, in der sich die Richtlinie befindet.

colname

Der Name der Spalte, der die Richtlinie angefügt ist.

rolename

Der Name der Rolle, der die Richtlinie angefügt ist.

mask_get_policy_for_user_on_column

Ruft die Richtlinie mit der höchsten Priorität ab, die für ein bestimmtes Spalten-/Benutzerpaar gilt.

Syntax

```
mask_get_policy_for_user_on_column
                        (relschema,
                        relname,
                        colname,
                        username);
```

Parameter

relschema

Der Name des Schemas, in dem sich die Richtlinie befindet.

relname

Der Name der Tabelle, in der sich die Richtlinie befindet.

colname

Der Name der Spalte, der die Richtlinie angefügt ist.

rolename

Der Name des Benutzers, dem die Richtlinie angefügt ist.

SVV_COLUMNS

Verwenden Sie SVV_COLUMNS zur Anzeige von Kataloginformationen zu den Spalten lokaler und externer Tabellen und Ansichten einschließlich Ansichten mit später Bindung.

SVV_COLUMNS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Die SVV_COLUMNS-Ansicht führt Tabellenmetadaten der Systemkatalogtabellen- (Tabellen mit einem PG-Präfix) und der SVV_EXTERNAL_COLUMNS-Systemansicht zusammen. Die Systemkatalogtabellen beschreiben Amazon-Redshift-Datenbanktabellen. SVV_EXTERNAL_COLUMNS beschreibt externe Tabellen, die mit Amazon Redshift Spectrum verwendet werden.

Alle Benutzer können sämtliche Zeilen der Systemkatalogtabellen sehen. Normale Benutzer können Spaltendefinitionen aus der Ansicht SVV_EXTERNAL_COLUMNS nur für externe Tabellen sehen, auf die Sie zugreifen können. Reguläre Benutzer können zwar Tabellenmetadaten in den Systemkatalogtabellen sehen, sie können jedoch nur Daten aus benutzerdefinierten Tabellen auswählen, wenn sie Besitzer der Tabelle sind oder Zugriff darauf haben.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
table_catalog	Text	Der Name des Katalogs, in dem sich die Tabelle befindet.
table_schema	Text	Der Name des Schemas für die Tabelle.
table_name	Text	Der Name der Tabelle.
column_name	Text	Der Name der Spalte.
ordinal_position	int	Die Position der Spalten in der Tabelle.
column_default	Text	Der Standardwert der Spalte.

Spaltenname	Datentyp	Beschreibung
is_nullable	Text	Ein Wert, der angibt, ob die Tabelle den Wert Null enthalten kann.
data_type	Text	Der Datentyp der Spalte.
character_maximum_length	int	Die maximale Anzahl von Zeichen in der Spalte.
numeric_precision	int	Die numerische Präzision. Wenn die Spalte data_type numerisch ist, gibt diese Spalte die Anzahl der signifikanten Stellen im Gesamtwert zurück.
numeric_precision_radix	int	Die Wurzel der numerischen Präzision. Wenn die Spalte data_type numerisch ist, gibt diese Spalte die Basis der Spalten numeric_precision und numeric_scale zurück.
numeric_scale	int	Die numerische Skala. Wenn die Spalte data_type numerisch ist, gibt diese Spalte die Anzahl der signifikanten Stellen im Dezimalwert zurück.
datetime_precision	int	Die Datum-/Uhrzeit-Präzision.
interval_type	Text	Der Intervalltyp.
interval_precision	Text	Die Intervallpräzision.
character_set_catalog	Text	Der Zeichensatzkatalog.

Spaltenname	Datentyp	Beschreibung
character_set_schema	Text	Das Zeichensatzschema.
character_set_name	Text	Der Name des Zeichensatzes.
collation_catalog	Text	Der Kollationskatalog.
collation_schema	Text	Das Kollationsschema.
collation_name	Text	Der Kollationsname.
domain_name	Text	Der Domänenname.
remarks	Text	Anmerkungen.

SVV_COLUMN_PRIVILEGES

Verwenden Sie SVV_COLUMN_PRIVILEGES, um die Spaltenberechtigungen anzuzeigen, die Benutzern, Rollen und Gruppen in der aktuellen Datenbank explizit gewährt werden.

SVV_COLUMN_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
namespace_name	text	Der Name des Namespace, in dem eine angegebene Beziehung vorhanden ist.
relation_name	text	Der Name der Beziehung.
column_name	Text	Der Name der Spalte.

Spaltenname	Datentyp	Beschreibung
privilegege_type	text	Der Typ der Berechtigung. Mögliche Werte sind SELECT und UPDATE.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_COLUMN_PRIVILEGES angezeigt.

```
SELECT
 namespace_name,relation_name,COLUMN_NAME,privilege_type,identity_name,identity_type
FROM svv_column_privileges WHERE relation_name = 'lineitem';

 namespace_name | relation_name | column_name | privilege_type | identity_name |
 identity_type
----------------+---------------+-------------+----------------+---------------
+----------------
    public      |   lineitem    | l_orderkey  |     SELECT     |    reguser    |
 user
    public      |   lineitem    | l_orderkey  |     SELECT     |     role1     |
 role
    public      |   lineitem    | l_partkey   |     SELECT     |    reguser    |
 user
    public      |   lineitem    | l_partkey   |     SELECT     |     role1     |
 role
```

SVV_DATABASE_PRIVILEGES

Verwenden Sie SVV_DATABASE_PRIVILEGES, um die Datenbankberechtigungen anzuzeigen, die Benutzern, Rollen und Gruppen in Ihrem Amazon-Redshift-Cluster explizit gewährt werden.

SVV_DATABASE_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	text	Name der Datenbank.
privilege_type	text	Der Typ der Berechtigung. Mögliche Werte sind USAGE, CREATE und TEMP.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenidentitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_DATABASE_PRIVILEGES angezeigt.

```
SELECT database_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_database_privileges
WHERE database_name = 'test_db';
```

```
 database_name | privilege_type | identity_name | identity_type | admin_option
---------------+----------------+---------------+---------------+--------------
     test_db   |     CREATE     |     reguser   |      user     |     False
     test_db   |     CREATE     |      role1    |      role     |     False
     test_db   |     TEMP       |      public   |      public   |     False
     test_db   |     TEMP       |      role1    |      role     |     False
```

SVV_DATASHARE_PRIVILEGES

Verwenden Sie SVV_DATASHARE_PRIVILEGES, um die Datashare-Berechtigungen anzuzeigen, die Benutzern, Rollen und Gruppen in Ihrem Amazon-Redshift-Cluster explizit gewährt werden.

SVV_DATASHARE_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
datashare_name	text	Der Name des Datashares.
privilege_type	text	Der Typ der Berechtigung. Mögliche Werte sind ALTER und SHARE.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenidentitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_DATASHARE_PRIVILEGES angezeigt.

```
SELECT datashare_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_datashare_privileges
WHERE datashare_name = 'demo_share';

 datashare_name | privilege_type |  identity_name | identity_type | admin_option
----------------+----------------+----------------+---------------+--------------
   demo_share   |     ALTER      |    superuser   |     user      |   False
   demo_share   |     ALTER      |    reguser     |     user      |   False
```

SVV_DATASHARES

Verwenden Sie SVV_DATASHARES, um eine Liste der im Cluster erstellten Datashares sowie der für den Cluster freigegebenen Datashares anzuzeigen.

SVV_DATASHARES ist für die folgenden Benutzer sichtbar:

- Superuser
- Datashare-Besitzer
- Benutzer mit ALTER- oder USAGE-Berechtigungen für ein Datashare

Anderen Benutzern werden keine Zeilen angezeigt. Weitere Informationen über ALTER- und USAGE-Berechtigungen finden Sie unter GRANT.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
share_name	varchar(128)	Der Name eines Datashares.
share_id	integer	Die ID des Datashares.
share_owner	integer	Der Eigentümer des Datashares.

Spaltenname	Datentyp	Beschreibung
source_database	varchar(128)	Die Quelldatenbank für diesen Datashare.
consumer_database	varchar(128)	Die Konsumentendatenbank, die aus diesem Datashare erstellt wird.
share_type	varchar(8)	Der Typ des Datashare s. Mögliche Werte sind INBOUND und OUTBOUND.
createdate	Timestamp ohne Zeitzone	Das Datum, an dem das Datashare erstellt wurde.
is_publicaccessible	Boolean	Diese Eigenschaft gibt an, ob ein Datashare für öffentlich zugängliche Cluster freigegeb en werden kann.
share_acl	varchar(256)	Die Zeichenfolge, die die Berechtigungen für den angegebenen Benutzer oder die Benutzergruppe für den Datashare definiert.
producer_account	varchar(16)	Die ID für das Datashare-Produzentenkonto.
producer_namespace	varchar(64)	Die eindeutige Cluster-ID des Produzenten-Clusters des Datashares.
managed_by	varchar(64)	Die Eigenschaft, die den AWS Dienst angibt, der die Datenfreigabe verwaltet.

Nutzungshinweise

Zusätzliche Metadaten abrufen — Mithilfe der in der `share_owner` Spalte zurückgegebenen Ganzzahl können Sie eine Verknüpfung herstellen, SVL_USER_INFO um Daten über den `usesysid` Eigentümer des Datenaustauschs abzurufen. Dazu gehören der Name und zusätzliche Eigenschaften.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_DATASHARES zurückgegeben.

```
SELECT share_owner, source_database, share_type, is_publicaccessible
FROM svv_datashares
WHERE share_name LIKE 'tickit_datashare%'
AND source_database = 'dev';

  share_owner | source_database | share_type  | is_publicaccessible
--------------+-----------------+-------------+---------------------
     100      |      dev        |   OUTBOUND  |        True
(1 rows)
```

Im folgenden Beispiel wird die Ausgabe von SVV_DATASHARES für ausgehende Datashares zurückgegeben.

```
SELECT share_name, share_owner, btrim(source_database), btrim(consumer_database),
 share_type, is_publicaccessible, share_acl, btrim(producer_account),
 btrim(producer_namespace), btrim(managed_by) FROM svv_datashares WHERE share_type =
 'OUTBOUND';

   share_name   | share_owner | source_database | consumer_database | share_type |
 is_publicaccessible | share_acl | producer_account|         producer_namespace
   | managed_by
----------------+-------------+-----------------+-------------------
+------------+---------------------+-----------+-----------------
+--------------------------------------+------------
    salesshare  |      1      |       dev       |                   |  OUTBOUND  |
   True         |           |   123456789012  | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |

 marketingshare |      1      |       dev       |                   |  OUTBOUND  |
   True         |           |   123456789012  | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
```

Im folgenden Beispiel wird die Ausgabe von SVV_DATASHARES für eingehende Datashares zurückgegeben.

```
SELECT share_name, share_owner, btrim(source_database), btrim(consumer_database),
 share_type, is_publicaccessible, share_acl, btrim(producer_account),
 btrim(producer_namespace), btrim(managed_by) FROM svv_datashares WHERE share_type =
 'INBOUND';

  share_name    | share_owner | source_database | consumer_database | share_type |
 is_publicaccessible | share_acl | producer_account |         producer_namespace
    | managed_by
---------------+-------------+-----------------+-------------------
+------------+---------------------+-----------+-----------------
+------------------------------------+------------
  salesshare    |             |                 |                   |  INBOUND   |
   False         |           |  123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
 |
 marketingshare |             |                 |                   |  INBOUND   |
  False         |           |  123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
 ADX
```

SVV_DATASHARE_CONSUMERS

Verwenden Sie SVV_DATASHARE_CONSUMERS, um eine Liste von Konsumenten für einen Datashare anzuzeigen, der in einem Cluster erstellt wurde.

SVV_DATASHARE_CONSUMERS ist für die folgenden Benutzer sichtbar:

- Superuser
- Datashare-Besitzer
- Benutzer mit ALTER- oder USAGE-Berechtigungen für ein Datashare

Anderen Benutzern werden keine Zeilen angezeigt. Weitere Informationen über ALTER- und USAGE-Berechtigungen finden Sie unter GRANT.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
share_name	varchar(128)	Der Name des Datashares.

Spaltenname	Datentyp	Beschreibung
consumer_account	varchar(16)	Die Konto-ID für den Datashare-Konsumenten.
consumer_namespace	varchar(64)	Die eindeutige Cluster-IT des Datashare-Konsumenten-Clusters.
share_date	Timestamp ohne Zeitzone	Das Datum, an dem das Datashare freigegeben wurde.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_DATASHARE_CONSUMERS zurückgegeben.

```
SELECT count(*)
FROM svv_datashare_consumers
WHERE share_name LIKE 'tickit_datashare%';

1
```

SVV_DATASHARE_OBJECTS

Verwenden Sie SVV_DATASHARE_OBJECTS, um eine Liste der Objekte in allen Datashares anzuzeigen, die im Cluster erstellt oder für den Cluster freigegeben wurden.

SVV_DATASHARE_OBJECTS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zum Anzeigen einer Liste von Datashares finden Sie unter SVV_DATASHARES.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
share_type	varchar(8)	Der Typ des angegeben en Datashares. Mögliche

Spaltenname	Datentyp	Beschreibung
		Werte sind OUTBOUND und INBOUND.
share_name	varchar(128)	Der Name des Datashares.
object_type	varchar(64)	Der Typ eines angegeben en Objekts. Mögliche Werte sind Schemata, Tabellen, Ansichten, spätbindende Ansichten, materialisierte Ansichten und Funktionen.
object_name	varchar(512)	Der Name des Objekts. Der Objektname umfasst den Schemanamen, z. B. schema1.t1.
producer_account	varchar(16)	Die ID für das Datashare-Produzentenkonto.
producer_namespace	varchar(64)	Die eindeutige Cluster-ID des Produzenten-Clusters des Datashares.
include_new	Boolean	Diese Eigenschaft gibt an, ob zukünftige Tabellen, Ansichten oder benutzerdefinierte SQL-Funktionen (UDFs), die in dem angegebenen Schema erstellt wurden, dem Datashare hinzugefügt werden sollen. Dieser Parameter ist nur für OUTBOUND-Datashares und nur für Schematypen im Datenspeicher relevant.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_DATASHARE_OBJECTS zurückgegeben.

```
SELECT share_type,
    btrim(share_name)::varchar(16) AS share_name,
    object_type,
    object_name
FROM svv_datashare_objects
WHERE share_name LIKE 'tickit_datashare%'
AND object_name LIKE '%tickit%'
ORDER BY object_name
LIMIT 5;

 share_type |     share_name     | object_type |          object_name
------------+--------------------+-------------+--------------------------------
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_category_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_date_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_event_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_listing_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_sales_redshift
```

```
SELECT * FROM SVV_DATASHARE_OBJECTS WHERE share_name like 'sales%';

share_type | share_name | object_type | object_name  | producer_account |
 producer_namespace          | include_new
-----------+------------+-------------+--------------+------------------
+--------------------------------------+-------------
 OUTBOUND  | salesshare | schema      | public       | 123456789012     |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |      t
 OUTBOUND  | salesshare | table       | public.sales | 123456789012     |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
```

SVV_DEFAULT_PRIVILEGES

Verwenden Sie SVV_DEFAULT_PRIVILEGES, um die Standardberechtigungen anzuzeigen, auf die ein Benutzer in einem Amazon-Redshift-Cluster zugreifen kann.

SVV_DEFAULT_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser

- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Standardberechtigungen sehen, die ihnen erteilt wurden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
schema_name	text	Der Name des Schemas.
object_type	text	Der Typ des Objekts. Mögliche Werte sind RELATION, FUNCTION oder PROCEDURE.
owner_id	Ganzzahl	Die Eigentümer-ID. Möglicher Wert ist die Benutzer-ID.
owner_name	text	Der Name des Eigentümers.
owner_type	text	Der Typ des Eigentümers. Möglicher Wert ist Benutzer.
privilege_type	text	Der Privilegtyp. Mögliche Werte sind INSERT, SELECT, UPDATE, DELETE, RULE, REFERENCES, TRIGGER, DROP und UPDATE.
grantee_id	Ganzzahl	Die ID des Berechtigungsempfängers. Mögliche Werte sind Benutzer-ID, Rollen-ID und Gruppen-ID.
grantee_type	text	Der Typ des Berechtigungsempfängers. Mögliche Werte sind Benutzer, Rolle, Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen Berechtigungen erteilen kann. Für den Rollen- und Gruppentyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe für SVV_DEFAULT_PRIVILEGES zurückgegeben.

```
SELECT * from svv_default_privileges;
```

```
 schema_name |    object_type    | owner_id | owner_name | owner_type | privilege_type
 | grantee_id | grantee_name | grantee_type | admin_option
-------------+------------------+--------- +------------+------------+---------------
+------------+--------------+--------------+-------------+
   public    |     RELATION      |    106   |     u1     |    user    |     UPDATE
 |     107    |      u2      |     user     |      f      |
   public    |     RELATION      |    106   |     u1     |    user    |     SELECT
 |     107    |      u2      |     user     |      f      |
```

SVV_DISKUSAGE

Amazon Redshift erstellt die Systemansicht SVV_DISKUSAGE durch Verbindung der Tabellen STV_TBL_PERM und STV_BLOCKLIST. Die Ansicht SVV_DISKUSAGE enthält Informationen zur Datenzuweisung für die Tabellen in einer Datenbank.

Verwenden Sie aggregierte Abfragen mit SVV_DISKUSAGE wie die folgenden Beispiele zeigen, um die Anzahl der Festplattenblöcke zu bestimmen, die pro Datenbank, Tabelle, Slice oder Spalte zugewiesen sind. Jeder Datenblock nimmt 1 MB in Anspruch. Sie können auch STV_PARTITIONS verwenden, um zusammenfassende Informationen zur Festplattennutzung anzuzeigen.

SVV_DISKUSAGE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> Diese Ansicht ist nur verfügbar, wenn bereitgestellte Cluster abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
db_id	integer	Datenbank-ID.
Name	character (72)	Tabellenname.

Spaltenname	Datentyp	Beschreibung
slice	integer	Datenslice, der der Tabelle zugewiesen ist.
col	integer	Null-basierter Index für die Spalte. Jeder Tabelle, die Sie erstellen , werden diese drei verborgenen Spalten angehängt: INSERT_XID, DELETE_XID, und ROW_ID (OID). Eine Tabelle mit drei vom Benutzer definierten Spalten hat tatsächlich sechs Spalten, und die benutzerd efinierten Spalten haben die interne Nummerierung 0, 1 und 2. Die Spalten INSERT_XID, DELETE_XID, und ROW_ID haben in diesem Beispiel die Nummern 3, 4 und 5.
tbl	integer	Tabellen-ID.
blocknum	integer	ID für den Datenblock.
num_values	integer	Anzahl der enthaltenen Werte für den Block.
minvalue	bigint	Mindestwert für den Block.
maxvalue	bigint	Höchstwert für den Block.
sb_pos	integer	Interne Kennung für die Position des Superblocks auf der Festplatte.
pinned	integer	Ob der Block im Rahmen des Pre-Load-Vorgangs im Speicher fixiert ist. 0 = false; 1 = true. Standard ist „false“.
on_disk	integer	Ob der Block automatisch auf der Festplatte gespeichert wird. 0 = false; 1 = true. Standard ist „false“.
modified	integer	Ob der Block modifiziert wurde. 0 = false; 1 = true. Standard ist „false“.
hdr_modified	integer	Ob der Block-Header modifiziert wurde. 0 = false; 1 = true. Standard ist „false“.
unsorted	integer	Ob der Block unsortiert ist. 0 = false; 1 = true. Der Standardwert ist „true“.
tombstone	integer	Zur internen Verwendung.

Spaltenname	Datentyp	Beschreibung
preferred_diskno	integer	Nummer der Festplatte, auf der sich der Block befinden sollte, sofern die Festplatte nicht ausgefallen ist. Sobald die Festplatte festgelegt ist, wird der Block wieder auf diese Festplatte übertragen.
temporary	integer	Ob der Block temporäre Daten enthält, etwa aus einer temporären Tabelle oder aus Abfragezwischenergebnissen. 0 = false; 1 = true. Standard ist „false“.
newblock	integer	Zeigt an, ob ein Block neu ist (true) oder noch nie einem Commit-Vorgang zu einer Festplatte unterzogen wurde (false). 0 = false; 1 = true.

Beispielabfragen

SVV_DISKUSAGE enthält eine Zeile pro zugewiesenem Festplattenblock, so dass eine Abfrage mit Auswahl aller Zeilen möglicherweise eine sehr große Zahl von Zeilen ausgibt. Wir empfehlen, nur aggregierte Abfragen mit SVV_DISKUSAGE zu verwenden.

Ausgabe der höchsten jemals zu Spalte 6 in der Tabelle USERS (der Spalte EMAIL) Anzahl von Blöcken:

```
select db_id, trim(name) as tablename, max(blocknum)
from svv_diskusage
where name='users' and col=6
group by db_id, name;

db_id  | tablename | max
--------+-----------+-----
175857 | users     |   2
(1 row)
```

Die folgende Abfrage gibt ähnliche Ergebnisse für alle Spalten in einer großen 10-Spalten-Tabelle mit der Bezeichnung SALESNEW aus. (Die letzten drei Zeilen, für die Spalten 10 bis 12, beziehen sich auf die verborgenen Metadatenspalten.)

```
select db_id, trim(name) as tablename, col, tbl, max(blocknum)
from svv_diskusage
```

```
where name='salesnew'
group by db_id, name, col, tbl
order by db_id, name, col, tbl;

db_id  | tablename  | col |  tbl   | max
--------+------------+-----+--------+-----
175857 | salesnew   |   0 | 187605 | 154
175857 | salesnew   |   1 | 187605 | 154
175857 | salesnew   |   2 | 187605 | 154
175857 | salesnew   |   3 | 187605 | 154
175857 | salesnew   |   4 | 187605 | 154
175857 | salesnew   |   5 | 187605 |  79
175857 | salesnew   |   6 | 187605 |  79
175857 | salesnew   |   7 | 187605 | 302
175857 | salesnew   |   8 | 187605 | 302
175857 | salesnew   |   9 | 187605 | 302
175857 | salesnew   |  10 | 187605 |   3
175857 | salesnew   |  11 | 187605 |   2
175857 | salesnew   |  12 | 187605 | 296
(13 rows)
```

SVV_EXTERNAL_COLUMNS

Verwenden Sie SVV_EXTERNAL_COLUMNS zur Anzeige der Details für Spalten in externen Tabellen. Verwenden Sie SVV_EXTERNAL_COLUMNS auch für datenbankübergreifende Abfragen, um Details zu allen Spalten der Tabelle in nicht verbundenen Datenbanken anzuzeigen, auf die Benutzer Zugriff haben.

SVV_EXTERNAL_COLUMNS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
redshift_database_name	text	Der Name der lokalen Amazon-Redshift-Datenbank.

Spaltenname	Datentyp	Beschreibung
schemaname	Text	Der Name des externen Amazon-Redshift-Schemas für die externe Tabelle.
tablename	Text	Der Name der externen Tabelle.
columnname	Text	Der Name der Spalte.
external_type	Text	Der Datentyp der Spalte.
columnnum	integer	Die externe Spaltennummer, beginnend bei 1.
part_key	integer	Wenn die Spalte ein Partition sschlüssel ist, die Reihenfolge des Schlüssels. Wenn es sich bei der Spalte nicht um eine Partition handelt, ist der Wert **0**.
is_nullable	Text	Definiert, ob eine Spalte löschbar ist oder nicht. Einige Werte sind `true`, `false` oder enthalten " ", eine leere Zeichenfolge, die keine Information darstellt.

SVV_EXTERNAL_DATABASES

Verwenden Sie SVV_EXTERNAL_DATABASES zur Anzeige von Details für externe Datenbanken.

SVV_EXTERNAL_DATABASES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
eskind	integer	Die Art des externen Katalogs für die Datenbank; **1** gibt einen Datenkatalog, **2** einen Hive-Metastore an.
esoptions	Text	Details des Katalogs, in dem sich die Datenbank befindet.
databasename	Text	Der Name der Datenbank in dem externen Katalog.
location	Text	Der Speicherort der Datenbank.
parameters	Text	Datenbankparameter.

SVV_EXTERNAL_PARTITIONS

Verwenden Sie SVV_EXTERNAL_PARTITIONS zur Anzeige der Details für Partitionen in externen Tabellen.

SVV_EXTERNAL_PARTITIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten..

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
schemaname	Text	Der Name des externen Amazon-Redshift-Schemas für die externe Tabelle mit den angegebenen Partitionen.
tablename	Text	Der Name der externen Tabelle.

Spaltenname	Datentyp	Beschreibung
values	Text	Werte für die Partition.
location	Text	Der Speicherort der Partition. Die Spaltengröße ist auf 128 Zeichen begrenzt. Längere Werte werden abgeschnitten.
input_format	Text	Das Eingabeformat.
output_format	Text	Das Ausgabeformat.
serialization_lib	Text	Die Serialisierungsbibliothek.
serde_parameters	text	SerDe Parameter.
compressed	integer	Ein Wert, der angibt, ob die Partition komprimiert ist; **1** gibt an, dass sie komprimiert ist, **0**, dass sie nicht komprimiert ist.
parameters	Text	Eigenschaften der Partition.

SVV_EXTERNAL_SCHEMAS

Verwenden Sie SVV_EXTERNAL_SCHEMAS zur Anzeige von Informationen über externe Schemata. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

SVV_EXTERNAL_SCHEMAS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
esoid	OID	ID des externen Schemas.
eskind	smallint	Der Typ des externen Katalogs für das externe Schema: 1 steht für einen Datenkatalog, 2 für einen Hive-Meta

Spaltenname	Datentyp	Beschreibung
		store, 3 für eine Verbundabfrage an Aurora PostgreSQL oder Amazon RDS PostgreSQL, 4 für ein Schema für eine lokale Amazon-Redshift-Datenbank, 5 für ein Schema für eine Remote-Amazon-Redshift-Datenbank, 6 für ein Schema für eine Systemtabelle, 8 für ein Schema für Remote-MySQL-Datenbanken, 9 für ein Schema für Amazon-Kinesis-Datenstrom und 10 für einen Amazon-Managed-Streaming-Datenstrom für Apache Kafka.
schemaname	Name	Name des externen Schemas.
esowner	integer	Benutzer-ID des Eigentümers des externen Schemas.
databasename	Text	Name der externen Datenbank.
esoptions	Text	Optionen für das externe Schema.

Beispiel

Das folgende Beispiel zeigt Einzelheiten für externe Schemata.

```
select * from svv_external_schemas;

esoid  | eskind | schemaname | esowner | databasename | esoptions

-------+--------+------------+---------+--------------
+-------------------------------------------------------------
100133 |      1 | spectrum   |     100 | redshift     |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

SVV_EXTERNAL_TABLES

Verwenden Sie SVV_EXTERNAL_TABLES zur Anzeige von Details für externe Tabellen. Weitere Informationen unter CREATE EXTERNAL SCHEMA. Verwenden Sie SVV_EXTERNAL_TABLES

auch für datenbankübergreifende Abfragen, um Metadaten zu allen Tabellen in nicht verbundenen Datenbanken anzuzeigen, auf die Benutzer Zugriff haben.

SVV_EXTERNAL_TABLES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
redshift_database_name	text	Der Name der lokalen Amazon-Redshift-Datenbank.
schemaname	Text	Der Name des externen Amazon-Redshift-Schemas für die externe Tabelle.
tablename	Text	Der Name der externen Tabelle.
tabletype	Text	Der Typ der Tabelle. Einige Werte sind TABLE, VIEW, MATERIALIZED VIEW oder enthalten " ", eine leere Zeichenfolge, die keine Information darstellt.
location	Text	Der Speicherort der Tabelle.
input_format	Text	Das Eingabeformat
output_format	Text	Das Ausgabeformat.
serialization_lib	Text	Die Serialisierungsbibliothek.
serde_parameters	text	SerDe Parameter.
compressed	integer	Ein Wert, der angibt, ob die Tabelle komprimiert ist; **1** gibt

Spaltenname	Datentyp	Beschreibung
		an, dass sie komprimiert ist, **0**, dass sie nicht komprimiert ist.
parameters	Text	Tabelleneigenschaften.

Beispiel

Das folgende Beispiel zeigt Details zu svv_external_tables mit einem Prädikat für das externe Schema, das von einer Verbundabfrage verwendet wird.

```
select schemaname, tablename from svv_external_tables where schemaname = 'apg_tpch';
schemaname  | tablename
------------+-----------
apg_tpch    | customer
apg_tpch    | lineitem
apg_tpch    | nation
apg_tpch    | orders
apg_tpch    | part
apg_tpch    | partsupp
apg_tpch    | region
apg_tpch    | supplier
(8 rows)
```

SVV_FUNCTION_PRIVILEGES

Verwenden Sie SVV_FUNCTION_PRIVILEGES, um die Funktionsberechtigungen anzuzeigen, die Benutzern, Rollen und Gruppen in der aktuellen Datenbank explizit gewährt werden.

SVV_FUNCTION_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
namespace_name	text	Der Name des Namespace, in dem eine angegebene Funktion vorhanden ist.
function_name	text	Der Name der Funktion.
argument_types	text	Der String, der den Typ des Eingabeargumente einer Funktion darstellt.
privilege ge_type	text	Der Typ der Berechtigung. Der mögliche Wert ist EXECUTE.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenidentitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_FUNCTION_PRIVILEGES angezeigt.

```
SELECT
 namespace_name,function_name,argument_types,privilege_type,identity_name,identity_type,admin_o
 FROM svv_function_privileges
WHERE identity_name IN ('role1', 'reguser');
```

```
 namespace_name | function_name |       argument_types       | privilege_type |
 identity_name | identity_type | admin_option
----------------+---------------+----------------------------+----------------
+----------------+---------------+-------------
    public      | test_func1    | integer                    |    EXECUTE     |
 role1     |     role      |  False
    public      | test_func2    | integer, character varying |    EXECUTE     |
 reguser    |     user      |  False
```

SVV_GEOGRAPHY_COLUMNS

Verwenden Sie SVV_GEOGRAPHY_COLUMNS, um die Liste der GEOGRAPHY-Spalten in Ihrem Data Warehouse anzuzeigen. Diese Spaltenliste enthält Spalten aus Datashares.

SVV_GEOGRAPHY_COLUMNS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
f_table_catalog	varchar(128)	Der Name der Datenbank, in der sich die Tabelle mit der GEOGRAPHY-Spalte befindet.
f_table_schema	varchar(128)	Der Name des Schemas, in dem sich die Tabelle mit der GEOGRAPHY-Spalte befindet.
f_table_name	varchar(128)	Der Name der Tabelle, in der sich die GEOGRAPHY-Spalte befindet.
f_geography_column	varchar(128)	Der Name der GEOGRAPHY-Spalte.
coord_dimension	Ganzzahl	Die Anzahl der Dimensionen der GEOGRAPHY-Daten.
srid	Ganzzahl	Der Spatial Reference System Identifier (SRID) der GEOGRAPHY-Daten.

Spaltenname	Datentyp	Beschreibung
Typ	varchar(128)	Der Name des Datentyps für koordinatenbasierte Geografie.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_GEOGRAPHY_COLUMNS angezeigt.

```
SELECT * FROM svv_geography_columns;

f_table_catalog  | f_table_schema  | f_table_name  | f_geography_column  |
 coord_dimension | srid |  type
-----------------+-----------------+--------------+--------------------
+-----------------+------+-------------
dev              | public          | spatial_test  | test_geography      | 2
    | 0    | GEOGRAPHY
```

SVV_GEOMETRY_COLUMNS

Verwenden Sie SVV_GEOMETRY_COLUMNS, um die Liste der GEOMETRY-Spalten in Ihrem Data Warehouse anzuzeigen. Diese Spaltenliste enthält Spalten aus Datashares.

SVV_GEOMETRY_COLUMNS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
f_table_catalog	varchar(128)	Der Name der Datenbank, in der sich die Tabelle mit der GEOMETRY-Spalte befindet.
f_table_schema	varchar(128)	Der Name des Schemas, in dem sich die Tabelle mit der GEOMETRY-Spalte befindet.
f_table_name	varchar(128)	Der Name der Tabelle, in der sich die GEOMETRY-Spalte befindet.

Spaltenname	Datentyp	Beschreibung
f_geometry_column	varchar(128)	Der Name der GEOMETRY-Spalte.
coord_dimension	Ganzzahl	Die Anzahl der Dimensionen der GEOMETRY-Daten.
srid	Ganzzahl	Der Spatial Reference System Identifier (SRID) der GEOMETRY-Spalte.
Typ	varchar(128)	Der Name des koordinatenbasierten Geometrietyps.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_GEOMETRY_COLUMNS angezeigt.

```
SELECT * FROM svv_geometry_columns;

f_table_catalog  | f_table_schema  | f_table_name  | f_geometry_column   |
 coord_dimension | srid |  type
-----------------+-----------------+---------------+--------------------
+-----------------+------+--------------
dev              | public          | accomodations | shape               | 2
    | 0    | GEOMETRY
dev              | public          | zipcode       | wkb_geometry        | 2
    | 0    | GEOMETRY
```

SVV_IAM_PRIVILEGES

Verwenden Sie SVV_IAM_PRIVILEGES, um explizit gewährte IAM-Berechtigungen für Benutzer, Rollen und Gruppen anzuzeigen.

SVV_IAM_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Einträge sehen, auf die sie Zugriff haben.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
iam_arn	text	Name des Namespace.
command_type	text	Arten von Berechtigungen. Mögliche Werte sind COPY, UNLOAD, CREATE MODEL oder EXTERNAL FUNCTION.
identity_id	Ganzzahl	Identitäts-ID. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Identitätsname.
identity_type	text	Identitätstyp. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.

Beispielabfragen

Das folgende Beispiel zeigt die Ergebnisse von SVV_IAM_PRIVILEGES.

```
SELECT * from SVV_IAM_PRIVILEGES ORDER BY IDENTITY_ID;
       iam_arn        | command_type | identity_id | identity_name | identity_type
----------------------+--------------+-------------+---------------+---------------
 default-aws-iam-role | COPY         |           0 | public        | public
 default-aws-iam-role | UNLOAD       |           0 | public        | public
 default-aws-iam-role | CREATE MODEL |           0 | public        | public
 default-aws-iam-role | EXFUNC       |           0 | public        | public
 default-aws-iam-role | COPY         |         106 | u1            | user
 default-aws-iam-role | UNLOAD       |         106 | u1            | user
 default-aws-iam-role | CREATE MODEL |         106 | u1            | user
 default-aws-iam-role | EXFUNC       |         106 | u1            | user
 default-aws-iam-role | COPY         |      118413 | r1            | role
 default-aws-iam-role | UNLOAD       |      118413 | r1            | role
 default-aws-iam-role | CREATE MODEL |      118413 | r1            | role
```

```
 default-aws-iam-role | EXFUNC       |      118413 | r1            | role
(12 rows)
```

SVV_IDENTITY_PROVIDERS

Die Ansicht SVV_IDENTITY_PROVIDERS gibt den Namen und zusätzliche Eigenschaften der Identitätsanbieter zurück. Weitere Informationen zum Erstellen eines Identitätsanbieters finden Sie unter CREATE IDENTITY PROVIDER.

SVV_IDENTITY_PROVIDERS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
uid	Ganzzahl	Die eindeutige ID des registrierten Identitätsanbieters.
Name	text	Der Name des Identitätsanbieters.
Typ	text	Der Identitätsanbietertyp.
instanceid	text	Das eindeutige Unterscheidungsmerkmal zwischen Instances desselben Typs.
namespc	text	Das Namespace-Präfix des Identitätsanbieters.
params	text	Das JSON-Objekt mit Parametern für den Identitätsanbieter.
aktiviert	bool	Gibt an, ob der Identitätsanbieter aktiviert ist.

Beispielabfragen

Führen Sie nach dem Erstellen von Identitätsanbietern eine Abfrage wie im Folgenden dargestellt aus, um die Eigenschaften der Identitätsanbieter anzuzeigen.

```
SELECT name, type, instanceid, namespc, params, enabled
FROM svv_identity_providers
ORDER BY 1;
```

Die Beispielausgabe enthält Parameterbeschreibungen.

```
       name       | type  |              instanceid              | namespc |

                                        params

                     | enabled
------------------+-------+--------------------------------------+---------
+-------------------------------------------------------------------------------------------------------------------
+---------
 rs5517_azure_idp | azure | e40d4bb2-7670-44ae-bfb8-5db013221d73 | abc     |
 {"issuer":"https://login.microsoftonline.com/e40d4bb2-7670-44ae-bfb8-5db013221d73/
v2.0", "client_id":"871c010f-5e61-4fb1-83ac-98610a7e9110", "client_secret":,
 "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift", "https://
analysis.windows.net/powerbi/connector/AWSRDS"]} | t
(1 row)
```

SVV_INTEGRATION

SVV_INTEGRATION zeigt Details zur Konfiguration von Integrationen an.

SVV_INTEGRATION ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu Null-ETL-Integrationen finden Sie unter Arbeiten mit Null-ETL-Integrationen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
integration_id	character (128)	Die Kennung, die der Integration zugeordnet ist.

Spaltenname	Datentyp	Beschreibung
target_database	character (128)	Die Datenbank in Amazon Redshift, die die Integrationsdaten empfängt.
Quelle	character (128)	Die Quelldaten für die Integration. Mögliche Typen sind u. a. `MySQL` und `PostgreSQL` .
state	character (128)	Der Zustand der Integration. Mögliche Werte sind `PendingDbConnectState` , `SchemaDiscoveryState` , `CdcRefreshState` und `ErrorState` .
current_lag	bigint	Die aktuelle Verzögerungszeit (Millisekunden) zwischen der Quelle und dem Ziel der Integration.
last_replicated_checkpoint	character (128)	Der zuletzt replizierte Checkpoint.
total_tables_replicated	Ganzzahl	Die Gesamtzahl der Tabellen, die sich derzeit im Status „repliziert“ befinden.
total_tables_failed	Ganzzahl	Die Gesamtzahl der Tabellen, die sich derzeit im Status „fehlgeschlagen“ befinden.
creation_time	Zeitstempel	Die Uhrzeit (UTC), zu der die Integration erstellt wird. Sie ist definiert als die Uhrzeit, zu der die Zieldatenbank aus der Integration erstellt wird.

Beispielabfragen

Der folgende SQL-Befehl zeigt die aktuell definierten Integrationen an.

```
select * from svv_integration;

           integration_id              | target_database | source |      state
   | current_lag |      last_replicated_checkpoint     | total_tables_replicated |
 total_tables_failed |       creation_time
```

```
--------------------------------------+----------------+--------+----------------
+------------+--------------------------------------+-----------------------
+---------------------+--------------------------
  99108e72-1cfd-414f-8cc0-0216acefac77 |     perfdb      |  MySQL | CdcRefreshState |
  56606106  | {"txn_seq":9834,"txn_id":126597515} |            152          |
 0         | 2023-09-19 21:05:27.520299
```

SVV_INTEGRATION_TABLE_STATE

SVV_INTEGRATION_TABLE_STATE zeigt Details zu Integrationsinformationen auf Tabellenebene an.

SVV_INTEGRATION_TABLE_STATE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Weitere Informationen finden Sie unter Arbeiten mit Null-ETL-Integrationen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
integration_id	character(128)	Die Kennung, die der Integration zugeordnet ist.
target_database	character(128)	Der Name der Amazon-Redshift-Datenbank.
schema_name	character(128)	Der Name des Amazon-Redshift-Schemas.
table_name	character(128)	Der Name der Tabelle.
table_state	character(128)	Der Zustand der Tabelle. Die möglichen Werte sind `Synced`, `Failed`, `Deleted`, `ResyncRequired` und `ResyncInitiated`.
table_last_replicated_checkpoint	character(128)	Die aktuellen synchronisierten Protokoll-Koordinaten.
Grund	character(256)	Der Grund für den letzten Statuswechsel. Häufige Gründe können nicht unterstützte Datentypen in Tabellen sein. Tabellen haben

Spaltenname	Datentyp	Beschreibung
		keine Primärschlüssel. Weitere Informati onen zur Behebung häufiger Probleme finden Sie unter Problembehandlung bei Null-ETL-Integrationen in Amazon Redshift.
last_updated_times tamp	Timestamp ohne Zeitzone	Die Uhrzeit (UTC), zu der die Tabelle zuletzt aktualisiert wurde.

Beispielabfragen

Der folgende SQL-Befehl zeigt das Protokoll der Integrationen an.

```
select * from svv_integration_table_state;

          integration_id              | target_database | schema_name |     table_name
   | Table_state  |table_last_replicated_checkpoint | reason | last_updated_timestamp
-------------------------------------+-----------------+-------------
+------------------+-------------+--------------------------------
+-------------------------------------
 4798e675-8f9f-4686-b05f-92c538e19629 |  sample_test2   |    sample   |
 SampleTestChannel |    Synced    |   {"txn_seq":3,"txn_id":3122}   |        |
 2023-05-12 12:40:30.656625
```

SVV_INTERLEAVED_COLUMNS

Verwenden Sie die Ansicht SVV_INTERLEAVED_COLUMNS, um festzustellen, ob eine Tabelle, die überlappende Sortierschlüssel verwendet, mit neu indiziert werden sollte VACUUM REINDEX. Weitere Informationen dazu, wie häufig VACUUM ausgeführt werden sollte und wann VACUUM REINDEX ausgeführt werden sollte, finden Sie unter Verwalten der Bereinigungszeiten.

SVV_INTERLEAVED_COLUMNS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
tbl	integer	Tabellen-ID.
col	integer	Null-basierter Index für die Spalte.
interleaved_skew	numeric(9,2)	Verhältnis, dass den Grad der Verzerrung in den überlappenden Sortierschlüsseln für eine Tabelle angibt. Ein Wert von 1,00 zeigt an, dass keine Verzerrung vorliegt, und größere Werte weisen auf größere Verzerrung hin. Tabellen mit starker Verzerrung sollten mit dem Befehl VACUUM REINDEX neu indiziert werden.
last_reindex	timestamp	Zeitpunkt der letzten Ausführung von VACUUM REINDEX für die angegebene Tabelle. Dieser Wert ist NULL, wenn eine Tabelle nie neu indiziert wurde oder wenn die zugrunde liegende Systemprotokolltabelle, STL_VACUUM, nach der letzten Neuindizierung gewechselt wurde.

Beispielabfragen

Führen Sie die folgende Abfrage aus, um Tabellen zu identifizieren, die möglicherweise neu indiziert werden müssen.

```
select tbl as tbl_id, stv_tbl_perm.name as table_name,
col, interleaved_skew, last_reindex
from svv_interleaved_columns, stv_tbl_perm
where svv_interleaved_columns.tbl = stv_tbl_perm.id
and interleaved_skew is not null;

 tbl_id | table_name | col | interleaved_skew | last_reindex
--------+------------+-----+------------------+---------------------
 100068 | lineorder  |   0 |             3.65 | 2015-04-22 22:05:45
 100068 | lineorder  |   1 |             2.65 | 2015-04-22 22:05:45
 100072 | customer   |   0 |             1.65 | 2015-04-22 22:05:45
 100072 | lineorder  |   1 |             1.00 | 2015-04-22 22:05:45
(4 rows)
```

SVV_LANGUAGE_PRIVILEGES

Verwenden Sie SVV_LANGUAGE_PRIVILEGES, um die Sprachberechtigungen anzuzeigen, die Benutzern, Rollen und Gruppen in der aktuellen Datenbank explizit gewährt werden.

SVV_LANGUAGE_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
language_name	text	Der Name der Sprache.
privilege_type	text	Der Typ der Berechtigung. Der mögliche Wert ist USAGE.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenid entitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_LANGUAGE_PRIVILEGES angezeigt.

```
SELECT language_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_language_privileges
WHERE identity_name IN ('role1', 'reguser');

 language_name | privilege_type | identity_name | identity_type | admin_option
---------------+----------------+---------------+---------------+--------------
   exfunc      |     USAGE      |    reguser    |     user      |    False
   exfunc      |     USAGE      |     role1     |     role      |    False
   plpythonu   |     USAGE      |    reguser    |     user      |    False
```

SVV_MASKING_POLICY

Verwenden Sie SVV_MASKING_POLICY, um alle in dem Cluster erstellten Maskierungsrichtlinien anzuzeigen.

Nur Superuser und Benutzer mit der Rolle `sys:secadmin` können SVV_MASKING_POLICY einsehen. Regulären Benutzern werden 0 Zeilen angezeigt.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
policy_database	text	Der Name der Datenbank, in der die Maskierungsrichtlinie erstellt wurde.
policy_name	text	Der Name der Maskierungsrichtlinie.
input_columns	text	Die in der WITH-Klausel der Anweisung CREATE POLICY angegebenen Attribute.
policy_expression	text	Der in der Richtlinie verwendete Maskierungsausdruck.
policy_modified_by	text	Der Name des Benutzers, der die Richtlinie zuletzt geändert hat.

Spaltenname	Datentyp	Beschreibung
policy_modified_time	Zeitstempel	Der Zeitstempel der Erstellung oder letzten Änderung der Richtlinie.

SVV_ML_MODEL_INFO

Informationen zum aktuellen Zustand des Machine-Learning-Modells.

SVV_ML_MODEL_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	char(128)	Die Datenbank des Modells.
schema_name	char(128)	Das Schema des Modells.
user_name	char(128)	Der Besitzer des Modells.
model_name	char(128)	Der Name des Modells
life_cycle	char(20)	Der Lebenszyklusstatus des Modells.
is_refreshable	integer	Der Status des Modells, ob es aktualisiert werden kann, ob ursprüngliche Tabellen und Spalten in der Trainingsabfrage noch vorhanden sind und ob der Benutzer weiterhin über die Berechtigungen für sie verfügt. Mögliche Werte sind: 1 (aktualisierbar) und 0 (nicht aktualisierbar).
model_state	char(128)	Der aktuelle Status des Modells.

Beispielabfrage

Die folgende Abfrage zeigt den aktuellen Status von Machine-Learning-Modellen an.

```
SELECT schema_name, model_name, model_state
FROM svv_ml_model_info;

 schema_name |        model_name             |             model_state
-------------+-------------------------------+-----------------------------------
 public      | customer_churn_auto_model     | Train Model On SageMaker In Progress
 public      | customer_churn_xgboost_model | Model is Ready
(2 row)
```

SVV_ML_MODEL_PRIVILEGES

Verwenden Sie SVV_ML_MODEL_PRIVILEGES, um die Berechtigungen für Machine-Learning-Modelle anzuzeigen, die Benutzern, Rollen und Gruppen im Cluster explizit gewährt werden.

SVV_ML_MODEL_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
namespace_name	text	Der Name des Namespace, in dem ein angegebenes Machine-Learning-Modell vorhanden ist.
model_name	text	Der Name des Machine-Learning-Modells.
model_version	Ganzzahl	Die Versionsnummer des Modells.
privilege_type	text	Der Typ der Berechtigung. Der mögliche Wert ist EXECUTE.

Spaltenname	Datentyp	Beschreibung
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenidentitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_ML_MODEL_PRIVILEGES angezeigt.

```
SELECT
 namespace_name,model_name,model_version,privilege_type,identity_name,identity_type,admin_optio
 FROM svv_ml_model_privileges
WHERE model_name = 'test_model';

 namespace_name | model_name | model_version | privilege_type |  identity_name |
 identity_type | admin_option
----------------+------------+---------------+----------------+----------------
+---------------+--------------
      public    | test_model |       1       |    EXECUTE     |     reguser    |
 user      |    False
      public    | test_model |       1       |    EXECUTE     |     role1      |
 role      |    False
```

SVV_MV_DEPENDENCY

Die Tabelle SVV_MV_DEPENDENCY zeigt die Abhängigkeiten materialisierter Ansichten von anderen materialisierten Ansichten in Amazon Redshift.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

SVV_MV_DEPENDENCY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	char(128)	Die Datenbank, die die angegebene materialisierte Ansicht enthält.
schema_name	char(128)	Das Schema der materialisierten Ansicht.
Name	char(128)	Der Name der materialisierten Ansicht.
dependent_database_name	char(128)	Die Datenbank der materialisierten Ansicht, von der diese materialisierte Ansicht abhängt.
dependent_schema_name	char(128)	Das Schema der materialisierten Ansicht, von dem diese materialisierte Ansicht abhängt.
dependent_name	char(128)	Der Name der materialisierten Ansicht, von dem diese materialisierte Ansicht abhängt.

Beispielabfrage

Die folgende Abfrage gibt eine Ausgabezeile zurück, die angibt, dass die materialisierte Ansicht `mv_over_foo` die materialisierte Ansicht `mv_foo` in ihrer Definition als Abhängigkeit verwendet.

```
CREATE SCHEMA test_ivm_setup;
CREATE TABLE test_ivm_setup.foo(a INT);
```

```
CREATE MATERIALIZED VIEW test_ivm_setup.mv_foo AS SELECT * FROM test_ivm_setup.foo;
CREATE MATERIALIZED VIEW test_ivm_setup.mv_over_foo AS SELECT * FROM
 test_ivm_setup.mv_foo;

SELECT * FROM svv_mv_dependency;

 database_name | schema_name          | name        | dependent_database_name |
 dependent_schema_name     | dependent_name
---------------+----------------------+-------------+-------------------------
+---------------------------+----------
 dev           | test_ivm_setup       | mv_over_foo |                     dev |
 test_ivm_setup            | mv_foo
```

SVV_MV_INFO

Die Tabelle SVV_MV_INFO enthält eine Zeile für jede materialisierte Ansicht, Angaben, ob die Daten veraltet sind, und Statusinformationen.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

SVV_MV_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	char(128)	Die Datenbank, die die materialisierte Ansicht enthält.
schema_name	char(128)	Das Schema der Datenbank.
user_name	char(128)	Der Benutzer, dem die materialisierte Ansicht gehört.
Name	char(128)	Der Name der materialisierten Ansicht.
is_stale	char(1)	Ein `t` gibt an, dass die materialisierte Ansicht veraltet ist. Eine veraltete materialisierte Ansicht

Spaltenname	Datentyp	Beschreibung
		ist eine, bei der zwar die Basistabellen aktualisiert wurden, aber nicht die materialisierte Ansicht. Diese Informationen sind möglicherweise nicht korrekt, wenn seit dem letzten Neustart keine Aktualisierung durchgeführt wurde.
state	integer	Der Status der materialisierten Ansicht wie folgt: • 0 – Die materialisierte Ansicht wird vollständig neu berechnet, wenn sie aktualisiert wird. • 1 – Die materialisierte Ansicht ist inkrementell. • 101 – Die materialisierte Ansicht kann aufgrund einer gelöschten Spalte nicht aktualisiert werden. Diese Einschränkung gilt auch dann, wenn die Spalte nicht in der materialisierten Ansicht verwendet wird. • 102 – Die materialisierte Ansicht kann aufgrund eines geänderten Spaltentyps nicht aktualisiert werden. Diese Einschränkung gilt auch dann, wenn die Spalte nicht in der materialisierten Ansicht verwendet wird. • 103 – Die materialisierte Ansicht kann aufgrund einer umbenannten Tabelle nicht aktualisiert werden. • 104 – Die materialisierte Ansicht kann aufgrund einer umbenannten Spalte nicht aktualisiert werden. Diese Einschränkung gilt auch dann, wenn die Spalte nicht in der materialisierten Ansicht verwendet wird. • 105 – Die materialisierte Ansicht kann aufgrund eines umbenannten Schemas nicht aktualisiert werden.

Spaltenname	Datentyp	Beschreibung
autorewrite	char(1)	Ein `t` gibt an, dass die materialisierte Ansicht für das automatische Umschreiben von Abfragen berechtigt ist.
autorefresh	char(1)	Ein `t` gibt an, dass die materialisierte Ansicht automatisch aktualisiert werden kann.

Beispielabfrage

Führen Sie die folgende Abfrage aus, um den Status aller materialisierten Ansichten anzuzeigen.

```
select * from svv_mv_info;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück.

```
database_name |       schema_name       | user_name |   name   |  is_stale | state |
 autorefresh | autorewrite
--------------+-------------------------+-----------+---------+-----------+-------
+-------------+----------------
 dev          | test_ivm_setup          | catch-22  | mv       |   f       |     1 |
       1 |           0
 dev          | test_ivm_setup          | lotr      | old_mv   |   t       |     1 |
       0 |           1
```

SVV_QUERY_INFLIGHT

Verwenden Sie die Ansicht SVV_QUERY_INFLIGHT, um festzustellen, welche Abfragen derzeit auf der Datenbank ausgeführt werden. Diese Ansicht verbindet STV_INFLIGHT mit STL_QUERYTEXT. SVV_QUERY_INFLIGHT zeigt keine Abfragen nur mit Leader-Knoten. Weitere Informationen finden Sie unter Exklusive Führungsknotenfunktionen.

SVV_QUERY_INFLIGHT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> Diese Ansicht ist nur verfügbar, wenn bereitgestellte Cluster abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
slice	integer	Slice, auf dem die Abfrage ausgeführt wird.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
pid	integer	Prozess-ID. Alle Abfragen in einer Sitzung werden in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen. Mit dieser Spalte können Sie eine Verbindung zur Tabelle STL_ERROR herstellen.
starttime	timestamp	Der Zeitpunkt des Beginns der Abfrage.
suspended	integer	Ob die Abfrage ausgesetzt wurde: **0** = false; **1** = true.
Text	character(200)	Abfragetext, in Schritten zu 200 Zeichen.
sequence	integer	Sequenznummer für Segmente von Abfrageanweisungen.

Beispielabfragen

Die Beispielausgabe unten zeigt zwei derzeit ausgeführte Abfragen, die Abfrage SVV_QUERY_INFLIGHT und Abfrage 428, die in der Tabelle auf drei Zeilen aufgeteilt ist. (Die Tabellen für die Start- und die Endzeit sind in dieser Beispielausgabe verkürzt.)

```
select slice, query, pid, starttime, suspended, trim(text) as statement, sequence
from svv_query_inflight
```

```
order by query, sequence;

slice|query| pid  |      starttime       |suspended| statement | sequence
-----+-----+------+----------------------+---------+-----------+---------
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | select ...|    0
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | enueid ...|    1
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | atname,...|    2
1012 | 429 | 1608 | 2012-04-10 13:53:... |       0 | select ...|    0
(4 rows)
```

SVV_QUERY_STATE

Verwenden Sie SVV_QUERY_STATE zur Anzeige von Informationen zur Laufzeit derzeit laufender Abfragen.

Die Ansicht SVV_QUERY_STATE enthält einen Teil der Daten der Tabelle STV_EXEC_STATE.

SVV_QUERY_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

> **Note**
>
> Diese Ansicht ist nur verfügbar, wenn bereitgestellte Cluster abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.

Spaltenname	Datentyp	Beschreibung
seg	Ganzzahl	Nummer des derzeit ausgeführten Abfragesegments. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt.
Schritt	Ganzzahl	Nummer des derzeit ausgeführten Abfrageschrittes. Ein Schritt ist die kleinste Einheit der Abfragelaufzeit. Jeder Schritt steht für eine diskrete Arbeitseinheit, etwa das Scannen einer Tabelle, die Ausgabe von Ergebnissen oder das Sortieren von Daten.
maxtime	Intervall	Maximale Zeitdauer (in Mikrosekunden) für die Ausführung dieses Schritts.
avgtime	Intervall	Durchschnittliche Zeitdauer (in Mikrosekunden) für die Ausführung dieses Schritts.
rows	bigint	Anzahl der vom laufenden Schritt produzierten Zeilen.
bytes	bigint	Anzahl der vom laufenden Schritt produzierten Bytes.
cpu	bigint	Zur internen Verwendung.
memory	bigint	Zur internen Verwendung.
rate_row	double precision	ows-per-second R-Rate seit dem Start der Abfrage, berechnet durch Summieren der Zeilen und Division durch die Anzahl der Sekunden zwischen dem Start der Abfrage und der aktuellen Uhrzeit.
rate_byte	double precision	ytes-per-second B-Rate seit dem Start der Abfrage, berechnet durch Summieren der Byte und Division durch die Anzahl der Sekunden zwischen dem Start der Abfrage und dem aktuellen Zeitpunkt.

Spaltenname	Datentyp	Beschreibung
label	character(25)	Abfragebeschriftung: ein Name für den Schritt, wie etwa `scan` oder `sort`.
is_diskbased	character(1)	Ob dieser Schritt der Abfrage als datenträgerbasierte Operation läuft: true (**t**) oder false (**f**). Nur bestimmte Schritte, wie etwa Hash-, Sortierungs- oder Aggregierungsschritte, können auf die Festplatte übertragen werden. Viele Arten von Schritten werden immer im Arbeitsspeicher ausgeführt.
workmem	bigint	Größe des Arbeitsspeichers (in Byte), der dem Abfrageschritt zugewiesen wurde.
num_parts	integer	Anzahl der Partitionen, in die eine Hash-Tabelle während eines Hash-Schrittes unterteilt wurde. Ein positiver Wert in dieser Spalte bedeutet nicht, dass der Hash-Schritt als festplattenbasierte Operation ausgeführt wird. Prüfen Sie den Wert in der Spalte IS_DISKBASED, um zu sehen, ob der Hash-Schritt festplattenbasiert war.
is_rrscan	character(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein Scan mit Bereichseinschränkung durchgeführt wurde. Der Standardwert ist „false“ (**f**).
is_delayed_scan	character(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein verzögerter Scan durchgeführt wurde. Der Standardwert ist „false“ (**f**).

Beispielabfragen

Ermittlung der Verarbeitungszeit einer Abfrage nach Schritten

Die folgende Abfrage zeigt, wie lange die Ausführung jedes Schritts der Abfrage mit der Abfrage-ID 279 dauerte und wie viele Datenzeilen Amazon Redshift verarbeitet hat:

```
select query, seg, step, maxtime, avgtime, rows, label
from svv_query_state
where query = 279
```

```
order by query, seg, step;
```

Diese Abfrage ruft die Verarbeitungsinformationen zu Abfrage 279 aus, wie die folgende Beispielausgabe zeigt:

```
query |   seg   | step | maxtime | avgtime |  rows   | label
------+---------+------+---------+---------+---------+------------------
  279 |       3 |    0 | 1658054 | 1645711 | 1405360 | scan
  279 |       3 |    1 | 1658072 | 1645809 |       0 | project
  279 |       3 |    2 | 1658074 | 1645812 | 1405434 | insert
  279 |       3 |    3 | 1658080 | 1645816 | 1405437 | distribute
  279 |       4 |    0 | 1677443 | 1666189 | 1268431 | scan
  279 |       4 |    1 | 1677446 | 1666192 | 1268434 | insert
  279 |       4 |    2 | 1677451 | 1666195 |       0 | aggr
(7 rows)
```

Ermittlung, ob zurzeit aktive Abfragen auf dem Datenträger ausgeführt werden

Die folgende Abfrage zeigt, ob derzeit aktive Abfragen auf der Festplatte ausgeführt werden:

```
select query, label, is_diskbased from svv_query_state
where is_diskbased = 't';
```

Diese Beispielausgabe zeigt alle derzeit auf der Festplatte ausgeführten aktiven Abfragen:

```
 query | label        | is_diskbased
-------+--------------+--------------
1025   | hash tbl=142 |      t
(1 row)
```

SVV_REDSHIFT_COLUMNS

Verwenden Sie SVV_REDSHIFT_COLUMNS, um eine Liste aller Spalten anzuzeigen, auf die ein Benutzer Zugriff hat. Dieser Satz von Spalten enthält die Spalten im Cluster und die Spalten aus Datashares, die von Remote-Clustern bereitgestellt werden.

SVV_REDSHIFT_COLUMNS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Der Name der Datenbank, in der die Tabelle existiert, die die Spalten enthält.
schema_name	varchar(128)	Der Name des Schemas für die Tabelle.
table_name	varchar(128)	Der Name der Tabelle.
column_name	varchar(128)	Der Name einer Spalte.
ordinal_position	integer	Die Position der Spalten in der Tabelle.
data_type	varchar(32)	Der Datentyp der Spalte.
column_default	varchar(4000)	Der Standardwert der Spalte.
is_nullable	varchar(3)	Ein Wert, der angibt, ob eine Spalte den Wert Null enthalten kann. Mögliche Werte sind `yes`, `no` und " " (eine leere Zeichenfolge, die keine Information darstellt).
encoding	varchar(128)	Der Kodierungstyp der Spalte.
distkey	Boolean	Ein Wert, der true ist, wenn diese Spalte der Verteilun gsschlüssel für die Tabelle ist, andernfalls false.
sortkey	integer	Ein Wert, der die Reihenfolge der Spalte im Sortierschlüssel angibt.

Spaltenname	Datentyp	Beschreibung
		Wenn die Tabelle einen zusammengesetzten Sortiersc hlüssel verwendet, haben alle zu dem Sortierschlüssel gehörenden Spalten einen positiven Wert, der die Position der Spalte in dem Sortierschlüssel angibt. Wenn eine Tabelle einen überlappenden Sortiersc hlüssel verwendet, hat jede zu dem Sortierschlüssel gehörende Spalte einen positiven oder negativen Wert. Hier gibt der absolute Wert die Position der Spalte im Sortierschlüssel an. Wenn `sortkey` 0 ist, gehört die Spalte nicht zu einem Sortierschlüssel.
column_acl	varchar(128)	Eine Zeichenfolge, die die Berechtigungen für den angegebenen Benutzer oder die Benutzergruppe für die Spalte definiert.
remarks	varchar(256)	Anmerkungen.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_REDSHIFT_COLUMNS zurückgegeben.

```
SELECT *
```

```
FROM svv_redshift_columns
WHERE database_name = 'tickit_db'
    AND TABLE_NAME = 'tickit_sales_redshift'
ORDER BY COLUMN_NAME,
    TABLE_NAME,
    database_name
LIMIT 5;

database_name | schema_name |       table_name      | column_name | ordinal_position |
 data_type | column_default | is_nullable | encoding | distkey | sortkey | column_acl
 | remarks
--------------+-------------+-----------------------+-------------+------------------
+-----------+----------------+-------------+----------+---------+---------
+-------------+--------
   tickit_db  |   public    | tickit_sales_redshift |   buyerid   |        4         |
 integer  |                |      NO     |   az64   |  False  |    0    |             |
   tickit_db  |   public    | tickit_sales_redshift |  commission |        9         |
 numeric  |      (8,2)     |     YES     |   az64   |  False  |    0    |             |
   tickit_db  |   public    | tickit_sales_redshift |    dateid   |        6         |
 smallint |                |      NO     |   none   |  False  |    1    |             |
   tickit_db  |   public    | tickit_sales_redshift |   eventid   |        5         |
 integer  |                |      NO     |   az64   |  False  |    0    |       |
   tickit_db  |   public    | tickit_sales_redshift |   listid    |        2         |
 integer  |                |      NO     |   az64   |  True   |    0    |             |
```

SVV_REDSHIFT_DATABASES

Verwenden Sie SVV_ REDSHIFT_DATABASES, um eine Liste aller Datenbanken anzuzeigen, auf die ein Benutzer Zugriff hat. Dazu gehören die Datenbanken auf dem Cluster und die Datenbanken, die aus Datashares erstellt werden, die von Remote-Clustern bereitgestellt werden.

SVV_REDSHIFT_DATABASES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Name der Datenbank.

Spaltenname	Datentyp	Beschreibung
database_owner	integer	Die Benutzer-ID des Datenbankeigentümers.
database_type	varchar(32)	Der Typ der Datenbank. Mögliche Typen sind lokale oder freigegebene Datenbanken.
database_acl	varchar(128)	Diese Information ist nur für die interne Verwendung gedacht.
database_options	varchar(128)	Die Eigenschaften der Datenbank.
database_isolation_level	varchar(128)	Die Isolationsstufe der Datenbank. Mögliche Werte sind u. a. `Snapshot Isolation` und `Serializable`.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_REDSHIFT_DATABASES zurückgegeben.

```
select database_name, database_owner, database_type, database_options,
 database_isolation_level
from  svv_redshift_databases;


database_name | database_owner | database_type | database_options |
 database_isolation_level
--------------+----------------+---------------+------------------+------------------
   dev        |  1             | local         | NULL             | Serializable
```

SVV_REDSHIFT_FUNCTIONS

Verwenden Sie SVV_REDSHIFT_FUNCTIONS, um eine Liste aller Funktionen anzuzeigen, auf die ein Benutzer Zugriff hat. Dieser Satz von Funktionen enthält die Funktionen im Cluster und die Funktionen aus Datashares, die von Remote-Clustern bereitgestellt werden.

SVV_REDSHIFT_FUNCTIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Der Name der Datenbank, in der sich der Cluster mit diesen Funktionen befindet.
schema_name	varchar(128)	Der Name des Schemas, das eine bestimmte Funktion angibt.
function_name	varchar(128)	Der Name einer angegebenen Funktion.
function_type	varchar(128)	Der Typ der Funktion. Mögliche Werte sind reguläre Funktionen, Aggregatf unktionen und gespeicherte Prozeduren.
argument_type	varchar(512)	Ein String, der den Typ des Eingabeargumente s einer Funktion darstellt.
result_type	varchar(128)	Der Datentyp des Rückgabew erts einer Funktion.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_REDSHIFT_FUNCTIONS zurückgegeben.

```
SELECT *
FROM svv_redshift_functions
WHERE database_name = 'tickit_db'
    AND SCHEMA_NAME = 'public'
ORDER BY function_name
LIMIT 5;

database_name | schema_name |      function_name    |  function_type   |
 argument_type  | result_type
--------------+-------------+-----------------------+------------------
+------------------+-------------
   tickit_db  |    public   |     shared_function   | REGULAR FUNCTION | integer,
 integer |   integer
```

SVV_REDSHIFT_SCHEMA_QUOTA

Zeigt das Kontingent und die aktuelle Datenträgernutzung für jedes Schema in einer Datenbank an.

SVV_REDSHIFT_SCHEMA_QUOTA ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Diese Ansicht ist verfügbar, wenn bereitgestellte Cluster oder Redshift-Serverless-Arbeitsgruppen abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	character(128)	Die Datenbank, die das Schema enthält.
schema_name	character(128)	Der Name des Schemas.
schema_owner	integer	Die interne Benutzer-ID des Schemabesitzers.

Spaltenname	Datentyp	Beschreibung
quota	integer	Die Menge an Speicherplatz (in MB), die das Schema verwenden kann.
disk_usage	integer	Der Speicherplatz (in MB), der zurzeit vom Schema verwendet wird.

Beispielabfrage

Im folgenden Beispiel werden das Kontingent und die aktuelle Speichernutzung für das Schema mit der Bezeichnung `sales_schema` angezeigt.

```
SELECT TRIM(SCHEMA_NAME) "schema_name", QUOTA, disk_usage FROM
 svv_redshift_schema_quota
WHERE SCHEMA_NAME = 'sales_schema';


schema_name   | quota | disk_usage
--------------+-------+-----------
sales_schema  | 2048  | 30
```

SVV_REDSHIFT_SCHEMAS

Verwenden Sie SVV_REDSHIFT_SCHEMAS, um eine Liste aller Schemata anzuzeigen, auf die ein Benutzer Zugriff hat. Dieser Satz von Schemata enthält die Schemata im Cluster und die Schemata aus Datashares, die von Remote-Clustern bereitgestellt werden.

SVV_REDSHIFT_SCHEMAS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Der Name der Datenbank, in der ein spezifiziertes Schema vorhanden ist.
schema_name	varchar(128)	Der Namespace- oder Schemaname.
schema_owner	integer	Die interne Benutzer-ID des Schemabesitzers.
schema_type	varchar(16)	Der Typ des Schemas. Mögliche Werte sind freigegeb ene und lokale Schemata.
schema_acl	varchar(128)	Die Zeichenfolge, die die Berechtigungen für den angegebenen Benutzer oder die Benutzergruppe für das Schema definiert.
schema_option	varchar(128)	Die Optionen des Schemas.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_REDSHIFT_SCHEMAS zurückgegeben.

```
SELECT *
FROM svv_redshift_schemas
WHERE database_name = 'tickit_db'
ORDER BY database_name,
    SCHEMA_NAME;

database_name |    schema_name     | schema_owner | schema_type | schema_acl |
 schema_option
```

```
-------------+--------------------+-------------+------------+-----------
+---------------
   tickit_db  |       public       |       1     |    shared  |           |
```

SVV_REDSHIFT_TABLES

Verwenden Sie SVV_REDSHIFT_TABLES, um eine Liste aller Tabellen anzuzeigen, auf die ein Benutzer Zugriff hat. Dieser Satz von Tabellen enthält die Tabellen im Cluster und die Tabellen aus Datashares, die von Remote-Clustern bereitgestellt werden.

SVV_REDSHIFT_TABLES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database_name	varchar(128)	Der Name der Datenbank, in der eine angegebene Tabelle vorhanden ist.
schema_name	varchar(128)	Der Name des Schemas für die Tabelle.
table_name	varchar(128)	Der Name der Tabelle.
table_type	varchar(128)	Der Typ der Tabelle. Mögliche Werte sind Ansichten und Tabellen.
table_acl	varchar(128)	Die Zeichenfolge, die die Berechtigungen für den angegebenen Benutzer oder die Benutzergruppe für die Tabelle definiert.
remarks	varchar(128)	Anmerkungen.
table_owner	varchar(128)	Der Eigentümer der Tabelle.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_REDSHIFT_TABLES zurückgegeben.

```
SELECT *
FROM svv_redshift_tables
WHERE database_name = 'tickit_db' AND TABLE_NAME LIKE 'tickit_%'
ORDER BY database_name,
TABLE_NAME;

database_name | schema_name |         table_name       | table_type | table_acl |
 remarks | table_owner
--------------+-------------+--------------------------+------------+-----------
+---------+-----------
   tickit_db  |    public   | tickit_category_redshift |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_date_redshift   |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_event_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |  tickit_listing_redshift |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_sales_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_users_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_venue_redshift  |    TABLE   |           |
```

Wenn der Wert table_acl Null ist, wurden keine Zugriffsrechte explizit für die entsprechende Tabelle gewährt.

SVV_RELATION_PRIVILEGES

Verwenden Sie SVV_RELATION_PRIVILEGES, um die Beziehungsberechtigungen (Tabellen und Ansichten) anzuzeigen, die Benutzern, Rollen und Gruppen in der aktuellen Datenbank explizit gewährt werden.

SVV_RELATION_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der SYSLOG ACCESS UNRESTRICTED-Berechtigung

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen. Weitere Hinweise zur Sichtbarkeit von Daten finden Sie unter. Sichtbarkeit der Daten in Systemtabellen und Ansichten

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
namespace_name	text	Der Name des Namespace, in dem eine angegebene Beziehung vorhanden ist.
relation_name	text	Der Name der Beziehung.
privilege_type	text	Der Typ der Berechtigung. Mögliche Werte sind INSERT, SELECT, UPDATE, DELETE, REFERENCES und DROP.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenidentitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_RELATION_PRIVILEGES angezeigt.

```
SELECT
 namespace_name,relation_name,privilege_type,identity_name,identity_type,admin_option
 FROM svv_relation_privileges
WHERE relation_name = 'orders' AND privilege_type = 'SELECT';

 namespace_name | relation_name | privilege_type |  identity_name | identity_type |
 admin_option
```

```
---------------+--------------+---------------+---------------+--------------
+-------------
     public     |    orders     |     SELECT     |    reguser     |     user      |
 False
     public     |    orders     |     SELECT     |     role1      |     role      |
 False
```

SVV_RLS_APPLIED_POLICY

Verwenden Sie SVV_RLS_APPLIED_POLICY, um die Anwendung von RLS-Richtlinien (Row-Level Security) auf Abfragen zu verfolgen, die auf geschützte RLS-Relationen verweisen.

SVV_RLS_APPLIED_POLICY ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Rolle `sys:operator`
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Beachten Sie, dass sys:secadmin diese Systemberechtigung nicht erteilt wird.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
username	text	Der Name des Benutzers, der die Abfrage ausgeführt hat.
query	Ganzzahl	Die ID der Abfrage.
xid	long	Der Kontext der Transaktion.
pid	Ganzzahl	Der führende Prozess, der die Abfrage ausführt.
recordtime	time	Die Zeit, zu der die Abfrage aufgezeichnet wurde.
command	char(1)	Der Befehl, für den die RLS-Richtlinie angewendet wurde. Mögliche Werte sind k für unbekannt, s für auswählen, u für aktualisieren, i für einfügen, y für Utility und d für löschen.
datname	text	Der Name der Datenbank der Relation, der die RLS-Richtlinie zugeordnet ist.

Spaltenname	Datentyp	Beschreibung
relschema	text	Der Name des Schemas der Relation, an die die RLS-Richtlinie angefügt ist.
relname	text	Der Name der Relation, der die RLS-Richtlinie zugeordnet ist.
polname	text	Der Name der RLS-Richtlinie, die der Relation angefügt ist.
poldefault	char(1)	Die Standardeinstellung der RLS-Richtlinie, die der Relation zugeordnet ist. Mögliche Werte sind `f` für false, wenn die Standardr ichtlinie false angewendet wurde, und `t` für true, wenn die Standardrichtlinie true angewendet wurde.

Beispielabfrage

Das folgende Beispiel zeigt das Ergebnis von SVV_RLS_APPLIED_POLICY. Um die SVV_RLS_APPLIED_POLICY abzufragen, müssen Sie über die Berechtigung ACCESS SYSTEM TABLE verfügen.

```
-- Check what RLS policies were applied to the run query.
SELECT username, command, datname, relschema, relname, polname, poldefault
FROM svv_rls_applied_policy
WHERE datname = CURRENT_DATABASE() AND query = PG_LAST_QUERY_ID();

 username | command |  datname  | relschema |          relname         |      polname
  | poldefault
----------+---------+-----------+-----------+--------------------------
+-----------------+------------
   molly  |    s    | tickit_db |   public  | tickit_category_redshift |
 policy_concerts |
```

SVV_RLS_ATTACHED_POLICY

Verwenden Sie SVV_RLS_ATTACHED_POLICY, um eine Liste aller Relationen und Benutzer anzuzeigen, denen eine oder mehrere RLS-Richtlinien auf Zeilenebene an die aktuell verbundene Datenbank angefügt sind.

Nur Benutzer mit der Rolle sys:secadmin können diese Ansicht abfragen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
relschema	text	Der Name des Schemas der Relation, an die die RLS-Richtlinie angefügt ist.
relname	text	Der Name der Relation, der die RLS-Richtlinie zugeordnet ist.
relkind	text	Der Typ des Objekts, z. B. Tabelle.
polname	text	Der Name der RLS-Richtlinie, die der Relation angefügt ist.
grantor	text	Der Name des Benutzers, der diese Richtlinie angefügt hat.
grantee	text	Der Name des Benutzers oder der Rolle, dem/der diese Richtlinie zugeordnet wurde.
granteekind	text	Die Art des Berechtigungsempfängers. Mögliche Werte sind Benutzer und Rolle.
is_pol_on	boolesch	Der Parameter, der angibt, ob eine RLS-Richtlinie für eine Tabelle ein- oder ausgeschaltet ist. Mögliche Werte sind true und false.
is_rls_on	boolesch	Der Parameter, der angibt, ob eine RLS-Richtlinie für eine Tabelle ein- oder ausgeschaltet ist. Mögliche Werte sind true und false.
rls_conjunction_type	character (3)	Der Parameter, der angibt, ob Relationen RLS-Richtlinien mit `and` oder `or` kombinieren.

Beispielabfrage

Das folgende Beispiel zeigt das Ergebnis von SVV_RLS_ATTACHED_POLICY.

```
--Inspect the policy in SVV_RLS_ATTACHED_POLICY
SELECT * FROM svv_rls_attached_policy;

 relschema |        relname           | relkind |     polname     | grantor | grantee
 | granteekind | is_pol_on | is_rls_on | rls_conjuntion_type
```

```
----------+-------------------------+---------+----------------+---------+----------
+------------+----------+----------+--------------------
 public    | tickit_category_redshift |  table  | policy_concerts |   bob   |  analyst
 |    role     |    True   |    True   |      and
 public    | tickit_category_redshift |  table  | policy_concerts |   bob   |  dbadmin
 |    role     |    True   |    True   |      and
```

SVV_RLS_POLICY

Verwenden Sie SVV_RLS_POLICY, um eine Liste aller RLS-Richtlinien anzuzeigen, die im Amazon-Redshift-Cluster erstellt wurden.

SVV_RLS_POLICY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
poldb	text	Der Name der Datenbank, in der die RLS-Richtlinie erstellt wird.
polname	text	Der Name der RLS-Richtlinie.
polalias	text	Der in der Richtliniendefinition verwendete Tabellenalias.
polatts	text	Die für die Richtliniendefinition bereitgestellten Attribute.
polqual	text	Die in der USING-Klausel der CREATE POLICY-Anweisung angegebene Richtlinienbedingung.
polenabled	boolesch	Gibt an, ob die Richtlinie global aktiviert ist.
polmodifiedby	text	Der Name des Benutzers, der die Richtlinie zuletzt erstellt oder geändert hat.
polmodifiedtime	Zeitstempel	Der Zeitstempel der Erstellung oder letzten Änderung der Richtlinie.

Beispielabfrage

Das folgende Beispiel zeigt das Ergebnis von SVV_RLS_POLICY.

```
-- Create some policies.
CREATE RLS POLICY pol1 WITH (a int) AS t USING ( t.a IS NOT NULL );
CREATE RLS POLICY pol2 WITH (c varchar(10)) AS t USING ( c LIKE '%public%');

-- Inspect the policy in SVV_RLS_POLICY
SELECT * FROM svv_rls_policy;

 poldb | polname | polalias |                     polatts                      |
         polqual                | polenabled | polmodifiedby |   polmodifiedtime
-------+---------+----------+--------------------------------------------------
+--------------------------------------+------------+---------------
+---------------------
 my_db | pol1    | t        | [{"colname":"a","type":"integer"}]               |
 "t"."a" IS NOT NULL                  | t          | policy_admin  | 2022-02-11
 14:40:49
 my_db | pol2    | t        | [{"colname":"c","type":"character varying(10)"}] |
 "t"."c" LIKE CAST('%public%' AS TEXT) | t          | policy_admin  | 2022-02-11
 14:41:28
```

SVV_RLS_RELATION

Verwenden Sie SVV_RLS_RELATION, um eine Liste aller Relationen anzuzeigen, die RLS-geschützt sind.

SVV_RLS_RELATION ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
datname	text	Der Name der Datenbank, die die Relation enthält.
relschema	text	Der Name des Schemas, das die Relation enthält.
relname	text	Der Name der Beziehung.

Spaltenname	Datentyp	Beschreibung
relkind	text	Der Typ der Relation, z. B. Tabellen oder Ansichten.
is_rls_on	boolesch	Der Parameter, der angibt, ob die Relation RLS-geschützt ist.
is_rls_datashare_on	boolesch	Der Parameter, der angibt, ob die Relation über Datashares RLS-geschützt ist.
rls_conjunction_type	character(3)	Der Parameter, der angibt, ob Relationen RLS-Richtlinien mit `and` oder `or` kombinieren.
rls_datashare_conjunction_type	character(3)	Der Parameter, der angibt, ob Relationen RLS-Richtlinien mit `and` oder `or` über Datashares kombinieren.

Beispielabfrage

Das folgende Beispiel zeigt das Ergebnis von SVV_RLS_RELATION.

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON FOR DATASHARES;


--Inspect RLS state on the relations using SVV_RLS_RELATION.
SELECT datname, relschema, relname, relkind, is_rls_on, is_rls_datashare_on FROM
 svv_rls_relation ORDER BY relname;

  datname  | relschema |        relname           | relkind | is_rls_on |
 is_rls_datashare_on | rls_conjunction_type | rls_datashare_conjunction_type
-----------+-----------+--------------------------+---------+-----------
+--------------------+----------------------+--------------------------------
 tickit_db |   public  | tickit_category_redshift |  table  |      t    |           t
       |          and         |              and
(1 row)
```

SVV_ROLE_GRANTS

Verwenden Sie SVV_ROLE_GRANTS, um eine Liste der Rollen anzuzeigen, denen explizit Rollen im Cluster gewährt werden.

SVV_ROLE_GRANTS ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
role_id	Ganzzahl	Die ID der Rolle.
role_name	text	Der Name der Rolle.
granted_role_id	Ganzzahl	Die ID der gewährten Rolle.
granted_role_name	text	Der Name der gewährten Rolle.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_ROLE_GRANTS zurückgegeben.

```
GRANT ROLE role1 TO ROLE role2;
GRANT ROLE role2 TO ROLE role3;

SELECT role_name, granted_role_name FROM svv_role_grants;

 role_name |  granted_role_name
-----------+--------------------
   role2   |      role1
   role3   |      role2
(2 rows)
```

SVV_ROLES

Verwenden Sie SVV_ROLES, um eine Liste der Rollen anzuzeigen, auf die ein Benutzer Zugriff hat.

SVV_ROLES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
role_id	Ganzzahl	Die Rollen-ID.
role_name	text	Der Name der Rolle.
role_owner	text	Der Name des Inhabers der Rolle.
external_id	text	Die eindeutige Kennung der Rolle beim externen Identität sanbieter.

Beispielabfrage

Im folgenden Beispiel wird die Ausgabe von SVV_ROLES zurückgegeben.

```
SELECT role_name,role_owner FROM svv_roles WHERE role_name IN ('role1', 'role2');

 role_name | role_owner
-----------+------------
   role1   | superuser
   role2   | superuser
```

SVV_SCHEMA_PRIVILEGES

Verwenden Sie SVV_SCHEMA_PRIVILEGES, um die Schemaberechtigungen anzuzeigen, die Benutzern, Rollen und Gruppen in der aktuellen Datenbank explizit gewährt werden.

SVV_SCHEMA_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
namespace_name	text	Der Name des Namespace, in dem ein angegebenes Schema vorhanden ist.
privilegege_type	text	Der Typ der Berechtigung. Mögliche Werte sind USAGE und CREATE.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind die Benutzer-ID, Rollen-ID und Gruppen-ID.
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind der Benutzer, die Rolle, die Gruppe und „Öffentlich“.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Berechtigung erteilen kann. Für den Rollen- und Gruppenidentitätstyp ist dies immer „false“.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis von SVV_SCHEMA_PRIVILEGES angezeigt.

```
SELECT namespace_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_schema_privileges
WHERE namespace_name = 'test_schema1';

 namespace_name | privilege_type |  identity_name | identity_type | admin_option
```

```
---------------+----------------+----------------+---------------+-------------
 test_schema1   |    USAGE       |     reguser    |     user      |   False
 test_schema1   |    USAGE       |     role1      |     role      |   False
```

SVV_SCHEMA_QUOTA_STATE

Zeigt das Kontingent und die aktuelle Datenträgerverwendung für jedes Schema an.

Reguläre Benutzer können Informationen zu Schemata anzeigen, für die sie über die USAGE-Berechtigung verfügen. Superuser können Informationen zu allen Schemata in der aktuellen Datenbank anzeigen.

SVV_SCHEMA_QUOTA_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> Diese Ansicht ist nur verfügbar, wenn bereitgestellte Cluster abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
schema_id	integer	Die Namespace- oder Schema-ID.
schema_name	Zeichen (128)	Der Namespace- oder Schemaname.
schema_owner	integer	Die interne Benutzer-ID des Schemabesitzers.
quota	integer	Die Menge an Speicherplatz (in MB), die das Schema verwenden kann.
disk_usage	integer	Der Speicherplatz (in MB), der zurzeit vom Schema verwendet wird.

Spaltenname	Datentyp	Beschreibung
disk_usage_pct	double precision	Der Prozentsatz des Speicherp latzes, der zurzeit vom Schema aus dem konfigurierten Kontingent verwendet wird.

Beispielabfrage

Im folgenden Beispiel werden das Kontingent und die aktuelle Speicherverwendung für das Schema angezeigt.

```
SELECT TRIM(SCHEMA_NAME) "schema_name", QUOTA, disk_usage, disk_usage_pct FROM
 svv_schema_quota_state
WHERE SCHEMA_NAME = 'sales_schema';
schema_name   | quota | disk_usage | disk_usage_pct
--------------+-------+------------+----------------
sales_schema  | 2048  | 30         | 1.46
(1 row)
```

SVV_SYSTEM_PRIVILEGES

SVV_SYSTEM_PRIVILEGES ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Identitäten sehen, auf die sie Zugriff haben oder die sie besitzen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
system_privilege	text	Der Name der Systemberechtigung.
identity_id	Ganzzahl	Die ID der Identität. Mögliche Werte sind Benutzer-ID und Rollen-ID.

Spaltenname	Datentyp	Beschreibung
identity_name	text	Der Name der Identität.
identity_type	text	Die Art der Identität. Mögliche Werte sind Benutzer und Rolle.

Beispielabfrage

Im folgenden Beispiel wird das Ergebnis für die angegebenen Parameter angezeigt.

```
SELECT system_privilege,identity_name,identity_type FROM svv_system_privileges
WHERE system_privilege = 'ALTER TABLE' AND identity_name = 'sys:superuser';

 system_privilege | identity_name | identity_type
------------------+---------------+---------------
   ALTER TABLE    | sys:superuser |     role
```

SVV_TABLE_INFO

Zeigt zusammenfassende Informationen zu Tabellen in der Datenbank. Die Ansicht filtert Systemtabellen aus und zeigt nur benutzerdefinierte Tabellen an.

Sie können die Ansicht SVV_TABLE_INFO verwenden, um Probleme beim Tabellendesign zu diagnostizieren und zu beheben, die sich auf die Abfrageleistung auswirken können. Dies umfasst Probleme mit der Kompressionskodierung, den Verteilungsschlüsseln, dem Sortierstil, der Verzerrung der Datenverteilung, der Tabellengröße und den Statistiken. Die Ansicht „SVV_TABLE_INFO“ gibt keine Informationen zu leeren Tabellen zurück.

Die Ansicht SVV_TABLE_INFO fasst Informationen aus STV_BLOCKLIST, STV_NODE_STORAGE_CAPACITY, STV_TBL_PERM, den Systemtabellen STV_SLICES und den Katalogtabellen PG_DATABASE, PG_ATTRIBUTE, PG_CLASS,PG_NAMESPACE und PG_TYPE zusammen.

SVV_TABLE_INFO ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten. Damit ein Benutzer die Ansicht abfragen kann, gewähren Sie ihm die SELECT-Berechtigung für SVV_TABLE_INFO.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
database	Text	Database name (Datenban kname).
schema	Text	Schemaname.
table_id	OID	Tabellen-ID.
table	Text	Tabellenname.
encoded	Text	Wert, der angibt, ob für eine der Spalten eine Kompressi onskodierung definiert ist.
diststyle	Text	Verteilungsstil- oder Verteilun gsschlüsselspalte, falls die Schlüsselverteilung definiert ist. Mögliche Werte sind EVEN, KEY(*column*), ALL, AUTO(ALL) , AUTO(EVEN) und AUTO(KEY(*column*)).
sortkey1	Text	Erste Spalte in dem Sortiersc hlüssel, falls ein Sortiersc hlüssel definiert ist. Zu den möglichen Werten gehören *column*, AUTO(SORT KEY) und AUTO(SORT KEY(*column*)).
max_varchar	integer	Größe der größten Spalte, die den Datentyp VARCHAR verwendet.
sortkey1_enc	character(32)	Kompressionskodierung der ersten Spalte in dem Sortiersc

Spaltenname	Datentyp	Beschreibung
		hlüssel, falls ein Sortiersc hlüssel definiert ist.
sortkey_num	integer	Anzahl der als Sortierschlüssel definierten Spalten.
size	bigint	Größe der Tabelle, in 1-MB-Datenblöcken.
pct_used	numeric(10,4)	Prozentsatz des verfügbar en Platzes, den die Tabelle verwendet.
empty	bigint	Zur internen Verwendung. Diese Spalte wird nicht mehr verwendet und in künftigen Versionen entfernt.
unsorted	numeric(5,2)	Prozentsatz der nicht sortierte n Zeilen in der Tabelle.
stats_off	numeric(5,2)	Zahl, die angibt, wie aktuell die Tabellenstatistik ist; 0 bedeutet aktuell, 100 veraltet.
tbl_rows	numeric(38,0)	Gesamtzahl der Zeilen in der Tabelle. Dieser Wert beinhalte t Zeilen, die zur Löschung markiert, jedoch noch nicht bereinigt wurden.

Spaltenname	Datentyp	Beschreibung
skew_sortkey1	numeric(19,2)	Verhältnis der Größe der größten Nicht-Sortierschlüssel-Spalte zur Größe der ersten Spalte des Sortierschlüssels, falls ein Sortierschlüssel definiert ist. Verwenden Sie diesen Wert für die Evaluierung der Effektivität des Sortierschlüssels.
skew_rows	numeric(19,2)	Verhältnis der Anzahl der Zeilen in dem Slice mit den meisten Zeilen zur Anzahl der Zeilen in dem Slice mit den wenigsten Zeilen.
estimated_visible_rows	numeric(38,0)	Die geschätzten Zeilen der Tabelle. Dieser Wert enthält keine Zeilen, die zum Löschen markiert sind.

Spaltenname	Datentyp	Beschreibung

risk_event	Text	Risikoinformationen zu einer Tabelle. Das Feld ist in zwei Teile unterteilt:
		
		`risk_type |xid|timestamp`
		
		• Der `risk_type` , wobei 1 angibt an, dass ein `COPY command with the EXPLICIT_IDS option` ausgeführt wurde. Amazon Redshift überprüft nicht länger, ob IDENTITY-Spalten in der Tabelle eindeutig sind. Weitere Informationen finden Sie unter EXPLICIT_IDS.
		• Die Transaktions-ID, `xid`, die das Risiko eingeführt hat.
		• Der `timestamp` , an dem der COPY-Befehl ausgeführt wurde.
		
		Im folgenden Beispiel finden Sie die Werte in dem Feld.
		
		`1|1107|2019-06-22 07:16:11.292952`

Spaltenname	Datentyp	Beschreibung
vacuum_sort_benefit	numeric(12,2)	Die geschätzte maximale prozentuale Verbesserung der Scan-Abfrageleistung bei der Ausführung der Vacuum-Sortierung.
create_time	Timestamp ohne Zeitzone	Der Zeitstempel der Erstellung der Tabelle.

Beispielabfragen

Das folgende Beispiel zeigt Kodierung, Verteilungsstil, Sortierung und Datenverzerrung für alle benutzerdefinierten Tabellen in der Datenbank. Hierbei muss „table“ in doppelte Anführungszeichen eingeschlossen werden, da es sich um ein reserviertes Wort handelt.

```
select "table", encoded, diststyle, sortkey1, skew_sortkey1, skew_rows
from svv_table_info
order by 1;

table          | encoded | diststyle       | sortkey1     | skew_sortkey1 | skew_rows
---------------+---------+-----------------+--------------+---------------+----------
category       | N       | EVEN            |              |               |
date           | N       | ALL             | dateid       |          1.00 |
event          | Y       | KEY(eventid)    | dateid       |          1.00 |      1.02
listing        | Y       | KEY(listid)     | dateid       |          1.00 |      1.01
sales          | Y       | KEY(listid)     | dateid       |          1.00 |      1.02
users          | Y       | KEY(userid)     | userid       |          1.00 |      1.01
venue          | N       | ALL             | venueid      |          1.00 |
(7 rows)
```

SVV_TABLES

Verwenden Sie SVV_TABLES zur Anzeige von Tabellen in lokalen und externen Katalogen.

SVV_TABLES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
table_catalog	Text	Der Name des Katalogs, in dem die Tabelle besteht.
table_schema	Text	Der Name des Schemas für die Tabelle.
table_name	Text	Der Name der Tabelle.
table_type	Text	Der Typ der Tabelle. Mögliche Werte sind Ansichten, externe Tabellen und Basistabellen.
remarks	Text	Anmerkungen.

SVV_TRANSACTIONS

Zeichnet Informationen zu Transaktionen auf, die derzeit Tabellen in der Datenbank sperren. Verwenden Sie die Ansicht SVV_TRANSACTIONS, um offene Transaktionen und Probleme mit Sperrkonflikten anzuzeigen. Weitere Informationen zu Sperren finden Sie unter Verwalten gleichzeitiger Schreiboperationen und LOCK.

SVV_TRANSACTIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
txn_owner	Text	Name des Eigentümers der Transaktion.
txn_db	Text	Name der mit der Transaktion verbundenen Datenbank.

Spaltenname	Datentyp	Beschreibung
xid	bigint	Transaktions-ID.
pid	integer	ID des Prozesses, der mit der Sperre verbunden ist.
txn_start	timestamp	Anfangszeit der Transaktion.
lock_mode	Text	Name des Sperrmodus, der von diesem Prozess ausgeführt oder angefragt wird. Wenn `lock_mode` `ExclusiveLock` ist und `granted` „true“ (`t`) handelt es sich bei dieser Transaktions-ID um eine offene Transaktion.
lockable_object_type	Text	Typ des Objekts, das die Sperre anfragt oder ausführt, `relation` bei einer Tabelle oder `transactionid` bei einer Transaktion.
relation	integer	Tabellen-ID für die Tabelle (Beziehung), die die Sperre abruft. Dieser Wert ist NULL, wenn `lockable_object_type` `transactionid` ist.
granted	Boolean	Wert, der angibt, ob die Sperre gewährt wurde (`t`) oder aussteht (`f`) .

Beispielabfragen

Der folgende Befehl zeigt alle aktiven Transaktionen und die von jeder Transaktion angefragten Sperren.

```
select * from svv_transactions;

 txn_
 lockable_
 owner | txn_db |  xid   |  pid  |         txn_start          |      lock_mode      |
 object_type    | relation | granted
-------+--------+--------+-------+----------------------------+---------------------
+----------------+----------+---------
 root  | dev    | 438484 | 22223 | 2016-03-02 18:42:18.862254 | AccessShareLock     |
 relation       |   100068 | t
 root  | dev    | 438484 | 22223 | 2016-03-02 18:42:18.862254 | ExclusiveLock       |
 transactionid  |          | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | AccessShareLock     |
 relation       |    50860 | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | AccessShareLock     |
 relation       |    52310 | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | ExclusiveLock       |
 transactionid  |          | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100068 | f
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | RowExclusiveLock    |
 relation       |    16688 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessShareLock     |
 relation       |   100064 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100166 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100171 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100190 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | ExclusiveLock       |
 transactionid  |          | t
(12 rows)

(12 rows)
```

SVV_USER_GRANTS

Verwenden Sie SVV_USER_GRANTS, um die Liste der Benutzer anzuzeigen, denen explizit Rollen im Cluster gewährt werden.

SVV_USER_GRANTS ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Andere Benutzer können nur Rollen sehen, die ihnen explizit erteilt wurden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die Benutzer-ID für den Benutzer.
user_name	text	Der Name des Benutzers.
role_id	Ganzzahl	Die Rollen-ID der gewährten Rolle.
role_name	text	Der Rollenname der gewährten Rolle.
admin_option	boolesch	Ein Wert, der angibt, ob der Benutzer anderen Benutzern und Rollen die Rolle gewähren kann.

Beispielabfragen

Die folgenden Abfragen gewähren Benutzern Rollen und zeigen die Liste der Benutzer an, denen Rollen explizit gewährt werden.

```
GRANT ROLE role1 TO reguser;
GRANT ROLE role2 TO reguser;
GRANT ROLE role1 TO superuser;
GRANT ROLE role2 TO superuser;

SELECT user_name,role_name,admin_option FROM svv_user_grants;
```

```
 user_name | role_name | admin_option
-----------+-----------+--------------
 superuser |  role1    | False
 reguser   |  role1    | False
 superuser |  role2    | False
  reguser  |  role2    | False
```

SVV_USER_INFO

In der Ansicht SVV_USER_INFO können Sie Daten zu Benutzern der Amazon-Redshift-Datenbank abrufen.

SVV_USER_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_name	text	Der Benutzernamen für die Rolle.
user_id	Ganzzahl	Die Benutzer-ID für den Benutzer.
createdb	boolesch	Ein Wert, der angibt, ob der Benutzer zum Erstellen von Datenbanken berechtigt ist.
superuser	boolesch	Ein Wert, der angibt, ob der Benutzer ein Superuser ist.
catalog_update	boolesch	Ein Wert, der angibt, ob der Benutzer Systemkataloge aktualisieren kann.
connection_limit	text	Die Anzahl der Verbindungen, die der Benutzer öffnen kann.
syslog_access	text	Ein Wert, der angibt, ob der Benutzer Zugriff auf die Systemprotokolle hat. Die beiden möglichen Werte sind `RESTRICTED` und `UNRESTRICTED` . `RESTRICTED` bedeutet, dass Benutzer, die keine Superuser sind, ihre eigenen Datensätze anzeigen können. `UNRESTRICTED`

Spaltenname	Datentyp	Beschreibung
		bedeutet, dass Benutzer, die keine Superuser sind, alle Datensätze in den Systemansichten und -tabellen anzeigen können, für die sie SELECT-Berechtigungen besitzen.
last_ddl_timestamp	Zeitstempel	Der Zeitstempel für die letzte Data Definition Language (DDL)-Ers tellungsanweisung, die vom Benutzer ausgeführt wurde.
session_timeout	Ganzzahl	Die maximale Zeit in Sekunden, die eine Sitzung ohne Timeout inaktiv oder untätig bleibt. 0 bedeutet, dass kein Timeout eingestellt ist. Informati onen zur Einstellung für den Leerlauf oder den Timeout bei Inaktivität des Clusters finden Sie unter Kontingente und Limits in Amazon Redshift im Amazon-Redshift-Verwaltungshandbuch.
external_user_id	text	Die eindeutige Kennung des Benutzers beim externen Identitätsanbieter.

Beispielabfragen

Mit dem folgenden Befehl werden Benutzerinformationen aus SVV_USER_INFO abgerufen.

```
SELECT * FROM SVV_USER_INFO;
```

SVV_VACUUM_PROGRESS

Diese Ansicht gibt eine Schätzung der Zeit aus, die der Abschluss einer Bereinigungsoperation, die derzeit läuft, in Anspruch nehmen wird.

SVV_VACUUM_PROGRESS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_VACUUM_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Informationen zu SVV_VACUUM_SUMMARY finden Sie unter SVV_VACUUM_SUMMARY.

Informationen zu SVL_VACUUM_PERCENTAGE finden Sie unter SVL_VACUUM_PERCENTAGE.

> **Note**
>
> Diese Ansicht ist nur verfügbar, wenn bereitgestellte Cluster abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
table_name	Text	Name der derzeit bereinigten Tabelle bzw. der Tabelle, die zuletzt bereinigt wurde, wenn derzeit kein Bereinigungsvorgang stattfindet.
status	Text	Beschreibung der zurzeit ausgeführten Aktivität im Rahmen der Bereinigungsoperation: • Initialisieren • Sortierung • Merge • Löschen • Select • Fehlgeschlagen • Complete • Übersprungen • Aufbau der INTERLEAVED SORTKEY-Reihenfolge
time_remaining_estimate	Text	Geschätzte verbleibende Zeit für den Abschluss der aktuellen Bereinigungsoperation in Minuten und Sekunden, zum Beispiel: **`5m 10s`**. Es wird keine Zeitschätzung ausgegeben, bevor der erste Sortiervorgang der Bereinigung abgeschlossen ist. Wenn derzeit keine Bereinigung durchgeführt wird, wird der zuletzt ausgeführte Bereinigungsvorgang mit **`Completed`** in der Spalte STATUS und einer leeren Spalte TIME_REMAINING_ESTIMATE angezeigt. Die Schätzung

Spaltenname	Datentyp	Beschreibung
		wird im Laufe des Bereinigungsvorgangs typischerweise immer zuverlässiger.

Beispielabfragen

Die folgenden Abfragen, im Abstand einiger Minuten ausgeführt, zeigen, dass eine große Tabelle mit der Bezeichnung SALESNEW bereinigt wird.

```
select * from svv_vacuum_progress;

table_name    |            status             | time_remaining_estimate
--------------+-------------------------------+-------------------------
salesnew      |  Vacuum: initialize salesnew  |
(1 row)
...
select * from svv_vacuum_progress;

table_name   |         status         | time_remaining_estimate
-------------+------------------------+-------------------------
salesnew     |  Vacuum salesnew sort  | 33m 21s
(1 row)
```

Die folgende Abfrage zeigt, dass derzeit keine Bereinigungsoperation durchgeführt wird. Die letzte bereinigte Tabelle war die Tabelle SALES.

```
select * from svv_vacuum_progress;

table_name   |  status  | time_remaining_estimate
-------------+----------+-------------------------
  sales      | Complete |
(1 row)
```

SVV_VACUUM_SUMMARY

Die Ansicht SVV_VACUUM_SUMMARY verbindet die Tabellen STL_VACUUM, STL_QUERY und STV_TBL_PERM zur Zusammenfassung von Informationen zu den vom System protokollierten Bereinigungsoperationen. Die Ansicht gibt eine Zeile pro Tabelle und Bereinigungsoperation aus. Die Ansicht zeichnet die verstrichene Zeit der Operation, die Anzahl der erstellten Sortierpartitionen, die

Anzahl der erforderlichen Zusammenführungsinkremente sowie Differenzen (Deltas) der Zeilen- und Blockzahlen vor und nach der Durchführung der Operation auf.

SVV_VACUUM_SUMMARY ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_VACUUM_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Informationen zu SVV_VACUUM_PROGRESS finden Sie unter SVV_VACUUM_PROGRESS.

Informationen zu SVL_VACUUM_PERCENTAGE finden Sie unter SVL_VACUUM_PERCENTAGE.

> **Note**
>
> Diese Ansicht ist nur verfügbar, wenn bereitgestellte Cluster abgefragt werden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
table_name	Text	Der Name der bereinigten Tabelle.
xid	bigint	Transaktions-ID der VACUUM-Operation.
sort_partitions	bigint	Anzahl der während der Sortierphase der Bereinigungsoperation sortierten Partitionen.
merge_increments	bigint	Anzahl der für den Abschluss der Zusammenführungsphase der Bereinigungsoperation erforderlichen Zusammenführungsinkremente.
elapsed_time	bigint	Verstrichene Laufzeit der Bereinigungsoperation (in Mikrosekunden).
row_delta	bigint	Differenz der Gesamtzahl der Tabellenzeilen vor und nach der Bereinigung.

Spaltenname	Datentyp	Beschreibung
sortedrow_delta	bigint	Differenz der Gesamtzahl der sortierten Tabellenzeilen vor und nach der Bereinigung.
block_delta	integer	Differenz der Anzahl der Blöcke für die Tabelle vor und nach der Bereinigung.
max_merge_partitions	integer	Diese Spalte wird für die Leistungsanalyse verwendet und enthält die maximale Anzahl der Partitionen, die die Bereinigung für die Tabelle pro Iteration der Zusammenführungsphase verarbeiten kann. (Die Bereinigung sortiert die nicht sortierte Region in eine oder mehrere sortierte Partitionen. Je nach der Anzahl der Spalten in der Tabelle und der aktuellen Amazon-Redshift-Konfiguration kann die Zusammenführungsphase eine maximale Anzahl von Partitionen in einer einzelnen Zusammenführungsiteration verarbeiten. Die Zusammenführungsphase funktioniert auch, wenn die Anzahl der sortierten Partitionen die maximale Anzahl der Zusammenführungspartitionen überschreitet, es werden jedoch weitere Zusammenführungsiterationen benötigt.)

Beispielabfrage

Die folgende Abfrage gibt statistische Daten für Bereinigungsoperationen auf drei verschiedenen Tabellen aus. Die Tabelle SALES wurde zweimal bereinigt.

```
select table_name, xid, sort_partitions as parts, merge_increments as merges,
elapsed_time, row_delta, sortedrow_delta as sorted_delta, block_delta
from svv_vacuum_summary
order by xid;

table_  | xid  |parts|merges| elapsed_ | row_    | sorted_ | block_
name    |      |     |      | time     | delta   | delta   | delta
--------+------+-----+------+----------+---------+---------+--------
users   | 2985 |   1 |    1 | 61919653 |       0 |   49990 |      20
category| 3982 |   1 |    1 | 24136484 |       0 |      11 |       0
sales   | 3992 |   2 |    1 | 71736163 |       0 | 1207192 |      32
```

```
sales   | 4000 |   1 |    1 | 15363010 | -851648 | -851648 |    -140
(4 rows)
```

SYS-Überwachungsansichten

Überwachungsansichten sind Systemansichten in Amazon Redshift, die zur Überwachung der Abfrage- und Workload-Ressourcennutzung von bereitgestellten Clustern und Serverless-Arbeitsgruppen verwendet werden. Diese Ansichten befinden sich im `pg_catalog`-Schema Um die von diesen Ansichten bereitgestellten Informationen anzuzeigen, führen Sie SQL SELECT-Anweisungen aus.

Sofern nicht anders angegeben, sind diese Ansichten für Amazon-Redshift-Cluster und Arbeitsgruppen von Amazon Redshift Serverless verfügbar.

SYS_SERVERLESS_USAGE sammelt nur Nutzungsdaten für Amazon Redshift Serverless.

Themen

- SYS_LOAD_ERROR_DETAIL
- SYS_LOAD_HISTORY
- SYS_MV_REFRESH_HISTORY
- SYS_MV_STATE
- SYS_PROCEDURE_CALL
- SYS_PROCEDURE_MESSAGES
- SYS_QUERY_DETAIL
- SYS_QUERY_HISTORY
- SYS_QUERY_TEXT
- SYS_RESTORE_LOG
- SYS_RESTORE_STATE
- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SERVERLESS_USAGE
- SYS_SESSION_HISTORY
- SYS_SPATIAL_SIMPLIFY
- SYS_STREAM_SCAN_ERRORS
- SYS_STREAM_SCAN_STATES
- SYS_TRANSACTION_HISTORY
- SYS_UDF_LOG
- SYS_UNLOAD_DETAIL
- SYS_UNLOAD_HISTORY
- SYS_USERLOG
- SYS_VACUUM_HISTORY

SYS_ANALYZE_COMPRESSION_HISTORY

Zeichnet Details zu Komrimierungsanalysen während der Ausführung von COPY- oder ANALYZE COMPRESSION-Befehlen auf.

SYS_ANALYZE_COMPRESSION_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	ID des Benutzers, der den Eintrag generiert hat.
start_time	timestamp	Die Zeit, zu der die Komprimierungsanalyse gestartet wurde.
transaction_id	bigint	Die Transaktions-ID der Komprimierungsanalyse.
table_id	Ganzzahl	Die Tabellen-ID der analysierten Tabelle.
table_name	character(128)	Der Name der analysierten Tabelle.
column_position	Ganzzahl	Der Index der Spalte in der Tabelle, die analysiert wurde, um die Komprimierungscodierung zu ermitteln.
old_encoding	character(15)	Der Codierungstyp vor der Komprimierungsanalyse.
new_encoding	character(15)	Der Codierungstyp nach der Komprimierungsanalyse.
mode	character(14)	Die möglichen Werte sind: PRESET Gibt an, dass `new_encoding` vom Amazon-Redshift-Befehl COPY basierend auf dem Datentyp der Spalte bestimmt wird. Es werden keine Stichproben der Daten genommen. ON Gibt an, dass `new_encoding` vom Amazon-Redshift-Befehl COPY basierend auf der Analyse von Beispieldaten bestimmt wird.

Spaltenname	Datentyp	Beschreibung
		ANALYZE ONLY Gibt an, dass `new_encoding` vom Amazon-Redshift-Befehl ANALYZE COMPRESSION basierend auf der Analyse von Beispieldaten bestimmt wird. Der Codierungstyp der analysierten Spalte wird jedoch nicht geändert.

Beispielabfragen

Das folgende Beispiel inspiziert die Details der Komprimierungsanalyse der `lineitem`-Tabelle durch die letzte Ausführung des Befehls COPY in dieser Sitzung.

```
select transaction_id, table_id, btrim(table_name) as table_name, column_position,
 old_encoding, new_encoding, mode
from sys_analyze_compression_history
where transaction_id = (select transaction_id from sys_query_history where query_id =
 pg_last_copy_id()) order by column_position;

 transaction_id  |  table_id   | table_name | column_position |  old_encoding   |
 new_encoding   |      mode
-----------------+-------------+------------+-----------------+-----------------
+-----------------+-------------
      8196       |   248126    | lineitem   |        0        | mostly32        |
 mostly32        | ON
      8196       |   248126    | lineitem   |        1        | mostly32        | lzo
           | ON
      8196       |   248126    | lineitem   |        2        | lzo             |
 delta32k        | ON
      8196       |   248126    | lineitem   |        3        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        4        | bytedict        |
 bytedict        | ON
      8196       |   248126    | lineitem   |        5        | mostly32        |
 mostly32        | ON
      8196       |   248126    | lineitem   |        6        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        7        | delta           | delta
           | ON
```

```
      8196      |   248126    | lineitem   |        8        | lzo             | zstd
           | ON
      8196      |   248126    | lineitem   |        9        | runlength       | zstd
           | ON
      8196      |   248126    | lineitem   |       10        | delta           | lzo
           | ON
      8196      |   248126    | lineitem   |       11        | delta           | delta
           | ON
      8196      |   248126    | lineitem   |       12        | delta           | delta
           | ON
      8196      |   248126    | lineitem   |       13        | bytedict        | zstd
           | ON
      8196      |   248126    | lineitem   |       14        | bytedict        | zstd
           | ON
      8196      |   248126    | lineitem   |       15        | text255         | zstd
           | ON
(16 rows)
```

SYS_ANALYZE_HISTORY

Protokolliert Einzelheiten zu ANALYZE-Operationen.

SYS_ANALYZE_HISTORY ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	ID des Benutzers, der den Eintrag generiert hat.
transaction_id	long	Die Transaktions-ID.
query_id	long	Die Abfrage-ID in SYS_QUERY_HISTORY.
database_name	char(30)	Name der Datenbank.
table_name	char(30)	Der Name der Tabelle.

Spaltenname	Datentyp	Beschreibung
table_id	Ganzzahl	Die ID der Tabelle.
is_automatic	char(1)	Der Wert ist „true“ (t), wenn die Operation standardmäßig eine Amazon-Redshift-Analyseoperation beinhaltete. Der Wert ist „false“ (f), wenn der ANALYZE-Befehl explizit ausgeführt wurde.
Status	char(15)	Das Ergebnis des ANALYZE-Befehls. Mögliche Werte sind Vollständig, Übersprungen und PredicateColumn.
start_time	Zeitstempel	Zeitpunkt nach UTC, an dem die Ausführung der ANALYZE-Operation gestartet wurde.
end_time	Zeitstempel	Zeitpunkt nach UTC, an dem die Ausführung der ANALYZE-Operation beendet wurde.
rows	double	Die Gesamtzahl der Zeilen in der Tabelle.
modified_rows	double	Die Gesamtzahl der Zeilen, die seit der letzten ANALYZE-Operation modifiziert wurden.
analyze_threshold_percent	Ganzzahl	Der Wert des Parameters analyze_threshold_percent.
last_analyze_time	Zeitstempel	Zeitpunkt nach UTC, an dem die Tabelle zuletzt analysiert wurde.

Beispielabfragen

```
 user_id | transaction_id | database_name | schema_name |      table_name     |
 table_id | is_automatic | Status |         start_time         |          end_time
     | rows | modified_rows | analyze_threshold_percent |  last_analyze_time
---------+----------------+---------------+------------+---------------------
+----------+--------------+--------+----------------------------+----------
```

```
+-----------------+------+--------------+--------------------------
+---------------------
     101 |           8006 |           dev |      public | test_table_562bf8dc
 |   110427 |            f |   Full | 2023-09-21 18:33:08.504646 | 2023-09-21
 18:33:24.296498 |    5 |             5 |                          0 | 2000-01-01
 00:00:00
```

SYS_APPLIED_MASKING_POLICY_LOG

Verwenden Sie SYS_APPLIED_MASKING_POLICY_LOG, um die Anwendung dynamischer Datenmaskierungsrichtlinien auf Abfragen nachzuverfolgen, die auf DDM-geschützte Beziehungen verweisen.

SYS_APPLIED_MASKING_POLICY_LOG ist für die folgenden Benutzer sichtbar:

- Superuser
- Benutzer mit der Rolle `sys:operator`
- Benutzer mit der Berechtigung ACCESS SYSTEM TABLE

Regulären Benutzern werden 0 Zeilen angezeigt.

Beachten Sie, dass SYS_APPLIED_MASKING_POLICY_LOG für Benutzer mit dieser Rolle nicht sichtbar ist. `sys:secadmin`

Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter. Dynamische Datenmaskierung

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
policy_name	text	Der Name der Maskierungsrichtlinie.
user_id	text	Die ID des Benutzers, der die Abfrage ausgeführt hat.

Spaltenname	Datentyp	Beschreibung
record_time	Zeitstempel	Die Uhrzeit, zu der der Eintrag in der Systemansicht aufgezeichnet wurde.
session_id	int	Die Prozess-ID.
transaction_id	long	Die Transaktions-ID.
query_id	int	Die Abfrage-ID.
database_name	text	Der Name der Datenbank, in der die Abfrage ausgeführt wurde.
relation_name	text	Der Name der Tabelle, auf die die Maskierungsrichtlinie angewendet wird.
schema_name	text	Der Name des Schemas, in dem sich die Tabelle befindet.
attachment_id	long	Die ID der angehängten Maskierungsrichtlinie.
relation_kind	text	Der Typ der Beziehung, auf die die Maskierungsrichtlinie angewendet wird. Mögliche Werte sind `TABLE`, `VIEW`, `LATE BINDING VIEW` und `MATERIALIZED VIEW` .

Beispielabfragen

Das folgende Beispiel zeigt, dass die `mask_credit_card_full` Maskierungsrichtlinie an die Tabelle angehängt ist. `credit_db.public.credit_cards`

```
select policy_name, database_name, relation_name, schema_name, relation_kind
from sys_applied_masking_policy_log;

policy_name           | database_name | relation_name | schema_name | relation_kind
----------------------+---------------+---------------+-------------+---------------
mask_credit_card_full | credit_db     | credit_cards  | public      | table
```

```
(1 row)
```

SYS_AUTO_TABLE_OPTIMIZATION

Zeichnet automatisierte Aktionen von Amazon Redshift in Tabellen auf, die für die automatische Optimierung definiert sind.

SYS_AUTO_TABLE_OPTIMIZATION ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
transaction_id	long	Die Transaktions-ID.
session_id	int	Die Sitzungs-ID des Prozesses, der den Alter-Befehl ausgeführt hat.
table_id	int	Die Tabellenkennung.
alter_table_type	character (32)	Die Art der Empfehlung. Mögliche Werte sind `distkey`, `sortkey` und `encode`.
Status	character (128)	Der Abschlussstatus der Empfehlung. Mögliche Werte sind `Start`, `Complete`, `Skipped`, `Abort`, `Checkpoint` und `Failed`.
event_time	Zeitstempel	Der Zeitstempel der Statusspalte.
alter_from	character (200)	Der vorherige Verteilungsstil und die Sortierschlüssel der Tabelle, bevor die Empfehlung angewendet wird. Der Wert wird in Schritten von je 200 Zeichen abgeschnitten.
alter_to	character (200)	Der aktuelle Verteilungsstil und die Sortierschlüssel der Tabelle nach Anwendung der Empfehlung. Der Wert wird in Schritten von je 200 Zeichen abgeschnitten.

Beispielabfragen

Im folgenden Beispiel zeigen die Zeilen im Ergebnis Aktionen an, die von Amazon Redshift ausgeführt wurden.

```
SELECT table_id, alter_table_type, status, event_time, alter_from
FROM SYS_AUTO_TABLE_OPTIMIZATION;

 table_id |  alter_table_type   |                         status
 |         event_time          |  alter_from
----------+---------------------+-----------------------------------------------------
+-----------------------------+-----------------
   118082 | sortkey             | Start
 | 2020-08-22 19:42:20.727049  |
   118078 | sortkey             | Start
 | 2020-08-22 19:43:54.728819  |
   118082 | sortkey             | Start
 | 2020-08-22 19:42:52.690264  |
   118072 | sortkey             | Start
 | 2020-08-22 19:44:14.793572  |
   118082 | sortkey             | Failed
 | 2020-08-22 19:42:20.728917  |
   118078 | sortkey             | Complete
 | 2020-08-22 19:43:54.792705  |  SORTKEY: None;
   118086 | sortkey             | Complete
 | 2020-08-22 19:42:00.72635   |  SORTKEY: None;
   118082 | sortkey             | Complete
 | 2020-08-22 19:43:34.728144  |  SORTKEY: None;
   118072 | sortkey             | Skipped:Retry exceeds the maximum limit for a table.
 | 2020-08-22 19:44:46.706155  |
   118086 | sortkey             | Start
 | 2020-08-22 19:42:00.685255  |
   118082 | sortkey             | Start
 | 2020-08-22 19:43:34.69531   |
   118072 | sortkey             | Start
 | 2020-08-22 19:44:46.703331  |
   118082 | sortkey             | Checkpoint: progress 14.755079%
 | 2020-08-22 19:42:52.692828  |
   118072 | sortkey             | Failed
 | 2020-08-22 19:44:14.796071  |
   116723 | sortkey             | Abort:This table is not AUTO.
 | 2020-10-28 05:12:58.479233  |
```

```
  110203 | distkey             | Abort:This table is not AUTO.
| 2020-10-28 05:45:54.67259   |
```

SYS_CONNECTION_LOG

Protokolliert Authentifizierungsversuche sowie Verbindungen und Verbindungstrennungen.

SYS_CONNECTION_LOG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
event	character(50)	Verbindungs- oder Authentifizierungsereignis.
record_time	Zeitstempel	Uhrzeit, zu der das Ereignis aufgetreten ist.
remote_host	character(45)	Name oder IP-Adresse des Remote-Hosts.
remote_port	character(32)	Portnummer für den Remote-Host.
session_id	Ganzzahl	Die mit der Anweisung verbundene Prozess-ID.
database_name	character(50)	Datenbankname.
user_name	character(50)	Benutzername
auth_method	character(32)	Authentifizierungsmethode.
duration	integer	Dauer der Verbindung in Mikrosekunden.
ssl_version	character(50)	Secure Sockets Layer (SSL)-Version.
ssl_cipher	character(128)	SSL-Verschlüsselungsverfahren.
mtu	integer	Maximum Transmission Unit (MTU).

Spaltenname	Datentyp	Beschreibung
ssl_compression	character(64)	SSL-Kompressionstyp.
ssl_expansion	character(64)	SSL-Expansionstyp.
iam_auth_guid	character(36)	Die IAM-Authentifizierungs-ID für die CloudTrail Anfrage.
application_name	character(250)	Der ursprüngliche oder aktualisierte Name der Anwendung für eine Sitzung.
driver_version	character(64)	Die Version des ODBC- oder JDBC-Treibers, die von Ihren SQL-Client-Tools von Drittanbietern eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellt.
os_version	character(64)	Die Version des Betriebssystems, das sich auf dem Clientcomputer befindet, der eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellt.
plugin_name	character(32)	Der Name des Plug-Ins, mit dem Sie eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellen.

Spaltenname	Datentyp	Beschreibung
protocol_version	integer	Die interne Protokollversion, die der Amazon-Redshift-Treiber beim Herstellen der Verbindung mit dem Server verwendet . Die Protokollversionen werden zwischen Treiber und Server ausgehandelt. Die Version beschreibt die verfügbaren Funktionen. Gültige Werte sind: • 0 (BASE_SERVER_PROTOCOL_VERSION) • 1 (EXTENDED_RESULT_METADATA_SERVER_PROTOCOL_VERSION) – Um einen Umlauf pro Abfrage einzusparen, sendet der Server zusätzliche Metadaten informationen zu Ergebnismengen. • 2 (BINARY_PROTOCOL_VERSION) – Abhängig vom Datentyp der Ergebnismenge sendet der Server Daten im Binärformat. • 3 (EXTENDED2_RESULT_METADATA_SERVER_PROTOCOL_VERSION) – Der Server sendet Informationen zur Groß- und Kleinschreibung (Sortierung) einer Spalte.
global_session_id	character(36)	Der global eindeutige Bezeichner für die aktuelle Sitzung. Die Sitzungs-ID bleibt auch nach Neustarts bei Knotenfehlern bestehen.

Beispielabfragen

Führen Sie folgende Abfrage aus, um die Details zu offenen Verbindungen anzuzeigen.

```
select record_time, user_name, database_name, remote_host, remote_port
from sys_connection_log
where event = 'initiating session'
and session_id not in
(select session_id from sys_connection_log
where event = 'disconnecting session')
order by 1 desc;
```

```
record_time         | user_name   | database_name   | remote_host   | remote_port

--------------------+-------------+-----------------+---------------
+--------------------------------
2014-11-06 20:30:06 | rdsdb       | dev             | [local]       |

2014-11-06 20:29:37 | test001     | test            | 10.49.42.138  | 11111

2014-11-05 20:30:29 | rdsdb       | dev             | 10.49.42.138  | 33333

2014-11-05 20:28:35 | rdsdb       | dev             | [local]       |
(4 rows)
```

Das folgende Beispiel illustriert einen fehlgeschlagenen Authentifizierungsversuch sowie eine erfolgreiche Verbindung und Verbindungstrennung.

```
select event, record_time, remote_host, user_name
from sys_connection_log order by record_time;

            event      |         record_time        |  remote_host  | user_name

-----------------------+----------------------------+---------------+---------
authentication failure | 2012-10-25 14:41:56.96391  | 10.49.42.138  | john

authenticated          | 2012-10-25 14:42:10.87613  | 10.49.42.138  | john

initiating session     | 2012-10-25 14:42:10.87638  | 10.49.42.138  | john

disconnecting session  | 2012-10-25 14:42:19.95992  | 10.49.42.138  | john

(4 rows)
```

Das folgende Beispiel zeigt die Version des ODBC-Treibers, das Betriebssystem auf dem Clientcomputer und das Plug-In, das für die Verbindung mit dem Amazon-Redshift-Cluster verwendet wird. In diesem Beispiel dient das verwendete Plug-In zur standardmäßigen ODBC-Treiberauthentifizierung unter Verwendung eines Anmeldenamens und eines Kennworts.

```
select driver_version, os_version, plugin_name from sys_connection_log;

driver_version                          |  os_version                       |
 plugin_name
```

```
---------------------------------------+-----------------------------------
+--------------------
Amazon Redshift ODBC Driver 1.4.15.0001 | Darwin 18.7.0 x86_64              | none
Amazon Redshift ODBC Driver 1.4.15.0001 | Linux 4.15.0-101-generic x86_64   | none
```

Das folgende Beispiel zeigt die Version des Betriebssystems auf dem Clientcomputer, die Treiberversion und die Protokollversion an.

```
select os_version, driver_version, protocol_version from sys_connection_log;

os_version                     |  driver_version              | protocol_version
-------------------------------+------------------------------+--------------------
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
```

SYS_COPY_JOB (Vorschau)

Dies ist eine Vorabversion der Dokumentation für Autocopy (SQL COPY JOB), die sich in der Vorabversion befindet. Sowohl die Dokumentation als auch die Funktion können sich ändern. Wir empfehlen, diese Funktion nur in Test- und nicht in Produktionsumgebungen zu verwenden. Die öffentliche Vorversion endet am 31. Juli 2024. Vorschau-Cluster werden zwei Wochen nach dem Ende der Vorschauversion automatisch entfernt. Weitere Informationen zu den Bedingungen für Vorschauversionen finden Sie unter Betas und Vorversionen in den AWS -Servicebedingungen.

Verwenden Sie SYS_COPY_JOB, um Details zu COPY JOB-Befehlen anzuzeigen.

Diese Ansicht enthält die erstellten COPY-JOB-Befehle.

SYS_COPY_JOB ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
job_id	bigint	Die Kennung des Kopierauftrags.
job_name	character(128)	Der Name des Kopierauftrags.
iam_role	character(128)	Die in der COPY-Anweisung angegebene IAM-Rolle.
job_text	character(256)	Die Parameter der COPY-Anweisung.
is_auto	Ganzzahl	Gibt an, ob COPY JOB automatisch von Amazon Redshift ausgeführt wird. 1 bedeutet wahr, `0` bedeutet falsch.
on_error_suspend	Ganzzahl	Diese Information ist nur für die interne Verwendung gedacht.

SYS_COPY_REPLACEMENTS

Zeigt ein Protokoll an, das aufzeichnet, wann ungültige UTF-8-Zeichen vom COPY-Befehl mit der Option ACCEPTINVCHARS ersetzt wurden. Für jede der ersten 100 Zeilen auf jedem Knoten-Slice, die mindestens eine Ersetzung erforderten, wird SYS_COPY_REPLACEMENTS ein Protokolleintrag hinzugefügt.

Sie können diese Ansicht verwenden, um Informationen zu Serverless-Arbeitsgruppen und bereitgestellten Clustern einzusehen.

SYS_COPY_REPLACEMENTS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	ID des Benutzers, der die Abfrage erstellt hat.
query_id	bigint	Die Abfrage-ID. Die Spalte, die verwendet wurde, um andere Systemtabellen und Anzeigen anzufügen.
table_id	integer	Die Tabellen-ID.
file_name	character (256)	Der vollständige Pfad zur Eingabedatei für den COPY-Befehl.
column_name	character (127)	Das erste Feld, das ein ungültiges UTF-8-Zeichen enthält.
line_number	bigint	Die Zeilennummer in der Eingabedatendatei, die ein ungültige s UTF-8-Zeichen enthält. -1 gibt an, dass die Zeilennummer nicht verfügbar ist, z. B. beim Kopieren aus einer Datendatei im Spaltenformat.
raw_line	character (1024)	Die Roh-Ladedaten, die ein ungültiges UTF-8-Zeichen enthalten.

Beispielabfragen

Das folgende Beispiel gibt Ersetzungen für die letzte COPY-Operation aus.

```
select query_idp, table_id, file_name, line_number, colname
from sys_copy_replacements
where query = pg_last_copy_id();

 query_id | table_id |   file_name                        | line_number | column_name
 ---------+----------+------------------------------------+-------------+--------
    96    |    26    | s3://mybucket/allusers_pipe.txt    |         123 | city
    96    |    26    | s3://mybucket/allusers_pipe.txt    |         456 | city
    96    |    26    | s3://mybucket/allusers_pipe.txt    |         789 | city
    96    |    26    | s3://mybucket/allusers_pipe.txt    |         012 | city
    96    |    26    | s3://mybucket/allusers_pipe.txt    |         119 | city
```

...

SYS_DATESHARE_CHANGE_LOG

Zeichnet die konsolidierte Ansicht auf, um Änderungen an Datashares sowohl in Produzenten- als auch Konsumenten-Clustern nachzuverfolgen.

SYS_DATASHARE_CHANGE_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die ID des Benutzers, der die Aktion ausführt.
user_name	varchar(128)	Der Name des Benutzers, der die Aktion ausführt.
session_id	Ganzzahl	Die ID der Sitzung.
transaction_id	bigint	Die ID der Transaktion.
share_id	integer	Die ID des betroffenen Datashares.
share_name	varchar(128)	Der Name des Datashares.
source_database_id	integer	Die ID der Datenbank, zu der der Datashare gehört.
source_database_name	varchar(128)	Der Name der Datenbank, zu der der Datashare gehört.
consumer_database_id	integer	Die ID der Datenbank, die aus dem Datashare importiert wurde.

Spaltenname	Datentyp	Beschreibung
consumer_database_name	varchar(128)	Der Name der Datenbank, die aus dem Datashare importiert wurde.
arn	varchar(192)	Der ARN der Ressource, die die importierte Datenbank unterstützt.
record_time	Zeitstempel	Der Zeitstempel der Aktion.
Aktion	varchar(128)	Die Aktion, die ausgeführt wird. Mögliche Werte sind CREATE DATASHARE, DROP DATASHARE, GRANT ALTER, REVOKE ALTER, GRANT SHARE, REVOKE SHARE, ALTER ADD, ALTER REMOVE, ALTER SET, GRANT USAGE, REVOKE USAGE, CREATE DATABASE, GRANT oder REVOKE USAGE für eine gemeinsam genutzte Datenbank, DROP SHARED DATABASE, ALTER SHARED DATABASE.
Status	integer	Der Status der Aktion. Mögliche Werte sind SUCCESS und ERROR-ERROR CODE
share_object_type	varchar(64)	Der Typ des Datenbankobjekts, das dem Datashare hinzugefügt oder daraus entfernt wurde. Mögliche Werte sind Schemata, Tabellen, Spalten, Funktionen und Ansichten. Dies ist ein Feld für den Produzenten-Cluster.
share_object_id	integer	Die ID des Datenbankobjekts, das dem Datashare hinzugefügt oder daraus entfernt wurde. Dies ist ein Feld für den Produzenten-Cluster.
share_object_name	varchar(128)	Der Name des Datenbankobjekts, das dem Datashare hinzugefügt oder daraus entfernt wurde. Dies ist ein Feld für den Produzenten-Cluster.
target_user_type	varchar(16)	Der Typ der Benutzer oder Gruppen, denen eine Berechtigung erteilt wurde. Dies ist ein Feld sowohl für den Produzenten- als auch den Konsumenten-Cluster.

Spaltenname	Datentyp	Beschreibung
target_user_id	Ganzzahl	Die ID der Benutzer oder Gruppen, denen eine Berechtigung erteilt wurde. Dies ist ein Feld sowohl für den Produzenten- als auch den Konsumenten-Cluster.
target_user_name	varchar(128)	Der Name der Benutzer oder Gruppen, denen eine Berechtigung erteilt wurde. Dies ist ein Feld sowohl für den Produzenten- als auch den Konsumenten-Cluster.
consumer_account	varchar(16)	Die Konto-ID des Datenverbrauchers. Dies ist ein Feld für den Produzenten-Cluster.
consumer_namespace	varchar(64)	Der Namespace des Datenverbraucherkontos. Dies ist ein Feld für den Produzenten-Cluster.
producer_account	varchar(16)	Die Konto-ID des Produzentenkontos, zu dem der Datashare gehört. Dies ist ein Feld für den Konsumenten-Cluster.
producer_namespace	varchar(64)	Der Namespace des Produktkontos, zu dem der Datashare gehört. Dies ist ein Feld für den Konsumenten-Cluster.
attribute_name	varchar(64)	Der Name eines Attributs des Datashares oder der gemeinsamen Datenbank.
attribute_value	varchar(128)	Der Wert eines Attributs des Datashares oder der gemeinsamen Datenbank.
Nachricht	varchar(512)	Die Fehlermeldung, wenn eine Aktion fehlschlägt.

Beispielabfragen

Das folgende Beispiel zeigt eine Ansicht SYS_DATASHARE_CHANGE_LOG.

```
SELECT DISTINCT action
FROM sys_datashare_change_log
WHERE share_object_name LIKE 'tickit%';
```

```
        action
-----------------------
 "ALTER DATASHARE ADD"
```

SYS_DATASHARE_CROSS_REGION_USAGE

Verwenden Sie die Ansicht SYS_DATASHARE_CROSS_REGION_USAGE, um einen Überblick über die regionsübergreifende Datenübertragungsnutzung zu erhalten, die durch eine regionsübergreifende Datasharing-Abfrage ausgelöst wurde. SYS_DATASHARE_CROSS_REGION_USAGE aggregiert Details auf Segmentebene.

SYS_DATASHARE_CROSS_REGION_USAGE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query_id	Ganzzahl	Die ID der Abfrage. Verwenden Sie diesen Wert, um andere Systemtabellen und Ansichten zu verbinden.
child_query_sequence	Ganzzahl	Die Reihenfolge der neu geschriebenen Benutzerabfrage, beginnend mit 1.
segment_id	bigint	Die Nummer des Segments. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
start_time	variieren	Die Uhrzeit in UTC, zu der die Übertragung der Daten begann.
end_time	time	Die Uhrzeit in UTC, zu der die Übertragung der Daten endete.
transferred_data	bigint	Die Anzahl der Bytes an Daten, die von einer Produzentenregion an eine Konsumentenregion übertragen wurden.
source_region	char(25)	Die Produzentenregion, aus der die Abfrage Daten übertragen hat.

Beispielabfragen

Das folgende Beispiel zeigt eine SYS_DATASHARE_CROSS_REGION_USAGE-Ansicht.

```
SELECT query, segment, transferred_data, source_region
from sys_datashare_cross_region_usage
where query = pg_last_query_id()
order by query,segment;

  query | segment | transferred_data | source_region
--------+---------+------------------+---------------
 200048 |       2 |          4194304 |    us-west-1
 200048 |       2 |          4194304 |    us-east-2
```

SYS_DATASHARE_USAGE_CONSUMER

Zeichnet die Aktivität und Verwendung von Datashares auf. Diese Ansicht ist nur für den Konsumenten-Cluster relevant.

SYS_DATASHARE_USAGE_CONSUMER ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die ID des Benutzers, der die Anforderung ausgibt.
session_id	Ganzzahl	Die ID des Leader-Prozesses, der die Abfrage ausführt.
transaction_id	bigint	Der Kontext der aktuellen Transaktion.
request_id	varchar(50)	Die eindeutige ID des angeforderten API-Aufrufs.
request_type	varchar(25)	Der Anforderungstyp, der an das Producer-Cluster getätigt wurde.

Spaltenname	Datentyp	Beschreibung
transaction_uid	varchar(50)	Die eindeutige ID der Transaktion.
record_time	Zeitstempel	Die Zeit, zu der die Aktion aufgezeichnet wird.
status	integer	Der Status des angeforderten API-Aufrufs.
error_message	varchar(512)	Die Meldung eines Fehlers.

Beispielabfragen

Das folgende Beispiel zeigt die Ansicht SYS_DATASHARE_USAGE_CONSUMER.

```
SELECT request_type, status, trim(error) AS error
FROM sys_datashare_usage_consumer

  request_type  | status | error_message
----------------+--------+---------------
 "GET RELATION" |   0    |
```

SYS_DATASHARE_USAGE_PRODUCER

Zeichnet die Aktivität und Verwendung von Datashares auf. Diese Ansicht ist nur für den Produzenten-Cluster relevant.

SYS_DATASHARE_USAGE_PRODUCER ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
share_id	integer	Die Objekt-ID (OID) des Datenspeichers.
share_name	varchar(128)	Der Name des Datashares.
request_id	varchar(50)	Die eindeutige ID des angeforderten API-Aufrufs.
request_type	varchar(25)	Der Anforderungstyp, der an das Producer-Cluster getätigt wurde.
object_type	varchar(64)	Der Typ des Objekts, das aus dem Datashare freigegeben wird. Mögliche Werte sind Schemata, Tabellen, Spalten, Funktionen und Ansichten.
object_oid	integer	Die ID des Objekts, das aus dem Datashare freigegeben wird.
object_name	varchar(128)	Der Name des Objekts, das aus dem Datashare freigegeben wird.
consumer_account	varchar(16)	Das Konto des Konsumentenkontos, für das der Datashare freigegeben wird.
consumer_namespace	varchar(64)	Der Namespace des Konsumentenkontos, für das der Datashare freigegeben wird.
consumer_transaction_uid	varchar(50)	Die eindeutige Transaktions-ID der Anweisung im Konsumenten-Cluster.
record_time	Zeitstempel	Die Zeit, zu der die Aktion aufgezeichnet wird.
status	integer	Der Status des Datashares.

Spaltenname	Datentyp	Beschreibung
error_message	varchar(512)	Die Meldung eines Fehlers.
consumer_region	char(64)	Die Region, in der sich der Konsumenten-Cluster befindet.

Beispielabfragen

Das folgende Beispiel zeigt die Ansicht SYS_DATASHARE_USAGE_PRODUCER.

```
SELECT DISTINCT
FROM sys_datashare_usage_producer
WHERE object_name LIKE 'tickit%';

   request_type
 ------------------
   "GET RELATION"
```

SYS_EXTERNAL_QUERY_DETAIL

Verwenden Sie SYS_QUERY_DETAIL, um Details für Abfragen auf Segmentebene anzuzeigen. Jede Zeile repräsentiert ein Segment aus einer bestimmten WLM-Abfrage mit Details wie der Anzahl der verarbeiteten Zeilen, der Anzahl der verarbeiteten Bytes und Partitionsinformationen externer Tabellen in Amazon S3. Jede Zeile in dieser Ansicht hat auch einen entsprechenden Eintrag in der Ansicht SYS_QUERY_DETAIL, außer dass diese Ansicht detailliertere Informationen zur externen Abfrageverarbeitung enthält.

SYS_EXTERNAL_QUERY_DETAIL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Abfrage gesendet hat.
query_id	bigint	Die Abfrage-ID der externen Abfrage.
transaction_id	bigint	Die Transaktions-ID.
child_query_sequence	integer	Die Reihenfolge der neu geschriebenen Benutzera bfrage. Beginnt mit 0, ähnlich wie segment_id.
segment_id	integer	Die Segment-ID des Abfragesegments.
source_type	character(32)	Der Datenquellentyp der Abfrage. Es kann sich um S3 für Redshift Spectrum und PG für Verbundabfragen handeln.
start_time	timestamp	Der Zeitpunkt, zu dem die Abfrage begann.
end_time	timestamp	Der Zeitpunkt, an dem die Abfrage abgeschlossen wurde.
duration	bigint	Die Zeit (Mikrosekunden), die für die Abfrage aufgewendet wurde.
total_partitions	integer	Die Anzahl der Partitionen, die eine Amazon-S3-Abfrage benötigt hat.

Spaltenname	Datentyp	Beschreibung
qualified_partitions	integer	Die Anzahl der Partitionen, die eine Amazon-S3-Abfrage gescannt hat.
scanned_files	bigint	Die Anzahl der von Amazon S3 gescannten Dateien.
returned_rows	bigint	Die Anzahl der gescannten Zeilen für eine Amazon-S3-Abfrage oder die Anzahl der zurückgegebenen Zeilen für eine Verbundabfrage.
returned_bytes	bigint	Die Anzahl der gescannten Bytes für eine Amazon-S3-Abfrage oder die Anzahl der zurückgegebenen Bytes für eine Verbundabfrage.
file_format	text	Das Dateiformat von Amazon-S3-Dateien.
file_location	text	Der Amazon-S3-Speicherort der externen Tabelle.
external_query_text	text	Der Abfragetext auf Segmentebene für eine Verbundabfrage.
warning_message	character(4000)	Die Warnmeldung, die angezeigt wird, wenn die Abfrage ausgeführt wird.
table_name	character(136)	Der Tabellenname des Schritts, der ausgeführt wird.

Spaltenname	Datentyp	Beschreibung
is_recursive	character(1)	Zeigt an, ob ein rekursive r Scan für Unterordner vorhanden ist.
is_nested	character(1)	Zeigt an, ob auf den Datentyp der verschachtelten Spalte zugegriffen wird.
s3list_time	bigint	Die Dauer der Dateiauflistung in Millisekunden.
get_partition_time	long	Zeit, die für das Auflisten und Qualifizieren von Partitionen für ein bestimmtes externes Objekt aus Apache Hive aufgewendet wurde. AWS Glue Data Catalog

Beispielabfragen

Die folgende Abfrage zeigt die Details der externen Abfrage.

```
SELECT query_id,
       segment_id,
       start_time,
       end_time,
       total_partitions,
       qualified_partitions,
       scanned_files,
       returned_rows,
       returned_bytes,
       trim(external_query_text) query_text,
       trim(file_location) file_location
FROM sys_external_query_detail
ORDER BY query_id, start_time DESC
LIMIT 2;
```

Beispielausgabe.

```
 query_id | segment_id |         start_time         |          end_time
   | total_partitions | qualified_partitions | scanned_files | returned_rows |
 returned_bytes | query_text | file_location
----------+------------+----------------------------+----------------------------
+------------------+----------------------+--------------+--------------
+----------------+------------+--------------
   763251 |          0 | 2022-02-15 22:32:23.312448 | 2022-02-15 22:32:24.036023 |
          3 |                    3 |             3 |         38203 |        2683414 |
          |
   763254 |          0 | 2022-02-15 22:32:40.17103  | 2022-02-15 22:32:40.839313 |
          3 |                    3 |             3 |         38203 |        2683414 |
          |
```

SYS_EXTERNAL_QUERY_ERROR

Sie können die Systemansicht SYS_EXTERNAL_QUERY_ERROR abfragen, um Informationen zu Redshift-Spectrum-Scanfehlern zu erhalten. SYS_EXTERNAL_QUERY_ERROR zeigt ein Beispiel für protokollierte Fehler an. Der Standardwert ist 10 Einträge pro Abfrage.

SYS_EXTERNAL_QUERY_ERROR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Der Bezeichner des Benutzers, der die Zeile generiert hat.
query_id	bigint	Der Bezeichner der Abfrage, die diese Zeile generiert hat.
file_location	char(256)	Der Speicherort der abgefragten Daten.
rowid	char(2100)	Die Fehlerstelle innerhalb der Datei. Die `rowid`-Teile werden durch einen `:` (Doppelpunkt) getrennt und weitere Teile können zu einem späteren Zeitpunkt hinzugefügt werden.

Spaltenname	Datentyp	Beschreibung
		`row_offset :row_group :row_id` Ein row_offset ist der Offset (in Byte) der Zeile innerhalb der Datei. Er wird für nicht unterstützte Dateiformate auf -1 festgelegt. Eine Tabelle ist in row_groups unterteilt. Jede Gruppe verfügt über Zeilen mit unterschiedlichen row_ids.
column_name	char(127)	Der Name der Spalte, die von der Abfrage zurückgegeben wird.
original_value	char(1024)	Ursprünglicher Wert wurde abgefragt.
modified_value	char(1024)	Der geänderte Wert wurde basierend auf der in der Abfrage angegebenen Konfigurationsoption für die Datenverarbeitung zurückgegeben.
trigger	char(128)	In der Abfrage angegebene Datenverarbeitungsoption.
action	char(128)	Aktion, die mit der in der Abfrage angegebenen Datenverarbeitungsoption verknüpft ist.
action_value	char(128)	Wert des Aktionsparameters, der mit der in der Abfrage angegebenen Datenverarbeitungsoption verbunden ist.
error_code	Ganzzahl	Ergebniscode der in der Abfrage angegebenen Datenverarbeitungsoption.

Beispielabfrage

Die folgende Abfrage gibt die Liste der Zeilen zurück, für die Datenbehandlungsvorgänge ausgeführt wurden.

```
SELECT * FROM sys_external_query_error;
```

Die Abfrage gibt Ergebnisse wie die folgenden zurück.

```
  user_id   query_id  file_location                                  rowid
column_name           original_value             modified_value       trigger
action               action_value                error_code
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0
league_nspi           34595                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:1
league_nspi           34151                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2
league_nspi           33223                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3
league_nspi           32808                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:4
league_nspi           32790                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:5
league_name           Spanish Primera Division   Spanish Primera Divi UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:6
league_name           Spanish Primera Division   Spanish Primera Divi UNSPECIFIED
TRUNCATE                                          156
```

SYS_INTEGRATION_ACTIVITY

SYS_INTEGRATION_ACTIVITY zeigt Details zu abgeschlossenen Integrationsvorgängen an.

SYS_INTEGRATION_ACTIVITY ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu Null-ETL-Integrationen finden Sie unter Arbeiten mit Null-ETL-Integrationen im Amazon-Redshift-Verwaltungshandbuch.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
integration_id	character (128)	Die Kennung, die der Integration zugeordnet ist.
target_database	character (128)	Die Datenbank in Amazon Redshift, die die Integrationsdaten empfängt.
Quelle	character (128)	Die Quelldaten für die Integration. Mögliche Typen sind u. a. `MySQL` und `PostgreSQL` .
checkpoint_name	character (128)	Der Name des Checkpoints, der Binlog-Koordinaten repliziert.
checkpoint_type	character (16)	Die Art des Checkpoints. Mögliche Werte sind: `snapshot`, `cdc`.
checkpoint_bytes	bigint	Die Anzahl der Byte in diesem Checkpoint.
last_commit_timestamp	Zeitstempel	Der Zeitstempel der letzten Übertragung in diesem Checkpoint.
modified_tables	Ganzzahl	Die Anzahl der Tabellen, die in dem Checkpoint geändert wurden.
integration_start_time	time	Die Uhrzeit (UTC), zu der die Integration für diesen Checkpoint gestartet wurde.
integration_end_time	time	Die Uhrzeit (UTC), zu der die Integration für diesen Checkpoint endete.

Beispielabfragen

Der folgende SQL-Befehl zeigt das Protokoll der Integrationen an.

```
select * from sys_integration_activity;

          integration_id              | target_database | source |
   checkpoint_name                  | checkpoint_type  | checkpoint_bytes |
 last_commit_timestamp   | modified_tables |   integration_start_time   |
 integration_end_time
-------------------------------------+-----------------+--------
+--------------------------------------------+------------------+------------------
+-------------------------+-----------------+---------------------------
+---------------------------
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_241_3_510.json     |        cdc       |        762       | 2023-05-10
 23:00:14.201 |         1       | 2023-05-10 23:00:45.054265 | 2023-05-10
 23:00:46.339826
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_16329_3_17839.json |        cdc       |       13488      | 2023-05-11
 01:33:57.411 |         2       | 2023-05-11 02:19:09.440121 | 2023-05-11
 02:19:16.090492
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_5103_3_5532.json   |        cdc       |        1657      | 2023-05-10
 23:13:14.205 |         2       | 2023-05-10 23:13:23.545487 | 2023-05-10
 23:13:25.652144
```

SYS_INTEGRATION_TABLE_STATE_CHANGE

SYS_INTEGRATION_TABLE_STATE_CHANGE zeigt Details zu den Änderungsprotokollen für den Tabellenstatus für Integrationen an.

Ein Superuser kann alle Zeilen in dieser Tabelle sehen.

Weitere Informationen finden Sie unter Arbeiten mit Zero-ETL-Integrationen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
integration_id	character (128)	Die Kennung, die der Integration zugeordnet ist.

Spaltenname	Datentyp	Beschreibung
database_name	character (128)	Der Name der Amazon-Redshift-Datenbank.
schema_name	character (128)	Der Name des Amazon-Redshift-Schemas.
table_name	character (128)	Der Name der Tabelle.
new_state	character (128)	Der Zustand der Tabelle. Mögliche Werte sind `Synced`, `ResyncRequired` , `ResyncInitiated` , `Deleted`, `Failed` und `ResyncDeleted` .
table_last_replicated_checkpoint	character (128)	Die aktuellen synchronisierten Protokoll-Koordinaten.
Grund für die Änderung des Bundesstaates	character (256)	Der Grund für den letzten Statuswechsel.
record_time	Zeitstempel	Die Uhrzeit (UTC), zu der dieser Datensatz aktualisiert wurde.

Beispielabfragen

Der folgende SQL-Befehl zeigt das Protokoll der Integrationen an.

```
select * from sys_integration_table_state_change;

            integration_id            | database_name | schema_name | table_name
 | new_state |  table_last_replicated_checkpoint   | state_change_reason |
 record_time
--------------------------------------+---------------+-------------+------------
+-----------+--------------------------------------+---------------------
+----------------------------
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest79   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                     | 2023-09-20
 19:39:50.087868
```

```
99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest56   |
Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
19:39:45.54005
99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest50   |
Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
19:40:20.362504
99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest80t3s | sbtest18   |
Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
19:40:32.544084
99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb        | sbtest40t3s | sbtest23   |
Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
15:49:05.186209
```

SYS_LOAD_DETAIL

Gibt Informationen zur Nachverfolgung eines Datenladevorgangs bzw. zur Fehlerbehebung aus.

Diese Ansicht zeichnet den Fortschritt jeder Datendatei auf, während sie in eine Datenbanktabelle geladen wird.

Diese Ansicht ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	ID des Benutzers, der den Eintrag generiert hat.
query_id	Ganzzahl	Abfrage-ID.
file_name	character(256)	Der zu ladende Dateiname.
bytes_scanned	Ganzzahl	Die Anzahl der von der Datei in Amazon S3 gescannten Bytes.
lines_scanned	integer	Anzahl der aus der Ladedatei gescannten Zeilen. Diese Zahl stimmt möglicherweise nicht mit der Anzahl der tatsächlich geladen Zeilen überein. So kann der Ladevorgang beispiels

Spaltenname	Datentyp	Beschreibung
		weise eine Anzahl ungültiger Datensätze scannen und tolerieren, je nach der MAXERROR-Option im COPY-Befehl.
record_time	Zeitstempel	Zeitpunkt der letzten Aktualisierung dieses Eintrags.
splits_scanned	Anzahl der Splits dieser Datei.	Anzahl der Splits dieser Datei.
start_time	Zeitstempel	Uhrzeit, zu der diese Dateiverarbeitung gestartet wurde.
end_time	Zeitstempel	Uhrzeit, zu der diese Dateiverarbeitung abgeschlossen wurde.

Beispielabfragen

Das folgende Beispiel gibt Details zur letzten COPY-Operation aus.

```
select query_id, trim(file_name) as file, record_time
from sys_load_detail
where query_id = pg_last_copy_id();

 query_id |               file                |          record_time
----------+-----------------------------------+----------------------------
 28554    | s3://dw-tickit/category_pipe.txt | 2013-11-01 17:14:52.648486
(1 row)
```

Die folgende Abfrage enthält Einträge für einen neuen Ladevorgang der Tabellen in der TICKIT-Datenbank:

```
select query_id, trim(file_name), record_time
from sys_load_detail
where file_name like '%tickit%' order by query_id;

 query_id |           btrim           |          record_time
----------+---------------------------+----------------------------
 22475    | tickit/allusers_pipe.txt | 2013-02-08 20:58:23.274186
 22478    | tickit/venue_pipe.txt    | 2013-02-08 20:58:25.070604
 22480    | tickit/category_pipe.txt | 2013-02-08 20:58:27.333472
```

```
 22482    | tickit/date2008_pipe.txt | 2013-02-08 20:58:28.608305
 22485    | tickit/allevents_pipe.txt| 2013-02-08 20:58:29.99489
 22487    | tickit/listings_pipe.txt | 2013-02-08 20:58:37.632939
 22593    | tickit/allusers_pipe.txt | 2013-02-08 21:04:08.400491
 22596    | tickit/venue_pipe.txt    | 2013-02-08 21:04:10.056055
 22598    | tickit/category_pipe.txt | 2013-02-08 21:04:11.465049
 22600    | tickit/date2008_pipe.txt | 2013-02-08 21:04:12.461502
 22603    | tickit/allevents_pipe.txt| 2013-02-08 21:04:14.785124
 22605    | tickit/listings_pipe.txt | 2013-02-08 21:04:20.170594

(12 rows)
```

Die Tatsache, dass ein Datensatz in die Protokolldatei für diese Systemansicht geschrieben wird, bedeutet nicht, dass der Ladevorgang erfolgreich im Rahmen seiner enthaltenden Transaktion bestätigt wurde. Fragen Sie zur Prüfung von Ladebestätigungen die Ansicht STL_UTILITYTEXT ab und suchen Sie nach dem COMMIT-Datensatz, der einer COPY-Transaktion entspricht. Beispielsweise verbindet diese Abfrage SYS_LOAD_DETAIL und STL_QUERY auf der Grundlage einer Unterabfrage für STL_UTILITYTEXT:

```
select l.query_id,rtrim(l.file_name),q.xid
from sys_load_detail l, stl_query q
where l.query_id=q.query
and exists
(select xid from stl_utilitytext where xid=q.xid and rtrim("text")='COMMIT');

 query_id |           rtrim           |  xid
----------+---------------------------+-------
 22600    | tickit/date2008_pipe.txt  | 68311
 22480    | tickit/category_pipe.txt  | 68066
  7508    | allusers_pipe.txt         | 23365
  7552    | category_pipe.txt         | 23415
  7576    | allevents_pipe.txt        | 23429
  7516    | venue_pipe.txt            | 23390
  7604    | listings_pipe.txt         | 23445
 22596    | tickit/venue_pipe.txt     | 68309
 22605    | tickit/listings_pipe.txt  | 68316
 22593    | tickit/allusers_pipe.txt  | 68305
 22485    | tickit/allevents_pipe.txt | 68071
  7561    | allevents_pipe.txt        | 23429
  7541    | category_pipe.txt         | 23415
  7558    | date2008_pipe.txt         | 23428
 22478    | tickit/venue_pipe.txt     | 68065
```

```
   526    | date2008_pipe.txt         |  2572
  7466    | allusers_pipe.txt         | 23365
 22482    | tickit/date2008_pipe.txt  | 68067
 22598    | tickit/category_pipe.txt  | 68310
 22603    | tickit/allevents_pipe.txt | 68315
 22475    | tickit/allusers_pipe.txt  | 68061
   547    | date2008_pipe.txt         |  2572
 22487    | tickit/listings_pipe.txt  | 68072
  7531    | venue_pipe.txt            | 23390
  7583    | listings_pipe.txt         | 23445
(25 rows)
```

SYS_LOAD_ERROR_DETAIL

Verwenden Sie SYS_LOAD_ERROR_DETAIL, um Details zu Fehlern bei COPY-Befehlen anzuzeigen. Jede Zeile repräsentiert einen COPY-Befehl. Sie enthält sowohl ausgeführte als auch abgeschlossene COPY-Befehle.

SYS_LOAD_ERROR_DETAIL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Kopie gesendet hat.
query_id	bigint	Die Abfrage-ID der Kopie.
transaction_id	bigint	Die Transaktions-ID.
session_id	integer	Die ID des Prozesses, der die Kopie ausführt.
database_name	character(64)	Der Name der Datenbank, mit der der Benutzer verbunden war, als die Kopie ausgegeben wurde.

Spaltenname	Datentyp	Beschreibung
table_id	integer	Die Tabellenkennung.
start_time	timestamp	Die Uhrzeit (in UTC), zu der die Kopie begann.
file_name	character(256)	Der vollständige Pfad zur zu ladenden Eingabedatei.
line_number	bigint	Die Zeilennummer in der Ladedatei mit dem Fehler. Beim Laden einer JSON-Datei ist dies die Nummer der letzten Zeile des JSON-Objekts mit dem Fehler.
column_name	character(127)	Das Feld mit dem Fehler.
column_type	character(10)	Der Datentyp des Felds mit dem Fehler.
column_length	character(10)	Die Spaltenlänge, falls zutreffend. Dieses Feld ist gefüllt, wenn für den Datentyp eine Längenbegrenzung gilt. So enthält diese Spalte beispielsweise für eine Spalte mit dem Datentyp „character(3)“ den Wert „3“.
position	integer	Die Position des Fehlers in dem Feld.
error_code	integer	Der Fehlercode.
error_message	character(512)	Die Erläuterung des Fehlers.

Beispielabfragen

Die folgende Abfrage zeigt die Ladefehlerdetails des copy-Befehls für eine bestimmte Abfrage.

```
SELECT query_id,
       table_id,
       start_time,
       trim(file_name) AS file_name,
       trim(column_name) AS column_name,
       trim(column_type) AS column_type,
       trim(error_message) AS error_message
FROM sys_load_error_detail
WHERE query_id = 762949
ORDER BY start_time
LIMIT 10;
```

Beispielausgabe.

```
 query_id | table_id |         start_time         |               file_name
      | column_name | column_type |                 error_message
----------+----------+----------------------------
+----------------------------------------+------------+------------
+-----------------------------------------------
   762949 |   137885 | 2022-02-15 22:14:46.759151 | s3://load-test/copyfail/
wrong_format_000 | id          | int4        | Invalid digit, Value 'a', Pos 0, Type:
 Integer
   762949 |   137885 | 2022-02-15 22:14:46.759151 | s3://load-test/copyfail/
wrong_format_001 | id          | int4        | Invalid digit, Value 'a', Pos 0, Type:
 Integer
```

SYS_LOAD_HISTORY

Verwenden Sie SYS_LOAD_HISTORY, um Details zu COPY-Befehlen anzuzeigen. Jede Zeile stellt einen COPY-Befehl mit akkumulierten Statistiken für einige der Felder dar. Sie enthält sowohl ausgeführte als auch abgeschlossene COPY-Befehle.

SYS_LOAD_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Kopie gesendet hat.
query_id	bigint	Die Abfrage-ID der Kopie.
transaction_id	bigint	Die Transaktions-ID.
session_id	integer	Die ID des Prozesses, der die Kopie ausführt.
database_name	text	Der Name der Datenbank , mit der der Benutzer verbunden war, als der Vorgang ausgegeben wurde.
status	text	Der Status der Kopie. Gültige Werte sind `running`, `completed` , `aborted`.
table_name	text	Der Name der Tabelle, die Informationen kopiert hat.
start_time	timestamp	Der Zeitpunkt, zu dem die Kopie begann.
end_time	timestamp	Der Zeitpunkt, an dem die Kopie abgeschlossen wurde.
duration	bigint	Die Dauer (in Mikrosekunden) des COPY-Befehls.
data_source	text	Der Amazon-S3-Speicherort der zu kopierenden Dateien.
file_format	text	Das Format der Quelldatei. Zu den Formaten gehören

Spaltenname	Datentyp	Beschreibung
		csv, txt, json, avro, orc oder parquet.
loaded_rows	bigint	Die Anzahl der in eine Tabelle kopierten Zeilen.
loaded_bytes	bigint	Die Anzahl der in eine Tabelle kopierten Bytes.
source_file_count	integer	Die Anzahl der Dateien in Quelldateien.
source_file_bytes	bigint	Die Anzahl der Bytes in Quelldateien.
file_count_scanned	Ganzzahl	Die Anzahl der gescannten Dateien von Amazon S3.
file_bytes_scanned	bigint	Die Anzahl der von der Datei in Amazon S3 gescannten Bytes.
error_count	bigint	Die Anzahl der Fehler.
copy_job_id	bigint	Die Kennung des Kopierauftrags. `0` gibt an, dass keine Auftragskennung vorhanden ist.

Beispielabfragen

Die folgende Abfrage zeigt die geladenen Zeilen, Bytes, Tabellen und Datenquellen bestimmter Kopierbefehle.

```
SELECT query_id,
       table_name,
       data_source,
       loaded_rows,
```

```
       loaded_bytes
FROM sys_load_history
WHERE query_id IN (6389,490791,441663,74374,72297)
ORDER BY query_id,
         data_source DESC;
```

Beispielausgabe.

```
 query_id |    table_name    |                                data_source
              | loaded_rows | loaded_bytes
----------+------------------
+-------------------------------------------------------------------------+-------------
+---------------
     6389 | store_returns    | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
store_returns/    |   287999764 | 1196240296158
    72297 | web_site         | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
web_site/         |          54 |         43808
    74374 | ship_mode        | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
ship_mode/        |          20 |          1320
   441663 | income_band      | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
income_band/      |          20 |          2152
   490791 | customer_address | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
customer_address/ |     6000000 |     722924305
```

Die folgende Abfrage zeigt die geladenen Zeilen, Bytes, Tabellen und Datenquellen von Kopierbefehlen.

```
SELECT query_id,
       table_name,
       data_source,
       loaded_rows,
       loaded_bytes
FROM sys_load_history
ORDER BY query_id DESC
LIMIT 10;
```

Beispielausgabe.

```
 query_id |       table_name       |                                data_source
                         | loaded_rows |  loaded_bytes
```

```
----------+------------------------
+-------------------------------------------------------------------------------
+-------------+-----------------
   491058 | web_site               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_site/               |          54 |           43808
   490947 | web_sales              | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_sales/              |   720000376 |  22971988122819
   490923 | web_returns            | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_returns/            |    71997522 |     96597496325
   490918 | web_page               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_page/               |        3000 |            1320
   490907 | warehouse              | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/warehouse/              |          20 |            1320
   490902 | time_dim               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/time_dim/               |       86400 |            1320
   490876 | store_sales            | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store_sales/            |  2879987999 | 151666241887933
   490870 | store_returns          | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store_returns/          |   287999764 |   1196405607941
   490865 | store                  | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store/                  |        1002 |          365507
```

Die folgende Abfrage zeigt die täglich geladenen Zeilen und Byte des copy-Befehls.

```
SELECT date_trunc('day',start_time) AS exec_day,
       SUM(loaded_rows) AS loaded_rows,
       SUM(loaded_bytes) AS loaded_bytes
FROM sys_load_history
GROUP BY exec_day
ORDER BY exec_day DESC;
```

Beispielausgabe.

```
      exec_day       | loaded_rows |   loaded_bytes
---------------------+-------------+------------------
 2022-01-20 00:00:00 |  6347386005 |  258329473070606
 2022-01-19 00:00:00 | 19042158015 |  775198502204572
 2022-01-18 00:00:00 | 38084316030 | 1550294469446883
 2022-01-17 00:00:00 | 25389544020 | 1033271084791724
 2022-01-16 00:00:00 | 19042158015 |  775222736252792
 2022-01-15 00:00:00 | 19834245387 |  798122849155598
 2022-01-14 00:00:00 | 75376544688 | 3077040926571384
```

SYS_MV_REFRESH_HISTORY

Die Ergebnisse enthalten Informationen über den Aktualisierungsverlauf aller materialisierten Ansichten. Zu den Ergebnissen gehören der Aktualisierungstyp, z. B. manuell oder automatisch, und der Status der letzten Aktualisierung.

SYS_MV_REFRESH_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die ID des Benutzers, der den Aktualisierungsvorgang gestartet hat.
session_id	Ganzzahl	Die Prozess-ID für den Prozess, der die Aktualisierung der materialisierten Ansicht ausführt.
transaction_id	bigint	Die Transaktions-ID.
database_name	char(128)	Die Datenbank, die die materialisierte Ansicht enthält.
schema_name	char(128)	Das Schema der materialisierten Ansicht.
mv_id	bigint	Objekt-ID der materialisierten Ansicht.
mv_name	char(128)	Der Name der materialisierten Ansicht.

Spaltenname	Datentyp	Beschreibung
refresh_type	char(32)	Die Art der Aktualisierung, z. B. manuell oder automatisch.
status	Text	Der Status der Aktualisierung. Detaillierte Informationen zu den Status finden Sie in der Statusspalte für SVL_MV_REFRESH_STATUS.
start_time	Zeitstempel	Die Anfangszeit der Aktualisierung.
end_time	Zeitstempel	Die Endzeit der Aktualisierung.
duration	bigint	Die Zeit in Mikrosekunden, die benötigt wurde, um die materialisierte Ansicht zu aktualisieren.

Beispielabfragen

Die folgende Abfrage zeigt den Aktualisierungsverlauf für materialisierte Ansichten.

```
SELECT user_id,
     session_id,
     transaction_id,
     database_name,
     schema_name,
     mv_id,
     mv_name,
     refresh_type,
     status,
     start_time,
     end_time,
     duration
     from sys_mv_refresh_history;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
 user_id | session_id | transaction_id | database_name | schema_name                |
 mv_id  |  mv_name           |  refresh_type  |  status
         |  start_time                |  end_time                  |  duration
---------+------------+----------------+--------------
+---------------------------+--------+--------------------+---------------
+-----------------------------------------------+--------------------------
+---------------------------+-----------
       1 | 1073815659 |          15066 | dev           | test_stl_mv_refresh_schema |
 203762 | mv_incremental     | Manual         | MV was already updated
         | 2023-10-26 15:59:20.952179 | 2023-10-26 15:59:20.952866 |      687
       1 | 1073815659 |          15068 | dev           | test_stl_mv_refresh_schema |
 203771 | mv_nonincremental  | Manual         | MV was already updated
         | 2023-10-26 15:59:21.008049 | 2023-10-26 15:59:21.008658 |      609
       1 | 1073815659 |          15070 | dev           | test_stl_mv_refresh_schema |
 203779 | mv_refresh_error   | Manual         | MV was already updated
         | 2023-10-26 15:59:21.064252 | 2023-10-26 15:59:21.064885 |      633
       1 | 1073815659 |          15074 | dev           | test_stl_mv_refresh_schema
 | 203762 | mv_incremental     | Manual         | Refresh successfully updated MV
 incrementally   | 2023-10-26 15:59:29.693329 | 2023-10-26 15:59:43.482842 | 13789513
       1 | 1073815659 |          15076 | dev           | test_stl_mv_refresh_schema |
 203771 | mv_nonincremental  | Manual         | Refresh successfully recomputed MV from
 scratch | 2023-10-26 15:59:43.550184 | 2023-10-26 15:59:47.880833 |  4330649
       1 | 1073815659 |          15078 | dev           | test_stl_mv_refresh_schema |
 203779 | mv_refresh_error   | Manual         | Refresh failed due to an internal error
         | 2023-10-26 15:59:47.949052 | 2023-10-26 15:59:52.494681 |  4545629
(6 rows)
```

SYS_MV_STATE

Die Ergebnisse enthalten Informationen über den Status aller materialisierten Ansichten. Es sind Informationen zur Basistabelle, Schemaeigenschaften und Informationen über aktuelle Ereignisse enthalten, wie z. B. das Löschen einer Spalte.

SYS_MV_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	bigint	Die ID des Benutzers, der das Ereignis erstellt hat.
transaction_id	bigint	Die Transaktions-ID des Ereignisses.
database_name	char(128)	Die Datenbank, die die materialisierte Ansicht enthält.
event_desc	char(500)	Das Ereignis, das die Statusänderung veranlasst hat. Zu den Beispielwerten gehören: • Spaltentyp wurde geändert • Spalte wurde gelöscht • Spalte wurde umbenannt • Schemaname wurde geändert • Konvertierung kleiner Tabellen • TRUNCATE • Vacuum Beachten Sie, dass es andere mögliche Werte für diese Spalte gibt.
start_time	Zeitstempel	Die Anfangszeit des Build.
base_table_database_name	char(128)	Der Name der Datenbank für die Basistabelle.

Spaltenname	Datentyp	Beschreibung
base_table_schema	char(128)	Das Schema der Basistabelle.
base_table_name	char(128)	Der Name der Basistabelle.
mv_schema	char(128)	Das Schema der materiali sierten Ansicht.
mv_name	char(128)	Der Name der materialisierten Ansicht.
state	character(32)	Der geänderte Status der materialisierten Ansicht, der wie folgt lautet: • Erneutes Berechnen • Nicht aktualisierbar

Beispielabfragen

Die folgende Abfrage zeigt den Status der materialisierten Ansichten an.

```
select * from sys_mv_state;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
 user_id | transaction_id | database_name | event_desc                  | start_time
              | base_table_database_name | base_table_schema | base_table_name     |
 mv_schema  | mv_name                    | state
---------+----------------+---------------+-----------------------------
+-----------------------------+-------------------------+------------------
+---------------------+-------------+----------------------------+--------------
 106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
 14:59:12.788268 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_a1f3f862 | Recompute
 106     | 12724          | tickit_db     | ALTER TABLE ALTER DISTSTYLE | 2023-07-26
 14:59:51.409014 | tickit_db                | mv_schema         | test_table_58102435 |
 mv_schema   | materialized_view_ca746631 | Recompute
```

```
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.822928 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5750a8d4 | Unrefreshable
106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
15:00:08.051244 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5750a8d4 | Unrefreshable
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.857755 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5750a8d4 | Unrefreshable
106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
15:00:08.051358 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5ef0d754 | Unrefreshable
106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
14:59:12.788159 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5750a8d4 | Recompute
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.857799 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
14:59:12.788327 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_5ef0d754 | Recompute
106     | 12727          | tickit_db     | ALTER TABLE ALTER SORTKEY   | 2023-07-26
15:00:08.006235 | tickit_db                 | mv_schema         | test_table_58102435 |
mv_schema   | materialized_view_ca746631 | Recompute
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.82297  | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
15:00:08.051321 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
```

SYS_PROCEDURE_CALL

Verwenden Sie die Ansicht SYS_PROCEDURE_CALL zum Abruf von Informationen zu Aufrufen gespeicherter Prozeduren, einschließlich Startzeit, Endzeit, Status des gespeicherten Prozeduraufrufs und Aufrufhierarchie für verschachtelte Aufrufe gespeicherter Prozeduren. Jeder Aufruf einer gespeicherten Prozedur erhält eine Abfrage-ID.

SYS_STORED_PROC_CALL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
session_user_id	Ganzzahl	Die ID des Benutzers, der die Sitzung erstellt hat und der Aufrufer des Aufrufs der gespeicherten Prozedur auf oberster Ebene ist.
security_user_id	Ganzzahl	Die ID des Benutzers, dessen Berechtigungen zum Ausführen der Anweisung in der gespeicherten Prozedur benutzt wurden. Wenn es sich bei der gespeicherten Prozedur um DEFINER handelt, ist dies die Benutzer-ID des Eigentümers der gespeicherten Prozedur.
query_id	Ganzzahl	Die Abfrage-ID des gespeicherten Prozeduraufrufs.
query_text	char(4000)	Der Text der Aufrufabfrage der gespeicherten Prozedur.
start_time	Zeitstempel	Zeitpunkt des Beginns der Abfrage, in UTC. Der Zeitstempel nutzt eine Genauigkeit von sechs Ziffern für Sekundenbruchteile, zum Beispiel 2009-06-12 11:29:19.131358.
end_time	Zeitstempel	Zeitpunkt der Beendigung der Abfrage, in UTC. Der Zeitstempel nutzt eine

Spaltenname	Datentyp	Beschreibung
		Genauigkeit von sechs Ziffern für Sekundenbruchteile, zum Beispiel: 2009-06-12 11:29:19.131358.
Status	char(10)	Der Status des Aufrufs der gespeicherten Prozedur. Wenn die gespeicherte Prozedur vom System angehalten oder vom Benutzer abgebrochen wurde, wird der Wert „abgebrochen“. Wenn der Aufruf der gespeicherten Prozedur abgeschlossen wurde, ist der Wert „erfolgre ich“.
caller_procedure_query_id	Ganzzahl	Wenn die Prozedur von einem anderen Prozeduraufruf aufgerufen wurde, enthält diese Spalte die Abfrage-ID des externen Aufrufs. Andernfalls ist das Feld NULL.

Beispielabfragen

Die folgende Abfrage gibt eine Hierarchie eines verschachtelten Aufrufs einer gespeicherten Prozedur zurück.

```
select query_id, datediff(seconds, start_time, end_time) as elapsed_time, status,
 trim(query_text) as call, caller_procedure_query_id from sys_procedure_call;
```

Beispielausgabe.

```
 query_id | elapsed_time | status  |                      call                      |
 caller_procedure_query_id
```

```
----------+--------------+---------+-----------------------------------------------------
+--------------------------
     3087 |           18 | success | CALL proc_bd906c98c45443ffa165e9552056902d(1)    |
          3085
     3085 |           18 | success | CALL proc_bd906c98c45443ffa165e9552056902d_2(1); |

(2 rows)
```

SYS_PROCEDURE_MESSAGES

SYS_PROCEDURE_MESSAGES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
transaction_id	bigint	Die Transaktions-ID.
query_id	Ganzzahl	Die Abfrage-ID des gespeicherten Prozeduraufrufs.
record_time	Zeitstempel	Die Zeit in UTC, zu der die Nachricht generiert wurde.
log_level	char(10)	Die Protokollebene der generierten Nachricht. Mögliche Werte sind LOG, INFO, NOTICE, WARNING und EXCEPTION.
Nachricht	char(1024)	Der Text der generierten Meldung.
line_number	Ganzzahl	Die Zeilennummer der generierten Meldung.

Beispielabfragen

Die folgende Abfrage zeigt eine Beispielausgabe von SYS_PROCEDURE_MESSAGES.

```
select transaction_id, query_id, record_time, log_level, trim(message), line_number
 from sys_procedure_messages;
```

```
transaction_id | query_id |        record_time         | log_level |           btrim
        | line_number
---------------+----------+----------------------------+-----------
+--------------------------+-------------
     25267     |   80562  | 2023-07-17 14:38:31.910136 |   NOTICE  |
 test_notice_msg_b9f1e749  |     8
     25267     |   80562  | 2023-07-17 14:38:31.910002 |    LOG    |
 test_log_msg_833c7420     |     6
     25267     |   80562  | 2023-07-17 14:38:31.910111 |    INFO   |
 test_info_msg_651373d9    |     7
     25267     |   80562  | 2023-07-17 14:38:31.910154 |   WARNING |
 test_warning_msg_831c5747 |     9
(4 rows)
```

SYS_QUERY_DETAIL

Verwenden Sie SYS_QUERY_DETAIL, um Details für Abfragen auf Schrittebene anzuzeigen. Jede Zeile stellt einen Schritt aus einer bestimmten WLM-Abfrage mit Details dar. Diese Ansicht enthält viele Arten von Abfragen wie DDL-, DML- und Dienstprogrammbefehle (z. B. copy und unload). Einige Spalten sind je nach Abfragetyp möglicherweise nicht relevant. Beispielsweise ist external_scanned_bytes für interne Tabellen nicht relevant.

SYS_QUERY_DETAIL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Abfrage gesendet hat.

Spaltenname	Datentyp	Beschreibung
query_id	bigint	Die Abfrage-ID.
child_query_sequence	Ganzzahl	Die Reihenfolge der neu geschriebenen Benutzera bfrage, beginnend mit 1.
stream_id	Ganzzahl	Die Stream-Kennung des Abfrage-Streams.
segment_id	Ganzzahl	Die Segmentkennung des Abfrageausführungssegments.
step_id	integer	Die Schritt-ID in einem Segment.
step_name	text	Der Schrittname in einem Segment. Mögliche Werte sind`aggregate` , `broadcast` `delete`,`distribut e` ,`hash`,`hashjoin`,`insert`,`limit`,`me savescan`,`sort`,`sortlimit` ,`unique`, und`window`.
table_id	integer	Die Tabellen-ID für permanent e Tabellenscans.
table_name	character(136)	Der Tabellenname des Schritts, der ausgeführt wird.
is_rrscan	character	Ein Wert, der angibt, ob ein Schritt ein Scanschritt ist. „True“ (t) zeigt an, dass für diesen Schritt ein Scan mit Bereichseinschränkung durchgeführt wurde.

Spaltenname	Datentyp	Beschreibung
start_time	Zeitstempel	Der Zeitpunkt, zu dem der Abfrageschritt begann.
end_time	Zeitstempel	Der Zeitpunkt, an dem der Abfrageschritt abgeschlossen wurde.
duration	bigint	Die Zeit (Mikrosekunden), die für den Schritt aufgewendet wurde.
Warnung	text	Die Beschreibung des Warnungsereignisses.
input_bytes	bigint	Die Eingabebytes für den aktuellen Schritt.
input_rows	bigint	Die Eingabezeilen für den aktuellen Schritt.
output_bytes	bigint	Die Ausgabebytes für den aktuellen Schritt.
output_rows	bigint	Die Ausgabezeilen für den aktuellen Schritt.
blocks_read	bigint	Die Anzahl der Blocks, die der Schritt gelesen hat.
blocks_write	bigint	Die Anzahl der Blocks, die der Schritt geschrieben hat.
local_read_IO	bigint	Die Anzahl der Blöcke, die aus dem lokalen Festplattencache gelesen wurden.

Spaltenname	Datentyp	Beschreibung
remote_read_IO	bigint	Die Anzahl der remote gelesenen Blöcke.
Quelle	text	Der Typ des Datenbank objekts, das gescannt wurde. Diese Spalte hat nur dann einen Wert, wenn die Zeile Schrittname den Wert `scan` hat.
data_skewness	Ganzzahl	Die Schiefe der Verteilung der Ausgabezeilen auf alle Schritte. Eine Zahl im Bereich von 0 bis 100 %. Je größer die Zahl ist, desto unausgewo gener ist die Verteilung.
time_skewness	Ganzzahl	Die Schiefe der Verteilung der Ausführungszeiten auf alle Schritte. Eine Zahl im Bereich von 0 bis 100 %. Je größer die Zahl ist, desto unausgewo gener ist die Verteilung.
is_active	character	Der Status der Abfrage auf Schrittebene. Mögliche Werte sind „t“, was darauf hinweist, dass der Schritt aktiv ausgeführt wird, oder „f“, was darauf hinweist, dass der Schritt abgeschlossen ist.
spilled_block_local_disk	bigint	Die Anzahl der Blöcke, die auf die lokale Festplatte übertrage n wurden.

Spaltenname	Datentyp	Beschreibung
spilled_block_remote_disk	bigint	Die Anzahl der Blöcke, die an Amazon Simple Storage Service weitergegeben wurden.
step_attribute	character(64)	Enthält Informationen zum zugehörigen Schritt. Mögliche Werte für Scan-Schritte: `multi-dimensional` .

Beispielabfragen

Das folgende Beispiel gibt die Ausgabe von SYS_QUERY_DETAIL zurück.

Die folgende Abfrage zeigt die Abfragemetadatendetails auf Schrittebene, einschließlich Schrittname, input_bytes, output_bytes, input_rows, output_rows.

```
SELECT query_id,
       child_query_sequence,
       stream_id,
       segment_id,
       step_id,
       trim(step_name) AS step_name,
       duration,
       input_bytes,
       output_bytes,
       input_rows,
       output_rows
FROM sys_query_detail
WHERE query_id IN (193929)
ORDER BY query_id,
         stream_id,
         segment_id,
         step_id DESC;
```

Beispielausgabe.

```
 query_id | child_query_sequence | stream_id | segment_id | step_id | step_name  |
 duration     | input_bytes | output_bytes | input_rows | output_rows
----------+----------------------+-----------+------------+---------+------------
+-----------------+-------------+--------------+------------+-------------
   193929 |                    2 |         0 |          0 |       3 | hash       |
      37144 |           0 |      9350272 |          0 |      292196
   193929 |                    5 |         0 |          0 |       3 | hash       |
       9492 |           0 |        23360 |          0 |        1460
   193929 |                    1 |         0 |          0 |       3 | hash       |
      46809 |           0 |      9350272 |          0 |      292196
   193929 |                    4 |         0 |          0 |       2 | return     |
       7685 |           0 |          896 |          0 |         112
   193929 |                    1 |         0 |          0 |       2 | project    |
      46809 |           0 |            0 |          0 |      292196
   193929 |                    2 |         0 |          0 |       2 | project    |
      37144 |           0 |            0 |          0 |      292196
   193929 |                    5 |         0 |          0 |       2 | project    |
       9492 |           0 |            0 |          0 |        1460
   193929 |                    3 |         0 |          0 |       2 | return     |
      11033 |           0 |        14336 |          0 |         112
   193929 |                    2 |         0 |          0 |       1 | project    |
      37144 |           0 |            0 |          0 |      292196
   193929 |                    1 |         0 |          0 |       1 | project    |
      46809 |           0 |            0 |          0 |      292196
   193929 |                    5 |         0 |          0 |       1 | project    |
       9492 |           0 |            0 |          0 |        1460
   193929 |                    3 |         0 |          0 |       1 | aggregate  |
      11033 |           0 |       201488 |          0 |          14
   193929 |                    4 |         0 |          0 |       1 | aggregate  |
       7685 |           0 |        28784 |          0 |          14
   193929 |                    5 |         0 |          0 |       0 | scan       |
       9492 |           0 |        23360 |     292196 |        1460
   193929 |                    4 |         0 |          0 |       0 | scan       |
       7685 |           0 |         1344 |        112 |         112
   193929 |                    2 |         0 |          0 |       0 | scan       |
      37144 |           0 |      7304900 |     292196 |      292196
   193929 |                    3 |         0 |          0 |       0 | scan       |
      11033 |           0 |        13440 |        112 |         112
   193929 |                    1 |         0 |          0 |       0 | scan       |
      46809 |           0 |      7304900 |     292196 |      292196
   193929 |                    5 |         0 |          0 |      -1 |            |
       9492 |       12288 |            0 |          0 |           0
```

```
   193929 |                   1 |         0 |          0 |      -1 |           |
      46809 |       16384 |           0 |          0 |          0
   193929 |                   2 |         0 |          0 |      -1 |           |
      37144 |       16384 |           0 |          0 |          0
   193929 |                   4 |         0 |          0 |      -1 |           |
       7685 |       28672 |           0 |          0 |          0
   193929 |                   3 |         0 |          0 |      -1 |           |
      11033 |      114688 |           0 |          0 |          0
```

Verwenden Sie das folgende Beispiel, um die Tabellen in Ihrer Datenbank in der Reihenfolge von den am häufigsten verwendeten zu den am wenigsten verwendeten Tabellen anzuzeigen. Ersetzen Sie *sample_data_dev* durch Ihre eigene Datenbank. Beachten Sie, dass bei dieser Abfrage ab dem Zeitpunkt Abfragen gezählt werden, an dem Ihr Cluster erstellt wird. Ihre Systemansichtsdaten werden jedoch nicht gespeichert, wenn in Ihrem Data Warehouse nicht genügend Speicherplatz vorhanden ist.

```
SELECT table_name, COUNT (DISTINCT query_id)
FROM SYS_QUERY_DETAIL
WHERE table_name LIKE 'sample_data_dev%'
GROUP BY table_name
ORDER BY COUNT(*) DESC;

+---------------------------------+-------+
|           table_name            | count |
+---------------------------------+-------+
| sample_data_dev.tickit.venue    |     4 |
| sample_data_dev.myunload1.venue |     3 |
| sample_data_dev.tickit.listing  |     1 |
| sample_data_dev.tickit.category |     1 |
| sample_data_dev.tickit.users    |     1 |
| sample_data_dev.tickit.date     |     1 |
| sample_data_dev.tickit.sales    |     1 |
| sample_data_dev.tickit.event    |     1 |
+---------------------------------+-------+
```

SYS_QUERY_HISTORY

Verwenden Sie SYS_QUERY_HISTORY, um Details zu Benutzerabfragen anzuzeigen. Jede Zeile stellt eine Benutzerabfrage mit akkumulierten Statistiken für einige der Felder dar. Diese Ansicht enthält viele Arten von Abfragen, wie Data Definition Language (DDL), Data Manipulation Language

(DML), Kopieren, Entladen und Amazon Redshift Spectrum. Sie enthält sowohl ausgeführte als auch abgeschlossene Abfragen.

SYS_QUERY_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Abfrage gesendet hat.
query_id	bigint	Die Abfrage-ID.
query_label	Zeichen (320)	Der Kurzname für die Abfrage.
transaction_id	bigint	Die Transaktions-ID.
session_id	integer	Die ID des Prozesses, der die Abfrage ausführt.
database_name	character(128)	Der Name der Datenbank , mit der der Benutzer verbunden war, als die Abfrage ausgegeben wurde.
query_type	character(32)	Der Abfragetyp, z. B. SELECT, INSERT, UPDATE, UNLOAD, COPY, COMMAND, DDL, UTILITY, CTAS und OTHER.
Status	character(10)	Der Status der Abfrage. Gültige Werte: planning (Planung), queued (In Warteschlange), running (In Ausführung), returning (Wird zurückgegeben), failed

Spaltenname	Datentyp	Beschreibung
		(Fehlgeschlagen), canceled (Abgebrochen) und success (Erfolgreich).
result_cache_hit	Boolesch	Gibt an, ob die Abfrage mit dem Ergebnis-Cache übereinstimmt.
start_time	timestamp	Der Zeitpunkt, zu dem die Abfrage begann.
end_time	timestamp	Der Zeitpunkt, an dem die Abfrage abgeschlossen wurde.
elapsed_time	bigint	Die gesamte Zeit (Mikrosek unden), die für die Abfrage aufgewendet wurde.
queue_time	bigint	Die Gesamtzeit (Mikrosek unden), die für die Abfragewa rteschlange der Serviceklasse aufgewendet wurde.
execution_time	bigint	Die Gesamtdauer (Mikrosek unden) der Ausführung in der Serviceklasse.
error_message	character(512)	Der Grund, warum eine Abfrage fehlgeschlagen ist.
returned_rows	bigint	Die Anzahl der von der Abfrage zurückgegebenen Zeilen.

Spaltenname	Datentyp	Beschreibung
returned_bytes	bigint	Die Anzahl der von der Abfrage zurückgegebenen Bytes.
query_text	character(4000)	Die Abfragezeichenfolge. Diese Zeichenfolge wird möglicherweise abgeschnitten.
redshift_version	character(256)	Die Amazon-Redshift-Version, als die Abfrage ausgeführt wurde.
usage_limit	character(150)	Liste der von der Abfrage erreichten Nutzungslimit-IDs.
compute_type	varchar(32)	Gibt an, ob die Abfrage auf dem Haupt-Cluster oder einem Nebenläufigkeitssk alierungs-Cluster ausgeführ t wird. Mögliche Werte sind `primary` (Abfrage wird auf dem Haupt-Cluster ausgeführ t), `secondary` (Abfrage wird auf dem sekundären Cluster ausgeführt) oder `primary-s cale` (Abfrage wird auf dem Parallelitäts-Cluster ausgeführ t). Dies gilt nur für bereitges tellte Cluster.
compile_time	bigint	Die Gesamtzeit (Mikrosek unden), die für die Kompilier ung der Abfrage aufgewendet wurde.

Spaltenname	Datentyp	Beschreibung
planning_time	bigint	Die Gesamtzeit (Mikrosek unden), die für die Planung der Abfrage aufgewendet wurde.
lock_wait_time	bigint	Die Gesamtzeit (Mikrosek unden), die für das Warten auf die Sperrbeziehung aufgewendet wurde.

Beispielabfragen

Die folgende Abfrage gibt laufende und in der Warteschlange stehende Abfragen zurück.

```
SELECT user_id,
       query_id,
       transaction_id,
       session_id,
       status,
       trim(database_name) AS database_name,
       start_time,
       end_time,
       result_cache_hit,
       elapsed_time,
       queue_time,
       execution_time
FROM sys_query_history
WHERE status IN ('running','queued')
ORDER BY start_time;
```

Beispielausgabe.

```
 user_id | query_id | transaction_id | session_id | status  | database_name |
 start_time         |          end_time          | result_cache_hit | elapsed_time |
 queue_time | execution_time
---------+----------+----------------+------------+---------+---------------
+---------------------------+---------------------------+-----------------
+--------------+------------+----------------
```

```
     101 |   760705 |         852337 | 1073832321 | running | tpcds_1t      |
 2022-02-15 19:03:19.67849 | 2022-02-15 19:03:19.739811 | f               |
 61321 |          0 |              0
```

Die folgende Abfrage gibt Startzeit, Endzeit, Warteschlangenzeit, verstrichene Zeit, Planungszeit und andere Metadaten für eine bestimmte Abfrage zurück.

```
SELECT user_id,
       query_id,
       transaction_id,
       session_id,
       status,
       trim(database_name) AS database_name,
       start_time,
       end_time,
       result_cache_hit,
       elapsed_time,
       queue_time,
       execution_time,
       planning_time,
       trim(query_text) as query_text
FROM sys_query_history
WHERE query_id = 3093;
```

Beispielausgabe.

```
user_id | query_id | transaction_id | session_id |   status   | database_name |
   start_time         |          end_time          | result_cache_hit | elapsed_time |
 queue_time | execution_time | planning_time | query_text
--------+----------+----------------+------------+------------
+---------------+----------------------------+----------------------------
+-----------------+-------------+-----------+----------------+--------------
+-------------------------------------
    106 |     3093 |          11759 | 1073750146 | success    | dev           |
 2023-03-16 16:53:17.840214 | 2023-03-16 16:53:18.106588 | f               |
 266374 |          0 |         105725 |        136589 | select count(*) from item;
```

Die folgende Abfrage listet die 10 neuesten SELECT-Abfragen auf.

```
SELECT query_id,
       transaction_id,
```

```
       session_id,
       start_time,
       elapsed_time,
       queue_time,
       execution_time,
       returned_rows,
       returned_bytes
FROM sys_query_history
WHERE query_type = 'SELECT'
ORDER BY start_time DESC limit 10;
```

Beispielausgabe.

```
 query_id | transaction_id | session_id |         start_time         | elapsed_time |
 queue_time | execution_time | returned_rows | returned_bytes
----------+----------------+------------+----------------------------+--------------
+------------+----------------+---------------+----------------
   526532 |          61093 | 1073840313 | 2022-02-09 04:43:24.149603 |       520571 |
        0 |         481293 |              1 |           3794
   526520 |          60850 | 1073840313 | 2022-02-09 04:38:27.24875  |       635957 |
        0 |         596601 |              1 |           3679
   526508 |          60803 | 1073840313 | 2022-02-09 04:37:51.118835 |       563882 |
        0 |         503135 |              5 |          17216
   526505 |          60763 | 1073840313 | 2022-02-09 04:36:48.636224 |       649337 |
        0 |         589823 |              1 |            652
   526478 |          60730 | 1073840313 | 2022-02-09 04:36:11.741471 |     14611321 |
        0 |       14544058 |              0 |              0
   526467 |          60636 | 1073840313 | 2022-02-09 04:34:11.91463  |     16711367 |
        0 |       16633767 |              1 |            575
   511617 |         617946 | 1074009948 | 2022-01-20 06:21:54.44481  |      9937090 |
        0 |        9899271 |            100 |          12500
   511603 |         617941 | 1074259415 | 2022-01-20 06:21:45.71744  |      8065081 |
        0 |        7582500 |            100 |           8889
   511595 |         617935 | 1074128320 | 2022-01-20 06:21:44.030876 |      1051270 |
        0 |        1014879 |              1 |             72
   511584 |         617931 | 1074030019 | 2022-01-20 06:21:42.764088 |       609033 |
        0 |         485887 |            100 |           8438
```

Die folgende Abfrage zeigt die tägliche Anzahl ausgewählter Abfragen und die durchschnittlich verstrichene Abfragezeit.

```
SELECT date_trunc('day',start_time) AS exec_day,
```

```
       status,
       COUNT(*) AS query_cnt,
       AVG(datediff (microsecond,start_time,end_time)) AS elapsed_avg
FROM sys_query_history
WHERE query_type = 'SELECT'
AND start_time >= '2022-01-14'
AND start_time <= '2022-01-18'
GROUP BY exec_day,
         status
ORDER BY exec_day,
         status;
```

Beispielausgabe.

```
      exec_day       | status  | query_cnt | elapsed_avg
---------------------+---------+-----------+------------
 2022-01-14 00:00:00 | success |      5253 |  56608048
 2022-01-15 00:00:00 | success |      7004 |  56995017
 2022-01-16 00:00:00 | success |      5253 |  57016363
 2022-01-17 00:00:00 | success |      5309 |  55236784
 2022-01-18 00:00:00 | success |      8092 |  54355124
```

Die folgende Abfrage zeigt die Leistung der täglich verstrichenen Abfragezeit.

```
SELECT distinct date_trunc('day',start_time) AS exec_day,
       query_count.cnt AS query_count,
       Percentile_cont(0.5) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P50_runtime,
       Percentile_cont(0.8) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P80_runtime,
       Percentile_cont(0.9) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P90_runtime,
       Percentile_cont(0.99) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P99_runtime,
       Percentile_cont(1.0) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS max_runtime
FROM sys_query_history
LEFT JOIN (SELECT  date_trunc('day',start_time) AS day, count(*) cnt
           FROM sys_query_history
           WHERE query_type = 'SELECT'
           GROUP by 1) query_count
ON date_trunc('day',start_time) = query_count.day
WHERE query_type = 'SELECT'
```

```
ORDER BY exec_day;
```

Beispielausgabe.

```
      exec_day       | query_count | p50_runtime | p80_runtime | p90_runtime |
 p99_runtime  | max_runtime
--------------------+-------------+-------------+-------------+-------------
+--------------+-------------
 2022-01-14 00:00:00 |        5253 |  16816922.0 |  69525096.0 | 158524917.8 |
 486322477.52 | 1582078873.0
 2022-01-15 00:00:00 |        7004 |  15896130.5 |  71058707.0 | 164314568.9 |
 500331542.07 | 1696344792.0
 2022-01-16 00:00:00 |        5253 |  15750451.0 |  72037082.2 | 159513733.4 |
 480372059.24 | 1594793766.0
 2022-01-17 00:00:00 |        5309 |  15394513.0 |  68881393.2 | 160254700.0 |
 493372245.84 | 1521758640.0
 2022-01-18 00:00:00 |        8092 |  15575286.5 |  68485955.4 | 154559572.5 |
 463552685.39 | 1542783444.0
 2022-01-19 00:00:00 |        5860 |  16648747.0 |  72470482.6 | 166485138.2 |
 492038228.67 | 1693483241.0
 2022-01-20 00:00:00 |        1751 |  15422072.0 |  69686381.0 | 162315385.0 |
 497066615.00 | 1439319739.0
 2022-02-09 00:00:00 |          13 |   6382812.0 |  17616161.6 |  21197988.4 |
 23021343.84 |   23168439.0
```

Die folgende Abfrage zeigt die Abfragetypverteilung.

```
SELECT query_type,
       COUNT(*) AS query_count
FROM sys_query_history
GROUP BY query_type
ORDER BY query_count DESC;
```

Beispielausgabe.

```
 query_type | query_count
------------+-------------
 UTILITY    |      134486
 SELECT     |       38537
 DDL        |        4832
 OTHER      |         768
```

```
LOAD       |          768
CTAS       |          748
COMMAND    |           92
```

SYS_QUERY_TEXT

Verwenden Sie SYS_QUERY_TEXT, um den Abfragetext aller Abfragen anzuzeigen. Jede Zeile stellt den Abfragetext von Abfragen mit bis zu 4 000 Zeichen dar, beginnend mit der Sequenznummer 0. Wenn die Abfrageanweisung mehr als 4 000 Zeichen enthält, werden zusätzliche Zeilen für die Anweisung protokolliert, indem die Sequenznummer für jede Zeile erhöht wird. In dieser Ansicht werden alle Benutzerabfragetexte protokolliert, z. B. DDL-, Utility-, Amazon-Redshift-Abfragen und reine Führungsknotenabfragen.

SYS_QUERY_TEXT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Abfrage gesendet hat.
query_id	bigint	Die Abfrage-ID.
transaction_id	bigint	Die ID der mit der Anweisung verbundenen Transaktion.
session_id	Ganzzahl	Die Prozess-ID der Sitzung, die die Abfrage ausführt.
start_time	Zeitstempel	Der Zeitpunkt, an dem die Abfrage gestartet wird.
sequence	Ganzzahl	Wenn eine einzelne Anweisung mehr als 4 000 Zeichen enthält, werden weitere Zeilen für diese

Spaltenname	Datentyp	Beschreibung
		Anweisung protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite Zeile usw.
text	Zeichen (4000)	Der Text der SQL-Abfrage, der in Schritten von 4 000 Zeichen angegeben ist. Diese Feld kann Sonderzeichen wie Backslash (\) und Zeilenumb ruch (\n) enthalten.

Beispielabfragen

Die folgende Abfrage gibt laufende und in der Warteschlange stehende Abfragen zurück.

```
SELECT user_id,
 query_id,
 transaction_id,
 session_id, start_time,
 sequence, trim(text) as text from sys_query_text
 ORDER BY sequence;
```

Beispielausgabe.

```
 user_id | query_id | transaction_id | session_id |        start_time          |
 sequence |                                                        text
--------+----------+----------------+------------+----------------------------
+----------
+-----------------------------------------------------------------------------------------------------
   100  |     4    |       1396     | 1073750220 | 2023-04-28 16:44:55.887184 |
   0    | SELECT trim(text) as text, sequence FROM sys_query_text WHERE query_id =
 pg_last_query_id() AND user_id > 1 AND start
_time > '2023-04-28 16:44:55.922705+00:00'::timestamp order by sequence;
```

Die folgende Abfrage gibt die Berechtigungen zurück, die Gruppen in Ihrer Datenbank erteilt oder entzogen wurden.

```
SELECT
```

```
    SPLIT_PART(text, ' ', 1) as grantrevoke,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'GROUP'))), ' ', 2) as group,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), ' '))), 'ON', 1) as type,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'ON'))), ' ', 2) || ' ' ||
 SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'ON'))), ' ', 3) as entity
FROM SYS_QUERY_TEXT
WHERE (text LIKE 'GRANT%' OR text LIKE 'REVOKE%') AND text LIKE '%GROUP%';

+-------------+----------+--------+----------+
| grantrevoke |  group   |  type  |  entity  |
+-------------+----------+--------+----------+
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | USAGE  | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
+-------------+----------+--------+----------+
```

SYS_RESTORE_LOG

Verwenden Sie SYS_RESTORE_LOG, um den Migrationsfortschritt jeder Tabelle in dem Cluster während einer klassischen Größenänderung zu RA3-Knoten zu überwachen. Dies erfasst den historischen Durchsatz der Datenmigration während der Größenänderung. Weitere Informationen zur klassischen Größenänderung auf RA3-Knoten finden Sie unter Klassische Größenänderung.

SYS_RESTORE_LOG ist nur für Superuser sichtbar.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
event_time	Zeitstempel	Ein Zeitstempel, der angibt, wann der Protokolleintrag aufgezeichnet wurde.
database_name	char(128)	Name der Datenbank.
schema_name	char(128)	Der Name des Schemas.
table_name	char(128)	Der Name der Tabelle.

Spaltenname	Datentyp	Beschreibung
table_id	Ganzzahl	Die ID der Tabelle.
action	char(128)	Die Aktion, die zum Zeitpunkt des Eintrags ausgeführt wurde. Zu den Werten können gehören: Migration gestartet , Checkpoint, fortgesetzt, abgeschlossen, abgebrochen oder zurückgesetzt.
table_size	long	Die Größe der Tabelle.
total_data_processed	long	Der Umfang der Daten in MB, die bis zu diesem Zeitpunkt für die Tabelle verarbeitet wurden.
delta_data_processed	long	Umfang der seit dem letzten event_time-Update verarbeit eten Daten in MB. Damit können Sie feststellen, wie viele Daten seit dem letzten aufgezeichneten Zeitinter vall verarbeitet wurden. Sie können dies mit table_size vergleichen, um ein Gefühl dafür zu bekommen, wie schnell die Datenverarbeitung voranschreitet.
Nachricht	char(512)	Eine ausführliche Erklärung für den Wert in der Aktionsspalte.

Spaltenname	Datentyp	Beschreibung
redistribution_type	char(32)	Der Umverteilungstyp für die Tabelle. Entweder eine KEY-Konvertierung oder eine EVEN-Neugewichtung saufgabe. Weitere Informationen zu Verteilungsstilen finden Sie unter Verteilungsstile.

Beispielabfragen

Die folgende Abfrage berechnet den Durchsatz der Datenverarbeitung mithilfe von SYS_RESTORE_LOG.

```
SELECT
    ROUND(sum(delta_data_processed) / 1024.0, 2) as data_processed_gb,
    ROUND(datediff(sec, min(event_time), max(event_time)) / 3600.0, 2) as duration_hr,
    ROUND(data_processed_gb/duration_hr, 2) as throughput_gb_per_hr
from sys_restore_log;
```

Beispielausgabe.

```
 data_processed_gb | duration_hr | throughput_gb_per_hr
-------------------+-------------+----------------------
              0.91 |        8.37 |                 0.11
(1 row)
```

Die folgende Abfrage zeigt alle Umverteilungstypen.

```
SELECT * from sys_restore_log ORDER BY event_time;
```

```
 database_name |     schema_name      |      table_name      | table_id |
 action             | total_data_processed | delta_data_processed |         event_time
        | table_size | message |   redistribution_type
---------------+----------------------+----------------------+----------
+----------------------------+----------------------+----------------------
+----------------------------+------------+---------+--------------------------
```

```
 dev           | schemaaaa877096d844d | customer_key         |   106424 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:18:00.744977 |        325 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t2_autokey   |   106430 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:18:02.756675 |         90 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t2_autokey   |   106430 |
 Redistribution completed    |                   90 |                    90 | 2024-01-05
 02:23:30.643718 |         90 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | customer_key         |   106424 |
 Redistribution completed    |                  325 |                   325 | 2024-01-05
 02:23:45.998249 |        325 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t1_even      |   106428 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:23:46.083849 |         30 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t5_auto_even |   106436 |
 Redistribution started      |                    0 |                       | 2024-01-05
 02:23:46.855728 |         45 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t5_auto_even |   106436 |
 Redistribution completed    |                   45 |                    45 | 2024-01-05
 02:24:16.343029 |         45 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t1_even      |   106428 |
 Redistribution completed    |                   30 |                    30 | 2024-01-05
 02:24:20.584703 |         30 |         | Rebalance Disteven Table
 dev           | schemaefd028a2a48a4c | customer_even        |   130512 |
 Redistribution started      |                    0 |                       | 2024-01-05
 04:54:55.641741 |        190 |         | Restore Disteven Table
 dev           | schemaefd028a2a48a4c | customer_even        |   130512 |
 Redistribution checkpointed |     29.4342113157737 |     29.4342113157737 | 2024-01-05
 04:55:04.770696 |        190 |         | Restore Disteven Table
(8 rows)
```

SYS_RESTORE_STATE

Verwenden Sie SYS_RESTORE_STATE, um den Migrationsfortschritt jeder Tabelle während einer klassischen Größenänderung zu überwachen. Dies ist insbesondere dann der Fall, wenn der Zielknotentyp RA3 ist. Weitere Informationen zur klassischen Größenänderung auf RA3-Knoten finden Sie unter Klassische Größenänderung.

SYS_RESTORE_STATE ist nur für den Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der die Abfrage gesendet hat.
database_name	char(64)	Der Name der Datenbank der Tabelle.
schema_id	Ganzzahl	Die Schema-ID der Tabelle.
table_id	Ganzzahl	Die ID der Tabelle.
table_name	char(128)	Der Name der Tabelle.
redistribution_status	char(128)	Der Status des Weiterver teilungsfortschritts der Tabelle. Mögliche Werte sind `Completed` , `In progress` und `Pending`.
percentage_redistributed	float	Der Prozentanteil des Weiterverteilungsfortschritts der Tabelle. Zulässige Werte sind 0 bis 100. Ein Wert von 25 gibt beispielsweise an, dass 25 % der Daten neu verteilt werden.
redistribution_type	char(32)	Der Umverteilungstyp für die Tabelle. Entweder eine KEY-Konvertierung oder eine EVEN-Neugewichtung saufgabe. Weitere Informati onen zu Verteilungsstilen finden Sie unter Verteilun gsstile.

Beispielabfragen

Die folgende Abfrage gibt Datensätze für laufende Abfragen und Abfragen in der Warteschlange zurück.

```
SELECT * FROM sys_restore_state;
```

Beispielausgabe.

```
 userid | database_name | schema_id | table_id |   table_name   | redistribution_status
 | precentage_redistributed |   redistribution_type
--------+---------------+-----------+----------+----------------
+-----------------------+--------------------------+-------------------------
    1   |     test1     |   124865  |  124878  | customer_key_4 |         Pending
 |      0                   |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124874  | customer_key_3 |         Pending
 |      0                   |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124870  | customer_key_2 |        Completed
 |     100                  |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124866  | customer_key_1 |       In progress
 |     13.52                |  Restore Distkey Table
```

Im Folgenden finden Sie den Status der Datenverarbeitung.

```
SELECT
    redistribution_status, ROUND(SUM(block_count) / 1024.0, 2) AS total_size_gb
FROM sys_restore_state sys inner join stv_tbl_perm stv
    on sys.table_id = stv.id
GROUP BY sys.redistribution_status;
```

Beispielausgabe.

```
 redistribution_status | total_size_gb
-----------------------+---------------
 Completed             |          0.07
 Pending               |          0.71
 In progress           |          0.20
(3 rows)
```

SYS_SCHEMA_QUOTA_VIOLATIONS

Zeichnet das Auftreten, die Transaktions-ID und andere nützliche Informationen auf, wenn ein Schema-Kontingent überschritten wird. Diese Systemtabelle ist eine Übersetzung von STL_SCHEMA_QUOTA_VIOLATIONS.

r_SYS_SCHEMA_QUOTA_VIOLATIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
owner_id	Ganzzahl	Die ID des Schemabesitzers.
user_id	Ganzzahl	ID des Benutzers, der den Eintrag generiert hat.
transaction_id	bigint	Die mit der Anweisung verbundene Transaktions-ID.
session_id	Ganzzahl	Die mit der Anweisung verbundene Prozess-ID.
schema_id	integer	Die Namespace- oder Schema-ID.
schema_name	Zeichen (128)	Der Namespace- oder Schemaname.
quota	integer	Die Menge an Speicherplatz (in MB), die das Schema verwenden kann.
disk_usage	integer	Der Speicherplatz (in MB), der zurzeit vom Schema verwendet wird.
record_time	Timestamp ohne Zeitzone	Der Zeitpunkt, an dem die Verletzung aufgetreten ist.

Beispielabfragen

Die folgende Abfrage zeigt das Ergebnis einer Kontingentverletzung:

```
SELECT user_id, TRIM(schema_name) "schema_name", quota, disk_usage, record_time FROM
sys_schema_quota_violations WHERE SCHEMA_NAME = 'sales_schema' ORDER BY timestamp DESC;
```

Diese Abfrage gibt die folgende Beispielausgabe für das angegebene Schema zurück:

```
user_id| schema_name  | quota | disk_usage | record_time
-------+--------------+-------+------------+----------------------------
104    | sales_schema | 2048  | 2798       | 2020-04-20 20:09:25.494723
(1 row)
```

SYS_SERVERLESS_USAGE

Verwenden Sie SYS_SERVERLESS_USAGE, um Details zur Amazon-Redshift-Serverless-Auslastung von Ressourcen anzuzeigen. Diese Systemansicht gilt nicht für bereitgestellte Amazon-Redshift-Cluster.

Diese Ansicht enthält die Zusammenfassung der Serverless-Nutzung, mit einer Genauigkeit von 1 Minute, einschließlich wie viel Rechenkapazität für die Verarbeitung von Abfragen verwendet wird und wie viel des von Amazon Redshift verwalteten Speichers verwendet wird. Die Rechenkapazität wird für ausgeführte Workloads in Redshift Processing Units (RPUs) pro Sekunde, also RPU-Sekunden, gemessen. RPUs werden verwendet, um Abfragen von Daten zu verarbeiten, die in ein Data Warehouse geladen wurden, aus einem Amazon S3-Data Lake abgefragt oder über Betriebsdatenbanken per Verbundabfrage aufgerufen werden. Amazon Redshift Serverless speichert die Daten 7 Tage lang in SYS_SERVERLESS_USAGE.

Beispiele für die Abrechnung von Datenverarbeitungskosten finden Sie unter Fakturierung für Amazon Redshift Serverless.

SYS_SERVERLESS_USAGE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
start_time	timestamp	Der Zeitpunkt, zu dem das Intervall begann.
end_time	timestamp	Der Zeitpunkt, an dem das Intervall abgeschlossen wurde.
compute_seconds	double precision	Die kumulierten genutzten Recheneinheiten (in RPU-Sekunden) für dieses Zeitinter vall. Dieser Wert berücksic htigt die dem Konto zugewiese ne RPU-Basiskapazität.
compute_capacity	double precision	Die durchschnittliche Anzahl von Recheneinheiten (Redshift Processing Units, RPUs), die während dieses Zeitintervalls zugewiesen wurden. Der Wert compute_capacity kann dynamisch geändert werden.
data_storage	Ganzzahl	Der durchschnittliche Datenspeicherplatz in MB, der während dieses Zeitintervalls verwendet wird. Der verwendete Datenspei cher kann sich dynamisch ändern, wenn Daten aus der Datenbank geladen oder gelöscht werden.

Spaltenname	Datentyp	Beschreibung
cross_region_transferred_data	Ganzzahl	Die für regionsübergreifende Datenfreigabe übermittelten Daten werden während dieses Zeitintervalls in Byte übermittelt.
charged_seconds	Ganzzahl	Die kumulierten berechneten Recheneinheiten (in RPU-Sekunden) für dieses Zeitintervall. Dieser Wert wird nach dem Ende einer Transaktion berechnet und kann daher während der Ausführung einer Transaktion 0 lauten. Verwenden Sie charged_seconds, um die Kosten für eine Amazon-Redshift-Serverless-Arbeitsgruppe zu berechnen. Dieser Wert berücksichtigt die RPU-Kapazität, die der Amazon-Redshift-Serverless-Arbeitsgruppe zugewiesen wurde.

Nutzungshinweise

- Es gibt Situationen, in denen compute_seconds 0, charged_seconds jedoch größer als 0 ist oder umgekehrt. Dies ist ein normales Verhalten, das sich aus der Art und Weise ergibt, wie Daten in der Systemansicht aufgezeichnet werden. Für eine genauere Darstellung der Details zur Serverless-Nutzung empfehlen wir, die Daten zu aggregieren.

Beispiel

Führen Sie die folgende Abfrage aus, um die Gesamtkosten für die verbrauchten RPU-Stunden für ein Zeitintervall durch Abfrage von charged_seconds abzurufen:

```
select trunc(start_time) "Day",
(sum(charged_seconds)/3600::double precision) * <Price for 1 RPU> as cost_incurred
from sys_serverless_usage
group by 1
order by 1
```

Beachten Sie, dass es während des Intervalls Leerlaufzeit geben kann. Die Leerlaufzeit wird nicht zu den verbrauchten RPUs hinzugerechnet.

SYS_SESSION_HISTORY

Verwenden Sie SYS_SESSION_HISTORY, um Informationen über die aktuellen aktiven Sitzungen und den Sitzungsverlauf anzuzeigen.

SYS_SESSION_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	char(50)	Die Kennung des Benutzers, der den Eintrag generiert hat
session_id	Ganzzahl	Die ID der mit der Anweisung verknüpften Sitzung.
database_name	char(50)	Name der Datenbank.
Status	char	Der Status der Sitzung. Mögliche Werte sind `active`, `timed out` und `closed`.

Spaltenname	Datentyp	Beschreibung
session_timeout	Ganzzahl	Die maximale Zeit in Sekunden, die eine Sitzung ohne Timeout inaktiv oder untätig bleibt. 0 bedeutet, dass kein Timeout eingestellt ist.
start_time	Zeitstempel	Der Zeitpunkt, an dem die Verbindung hergestellt wurde.
end_time	Zeitstempel	Der Zeitpunkt, an dem, mit dem die Verbindung beendet wurde.

Beispiel

Im Folgenden finden Sie ein Beispiel für eine Ausgabe von SYS_SESSION_HISTORY.

```
select * from sys_session_history;
 user_id | session_id |   database_name  | status | session_timeout |
 start_time         |          end_time
---------+------------+------------------+--------+-----------------
+----------------------------+----------------------------
       1 | 1073971370 | dev              | closed |        0        | 2023-07-17
 15:50:12.030104 | 2023-07-17 15:50:12.123218
       1 | 1073979694 | dev              | closed |        0        | 2023-07-17
 15:50:24.117947 | 2023-07-17 15:50:24.131859
       1 | 1073873049 | dev              | closed |        0        | 2023-07-17
 15:49:29.067398 | 2023-07-17 15:49:29.070294
       1 | 1073873086 | database18127a4a | closed |        0        | 2023-07-17
 15:49:29.119018 | 2023-07-17 15:49:29.125925
       1 | 1073832112 | dev              | closed |        0        | 2023-07-17
 15:49:29.164688 | 2023-07-17 15:49:29.179631
       1 | 1073987697 | dev              | closed |        0        | 2023-07-17
 15:49:29.26749  | 2023-07-17 15:49:29.273034
       1 | 1073922429 | dev              | closed |        0        | 2023-07-17
 15:49:33.35315  | 2023-07-17 15:49:33.367499
       1 | 1073766783 | dev              | closed |        0        | 2023-07-17
 15:49:45.38237  | 2023-07-17 15:49:45.396902
       1 | 1073807506 | dev              | active |        0        | 2023-07-17
 15:51:48        |
```

SYS_SPATIAL_SIMPLIFY

Sie können die Systemansicht SYS_SPATIAL_SIMPLIFY mit dem Befehl COPY abfragen, um Informationen zu vereinfachten räumlichen Geometrieobjekten zu erhalten. Wenn Sie COPY für ein Shapefile verwenden, können Sie zur Erfassung die Optionen SIMPLIFY `tolerance`, SIMPLIFY AUTO und SIMPLIFY AUTO `max_tolerance` verwenden. Das Ergebnis der Vereinfachung wird in der Systemansicht SYS_SPATIAL_SIMPLIFY zusammengefasst.

Wenn SIMPLIFY AUTO `max_tolerance` festgelegt ist, enthält diese Ansicht eine Zeile für jede Geometrie, die die maximale Größe überschritten hat. Wenn SIMPLIFY `tolerance` festgelegt ist, wird eine Zeile für den gesamten COPY-Vorgang gespeichert. Diese Zeile verweist auf die Abfrage-ID von COPY und die angegebene Vereinfachungstoleranz.

Weitere Informationen zum Laden einer Shapefile finden Sie unter Laden eines Shapefile in Amazon Redshift.

SYS_SPATIAL_SIMPLIFY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query_id	bigint	Die ID der Abfrage (Befehl COPY), die diese Zeile generiert hat.
line_number	bigint	Wenn COPY `SIMPLIFY AUTO` angegeben ist, ist dieser Wert die Datensatznummer des vereinfachten Datensatzes im Shapefile.
maximum_tolerance	double precision	Der im Befehl COPY angegebene Abstandstoleranzwert. Dies ist entweder der maximale Toleranzwert mit der Option `SIMPLIFY AUTO` oder der feste Toleranzwert mit der Option `SIMPLIFY`.
initial_size	bigint	Die Größe des `GEOMETRY`-Datenwerts in Byte vor der Vereinfachung.

Spaltenname	Datentyp	Beschreibung
simplified	char(1)	Wenn die Option COPY `SIMPLIFY AUTO` angegeben ist, `t` wenn die Geometrie erfolgreich vereinfacht wurde, ansonsten `f`. Die Geometrie wird möglicherweise nicht erfolgreich vereinfacht, wenn nach der Vereinfachung mit der angegebenen maximalen Toleranz ihre Größe immer noch die maximale Geometriegröße überschreitet.
final_size	bigint	Wenn die Option COPY `SIMPLIFY AUTO` ist, ist dies die Größe der Geometrie nach der Vereinfachung in Byte.
final_tolerance	double precision	Für die Vereinfachung gewählte endgültige Toleranz.

Beispielabfrage

Die folgende Abfrage gibt die Liste der Datensätze zurück, die COPY vereinfacht hat.

```
SELECT * FROM sys_spatial_simplify;


 query_id | line_number | maximum_tolerance | initial_size | simplified | final_size |
  final_tolerance
----------+-------------+-------------------+--------------+------------+------------
+----------------------
    20    |     1184704 |                -1 |      1513736 | t          |    1008808 |
  1.276386653895e-05
    20    |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

SYS_STREAM_SCAN_ERRORS

Zeichnet Fehler für Datensätze auf, die über die Streaming-Erfassung geladen wurden.

SYS_STREAM_SCAN_ERRORS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
external_schema_name	character (128)	Der Name des Schemas des Kinesis-Streams oder des Amazon-MSK-Themas. Die Groß- und Kleinschreibung wird berücksichtigt.
stream_name	character (255)	Der Name des Streams oder Themas. Die Groß- und Kleinschreibung wird berücksichtigt.
mv_name	character (128)	Der Name der zugeordneten materialisierten Ansicht. Leer, wenn keine vorhanden. Die Groß- und Kleinschreibung wird berücksichtigt.
transaction_id	bigint	Die Transaktions-ID.
query_id	bigint	Die Abfrage-ID.
stream_timestamp_type	character (1)	Der Typ des Stream-Zeitstempels. Die Groß- und Kleinschreibung wird berücksichtigt.
stream_timestamp	Timestamp ohne Zeitzone	Der Zeitpunkt, zu dem der Datensatz angekommen ist.
record_time	Timestamp ohne Zeitzone	Der Zeitpunkt, zu dem die Fehlermeldung protokolliert wurde.
partition_id	character (128)	Die Partitions-/Shard-ID. Die Groß- und Kleinschreibung wird berücksichtigt.

Spaltenname	Datentyp	Beschreibung
position	character (128)	Die Position des Datensatzes. Dies entspricht der Sequenznummer in Kinesis oder dem Offset in Amazon MSK. Die Groß- und Kleinschreibung wird berücksichtigt.
error_code	integer	Der Fehlercode.
error_reason	character (128)	Die Fehlerursache. Die Groß- und Kleinschreibung wird berücksichtigt.

SYS_STREAM_SCAN_STATES

Zeichnet Scanstatus für Datensätze auf, die über die Streaming-Erfassung geladen wurden.

SYS_STREAM_SCAN_STATES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
external_schema_name	character (128)	Der Name des externen Schemas. Die Groß- und Kleinschreibung wird berücksichtigt.
stream_name	character (255)	Name des Streams. Die Groß- und Kleinschreibung wird berücksichtigt.
mv_name	character (128)	Der Name der zugeordneten materialisierten Ansicht. Leer, wenn keine vorhanden. Die Groß- und Kleinschreibung wird berücksichtigt.
transaction_id	bigint	Die Transaktions-ID.

Spaltenname	Datentyp	Beschreibung
query_id	bigint	Die Abfrage-ID.
record_time	Timestamp ohne Zeitzone	Der Zeitpunkt, zu dem die Daten protokolliert wurden.
partition_id	character (128)	Die Partitions- oder Shard-ID. Die Groß- und Kleinschreibung wird berücksichtigt.
latest_position	character (128)	Die Position des letzten Datensatzes, der in dem Stapel gelesen wurde. Dies entspricht der Sequenznummer in Kinesis oder dem Offset in Amazon MSK. Die Groß- und Kleinschreibung wird berücksichtigt.
scanned_rows	bigint	Die Anzahl der Datensätze, die in dem Stapel gescannt wurden.
skipped_rows	bigint	Die Anzahl der Datensätze, die in dem Stapel übersprungen wurden.
scanned_bytes	bigint	Die Anzahl der Bytes, die in dem Stapel gescannt wurden.
stream_record_time_min	Timestamp ohne Zeitzone	Kinesis- oder Amazon MSK-Ankunftszeit für den frühesten Datensatz im Stapel.
stream_record_time_max	Timestamp ohne Zeitzone	Kinesis- oder Amazon MSK-Ankunftszeit für den spätesten Datensatz im Stapel.

Die folgende Abfrage zeigt Stream- und Themendaten für bestimmte Abfragen.

```
select
 query_id,mv_name::varchar,external_schema_name::varchar,stream_name::varchar,sum(scanned_rows)
 total_records,
```

```
sum(scanned_bytes) total_bytes from sys_stream_scan_states where query in
 (5401180,8601939) group by 1,2,3,4;

  query_id  |    mv_name     | external_schema_name |   stream_name   | total_records |
  total_bytes
------------+----------------+----------------------+-----------------+---------------
+----------------
 5401180    | kinesistest    | kinesis              | kinesisstream   |    1493255696 |
 3209006490704
 8601939    | msktest        | msk                  | mskstream       |      14677023 |
   31056580668
(2 rows)
```

SYS_TRANSACTION_HISTORY

Verwenden Sie SYS_TRANSACTION_HISTORY, um Details einer Transaktion anzuzeigen, wenn Sie eine Abfrage verfolgen.

SYS_TRANSACTION_HISTORY ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	ID des Benutzers, der den Eintrag generiert hat.
transaction_id	bigint	Die ID der Transaktion.
isolation_level	text	Die Isolationsstufe der Transaktion. Mögliche Werte sind `Serializable` und `Snapshot Isolation` .
status	text	Der Status der Transaktion. Mögliche Status sind `committed` und `rolledback` .

Spaltenname	Datentyp	Beschreibung
transaction_start_time	Zeitstempel	Anfangszeit der Transaktion.
commit_start_time	Zeitstempel	Die Startzeit des Commit.
commit_end_time	Zeitstempel	Die Endzeit des Commit.
blocks_committed	bigint	Die Anzahl der Blöcke, die im Rahmen dieses Commit-Vorgangs geschrieben werden mussten.
undo_transaction_id	bigint	Die ID der Undo-Transaktion, wenn Transaktionen rückgängig gemacht wurden. Andernfalls ist dieser Wert -1.

Beispielabfragen

```
select * from sys_transaction_history order by transaction_start_time desc;

 user_id | transaction_id | isolation_level |   status   |   transaction_start_time
   |     commit_start_time      |      commit_end_time       | blocks_committed |
 undo_transaction_id
---------+----------------+-----------------+------------+--------------------------
+---------------------------+---------------------------+-----------------
+---------------------
     100 |           1310 | Serializable    | committed  | 2023-08-27 21:03:11.822205 |
 2023-08-28 21:03:11.825287 | 2023-08-28 21:03:11.854883 |               17 |
        -1
     101 |           1345 | Serializable    | committed  | 2023-08-27 21:03:12.000278 |
 2023-08-28 21:03:12.003736 | 2023-08-28 21:03:12.030061 |               17 |
        -1
     102 |           1367 | Serializable    | committed  | 2023-08-27 21:03:12.1532   |
 2023-08-28 21:03:12.156124 | 2023-08-28 21:03:12.186468 |               17 |
        -1
     100 |           1370 | Serializable    | committed  | 2023-08-27 21:03:12.199316 |
 2023-08-28 21:03:12.204854 | 2023-08-28 21:03:12.238186 |               24 |
        -1
```

```
    100 |           1408 | Serializable    | committed  | 2023-08-27 21:03:53.891107 |
2023-08-28 21:03:53.894825 | 2023-08-28 21:03:53.927465 |               17 |
       -1
    100 |           1409 | Serializable    | rolledback | 2023-08-27 21:03:53.936431 |
2000-01-01 00:00:00        | 2023-08-28 21:04:08.712532 |                0 |
     1409
    101 |           1415 | Serializable    | committed  | 2023-08-27 21:04:24.283188 |
2023-08-28 21:04:24.289196 | 2023-08-28 21:04:24.374318 |               25 |
       -1
    101 |           1416 | Serializable    | committed  | 2023-08-27 21:04:24.38818  |
2023-08-28 21:04:24.391688 | 2023-08-28 21:04:24.415135 |               17 |
       -1
    100 |           1417 | Serializable    | rolledback | 2023-08-27 21:04:24.424252 |
2000-01-01 00:00:00        | 2023-08-28 21:04:28.354826 |                0 |
     1417
    101 |           1418 | Serializable    | rolledback | 2023-08-27 21:04:24.425195 |
2000-01-01 00:00:00        | 2023-08-28 21:04:28.680355 |                0 |
     1418
    100 |           1420 | Serializable    | committed  | 2023-08-27 21:04:28.697607 |
2023-08-28 21:04:28.702374 | 2023-08-28 21:04:28.735541 |               23 |
       -1
    101 |           1421 | Serializable    | committed  | 2023-08-27 21:04:28.744854 |
2023-08-28 21:04:28.749344 | 2023-08-28 21:04:28.779029 |               23 |
       -1
    101 |           1423 | Serializable    | committed  | 2023-08-27 21:04:28.78942  |
2023-08-28 21:04:28.791556 | 2023-08-28 21:04:28.817485 |               16 |
       -1
    100 |           1430 | Serializable    | committed  | 2023-08-27 21:04:28.917788 |
2023-08-28 21:04:28.919993 | 2023-08-28 21:04:28.944812 |               16 |
       -1
    102 |           1494 | Serializable    | committed  | 2023-08-27 21:04:37.029058 |
2023-08-28 21:04:37.033137 | 2023-08-28 21:04:37.062001 |               16 |
       -1
```

SYS_UDF_LOG

Zeichnet systemdefinierte Fehlermeldungen und Warnungen auf, die während der Ausführung einer benutzerdefinierten Funktion (User-Defined Function, UDF) generiert wurden.

SYS_UDF_LOG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query_id	bigint	Die Abfrage-ID.
function_name	text	Der Name der benutzerdefinierten Funktion.
record_time	Zeitstempel	Der Zeitpunkt der Erstellung des Datensatzes.
sequence	Ganzzahl	Die Reihenfolge einer einzelnen Protokollnachricht.
Nachricht	text	Der Text der Protokollnachricht.

Beispielabfragen

Das folgende Beispiel zeigt, wie benutzerdefinierte Funktionen mit systemdefinierten Fehlern umgehen. Der erste Block zeigt die Definition für eine benutzerdefinierte Funktion, die die Umkehrung eines Arguments ausgibt. Wenn Sie die Funktion ausführen und 0 als Argument angeben, gibt die Funktion einen Fehler aus. Die letzte Anweisung gibt die in SYS_UDF_LOG protokollierte Fehlermeldung zurück.

```
-- Create a function to find the inverse of a number.
CREATE OR REPLACE FUNCTION f_udf_inv(a int)

RETURNS float

IMMUTABLE AS $$return 1/a

$$ LANGUAGE plpythonu;

-- Run the function with 0 to create an error.
Select f_udf_inv(0);

-- Query SYS_UDF_LOG to view the message.
Select query_id, record_time, message::varchar from sys_udf_log;
```

```
query_id    |    record_time              |                message
----------+----------------------------
+-----------------------------------------------------
2211        | 2023-08-23 15:53:11.360538 |  ZeroDivisionError: integer division or
 modulo by zero line 2, in f_udf_inv\n return 1/a\n
```

Das folgende Beispiel fügt der benutzerdefinierten Funktion Protokollierung und eine Warnmeldung hinzu, so dass eine Division durch Null zu einer Warnmeldung und nicht zum Anhalten mit einer Fehlermeldung führt.

```
-- Create a function to find the inverse of a number and log a warning if you input 0.
CREATE OR REPLACE FUNCTION f_udf_inv_log(a int)
  RETURNS float IMMUTABLE
 AS $$
  import logging
  logger = logging.getLogger() #get root logger
  if a==0:
    logger.warning('You attempted to divide by zero.\nReturning zero instead of error.
\n')
    return 0
  else:
     return 1/a
$$ LANGUAGE plpythonu;

-- Run the function with 0 to trigger the warning.
Select f_udf_inv_log(0);

-- Query SYS_UDF_LOG to view the message.
Select query_id, record_time, message::varchar from sys_udf_log;

 query_id |        record_time         |                                  message
----------+----------------------------
+-----------------------------------------------------------------------------
     0   | 2023-08-23 16:10:48.833503 | WARNING: You attempted to divide by zero.
\nReturning zero instead of error.\n
```

SYS_UNLOAD_DETAIL

Verwenden Sie SYS_UNLOAD_DETAIL, um die Details zu einem UNLOAD-Vorgang anzuzeigen. Für jede von einer UNLOAD-Anweisung erstellte Datei wird eine Zeile aufgezeichnet. Beispiel: Wenn ein UNLOAD-Vorgang 12 Dateien erstellt, enthält SYS_UNLOAD_DETAIL 12 entsprechende Zeilen.

SYS_UNLOAD_DETAIL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die Kennung des Benutzers, der den Eintrag generiert hat
query_id	Ganzzahl	Die Abfrage-ID des UNLOAD-Befehls.
session_id	Ganzzahl	Die ID des mit der Abfragean weisung verbundenen Prozesses
transaction_id	bigint	Die ID der mit der Abfragean weisung verbundenen Transaktion
file_name	Zeichen (1280)	Der vollständige Amazon-S3-Objektpfad für die Datei.
start_time	Zeitstempel	Der Zeitpunkt des Transakti onsbeginns
end_time	Zeitstempel	Der Zeitpunkt des Transakti onsabschlusses
line_count	bigint	Die Anzahl der in die Datei entladenen Zeilen.
transfer_size	bigint	Die Anzahl der übertragenen Bytes.
file_format	Zeichen (10)	Das Dateiformat der entladene n Dateien

Beispielabfragen

Die folgende Abfrage zeigt die Details der entladenen Abfrage, einschließlich Format, Zeilen und Dateianzahl des unload-Befehls.

```
select query_id, substring(file_name, 0, 50), transfer_size, file_format from
 sys_unload_detail;
```

Beispielausgabe.

```
 query_id |                    substring                     | transfer_size |
 file_format
----------+--------------------------------------------------+---------------
+-------------
     9272 | s3://my-bucket/my_unload_doc_venue0000_part_00.gz |        395886 | Text

     9272 | s3://my-bucket/my_unload_doc_venue0001_part_00.gz |        406444 | Text

     9272 | s3://my-bucket/my_unload_doc_venue0002_part_00.gz |        409431 | Text

     9272 | s3://my-bucket/my_unload_doc_venue0003_part_00.gz |        403051 | Text

     9272 | s3://my-bucket/my_unload_doc_venue0004_part_00.gz |        413592 | Text

     9272 | s3://my-bucket/my_unload_doc_venue0005_part_00.gz |        395689 | Text

(6 rows)
```

SYS_UNLOAD_HISTORY

Verwenden Sie UNLOAD_HISTORY, um Details zu UNLOAD-Befehlen anzuzeigen. Jede Zeile stellt einen UNLOAD-Befehl mit akkumulierten Statistiken für einige der Felder dar. Sie enthält sowohl ausgeführte als auch abgeschlossene UNLOAD-Befehle.

SYS_UNLOAD_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der den Entladevorgang gestartet hat.
query_id	bigint	Die Abfrage-ID des UNLOAD-Befehls.
transaction_id	bigint	Die Transaktions-ID.
session_id	integer	Die ID des Prozesses, der den Entladevorgang ausführt.
database_name	text	Der Name der Datenbank , mit der der Benutzer verbunden war, als der Vorgang ausgegeben wurde.
status	text	Der Status des UNLOAD-Befehls. Gültige Werte sind u. a.: `running`, `completed` , `aborted` und `unknown`.
start_time	timestamp	Der Zeitpunkt, zu dem der Entladevorgang begann.
end_time	timestamp	Der Zeitpunkt, an dem der Entladevorgang abgeschlossen wurde.
duration	bigint	Die Dauer (in Mikrosekunden) des UNLOAD-Befehls.
file_format	text	Gibt das Dateiformat der Ausgabedateien an.
compression_type	text	Der Komprimierungstyp.

Spaltenname	Datentyp	Beschreibung
unloaded_location	text	Der Amazon-S3-Speicherort von entladenen Dateien.
unloaded_rows	bigint	Die Anzahl der Zeilen.
unloaded_files_count	bigint	Die Anzahl der Dateien in der Ausgabedatei.
unloaded_files_size	bigint	Die Dateigröße der Ausgabedatei.
error_message	text	Die Fehlermeldung des UNLOAD-Befehls.

Beispielabfragen

Die folgende Abfrage zeigt die Details der entladenen Abfrage, einschließlich Format, Zeilen und Dateianzahl des unload-Befehls.

```
SELECT query_id,
       file_format,
       start_time,
       duration,
       unloaded_rows,
       unloaded_files_count
FROM sys_unload_history
ORDER BY query_id,
file_format limit 100;
```

Beispielausgabe.

```
 query_id | file_format |         start_time         | duration | unloaded_rows |
 unloaded_files_count
----------+-------------+----------------------------+----------+---------------
+----------------------
   527067 | Text        | 2022-02-09 05:18:35.844452 |  5932478 |            10 |
             1
```

SYS_USERLOG

Zeichnet die Details der folgenden Änderungen an einem Datenbankbenutzer auf:

- Benutzer erstellen
- Benutzer entfernen
- Benutzer ändern (umbenennen)
- Benutzer ändern (Eigenschaften ändern)

Sie können diese Ansicht abfragen, um Informationen zu Serverless-Arbeitsgruppen und bereitgestellten Clustern einzusehen.

SYS_USERLOG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	integer	Die ID des Benutzers, der den Entladevorgang gestartet hat.
user_name	character(50)	Benutzername des von der Änderung betroffenen Benutzers.
original_user_name	character(50)	Der ursprüngliche Benutzername bei einer Umbenennungsaktion. Dieses Feld ist bei anderen Aktionen leer.
Aktion	character(10)	Die Aktion, die erfolgt ist. Gültige Werte sind ändern, erstellen, löschen und umbenennen.
has_create_db_privs	Ganzzahl	„True“ (der Wert 1) zeigt an, dass der Benutzer über

Spaltenname	Datentyp	Beschreibung
		Berechtigungen zum Erstellen von Datenbanken verfügt.
is_superuser	Ganzzahl	Wenn „True“ (der Wert 1) gilt, kann der Benutzer Systemkat aloge aktualisieren.
has_update_catalog_privs	Ganzzahl	Wenn „True“ (der Wert 1) gilt, kann der Benutzer Systemkat aloge aktualisieren.
password_expiration	Zeitstempel	Das Ablaufdatum des Passworts.
session_id	Ganzzahl	Die Prozess-ID.
transaction_id	bigint	Die Transaktions-ID.
record_time	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC.

Beispielabfragen

Das folgende Beispiel führt vier Benutzeraktionen aus und fragt dann die SYS_USERLOG-Ansicht ab.

```
CREATE USER userlog1 password 'Userlog1';
ALTER USER userlog1 createdb createuser;
ALTER USER userlog1 rename  to userlog2;
DROP user userlog2;

SELECT user_id, user_name, original_user_name, action, has_create_db_privs,
 is_superuser from SYS_USERLOG order by record_time desc;
```

```
user_id |  user_name | original_user_name |  action | has_create_db_privs |
 is_superuser
--------+------------+--------------------+---------+---------------------+------------
```

```
   108 |  userlog2  |                 | drop    |                  1 |   1
   108 |  userlog2  |     userlog1    | rename  |                  1 |   1
   108 |  userlog1  |                 | alter   |                  1 |   1
   108 |  userlog1  |                 | create  |                  0 |   0
(4 rows)
```

SYS_VACUUM_HISTORY

Verwenden Sie SYS_VACUUM_HISTORY, um Details zu Benutzerabfragen anzuzeigen. Weitere Informationen zum VACUUM-Befehl finden Sie unter VACUUM.

SYS_VACUUM_HISTORY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
user_id	Ganzzahl	Die ID des Benutzers, der die Abfrage initiiert hat.
transaction_id	long	Die Transaktions-ID der VACUUM-Anweisung.
query_id	long	Die Abfragekennung für die VACUUM-Anweisung. Sie können diese Tabelle mit der Ansicht SYS_QUERY_DETAIL verbinden, um die einzelnen SQL-Anweisungen zu sehen, die für eine bestimmte VACUUM-Transaktion ausgeführt wurden. Wenn Sie die gesamte Datenbank bereinigen, wird jede Tabelle in einer separaten Transakti on bereinigt. Für automatis

Spaltenname	Datentyp	Beschreibung
		ierte VACUUM-Operationen ist dieser Wert null.
database_name	text	Name der Datenbank.
schema_name	text	Der Name des Schemas.
table_name	Text	Der Name der Tabelle.
table_id	Ganzzahl	Die ID der Tabelle.
vacuum_type	character	Der Typ der VACUUM-Operation. Die möglichen Werte lauten wie folgt: • **`Delete`** • **`Sort`** • **`Reindex`** • **`Recluster`** • **`Full`** Weitere Informationen zu VACCUM-Typen finden Sie unter VACUUM.
is_automatic	boolesch	`true`, wenn es sich bei dem Vorgang um eine automatische Bereinigung handelt. Andernfalls `false`.

Spaltenname	Datentyp	Beschreibung
Status	character	Beschreibung der zurzeit ausgeführten Aktivität im Rahmen der Bereinigu ngsoperation: • Initialisieren • Sortierung • Merge • Löschen • Select • Fehlgeschlagen • Complete • Übersprungen • Aufbau der INTERLEAVED SORTKEY-Reihenfolge
start_time	Zeitstempel	Die Uhrzeit, zu welcher der Bereinigungsvorgang begann.
end_time	Zeitstempel	Die Uhrzeit, zu welcher der Bereinigungsvorgang endete. Wenn der Vorgang noch nicht abgeschlossen wurde, ist dieses Feld leer.
record_time	Zeitstempel	Die Uhrzeit, zu welcher der Bereinigungsvorgang in SYS_VACUUM_HISTORY aufgezeichnet wurde.

Spaltenname	Datentyp	Beschreibung
duration	Ganzzahl	Die Anzahl der Mikrosekunden zwischen dem Beginn und dem Ende des Bereinigungsvorgangs. Wenn der Bereinigungsvorgang noch nicht abgeschlossen wurde, ist dieses Feld leer.
rows_before_vacuum	bigint	Die tatsächliche Anzahl der Zeilen in der Tabelle zzgl. aller gelöschten Zeilen, die immer noch auf der Festplatte gespeichert sind (und auf die Bereinigung warten).
size_before_vacuum	Ganzzahl	Die Größe der Tabelle (in MB) vor Beginn des Bereinigungsvorgangs.
reclaimable_rows	bigint	Die Anzahl der Zeilen, die der Bereinigungsvorgang schätzungsweise zurückgewinnen wird, bevor er beginnt.
reclaimed_rows	bigint	Die Anzahl der Zeilen, die durch den Bereinigungsvorgang zurückgewonnen wurden.
reclaimed_blocks	bigint	Die Anzahl der Blöcke, die durch den Bereinigungsvorgang zurückgewonnen wurden.
sortedrows_before_vacuum	Ganzzahl	Die Anzahl der sortierten Zeilen in der Tabelle, bevor der Bereinigungsvorgang gestartet wurde.

Spaltenname	Datentyp	Beschreibung
sortedrows_after_vacuum	Ganzzahl	Die zusätzliche Anzahl sortierter Zeilen in der Tabelle nach Abschluss des Bereinigungsvorgangs. Dabei werden die Zeilen, die in `sortedrows_before_vacuum` gezählt wurden, nicht berücksichtigt.

Systemansichtszuordnung für die Migration zu SYS-Überwachungsansichten

Wenn Sie Ihren von Amazon Redshift bereitgestellten Cluster zu Amazon Redshift Serverless migrieren, verweisen Ihre Überwachungs- oder Diagnoseabfragen möglicherweise auf Systemansichten, die nur auf bereitgestellten Clustern verfügbar sind. Sie können Ihre Abfragen aktualisieren, um die SYS-Überwachungsansichten zu benutzen. Diese Seite enthält nur bereitgestellte Zuordnungen zu SYS-Ansichten, auf die Sie bei der Aktualisierung Ihrer Abfragen zurückgreifen können.

Themen

- SYS_QUERY_HISTORY
- SYS_QUERY_DETAIL
- SYS_RESTORE_LOG
- SYS_RESTORE_STATE
- SYS_TRANSACTION_HISTORY
- SYS_QUERY_TEXT
- SYS_CONNECTION_LOG
- SYS_SESSION_HISTORY
- SYS_LOAD_DETAIL
- SYS_LOAD_HISTORY
- SYS_LOAD_ERROR_DETAIL
- SYS_UNLOAD_HISTORY

- SYS_UNLOAD_DETAIL
- SYS_COPY_REPLACEMENTS
- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_DATASHARE_CROSS_REGION_USAGE
- SYS_DATESHARE_CHANGE_LOG
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_EXTERNAL_QUERY_ERROR
- SYS_VACUUM_HISTORY
- SYS_ANALYZE_HISTORY
- SYS_ANALYZE_COMPRESSION_HISTORY
- SYS_MV_REFRESH_HISTORY
- SYS_MV_STATE
- SYS_PROCEDURE_CALL
- SYS_PROCEDURE_MESSAGES
- SYS_UDF_LOG
- SYS_USERLOG
- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SPATIAL_SIMPLIFY

SYS_QUERY_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_QUERY_HISTORY definiert.

- STL_DDLTEXT
- STL_ERROR
- STL_QUERY
- STL_UTILITYTEXT
- STL_WLM_QUERY
- STV_INFLIGHT
- STV_RECENTS

- STV_WLM_QUERY_STATE
- SVL_COMPILE
- SVL_MULTI_STATEMENT_VIOLATIONS
- SVL_QLOG
- SVL_QUERY_QUEUE_INFO
- SVL_STATEMENTTEXT
- SVL_TERMINATE

SYS_QUERY_DETAIL

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_QUERY_DETAIL definiert.

- STL_AGGR
- STL_ALERT_EVENT_LOG
- STL_BCAST
- STL_DELETE
- STL_DIST
- STL_EXPLAIN
- STL_HASH
- STL_HASHJOIN
- STL_INSERT
- STL_LIMIT
- STL_MERGE
- STL_MERGEJOIN
- STL_NESTLOOP
- STL_PARSE
- STL_PLAN_INFO
- STL_PROJECT
- STL_QUERY_METRICS
- STL_RETURN
- STL_SAVE

- STL_SCAN
- STL_SORT
- STL_STREAM_SEGS
- STL_UNIQUE
- STL_WINDOW
- STV_EXEC_STATE
- STV_QUERY_METRICS
- SVCS_QUERY_SUMMARY
- SVL_QUERY_METRICS
- SVL_QUERY_METRICS_SUMMARY
- SVL_QUERY_REPORT
- SVL_QUERY_SUMMARY
- SVV_QUERY_STATE

SYS_RESTORE_LOG

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_RESTORE_LOG definiert.

- SVL_RESTORE_ALTER_TABLE_PROGRESS

SYS_RESTORE_STATE

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_RESTORE_STATE definiert.

- STV_XRESTORE_ALTER_QUEUE_STATE

SYS_TRANSACTION_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_TRANSACTION_HISTORY definiert.

- STL_COMMIT_STATS
- STL_TR_CONFLICT
- STL_UNDONE

SYS_QUERY_TEXT

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_QUERY_TEXT definiert.

- STL_QUERYTEXT

SYS_CONNECTION_LOG

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_CONNECTION_LOG definiert.

- STL_CONNECTION_LOG

SYS_SESSION_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_SESSION_HISTORY definiert.

- STL_SESSIONS
- STL_RESTARTED_SESSIONS
- STV_SESSIONS

SYS_LOAD_DETAIL

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_LOAD_DETAIL definiert.

- STL_LOAD_COMMITS

SYS_LOAD_HISTORY

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_LOAD_HISTORY definiert.

- STL_LOAD_COMMITS

SYS_LOAD_ERROR_DETAIL

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_LOAD_ERROR_DETAIL definiert.

- STL_LOADERROR_DETAIL
- STL_LOAD_ERRORS

SYS_UNLOAD_HISTORY

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_UNLOAD_HISTORY definiert.

- STL_UNLOAD_LOG

SYS_UNLOAD_DETAIL

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_UNLOAD_DETAIL definiert.

- STL_UNLOAD_LOG

SYS_COPY_REPLACEMENTS

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_COPY_REPLACEMENTS definiert.

- STL_REPLACEMENTS

SYS_DATASHARE_USAGE_CONSUMER

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_DATASHARE_USAGE_CONSUMER definiert.

- SVL_DATASHARE_USAGE_CONSUMER

SYS_DATASHARE_USAGE_PRODUCER

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_DATASHARE_USAGE_PRODUCER definiert.

- SVL_DATASHARE_USAGE_PRODUCER

SYS_DATASHARE_CROSS_REGION_USAGE

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_DATASHARE_CROSS_REGION_USAGE definiert.

- SVL_DATASHARE_CROSS_REGION_USAGE

SYS_DATESHARE_CHANGE_LOG

Einige oder alle Spalten in der folgenden Tabelle sind auch in SYS_DATESHARE_CHANGE_LOG definiert.

- SVL_DATASHARE_CHANGE_LOG

SYS_EXTERNAL_QUERY_DETAIL

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_EXTERNAL_QUERY_DETAIL definiert.

- SVL_FEDERATED_QUERY
- SVL_S3LIST
- SVL_S3QUERY
- SVL_S3QUERY_SUMMARY

SYS_EXTERNAL_QUERY_ERROR

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_EXTERNAL_QUERY_ERROR definiert.

- SVL_SPECTRUM_SCAN_ERROR

SYS_VACUUM_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_VACUUM_HISTORY definiert.

- STL_VACUUM

- SVL_VACUUM_PERCENTAGE
- SVV_VACUUM_PROGRESS
- SVV_VACUUM_SUMMARY

SYS_ANALYZE_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_ANALYZE_HISTORY definiert.

- STL_ANALYZE

SYS_ANALYZE_COMPRESSION_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_ANALYZE_COMPRESSION_HISTORY definiert.

- STL_ANALYZE_COMPRESSION

SYS_MV_REFRESH_HISTORY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_MV_REFRESH_HISTORY definiert.

- SVL_MV_REFRESH_STATUS

SYS_MV_STATE

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_MV_STATE definiert.

- STL_MV_STATE

SYS_PROCEDURE_CALL

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_PROCEDURE_CALL definiert.

- SVL_STORED_PROC_CALL

SYS_PROCEDURE_MESSAGES

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_PROCEDURE_MESSAGES definiert.

- SVL_STORED_PROC_MESSAGES

SYS_UDF_LOG

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_UDF_LOG definiert.

- SVL_UDF_LOG

SYS_USERLOG

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_USERLOG definiert.

- STL_USERLOG

SYS_SCHEMA_QUOTA_VIOLATIONS

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_SCHEMA_QUOTA_VIOLATIONS definiert.

- STL_SCHEMA_QUOTA_VIOLATIONS

SYS_SPATIAL_SIMPLIFY

Einige oder alle Spalten in den folgenden Tabellen sind auch in SYS_SPATIAL_SIMPLIFY definiert.

- SVL_SPATIAL_SIMPLIFY

Systemüberwachung (nur bereitgestellt)

Die folgenden Systemtabellen und Ansichten können auf bereitgestellten Clustern abgefragt werden. STL- und STV-Tabellen und -Ansichten enthalten eine Teilmenge der Daten aus verschiedenen

Systemtabellen. Diese ermöglichen den schnelleren und einfacheren Zugriff auf häufig abgefragte Daten in diesen Tabellen.

SVCS-Ansichten enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. SVL-Ansichten enthalten nur Informationen für Abfragen, die auf dem Haupt-Cluster ausgeführt werden, mit Ausnahme von SVL_STATEMENTTEXT. SVL_STATEMENTTEXT kann Informationen für Abfragen enthalten, die auf Nebenläufigkeitsskalierungs-Clustern sowie auf dem Haupt-Cluster ausgeführt werden.

Themen

STL-Ansichten für die Protokollierung

STL-Systemansichten werden aus Amazon-Redshift-Protokolldateien erstellt, um einen Systemverlauf zu präsentieren.

Diese Dateien befinden sich auf jedem Knoten in dem Data Warehouse-Cluster. Die STL-Ansichten beziehen die Informationen aus den Protokollen und formatieren sie zu verwendbaren Ansichten für Systemadministratoren.

Aufbewahrung von Protokollen – STL-Systemansichten speichern den Protokollverlauf von sieben Tagen. Die Aufbewahrung von Protokollen ist für alle Clustergrößen und Knotentypen garantiert und wird nicht durch Änderungen der Cluster-Workload beeinflusst. Die Aufbewahrung von Protokollen wird auch nicht vom Cluster-Status beeinflusst, z. B. wenn der Cluster angehalten wird. Der Protokollverlauf umfasst nur dann weniger als sieben Tage, wenn der Cluster neu ist. Sie müssen keine Maßnahmen ergreifen, um Protokolle aufzubewahren, aber Sie müssen regelmäßig Protokolldaten in andere Tabellen kopieren oder sie nach Amazon S3 entladen, um Protokolldaten zu speichern, die älter als 7 Tage sind.

Themen

- STL_ANALYZE_COMPRESSION
- STL_BCAST
- STL_COMMIT_STATS
- STL_CONNECTION_LOG
- STL_DDLTEXT
- STL_DELETE
- STL_DISK_FULL_DIAG
- STL_DIST
- STL_ERROR
- STL_EXPLAIN
- STL_FILE_SCAN
- STL_HASH
- STL_HASHJOIN
- STL_INSERT
- STL_LIMIT
- STL_LOAD_COMMITS
- STL_LOAD_ERRORS
- STL_LOADERROR_DETAIL
- STL_MERGE
- STL_MERGEJOIN
- STL_MV_STATE
- STL_NESTLOOP
- STL_PARSE
- STL_PLAN_INFO
- STL_PROJECT
- STL_QUERY
- STL_QUERY_METRICS
- STL_QUERYTEXT
- STL_REPLACEMENTS
- STL_RESTARTED_SESSIONS

- STL_RETURN
- STL_S3CLIENT
- STL_S3CLIENT_ERROR
- STL_SAVE
- STL_SCAN
- STL_SCHEMA_QUOTA_VIOLATIONS
- STL_SESSIONS
- STL_SORT
- STL_SSHCLIENT_ERROR
- STL_STREAM_SEGS
- STL_TR_CONFLICT
- STL_UNDONE
- STL_UNIQUE
- STL_UNLOAD_LOG
- STL_USAGE_CONTROL
- STL_USERLOG
- STL_UTILITYTEXT
- STL_VACUUM
- STL_WINDOW
- STL_WLM_ERROR
- STL_WLM_RULE_ACTION
- STL_WLM_QUERY

STL_AGGR

Analysiert aggregierte Ausführungsschritte für Abfragen. Diese Schritte treten bei der Ausführung aggregierter Funktionen und von GROUP BY-Klauseln auf.

STL_AGGR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_AGGR enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkomma stellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6

Spaltenname	Datentyp	Beschreibung
		Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
slots	integer	Anzahl der Hash-Buckets.
occupied	integer	Anzahl der Slots, die Datensätze enthalten.
maxlength	integer	Größe des größten Slots.
tbl	integer	Tabellen-ID.
is_diskbased	character(1)	Bei „true (t)“ wurde die Abfrage als festplattenbasierte Operation ausgeführt. Bei „false (f)“ wurde die Abfrage im Arbeitsspeicher ausgeführt.
workmem	bigint	Größe des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.
type	character(6)	Die Art des Schritts. Folgende Werte sind zulässig: • HASHED. Zeigt an, dass der Schritt die gruppierte, nicht sortierte Aggregierung verwendet hat. • PLAIN. Zeigt an, dass der Schritt die nicht gruppierte, skalare Aggregierung verwendet hat. • SORTED. Zeigt an, dass der Schritt die gruppierte, sortierte Aggregierung verwendet hat.
resizes	integer	Diese Information ist nur für die interne Verwendung gedacht.
flushable	integer	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Gibt Informationen zu den aggregierten Ausführungsschritten für SLICE 1 und TBL 239 aus.

```
select query, segment, bytes, slots, occupied, maxlength, is_diskbased, workmem, type
from stl_aggr where slice=1 and tbl=239
order by rows
limit 10;
```

```
 query | segment | bytes |  slots  | occupied | maxlength | is_diskbased |  workmem  |
 type
-------+---------+-------+---------+----------+-----------+--------------+-----------
+--------
   562 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   616 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   546 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   547 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   685 |       1 |    32 | 4194304 |        1 |         0 | f            | 383385600 |
 HASHED
   652 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   680 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   658 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   686 |       0 |     8 |       0 |        0 |         0 | f            |         0 |
 PLAIN
   695 |       1 |    32 | 4194304 |        1 |         0 | f            | 383385600 |
 HASHED
(10 rows)
```

STL_ALERT_EVENT_LOG

Zeichnet eine Warnung auf, wenn der Abfrageoptimierer Bedingungen identifiziert, die auf Leistungsprobleme hinweisen können. Verwenden Sie die Ansicht STL_ALERT_EVENT_LOG, um Möglichkeiten zur Verbesserung der Abfrageleistung zu identifizieren.

Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Weitere Informationen finden Sie unter Verarbeitung von Abfragen.

STL_ALERT_EVENT_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_ALERT_EVENT_LOG enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
pid	integer	Die mit der Anweisung und dem Slice verbundene Prozess-ID. Eine Abfrage kann mehrere PIDs haben, wenn sie für mehrere Slices ausgeführt wird.

Spaltenname	Datentyp	Beschreibung
xid	bigint	Mit der Anweisung verbundene Transaktions-ID.
event	character (1024)	Beschreibung des Warnungsereignisses.
solution	character (1024)	Empfohlene Lösung.
event_time	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .

Nutzungshinweise

Sie können mit STL_ALERT_EVENT_LOG potenzielle Probleme in Ihren Abfragen identifizieren und dann den Vorgehensweisen in Optimieren der Abfrageleistung folgen, um Ihr Datenbankdesign zu optimieren und Ihre Abfragen neu zu gestalten. STL_ALERT_EVENT_LOG zeichnet die folgenden Warnungen auf:

- Fehlende Statistik

  Es fehlen statistische Daten. Führen Sie nach Datenladevorgängen oder größeren Aktualisierungen ANALYZE aus, und verwenden Sie STATUPDATE mit COPY-Operationen. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift.

- Verschachtelte Schleife

  Eine verschachtelte Schleife ist normalerweise ein cartesisches Produkt. Prüfen Sie Ihre Abfrage, um sicherzustellen, dass alle beteiligten Tabellen effizient verbunden sind.

- Sehr selektiver Filter

  Das Verhältnis der zurückgegebenen zu den gescannten Zeilen ist kleiner als 0,05. Bei den gescannten Zeilen handelt es sich um den Wert von `rows_pre_user_filter` und bei den zurückgegebenen Zeilen um den Wert der Zeilen in der Systemansicht STL_SCAN. Zeigt an, dass

die Abfrage eine ungewöhnlich große Zahl von Zeilen scannt, um den Ergebnissatz zu bestimmen. Die Ursache können fehlende oder inkorrekte Sortierschlüssel sein. Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.

- Übermäßig viele Geisterzeilen

  Ein Scan übersprang eine sehr große Zahl von Zeilen, die als gelöscht markiert sind, jedoch nicht bereinigt wurden, oder es wurden Zeilen eingefügt, aber nicht bestätigt (Commit). Weitere Informationen finden Sie unter Bereinigen von Tabellen.

- Large-Verteilung

  Es wurden mehr als 1.000.000 Zeilen für Hash-Join oder Aggregierung neu verteilt. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

- Large-Broadcast

  Für mehr als 1.000.000 Zeilen wurde ein Broadcast für Hash-Join durchgeführt. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

- Serielle Ausführung

  In dem Abfrageplan wird ein Umverteilungsstil von DS_DIST_ALL_INNER angezeigt, der die serielle Ausführung erzwingt, da die gesamte innere Tabelle an einen einzelnen Knoten umverteilt wurde. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

Beispielabfragen

Die folgende Abfrage zeigt Warnungsereignisse für vier Abfragen an.

```
SELECT query, substring(event,0,25) as event,
substring(solution,0,25) as solution,
trim(event_time) as event_time from stl_alert_event_log order by query;

 query |             event             |          solution            |     event_time

-------+-------------------------------+-------------------------------
+---------------------
  6567 | Missing query planner statist | Run the ANALYZE command      | 2014-01-03
 18:20:58
  7450 | Scanned a large number of del | Run the VACUUM command to rec| 2014-01-03
 21:19:31
  8406 | Nested Loop Join in the query | Review the join predicates to| 2014-01-04
 00:34:22
```

```
 29512 | Very selective query filter:r | Review the choice of sort key| 2014-01-06
 22:00:00

(4 rows)
```

STL_ANALYZE

Zeichnet Einzelheiten zu ANALYZE-Operationen auf.

SYS_ANALYZE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_ANALYZE_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	long	Die Transaktions-ID.
Datenbank	char(30)	Der Datenbankname.
table_id	integer	Die Tabellen-ID.
status	char(15)	Das Ergebnis des ANALYZE-Befehls. Mögliche Werte sind `Full`, `Skipped` und `PredicateColumn` .
rows	double	Die Gesamtzahl der Zeilen in der Tabelle.
modified_rows	double	Die Gesamtzahl der Zeilen, die seit der letzten ANALYZE-Operation modifiziert wurden.
threshold_percent	integer	Der Wert des `analyze_threshold_percent` -Parameters.

Spaltenname	Datentyp	Beschreibung
is_auto	char(1)	Der Wert ist „true“ (t), wenn die Operation standardmäßig eine Amazon-Redshift-Analyseoperation umfasste. Der Wert ist „false (f)“, wenn der ANALYZE-Befehl explizit ausgeführt wurde.
starttime	timestamp	Zeitpunkt nach UTC, an dem die Ausführung der Analyseoperation gestartet wurde.
endtime	timestamp	Zeitpunkt nach UTC, an dem die Ausführung der Analyseoperation beendet wurde.
prevtime	timestamp	Zeitpunkt nach UTC, an dem die Tabelle zuletzt analysiert wurde.
num_predicate_cols	integer	Die aktuelle Anzahl der Prädikatspalten in der Tabelle.
num_new_predicate_cols	integer	Die Anzahl der neuen Prädikatspalten in der Tabelle seit der letzten Analyseoperation.
is_background	character(1)	Der Wert ist „true“ (t), wenn die Analyse von einer automatischen Analyseoperation ausgeführt wurde. Andernfalls ist der Wert auf „false ()“. (f).
auto_analyze_phase	character(100)	Zur internen Verwendung reserviert.
schema_name	char(128)	Der Name des Schemas für die Tabelle.
table_name	char(136)	Der Name der Tabelle.

Beispielabfragen

Das folgende Beispiel verbindet STV_TBL_PERM zur Anzeige des Tabellennamens und der Ausführungsdetails.

```
select distinct a.xid, trim(t.name) as name, a.status, a.rows, a.modified_rows,
 a.starttime, a.endtime
from stl_analyze a
join stv_tbl_perm t  on t.id=a.table_id
where name = 'users'
order by starttime;

xid    | name  | status          | rows  | modified_rows | starttime           |
 endtime
-------+-------+-----------------+-------+---------------+---------------------
+---------------------
  1582 | users | Full            | 49990 |         49990 | 2016-09-22 22:02:23 |
 2016-09-22 22:02:28
244287 | users | Full            | 24992 |         74988 | 2016-10-04 22:50:58 |
 2016-10-04 22:51:01
244712 | users | Full            | 49984 |         24992 | 2016-10-04 22:56:07 |
 2016-10-04 22:56:07
245071 | users | Skipped         | 49984 |             0 | 2016-10-04 22:58:17 |
 2016-10-04 22:58:17
245439 | users | Skipped         | 49984 |          1982 | 2016-10-04 23:00:13 |
 2016-10-04 23:00:13
(5 rows)
```

STL_ANALYZE_COMPRESSION

Zeichnet Details zu Komrimierungsanalysen während der Ausführung von COPY- oder ANALYZE COMPRESSION-Befehlen auf.

STL_ANALYZE_COMPRESSION ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_ANALYZE_COMPRESSION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
start_time	timestamp	Die Zeit, zu der die Komprimierungsanalyse gestartet wurde.
xid	bigint	Die Transaktions-ID der Komprimierungsanalyse.
tbl	integer	Die Tabellen-ID der analysierten Tabelle.
tablename	character(128)	Der Name der analysierten Tabelle.
col	integer	Der Index der Spalte in der Tabelle, die analysiert wurde, um die Komprimierungscodierung zu ermitteln.
old_encoding	character(15)	Der Codierungstyp vor der Komprimierungsanalyse.
new_encoding	character(15)	Der Codierungstyp nach der Komprimierungsanalyse.
mode	character(14)	Die möglichen Werte sind: PRESET Gibt an, dass `new_encoding` vom Amazon-Redshift-Befehl COPY basierend auf dem Datentyp der Spalte bestimmt wird. Es werden keine Stichproben der Daten genommen. ON Gibt an, dass `new_encoding` vom Amazon-Redshift-Befehl COPY basierend auf der Analyse von Beispieldaten bestimmt wird.

Spaltenname	Datentyp	Beschreibung
		ANALYZE ONLY Gibt an, dass `new_encoding` vom Amazon-Redshift-Befehl ANALYZE COMPRESSION basierend auf der Analyse von Beispieldaten bestimmt wird. Der Codierungstyp der analysierten Spalte wird jedoch nicht geändert.
best_compression_encoding	character(15)	Der Kodierungstyp, der das beste Komprimierungsverhältnis bietet.
empfohlene_Bytes	character(15)	Die Byte, die bei der Übernahme der neuen Kodierung verwendet wurden.
best_compression_bytes	character(15)	Die verwendeten Bytes, indem die beste Kompressionskodierung verwendet wurde.
ndv	bigint	Die Anzahl der unterschiedlichen Werte in den Stichprobenzeilen.

Beispielabfragen

Das folgende Beispiel inspiziert die Details der Komprimierungsanalyse der `lineitem`-Tabelle durch die letzte Ausführung des Befehls COPY in dieser Sitzung.

```
select xid, tbl, btrim(tablename) as tablename, col, old_encoding, new_encoding,
 best_compression_encoding, mode
from stl_analyze_compression
where xid = (select xid from stl_query where query = pg_last_copy_id()) order by col;

 xid  |  tbl   | tablename | col |  old_encoding   |  new_encoding   |
 best_compression_encoding |      mode
------+--------+-----------+-----+-----------------+-----------------
+---------------------------+---------------
 5308 | 158961 | $lineitem |   0 | mostly32        | az64            | delta
          | ON
```

```
 5308 | 158961 | $lineitem |   1 | mostly32        | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   2 | lzo             | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   3 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   4 | bytedict        | az64            | bytedict
          | ON
 5308 | 158961 | $lineitem |   5 | mostly32        | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   6 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   7 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   8 | lzo             | lzo             | lzo
          | ON
 5308 | 158961 | $lineitem |   9 | runlength       | runlength       | runlength
          | ON
 5308 | 158961 | $lineitem |  10 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |  11 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |  12 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |  13 | bytedict        | bytedict        | bytedict
          | ON
 5308 | 158961 | $lineitem |  14 | bytedict        | bytedict        | bytedict
          | ON
 5308 | 158961 | $lineitem |  15 | text255         | text255         | text255
          | ON
(16 rows)
```

STL_BCAST

Protokolliert Informationen zur Netzwerkaktivität während der Ausführung von Abfrageschritten, die Daten-Broadcasts durchführen. Der Netzwerkdatenverkehr wird nach Anzahl der Zeilen, Byte und Paketen erfasst, die während eines bestimmten Schrittes auf einem bestimmten Slice über das Netzwerk gesendet werden. Die Dauer des Schrittes ist die Differenz zwischen dem protokollierten Start- und dem Endzeitpunkt.

Suchen Sie zur Identifizierung von Broadcast-Schritten in einer Abfrage nach bcast-Beschriftungen in der Ansicht SVL_QUERY_SUMMARY, oder führen Sie den Befehl EXPLAIN aus, und suchen Sie dann nach Schrittattributen, die „bcast“ enthalten.

STL_BCAST ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_BCAST enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkomma

Spaltenname	Datentyp	Beschreibung
		stellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
packets	integer	Gesamtzahl der über das Netzwerk gesendeten Pakete.

Beispielabfragen

Das folgende Beispiel gibt Broadcast-Informationen für die Abfragen aus, die ein oder mehrere Pakete enthalten, und bei denen die Differenz zwischen Beginn und Ende der Abfrage eine Sekunde oder mehr betrug.

```
select query, slice, step, rows, bytes, packets, datediff(seconds, starttime, endtime)
from stl_bcast
where packets>0 and datediff(seconds, starttime, endtime)>0;
```

```
 query | slice | step | rows | bytes | packets | date_diff
-------+-------+------+------+-------+---------+-----------
   453 |     2 |    5 |    1 |   264 |       1 |         1
   798 |     2 |    5 |    1 |   264 |       1 |         1
  1408 |     2 |    5 |    1 |   264 |       1 |         1
  2993 |     0 |    5 |    1 |   264 |       1 |         1
  5045 |     3 |    5 |    1 |   264 |       1 |         1
  8073 |     3 |    5 |    1 |   264 |       1 |         1
  8163 |     3 |    5 |    1 |   264 |       1 |         1
```

```
  9212 |     1 |    5 |    1 |   264 |       1 |         1
  9873 |     1 |    5 |    1 |   264 |       1 |         1
(9 rows)
```

STL_COMMIT_STATS

Bietet Metriken zur Commit-Leistung, einschließlich des Timings der verschiedenen Commit-Phasen und der Anzahl der Commit-Blöcke. Fragen Sie STL_COMMIT_STATS ab, um festzustellen, welcher Teil einer Transaktion für Commit aufgewendet wurde, und wie hoch die Warteschlangenaktivität ist.

STL_COMMIT_STATS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_TRANSACTION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
xid	bigint	ID der im Commit-Prozess befindlichen Transaktion.
node	integer	Knotennummer. -1 ist der Führungsknoten.
startqueue	timestamp	Beginn des Warteschlangenvorgangs für Commit.
startwork	timestamp	Beginn des Commit-Vorgangs.
endflush	timestamp	Ende der Flush-Phase für kontaminierte Blöcke.
endstage	timestamp	Ende der Metadaten-Staging-Phase.
endlocal	timestamp	Ende der lokalen Commit-Phase.
startglobal	timestamp	Beginn der globalen Phase.
endtime	timestamp	Ende des Commit-Vorgangs.

Spaltenname	Datentyp	Beschreibung
queuelen	bigint	Anzahl der Transaktionen, die in der Commit-Warteschlange vor dieser Transaktion lagen.
permblocks	bigint	Anzahl der vorhandenen permanenten Blöcke zum Zeitpunkt dieses Commit-Vorgangs.
newblocks	bigint	Anzahl der neuen permanenten Blöcke zum Zeitpunkt dieses Commit-Vorgangs.
dirtyblocks	bigint	Anzahl der Blöcke, die im Rahmen dieses Commit-Vorgangs geschrieben werden mussten.
headers	bigint	Anzahl der Block-Köpfe, die im Rahmen dieses Commit-Vorgangs geschrieben werden mussten.
numxids	integer	Die Anzahl der aktiven DML-Transaktionen.
oldestxid	bigint	Die XID der ältesten aktiven DML-Transaktion.
extwritelatency	bigint	Diese Information ist nur für die interne Verwendung gedacht.
metadatawritten	int	Diese Information ist nur für die interne Verwendung gedacht.
tombstonedblocks	bigint	Diese Information ist nur für die interne Verwendung gedacht.
tossedblocks	bigint	Diese Information ist nur für die interne Verwendung gedacht.
batched_by	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfrage

```
select node, datediff(ms,startqueue,startwork) as queue_time,
datediff(ms, startwork, endtime) as commit_time, queuelen
```

```
from stl_commit_stats
where xid = 2574
order by node;

node | queue_time   | commit_time | queuelen
-----+--------------+-------------+---------
  -1 |            0 |         617 |        0
   0 | 444950725641 |         616 |        0
   1 | 444950725636 |         616 |        0
```

STL_CONNECTION_LOG

Protokolliert Authentifizierungsversuche sowie Verbindungen und Verbindungstrennungen.

STL_CONNECTION_LOG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_CONNECTION_LOG zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
event	character(50)	Verbindungs- oder Authentifizierungsereignis.
recordtime	timestamp	Uhrzeit, zu der das Ereignis aufgetreten ist.
remotehost	character(45)	Name oder IP-Adresse des Remote-Hosts.
remoteport	character(32)	Portnummer für den Remote-Host.
pid	integer	Die mit der Anweisung verbundene Prozess-ID.
dbname	character(50)	Datenbankname.
Benutzernamen ein	character(50)	Benutzername.

Spaltenname	Datentyp	Beschreibung
authmethod	character(32)	Authentifizierungsmethode.
duration	integer	Dauer der Verbindung in Mikrosekunden.
sslversion	character(50)	Secure Sockets Layer (SSL)-Version.
sslcipher	character(128)	SSL-Verschlüsselungsverfahren.
mtu	integer	Maximum Transmission Unit (MTU).
sslcompression	character(64)	SSL-Kompressionstyp.
sslexpansion	character(64)	SSL-Expansionstyp.
iamauthguid	character(36)	Die IAM-Authentifizierungs-ID für die CloudTrail Anfrage.
application_name	character(250)	Der ursprüngliche oder aktualisierte Name der Anwendung für eine Sitzung.
os_version	character(64)	Die Version des Betriebssystems, das sich auf dem Clientcomputer befindet, der eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellt.
driver_version	character(64)	Die Version des ODBC- oder JDBC-Treibers, die von Ihren SQL-Client-Tools von Drittanbietern eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellt.
plugin_name	character(32)	Der Name des Plug-Ins, mit dem Sie eine Verbindung zu Ihrem Amazon-Redshift-Cluster herstellen.

Spaltenname	Datentyp	Beschreibung
protocol_version	integer	Die interne Protokollversion, die der Amazon-Redshift-Treiber beim Herstellen der Verbindung mit dem Server verwendet . Die Protokollversionen werden zwischen Treiber und Server ausgehandelt. Die Version beschreibt die verfügbaren Funktionen. Gültige Werte sind: • 0 (BASE_SERVER_PROTOCOL_VERSION) • 1 (EXTENDED_RESULT_METADATA_SERVER_PROTOCOL_VERSION) – Um einen Umlauf pro Abfrage einzusparen, sendet der Server zusätzliche Metadaten informationen zu Ergebnismengen. • 2 (BINARY_PROTOCOL_VERSION) – Abhängig vom Datentyp der Ergebnismenge sendet der Server Daten im Binärformat. • 3 (EXTENDED2_RESULT_METADATA_SERVER_PROTOCOL_VERSION) – Der Server sendet Informationen zur Groß- und Kleinschreibung (Sortierung) einer Spalte.
sessionid	character(36)	Der global eindeutige Bezeichner für die aktuelle Sitzung. Die Sitzungs-ID bleibt auch nach Neustarts bei Knotenfehlern bestehen.
Kompression	character(16)	Der für die Verbindung verwendete Komprimierungsalgorithmus.

Beispielabfragen

Führen Sie folgende Abfrage aus, um die Details zu offenen Verbindungen anzuzeigen.

```
select recordtime, username, dbname, remotehost, remoteport
from stl_connection_log
where event = 'initiating session'
and pid not in
(select pid from stl_connection_log
where event = 'disconnecting session')
```

```
order by 1 desc;

recordtime          | username    | dbname     | remotehost    | remoteport

--------------------+-------------+------------+---------------+------------
2014-11-06 20:30:06 | rdsdb       | dev        | [local]       |

2014-11-06 20:29:37 | test001     | test       | 10.49.42.138  | 11111

2014-11-05 20:30:29 | rdsdb       | dev        | 10.49.42.138  | 33333

2014-11-05 20:28:35 | rdsdb       | dev        | [local]       |
(4 rows)
```

Das folgende Beispiel illustriert einen fehlgeschlagenen Authentifizierungsversuch sowie eine erfolgreiche Verbindung und Verbindungstrennung.

```
select event, recordtime, remotehost, username
from stl_connection_log order by recordtime;

            event      |         recordtime        |  remotehost  | username

-----------------------+---------------------------+--------------+---------
authentication failure | 2012-10-25 14:41:56.96391 | 10.49.42.138 | john

authenticated          | 2012-10-25 14:42:10.87613 | 10.49.42.138 | john

initiating session     | 2012-10-25 14:42:10.87638 | 10.49.42.138 | john

disconnecting session  | 2012-10-25 14:42:19.95992 | 10.49.42.138 | john

(4 rows)
```

Das folgende Beispiel zeigt die Version des ODBC-Treibers, das Betriebssystem auf dem Clientcomputer und das Plug-In, das für die Verbindung mit dem Amazon-Redshift-Cluster verwendet wird. In diesem Beispiel dient das verwendete Plug-In zur standardmäßigen ODBC-Treiberauthentifizierung unter Verwendung eines Anmeldenamens und eines Kennworts.

```
select driver_version, os_version, plugin_name from stl_connection_log;

driver_version                          |  os_version                       |
 plugin_name
```

```
---------------------------------------+-----------------------------------
+--------------------
Amazon Redshift ODBC Driver 1.4.15.0001 | Darwin 18.7.0 x86_64              | none
Amazon Redshift ODBC Driver 1.4.15.0001 | Linux 4.15.0-101-generic x86_64   | none
```

Das folgende Beispiel zeigt die Version des Betriebssystems auf dem Clientcomputer, die Treiberversion und die Protokollversion an.

```
select os_version, driver_version, protocol_version from stl_connection_log;

os_version                     |  driver_version              | protocol_version
-------------------------------+------------------------------+--------------------
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
```

STL_DDLTEXT

Erfasst die folgenden DDL-Anweisungen, die auf dem System ausgeführt wurden.

Diese DDL-Anweisungen enthalten die folgenden Abfragen und Objekte:

- CREATE SCHEMA, TABLE, VIEW
- DROP SCHEMA, TABLE, VIEW
- ALTER SCHEMA, TABLE

Vgl. auch STL_QUERYTEXT, STL_UTILITYTEXT und SVL_STATEMENTTEXT. Diese Ansichten zeigen einen Verlauf der SQL-Befehle, die auf dem System ausgeführt wurden. Dieser Verlauf ist nützlich für die Fehlerbehebung sowie für die Erstellung eines Audit-Trails für alle Systemaktivitäten.

Verwenden Sie die Spalten STARTTIME und ENDTIME, um zu erfahren, welche Anweisungen während eines bestimmten Zeitraums protokolliert wurden. Lange SQL-Textblöcke werden in 200 Zeichen lange Zeilen umgebrochen; die Spalte SEQUENCE identifiziert Textfragmente, die zu einer einzelnen Anweisung gehören.

STL_DDLTEXT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Mit der Anweisung verbundene Transaktions-ID.
pid	integer	Die mit der Anweisung verbundene Prozess-ID.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist dieses Feld leer.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
sequence	integer	Wenn eine einzelne Anweisung mehr als 200 Zeichen enthält, werden weitere Zeilen für diese Anweisung protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite usw.

Spaltenname	Datentyp	Beschreibung
Text	character(200)	SQL-Text, in Schritten von je 200 Zeichen. Diese Feld kann Sonderzeichen wie Backslash (\\) und Zeilenumbruch (\n) enthalten.

Beispielabfragen

Die folgende Abfrage gibt Datensätze zurück, die zuvor ausgeführte DDL-Anweisungen enthalten.

```
select xid, starttime, sequence, substring(text,1,40) as text
from stl_ddltext order by xid desc, sequence;
```

Unten sehen Sie eine Beispielausgabe mit vier CREATE TABLE-Anweisungen. Die DDL-Anweisungen erscheinen in der `text`-Spalte, die zur leichteren Lesbarkeit abgeschnitten wird.

```
 xid  |         starttime          | sequence |                  text
------+----------------------------+----------
+----------------------------------------
 1806 | 2013-10-23 00:11:14.709851 |        0 | CREATE TABLE supplier ( s_suppkey int4
 N
 1806 | 2013-10-23 00:11:14.709851 |        1 |  s_comment varchar(101) NOT NULL )
 1805 | 2013-10-23 00:11:14.496153 |        0 | CREATE TABLE region ( r_regionkey int4
 N
 1804 | 2013-10-23 00:11:14.285986 |        0 | CREATE TABLE partsupp ( ps_partkey int8
 1803 | 2013-10-23 00:11:14.056901 |        0 | CREATE TABLE part ( p_partkey int8 NOT
 N
 1803 | 2013-10-23 00:11:14.056901 |        1 | ner char(10) NOT NULL , p_retailprice
 nu
(6 rows)
```

Wiederherstellen von gespeichertem SQL

Die folgende SQL listet Zeilen auf, die in der Spalte `text` von STL_DDLTEXT gespeichert sind. Die Zeilen sind geordnet nach `xid` und `sequence`. Wenn die ursprüngliche SQL mehr als 200 Zeichen auf mehreren Zeilen umfasste, kann STL_DDLTEXT mehrere Zeilen nach `sequence` enthalten.

```
SELECT xid, sequence, LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text)
 END, '') WITHIN GROUP (ORDER BY sequence) as query_statement
```

```
FROM stl_ddltext GROUP BY xid, sequence ORDER BY xid, sequence;
```

```
xid     |  sequence | query_statement
--------+-----------+-----------------
7886671    0          create external schema schema_spectrum_uddh\nfrom data catalog
\ndatabase 'spectrum_db_uddh'\niam_role ''\ncreate external database if not exists;
7886752    0          CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league\n(\n
  league_rank smallint,\n  prev_rank   smallint,\n  club_name   varchar(15),\n
 league_name varchar(20),\n  league_off  decimal(6,2),\n  le
7886752    1          ague_def  decimal(6,2),\n  league_spi  decimal(6,2),\n
 league_nspi smallint\n)\nROW FORMAT DELIMITED \n    FIELDS TERMINATED BY ',' \n
 LINES TERMINATED BY '\\n\\l'\nstored as textfile\nLOCATION 's
7886752    2          3://mybucket-spectrum-uddh/'\ntable properties
 ('skip.header.line.count'='1');
...
```

Um das in der Tabelle `text` von STL_DDLTEXT gespeicherte SQL wiederherzustellen, führen Sie den folgenden SQL-Code aus. Dadurch werden die DDL-Anweisungen aus einem oder mehreren Segmenten in der `text`-Spalte kombiniert. Bevor Sie das wiederhergestellte SQL ausführen, ersetzen Sie alle Sonderzeichen (\n) durch eine neue Zeile in Ihrem SQL-Client. Die Ergebnisse der folgenden SELECT-Anweisung stellen drei Zeilen in der Reihenfolge zusammen, um das SQL zu rekonstruieren, im Feld `query_statement`.

```
SELECT LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END) WITHIN
 GROUP (ORDER BY sequence) as query_statement
FROM stl_ddltext GROUP BY xid, endtime order by xid, endtime;
```

```
query_statement
--------------
create external schema schema_spectrum_uddh\nfrom data catalog\ndatabase
 'spectrum_db_uddh'\niam_role ''\ncreate external database if not exists;
CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league\n(\n  league_rank smallint,
\n  prev_rank   smallint,\n  club_name   varchar(15),\n  league_name varchar(20),\n
 league_off  decimal(6,2),\n  league_def  decimal(6,2),\n  league_spi  decimal(6,2),
\n  league_nspi smallint\n)\nROW FORMAT DELIMITED \n    FIELDS TERMINATED BY ',' \n
  LINES TERMINATED BY '\\n\\l'\nstored as textfile\nLOCATION 's3://mybucket-spectrum-
uddh/'\ntable properties ('skip.header.line.count'='1');
```

STL_DELETE

Analysiert Lösch-Ausführungsschritte für Abfragen.

STL_DELETE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_DELETE enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von

Spaltenname	Datentyp	Beschreibung
		6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
tbl	integer	Tabellen-ID.

Beispielabfragen

Zur Erstellung einer Zeile in STL_DELETE fügt das folgende Beispiel eine Zeile in die Tabelle EVENT ein und löscht sie dann.

Fügen Sie zuerst eine Zeile in die Tabelle EVENT ein, und prüfen Sie, dass dies geschah.

```
insert into event(eventid,venueid,catid,dateid,eventname)
values ((select max(eventid)+1 from event),95,9,1857,'Lollapalooza');
```

```
select * from event
where eventname='Lollapalooza'
order by eventid;
```

```
 eventid | venueid | catid | dateid |  eventname   |      starttime
---------+---------+-------+--------+--------------+---------------------
    4274 |     102 |     9 |   1965 | Lollapalooza | 2008-05-01 19:00:00
    4684 |     114 |     9 |   2105 | Lollapalooza | 2008-10-06 14:00:00
    5673 |     128 |     9 |   1973 | Lollapalooza | 2008-05-01 15:00:00
```

```
    5740 |      51 |     9 |   1933 | Lollapalooza | 2008-04-17 15:00:00
    5856 |     119 |     9 |   1831 | Lollapalooza | 2008-01-05 14:00:00
    6040 |     126 |     9 |   2145 | Lollapalooza | 2008-11-15 15:00:00
    7972 |      92 |     9 |   2026 | Lollapalooza | 2008-07-19 19:30:00
    8046 |      65 |     9 |   1840 | Lollapalooza | 2008-01-14 15:00:00
    8518 |      48 |     9 |   1904 | Lollapalooza | 2008-03-19 15:00:00
    8799 |      95 |     9 |   1857 | Lollapalooza |
(10 rows)
```

Löschen Sie dann die Zeile, die Sie der Tabelle EVENT hinzugefügt haben, und prüfen Sie, das dies geschah.

```
delete from event
where eventname='Lollapalooza' and eventid=(select max(eventid) from event);
```

```
select * from event
where eventname='Lollapalooza'
order by eventid;
```

```
 eventid | venueid | catid | dateid |  eventname   |      starttime
---------+---------+-------+--------+--------------+---------------------
    4274 |     102 |     9 |   1965 | Lollapalooza | 2008-05-01 19:00:00
    4684 |     114 |     9 |   2105 | Lollapalooza | 2008-10-06 14:00:00
    5673 |     128 |     9 |   1973 | Lollapalooza | 2008-05-01 15:00:00
    5740 |      51 |     9 |   1933 | Lollapalooza | 2008-04-17 15:00:00
    5856 |     119 |     9 |   1831 | Lollapalooza | 2008-01-05 14:00:00
    6040 |     126 |     9 |   2145 | Lollapalooza | 2008-11-15 15:00:00
    7972 |      92 |     9 |   2026 | Lollapalooza | 2008-07-19 19:30:00
    8046 |      65 |     9 |   1840 | Lollapalooza | 2008-01-14 15:00:00
    8518 |      48 |     9 |   1904 | Lollapalooza | 2008-03-19 15:00:00
(9 rows)
```

Fragen Sie dann STL_DELETE ab, um die Ausführungsschritte für die Löschung anzuzeigen. In diesem Beispiel gab die Abfrage mehr als 300 Zeilen aus, die Ausgabe unten wurde daher für die Demonstration verkürzt.

```
select query, slice, segment, step, tasknum, rows, tbl from stl_delete order by query;
```

```
 query | slice | segment | step | tasknum | rows |  tbl
```

```
-------+-------+---------+------+---------+------+--------
     7 |     0 |       0 |    1 |       0 |    0 | 100000
     7 |     1 |       0 |    1 |       0 |    0 | 100000
     8 |     0 |       0 |    1 |       2 |    0 | 100001
     8 |     1 |       0 |    1 |       2 |    0 | 100001
     9 |     0 |       0 |    1 |       4 |    0 | 100002
     9 |     1 |       0 |    1 |       4 |    0 | 100002
    10 |     0 |       0 |    1 |       6 |    0 | 100003
    10 |     1 |       0 |    1 |       6 |    0 | 100003
    11 |     0 |       0 |    1 |       8 |    0 | 100253
    11 |     1 |       0 |    1 |       8 |    0 | 100253
    12 |     0 |       0 |    1 |       0 |    0 | 100255
    12 |     1 |       0 |    1 |       0 |    0 | 100255
    13 |     0 |       0 |    1 |       2 |    0 | 100257
    13 |     1 |       0 |    1 |       2 |    0 | 100257
    14 |     0 |       0 |    1 |       4 |    0 | 100259
    14 |     1 |       0 |    1 |       4 |    0 | 100259
    ...
```

STL_DISK_FULL_DIAG

Protokolliert Informationen zu Fehlern, die aufgezeichnet werden, wenn der Datenträger voll ist.

STL_DISK_FULL_DIAG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
currenttime	bigint	Tag und Uhrzeit der Fehlererstellung in Mikrosekunden seit dem 1. Januar 2000.
node_num	bigint	Die ID des Knotens.
query_id	bigint	Die ID der Abfrage, die den Fehler verursacht hat.
temp_blocks	bigint	Die Anzahl der temporären Blöcke, die von der Abfrage erstellt wurden

Beispielabfragen

Das folgende Beispiel gibt Details zu den gespeicherten Daten zurück, wenn der Datenträger voll ist und ein entsprechender Fehler ausgegeben wird.

```
select * from stl_disk_full_diag
```

Im folgenden Beispiel wird die Zeichenfolge `currenttime` in einen Zeitstempel konvertiert.

```
select '2000-01-01'::timestamp + (currenttime/1000000.0)* interval '1 second' as
 currenttime,node_num,query_id,temp_blocks from pg_catalog.stl_disk_full_diag;
```

```
        currenttime         | node_num | query_id | temp_blocks
----------------------------+----------+----------+-------------
 2019-05-18 19:19:18.609338 |        0 |   569399 |       70982
 2019-05-18 19:37:44.755548 |        0 |   569580 |       70982
 2019-05-20 13:37:20.566916 |        0 |   597424 |       70869
```

STL_DIST

Protokolliert Informationen zur Netzwerkaktivität während der Ausführung von Abfrageschritten, die Daten verteilen. Der Netzwerkdatenverkehr wird nach Anzahl der Zeilen, Byte und Paketen erfasst, die während eines bestimmten Schrittes auf einem bestimmten Slice über das Netzwerk gesendet werden. Die Dauer des Schrittes ist die Differenz zwischen dem protokollierten Start- und dem Endzeitpunkt.

Suchen Sie zur Identifizierung von Verteilungsschritten in einer Abfrage nach dist-Beschriftungen in der Ansicht QUERY_SUMMARY, oder führen Sie den Befehl EXPLAIN aus, und suchen Sie dann nach Schrittattributen, die „dist“ enthalten.

STL_DIST ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_DIST enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschl ange und Zeit für die Ausführung mit einer Genauigkeit

Spaltenname	Datentyp	Beschreibung
		von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
packets	integer	Gesamtzahl der über das Netzwerk gesendeten Pakete.

Beispielabfragen

Das folgende Beispiel gibt Verteilungsinformationen für Abfragen mit einem oder mehreren Paketen und einer Dauer über Null aus.

```
select query, slice, step, rows, bytes, packets,
datediff(seconds, starttime, endtime) as duration
from stl_dist
where packets>0 and datediff(seconds, starttime, endtime)>0
order by query
limit 10;
```

```
 query  | slice | step |  rows  |  bytes  | packets | duration
--------+-------+------+--------+---------+---------+----------
    567 |     1 |    4 |  49990 | 6249564 |     707 |        1
    630 |     0 |    5 |   8798 |  408404 |      46 |        2
    645 |     1 |    4 |   8798 |  408404 |      46 |        1
    651 |     1 |    5 | 192497 | 9226320 |    1039 |        6
    669 |     1 |    4 | 192497 | 9226320 |    1039 |        4
    675 |     1 |    5 |   3766 |  194656 |      22 |        1
    696 |     0 |    4 |   3766 |  194656 |      22 |        1
    705 |     0 |    4 |    930 |   44400 |       5 |        1
 111525 |     0 |    3 |     68 |   17408 |       2 |        1
(9 rows)
```

STL_ERROR

Zeichnet von der Amazon-Redshift-Datenbankengine generierte interne Verarbeitungsfehler auf. STL_ERROR zeichnet keine SQL-Fehler oder -Meldungen auf. Die Informationen in STL_ERROR sind nützlich für die Behebung bestimmter Fehler. Ein AWS Support-Techniker bittet Sie möglicherweise im Rahmen der Fehlerbehebung um die Angabe dieser Informationen.

STL_ERROR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Eine Liste der Fehlercodes, die beim Laden mit dem COPY-Befehl generiert werden, finden Sie unter Ladefehlerreferenz.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
Verarbeitung	character(12)	Der Prozess, der die Ausnahme ausgegeben hat.
recordtime	timestamp	Zeitpunkt, zu dem der Fehler aufgetreten ist.
pid	integer	Prozess-ID. Die Tabelle STL_QUERY enthält Prozess-IDs und eindeutige Abfrage-IDs für ausgeführte Abfragen.
errcode	integer	Der Fehlercode, der der Fehlerkategorie entspricht.
file	character(90)	Name der Quelldatei, in der der Fehler aufgetreten ist.

Spaltenname	Datentyp	Beschreibung
linenum	integer	Zeilennummer in der Quelldatei, in der der Fehler aufgetreten ist.
context	character(100)	Ursache des Fehlers.
error	character(512)	Fehlermeldung.

Beispielabfragen

Das folgende Beispiel ruft die Fehlerinformationen aus STL_ERROR ab.

```
select process, errcode, linenum as line,
trim(error) as err
from stl_error;

   process    | errcode | line |                                 err
--------------+---------+------
+-----------------------------------------------------------------
 padbmaster   |    8001 |  194 | Path prefix: s3://redshift-downloads/testnulls/
venue.txt*
 padbmaster   |    8001 |  529 | Listing bucket=redshift-downloads prefix=tests/
category-csv-quotes
 padbmaster   |       2 |  190 | database "template0" is not currently accepting
 connections
 padbmaster   |      32 | 1956 | pq_flush: could not send data to client: Broken pipe
(4 rows)
```

STL_EXPLAIN

Zeigt den EXPLAIN-Plan für eine Abfrage an, die zur Ausführung übermittelt wurde.

STL_EXPLAIN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_EXPLAIN enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
nodeid	integer	Plan-Knoten-ID, wo ein Knoten mit einem oder mehreren Schritten in der Ausführung der Abfrage verwunden ist.
parentid	integer	Plan-Knoten-ID für einen übergeordneten Knoten. Zu einem übergeordneten Knoten gehört eine bestimmte Anzahl untergeordneter Knoten. So ist beispielsweise ein Zusammenführungsknoten der übergeordnete Knoten der Scans in den verbundenen Tabellen.
plannode	character(400)	Der Knotentext aus der EXPLAIN-Ausgabe. Planknoten, die sich auf die Ausführung auf Datenverarbeitungsknoten beziehen, haben das Präfix **XN** in der EXPLAIN-Ausgabe.
info	character(400)	Qualifizierer- und Filterinformationen für den Plan-Knoten. Diese Spalte enthält zum Beispiel Join-Bedingungen und Einschränkungen für WHERE-Klauseln.

Beispielabfragen

Sehen Sie sich die folgende EXPLAIN-Ausgabe für eine aggregierte Join-Abfrage an:

```
explain select avg(datediff(day, listtime, saletime)) as avgwait
from sales, listing where sales.listid = listing.listid;
                                QUERY PLAN

------------------------------------------------------------------------------
 XN Aggregate  (cost=6350.30..6350.31 rows=1 width=16)
  ->  XN Hash Join DS_DIST_NONE  (cost=47.08..6340.89 rows=3766 width=16)
        Hash Cond: ("outer".listid = "inner".listid)
        -> XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=12)
        -> XN Hash  (cost=37.66..37.66 rows=3766 width=12)
              -> XN Seq Scan on sales  (cost=0.00..37.66 rows=3766 width=12)
(6 rows)
```

Wenn Sie diese Abfrage ausführen und die Abfrage-ID 10 ist, können Sie anhand der Tabelle STL_EXPLAIN die gleichen Arten von Informationen sehen, die der Befehl EXPLAIN ausgibt:

```
select query,nodeid,parentid,substring(plannode from 1 for 30),
substring(info from 1 for 20) from stl_explain
where query=10 order by 1,2;

query| nodeid |parentid|           substring            |    substring
-----+--------+--------+--------------------------------+-------------------
10   |      1 |      0 |XN Aggregate  (cost=6717.61..6  |
10   |      2 |      1 |  -> XN Merge Join DS_DIST_NO   | Merge Cond:("outer"
10   |      3 |      2 |       -> XN Seq Scan on lis    |
10   |      4 |      2 |       -> XN Seq Scan on sal    |
(4 rows)
```

Betrachten Sie folgende Abfrage:

```
select event.eventid, sum(pricepaid)
from event, sales
where event.eventid=sales.eventid
group by event.eventid order by 2 desc;

eventid |   sum
--------+----------
    289 | 51846.00
   7895 | 51049.00
```

```
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
...
```

Wenn die ID dieser Abfrage 15 ist, gibt die folgende Systemansichtsabfrage die ausgeführten Plan-Knoten aus. In diesem Fall ist die Reihenfolge der Knoten umgekehrt, um die tatsächliche Ausführungsreihenfolge zu zeigen:

```
select query,nodeid,parentid,substring(plannode from 1 for 56)
from stl_explain where query=15 order by 1, 2 desc;

query|nodeid|parentid|                          substring
-----+------+--------+-----------------------------------------------------------
15   |    8 |      7 |                               -> XN Seq Scan on eve
15   |    7 |      5 |                          -> XN Hash(cost=87.98..87.9
15   |    6 |      5 |                          -> XN Seq Scan on sales(cos
15   |    5 |      4 |                     -> XN Hash Join DS_DIST_OUTER(cos
15   |    4 |      3 |               -> XN HashAggregate(cost=862286577.07..
15   |    3 |      2 |         -> XN Sort(cost=1000862287175.47..10008622871
15   |    2 |      1 |  -> XN Network(cost=1000862287175.47..1000862287197.
15   |    1 |      0 |XN Merge(cost=1000862287175.47..1000862287197.46 rows=87
(8 rows)
```

Die folgende Abfrage ruft die Abfrage-IDs für alle Abfrage-Pläne ab, die eine Fensterfunktion beinhalten:

```
select query, trim(plannode) from stl_explain
where plannode like '%Window%';

query|                                    btrim
-----+-------------------------------------------------------------------------
26   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
27   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
(2 rows)
```

STL_FILE_SCAN

Gibt die Dateien aus, die Amazon Redshift beim Laden von Daten mit dem COPY-Befehl gelesen hat.

Die Abfrage dieser Ansicht kann beim Beheben von Datenladefehlern helfen. STL_FILE_SCAN kann besonders bei der Feststellung von Problemen bei parallelen Datenladevorgängen nützlich sein, da

parallele Datenladevorgänge typischerweise zahlreiche Dateien mit einem einzigen COPY-Befehl laden.

STL_FILE_SCAN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_FILE_SCAN enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_LOAD_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
Name	character(90)	Vollständiger Pfad und Name der geladenen Datei.
lines	bigint	Anzahl der aus der Datei gelesenen Zeilen.
bytes	bigint	Anzahl der aus der Datei gelesenen Bytes.
loadtime	bigint	Zeitdauer für das Laden der Datei (in Mikrosekunden).

Spaltenname	Datentyp	Beschreibung
curtime	Zeitstempel	Zeitstempel für den Zeitpunkt, an dem Amazon Redshift mit der Verarbeitung der Datei begonnen hat.
is_partial	integer	Wert, der bei true (1) angibt, dass die Eingabedatei während eines COPY-Vorgangs in Bereiche aufgeteilt wird. Wenn dieser Wert false (0) ist, wird die Eingabedatei nicht geteilt.
start_offset	bigint	Wenn die Eingabedatei während eines COPY-Vorgangs geteilt wird, gibt dieser Wert den Offset-Wert der Teilung (in Byte) an. Wenn die Datei nicht geteilt wird, ist dieser Wert 0.

Beispielabfragen

Die folgende Abfrage ruft die Namen und Ladezeiten aller Dateien ab, für deren Lesevorgang Amazon Redshift mehr als 1 000 000 Mikrosekunden benötigt hat.

```
select trim(name)as name, loadtime from stl_file_scan
where loadtime > 1000000;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück.

```
           name             | loadtime
----------------------------+----------
 listings_pipe.txt          |  9458354
 allusers_pipe.txt          |  2963761
 allevents_pipe.txt         |  1409135
 tickit/listings_pipe.txt   |  7071087
 tickit/allevents_pipe.txt  |  1237364
 tickit/allusers_pipe.txt   |  2535138
 listings_pipe.txt          |  6706370
 allusers_pipe.txt          |  3579461
 allevents_pipe.txt         |  1313195
 tickit/allusers_pipe.txt   |  3236060
 tickit/listings_pipe.txt   |  4980108
(11 rows)
```

STL_HASH

Analysiert Hash-Ausführungsschritte für Abfragen.

STL_HASH ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_HASH enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von

Spaltenname	Datentyp	Beschreibung
		6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
slots	integer	Gesamtzahl der Hash-Buckets.
occupied	integer	Gesamtzahl der Slots, die Datensätze enthalten.
maxlength	integer	Größe des größten Slots.
tbl	integer	Tabellen-ID.
is_diskbased	character(1)	Bei „true (t)“ wurde die Abfrage als festplattenbasierte Operation ausgeführt. Bei „false (f)“ wurde die Abfrage im Arbeitsspeicher ausgeführt.
workmem	bigint	Gesamtgröße des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.
num_parts	integer	Gesamtzahl der Partitionen, in die eine Hash-Tabelle während eines Hash-Schrittes unterteilt wurde.
est_rows	bigint	Die geschätzte Anzahl der Zeilen für den Hash-Vorgang.

Spaltenname	Datentyp	Beschreibung
num_blocks_permitted	integer	Diese Information ist nur für die interne Verwendung gedacht.
resizes	integer	Diese Information ist nur für die interne Verwendung gedacht.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.
runtime_filter_size	integer	Die Größe des Laufzeitfilters in Bytes
max_runtime_filter_size	integer	Die maximale Größe des Laufzeitfilters in Bytes

Beispielabfragen

Das folgende Beispiel gibt Informationen zur Anzahl der Partitionen, die in einem Hash für Abfrage 720 verwendet wurden, und gibt an, dass keiner der Schritte auf der Festplatte ausgeführt wurde.

```
select slice, rows, bytes, occupied, workmem, num_parts, est_rows,
 num_blocks_permitted, is_diskbased
from stl_hash
where query=720 and segment=5
order by slice;
```

```
 slice | rows | bytes  | occupied | workmem  | num_parts | est_rows |
 num_blocks_permitted | is_diskbased
-------+------+--------+----------+----------+-----------+----------
+----------------------+--------------
     0 |  145 | 585800 |        1 | 88866816 |        16 |        1 |
 52              f
     1 |    0 |      0 |        0 |        0 |        16 |        1 |
 52              f
```

```
(2 rows)
```

STL_HASHJOIN

Analysiert Hash-Join-Ausführungsschritte für Abfragen.

STL_HASHJOIN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_HASHJOIN enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange

Spaltenname	Datentyp	Beschreibung
		und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
tbl	integer	Tabellen-ID.
num_parts	integer	Gesamtzahl der Partitionen, in die eine Hash-Tabelle während eines Hash-Schrittes unterteilt wurde.

Spaltenname	Datentyp	Beschreibung
join_type	integer	Die Art des Joins für den Schritt: • 0. Die Abfrage hat einen inneren Join verwendet. • 1. Die Abfrage hat einen linken äußeren Join verwendet. • 2. Die Abfrage hat einen vollständigen äußeren Join verwendet. • 3. Die Abfrage hat einen rechten äußeren Join verwendet. • 4. Die Abfrage hat einen UNION-Operator verwendet. • 5. Die Abfrage hat eine IN-Bedingung verwendet. • 6. Diese Information ist nur für die interne Verwendung gedacht. • 7. Diese Information ist nur für die interne Verwendung gedacht. • 8. Diese Information ist nur für die interne Verwendung gedacht. • 9. Diese Information ist nur für die interne Verwendung gedacht. • 10. Diese Information ist nur für die interne Verwendung gedacht. • 11. Diese Information ist nur für die interne Verwendung gedacht. • 12. Diese Information ist nur für die interne Verwendung gedacht.
hash_looped	character(1)	Diese Information ist nur für die interne Verwendung gedacht.
switched_parts	character(1)	Zeigt an, ob die Build- (oder äußere) und die Sonden- (oder innere) Seite vertauscht wurden.

Spaltenname	Datentyp	Beschreibung
used_prefetching	character(1)	Diese Information ist nur für die interne Verwendung gedacht.
hash_segment	integer	Das Segment des entsprechenden Hash-Schritts.
hash_step	integer	Die Schrittnummer des entsprechenden Hash-Schritts.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.
Verteilung	integer	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Das folgende Beispiel gibt die Anzahl der in einem Hash-Join für Abfrage 720 verwendeten Partitionen aus.

```
select query, slice, tbl, num_parts
from stl_hashjoin
where query=720 limit 10;
```

```
 query | slice | tbl | num_parts
-------+-------+-----+-----------
   720 |     0 | 243 |         1
   720 |     1 | 243 |         1
(2 rows)
```

STL_INSERT

Analysiert Einfügungs-Ausführungsschritte für Abfragen.

STL_INSERT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_INSERT enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschl ange und Zeit für die Ausführung mit einer Genauigkeit

Spaltenname	Datentyp	Beschreibung
		von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
tbl	integer	Tabellen-ID.
inserted_mega_value	character(1)	Diese Information ist nur für die interne Verwendung gedacht. Diese Information zeigt an, ob der angegebene Einfügeschritt einen großen Wert eingefügt hat. Ein großer Wert wird in mehreren Blöcken gespeichert. Die Blockgröße beträgt standardmäßig 1 MB, ein großer Wert ist in einer Standardeinstellung größer als 1 MB.

Beispielabfragen

Das folgende Beispiel gibt Einfügungs-Ausführungsschritte für die aktuelle Abfrage aus.

```
select slice, segment, step, tasknum, rows, tbl
from stl_insert
where query=pg_last_query_id();
```

```
 slice | segment | step | tasknum | rows  |  tbl
-------+---------+------+---------+-------+--------
     0 |       2 |    2 |      15 | 24958 | 100548
     1 |       2 |    2 |      15 | 25032 | 100548
(2 rows)
```

STL_LIMIT

Analysiert die Ausführungsschritte, die auftreten, wenn eine LIMIT-Klausel in einer SELECT-Abfrage verwendet wird.

STL_LIMIT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_LIMIT enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .

Spaltenname	Datentyp	Beschreibung
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Um eine Zeile in STL_LIMIT zu generieren, führt dieses Beispiel zuerst die folgende Abfrage gegen die Tabelle VENUE unter Verwendung der LIMIT-Klausel durch.

```
select * from venue
order by 1
limit 10;
```

```
 venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
       1 | Toyota Park                | Bridgeview      | IL         |          0
       2 | Columbus Crew Stadium      | Columbus        | OH         |          0
       3 | RFK Stadium                | Washington      | DC         |          0
       4 | CommunityAmerica Ballpark  | Kansas City     | KS         |          0
       5 | Gillette Stadium           | Foxborough      | MA         |      68756
       6 | New York Giants Stadium    | East Rutherford | NJ         |      80242
       7 | BMO Field                  | Toronto         | ON         |          0
       8 | The Home Depot Center      | Carson          | CA         |          0
       9 | Dick's Sporting Goods Park | Commerce City   | CO         |          0
      10 | Pizza Hut Park             | Frisco          | TX         |          0
(10 rows)
```

Führen Sie anschließend die folgende Abfrage durch, um die Abfrage-ID der letzten Abfrage zu finden, die gegen die Tabelle VENUE ausgeführt wurde.

```
select max(query)
from stl_query;
```

```
  max
--------
 127128
(1 row)
```

Optional können Sie die folgende Abfrage ausführen, um zu prüfen, ob die Abfrage-ID der vorher ausgeführten LIMIT-Abfrage entspricht.

```
select query, trim(querytxt)
from stl_query
where query=127128;
```

```
 query  |                  btrim
--------+-----------------------------------------
 127128 | select * from venue order by 1 limit 10;
(1 row)
```

Führen Sie abschließend die folgende Abfrage aus, um Informationen zu der LIMIT-Abfrage aus der Tabelle STL_LIMIT abzurufen.

```
select slice, segment, step, starttime, endtime, tasknum
from stl_limit
where query=127128
order by starttime, endtime;
```

```
  slice | segment | step |         starttime          |          endtime           |
 tasknum
 -------+---------+------+----------------------------+----------------------------
+---------
      1 |       1 |    3 | 2013-09-06 22:56:43.608114 | 2013-09-06 22:56:43.609383 |
   15
      0 |       1 |    3 | 2013-09-06 22:56:43.608708 | 2013-09-06 22:56:43.609521 |
   15
```

```
  10000 |       2 |    2 | 2013-09-06 22:56:43.612506 | 2013-09-06 22:56:43.612668 |
    0
(3 rows)
```

STL_LOAD_COMMITS

Gibt Informationen zur Nachverfolgung eines Datenladevorgangs bzw. zur Fehlerbehebung aus.

Diese Ansicht zeichnet den Fortschritt jeder Datendatei auf, während sie in eine Datenbanktabelle geladen wird.

STL_LOAD_COMMITS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_LOAD_COMMITS enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_LOAD_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Für diesen Eintrag geladener Slice.
Name	character(256)	Systemdefinierter Wert.

Spaltenname	Datentyp	Beschreibung
filename	character(256)	Name der verfolgten Datei.
byte_offset	integer	Diese Information ist nur für die interne Verwendung gedacht.
lines_scanned	integer	Anzahl der aus der Ladedatei gescannten Zeilen. Diese Zahl stimmt möglicherweise nicht mit der Anzahl der tatsächlich geladen Zeilen überein. So kann der Ladevorgang beispiels weise eine Anzahl ungültiger Datensätze scannen und tolerieren, je nach der MAXERROR-Option im COPY-Befehl.
Fehler	integer	Diese Information ist nur für die interne Verwendung gedacht.
curtime	timestamp	Zeitpunkt der letzten Aktualisierung dieses Eintrags.
status	integer	Diese Information ist nur für die interne Verwendung gedacht.
file_format	character(16)	Format der Ladedatei. Die möglichen Werte lauten wie folgt: • Avro • JSON • ORC • Parquet • Text
is_partial	integer	Wert, der bei true (1) angibt, dass die Eingabedatei während eines COPY-Vorgangs in Bereiche aufgeteilt wird. Wenn dieser Wert false (0) ist, wird die Eingabedatei nicht geteilt.
start_offset	bigint	Wenn die Eingabedatei während eines COPY-Vorgangs geteilt wird, gibt dieser Wert den Offset-Wert der Teilung (in Byte) an. Jede Dateiaufteilung wird als separater Datensatz mit dem entsprechenden start_offset-Wert protokolliert. Wenn die Datei nicht geteilt wird, ist dieser Wert 0.

Spaltenname	Datentyp	Beschreibung
copy_job_id	bigint	Die Kennung des Kopierauftrags. 0 gibt an, dass keine Auftragskennung vorhanden ist.

Beispielabfragen

Das folgende Beispiel gibt Details zur letzten COPY-Operation aus.

```
select query, trim(filename) as file, curtime as updated
from stl_load_commits
where query = pg_last_copy_id();

 query |               file               |          updated
-------+----------------------------------+----------------------------
 28554 | s3://dw-tickit/category_pipe.txt | 2013-11-01 17:14:52.648486
(1 row)
```

Die folgende Abfrage enthält Einträge für einen neuen Ladevorgang der Tabellen in der TICKIT-Datenbank:

```
select query, trim(filename), curtime
from stl_load_commits
where filename like '%tickit%' order by query;
```

```
 query |           btrim           |          curtime
-------+---------------------------+----------------------------
 22475 | tickit/allusers_pipe.txt  | 2013-02-08 20:58:23.274186
 22478 | tickit/venue_pipe.txt     | 2013-02-08 20:58:25.070604
 22480 | tickit/category_pipe.txt  | 2013-02-08 20:58:27.333472
 22482 | tickit/date2008_pipe.txt  | 2013-02-08 20:58:28.608305
 22485 | tickit/allevents_pipe.txt | 2013-02-08 20:58:29.99489
 22487 | tickit/listings_pipe.txt  | 2013-02-08 20:58:37.632939
 22593 | tickit/allusers_pipe.txt  | 2013-02-08 21:04:08.400491
 22596 | tickit/venue_pipe.txt     | 2013-02-08 21:04:10.056055
 22598 | tickit/category_pipe.txt  | 2013-02-08 21:04:11.465049
 22600 | tickit/date2008_pipe.txt  | 2013-02-08 21:04:12.461502
 22603 | tickit/allevents_pipe.txt | 2013-02-08 21:04:14.785124
 22605 | tickit/listings_pipe.txt  | 2013-02-08 21:04:20.170594
```

```
(12 rows)
```

Die Tatsache, dass ein Datensatz in die Protokolldatei für diese Systemansicht geschrieben wird, bedeutet nicht, dass der Ladevorgang erfolgreich im Rahmen seiner enthaltenden Transaktion bestätigt wurde. Fragen Sie zur Prüfung von Ladebestätigungen die Ansicht STL_UTILITYTEXT ab und suchen Sie nach dem COMMIT-Datensatz, der einer COPY-Transaktion entspricht. Beispielsweise verbindet diese Abfrage STL_LOAD_COMMITS und STL_QUERY auf der Grundlage einer Unterabfrage gegen STL_UTILITYTEXT:

```
select l.query,rtrim(l.filename),q.xid
from stl_load_commits l, stl_query q
where l.query=q.query
and exists
(select xid from stl_utilitytext where xid=q.xid and rtrim("text")='COMMIT');

 query |           rtrim           |  xid
-------+---------------------------+-------
 22600 | tickit/date2008_pipe.txt  | 68311
 22480 | tickit/category_pipe.txt  | 68066
  7508 | allusers_pipe.txt         | 23365
  7552 | category_pipe.txt         | 23415
  7576 | allevents_pipe.txt        | 23429
  7516 | venue_pipe.txt            | 23390
  7604 | listings_pipe.txt         | 23445
 22596 | tickit/venue_pipe.txt     | 68309
 22605 | tickit/listings_pipe.txt  | 68316
 22593 | tickit/allusers_pipe.txt  | 68305
 22485 | tickit/allevents_pipe.txt | 68071
  7561 | allevents_pipe.txt        | 23429
  7541 | category_pipe.txt         | 23415
  7558 | date2008_pipe.txt         | 23428
 22478 | tickit/venue_pipe.txt     | 68065
   526 | date2008_pipe.txt         |  2572
  7466 | allusers_pipe.txt         | 23365
 22482 | tickit/date2008_pipe.txt  | 68067
 22598 | tickit/category_pipe.txt  | 68310
 22603 | tickit/allevents_pipe.txt | 68315
 22475 | tickit/allusers_pipe.txt  | 68061
   547 | date2008_pipe.txt         |  2572
 22487 | tickit/listings_pipe.txt  | 68072
  7531 | venue_pipe.txt            | 23390
```

```
  7583 | listings_pipe.txt         | 23445
(25 rows)
```

In den folgenden Beispielen werden die Spaltenwerte is_partial und start_offset herausgestellt.

```
-- Single large file copy without scan range
SELECT count(*) FROM stl_load_commits WHERE query = pg_last_copy_id();
1

-- Single large uncompressed, delimited file copy with scan range
SELECT count(*) FROM stl_load_commits WHERE query = pg_last_copy_id();
16

-- Scan range offset logging in the file at 64MB boundary.
SELECT start_offset FROM stl_load_commits
WHERE query = pg_last_copy_id() ORDER BY start_offset;
0
67108864
134217728
201326592
268435456
335544320
402653184
469762048
536870912
603979776
671088640
738197504
805306368
872415232
939524096
1006632960
```

STL_LOAD_ERRORS

Zeigt die Datensätze für alle Amazon-Redshift-Ladefehler an.

STL_LOAD_ERRORS enthält einen Verlauf aller Amazon-Redshift-Ladefehler. Vgl. Ladefehlerreferenz für eine umfassende Liste möglicher Ladefehler und Erläuterungen dazu.

Fragen Sie STL_LOADERROR_DETAIL ab, um weitere Details zu erhalten, z. B. die genaue Datenzeile und Spalte, in der ein Parsingfehler aufgetreten ist, nachdem Sie STL_LOAD_ERRORS abgefragt haben, um allgemeine Informationen zum Fehler zu erhalten.

STL_LOAD_ERRORS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_LOAD_ERRORS enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_LOAD_ERROR_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
slice	integer	Slice, auf dem der Fehler aufgetreten ist.
tbl	integer	Tabellen-ID.
starttime	timestamp	Anfangszeit nach UTC des Ladevorgangs.
Sitzung	integer	Sitzungs-ID der Sitzung, die den Ladevorgang durchführt.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
filename	character(256)	Der vollständige Pfad zur Eingabedatei für den Ladevorgang.
line_number	bigint	Zeilennummer in der Ladedatei mit dem Fehler. Für COPY von JSON ist dies die Nummer der letzten Zeile des JSON-Objekts mit dem Fehler.

Spaltenname	Datentyp	Beschreibung
colname	character(127)	Feld mit dem Fehler.
type	character(10)	Datentyp des Feldes.
col_length	character(10)	Spaltenlänge, falls anwendbar. Dieses Feld ist gefüllt, wenn für den Datentyp eine Längenbegrenzung gilt. So enthält diese Spalte beispielsweise für eine Spalte mit dem Datentyp „character(3)“ den Wert „3“.
position	integer	Position des Fehlers in dem Feld.
raw_line	character(1024)	Roh-Ladedaten, die den Fehler enthalten. Multibyte-Zeichen in den Ladedaten werden durch einen Punkt ersetzt.
raw_field_value	char(1024)	Der Wert des Feldes „colname“ vor dem Parsing, der zu dem Ladefehler geführt hat.
err_code	integer	Fehlercode.
err_reason	character(100)	Erläuterung zu dem Fehler.
is_partial	integer	Wert, der bei true (1) angibt, dass die Eingabedatei während eines COPY-Vorgangs in Bereiche aufgeteilt wird. Wenn dieser Wert false (0) ist, wird die Eingabedatei nicht geteilt.
start_offset	bigint	Wenn die Eingabedatei während eines COPY-Vorgangs geteilt wird, gibt dieser Wert den Offset-Wert der Teilung (in Byte) an. Wenn die Zeilennummer in der Datei unbekannt ist, lautet die Zeilennummer -1. Wenn die Datei nicht geteilt wird, ist dieser Wert 0.
copy_job_id	bigint	Die Kennung des Kopierauftrags. `0` gibt an, dass keine Auftragskennung vorhanden ist.

Beispielabfragen

Die folgende Abfrage verbindet STL_LOAD_ERRORS mit STL_LOADERROR_DETAIL zur Anzeige der Details der Fehler, die während des letzten Ladevorgangs aufgetreten sind.

```
select d.query, substring(d.filename,14,20),
d.line_number as line,
substring(d.value,1,16) as value,
substring(le.err_reason,1,48) as err_reason
from stl_loaderror_detail d, stl_load_errors le
where d.query = le.query
and d.query = pg_last_copy_id();

 query |    substring      | line |  value   |             err_reason
-------+-------------------+------+----------+----------------------------
    558| allusers_pipe.txt |  251 | 251      | String contains invalid or
                                              unsupported UTF8 code
    558| allusers_pipe.txt |  251 | ZRU29FGR | String contains invalid or
                                              unsupported UTF8 code
    558| allusers_pipe.txt |  251 | Kaitlin  | String contains invalid or
                                              unsupported UTF8 code
    558| allusers_pipe.txt |  251 | Walter   | String contains invalid or
                                              unsupported UTF8 code
```

Das folgende Beispiel verwendet STL_LOAD_ERRORS mit STV_TBL_PERM zur Erstellung einer neuen Ansicht und nutzt diese dann, um festzustellen, welche Fehler beim Laden der Daten in die Tabelle EVENT aufgetreten sind:

```
create view loadview as
(select distinct tbl, trim(name) as table_name, query, starttime,
trim(filename) as input, line_number, colname, err_code,
trim(err_reason) as reason
from stl_load_errors sl, stv_tbl_perm sp
where sl.tbl = sp.id);
```

Anschließend gibt die folgende Abfrage den letzten Fehler aus, der beim Laden der Tabelle EVENT aufgetreten ist:

```
select table_name, query, line_number, colname, starttime,
trim(reason) as error
from loadview
```

```
where table_name ='event'
order by line_number limit 1;
```

Die Abfrage gibt den letzten Fehler aus, der in der Tabelle EVENT aufgetreten ist. Wenn keine Ladefehler aufgetreten sind, gibt die Abfrage null Zeilen aus. In diesem Beispiel gibt die Abfrage einen einzigen Fehler aus:

```
 table_name | query | line_number | colname | error | starttime
------+-----+----+----+-------------------------------------------------------
+----------------------
event | 309 |  0 |  5 | Error in Timestamp value or format [%Y-%m-%d %H:%M:%S] |
 2014-04-22 15:12:44

(1 row)
```

In Fällen, in denen der COPY-Befehl große, unkomprimierte, durch Text getrennte Dateidaten automatisch aufteilt, um Parallelität zu ermöglichen, zeigen die Spalten line_number, is_partial und start_offset Informationen zu Aufteilungen. (Die Zeilennummer kann unbekannt sein, wenn die Zeilennummer aus der Originaldatei nicht verfügbar ist.)

```
--scan ranges information
SELECT line_number, POSITION, btrim(raw_line), btrim(raw_field_value),
btrim(err_reason), is_partial, start_offset FROM stl_load_errors
WHERE query = pg_last_copy_id();

--result
-1,51,"1008771|13463413|463414|2|28.00|38520.72|0.06|0.07|NO|1998-08-30|1998-09-25|
1998-09-04|TAKE BACK RETURN|RAIL|ans cajole sly","NO","Char length exceeds DDL
 length",1,67108864
```

STL_LOADERROR_DETAIL

Zeigt ein Protokoll von Datenparsingfehlern an, die bei der Verwendung eines COPY-Befehls zum Laden von Tabellen aufgetreten sind. Um Festplattenspeicherplatz zu sparen, werden maximal 20 Fehler pro Knotenslice für jeden Ladevorgang protokolliert.

Ein Parsingfehler tritt auf, wenn Amazon Redshift ein Feld in einer Datenzeile beim Laden in eine Tabelle nicht parsen kann. Zum Beispiel: Wenn eine Tabellenspalte den Datentyp „Ganzzahl“ erwartet, die Datendatei in dem betreffenden Feld jedoch eine Buchstabenzeichenfolge enthält, führt dies zu einem Parsingfehler.

Fragen Sie STL_LOADERROR_DETAIL ab, um weitere Details zu erhalten, etwa die genaue Datenzeile und Spalte, in der ein Parsingfehler aufgetreten ist, nachdem Sie STL_LOAD_ERRORS abgefragt haben, um allgemeine Informationen zu dem Fehler zu erhalten.

Die Ansicht STL_LOADERROR_DETAIL enthält alle Datenspalten, einschließlich aller Spalten vor der Spalte, in der der Parsingfehler aufgetreten ist. Verwenden Sie das Feld VALUE, um den Datenwert zu sehen, der in dieser Spalte tatsächlich geparst wurde, einschließlich der bis zu dem Fehler korrekt geparsten Spalten.

Diese Ansicht ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_LOADERROR_DETAIL enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_LOAD_ERROR_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
slice	integer	Slice, auf dem der Fehler aufgetreten ist.
Sitzung	integer	Sitzungs-ID der Sitzung, die den Ladevorgang durchführt.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.

Spaltenname	Datentyp	Beschreibung
filename	character(256)	Der vollständige Pfad zur Eingabedatei für den Ladevorgang.
line_number	bigint	Zeilennummer in der Ladedatei mit dem Fehler.
field	integer	Feld mit dem Fehler.
colname	character(1024)	Spaltenname.
Wert	character(1024)	geparster Datenwert des Feldes. (Kann verkürzt werden.) Multibyte-Zeichen in den Ladedaten werden durch einen Punkt ersetzt.
is_null	integer	Ob der Parsingwert Null ist oder nicht.
type	character(10)	Datentyp des Feldes.
col_length	character(10)	Spaltenlänge, falls anwendbar. Dieses Feld ist gefüllt, wenn für den Datentyp eine Längenbegrenzung gilt. So enthält diese Spalte beispielsweise für eine Spalte mit dem Datentyp „character(3)“ den Wert „3“.

Beispielabfrage

Die folgende Abfrage verbindet STL_LOAD_ERRORS mit STL_LOADERROR_DETAIL zur Anzeige der Details eines Parsingfehlers, der beim Laden der Tabelle EVENT (Tabellen-ID 100133) aufgetreten ist:

```
select d.query, d.line_number, d.value,
le.raw_line, le.err_reason
from stl_loaderror_detail d, stl_load_errors le
where
d.query = le.query
and tbl = 100133;
```

Die folgende Beispielausgabe zeigt die erfolgreich geladenen Spalten, einschließlich der Spalte mit dem Fehler. In diesem Beispiel wurden zwei Spalten erfolgreich geladen, bevor der Parsingfehler in der dritten Spalte auftrat, wobei eine Zeichenfolge für ein Feld, in dem eine Ganzzahl erwartet wurde, nicht korrekt geparst werden konnte. Da in dem Feld eine Ganzzahl erwartet wurde, ergab das Parsing der Zeichenfolge „aaa“, d. h. nicht initialisierte Daten, Null, und es wurde ein Parsingfehler ausgegeben. Die Ausgabe zeigt den Rohwert, den geparsten Wert und den Grund für den Fehler:

```
query  | line_number | value | raw_line | err_reason
-------+-------------+-------+----------+----------------
4      |      3      |  1201 |  1201    | Invalid digit
4      |      3      |   126 |   126    | Invalid digit
4      |      3      |       |   aaa    | Invalid digit
(3 rows)
```

Wenn eine Abfrage STL_LOAD_ERRORS und STL_LOADERROR_DETAIL verbindet, zeigt Sie eine Fehlerursache für jede Spalte in der Datenzeile an; dies bedeutet nur, dass in der betreffenden Zeile ein Fehler aufgetreten ist. Die letzte Zeile in den Ergebnissen enthält die eigentliche Spalte, in der der Parsingfehler aufgetreten ist.

STL_MERGE

Analysiert Zusammenführungs-Ausführungsschritte für Abfragen. Diese Schritte treten auf, wenn die Ergebnisse paralleler Operationen (etwa Sortierungen oder Joins) für die weitere Verarbeitung zusammengeführt werden.

STL_MERGE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_MERGE enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.

Beispielabfragen

Das folgende Beispiele gibt 10 Ergebnisse zu Zusammenführungsausführungen aus.

```
select query, step, starttime, endtime, tasknum, rows
```

```
from stl_merge
limit 10;
```

```
 query | step |       starttime     |        endtime      | tasknum | rows
-------+------+---------------------+---------------------+---------+------
     9 |    0 | 2013-08-12 20:08:14 | 2013-08-12 20:08:14 |       0 |    0
    12 |    0 | 2013-08-12 20:09:10 | 2013-08-12 20:09:10 |       0 |    0
    15 |    0 | 2013-08-12 20:10:24 | 2013-08-12 20:10:24 |       0 |    0
    20 |    0 | 2013-08-12 20:11:27 | 2013-08-12 20:11:27 |       0 |    0
    26 |    0 | 2013-08-12 20:12:28 | 2013-08-12 20:12:28 |       0 |    0
    32 |    0 | 2013-08-12 20:14:33 | 2013-08-12 20:14:33 |       0 |    0
    38 |    0 | 2013-08-12 20:16:43 | 2013-08-12 20:16:43 |       0 |    0
    44 |    0 | 2013-08-12 20:17:05 | 2013-08-12 20:17:05 |       0 |    0
    50 |    0 | 2013-08-12 20:18:48 | 2013-08-12 20:18:48 |       0 |    0
    56 |    0 | 2013-08-12 20:20:48 | 2013-08-12 20:20:48 |       0 |    0
(10 rows)
```

STL_MERGEJOIN

Analysiert Zusammenführungs-/Verbindungs-Ausführungsschritte für Abfragen.

STL_MERGEJOIN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_MERGEJOIN enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
tbl	integer	Tabellen-ID. Dies ist die ID der inneren Tabelle, die für den Zusammenführungs-/Verbindungsvorgang verwendet wurde.

Spaltenname	Datentyp	Beschreibung
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Das folgende Beispiel gibt Zusammenführungs-/Verbindungs-Ausführungsschritte für die aktuelle Abfrage aus.

```
select sum(s.qtysold), e.eventname
from event e, listing l, sales s
where e.eventid=l.eventid
and l.listid= s.listid
group by e.eventname;

select * from stl_mergejoin where query=pg_last_query_id();
```

```
 userid | query | slice | segment | step |         starttime   |          endtime    |
 tasknum | rows | tbl
--------+-------+-------+---------+------+---------------------+---------------------
+---------+------+-----
    100 | 27399 |     3 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43428 | 240
    100 | 27399 |     0 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43159 | 240
    100 | 27399 |     2 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |42778 | 240
    100 | 27399 |     1 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43091 | 240
```

STL_MV_STATE

Die Ansicht STL_MV_STATE enthält eine Zeile für jeden Statusübergang einer materialisierten Ansicht.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

STL_MV_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_MV_STATE zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	bigint	Die ID des Benutzers, der das Ereignis erstellt hat.
starttime	timestamp	Die Anfangszeit des Build.
xid	bigint	Die Transaktions-ID des Ereignisses.
event_desc	char(500)	Das Ereignis, das die Statusänderung veranlasst hat. Zu den Beispielwerten gehören: • Spaltentyp wurde geändert • Spalte wurde gelöscht • Spalte wurde umbenannt • Schemaname wurde geändert • Konvertierung kleiner Tabellen • TRUNCATE • Vacuum Beachten Sie, dass es andere mögliche Werte für diese Spalte gibt.
db_name	char(128)	Die Datenbank, die die materialisierte Ansicht enthält.

Spaltenname	Datentyp	Beschreibung
base_table_schema	char(128)	Das Schema der Basistabelle.
base_table_name	char(128)	Der Name der Basistabelle.
mv_schema	char(128)	Das Schema der materialisierten Ansicht.
mv_name	char(128)	Der Name der materialisierten Ansicht.
state	character(32)	Der geänderte Status der materialisierten Ansicht wie folgt: • Erneutes Berechnen • Nicht aktualisierbar

Die folgende Tabelle zeigt Beispielkombinationen von `event_desc` und `state`.

```
          event_desc     |     state
-------------------------+---------------
 TRUNCATE                | Recompute
 TRUNCATE                | Recompute
 Small table conversion  | Recompute
 Vacuum                  | Recompute
 Column was renamed      | Unrefreshable
 Column was dropped      | Unrefreshable
 Table was renamed       | Unrefreshable
 Column type was changed | Unrefreshable
 Schema name was changed | Unrefreshable
```

Beispielabfrage

Führen Sie die folgende Abfrage aus, um das Protokoll der Statusübergänge materialisierter Ansichten anzuzeigen.

```
select * from stl_mv_state;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
 userid |         starttime          | xid  |            event_desc       | db_name |
 base_table_schema   |   base_table_name    |      mv_schema       | mv_name       |
  state
--------+----------------------------+------+------------------------------+---------
+---------------------+---------------------+---------------------+--------------
+---------------
    138 | 2020-02-14 02:21:25.578885 | 5180 | TRUNCATE                     | dev     |
 public              | mv_base_table        | public              | mv_test       |
 Recompute
    138 | 2020-02-14 02:21:56.846774 | 5275 | Column was dropped           | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    100 | 2020-02-13 22:09:53.041228 | 1794 | Column was renamed           | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
      1 | 2020-02-13 22:10:23.630914 | 1893 | ALTER TABLE ALTER SORTKEY    | dev     |
 public              | mv_base_table_sorted | public              | mv_test       |
 Recompute
      1 | 2020-02-17 22:57:22.497989 | 8455 | ALTER TABLE ALTER DISTSTYLE | dev     |
 public              | mv_base_table        | public              | mv_test       |
 Recompute
    173 | 2020-02-17 22:57:23.591434 | 8504 | Table was renamed            | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    173 | 2020-02-17 22:57:27.229423 | 8592 | Column type was changed      | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    197 | 2020-02-17 22:59:06.212569 | 9668 | TRUNCATE                     | dev     |
 schemaf796e415850f4f | mv_base_table        | schemaf796e415850f4f | mv_test       |
 Recompute
    138 | 2020-02-14 02:21:55.705655 | 5226 | Column was renamed           | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
      1 | 2020-02-14 02:22:26.292434 | 5325 | ALTER TABLE ALTER SORTKEY    | dev     |
 public              | mv_base_table_sorted | public              | mv_test       |
 Recompute
```

STL_NESTLOOP

Analysiert Nested-Loop-Ausführungsschritte für Abfragen.

STL_NESTLOOP ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_NESTLOOP enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von

Spaltenname	Datentyp	Beschreibung
		6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
tbl	integer	Tabellen-ID.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Da die folgende Abfrage die Tabelle CATEGORY nicht verbindet, führt sie zu einem partiellen cartesischen Produkt; dies wird nicht empfohlen. Sie wird hier nur zur Illustration einer eingebetteten Schleife gezeigt.

```
select count(event.eventname), event.eventname, category.catname, date.caldate
from event, category, date
where event.dateid = date.dateid
group by event.eventname, category.catname, date.caldate;
```

Die folgende Abfrage zeigt die Ergebnisse der vorherigen Abfrage in der Ansicht STL_NESTLOOP.

```
select query, slice, segment as seg, step,
datediff(msec, starttime, endtime) as duration, tasknum, rows, tbl
from stl_nestloop
where query = pg_last_query_id();
```

```
 query | slice | seg | step | duration | tasknum | rows  | tbl
-------+-------+-----+------+----------+---------+-------+-----
  6028 |     0 |   4 |    5 |       41 |      22 | 24277 | 240
  6028 |     1 |   4 |    5 |       26 |      23 | 24189 | 240
  6028 |     3 |   4 |    5 |       25 |      23 | 24376 | 240
  6028 |     2 |   4 |    5 |       54 |      22 | 23936 | 240
```

STL_PARSE

Analysiert die Abfrageschritte zum Parsing von Zeichenfolgen in binäre Werte für den Ladevorgang.

STL_PARSE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_PARSE enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.

Spaltenname	Datentyp	Beschreibung
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.

Beispielabfragen

Das folgende Beispiel gibt alle Abfrageschrittergebnisse für Slice 1 und Segment 0 aus, bei denen das Parsing von Zeichenfolgen in binäre Werte durchgeführt wurde.

```
select query, step, starttime, endtime, tasknum, rows
from stl_parse
where slice=1 and segment=0;
```

```
 query | step |     starttime       |        endtime      | tasknum |  rows
-------+------+---------------------+---------------------+---------+--------
   669 |    1 | 2013-08-12 22:35:13 | 2013-08-12 22:35:17 |      32 | 192497
   696 |    1 | 2013-08-12 22:35:49 | 2013-08-12 22:35:49 |      32 |      0
   525 |    1 | 2013-08-12 22:32:03 | 2013-08-12 22:32:03 |      13 |  49990
   585 |    1 | 2013-08-12 22:33:18 | 2013-08-12 22:33:19 |      13 |    202
```

```
   621 |    1 | 2013-08-12 22:34:03 | 2013-08-12 22:34:03 |      27 |    365
   651 |    1 | 2013-08-12 22:34:47 | 2013-08-12 22:34:53 |      35 | 192497
   590 |    1 | 2013-08-12 22:33:28 | 2013-08-12 22:33:28 |      19 |      0
   599 |    1 | 2013-08-12 22:33:39 | 2013-08-12 22:33:39 |      31 |     11
   675 |    1 | 2013-08-12 22:35:26 | 2013-08-12 22:35:27 |      38 |   3766
   567 |    1 | 2013-08-12 22:32:47 | 2013-08-12 22:32:48 |      23 |  49990
   630 |    1 | 2013-08-12 22:34:17 | 2013-08-12 22:34:17 |      36 |      0
   572 |    1 | 2013-08-12 22:33:04 | 2013-08-12 22:33:04 |      29 |      0
   645 |    1 | 2013-08-12 22:34:37 | 2013-08-12 22:34:38 |      29 |   8798
   604 |    1 | 2013-08-12 22:33:47 | 2013-08-12 22:33:47 |      37 |      0
(14 rows)
```

STL_PLAN_INFO

Verwenden Sie die Ansicht STL_PLAN_INFO zur Betrachtung der EXPLAIN-Ausgabe für eine Abfrage als Satz von Zeilen. Diese ist eine alternative Betrachtungsweise von Abfrageplänen.

STL_PLAN_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_PLAN_INFO enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.

Spaltenname	Datentyp	Beschreibung
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
nodeid	integer	Plan-Knoten-ID, wo ein Knoten mit einem oder mehreren Schritten in der Ausführung der Abfrage verwunden ist.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Zahl, mit der der Abfrageschritt identifiziert wird.
locus	integer	Ort, an dem der Schritt ausgeführt wird. 0, wenn auf einem Datenverarbeitungsknoten, und 1, wenn auf dem Führungsknoten.
plannode	integer	Enumerationswert des Planknotens. Vgl. die folgende Tabelle für Enum-Werte für Plannode. (Die Spalte PLANNODE in STL_EXPLAIN enthält den Planknotentext.)
startupcost	double precision	Die geschätzten relativen Kosten für die Rückgabe der ersten Zeile für diesen Schritt.
totalcost	double precision	Die geschätzten relativen Kosten für die Ausführung des Schrittes.
rows	bigint	Die geschätzte Anzahl der Zeilen, die durch den Schritt erstellt werden.
bytes	bigint	Die geschätzte Anzahl der Bytes, die durch den Schritt erstellt werden.

Beispielabfragen

Die folgenden Beispiele vergleichen die Abfragepläne für eine einfache SELECT-Abfrage mit dem Befehl EXPLAIN und Abfrage der Ansicht STL_PLAN_INFO.

```
explain select * from category;
```

```
QUERY PLAN
------------------------------------------------------------
XN Seq Scan on category (cost=0.00..0.11 rows=11 width=49)
(1 row)

select * from category;
catid | catgroup | catname | catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports | MLB | Major League Baseball
3 | Sports | NFL | National Football League
5 | Sports | MLS | Major League Soccer
...

select * from stl_plan_info where query=256;

query | nodeid | segment | step | locus | plannode | startupcost | totalcost
| rows | bytes
-------+--------+---------+------+-------+----------+-------------+-----------+------
+-------
256 | 1 | 0 | 1 | 0 | 104 | 0 | 0.11 | 11 | 539
256 | 1 | 0 | 0 | 0 | 104 | 0 | 0.11 | 11 | 539
(2 rows)
```

In diesem Beispiel bezieht sich PLANNODE 104 auf den sequenziellen Scan der Tabelle CATEGORY.

```
select distinct eventname from event order by 1;

eventname
------------------------------------------------------------------------------
.38 Special
3 Doors Down
70s Soul Jam
A Bronx Tale
...

explain select distinct eventname from event order by 1;

QUERY PLAN
------------------------------------------------------------------------------------------
XN Merge (cost=1000000000136.38..1000000000137.82 rows=576 width=17)
Merge Key: eventname
-> XN Network (cost=1000000000136.38..1000000000137.82 rows=576
```

```
width=17)
Send to leader
-> XN Sort (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Sort Key: eventname
-> XN Unique (cost=0.00..109.98 rows=576 width=17)
-> XN Seq Scan on event (cost=0.00..87.98 rows=8798
width=17)
(8 rows)

select * from stl_plan_info where query=240 order by nodeid desc;

query | nodeid | segment | step | locus | plannode | startupcost |
totalcost | rows | bytes
-------+--------+---------+------+-------+----------+-----------------
+------------------+------+--------
240 | 5 | 0 | 0 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 5 | 0 | 1 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 4 | 0 | 2 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 0 | 3 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 0 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 1 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 3 | 1 | 2 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 3 | 2 | 0 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 2 | 2 | 1 | 0 | 123 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 1 | 3 | 0 | 0 | 122 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
(10 rows)
```

STL_PROJECT

Enthält Zeilen für Abfrageschritte, die zur Evaluierung von Ausdrücken verwendet werden.

STL_PROJECT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_PROJECT enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-

Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.

Spaltenname	Datentyp	Beschreibung
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Das folgende Beispiel gibt alle Zeilen für Abfrageschritte aus, die zur Evaluierung von Ausdrücken für Slice 0 und Segment 1 verwendet wurden.

```
select query, step, starttime, endtime, tasknum, rows
from stl_project
where slice=0 and segment=1;
```

```
 query  | step |         starttime   |          endtime    | tasknum | rows
--------+------+---------------------+---------------------+---------+------
  86399 |    2 | 2013-08-29 22:01:21 | 2013-08-29 22:01:21 |      25 |   -1
  86399 |    3 | 2013-08-29 22:01:21 | 2013-08-29 22:01:21 |      25 |   -1
    719 |    1 | 2013-08-12 22:38:33 | 2013-08-12 22:38:33 |       7 |   -1
  86383 |    1 | 2013-08-29 21:58:35 | 2013-08-29 21:58:35 |       7 |   -1
    714 |    1 | 2013-08-12 22:38:17 | 2013-08-12 22:38:17 |       2 |   -1
  86375 |    1 | 2013-08-29 21:57:59 | 2013-08-29 21:57:59 |       2 |   -1
  86397 |    2 | 2013-08-29 22:01:20 | 2013-08-29 22:01:20 |      19 |   -1
    627 |    1 | 2013-08-12 22:34:13 | 2013-08-12 22:34:13 |      34 |   -1
  86326 |    2 | 2013-08-29 21:45:28 | 2013-08-29 21:45:28 |      34 |   -1
  86326 |    3 | 2013-08-29 21:45:28 | 2013-08-29 21:45:28 |      34 |   -1
  86325 |    2 | 2013-08-29 21:45:27 | 2013-08-29 21:45:27 |      28 |   -1
  86371 |    1 | 2013-08-29 21:57:42 | 2013-08-29 21:57:42 |       4 |   -1
 111100 |    2 | 2013-09-03 19:04:45 | 2013-09-03 19:04:45 |      12 |   -1
    704 |    2 | 2013-08-12 22:36:34 | 2013-08-12 22:36:34 |      37 |   -1
    649 |    2 | 2013-08-12 22:34:47 | 2013-08-12 22:34:47 |      38 |   -1
    649 |    3 | 2013-08-12 22:34:47 | 2013-08-12 22:34:47 |      38 |   -1
    632 |    2 | 2013-08-12 22:34:22 | 2013-08-12 22:34:22 |      13 |   -1
    705 |    2 | 2013-08-12 22:36:48 | 2013-08-12 22:36:49 |      13 |   -1
    705 |    3 | 2013-08-12 22:36:48 | 2013-08-12 22:36:49 |      13 |   -1
      3 |    1 | 2013-08-12 20:07:40 | 2013-08-12 20:07:40 |       3 |   -1
  86373 |    1 | 2013-08-29 21:57:58 | 2013-08-29 21:57:58 |       3 |   -1
 107976 |    1 | 2013-09-03 04:05:12 | 2013-09-03 04:05:12 |       3 |   -1
  86381 |    1 | 2013-08-29 21:58:35 | 2013-08-29 21:58:35 |       8 |   -1
  86396 |    1 | 2013-08-29 22:01:20 | 2013-08-29 22:01:20 |      15 |   -1
```

```
   711 |    1 | 2013-08-12 22:37:10 | 2013-08-12 22:37:10 |      20 |   -1
 86324 |    1 | 2013-08-29 21:45:27 | 2013-08-29 21:45:27 |      24 |   -1
(26 rows)
```

STL_QUERY

Gibt Ausführungsinformationen zu einer Datenbankabfrage aus.

> **Note**
>
> Die Ansichten STL_QUERY und STL_QUERYTEXT enthalten nur Informationen zu Abfragen, nicht zu weiteren Utility- und DDL-Befehlen. Für eine Liste und Informationen zu allen von Amazon Redshift ausgeführten Anweisungen können Sie auch die Ansichten STL_DDLTEXT und STL_UTILITYTEXT abfragen. Für eine vollständige Liste aller von Amazon Redshift ausgeführten Anweisungen können Sie die Ansicht SVL_STATEMENTTEXT abfragen.

STL_QUERY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasi

Spaltenname	Datentyp	Beschreibung
		ert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist der Wert in diesem Feld `default`.
xid	bigint	Transaktions-ID.
pid	integer	Prozess-ID. Normalerweise werden alle Abfragen in einer Sitzung in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen. Im Anschluss an bestimmte interne Ereignisse startet Amazon Redshift möglicherweise eine aktive Sitzung neu und weist eine neue PID zu. Weitere Informationen finden Sie unter STL_RESTARTED_SESSIONS.
Datenbank	character(32)	Der Name der Datenbank, mit der der Benutzer verbunden war, als die Abfrage ausgegeben wurde.
querytxt	character(4000)	Tatsächlicher Abfragetext der Abfrage.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .

Spaltenname	Datentyp	Beschreibung
aborted	integer	Wenn eine Abfrage vom System oder einem Benutzer beendet wurde, enthält diese Spalte den Wert **1**. Wenn die Abfrage abgeschlossen ist (einschließlich der Ausgabe der Ergebnisse an den Client), enthält diese Spalte den Wert **0**. Wenn ein Client die Verbindung vor dem Erhalt der Ergebnisse trennt, wird die Abfrage als abgebrochen markiert (**1**), auch wenn sie im Backend erfolgreich abgeschlossen wurde.
insert_pristine	integer	Ob Schreibabfragen ausgeführt werden können/konnten, während die aktuelle Abfrage läuft/lief. 1 = keine Schreibabfragen erlaubt. 0 = Schreibabfragen erlaubt. Diese Spalte dient zur Verwendung beim Debugging.
concurrency_scaling_status	integer	Gibt an, ob die Abfrage auf dem Haupt-Cluster oder einem Nebenläufigkeitsskalierungs-Cluster ausgeführt wurde. Die möglichen Werte lauten wie folgt: 0 – Wurde auf dem Haupt-Cluster ausgeführt 1 – Wurde auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt Größer als 1 – Wurde auf dem Haupt-Cluster ausgeführt

Beispielabfragen

Die folgende Abfrage führt die fünf letzten Abfragen auf.

```
select query, trim(querytxt) as sqlquery
from stl_query
order by query desc limit 5;

query |                                  sqlquery
------+-------------------------------------------------
129 | select query, trim(querytxt) from stl_query order by query;
```

```
128 | select node from stv_disk_read_speeds;
127 | select system_status from stv_gui_status
126 | select * from systable_topology order by slice
125 | load global dict registry
(5 rows)
```

Die folgende Abfrage gibt die verstrichene Zeit (in absteigender Reihenfolge) für Abfragen aus, die am 15. Februar 2013 ausgeführt wurden.

```
select query, datediff(seconds, starttime, endtime),
trim(querytxt) as sqlquery
from stl_query
where starttime >= '2013-02-15 00:00' and endtime < '2013-02-16 00:00'
order by date_diff desc;

 query | date_diff |  sqlquery
-------+-----------+--------------------------------------------
 55    |       119 | padb_fetch_sample: select count(*) from category
121    |         9 | select * from svl_query_summary;
181    |         6 | select * from svl_query_summary where query in(179,178);
172    |         5 | select * from svl_query_summary where query=148;
...
(189 rows)
```

Die folgende Abfrage zeigt die Warteschlangenzeit und Ausführungszeit für Abfragen an. Abfragen mit `concurrency_scaling_status = 1` wurden auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt. Alle anderen Abfragen wurden auf dem Haupt-Cluster ausgeführt.

```
SELECT w.service_class AS queue
     , q.concurrency_scaling_status
     , COUNT( * ) AS queries
     , SUM( q.aborted )  AS aborted
     , SUM( ROUND( total_queue_time::NUMERIC / 1000000,2 ) ) AS queue_secs
     , SUM( ROUND( total_exec_time::NUMERIC / 1000000,2 ) )  AS exec_secs
FROM stl_query q
     JOIN stl_wlm_query w
          USING (userid,query)
WHERE q.userid > 1
  AND service_class > 5
  AND q.starttime > '2019-03-01 16:38:00'
  AND q.endtime   < '2019-03-01 17:40:00'
GROUP BY 1,2
```

```
ORDER BY 1,2;
```

STL_QUERY_METRICS

Enthält Metrikinformationen, etwa die Anzahl der verarbeiteten Zeilen, die CPU-Nutzung, Input/Output- und Festplattennutzung für aktive Abfragen, die in benutzerdefinierten Abfragewarteschlangen (Service-Klassen) abgeschlossen wurden. Zur Anzeige von Metriken für derzeit ausgeführte aktive Abfragen vgl. die Systemansicht STV_QUERY_METRICS.

Die Abfragemetriken werden in Intervallen von einer Sekunde erfasst. Daher können verschiedene Ausführungen einer Abfrage leicht abweichende Zeiten ergeben. Abfragesegmente, die in weniger als einer Sekunde ausgeführt werden, werden möglicherweise nicht aufgezeichnet.

STL_QUERY_METRICS verfolgt und aggregiert Metriken auf Abfrage-, Segment- und Schrittebene. Informationen zu Abfragesegmenten und Schritten finden Sie unter Workflow der Abfrageplanung und -ausführung. Viele Metriken (wie etwa `max_rows`, `cpu_time` u. dgl.) werden über Knoten-Slices hinweg summiert. Weitere Informationen über Knoten-Slices finden Sie unter Architektur des Data Warehouse-Systems.

Um die Ebene festzustellen, auf der die Zeile die Metriken meldet, prüfen Sie die Spalten `segment` und `step_type`.

- Wenn `segment` und `step_type` den Wert `-1` haben, meldet die Zeile die Metriken auf Abfrageebene.
- Wenn `segment` nicht den Wert `-1` und `step_type` den Wert `-1` hat, meldet die Zeile die Metriken auf Segmentebene.
- Wenn `segment` und `step_type` nicht den Wert `-1` haben, meldet die Zeile die Metriken auf Schrittebene.

Die Ansicht SVL_QUERY_METRICS und die Ansicht SVL_QUERY_METRICS_SUMMARY aggregieren die Daten in dieser Ansicht und präsentieren die Informationen in zugänglicherer Form.

STL_QUERY_METRICS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert,

dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der die Abfrage ausgeführt hat, die den Eintrag generierte.
service_class	integer	ID für die Service-Klasse. Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Nur für benutzerdefinierte Abfragen werden Metriken gemeldet.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
segment	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt. Wenn der Segmentwert -1 ist, werden Segment-Metrikwerte zur Abfrageebene aufgerollt.
step_type	integer	Typ des Schritts, der ausgeführt wurde. Eine detaillierte Beschreibung der Schritttypen finden Sie unter Schritttypen.
starttime	timestamp	Uhrzeit in UTC, zu der die Ausführung der Abfrage begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: `2009-06-12 11:29:19.131358` .
slices	integer	Anzahl der Slices für den Cluster.
max_rows	bigint	Höchstzahl der für einen Schritt ausgegebenen Zeilen, über alle Slices hinweg aggregiert.
rows	bigint	Anzahl der von einem Schritt verarbeiteten Zeilen.
max_cpu_time	bigint	Die maximale verwendete CPU-Zeit, in Mikrosekunden. Auf Segmentebene die maximale von dem Segment verwendet

Spaltenname	Datentyp	Beschreibung
		e CPU-Zeit, über alle Slices. Auf Abfrageebene die maximale von einem Abfragesegment verwendete CPU-Zeit.
cpu_time	bigint	Die verwendete CPU-Zeit, in Mikrosekunden. Auf Segmenteb ene die gesamte von dem Segment verwendete CPU-Zeit, über alle Slices. Auf Abfrageebene die gesamte CPU-Zeit für die Abfrage, über alle Slices und Segmente.
max_block s_read	bigint	Die Höchstzahl der von dem Segment gelesenen 1 MB-Blöcke, aggregiert über alle Slices. Auf Segmentebene die Höchstzahl der für das Segment gelesenen 1 MB-Blöcke, über alle Slices. Auf Abfrageebene die Höchstzahl der von einem Abfragesegment gelesenen 1 MB-Blöcke.
blocks_read	bigint	Anzahl der von der Abfrage oder dem Segment gelesenen 1 MB-Blöcke.
max_run_time	bigint	Die maximale verstrichene Zeit für ein Segment, in Mikroseku nden. Auf Segmentebene die maximale Laufzeit für das Segment, über alle Slices. Auf Abfrageebene die maximale Laufzeit für ein Abfragesegment.
run_time	bigint	Die gesamte Laufzeit, summiert über alle Slices. Die Laufzeit enthält nicht die Wartezeit. Auf Segmentebene die Laufzeit für das Segment, summiert über alle Slices. Auf Abfrageebene die Laufzeit für die Abfrage, summiert über alle Slices und Segmente. Da dieser Wert eine Summe ist, ist die Laufzeit nicht mit der Abfrageau sführungszeit verbunden.
max_block s_to_disk	bigint	Der maximale Festplattenspeicherplatz, der zum Schreiben von Zwischenergebnissen verwendet wird, in MB-Blöcke n. Auf Segmentebene der maximale von dem Segment verwendete Festplattenspeicherplatz, über alle Slices. Auf Abfrageebene der maximale von einem Abfragesegment verwendete Festplattenspeicherplatz.

Spaltenname	Datentyp	Beschreibung
blocks_to_disk	bigint	Der Festplattenspeicherplatz, der von einer Abfrage oder einem Segment zum Schreiben von Zwischenergebnissen verwendet wird, in MB-Blöcken.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
max_query_scan_size	bigint	Die maximale von einer Abfrage gescannte Datenmenge, in MB. Auf Segmentebene die maximale von dem Segment gescannte Datenmenge, über alle Slices. Auf Abfrageebene die maximale von einem Abfragesegment gescannte Datenmenge.
query_scan_size	bigint	Die von einer Abfrage gescannte Datenmenge, in MB.
query_priority	integer	Die Priorität der Abfrage. Mögliche Werte sind -1, 0, 1, 2, 3 und 4, wobei -1 bedeutet, dass die Abfragepriorität nicht unterstützt wird.
query_queue_time	bigint	Die Zeitspanne in Mikrosekunden, über die hinweg sich die Abfrage in der Warteschlange befand.
service_class_name	character (64)	Der Name der Serviceklasse.

Beispielabfrage

Um Abfragen mit einer hohen CPU-Zeit (über 1.000 Sekunden) zu finden, führen Sie die folgende Abfrage aus.

```
Select query, cpu_time / 1000000 as cpu_seconds
from stl_query_metrics where segment = -1 and cpu_time > 1000000000
order by cpu_time;

query | cpu_seconds
------+------------
25775 |        9540
```

Um aktive Abfragen mit einem Nested Loop-Join zu finden, die mehr als eine Million Zeilen ausgegeben haben, führen Sie die folgende Abfrage aus.

```
select query, rows
from stl_query_metrics
where step_type = 15 and rows > 1000000
order by rows;

query | rows
------+-----------
25775 | 2621562702
```

Um aktive Abfragen zu finden, die länger als 60 Sekunden liefen und weniger als 10 Sekunden CPU-Zeit genutzt haben, führen Sie die folgende Abfrage aus.

```
select query, run_time/1000000 as run_time_seconds
from stl_query_metrics
where segment = -1 and run_time > 60000000 and cpu_time < 10000000;

query | run_time_seconds
------+-----------------
25775 |              114
```

STL_QUERYTEXT

Erfasst den Abfragetext für SQL-Befehle.

Fragen Sie die Ansicht STL_QUERYTEXT, ab, um den SQL-Text zu erfassen, der für die folgenden Anweisungen protokolliert wurde:

- SELECT, SELECT INTO
- INSERT, UPDATE, DELETE
- COPY
- UNLOAD
- Die durch das Ausführen von VACUUM und ANALYZE generierten Abfragen
- CREATE TABLE AS (CTAS)

Um Aktivitäten für diese Anweisungen über einen bestimmten Zeitraum abzufragen, verbinden Sie die Ansichten STL_QUERYTEXT und STL_QUERY.

> **Note**
>
> Die Ansichten STL_QUERY und STL_QUERYTEXT enthalten nur Informationen zu Abfragen, nicht zu weiteren Utility- und DDL-Befehlen. Für eine Liste und Informationen zu allen von Amazon Redshift ausgeführten Anweisungen können Sie auch die Ansichten STL_DDLTEXT und STL_UTILITYTEXT abfragen. Für eine vollständige Liste aller von Amazon Redshift ausgeführten Anweisungen können Sie die Ansicht SVL_STATEMENTTEXT abfragen.

Vgl. auch STL_DDLTEXT, STL_UTILITYTEXT und SVL_STATEMENTTEXT.

STL_QUERYTEXT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_TEXT zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Transaktions-ID.
pid	integer	Prozess-ID. Normalerweise werden alle Abfragen in einer Sitzung in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen. Im Anschluss an bestimmte interne Ereignisse startet Amazon Redshift möglicher weise eine aktive Sitzung neu und weist eine neue PID zu. Weitere Informationen finden Sie unter STL_RESTARTED_SESSIONS. Mit dieser Spalte können Sie eine Verbindung zur Ansicht STL_ERROR herstellen.

Spaltenname	Datentyp	Beschreibung
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
sequence	integer	Wenn eine einzelne Anweisung mehr als 200 Zeichen enthält, werden weitere Zeilen für diese Anweisung protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite usw.
Text	character(200)	SQL-Text, in Schritten von je 200 Zeichen. Diese Feld kann Sonderzeichen wie Backslash (\\) und Zeilenumbruch (\n) enthalten.

Beispielabfragen

Mit der Funktion PG_BACKEND_PID() können Sie Informationen zur aktuellen Sitzung abrufen. Beispielsweise gibt die folgende Abfrage die Abfrage-ID und einen Teil des Abfragetexts für Abfragen aus, die in der aktuellen Sitzung abgeschlossen wurden.

```
select query, substring(text,1,60)
from stl_querytext
where pid = pg_backend_pid()
order by query desc;

 query |                          substring
-------+------------------------------------------------------------
 28262 | select query, substring(text,1,80) from stl_querytext where
 28252 | select query, substring(path,0,80) as path from stl_unload_l
 28248 | copy category from 's3://dw-tickit/manifest/category/1030_ma
 28247 | Count rows in target table
 28245 | unload ('select * from category') to 's3://dw-tickit/manifes
 28240 | select query, substring(text,1,40) from stl_querytext where
(6 rows)
```

Wiederherstellen von gespeichertem SQL

Um das in der Spalte `text` von STL_QUERYTEXT gespeicherte SQL wiederherzustellen, führen Sie eine SELECT-Anweisung aus, um SQL von mindestens einem Teil der Spalte `text` zu erstellen.

Bevor Sie das wiederhergestellte SQL ausführen, ersetzen Sie alle Sonderzeichen (\n) durch einen Zeilenumbruch. Das Ergebnis der folgenden SELECT-Anweisung besteht aus Zeilen von wiederhergestelltem SQL im Feld `query_statement`.

```
SELECT query, LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END)
 WITHIN GROUP (ORDER BY sequence) as query_statement, COUNT(*) as row_count
FROM stl_querytext GROUP BY query ORDER BY query desc;
```

Mit der folgenden Abfrage werden beispielsweise 3 Spalten ausgewählt. Die Abfrage selbst ist länger als 200 Zeichen und wird in mehreren Teilen in STL_QUERYTEXT gespeichert.

```
select
1 AS a012345678901234567890123456789012345678901234567890123456789012345678901234567890,
2 AS b012345678901234567890123456789012345678901234567890123456789012345678901234567890,
3 AS b012345678901234567890123456789012345678901234
FROM stl_querytext;
```

In diesem Beispiel wird die Abfrage in 2 Teilen (Zeilen) in der Spalte `text` von STL_QUERYTEXT gespeichert.

```
select query, sequence, text
from stl_querytext where query=pg_last_query_id() order by query desc, sequence limit
 10;
```

```
query  | sequence |
                         text

-------+----------
+----------------------------------------------------------------------------------------------------------------
    45 |        0 | select\n1 AS
 a012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n2 AS
 b012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234
    45 |        1 | \nFROM stl_querytext;
```

Um das in STL_QUERYTEXT gespeicherte SQL wiederherzustellen, führen Sie den folgenden SQL-Code aus.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS text
from stl_querytext where query=pg_last_query_id();
```

Um das resultierende wiederhergestellte SQL in Ihrem Client zu verwenden, ersetzen Sie alle Sonderzeichen (\n) durch einen Zeilenumbruch.

```
                    text

----------------------------------------------------------------------------------------------------
 select\n1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
\n2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234\nFROM stl_querytext;
```

STL_REPLACEMENTS

Zeigt ein Protokoll an, das aufzeichnet, wann ungültige UTF-8-Zeichen vom COPY-Befehl mit der Option ACCEPTINVCHARS ersetzt wurden. Für jede der ersten 100 Zeilen auf jedem Knoten-Slice, die mindestens eine Ersetzung erforderten, wird STL_REPLACEMENTS ein Protokolleintrag hinzugefügt.

STL_REPLACEMENTS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_NESTLOOP enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_COPY_REPLACEMENTS zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Nummer des Knoten-Slices, auf dem der Ersatz vorgenomm en wurde.
tbl	integer	Tabellen-ID.
starttime	timestamp	Anfangszeit nach UTC des COPY-Befehls.
Sitzung	integer	Sitzungs-ID der Sitzung, die den COPY-Befehl durchführt.
filename	character (256)	Der vollständige Pfad zur Eingabedatei für den COPY-Befehl.
line_number	bigint	Nummer der Zeile in der Eingabedatendatei, die ein ungültige s UTF-8-Zeichen enthielt. -1 gibt an, dass die Zeilennummer nicht verfügbar ist, z. B. beim Kopieren aus einer Spaltenda tendatei.
colname	character (127)	Das erste Feld, das ein ungültiges UTF-8-Zeichen enthielt.
raw_line	character (1024)	Roh-Ladedaten, die ein ungültiges UTF-8-Zeichen enthielten.

Beispielabfragen

Das folgende Beispiel gibt Ersetzungen für die letzte COPY-Operation aus.

```
select query, session, filename, line_number, colname
from stl_replacements
where query = pg_last_copy_id();
```

```
 query | session |   filename                          | line_number | colname
 ------+---------+-------------------------------------+-------------+--------
    96 |    6314 | s3://mybucket/allusers_pipe.txt   |         251 | city
    96 |    6314 | s3://mybucket/allusers_pipe.txt   |         317 | city
    96 |    6314 | s3://mybucket/allusers_pipe.txt   |         569 | city
    96 |    6314 | s3://mybucket/allusers_pipe.txt   |         623 | city
    96 |    6314 | s3://mybucket/allusers_pipe.txt   |         694 | city
...
```

STL_RESTARTED_SESSIONS

Zur Wahrung der fortdauernden Verfügbarkeit nach bestimmten internen Ereignissen kann Amazon Redshift eine aktive Sitzung mit einer neuen Prozess-ID (PID) starten. Wenn Amazon Redshift eine Sitzung neu startet, zeichnet STL_RESTARTED_SESSIONS die neue und die alte Prozess-ID (PID) auf.

Weitere Informationen finden Sie in den Beispielen nach diesem Abschnitt.

STL_RESTARTED_SESSIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_SESSION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
currenttime	timestamp	Zeitpunkt des Ereignisses.
dbname	character (50)	Name der mit der Sitzung verbundenen Datenbank.
newpid	integer	Prozess-ID der neu gestarteten Sitzung.
oldpid	integer	Prozess-ID der ursprünglichen Sitzung.

Spaltenname	Datentyp	Beschreibung
Benutzernamen ein	character (50)	Name des mit der Sitzung verbundenen Benutzers.
remotehost	character (45)	Name oder IP-Adresse des Remote-Hosts.
remoteport	character (32)	Portnummer des Remote-Hosts.
parkedtime	timestamp	Diese Information ist nur für die interne Verwendung gedacht.
session_vars	character (2000)	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Das folgende Beispiel verbindet STL_RESTARTED_SESSIONS mit STL_SESSIONS zur Anzeige der Benutzernamen für neu gestartete Sitzungen.

```
select process, stl_restarted_sessions.newpid, user_name
from stl_sessions
inner join stl_restarted_sessions on stl_sessions.process =
 stl_restarted_sessions.oldpid
order by process;

...
```

STL_RETURN

Enthält Details für Rückgabe-Schritte in Abfragen. Ein Return-Schritt gibt die Ergebnisse der auf den Datenverarbeitungsknoten ausgeführten Abfragen an den Führungsknoten aus. Der Führungsknoten führt dann die Daten zusammen und gibt die Ergebnisse an den anfragenden Client aus. Bei auf dem Führungsknoten ausgeführten Abfragen gibt ein Return-Schritt die Ergebnisse an den Client aus.

Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Weitere Informationen finden Sie unter Verarbeitung von Abfragen.

STL_RETURN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_RETURN enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die

Spaltenname	Datentyp	Beschreibung
		Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
packets	integer	Gesamtzahl der über das Netzwerk gesendeten Pakete.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Die folgende Abfrage zeigt, welche Schritte in der letzten Abfrage auf jedem Slice ausgeführt wurden.

```
SELECT query, slice, segment, step, endtime, rows, packets
from stl_return where query = pg_last_query_id();

 query |  slice | segment | step |          endtime           | rows | packets
-------+--------+---------+------+----------------------------+------+---------
     4 |      2 |       3 |    2 | 2013-12-27 01:43:21.469043 |    3 |       0
     4 |      3 |       3 |    2 | 2013-12-27 01:43:21.473321 |    0 |       0
     4 |      0 |       3 |    2 | 2013-12-27 01:43:21.469118 |    2 |       0
     4 |      1 |       3 |    2 | 2013-12-27 01:43:21.474196 |    0 |       0
     4 |      4 |       3 |    2 | 2013-12-27 01:43:21.47704  |    2 |       0
     4 |      5 |       3 |    2 | 2013-12-27 01:43:21.478593 |    0 |       0
     4 |   12811|       4 |    1 | 2013-12-27 01:43:21.480755 |    0 |       0
(7 rows)
```

STL_S3CLIENT

Übertragungszeit für Datensätze und andere Leistungsmetriken.

Verwenden Sie die Tabelle STL_S3CLIENT zur Ermittlung der Zeit für die Übertragung der Daten von Amazon S3.

STL_S3CLIENT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
recordtime	timestamp	Zeitpunkt der Protokollierung des Datensatzes.
pid	integer	Prozess-ID. Alle Abfragen in einer Sitzung werden in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen.
http_method	character (64)	Name der HTTP-Methode, die der Amazon-S3-Abfrage entspricht.
Bucket	character (64)	Name des S3 Buckets.
Schlüssel	character (256)	Schlüssel, der dem Amazon-S3-Objekt entspricht.
transfer_size	bigint	Anzahl der übertragenen Bytes.
data_size	bigint	Anzahl der Datenbytes. Dieser Wert ist für nicht komprimie rte Daten identisch mit transfer_size. Wenn Kompression

Spaltenname	Datentyp	Beschreibung
		verwendet wurde, ist dies die Größe der nicht komprimierten Daten.
start_time	bigint	Zeitpunkt, an dem die Übertragung begann (in Mikrosekunden seit dem 1. Januar 2000).
end_time	bigint	Zeitpunkt, an dem die Übertragung endete (in Mikrosekunden seit dem 1. Januar 2000).
transfer_time	bigint	Zeitdauer der Übertragung (in Mikrosekunden).
compression_time	bigint	Anteil der Übertragungszeit, der für die Dekomprimierung aufgewendet wurde (in Mikrosekunden).
connect_time	bigint	Zeit ab dem Beginn bis zum Abschluss der Verbindung zu dem Remote Server (in Mikrosekunden).
app_connect_time	bigint	Zeit ab dem Beginn bis zum Abschluss der SSL-Verbindung/ des Handshakes zu dem Remote Host (in Mikrosekunden).
retries	bigint	Anzahl der erneuten Übertragungsversuche.
request_id	char(32)	Anforderungs-ID vom Amazon-S3-HTTP-Antwort-Header
extended_ request_id	char(128)	Erweiterte Anforderungs-ID von der Amazon-S3-HTTP-Hea der-Antwort (x-amz-id-2).
ip_address	char(64)	IP-Adresse des Servers (ip V4 oder V6).
is_partial	integer	Wert, der bei true (1) angibt, dass die Eingabedatei während eines COPY-Vorgangs in Bereiche aufgeteilt wird. Wenn dieser Wert false (0) ist, wird die Eingabedatei nicht geteilt.
start_offset	bigint	Wenn die Eingabedatei während eines COPY-Vorgangs geteilt wird, gibt dieser Wert den Offset-Wert der Teilung (in Byte) an. Wenn die Datei nicht geteilt wird, ist dieser Wert 0.

Beispielabfrage

Die folgende Abfrage gibt die Zeit für das Laden von Dateien mit einem COPY-Befehl aus.

```
select slice, key, transfer_time
from stl_s3client
where query = pg_last_copy_id();
```

Ergebnis

```
 slice |   key                        | transfer_time
 ------+------------------------------+---------------
     0 | listing10M0003_part_00       |    16626716
     1 | listing10M0001_part_00       |    12894494
     2 | listing10M0002_part_00       |    14320978
     3 | listing10M0000_part_00       |    11293439
  3371 | prefix=listing10M;marker=    |       99395
```

Mit der folgenden Abfrage werden `start_time` und `end_time` in einen Zeitstempel konvertiert.

```
select userid,query,slice,pid,recordtime,start_time,end_time,
'2000-01-01'::timestamp + (start_time/1000000.0)* interval '1 second' as start_ts,
'2000-01-01'::timestamp + (end_time/1000000.0)* interval '1 second' as end_ts
from stl_s3client where query> -1 limit 5;
```

```
 userid | query | slice |  pid  |         recordtime         |   start_time    |
 end_time     |          start_ts          |           end_ts
--------+-------+-------+-------+----------------------------+-----------------
+-----------------+----------------------------+----------------------------
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.207839 | 616436837154256 |
 616436837207838 | 2019-07-14 16:27:17.154256 | 2019-07-14 16:27:17.207838
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.252521 | 616436837208208 |
 616436837252520 | 2019-07-14 16:27:17.208208 | 2019-07-14 16:27:17.25252
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.284376 | 616436837208460 |
 616436837284374 | 2019-07-14 16:27:17.20846  | 2019-07-14 16:27:17.284374
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.285307 | 616436837208980 |
 616436837285306 | 2019-07-14 16:27:17.20898  | 2019-07-14 16:27:17.285306
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.353853 | 616436837302216 |
 616436837353851 | 2019-07-14 16:27:17.302216 | 2019-07-14 16:27:17.353851
```

STL_S3CLIENT_ERROR

Zeichnet die Fehler auf, die beim Laden einer Datei von Amazon S3 auf einem Slice aufgetreten sind.

Verwenden Sie STL_S3CLIENT_ERROR, um Details zu Fehlern zu ermitteln, die bei der Übertragung von Daten von Amazon S3 im Rahmen eines COPY-Befehls aufgetreten sind.

STL_S3CLIENT_ERROR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen. Die Abfrage-ID -1 ist für den internen Gebrauch bestimmt.
sliceid	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
recordtime	timestamp	Zeitpunkt der Protokollierung des Datensatzes.
pid	integer	Prozess-ID. Alle Abfragen in einer Sitzung werden in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen.
http_method	character (64)	Name der HTTP-Methode, die der Amazon-S3-Abfrage entspricht.
Bucket	character (64)	Amazon-S3-Bucket-Name.
Schlüssel	character (256)	Schlüssel, der dem Amazon-S3-Objekt entspricht.

Spaltenname	Datentyp	Beschreibung
error	character (1024)	Fehlermeldung.
is_partial	integer	Wert, der bei true (1) angibt, dass die Eingabedatei während eines COPY-Vorgangs in Bereiche aufgeteilt wird. Wenn dieser Wert false (0) ist, wird die Eingabedatei nicht geteilt.
start_offset	bigint	Wenn die Eingabedatei während eines COPY-Vorgangs geteilt wird, gibt dieser Wert den Offset-Wert der Teilung (in Byte) an. Wenn die Datei nicht geteilt wird, ist dieser Wert 0.

Nutzungshinweise

Wenn Sie mehrere Fehler mit „Verbindungstimeout“ sehen, liegt möglicherweise ein Netzwerkproblem vor. Wenn Sie Enhanced VPC Routing verwenden, prüfen Sie, ob ein gültiger Netzwerkpfad zwischen der VPC Ihres Clusters und Ihren Datenressourcen besteht. Weitere Informationen finden Sie unter Amazon Redshift Enhanced VPC Routing.

Beispielabfrage

Die folgende Abfrage gibt die Fehler von während der aktuellen Sitzung ausgeführten COPY-Befehlen aus.

```
select query, sliceid, substring(key from 1 for 20) as file,
substring(error from 1 for 35) as error
from stl_s3client_error
where pid = pg_backend_pid()
order by query desc;
```

Ergebnis

```
 query  | sliceid |     file           |              error

--------+---------+--------------------+-----------------------------------
 362228 |      12 | part.tbl.25.159.gz | transfer closed with 1947655 bytes
 362228 |      24 | part.tbl.15.577.gz | transfer closed with 1881910 bytes
 362228 |       7 | part.tbl.22.600.gz | transfer closed with 700143 bytes r
```

```
 362228 |      22 | part.tbl.3.34.gz   | transfer closed with 2334528 bytes
 362228 |      11 | part.tbl.30.274.gz | transfer closed with 699031 bytes r
 362228 |      30 | part.tbl.5.509.gz  | Unknown SSL protocol error in conne
 361999 |      10 | part.tbl.23.305.gz | transfer closed with 698959 bytes r
 361999 |      19 | part.tbl.26.582.gz | transfer closed with 1881458 bytes
 361999 |       4 | part.tbl.15.629.gz | transfer closed with 2275907 bytes
 361999 |      20 | part.tbl.6.456.gz  | transfer closed with 692162 bytes r
(10 rows)
```

STL_SAVE

Enthält Details für Speicher-Schritte in Abfragen. Ein Speicher-Schritt speichert den Eingangsstream in einer temporären Tabelle. Eine temporäre Tabelle speichert sofort die Ergebnisse während der Abfrageausführung.

Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Weitere Informationen finden Sie unter Verarbeitung von Abfragen.

STL_SAVE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_SAVE enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.

Spaltenname	Datentyp	Beschreibung
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
tbl	integer	ID der materialisierten temporären Tabelle.
is_diskbased	character(1)	Ob dieser Schritt der Abfrage als festplattenbasierte Operation ausgeführt wurde: true (`t`) oder false (`f`).
workmem	bigint	Größe des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.

Beispielabfragen

Die folgende Abfrage zeigt, welche Speicherschritte in der letzten Abfrage auf jedem Slice ausgeführt wurden.

```
select query, slice, segment, step, tasknum, rows,  tbl
from stl_save where query = pg_last_query_id();

 query | slice | segment | step | tasknum | rows | tbl
-------+-------+---------+------+---------+------+-----
 52236 |     3 |       0 |    2 |      21 |    0 | 239
 52236 |     2 |       0 |    2 |      20 |    0 | 239
 52236 |     2 |       2 |    2 |      20 |    0 | 239
 52236 |     3 |       2 |    2 |      21 |    0 | 239
 52236 |     1 |       0 |    2 |      21 |    0 | 239
 52236 |     0 |       0 |    2 |      20 |    0 | 239
 52236 |     0 |       2 |    2 |      20 |    0 | 239
 52236 |     1 |       2 |    2 |      21 |    0 | 239
(8 rows)
```

STL_SCAN

Analysiert Tabellen-Scan-Schritte für Abfragen. Die Schrittnummer für Zeilen in dieser Tabelle ist immer 0, da ein Scan der erste Schritt in einem Segment ist.

STL_SCAN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_SCAN enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
fetches	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Spaltenname	Datentyp	Beschreibung
type	integer	ID des Scan-Typs. Eine Liste der Werte finden Sie in der folgenden Tabelle.
tbl	integer	Tabellen-ID.
is_rrscan	character(1)	„True“ (t) zeigt an, dass für diesen Schritt ein Scan mit Bereichseinschränkung durchgeführt wurde.
is_delayed_scan	character(1)	Diese Information ist nur für die interne Verwendung gedacht.
rows_pre_filter	bigint	Bei Scans permanenter Tabellen die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen ausgegeben wurden (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter.
rows_pre_user_filter	bigint	Bei Scans permanenter Tabellen die Anzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen verarbeitet wurden (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter.
perm_table_name	character(136)	Bei Scans permanenter Tabellen der Name der gescannten Tabelle.
is_rlf_scan	character(1)	„True“ (t) zeigt an, dass für diesen Schritt eine Filterung auf Zeilenebene durchgeführt wurde.
is_rlf_scan_reason	integer	Diese Information ist nur für die interne Verwendung gedacht.
num_em_blocks	integer	Diese Information ist nur für die interne Verwendung gedacht.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Spaltenname	Datentyp	Beschreibung
runtime_filtering	character(1)	Gibt bei true (t) an, dass Laufzeitfilter angewendet werden.
scan_region	Ganzzahl	Diese Information ist nur für die interne Verwendung gedacht.
num_sortkey_as_predicate	Ganzzahl	Diese Information ist nur für die interne Verwendung gedacht.
row_fetcher_state	Ganzzahl	Diese Information ist nur für die interne Verwendung gedacht.
consumed_scan_ranges	bigint	Diese Information ist nur für die interne Verwendung gedacht.
work_stealing_reason	bigint	Diese Information ist nur für die interne Verwendung gedacht.
is_vectorized_scan	character(1)	Diese Information ist nur für die interne Verwendung gedacht.
is_vectorized_scan_reason	Ganzzahl	Diese Information ist nur für die interne Verwendung gedacht.
row_fetcher_reason	bigint	Diese Information ist nur für die interne Verwendung gedacht.
topology_signature	bigint	Diese Information ist nur für die interne Verwendung gedacht.
use_tpm_partition	character(1)	Diese Information ist nur für die interne Verwendung gedacht.

Spaltenname	Datentyp	Beschreibung
is_rrscan_expr	character(1)	Diese Information ist nur für die interne Verwendung gedacht.
scanned_mega_value	character(1)	Diese Information ist nur für die interne Verwendung gedacht. Diese Information zeigt an, ob der bestimmte Scanschritt einen großen Wert gescannt hat. Ein großer Wert wird in mehreren Blöcken gespeichert. Die Blockgröße beträgt standardmäßig 1 MB, ein großer Wert ist in einer Standardeinstellung größer als 1 MB.

Scan-Typen

Typ-ID	Beschreibung
1	Daten aus dem Netzwerk.
2	Permanente Benutzertabellen in komprimiertem gemeinsam genutztem Speicher.
3	Temporäre zeilenweise Tabellen.
21	Laden Sie Dateien von Amazon S3.
22	Laden Sie Tabellen aus Amazon DynamoDB
23	Daten von einer Remote-SSH-Verbindung laden.
24	Daten von einem Remote-Cluster (sortierte Region) laden. Dies wird zur Größenanpassung verwendet.
25	Daten von einem Remote-Cluster (nicht sortierte Region) laden. Dies wird zur Größenanpassung verwendet.
28	Lesen von Daten aus einer Zeitreihenansicht mit UNION ALL auf mehreren Tabellen

Typ-ID	Beschreibung
29	Lesen von Daten aus externen Amazon-S3-Tabellen
30	Lesen der Partitionsinformationen einer externen Amazon-S3-Tabelle
33	Lesen von Daten aus einer Remote-Postgres-Tabelle
36	Lesen von Daten aus einer Remote-MySQL-Tabelle
37	Lesen von Daten aus einem Remote-Kinesis-Stream

Nutzungshinweise

Idealerweise sollte `rows` relativ nahe bei `rows_pre_filter` liegen. Eine große Differenz zwischen `rows` und `rows_pre_filter` zeigt an, dass die Ausführungsengine Zeilen scannt, die später verworfen werden, was nicht effizient ist. Die Differenz zwischen `rows_pre_filter` und `rows_pre_user_filter` ist die Anzahl von Geisterzeilen in dem Scan. Führen Sie einen VACUUM-Vorgang durch, um die zum Löschen markierten Zeilen zu entfernen. Die Differenz zwischen `rows` und `rows_pre_user_filter` ist die Anzahl der Zeilen, die von der Abfrage gefiltert wurden. Wenn sehr viele Zeilen von dem Benutzerfilter verworfen werden, prüfen Sie Ihre Auswahl der Sortierspalte oder, wenn eine große nicht sortierte Region dafür verantwortlich ist, führen Sie eine Bereinigung durch.

Beispielabfragen

Das folgende Beispiel zeigt, dass `rows_pre_filter` größer als `rows_pre_user_filter` ist, da die Tabelle gelöschte Zeilen enthält, die noch nicht bereinigt wurden (Geisterzeilen).

```
SELECT query, slice, segment,step,rows, rows_pre_filter, rows_pre_user_filter
from stl_scan where query = pg_last_query_id();

 query |  slice | segment | step | rows  | rows_pre_filter | rows_pre_user_filter
-------+--------+---------+------+-------+-----------------+----------------------
 42915 |      0 |       0 |    0 | 43159 |           86318 |                43159
 42915 |      0 |       1 |    0 |     1 |               0 |                    0
 42915 |      1 |       0 |    0 | 43091 |           86182 |                43091
 42915 |      1 |       1 |    0 |     1 |               0 |                    0
 42915 |      2 |       0 |    0 | 42778 |           85556 |                42778
 42915 |      2 |       1 |    0 |     1 |               0 |                    0
```

```
 42915 |      3 |       0 |    0 | 43428 |           86856 |                43428
 42915 |      3 |       1 |    0 |     1 |               0 |                    0
 42915 |  10000 |       2 |    0 |     4 |               0 |                    0
(9 rows)
```

STL_SCHEMA_QUOTA_VIOLATIONS

Zeichnet das Auftreten, den Zeitstempel, die XID und andere nützliche Informationen auf, wenn ein Schema-Kontingent überschritten wird.

STL_SCHEMA_QUOTA_VIOLATIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_SCHEMA_QUOTA_VIOLATIONS zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
ownerid	integer	Die ID des Schemabesitzers.
xid	bigint	Die mit der Anweisung verbundene Transaktions-ID.
pid	integer	Die mit der Anweisung verbundene Prozess-ID.
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
schema_id	integer	Die Namespace- oder Schema-ID.
schema_name	Zeichen (128)	Der Namespace- oder Schemaname.
quota	integer	Die Menge an Speicherplatz (in MB), die das Schema verwenden kann.

Spaltenname	Datentyp	Beschreibung
disk_usage	integer	Der Speicherplatz (in MB), der zurzeit vom Schema verwendet wird.
disk_usage_pct	double precision	Der Prozentsatz des Festplattenspeichers, der zurzeit vom Schema aus dem konfigurierten Kontingent verwendet wird.
timestamp	Timestamp ohne Zeitzone	Der Zeitpunkt, an dem die Verletzung aufgetreten ist.

Beispielabfragen

Die folgende Abfrage zeigt das Ergebnis einer Kontingentverletzung:

```
SELECT userid, TRIM(SCHEMA_NAME) "schema_name", quota, disk_usage, disk_usage_pct,
 timestamp FROM
stl_schema_quota_violations WHERE SCHEMA_NAME = 'sales_schema' ORDER BY timestamp DESC;
```

Diese Abfrage gibt die folgende Beispielausgabe für das angegebene Schema zurück:

```
userid | schema_name  | quota | disk_usage | disk_usage_pct |timestamp
-------+--------------+-------+------------+----------------
+----------------------------
104    | sales_schema | 2048  | 2798       |  136.62        | 2020-04-20
 20:09:25.494723
(1 row)
```

STL_SESSIONS

Gibt Informationen zum Verlauf der Benutzersitzungen aus.

STL_SESSIONS unterscheidet sich darin von STV_SESSIONS, dass STL_SESSIONS den Sitzungsverlauf enthält, während STV_SESSIONS die aktuell aktiven Sitzungen enthält.

STL_SESSIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_SESSION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
starttime	timestamp	Zeitpunkt des Beginns der Sitzung, nach UTC.
endtime	timestamp	Zeitpunkt des Endes der Sitzung, nach UTC.
Verarbeitung	integer	Prozess-ID der Sitzung.
user_name	character(50)	Der Name des mit der Sitzung verbundenen Benutzers.
db_name	character(50)	Name der mit der Sitzung verbundenen Datenbank.
timeout_sec	int	Die maximale Zeit in Sekunden, die eine Sitzung ohne Timeout inaktiv oder untätig bleibt. 0 bedeutet, dass kein Timeout eingestellt ist.
timed_out	int	Ein Wert, der angibt, ob eine Sitzung das Zeitlimit überschritten hat: 1, wenn das Zeitlimit überschritten wurde, andernfalls 0.

Beispielabfragen

Geben Sie die folgende Abfrage ein, um den Sitzungsverlauf für die TICKIT-Datenbank anzuzeigen:

```
select starttime, process, user_name, timeout_sec, timed_out
from stl_sessions
```

```
where db_name='tickit' order by starttime;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
    starttime              | process |  user_name              | timeout_sec | timed_out
---------------------------+---------+-------------------------+-------------
+-------------
2008-09-15 09:54:06.746705 |   32358 | dwuser                  | 120         | 1
2008-09-15 09:56:34.30275  |   32744 | dwuser                  | 60          | 1
2008-09-15 11:20:34.694837 |   14906 | dwuser                  | 0           | 0
2008-09-15 11:22:16.749818 |   15148 | dwuser                  | 0           | 0
2008-09-15 14:32:44.66112  |   14031 | dwuser                  | 0           | 0
2008-09-15 14:56:30.22161  |   18380 | dwuser                  | 0           | 0
2008-09-15 15:28:32.509354 |   24344 | dwuser                  | 0           | 0
2008-09-15 16:01:00.557326 |   30153 | dwuser                  | 120         | 1
2008-09-15 17:28:21.419858 |   12805 | dwuser                  | 0           | 0
2008-09-15 20:58:37.601937 |   14951 | dwuser                  | 60          | 1
2008-09-16 11:12:30.960564 |   27437 | dwuser                  | 60          | 1
2008-09-16 14:11:37.639092 |   23790 | dwuser                  | 3600        | 1
2008-09-16 15:13:46.02195  |    1355 | dwuser                  | 120         | 1
2008-09-16 15:22:36.515106 |    2878 | dwuser                  | 120         | 1
2008-09-16 15:44:39.194579 |    6470 | dwuser                  | 120         | 1
2008-09-16 16:50:27.02138  |   17254 | dwuser                  | 120         | 1
2008-09-17 12:05:02.157208 |    8439 | dwuser                  | 3600        | 0
(17 rows)
```

STL_SORT

Zeigt die Sortier-Ausführungsschritte für Abfragen an, etwa Schritte, die die ORDER BY-Verarbeitung verwenden.

STL_SORT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_SORT enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf

Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.

Spaltenname	Datentyp	Beschreibung
bytes	bigint	Größe, in Bytes, aller Ausgabezeilen für den Schritt.
tbl	integer	Tabellen-ID.
is_diskbased	character(1)	Bei „true (t)“ wurde die Abfrage als festplattenbasierte Operation ausgeführt. Bei „false (f)“ wurde die Abfrage im Arbeitsspeicher ausgeführt.
workmem	bigint	Gesamtgröße des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.
Prüfsumme	bigint	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfragen

Das folgende Beispiel gibt die Sortierergebnisse für Slice 0 und Segment 1 aus.

```
select query, bytes, tbl, is_diskbased, workmem
from stl_sort
where slice=0 and segment=1;
```

```
 query |  bytes  | tbl | is_diskbased |  workmem
-------+---------+-----+--------------+-----------
   567 | 3126968 | 241 | f            | 383385600
   604 |    5292 | 242 | f            | 383385600
   675 |  104776 | 251 | f            | 383385600
   525 | 3126968 | 251 | f            | 383385600
   585 |    5068 | 241 | f            | 383385600
   630 |  204808 | 266 | f            | 383385600
   704 |       0 | 242 | f            |         0
   669 | 4606416 | 241 | f            | 383385600
   696 |  104776 | 241 | f            | 383385600
   651 | 4606416 | 254 | f            | 383385600
   632 |       0 | 256 | f            |         0
   599 |     396 | 241 | f            | 383385600
 86397 |       0 | 242 | f            |         0
```

```
   621 |    5292 | 241 | f            | 383385600
 86325 |       0 | 242 | f            |         0
   572 |    5068 | 242 | f            | 383385600
   645 |  204808 | 241 | f            | 383385600
   590 |     396 | 242 | f            | 383385600
(18 rows)
```

STL_SSHCLIENT_ERROR

Zeichnet alle vom SSH-Client erkannten Fehler auf.

STL_SSHCLIENT_ERROR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
recordtime	timestamp	Zeitpunkt, zu dem der Fehler protokolliert wurde.
pid	integer	Der Prozess, der den Fehler protokolliert hat.
ssh_username	character (1024)	Der Name des SSH-Benutzers.
endpoint	character (1024)	Der SSH-Endpunkt.
command	character (4096)	Der vollständige SSH-Befehl.

Spaltenname	Datentyp	Beschreibung
error	character (1024)	Die Fehlermeldung.

STL_STREAM_SEGS

Führt das Verhältnis zwischen Streams und gleichzeitigen Segmenten auf.

Bei den Streams handelt es sich in diesem Kontext um Amazon-Redshift-Streams. Diese Systemansicht gilt nicht für Streaming-Erfassung.

STL_STREAM_SEGS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_STREAM_SEGS enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.

Spaltenname	Datentyp	Beschreibung
stream	integer	Der Satz gleichzeitiger Segmente einer Abfrage.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.

Beispielabfragen

Geben Sie die folgende Abfrage ein, um das Verhältnis zwischen Streams und gleichzeitigen Segmenten für die letzte Abfrage anzuzeigen:

```
select *
from stl_stream_segs
where query = pg_last_query_id();

 query | stream | segment
-------+--------+---------
    10 |      1 |       2
    10 |      0 |       0
    10 |      2 |       4
    10 |      1 |       3
    10 |      0 |       1
(5 rows)
```

STL_TR_CONFLICT

Zeigt Informationen zur Identifizierung und Lösung von Transaktionskonflikten mit Datenbanktabellen an.

Ein Transaktionskonflikt tritt auf, wenn zwei oder mehr Benutzer Datenzeilen aus Tabellen abfragen und bearbeiten und ihre Transaktionen nicht in serieller Reihenfolge abgearbeitet werden können. Die Transaktion, die eine Anweisung ausführt, die die Serialisierbarkeit verhindern würde, wird beendet. Für die Transaktion wird ein Rollback ausgeführt. Immer wenn ein Transaktionskonflikt auftritt, schreibt Amazon Redshift eine Datenzeile mit den Details der abgebrochenen Transaktion in die Systemtabelle STL_TR_CONFLICT. Weitere Informationen finden Sie unter Serialisierbare Isolierung.

STL_TR_CONFLICT ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_TRANSACTION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
xact_id	bigint	Transaktions-ID der zurückabgewickelten Transaktion.
process_id	bigint	Der der Transaktion, für die ein Rollback durchgeführt wurde, zugeordnete Prozess
xact_start_ts	Zeitstempel	Zeitpunkt (UTC), an dem die Transaktion begann.
abort_time	Zeitstempel	Zeitpunkt (UTC), an dem die Transaktion beendet wurde.
table_id	bigint	Tabellen-ID der Tabelle, in der der Konflikt auftrat.

Beispielabfrage

Führen Sie eine Abfrage mit der Tabellen-ID aus, um Informationen zu Konflikten unter Beteiligung einer bestimmten Tabelle zu erhalten:

```
select * from stl_tr_conflict where table_id=100234
order by xact_start_ts;

xact_id|process_|      xact_start_ts       |        abort_time        |table_
       |id      |                          |                          |id
-------+--------+--------------------------+--------------------------+------
  1876 |  8551  |2010-03-30 09:19:15.852326|2010-03-30 09:20:17.582499|100234
  1928 | 15034  |2010-03-30 13:20:00.636045|2010-03-30 13:20:47.766817|100234
  1991 | 23753  |2010-04-01 13:05:01.220059|2010-04-01 13:06:06.94098 |100234
  2002 | 23679  |2010-04-01 13:17:05.173473|2010-04-01 13:18:27.898655|100234
(4 rows)
```

Sie finden die Tabellen-ID im Abschnitt DETAIL der Fehlermeldung bei Serialisierbarkeitsfehlern (Fehler 1023).

STL_UNDONE

Zeigt Informationen zu Transaktionen an, die rückgängig gemacht wurden.

STL_UNDONE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_TRANSACTION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xact_id	bigint	ID für die Rückgängig-Transaktion.
xact_id_undone	bigint	ID der Transaktion, die rückgängig gemacht wurde.
undo_start_ts	timestamp	Anfangszeit der Rückgängig-Transaktion.
undo_end_ts	timestamp	Endzeit der Rückgängig-Transaktion.
table_id	bigint	ID der Tabelle, die von der Rückgängig-Transaktion betroffen war.

Beispielabfrage

Geben Sie den folgenden Befehl ein, um ein knappes Protokoll aller rückgängig gemachter Transaktionen anzuzeigen:

```
select xact_id, xact_id_undone, table_id from stl_undone;
```

Dieser Befehl gibt die folgende Beispielausgabe aus:

```
 xact_id | xact_id_undone | table_id
---------+----------------+----------
1344 |           1344 |   100192
1326 |           1326 |   100192
1551 |           1551 |   100192
(3 rows)
```

STL_UNIQUE

Analysiert die Ausführungsschritte, die auftreten, wenn eine DISTINCT-Funktion in der SELECT-Liste verwendet wird, oder wenn in einer UNION- oder INTERSECT-Abfrage Duplikate entfernt werden.

STL_UNIQUE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_UNIQUE enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **2009-06-12 11:29:19.131358** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
type	character(6)	Die Art des Schritts. Folgende Werte sind zulässig: • HASHED. Zeigt an, dass der Schritt die gruppierte, nicht sortierte Aggregierung verwendet hat.

Spaltenname	Datentyp	Beschreibung
		• PLAIN. Zeigt an, dass der Schritt die nicht gruppierte, skalare Aggregierung verwendet hat. • SORTED. Zeigt an, dass der Schritt die gruppierte, sortierte Aggregierung verwendet hat.
is_diskbased	character(1)	Bei „true (t)“ wurde die Abfrage als festplattenbasierte Operation ausgeführt. Bei „false (f)“ wurde die Abfrage im Arbeitsspeicher ausgeführt.
slots	integer	Gesamtzahl der Hash-Buckets.
workmem	bigint	Gesamtgröße des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.
max_buffers_used	bigint	Maximale Anzahl der in der Hash-Tabelle vor der Übertragung auf die Festplatte verwendeten Puffer.
resizes	integer	Diese Information ist nur für die interne Verwendung gedacht.
occupied	Ganzzahl	Diese Information ist nur für die interne Verwendung gedacht.
flushable	integer	Diese Information ist nur für die interne Verwendung gedacht.
used_unique_prefetching	character(1)	Diese Information ist nur für die interne Verwendung gedacht.
bytes	biginit	Die Byteanzahl aller Ausgabezeilen für den Schritt.

Beispielabfragen

Angenommen, Sie führen die folgende Abfrage aus:

```
select distinct eventname
from event order by 1;
```

Für den Fall, dass die ID der vorherigen Abfrage 6313 ist, zeigt das folgende Beispiel die Anzahl der Zeilen, die der eindeutige Schritt für jeden Slice in den Segmenten 0 und 1 erstellt hat.

```
select query, slice, segment, step, datediff(msec, starttime, endtime) as msec,
 tasknum, rows
from stl_unique where query = 6313
order by query desc, slice, segment, step;
```

```
 query | slice | segment | step | msec | tasknum | rows
-------+-------+---------+------+------+---------+------
  6313 |     0 |       0 |    2 |    0 |      22 |  550
  6313 |     0 |       1 |    1 |  256 |      20 |  145
  6313 |     1 |       0 |    2 |    1 |      23 |  540
  6313 |     1 |       1 |    1 |   42 |      21 |  127
  6313 |     2 |       0 |    2 |    1 |      22 |  540
  6313 |     2 |       1 |    1 |  255 |      20 |  158
  6313 |     3 |       0 |    2 |    1 |      23 |  542
  6313 |     3 |       1 |    1 |   38 |      21 |  146
(8 rows)
```

STL_UNLOAD_LOG

Zeichnet die Details für eine Entladeoperation auf.

STL_UNLOAD_LOG zeichnet für jede von einer UNLOAD-Anweisung erstellte Datei eine Zeile auf. Zum Beispiel: Wenn ein UNLOAD-Vorgang 12 Dateien erstellt, enthält STL_UNLOAD_LOG 12 entsprechende Zeilen.

STL_UNLOAD_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_UNLOAD_LOG enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht

enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_UNLOAD_HISTORY und SYS_UNLOAD_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Die Abfrage-ID.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
pid	integer	Die mit der Abfrageanweisung verbundene Prozess-ID.
path (Pfad)	character(1280)	Der vollständige Amazon-S3-Objektpfad für die Datei.
start_time	timestamp	Anfangszeit der Transaktion.
end_time	timestamp	Endzeit der Transaktion.
line_count	bigint	Anzahl der in die Datei entladenen Zeilen.
transfer_size	bigint	Anzahl der übertragenen Bytes.
file_format	character(10)	Format der entladenen Datei.

Beispielabfrage

Um eine Liste der Dateien zu erhalten, die durch einen UNLOAD-Befehl in Amazon S3 geschrieben wurden, rufen Sie nach Abschluss des UNLOAD-Befehls eine Amazon-S3-Listenoperation auf. Sie können auch STL_UNLOAD_LOG abfragen.

Die folgende Abfrage gibt den Pfadnamen für Dateien aus, die durch einen UNLOAD-Befehl für die zuletzt ausgeführte Abfrage erstellt wurden:

```
select query, substring(path,0,40) as path
from stl_unload_log
where query = pg_last_query_id()
order by path;
```

Dieser Befehl gibt die folgende Beispielausgabe aus:

```
 query |             path
-------+---------------------------------------
  2320 | s3://my-bucket/venue0000_part_00
  2320 | s3://my-bucket/venue0001_part_00
  2320 | s3://my-bucket/venue0002_part_00
  2320 | s3://my-bucket/venue0003_part_00
(4 rows)
```

STL_USAGE_CONTROL

Die Ansicht STL_USAGE_CONTROL enthält Informationen, die protokolliert werden, wenn ein Nutzungslimit erreicht wird. Weitere Informationen zu Nutzungslimits finden Sie unter Verwalten der Nutzungslimits im Amazon-Redshift-Verwaltungshandbuch.

STL_USAGE_CONTROL ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
eventtime	Zeitstempel	Die Uhrzeit (UTC), zu der die Abfrage ein Nutzungslimit überschritten hat.
query	integer	Die Abfrage-ID. Sie können diese ID verwenden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
xid	bigint	Die Transaktions-ID.

Spaltenname	Datentyp	Beschreibung
pid	integer	Die Prozess-ID, die der Abfrage zugeordnet ist.
usage_limit_id	character(40)	Ein Universally Unique Identifier (UUID), der von Amazon Redshift generiert wird, beispielsweise `25d9297e-3e7b-41c8-9f4d-c4b6eb731c09` .
feature_type	character(30)	Das Feature, dessen Nutzungslimit überschritten wurde. Mögliche Werte sind `CONCURRENCY_SCALING` und `SPECTRUM`.

Beispielabfrage

Im folgenden SQL-Beispiel werden einige der Informationen zurückgegeben, die protokolliert werden, wenn ein Nutzungslimit erreicht wird.

```
select query, pid, eventtime, feature_type
from stl_usage_control
order by eventtime desc
limit 5;
```

STL_USERLOG

Zeichnet die Details der folgenden Änderungen an einem Datenbankbenutzer auf:

- Benutzer erstellen
- Benutzer entfernen
- Benutzer ändern (umbenennen)
- Benutzer ändern (Eigenschaften ändern)

STL_USERLOG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_USERLOG zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie

leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des von der Änderung betroffenen Benutzers.
Benutzernamen ein	character(50)	Benutzername des von der Änderung betroffenen Benutzers.
oldusername	character(50)	Bei einer Umbenennungsaktion der ursprüngliche Benutzername. Bei anderen Aktionen ist dieses Feld leer.
Aktion	character(10)	Erfolgte Aktion. Zulässige Werte: • Ändern • Erstellen • Entfernen • Umbenennen
usecreatedb	integer	„True“ (1) zeigt an, dass der Benutzer über Berechtigungen zum Erstellen von Datenbanken verfügt.
usesuper	integer	„True“ (1) zeigt an, dass es sich um einen Superuser handelt.
usecatupd	integer	„True“ (1) zeigt an, dass der Benutzer Systemkataloge aktualisieren kann.
valuntil	timestamp	Ablaufdatum des Passworts.
pid	integer	Prozess-ID.
xid	bigint	Transaktions-ID.
recordtime	timestamp	Zeitpunkt des Beginns der Abfrage, nach UTC.

Beispielabfragen

Das folgende Beispiel führt vier Benutzeraktionen aus und fragt dann die STL_USERLOG-Ansicht ab.

```
create user userlog1 password 'Userlog1';
alter user userlog1 createdb createuser;
alter user userlog1 rename  to userlog2;
drop user userlog2;

select userid, username, oldusername, action, usecreatedb, usesuper from stl_userlog
 order by recordtime desc;
```

```
 userid |  username | oldusername |  action | usecreatedb | usesuper
--------+-----------+-------------+---------+-------------+----------
    108 | userlog2  |             | drop    |           1 |   1
    108 | userlog2  | userlog1    | rename  |           1 |   1
    108 | userlog1  |             | alter   |           1 |   1
    108 | userlog1  |             | create  |           0 |   0
 (4 rows)
```

STL_UTILITYTEXT

Erfasst den Text von auf der Datenbank ausgeführten Nicht-SELECT SQL-befehlen.

Fragen Sie die Ansicht STL_UTILITYTEXT ab, um die folgende Teilmenge von SQL-Anweisungen zu erfassen, die auf dem System ausgeführt wurden:

- ABORT, BEGIN, COMMIT, END, ROLLBACK
- ANALYZE
- CALL
- CANCEL
- COMMENT
- CREATE, ALTER, DROP DATABASE
- CREATE, ALTER, DROP USER

- EXPLAIN
- GRANT, REVOKE
- LOCK
- RESET
- SET
- ZEIGEN
- TRUNCATE

Vgl. auch STL_DDLTEXT, STL_QUERYTEXT und SVL_STATEMENTTEXT.

Verwenden Sie die Spalten STARTTIME und ENDTIME, um zu erfahren, welche Anweisungen während eines bestimmten Zeitraums protokolliert wurden. Lange SQL-Textblöcke werden in 200 Zeichen lange Zeilen umgebrochen; die Spalte SEQUENCE identifiziert Textfragmente, die zu einer einzelnen Anweisung gehören.

STL_UTILITYTEXT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Transaktions-ID.
pid	integer	Die mit der Abfrageanweisung verbundene Prozess-ID.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP

Spaltenname	Datentyp	Beschreibung
		definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist dieses Feld leer.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
sequence	integer	Wenn eine einzelne Anweisung mehr als 200 Zeichen enthält, werden weitere Zeilen für diese Anweisung protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite usw.
Text	character(200)	SQL-Text, in Schritten von je 200 Zeichen. Diese Feld kann Sonderzeichen wie Backslash (\\) und Zeilenumbruch (\n) enthalten.

Beispielabfragen

Die folgende Abfrage gibt den Text für „Utility“-Befehle aus, die am 26. Januar 2012 ausgeführt wurden. In diesem Fall wurden einige SET-befehle und ein SHOW ALL-Befehl ausgeführt:

```
select starttime, sequence, rtrim(text)
from stl_utilitytext
where starttime like '2012-01-26%'
order by starttime, sequence;

starttime          | sequence |              rtrim
```

```
---------------------------+-----+---------------------------------
2012-01-26 13:05:52.529235 |   0 | show all;
2012-01-26 13:20:31.660255 |   0 | SET query_group to ''
2012-01-26 13:20:54.956131 |   0 | SET query_group to 'soldunsold.sql'
...
```

Wiederherstellen von gespeichertem SQL

Um das in der Spalte `text` von STL_UTILITYTEXT gespeicherte SQL wiederherzustellen, führen Sie eine SELECT-Anweisung aus, um SQL von mindestens einem Teil der Spalte `text` zu erstellen. Bevor Sie das wiederhergestellte SQL ausführen, ersetzen Sie alle Sonderzeichen (\n) durch einen Zeilenumbruch. Das Ergebnis der folgenden SELECT-Anweisung besteht aus Zeilen von wiederhergestelltem SQL im Feld `query_statement`.

```
SELECT LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END) WITHIN
 GROUP (ORDER BY sequence) as query_statement
FROM stl_utilitytext GROUP BY xid order by xid;
```

Die folgende Abfrage setzt beispielsweise die query_group auf eine Zeichenfolge von Nullen. Die Abfrage selbst ist länger als 200 Zeichen und wird in mehreren Teilen in STL_UTILITYTEXT gespeichert.

```
set query_group to
 '0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
             000000';
```

In diesem Beispiel wird die Abfrage in 2 Teilen (Zeilen) in der Spalte `text` von STL_UTILITYTEXT gespeichert.

```
select query, sequence, text
from stl_utilitytext where query=pg_last_query_id() order by query desc, sequence limit
 10;
```

```


         starttime          | sequence |
                                                text

----------------------------+----------
+-----------------------------------------------------------------------------------------------
```

```
 2019-07-23 22:55:34.926198 |        0 | set query_group to
 '0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000\n00000000000
 2019-07-23 22:55:34.926198 |        1 |                   000000';
```

Um das in STL_UTILITYTEXT gespeicherte SQL wiederherzustellen, führen Sie den folgenden SQL-Code aus.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS query_statement
from stl_utilitytext where query=pg_last_query_id();
```

Um das resultierende wiederhergestellte SQL in Ihrem Client zu verwenden, ersetzen Sie alle Sonderzeichen (\n) durch einen Zeilenumbruch.

```
                                                 query_statement


-------------------------------------------------------------------------------------------------------
 set query_group to
 '0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000\n00000000000
                  000000';
```

STL_VACUUM

Zeigt die Statistiken für Zeilen und Blöcke für Tabellen an, die bereinigt wurden.

Die Ansicht zeigt Informationen dazu an, wann jede Bereinigungsaktion gestartet und beendet wurde, und zeigt die Vorteile der Ausführung dieser Operation. Für Informationen zu den Voraussetzungen für diesen Befehl vgl. die Beschreibung des Befehls VACUUM.

STL_VACUUM ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_VACUUM_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Die Transaktions-ID der VACUUM-Anweisung. Sie können diese Tabelle mit der Ansicht STL_QUERY verbinden, um die einzelnen SQL-Anweisungen zu sehen, die für eine bestimmte VACUUM-Transaktion ausgeführt wurden. Wenn Sie die gesamte Datenbank bereinigen, wird jede Tabelle in einer separaten Transakti on bereinigt.
table_id	integer	Die Tabellen-ID.
status	character(30)	Der Status der VACUUM-Operation für jede Tabelle: Folgende Werte sind möglich: • **`Started`** • **`Started Delete Only`** • **`Started Delete Only (Sorted >= nn%)`** Nur die Löschen-Phase wurde für eine VACUUM FULL-Aktion gestartet. Die Sortieren-Phase wurde übergangen, weil die Tabelle bereits bei oder über dem Sortierschwellenwert sortiert war. • **`Started Sort Only`** • **`Started Ranged Partition`** • **`Started Reindex`** • **`Finished`** Zeitpunkt des Abschlusses der Operation für die Tabelle. Um zu ermitteln, wie lange eine Bereinigu ngsaktion für eine bestimmte Tabelle dauerte, subtrahieren Sie die Startzeit von der Endzeit für eine bestimmte Transaktions- und Tabellen-ID.

Spaltenname	Datentyp	Beschreibung
		• **Skipped** Die Tabelle wurde übergangen, weil sie vollständig sortiert und keine Zeile zur Löschung markiert war. • **Skipped (delete only)** Die Tabelle wurde übergangen, weil DELETE ONLY angegeben und keine Zeile zur Löschung markiert war. • **Skipped (sort only)** Die Tabelle wurde übergangen, weil SORT ONLY angegeben und die Tabelle bereits vollständig sortiert war. • **Skipped (sort only, sorted>=xx%)** Die Tabelle wurde übergangen, weil SORT ONLY angegeben und die Tabelle bereits vollständig bei oder über dem Sortierschwellenwert sortiert war. • **Skipped (0 rows)** Die Tabelle wurde übersprungen, weil sie leer war. • **VacuumBG** Im Hintergrund wurde eine automatische Bereinigungsoperation ausgeführt. Dieser Status wird anderen Status vorangestellt, wenn diese automatisch ausgeführt werden. So hätte beispielsweise eine „Delete only“-Bereinigung automatisch eine Startzeile mit dem Status `[VacuumBG] Started Delete Only`. Für weitere Informationen über den VACUUM-Sortierschwellenwert vgl. VACUUM.

Spaltenname	Datentyp	Beschreibung
rows	bigint	Die tatsächliche Anzahl der Zeilen in der Tabelle zzgl. aller gelöschten Zeilen, die immer noch auf der Festplatte gespeichert sind (und auf die Bereinigung warten). Diese Spalte zeigt die Anzahl vor dem Beginn der Bereinigung für Zeilen mit dem Status **`Started`** und die Anzahl nach dem Abschluss der Bereinigung für Zeilen mit dem Status **`Finished`** an.
sortedrows	integer	Die Anzahl der sortierten Zeilen in der Tabelle. Diese Spalte zeigt die Anzahl vor dem Beginn der Bereinigung für Zeilen mit dem Status **`Started`** in der Statusspalte und die Anzahl nach dem Abschluss der Bereinigung für Zeilen mit dem Status **`Finished`** in der Statusspalte an.
blocks	integer	Die Gesamtzahl der zum Speichern der Tabellendaten verwendeten Datenblöcke vor der Bereinigungsoperation (Zeilen mit dem Status **`Started`**) und nach der Bereinigungsoperation (Spalte **`Finished`**). Jeder Datenblock nimmt 1 MB in Anspruch.
max_merge_partitions	integer	Diese Spalte wird für die Leistungsanalyse verwendet und enthält die maximale Anzahl der Partitionen, die die Bereinigung für die Tabelle pro Iteration der Zusammenführungsphase verarbeiten kann. (Die Bereinigung sortiert die nicht sortierte Region in eine oder mehrere sortierte Partitionen. Je nach der Anzahl der Spalten in der Tabelle und der aktuellen Amazon-Redshift-Konfiguration kann die Zusammenführungsphase eine maximale Anzahl von Partitionen in einer einzelnen Zusammenführungsiteration verarbeiten. Die Zusammenführungsphase funktioniert auch, wenn die Anzahl der sortierten Partitionen die maximale Anzahl der Zusammenführungspartitionen überschreitet, es werden jedoch weitere Zusammenführungsiterationen benötigt.)

Spaltenname	Datentyp	Beschreibung
eventtime	Zeitstempel	Zeitpunkt des Beginns bzw. Endes der Bereinigungsaktion.
reclaimable_rows	bigint	Die Anzahl der zurückgewinnbaren Zeilen für die aktuelle cutoff_xid. Diese Spalte zeigt die von Redshift geschätzte Anzahl der zurückgewinnbaren Zeilen vor Beginn der Bereinigung für Zeilen mit dem Status **`Started`** und die tatsächliche Zahl der zurückgewinnbaren Zeilen, die nach der Bereinigung für Zeilen mit dem Status **`Finished`** verbleiben.
reclaimable_space_mb	bigint	Zurückgewinnbarer Speicherplatz in MB für die aktuelle cutoff_xid. Diese Spalte zeigt die von Redshift geschätzte Menge an zurückgewinnbarem Speicher vor Beginn der Bereinigung für Zeilen mit dem Status **`Started`** und die tatsächliche Menge an zurückgewinnbarem Speicher, die nach der Bereinigung für Zeilen mit dem Status **`Finished`** verbleibt.
cutoff_xid	bigint	Die Cutoff-Transaktions-ID der VACUUM-Operation. Alle Transaktionen nach dem Cutoff sind nicht in der VACUUM-Operation enthalten.
is_recluster	Ganzzahl	Wenn 1 (wahr), hat die VACUUM-Operation den Recluster-Algorithmus ausgeführt, wenn 0 (falsch), nicht.

Beispielabfragen

Die folgende Abfrage ergibt Bereinigungsstatistiken für Tabelle 108313. Die Tabelle wurde nach einer Reihe von Einfügungs- und Löschvorgängen bereinigt.

```
select xid, table_id, status, rows, sortedrows, blocks, eventtime,
                reclaimable_rows, reclaimable_space_mb
from stl_vacuum where table_id=108313 order by eventtime;
```

```
 xid   | table_id | status                  | rows | sortedrows | blocks |  eventtime
         | reclaimable_rows | reclaimable_space_mb
-------+----------+-------------------------+------+------------+--------
+---------------------+-----------------+---------------------
 14294 |   108313 | Started                 | 1950 |        408 |     28 |  2016-05-19
 17:36:01 |              984 |                   17
 14294 |   108313 | Finished                |  966 |        966 |     11 |  2016-05-19
 18:26:13 |                0 |                    0
 15126 |   108313 | Skipped(sorted>=95%)    |  966 |        966 |     11 |  2016-05-19
 18:26:38 |                0 |                    0
```

Zu Beginn der VACUUM-Operation enthielt die Tabelle 1 950 Zeilen in 28 1-MB-Blöcken. Amazon Redshift schätzte, mit einer Bereinigungsoperation 984 Zeilen oder 17 Blöcke Speicherplatz zurückgewinnen zu können.

In der Zeile für den Status „Finished“ (Abgeschlossen) zeigt die Spalte ROWS (Zeilen) einen Wert von 966 an, der Wert der Spalte BLOCKS (Blöcke) ist von 28 auf 11 zurückgegangen. Bei der Bereinigung wurde die geschätzte Menge Speicherplatz zurückgewonnen, sodass nach Abschluss der Bereinigungsoperation keine zurückgewinnbaren Zeilen bzw. kein zurückgewinnbarer Speicherplatz mehr übrig war(en).

In der Sortier-Phase (Transaktion 15126) konnte die Bereinigung die Tabelle übergehen, da die Zeilen in der Reihenfolge des Sortierschlüssels eingegeben wurden.

Das folgende Beispiel zeigt die Statistik für eine SORT ONLY-Bereinigung auf der Tabelle SALES (Tabelle 110116 in diesem Beispiel) nach einer großen INSERT-Operation:

```
vacuum sort only sales;

select xid, table_id, status, rows, sortedrows, blocks, eventtime
from stl_vacuum order by xid, table_id, eventtime;

xid |table_id|     status      | rows  |sortedrows|blocks|     eventtime
----+--------+-----------------+-------+----------+------+--------------------
...
2925| 110116 |Started Sort Only|1379648|   172456 |  132 | 2011-02-24 16:25:21...
2925| 110116 |Finished         |1379648|  1379648 |  132 | 2011-02-24 16:26:28...
```

STL_WINDOW

Analysiert die Abfrageschritte, die Fensterfunktionen ausführen.

STL_WINDOW ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> STL_WINDOW enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Die Nummer, die das Slice angibt, in dem die Abfrage ausgeführt wurde.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
starttime	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .

Spaltenname	Datentyp	Beschreibung
endtime	Zeitstempel	Zeitpunkt in UTC, an dem die Abfrage abgeschlossen wurde. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .
tasknum	Ganzzahl	Nummer des Abfrageaufgabenprozesses, der der Ausführung des Schritts zugeordnet wurde.
rows	bigint	Gesamtzahl der Zeilen, die verarbeitet wurden.
is_diskbased	character(1)	Bei „true (t)“ wurde die Abfrage als festplattenbasierte Operation ausgeführt. Bei „false (f)“ wurde die Abfrage im Arbeitsspeicher ausgeführt.
workmem	bigint	Gesamtgröße des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.

Beispielabfragen

Das folgende Beispiel gibt die Fensterfunktionsergebnisse für Slice 0 und Segment 3 aus.

```
select query, tasknum, rows, is_diskbased, workmem
from stl_window
where slice=0 and segment=3;
```

```
 query | tasknum | rows | is_diskbased | workmem
-------+---------+------+--------------+----------
 86326 |      36 | 1857 | f            | 95256616
   705 |      15 | 1857 | f            | 95256616
 86399 |      27 | 1857 | f            | 95256616
   649 |      10 |    0 | f            | 95256616
(4 rows)
```

STL_WLM_ERROR

Zeichnet alle WLM-bezogenen Fehler bei ihrem Auftreten auf.

STL_WLM_ERROR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
recordtime	timestamp	Zeitpunkt, zu dem der Fehler aufgetreten ist.
pid	integer	ID des Prozesses, der den Fehler generiert hat.
error_string	character(256)	Fehlerbeschreibung.

STL_WLM_RULE_ACTION

Zeigt Einzelheiten zu Aktionen aufgrund von WLM-Abfrageüberwachungsregeln im Zusammenhang mit benutzerdefinierten Abfragen auf. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

STL_WLM_RULE_ACTION ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Der Benutzer, der die Abfrage ausgeführt hat.
query	integer	Abfrage-ID.

Spaltenname	Datentyp	Beschreibung
service_class	integer	ID für die Service-Klasse. Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Service-Klassen über 5 sind benutzerdefinierte Warteschlangen.
Regel	character(256)	Name einer Abfrageüberwachungsregel.
Aktion	character(256)	Resultierende Aktion. Die möglichen Werte lauten wie folgt: • log • Hop (Neuzuweisung) • Hop (Neustart) • Abbruch • change_query_priority • Keine Der Wert none gibt an, dass die Prädikate der Regel erfüllt waren, dass die Aktion jedoch von einer anderen Regel mit höherem Schweregrad übergangen wurde.
recordtime	Zeitstempel	Uhrzeit, zu der die Aktion in UTC protokolliert wurde.
action_value	character(256)	Wenn für `action` `change_query_priority` angegeben ist, sind die Werte `highest`, `high`, `normal`, `low` und `lowest` möglich. Wenn für `action` `log`, `hop`, oder `abort` angegeben ist, ist der Wert leer.
service_class_name	character(64)	Der Name der Serviceklasse.

Beispielabfragen

Das folgende Beispiel findet Abfragen, die von einer Abfrageüberwachungsregel beendet wurden.

```
Select query, rule
from stl_wlm_rule_action
where action = 'abort'
order by query;
```

STL_WLM_QUERY

Enthält einen Datensatz für jeden Ausführungsversuch einer Abfrage in einer von WLM berücksichtigten Service-Klasse.

STL_WLM_QUERY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	integer	Transaktions-ID der Abfrage oder Unterabfrage,
Aufgabe	integer	ID zur Verfolgung einer Abfrage durch den Workload Manager. Kann mit mehreren Abfrage-IDs verbunden werden. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
query	integer	Abfrage-ID. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
service_class	integer	ID für die Service-Klasse. Eine Liste von Servicekl assen-IDs finden Sie unter WLM-Serviceklassen-IDs.

Spaltenname	Datentyp	Beschreibung
slot_count	Ganzzahl	Anzahl der WLM-Abfrageslots, die eine Abfrage verwendet, entsprechend der für die Warteschlange festgelegten Gleichzeitigkeitsstufe. Standard = 1. Weitere Informationen finden Sie unter wlm_query_slot_count.
service_class_start_time	timestamp	Zeitpunkt, zu dem die Abfrage der Service-Klasse zugewiesen wurde. Diese Zeit ist in der UTC-Zeitzone.
queue_start_time	timestamp	Zeitpunkt, zu dem die Abfrage in die Warteschlange für die Service-Klasse gesetzt wurde. Diese Zeit ist in der UTC-Zeitzone.
queue_end_time	timestamp	Zeitpunkt, zu dem die Abfrage die Warteschlange für die Service-Klasse verlassen hat. Diese Zeit ist in der UTC-Zeitzone.
total_queue_time	bigint	Gesamtzahl der Mikrosekunden, die die Abfrage in der Warteschlange verbracht hat
exec_start_time	timestamp	Zeitpunkt, zu dem die Ausführung der Abfrage in der Service-Klasse begann. Diese Zeit ist in der UTC-Zeitzone.
exec_end_time	timestamp	Zeitpunkt, zu dem die Ausführung der Abfrage in der Service-Klasse abgeschlossen wurde. Diese Zeit ist in der UTC-Zeitzone.
total_exec_time	bigint	Anzahl der Mikrosekunden, für die die Abfrage ausgeführt wurde.
service_class_end_time	timestamp	Zeitpunkt, zu dem die Abfrage die Service-Klasse verließ. Diese Zeit ist in der UTC-Zeitzone.
final_state	character(16)	Reserviert für Systemverwendung.

Spaltenname	Datentyp	Beschreibung
est_peak_mem	bigint	Reserviert für die Systemverwendung.
query_priority	char(20)	Die Priorität der Abfrage. Mögliche Werte sind n/a, lowest, low, normal, high und highest, wobei n/a bedeutet, dass die Abfragepriorität nicht unterstützt wird.
service_class_name	character(64)	Der Name der Serviceklasse. Weitere Informationen zu Serviceklassen finden Sie unter WLM-Systemtabellen und Ansichten.

Beispielabfragen

Anzeige der durchschnittlichen Abfragewartezeit und Ausführung

Die folgenden Abfragen zeigen die aktuelle Konfiguration für Service-Klassen über 4 an. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

Die folgende Abfrage gibt die durchschnittliche Zeit (in Mikrosekunden) aus, die jede Abfrage in Abfragewarteschlangen und Ausführung für jede Service-Klasse verbracht hat.

```
select service_class as svc_class, count(*),
avg(datediff(microseconds, queue_start_time, queue_end_time)) as avg_queue_time,
avg(datediff(microseconds, exec_start_time, exec_end_time )) as avg_exec_time
from stl_wlm_query
where service_class > 4
group by service_class
order by service_class;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
 svc_class | count | avg_queue_time | avg_exec_time
-----------+-------+----------------+---------------
         5 | 20103 |              0 |         80415
         5 |  3421 |          34015 |        234015
         6 |    42 |              0 |        944266
         7 |   196 |           6439 |       1364399
(4 rows)
```

Anzeige der maximalen Abfragewartezeit und Ausführung

Die folgende Abfrage gibt die maximale Zeit (in Mikrosekunden) aus, die jede Abfrage in Abfragewarteschlangen und Ausführung für jede Service-Klasse verbracht hat.

```
select service_class as svc_class, count(*),
max(datediff(microseconds, queue_start_time, queue_end_time)) as max_queue_time,
max(datediff(microseconds, exec_start_time, exec_end_time )) as max_exec_time
from stl_wlm_query
where svc_class > 5
group by service_class
order by service_class;
```

```
 svc_class | count | max_queue_time | max_exec_time
-----------+-------+----------------+---------------
         6 |    42 |              0 |       3775896
         7 |   197 |          37947 |      16379473
(4 rows)
```

STV-Tabellen für Snapshot-Daten

STV-Tabellen sind virtuelle Systemtabellen, die Snapshots der aktuellen Systemdaten enthalten.

Themen

- STV_ACTIVE_CURSORS
- STV_BLOCKLIST
- STV_CURSOR_CONFIGURATION
- STV_DB_ISOLATION_LEVEL
- STV_EXEC_STATE
- STV_INFLIGHT
- STV_LOAD_STATE
- STV_LOCKS
- STV_ML_MODEL_INFO
- STV_MV_DEPS
- STV_MV_INFO
- STV_NODE_STORAGE_CAPACITY
- STV_PARTITIONS

- STV_QUERY_METRICS
- STV_RECENTS
- STV_SESSIONS
- STV_SLICES
- STV_STARTUP_RECOVERY_STATE
- STV_TBL_PERM
- STV_TBL_TRANS
- STV_WLM_CLASSIFICATION_CONFIG
- STV_WLM_QMR_CONFIG
- STV_WLM_QUERY_QUEUE_STATE
- STV_WLM_QUERY_STATE
- STV_WLM_QUERY_TASK_STATE
- STV_WLM_SERVICE_CLASS_CONFIG
- STV_WLM_SERVICE_CLASS_STATE
- STV_XRESTORE_ALTER_QUEUE_STATE

STV_ACTIVE_CURSORS

STV_ACTIVE_CURSORS zeigt Details zu aktuell offenen Cursors an. Weitere Informationen finden Sie unter DECLARE.

STV_ACTIV_CURSORS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten. Ein Benutzer kann nur Cursors sehen, die er selbst geöffnet hat. Ein Superuser kann alle Cursors sehen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
Name	character (256)	Name des Cursors.

Spaltenname	Datentyp	Beschreibung
xid	bigint	Transaktionskontext.
pid	integer	Leader-Prozess, der die Abfrage ausführt.
starttime	timestam	Zeitpunkt, zu dem der Cursor deklariert wurde.
row_count	bigint	Anzahl der Zeilen im Cursor-Ergebnissatz.
byte_count	bigint	Anzahl der Bytes im Cursor-Ergebnissatz.
fetched_rows	bigint	Anzahl der aktuell vom Cursor-Ergebnissatz abgerufenen Zeilen.

STV_BLOCKLIST

STV_BLOCKLIST enthält die Anzahl der 1 MB-Festplattenblöcke, die von jedem Slice, jeder Tabelle oder Spalte in einer Datenbank verwendet werden.

Verwenden Sie aggregierte Abfragen mit STV_BLOCKLIST, wie die folgenden Beispiele zeigen, um die Anzahl der 1 MB-Festplattenblöcke zu bestimmen, die pro Datenbank, Tabelle, Slice oder Spalte zugewiesen sind. Sie können auch STV_PARTITIONS verwenden, um zusammenfassende Informationen zur Festplattennutzung anzuzeigen.

STV_BLOCKLIST ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
slice	integer	Knoten-Slice.
col	integer	Null-basierter Index für die Spalte. Jeder Tabelle, die Sie erstellen , werden diese drei verborgenen Spalten angehängt: INSERT_XID, DELETE_XID, und ROW_ID (OID). Eine Tabelle mit drei vom Benutzer

Spaltenname	Datentyp	Beschreibung
		definierten Spalten hat tatsächlich sechs Spalten, und die benutzerdefinierten Spalten haben die interne Nummerierung 0, 1 und 2. Die Spalten INSERT_XID, DELETE_XID, und ROW_ID haben in diesem Beispiel die Nummern 3, 4 und 5.
tbl	integer	Tabellen-ID für die Datenbanktabelle.
blocknum	integer	ID für den Datenblock.
num_values	integer	Anzahl der enthaltenen Werte für den Block.
extended_limits	integer	Zur internen Verwendung.
minvalue	bigint	Minimaler Datenwert des Blocks. Speichert die ersten acht Zeichen als 64-Bit-Ganzzahl für nicht-numerische Daten. Zum Scannen von Festplatten verwendet.
maxvalue	bigint	Maximaler Datenwert des Blocks. Speichert die ersten acht Zeichen als 64-Bit-Ganzzahl für nicht-numerische Daten. Zum Scannen von Festplatten verwendet.
sb_pos	integer	Interne Amazon-Redshift-Kennung für die Superblock-Position auf der Festplatte.
pinned	integer	Ob der Block im Rahmen des Pre-Load-Vorgangs im Speicher fixiert ist. 0 = false; 1 = true. Standard ist „false“.
on_disk	integer	Ob der Block automatisch auf der Festplatte gespeichert wird. 0 = false; 1 = true. Standard ist „false“.
modified	integer	Ob der Block modifiziert wurde. 0 = false; 1 = true. Standard ist „false“.
hdr_modified	integer	Ob der Block-Header modifiziert wurde. 0 = false; 1 = true. Standard ist „false“.

Spaltenname	Datentyp	Beschreibung
unsorted	integer	Ob der Block unsortiert ist. 0 = false; 1 = true. Der Standardwert ist „true“.
tombstone	integer	Zur internen Verwendung.
preferred_diskno	integer	Nummer der Festplatte, auf der sich der Block befinden sollte, sofern die Festplatte nicht ausgefallen ist. Sobald die Festplatte festgelegt ist, wird der Block wieder auf diese Festplatte übertragen.
temporary	integer	Ob der Block temporäre Daten enthält, etwa aus einer temporären Tabelle oder aus Abfragezwischenergebnissen. 0 = false; 1 = true. Standard ist „false“.
newblock	integer	Zeigt an, ob ein Block neu ist (true) oder noch nie einem Commit-Vorgang zu einer Festplatte unterzogen wurde (false). 0 = false; 1 = true.
num_readers	integer	Anzahl der Referenzen auf jedem Block.
flags	integer	Interne Amazon-Redshift-Flags für den Block-Header.

Beispielabfragen

STV_BLOCKLIST enthält eine Zeile pro zugewiesenem Festplattenblock, so dass eine Abfrage mit Auswahl aller Zeilen möglicherweise eine sehr große Zahl von Zeilen ausgibt. Wir empfehlen, nur aggregierte Abfragen mit STV_BLOCKLIST zu verwenden.

Die Ansicht SVV_DISKUSAGE bietet ähnliche Informationen in einem benutzerfreundlichen Format – das folgende Beispiel illustriert eine Verwendung der STV_BLOCKLIST-Tabelle.

Um die Anzahl der 1 MB-Blöcke zu bestimmen, die jede Spalte in der VENUE-Tabelle enthält, geben Sie die folgende Abfrage ein:

```
select col, count(*)
from stv_blocklist, stv_tbl_perm
where stv_blocklist.tbl = stv_tbl_perm.id
and stv_blocklist.slice = stv_tbl_perm.slice
```

```
and stv_tbl_perm.name = 'venue'
group by col
order by col;
```

Diese Abfrage gibt die Anzahl der jeder Spalte in der VENUE-Tabelle zugeordneten 1 MB-Blöcke zurück, vgl. die folgenden Beispieldaten:

```
 col | count
-----+-------
   0 |  4
   1 |  4
   2 |  4
   3 |  4
   4 |  4
   5 |  4
   7 |  4
   8 |  4
(8 rows)
```

Die folgende Abfrage zeigt, ob die Tabellendaten tatsächlich über alle Slices verteilt sind:

```
select trim(name) as table, stv_blocklist.slice, stv_tbl_perm.rows
from stv_blocklist,stv_tbl_perm
where stv_blocklist.tbl=stv_tbl_perm.id
and stv_tbl_perm.slice=stv_blocklist.slice
and stv_blocklist.id > 10000 and name not like '%#m%'
and name not like 'systable%'
group by name, stv_blocklist.slice, stv_tbl_perm.rows
order by 3 desc;
```

Diese Abfrage erstellt die folgende Beispielausgabe, die die gleichmäßige Verteilung der Daten für die Tabelle mit den meisten Zeilen anzeigt:

```
table   | slice | rows
----------+-------+-------
listing  |    13 | 10527
listing  |    14 | 10526
listing  |     8 | 10526
listing  |     9 | 10526
listing  |     7 | 10525
listing  |     4 | 10525
```

```
listing  |    17 | 10525
listing  |    11 | 10525
listing  |     5 | 10525
listing  |    18 | 10525
listing  |    12 | 10525
listing  |     3 | 10525
listing  |    10 | 10525
listing  |     2 | 10524
listing  |    15 | 10524
listing  |    16 | 10524
listing  |     6 | 10524
listing  |    19 | 10524
listing  |     1 | 10523
listing  |     0 | 10521
...
(180 rows)
```

Die folgende Abfrage bestimmt, ob ein Tombstone-Block auf die Festplatte geschrieben wurde:

```
select slice, col, tbl, blocknum, newblock
from stv_blocklist
where  tombstone > 0;

slice | col |   tbl  | blocknum | newblock
-------+-----+--------+----------+----------
4     |  0  | 101285 |    0     |   1
4     |  2  | 101285 |    0     |   1
4     |  4  | 101285 |    1     |   1
5     |  2  | 101285 |    0     |   1
5     |  0  | 101285 |    0     |   1
5     |  1  | 101285 |    0     |   1
5     |  4  | 101285 |    1     |   1
...
(24 rows)
```

STV_CURSOR_CONFIGURATION

STV_CURSOR_CONFIGURATION zeigt Einschränkungen der Cursorkonfiguration an. Weitere Informationen finden Sie unter Einschränkungen für Cursors.

STV_CURSOR_CONFIGURATION ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
current_cursor_count	integer	Anzahl der aktuell geöffneten Cursors.
max_diskspace_usable	integer	Größe des für Cursors verfügbaren Festplattenspeicherplatzes, in Megabyte. Diese Einschränkung basiert auf der maximalen Größe des Cursor-Ergebnissatzes für den Cluster.
current_diskspace_used	integer	Größe des derzeit für Cursors verwendeten Festplattenspeicherplatzes, in Megabyte.

STV_DB_ISOLATION_LEVEL

STV_DB_ISOLATION_LEVEL zeigt die aktuelle Isolationsstufe für Datenbanken an. Weitere Hinweise zu Isolationsstufen finden Sie unter CREATE DATABASE.

STV_DB_ISOLATION_LEVEL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
db_name	character (128)	Der Datenbankname.
isolation_level	character (20)	Die Isolationsstufe der Datenbank. Mögliche Werte sind `Serializable` und `Snapshot Isolation` .

STV_EXEC_STATE

Verwenden Sie die Tabelle STV_EXEC_STATE, um Informationen zu Abfragen und Abfrageschritten abzurufen, die derzeit aktiv auf Datenverarbeitungsknoten ausgeführt werden.

Diese Informationen werden normalerweise nur zur Behebung technischer Probleme verwendet. Die Ansichten SVV_QUERY_STATE und SVL_QUERY_SUMMARY extrahieren ihre Informationen aus STV_EXEC_STATE.

STV_EXEC_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Knoten-Slice, bei dem der Schritt abgeschlossen wurde.
segment	integer	Segment der Abfrage, das ausgeführt wurde. Ein Abfragesegment ist eine Serie von Schritten.
Schritt	integer	Schritt des Abfragesegments, der abgeschlossen wurde. Ein Schritt ist die kleinste Einheit, die eine Abfrage ausführt.
starttime	timestamp	Zeit der Ausführung des Schritts.
currenttime	timestamp	Aktuelle Uhrzeit.

Spaltenname	Datentyp	Beschreibung
tasknum	integer	Abfrageaufgabenprozess, der zum Abschließen des Schritts zugewiesen wird.
rows	bigint	Anzahl der verarbeiteten Zeilen.
bytes	bigint	Anzahl der verarbeiteten Bytes.
label	char(256)	Schritt-Etikett, bestehend aus dem Namen eines Abfrageschritts und, falls anwendbar, der Tabellen-ID und dem Tabellennamen (zum Beispiel)., `scan tbl=100448 name =user`). Dreistellige Tabellen-IDs beziehen sich normalerweise auf Scans temporärer Tabellen. Wenn Sie `tbl=0` sehen, bezieht sich dies auf einen Scan eines konstanten Werts.
is_diskbased	char(1)	Ob dieser Schritt der Abfrage als festplattenbasierte Operation ausgeführt wurde: true (**t**) oder false (**f**). Nur bestimmte Schritte, wie etwa Hash-, Sortierungs- oder Aggregierungsschritte, können auf die Festplatte übertragen werden. Viele Arten von Schritten werden immer im Arbeitsspeicher ausgeführt.
workmem	bigint	Größe des Arbeitsspeichers in Byte, der dem Schritt zugewiesen wurde.
num_parts	integer	Anzahl der Partitionen, in die eine Hash-Tabelle während eines Hash-Schrittes unterteilt wurde. Ein positiver Wert in dieser Spalte bedeutet nicht, dass der Hash-Schritt als festplattenbasierte Operation ausgeführt wurde. Prüfen Sie den Wert in der Spalte IS_DISKBASED, um zu sehen, ob der Hash-Schritt festplattenbasiert war.
is_rrscan	char(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein Scan mit Bereichseinschränkung durchgeführt wurde. Der Standardwert ist „false“ (**f**).

Spaltenname	Datentyp	Beschreibung
is_delayed_scan	char(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein verzögerter Scan durchgeführt wurde. Der Standardwert ist „false“ (**f**).

Beispielabfragen

Anstelle einer direkten Abfrage von STV_EXEC_STATE empfiehlt Amazon Redshift die Abfrage von SVL_QUERY_SUMMARY oder SVV_QUERY_STATE, um die Informationen in STV_EXEC_STATE in einem benutzerfreundlicheren Format zu erhalten. Für weitere Einzelheiten vgl. die Dokumentation der Tabelle SVL_QUERY_SUMMARY oder SVV_QUERY_STATE.

STV_INFLIGHT

Verwenden Sie die Tabelle STV_INFLIGHT, um festzustellen, welche Abfragen derzeit auf dem Cluster ausgeführt werden. Bei der Fehlerbehebung ist dies hilfreich, um den Status lang andauernder Abfragen zu überprüfen.

STV_INFLIGHT zeigt keine Abfragen nur mit Führungsknoten. Weitere Informationen finden Sie unter Exklusive Führungsknotenfunktionen. STV_INFLIGHT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Fehlerbehebung mit STV_INFLIGHT

Wenn Sie STV_INFLIGHT verwenden, um Fehler im Hinblick auf die Leistung einer Abfrage oder einer Sammlung von Abfragen zu beheben, beachten Sie Folgendes:

- Lang andauernde offene Transaktionen erhöhen im Allgemeinen die Last. Diese offenen Transaktionen können zu längeren Laufzeiten für andere Abfragen führen.
- Lang andauernde COPY- und ETL-Aufträge können sich auf andere Abfragen auswirken, die im Cluster ausgeführt werden, wenn sie viele Rechenressourcen in Anspruch nehmen. In den meisten

Fällen erhöht das Verlagern dieser lang andauernden Aufträgen auf Zeiten geringer Auslastung die Leistung für Berichts- oder Analyse-Workloads.

- Es gibt Ansichten, die verwandte Informationen zu STV_INFLIGHT bereitstellen. Dazu gehören STL_QUERYTEXT, die den Abfragetext für SQL-Befehle erfasst, und SVV_QUERY_INFLIGHT, die STV_INFLIGHT mit STL_QUERYTEXT verbindet. Sie können auch STV_RECENTS mit STV_INFLIGHT zur Fehlerbehebung verwenden. STV_RECENTS kann beispielsweise angeben, ob sich bestimmte Abfragen im Status Wird ausgeführt oder Fertig befinden. Wenn Sie diese Informationen mit den Ergebnissen von STV_INFLIGHT kombinieren, erhalten Sie mehr Informationen über die Eigenschaften einer Abfrage und die Auswirkungen auf die Rechenressourcen.

Sie können laufende Abfragen auch mithilfe der Amazon-Redshift-Konsole überwachen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
slice	integer	Slice, auf dem die Abfrage ausgeführt wird.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist dieses Feld leer.
xid	bigint	Transaktions-ID.
pid	integer	Prozess-ID. Alle Abfragen in einer Sitzung werden in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen. Mit dieser Spalte können Sie eine Verbindung zur Tabelle STL_ERROR herstellen.

Spaltenname	Datentyp	Beschreibung
starttime	timestamp	Der Zeitpunkt des Beginns der Abfrage.
Text	character(100)	Abfragetext, auf 100 Zeichen verkürzt, wenn die Anweisung diesen Grenzwert überschreitet.
suspended	integer	Ob die Abfrage ausgesetzt wurde: 0 = false; 1 = true.
insert_pristine	integer	Ob Schreibabfragen ausgeführt werden können/konnten, während die aktuelle Abfrage läuft/lief. 1 = keine Schreibabfragen erlaubt. 0 = Schreibabfragen erlaubt. Diese Spalte dient zur Verwendung beim Debugging.
concurrency_scaling_status	integer	Gibt an, ob die Abfrage auf dem Haupt-Cluster oder einem Nebenläufigkeitsskalierungs-Cluster ausgeführt wurde. Die möglichen Werte lauten wie folgt: 0 – Wurde auf dem Haupt-Cluster ausgeführt 1 – Wurde auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt

Beispielabfragen

Um alle derzeit auf der Datenbank ausgeführten Abfragen anzuzeigen, geben Sie die folgende Abfrage ein:

```
select * from stv_inflight;
```

Die Beispielausgabe unten zeigt zwei derzeit ausgeführte Abfragen, einschließlich der STV_INFLIGHT-Abfrage selbst und einer Abfrage, die mit einem Skript mit der Bezeichnung ausgeführt wurde `avgwait.sql`:

```
select slice, query, trim(label) querylabel, pid,
starttime, substring(text,1,20) querytext
from stv_inflight;
```

```
slice|query|querylabel | pid |        starttime         |      querytext
-----+-----+-----------+-----+--------------------------+-------------------
1011 |  21 |           | 646 |2012-01-26 13:23:15.645503|select slice, query,
1011 |  20 |avgwait.sql| 499 |2012-01-26 13:23:14.159912|select avg(datediff(
(2 rows)
```

Die folgende Abfrage wählt mehrere Spalten aus, einschließlich concurrency_scaling_status. Diese Spalte gibt an, ob Abfragen an den Nebenläufigkeitsskalierungs-Cluster gesendet werden. Wenn der Wert für einige Ergebnisse 1 lautet, ist dies ein Hinweis darauf, dass Rechenressourcen für die Nebenläufigkeitsskalierung verwendet werden. Weitere Informationen finden Sie unter Arbeiten mit Nebenläufigkeitsskalierung.

```
select userid,
query,
pid,
starttime,
text,
suspended,
concurrency_scaling_status
 from STV_INFLIGHT;
```

Die Beispielausgabe zeigt, wie eine Abfrage an den Nebenläufigkeitsskalierungs-Cluster gesendet wird.

```
 query  | pid     |        starttime           |   text                | suspended
 |  concurrency_scaling_status
--------+---------
+---------------------------|-----------------------|--------------|-----------------------
1234567 | 123456  | 2012-01-26 13:23:15.645503 | select userid, query...  0
    1
2345678 | 234567  | 2012-01-26 13:23:14.159912 | select avg(datediff(...  0
    0
(2 rows)
```

Weitere Tipps zur Fehlerbehebung bei der Abfrageleistung finden Sie unter Fehlerbehebung bei Abfragen.

STV_LOAD_STATE

Verwenden Sie die Tabelle STV_LOAD_STATE, um Informationen zum aktuellen Status laufender COPY-Anweisungen zu finden.

Der COPY-Befehl aktualisiert diese Tabelle nach jeder Million geladener Datensätze.

STV_LOAD_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
Sitzung	integer	Sitzungs-PID des Prozesses, der den Ladevorgang durchführt.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Knoten-Slice-Nummer.
pid	integer	Prozess-ID. Alle Abfragen in einer Sitzung werden in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen.
recordtime	timestamp	Zeitpunkt der Protokollierung des Datensatzes.
bytes_to_load	bigint	Gesamtzahl der Bytes, die von diesem Slice zu laden sind. Der Wert ist 0, wenn die zu ladenden Dateien komprimiert sind
bytes_loaded	bigint	Anzahl der Bytes, die von diesem Slice geladen wurden. Wenn die geladenen Daten komprimiert sind, ist dies die Anzahl der Bytes, die nach der Dekomprimierung der Daten geladen wurden.
bytes_to_load_compressed	bigint	Gesamtzahl der Bytes komprimierter Daten, die von diesem Slice zu laden sind. Der Wert ist 0, wenn die zu ladenden Dateien nicht komprimiert sind.

Spaltenname	Datentyp	Beschreibung
bytes_loaded_compressed	bigint	Anzahl der Bytes komprimierter Daten, die von diesem Slice zu laden sind. Der Wert ist 0, wenn die zu ladenden Dateien nicht komprimiert sind.
lines	integer	Anzahl der Zeilen, die von diesem Slice geladen wurden.
num_files	integer	Anzahl der Dateien, die von diesem Slice zu laden sind.
num_files_complete	integer	Anzahl der Dateien, die von diesem Slice geladen wurden.
current_file	character (256)	Name der Datei, die von diesem Slice geladen wird.
pct_complete	integer	Von diesem Slice abgeschlossene Datenladung, in Prozent.

Beispielabfrage

Um den Fortschritt jedes Slices für einen COPY-Befehl anzuzeigen, geben Sie die folgende Abfrage ein. Dieses Beispiel verwendet die Funktion PG_LAST_COPY_ID(), um Informationen für den letzten COPY-Befehl abzurufen.

```
select slice , bytes_loaded, bytes_to_load , pct_complete from stv_load_state where
 query = pg_last_copy_id();

 slice | bytes_loaded | bytes_to_load | pct_complete
-------+--------------+---------------+--------------
     2 |            0 |             0 |            0
     3 |     12840898 |      39104640 |           32
(2 rows)
```

STV_LOCKS

Verwenden Sie die Tabelle STV_LOCKS zur Anzeige derzeitiger Aktualisierungen in den Tabellen der Datenbank.

Amazon Redshift sperrt Tabellen, um zu verhindern, dass zwei Benutzer gleichzeitig dieselbe Tabelle aktualisieren. Während die Tabelle STV_LOCKS alle derzeitigen Tabellenaktualisierungen anzeigt,

fragen Sie die Tabelle STL_TR_CONFLICT ab, um ein Protokoll von Sperrkonflikten anzuzeigen. Verwenden Sie die Ansicht SVV_TRANSACTIONS, um offene Transaktionen und Probleme mit Sperrkonflikten anzuzeigen.

STV_LOCKS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
table_id	bigint	Tabellen-ID für die Tabelle, die die Sperre abruft.
last_commit	timestamp	Zeitstempel des letzten Commit in der Tabelle.
last_update	timestamp	Zeitstempel der letzten Aktualisierung für die Tabelle.
lock_owner	bigint	Mit der Sperre verbundene Transaktions-ID.
lock_owner_pid	bigint	ID des Prozesses, der mit der Sperre verbunden ist.
lock_owner_start_ts	timestamp	Zeitstempel für die Startzeit der Transaktion.
lock_owner_end_ts	timestamp	Zeitstempel für die Endzeit der Transaktion.
lock_status	character (22)	Status des Prozesses: Warten auf eine Sperre oder mit Sperre.

Beispielabfrage

Um alle Sperren in aktuellen Transaktionen anzuzeigen, geben Sie den folgenden Befehl ein:

```
select table_id, last_update, lock_owner, lock_owner_pid from stv_locks;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück, die drei derzeit aktive Sperren zeigt:

```
 table_id |        last_update         | lock_owner | lock_owner_pid
----------+----------------------------+------------+----------------
100004  | 2008-12-23 10:08:48.882319 |       1043 |           5656
100003  | 2008-12-23 10:08:48.779543 |       1043 |           5656
100140  | 2008-12-23 10:08:48.021576 |       1043 |           5656
(3 rows)
```

STV_ML_MODEL_INFO

Informationen zum aktuellen Zustand des Machine-Learning-Modells.

STV_ML_MODEL_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
schema_name	char(128)	Der Namespace des Modells.
user_name	char(128)	Der Besitzer des Modells.
model_name	char(128)	Der Name des Modells
life_cycle	char(20)	Der Lebenszyklusstatus des Modells.
is_refreshable	integer	Der Status des Modells, ob es aktualisiert werden kann, ob ursprüngliche Tabellen und Spalten in der Trainingsabfrage noch vorhanden sind und ob der Benutzer weiterhin über die Berechtigungen für sie verfügt. Mögliche Werte sind: 1 (aktualis ierbar) und 0 (nicht aktualisierbar).
model_state	char(128)	Der aktuelle Status des Modells.

Beispielabfrage

Die folgende Abfrage zeigt den aktuellen Status von Machine-Learning-Modellen an.

```
SELECT schema_name, model_name, model_state
FROM stv_ml_model_info;

 schema_name |        model_name             |             model_state
-------------+-------------------------------+-------------------------------------
 public      | customer_churn_auto_model     | Train Model On SageMaker In Progress
 public      | customer_churn_xgboost_model | Model is Ready
(2 row)
```

STV_MV_DEPS

Die Tabelle STV_MV_DEPS zeigt die Abhängigkeiten materialisierter Ansichten von anderen materialisierten Ansichten in Amazon Redshift.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

STV_MV_DEPS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
db_name	char(128)	Die Datenbank, die die angegebene materialisierte Ansicht enthält.
schema	char(128)	Das Schema der materialisierten Ansicht.
Name	char(128)	Der Name der materialisierten Ansicht.
ref_schema	char(128)	Das Schema der materialisierten Ansicht, von dem diese materialisierte Ansicht abhängt.
ref_name	char(128)	Der Name der materialisierten Ansicht, von dem diese materialisierte Ansicht abhängt.
ref_database_name	char(128)	Der Name der Datenbank, von der diese materialisierte Ansicht abhängt.

Beispielabfrage

Die folgende Abfrage gibt eine Ausgabezeile zurück, die angibt, dass die materialisierte Ansicht `mv_over_foo` die materialisierte Ansicht `mv_foo` in ihrer Definition als Abhängigkeit verwendet.

```
CREATE SCHEMA test_ivm_setup;
CREATE TABLE test_ivm_setup.foo(a INT);
CREATE MATERIALIZED VIEW test_ivm_setup.mv_foo AS SELECT * FROM test_ivm_setup.foo;
CREATE MATERIALIZED VIEW test_ivm_setup.mv_over_foo AS SELECT * FROM
 test_ivm_setup.mv_foo;

SELECT * FROM stv_mv_deps;

 db_name | schema          | name        |   ref_schema   | ref_name |
 ref_database_name
---------+-----------------+-------------+----------------+----------
+------------------
 dev     | test_ivm_setup  | mv_over_foo | test_ivm_setup | mv_foo   | dev
```

STV_MV_INFO

Die Tabelle STV_MV_INFO enthält eine Zeile für jede materialisierte Ansicht, Angaben, ob die Daten veraltet sind, und Statusinformationen.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

STV_MV_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
db_name	char(128)	Die Datenbank, die die materialisierte Ansicht enthält.
schema	char(128)	Das Schema der Datenbank.
Name	char(128)	Der Name der materialisierten Ansicht.

Spaltenname	Datentyp	Beschreibung
updated_upto_xid	bigint	Zur internen Verwendung reserviert.
is_stale	char(1)	Ein `t` gibt an, dass die materialisierte Ansicht veraltet ist. Eine veraltete materialisierte Ansicht ist eine, bei der zwar die Basistabellen aktualisiert wurden, aber nicht die materialisierte Ansicht. Die Informationen sind möglicherweise nicht korrekt, wenn seit dem letzten Neustart keine Aktualisierung durchgeführt wurde. Die Spalte `is_stale` ist immer auf `t` gesetzt, wenn die materialisierte Ansicht von einer veränderlichen Funktion abhängt. Eine veränderliche Funktion gibt ein anderes Ergebnis zurück, wenn dasselbe Argument oder dieselben Argumente angegeben werden. Beispielsweise sind die meisten Funktionen, die ein Datum oder einen Zeitstempel zurückgeben, veränderliche Funktionen.
owner_user_name	char(128)	Der Benutzer, dem die materialisierte Ansicht gehört.

Spaltenname	Datentyp	Beschreibung
state	integer	Der Status der materialisierten Ansicht wie folgt: • 0 – Die materialisierte Ansicht wird vollständig neu berechnet, wenn sie aktualisiert wird. • 1 – Die materialisierte Ansicht ist inkrementell. • 101 – Die materialisierte Ansicht kann aufgrund einer gelöschten Spalte nicht aktualisiert werden. Diese Einschränkung gilt auch dann, wenn die Spalte nicht in der materialisierten Ansicht verwendet wird. • 102 – Die materialisierte Ansicht kann aufgrund eines geänderten Spaltentyps nicht aktualisiert werden. Diese Einschränkung gilt auch dann, wenn die Spalte nicht in der materialisierten Ansicht verwendet wird. • 103 – Die materialisierte Ansicht kann aufgrund einer umbenannten Tabelle nicht aktualisiert werden. • 104 – Die materialisierte Ansicht kann aufgrund einer umbenannten Spalte nicht aktualisiert werden. Diese Einschränkung gilt auch dann, wenn die Spalte nicht in der materialisierten Ansicht verwendet wird. • 105 – Die materialisierte Ansicht kann aufgrund eines umbenannten Schemas nicht aktualisiert werden.
autorewrite	char(1)	Ein `t` gibt an, dass die materialisierte Ansicht für das automatische Umschreiben von Abfragen berechtigt ist.
autorefresh	char(1)	Ein `t` gibt an, dass die materialisierte Ansicht automatisch aktualisiert werden kann.

Beispielabfrage

Führen Sie die folgende Abfrage aus, um den Status aller materialisierten Ansichten anzuzeigen.

```
select * from stv_mv_info;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück.

```
 db_name |       schema       |   name   | updated_upto_xid | is_stale | owner_user_name
 | state | autorefresh | autorewrite
---------+--------------------+---------+------------------+----------
+-----------------+-------+-------------+------------
 dev     | test_ivm_setup     | mv       |             1031 | f        | catch-22
 |     1 |           1 |           0
 dev     | test_ivm_setup     | old_mv   |              988 | t        | lotr
 |     1 |           0 |           1
```

STV_NODE_STORAGE_CAPACITY

Die Tabelle STV_NODE_STORAGE_CAPACITY zeigt Details zur Gesamtspeicherkapazität und der gesamten genutzten Kapazität für jeden Knoten in einem Cluster an. Sie enthält eine Zeile für jeden Knoten.

STV_NODE_STORAGE_CAPACITY ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
Knoten	integer	Die Knotennummer.
used	integer	Die Anzahl der 1-MB-Festplattenblöcke, die derzeit auf dem Knoten verwendet werden. Bei RA3-Knotentypen umfassen die verwendeten Blöcke sowohl lokal zwischengespeicherte Blöcke als auch Blöcke, die in Amazon S3 gespeichert sind.

Spaltenname	Datentyp	Beschreibung
Kapazität	integer	Die gesamte Speicherkapazität, die für den Knoten in 1-MB-Blöcken bereitgestellt wird. Die Kapazität umfasst Speicherplatz, der von Amazon Redshift auf DC2-Knotentypen für den internen Gebrauch reserviert wird. Die Kapazität ist größer als die nominale Knotenkapazität, d. h. der für Benutzerdaten verfügbaren Knotenspeicher. Bei RA3-Knotentypen entspricht diese Kapazität dem gesamten verwalteten Speicherkontingent für den Cluster. Weitere Informationen zur Kapazität nach Knotentyp finden Sie unter Details zu Knotentypen im Amazon-Redshift-Verwaltungshandbuch.

Beispielabfragen

> **Note**
>
> Die Ergebnisse der folgenden Beispiele variieren je nach den Knotenspezifikationen Ihres Clusters. Fügen Sie SQL SELECT Spalte `capacity` hinzu, um die Kapazität Ihres Clusters abzurufen.

Die folgende Abfrage gibt den belegten Speicherplatz und die Gesamtkapazität in 1-MB-Speicherblöcken zurück. Dieses Beispiel wurde auf einem dc2.8xlarge-Cluster mit zwei Knoten ausgeführt.

```
select node, used from stv_node_storage_capacity order by node;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück.

```
 node | used
------+-------
    0 | 30597
    1 | 27089
```

Die folgende Abfrage gibt den belegten Speicherplatz und die Gesamtkapazität in 1-MB-Speicherblöcken zurück. Dieses Beispiel wurde auf einem ra3.16xlarge-Cluster mit zwei Knoten ausgeführt.

```
select node, used from stv_node_storage_capacity order by node;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück.

```
 node | used
------+-------
    0 | 30591
    1 | 27103
```

STV_PARTITIONS

Verwenden Sie die Tabelle STV_PARTITIONS, um die Festplattengeschwindigkeitsleistung und die Festplattennutzung für Amazon Redshift zu bestimmen.

STV_PARTITIONS enthält eine Zeile pro Knoten und logischer Festplattenpartition.

STV_PARTITIONS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
owner	integer	Festplattenknoten, der Eigentümer der Partition ist.
host (Host)	integer	Knoten, der physisch mit der Partition verbunden ist.
diskno	integer	Festplatte, die die Partition enthält.
part_begin	bigint	Offset der Partition. Rohdatenträger sind logisch so partition iert, dass Platz für Spiegelblöcke geöffnet wird.
part_end	bigint	Ende der Partition.

Spaltenname	Datentyp	Beschreibung
used	integer	Anzahl der 1 MB-Festplattenblöcke, die derzeit auf der Partition verwendet werden.
tossed	integer	Anzahl der Blöcke, die zur Löschung bereit sind, jedoch noch nicht entfernt wurden, da die Freigabe ihrer Festplatt enadressen nicht sicher ist. Würden die Adressen sofort freigegeben, könnte eine ausstehende Transaktion einen Schreibvorgang am selben Ort auf der Festplatte durchführ en. Daher werden diese Blöcke beim nächsten Commit freigegeben. Festplattenblöcke können beispielsweise dann als eingeschränkt markiert werden, wenn eine Tabellens palte entfernt wird, bei INSERT-Operationen oder bei festplatt enbasierten Abfrageoperationen.
Kapazität	integer	Gesamtkapazität der Partition in 1 MB-Festplattenblöcken.
reads	bigint	Anzahl der Lesevorgänge seit dem letzten Neustart des Clusters.
writes	bigint	Anzahl der Schreibvorgänge seit dem letzten Neustart des Clusters.
seek_forward	integer	Anzahl der Fälle, in denen eine Anfrage nicht für die folgende Adresse gilt (vor dem Hintergrund der vorherigen Anfragead resse).
seek_back	integer	Anzahl der Fälle, in denen eine Anfrage nicht für die vorherige Adresse gilt (vor dem Hintergrund der folgenden Adresse).
is_san	integer	Ob die Partition zu einem SAN gehört. Gültige Werte sind **0** (false) oder **1** (true).
failed	integer	Diese Spalte ist veraltet.
mbps	integer	Festplattengeschwindigkeit in Megabyte pro Sekunde.

Spaltenname	Datentyp	Beschreibung
mount	character (256)	Verzeichnispfad zu dem Gerät.

Beispielabfrage

Die folgende Abfrage gibt den verwendeten Festplattenspeicherplatz und die Festplattenkapazität (in 1 MB-Blöcken) aus und berechnet die Festplattennutzung als Prozentsatz des Roh-Festplattenspeicherplatzes. Der Roh-Festplattenspeicherplatz schließt den Platz ein, der von Amazon Redshift für die interne Verwendung reserviert ist, ist also größer als die nominelle Festplattenkapazität, bei der es sich um den Festplattenspeicherplatz handelt, der dem Benutzer zur Verfügung steht. Die Metrik Percentage of Disk Space Used (Prozentsatz des verwendeten Datenträgerplatzes) auf der Registerkarte Performance (Leistung) der Amazon-Redshift-Managementkonsole zeigt den Prozentsatz der nominellen Datenträgerkapazität an, die von Ihrem Cluster verwendet wird. Sie sollten die Metrik Percentage of Disk Space Used (Prozentsatz des verwendeten Datenträgerplatzes) überwachen, damit Ihre Nutzung innerhalb der nominellen Datenträgerkapazität Ihres Clusters bleibt.

> **Important**
>
> Wir empfehlen nachdrücklich, die nominelle Festplattenkapazität Ihres Clusters nicht zu überschreiten. Auch wenn dies unter bestimmten Umständen technisch möglich ist, beeinträchtigt die Überschreitung der nominellen Datenträgerkapazität die Fehlertoleranz Ihres Clusters und erhöht das Risiko von Datenverlusten.

Dieses Beispiel wurde auf einem zwei-Knoten-Cluster mit sechs logischen Festplattenpartitionen pro Knoten ausgeführt. Der Platz wird gleichmäßig über die Festplatte hinweg verwendet, wobei ca. 25 % jeder Festplatte genutzt werden.

```
select owner, host, diskno, used, capacity,
(used-tossed)/capacity::numeric *100 as pctused
from stv_partitions order by owner;

 owner | host | diskno |  used  | capacity | pctused
-------+------+--------+--------+----------+---------
   0   |  0   |    0   | 236480 |  949954  | 24.9
```

```
   0   |  0   |    1   | 236420 |  949954  | 24.9
   0   |  0   |    2   | 236440 |  949954  | 24.9
   0   |  1   |    2   | 235150 |  949954  | 24.8
   0   |  1   |    1   | 237100 |  949954  | 25.0
   0   |  1   |    0   | 237090 |  949954  | 25.0
   1   |  1   |    0   | 236310 |  949954  | 24.9
   1   |  1   |    1   | 236300 |  949954  | 24.9
   1   |  1   |    2   | 236320 |  949954  | 24.9
   1   |  0   |    2   | 237910 |  949954  | 25.0
   1   |  0   |    1   | 235640 |  949954  | 24.8
   1   |  0   |    0   | 235380 |  949954  | 24.8

(12 rows)
```

STV_QUERY_METRICS

Enthält Metrikinformationen, etwa die Anzahl der verarbeiteten Zeilen, die CPU-Nutzung, Input/Output- und Festplattennutzung für aktive Abfragen, die in benutzerdefinierten Abfragewarteschlangen (Service-Klassen) ausgeführt werden. Für die Anzeige von Metriken für abgeschlossene Abfragen vgl. die Systemtabelle STL_QUERY_METRICS.

Die Abfragemetriken werden in Intervallen von einer Sekunde erfasst. Daher können verschiedene Ausführungen einer Abfrage leicht abweichende Zeiten ergeben. Abfragesegmente, die in weniger als einer Sekunde ausgeführt werden, werden möglicherweise nicht aufgezeichnet.

STV_QUERY_METRICS verfolgt und aggregiert Metriken auf Abfrage-, Segment- und Schrittebene. Informationen zu Abfragesegmenten und Schritten finden Sie unter Workflow der Abfrageplanung und -ausführung. Viele Metriken (wie etwa `max_rows`, `cpu_time` u. dgl.) werden über Knoten-Slices hinweg summiert. Weitere Informationen über Knoten-Slices finden Sie unter Architektur des Data Warehouse-Systems.

Um die Ebene festzustellen, auf der die Zeile die Metriken meldet, prüfen Sie die Spalten `segment` und `step_type`:

- Wenn `segment` und `step_type` den Wert `-1` haben, meldet die Zeile die Metriken auf Abfrageebene.
- Wenn `segment` nicht den Wert `-1` und `step_type` den Wert `-1` hat, meldet die Zeile die Metriken auf Segmentebene.
- Wenn `segment` und `step_type` nicht den Wert `-1` haben, meldet die Zeile die Metriken auf Schrittebene.

STV_QUERY_METRICS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der die Abfrage ausgeführt hat, die den Eintrag generierte.
service_class	integer	ID für die WLM-Abfragewartschlange (Service-Klasse). Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Nur für benutzerdefinierte Abfragen werden Metriken gemeldet.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
starttime	timestamp	Uhrzeit in UTC, zu der die Ausführung der Abfrage begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: `2009-06-12 11:29:19.131358` .
slices	integer	Anzahl der Slices für den Cluster.
segment	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt. Wenn der Segmentwert -1 ist, werden Segment-Metrikwerte zur Abfrageebene aufgerollt.
step_type	integer	Typ des Schritts, der ausgeführt wurde. Eine detaillierte Beschreibung der Schritttypen finden Sie unter Schritttypen.

Spaltenname	Datentyp	Beschreibung
rows	bigint	Anzahl der von einem Schritt verarbeiteten Zeilen.
max_rows	bigint	Höchstzahl der für einen Schritt ausgegebenen Zeilen, über alle Slices hinweg aggregiert.
cpu_time	bigint	Die verwendete CPU-Zeit, in Mikrosekunden. Auf Segmenteb ene die gesamte von dem Segment verwendete CPU-Zeit, über alle Slices. Auf Abfrageebene die gesamte CPU-Zeit für die Abfrage, über alle Slices und Segmente.
max_cpu_time	bigint	Die maximale verwendete CPU-Zeit, in Mikrosekunden. Auf Segmentebene die maximale von dem Segment verwendet e CPU-Zeit, über alle Slices. Auf Abfrageebene die maximale von einem Abfragesegment verwendete CPU-Zeit.
blocks_read	bigint	Anzahl der von der Abfrage oder dem Segment gelesenen 1 MB-Blöcke.
max_block s_read	bigint	Die Höchstzahl der von dem Segment gelesenen 1 MB-Blöcke, aggregiert über alle Slices. Auf Segmentebene die Höchstzahl der für das Segment gelesenen 1 MB-Blöcke, über alle Slices. Auf Abfrageebene die Höchstzahl der von einem Abfragesegment gelesenen 1 MB-Blöcke.
run_time	bigint	Die gesamte Laufzeit, summiert über alle Slices. Die Laufzeit enthält nicht die Wartezeit. Auf Segmentebene die Laufzeit für das Segment, summiert über alle Slices. Auf Abfrageebene die Laufzeit für die Abfrage, summiert über alle Slices und Segmente. Da dieser Wert eine Summe ist, ist die Laufzeit nicht mit der Abfrageau sführungszeit verbunden.
max_run_time	bigint	Die maximale verstrichene Zeit für ein Segment, in Mikroseku nden. Auf Segmentebene die maximale Laufzeit für das Segment, über alle Slices. Auf Abfrageebene die maximale Laufzeit für ein Abfragesegment.

Spaltenname	Datentyp	Beschreibung
max_block s_to_disk	bigint	Der maximale Festplattenspeicherplatz, der zum Schreiben von Zwischenergebnissen verwendet wird, in 1 MB-Blöcke n. Auf Segmentebene der maximale von dem Segment verwendete Festplattenspeicherplatz, über alle Slices. Auf Abfrageebene der maximale von einem Abfragesegment verwendete Festplattenspeicherplatz.
blocks_to_disk	bigint	Der Festplattenspeicherplatz, der von einer Abfrage oder einem Segment zum Schreiben von Zwischenergebnissen verwendet wird, in 1 MB-Blöcken.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
max_query _scan_size	bigint	Die maximale von einer Abfrage gescannte Datenmenge, in MB. Auf Segmentebene die maximale von dem Segment gescannte Datenmenge, über alle Slices. Auf Abfrageeb ene die maximale von einem Abfragesegment gescannte Datenmenge.
query_scan_size	bigint	Die von einer Abfrage gescannte Datenmenge, in MB.
query_priority	integer	Die Priorität der Abfrage. Mögliche Werte sind -1, 0, 1, 2, 3 und 4, wobei -1 bedeutet, dass die Abfragepriorität nicht unterstützt wird.
query_que ue_time	bigint	Die Zeitspanne in Mikrosekunden, über die hinweg sich die Abfrage in der Warteschlange befand.

Schritttypen

Die folgende Tabelle listet die für Datenbankbenutzer relevanten Schritttypen auf. Die Tabelle führt die nur zur internen Verwendung gedachten Schritttypen nicht auf. Wenn der Schritttyp -1 ist, wird die Metrik nicht auf Schrittebene gemeldet.

Step type (Schritttyp)	Beschreibung
1	Tabelle scannen
2	Zeilen einfügen
3	Zeilen aggregieren
6	Sortierungsschritt
7	Zusammenführungsschritt
8	Verteilungsschritt
9	Broadcast-Verteilungsschritt
10	Hash join
11	Merge Join
12	Speicherschritt
14	Hash
15	Nested loop join
16	Felder und Ausdrücke projizieren
17	Begrenzen der Anzahl der zurückgegebenen Zeilen
18	Eindeutig
20	Zeilen löschen
26	Begrenzen der Anzahl der zurückgegebenen sortierten Zeilen
29	Berechnen einer Fensterfunktion
32	UDF
33	Eindeutig

Step type (Schritttyp)	Beschreibung
37	Gibt Zeilen aus dem Führungsknoten an den Client zurück
38	Gibt Zeilen von den Datenverarbeitungsknoten zum Führungsknoten zurück
40	Spektrum-Scan.

Beispielabfrage

Um aktive Abfragen mit einer hohen CPU-Zeit (über 1.000 Sekunden) zu finden, führen Sie die folgende Abfrage aus.

```
select query, cpu_time / 1000000 as cpu_seconds
from stv_query_metrics where segment = -1 and cpu_time > 1000000000
order by cpu_time;

query | cpu_seconds
------+------------
25775 |        9540
```

Um aktive Abfragen mit einem Nested Loop-Join zu finden, die mehr als eine Million Zeilen ausgegeben haben, führen Sie die folgende Abfrage aus.

```
select query, rows
from stv_query_metrics
where step_type = 15 and rows > 1000000
order by rows;

query | rows
------+-----------
25775 | 1580225854
```

Um aktive Abfragen zu finden, die länger als 60 Sekunden liefen und weniger als 10 Sekunden CPU-Zeit genutzt haben, führen Sie die folgende Abfrage aus.

```
select query, run_time/1000000 as run_time_seconds
from stv_query_metrics
```

```
where segment = -1 and run_time > 60000000 and cpu_time < 10000000;

query | run_time_seconds
------+-----------------
25775 |              114
```

STV_RECENTS

Verwenden Sie die Tabelle STV_RECENTS, um Informationen zu den derzeit aktiven und kürzlich ausgeführten Abfragen gegen eine Datenbank zu erhalten.

STV_RECENTS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Fehlerbehebung mit STV_RECENTS

STV_RECENTS ist besonders hilfreich, um festzustellen, ob eine Abfrage oder Sammlung von Abfragen gerade ausgeführt wird oder abgeschlossen ist. Dabei wird auch die Dauer angezeigt, wie lange eine Abfrage ausgeführt wurde. Dies ist hilfreich, um ein Gefühl dafür zu bekommen, welche Abfragen lange andauern.

Sie können STV_RECENTS mit anderen Systemansichten verbinden, z. B. STV_INFLIGHT, um zusätzliche Metadaten über laufende Abfragen zu sammeln. (Im Abschnitt mit den Beispielabfragen finden Sie ein Beispiel, das die Vorgehensweise zeigt.) Sie können auch zurückgegebene Datensätze aus dieser Ansicht zusammen mit den Überwachungsfunktionen in der Amazon-Redshift-Konsole für die Fehlerbehebung in Echtzeit verwenden.

Zu den Systemansichten, die STV_RECENTS ergänzen, gehören STL_QUERYTEXT, die den Abfragetext für SQL-Befehle abruft, und SVV_QUERY_INFLIGHT, die STV_INFLIGHT mit STL_QUERYTEXT verbindet.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
status	character(20)	Abfragestatus. Gültige Werte sind **`Running`**, **`Done`**.
starttime	timestamp	Der Zeitpunkt des Beginns der Abfrage.
duration	integer	Anzahl der Mikrosekunden seit dem Beginn der Sitzung.
user_name	character(50)	Name des Benutzers, der den Prozess ausgeführt hat.
db_name	character(50)	Name der Datenbank.
query	character(600)	Abfragetext, bis zu 600 Zeichen. Darüber hinaus gehende Zeichen werden abgeschnitten.
pid	integer	Prozess-ID für die zu der Abfrage gehörende Sitzung; diese ist immer -1 für abgeschlossene Abfragen.

Beispielabfragen

Um festzustellen, welche Abfragen derzeit für die Datenbank ausgeführt werden, geben Sie die folgende Abfrage ein:

```
select user_name, db_name, pid, query
from stv_recents
where status = 'Running';
```

Die Beispielausgabe unten zeigt eine einzelne Abfrage, die auf der TICKIT-Datenbank ausgeführt wird:

```
user_name | db_name |   pid   | query
----------+---------+---------+-------------
dwuser    | tickit  |  19996  |select venuename, venueseats from
venue where venueseats > 50000 order by venueseats desc;
```

Das folgende Beispiel gibt eine Liste von Abfragen (falls vorhanden) zurück, die derzeit ausgeführt werden oder in einer Warteschlange auf ihre Ausführung warten:

```
select * from stv_recents where status<>'Done';

status |    starttime        | duration |user_name|db_name| query     | pid
-------+---------------------+----------+---------+-------+-----------+------
Running| 2010-04-21 16:11... | 281566454| dwuser  |tickit | select ...| 23347
```

Diese Abfrage gibt nur Ergebnisse zurück, wenn Sie eine Reihe gleichzeitiger Abfragen ausführen und sich einige davon in einer Warteschlange befinden.

Das folgende Beispiel ist eine Erweiterung des vorherigen Beispiels. In diesem Fall werden tatsächlich ausgeführte (nicht wartende) Abfragen aus dem Ergebnis ausgeschlossen:

```
select * from stv_recents where status<>'Done'
and pid not in (select pid from stv_inflight);
...
```

Weitere Tipps zur Fehlerbehebung bei der Abfrageleistung finden Sie unter Fehlerbehebung bei Abfragen.

STV_SESSIONS

Verwenden Sie die Tabelle STV_SESSIONS zur Anzeige von Informationen zu den aktiven Benutzersitzungen für Amazon Redshift.

Verwenden Sie zur Anzeige des Sitzungsverlaufs die Tabelle STL_SESSIONS anstelle von STV_SESSIONS.

STV_SESSIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_SESSION_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
starttime	timestamp	Zeitpunkt des Beginns der Sitzung.
Verarbeitung	integer	Prozess-ID der Sitzung.
user_name	character(50)	Der mit der Sitzung verbundene Benutzer.
db_name	character(50)	Name der mit der Sitzung verbundenen Datenbank.
timeout_sec	int	Die maximale Zeit in Sekunden, die eine Sitzung ohne Timeout inaktiv oder untätig bleibt. 0 bedeutet, dass kein Timeout eingestellt ist.

Beispielabfragen

Um kurz zu prüfen, ob andere Benutzer derzeit bei Amazon Redshift angemeldet sind, geben Sie die folgende Abfrage ein:

```
select count(*)
from stv_sessions;
```

Wenn das Ergebnis größer als Eins ist, ist derzeit mindestens ein anderer Benutzer bei der Datenbank angemeldet.

Um alle aktiven Sitzungen für Amazon Redshift anzuzeigen, geben Sie die folgende Abfrage ein:

```
select *
from stv_sessions;
```

Das folgende Ergebnis zeigt vier aktive Sitzungen, die derzeit auf Amazon Redshift: ausgeführt werden:

```
       starttime          | process |user_name                    | db_name
      | timeout_sec
-------------------------+---------+-----------------------------
+---------------------------+------------
  2018-08-06 08:44:07.50 |   13779 | IAMA:aws_admin:admin_grp    | dev
      | 0
  2008-08-06 08:54:20.50 |   19829 | dwuser                      | dev
      | 120
  2008-08-06 08:56:34.50 |   20279 | dwuser                      | dev
      | 120
  2008-08-06 08:55:00.50 |   19996 | dwuser                      | tickit
      | 0
(3 rows)
```

Der Benutzername mit dem Präfix IAMA gibt an, dass sich der Benutzer mithilfe von verbundbasiertem Single Sign-On (SSO, Einmalanmeldung) angemeldet hat. Weitere Informationen finden Sie unter Verwenden der IAM-Authentifizierung zur Erstellung von Benutzeranmeldeinformationen für die Datenbank.

STV_SLICES

Verwenden Sie die Tabelle STV_SLICES zur Anzeige der aktuellen Zuweisung eines Slices zu einem Knoten.

Die Informationen in STV_SLICES werden hauptsächlich für Untersuchungszwecke verwendet.

STV_SLICES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
Knoten	integer	Cluster-Knoten, auf dem sich der Slice befindet.
slice	integer	Knoten-Slice.
localslice	integer	Diese Information ist nur für die interne Verwendung gedacht.

Spaltenname	Datentyp	Beschreibung
type	character (1)	Diese Information ist nur für die interne Verwendung gedacht.

Beispielabfrage

Um anzuzeigen, welche Cluster-Knoten welche Slices verwalten, geben Sie die folgende Abfrage ein:

```
select node, slice from stv_slices;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
 node | slice
------+-------
    0 |     2
    0 |     3
    0 |     1
    0 |     0
(4 rows)
```

STV_STARTUP_RECOVERY_STATE

Zeichnet den Status von Tabellen auf, die während Cluster-Neustartoperationen vorübergehend gesperrt sind. Amazon Redshift sperrt Tabellen vorübergehend, während sie bearbeitet werden, um angehaltene Transaktionen nach einem Cluster-Neustart aufzulösen.

STV_STARTUP_RECOVERY_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
db_id	integer	Datenbank-ID.

Spaltenname	Datentyp	Beschreibung
table_id	integer	Tabellen-ID.
table_name	character(137)	Tabellenname.

Beispielabfragen

Um zu überwachen, welche Tabellen vorübergehend gesperrt sind, führen Sie nach einem Cluster-Neustart die folgende Abfrage aus.

```
select * from STV_STARTUP_RECOVERY_STATE;

  db_id | tbl_id | table_name
--------+--------+------------
 100044 | 100058 | lineorder
 100044 | 100068 | part
 100044 | 100072 | customer
 100044 | 100192 | supplier
(4 rows)
```

STV_TBL_PERM

Die Tabelle STV_TBL_PERM enthält Informationen zu den permanenten Tabellen in Amazon Redshift, einschließlich der temporären Tabellen, die von einem Benutzer für die aktuelle Sitzung erstellt wurden. STV_TBL_PERM enthält Informationen für alle Tabellen in allen Datenbanken.

Diese Tabelle unterscheidet sich von STV_TBL_TRANS, die Informationen über temporäre Datenbanktabellen enthält, die das System bei der Verarbeitung von Abfragen erstellt.

STV_TBL_PERM ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
slice	integer	Knoten-Slice, der der Tabelle zugewiesen ist.
id	integer	Tabellen-ID.
Name	character (72)	Tabellenname.
rows	bigint	Anzahl der Datenzeilen in dem Slice.
sorted_rows	bigint	Anzahl der Zeilen in dem Slice, die bereits auf der Festplatte sortiert sind. Wenn diese Zahl nicht der ROWS-Zahl entspricht, bereinigen Sie die Tabelle, um die Zeilen erneut anzuordnen.
temp	integer	Ob es sich bei der Tabelle um eine temporäre Tabelle handelt. 0 = false; 1 = true.
db_id	integer	ID der Datenbank, in der die Tabelle erstellt wurde.
insert_pristine	integer	Zur internen Verwendung.
delete_pristine	integer	Zur internen Verwendung.
backup	integer	Wert, der anzeigt, ob die Tabelle in Cluster-Snapshots einbezogen ist. 0 = no; 1 = yes. Für weitere Informationen vgl. den Parameter BACKUP für den Befehl CREATE TABLE.
dist_style	Ganzzahl	Verteilungsstil der Tabelle, zu der der Slice gehört. Informationen zu den Werten finden Sie unter Anzeigen von Verteilungsstilen. Informationen zu Verteilungsstilen finden Sie unter Verteilungsstile.
block_count	Ganzzahl	Anzahl der vom Slice verwendeten Blöcke. Der Wert ist -1, wenn die Anzahl der Blöcke nicht berechnet werden kann.

Beispielabfragen

Die folgende Abfrage gibt eine Liste distinkter Tabellen-IDs und -namen aus:

```
select distinct id, name
from stv_tbl_perm order by name;

   id   |          name
--------+------------------------
 100571 | category
 100575 | date
 100580 | event
 100596 | listing
 100003 | padb_config_harvest
 100612 | sales
...
```

Andere Systemtabellen verwenden Tabellen-IDs, es ist daher sehr nützlich zu wissen, welche Tabellen-ID zu einer bestimmten Tabelle gehört. In diesem Beispiel wird SELECT DISTINCT verwendet, um die Duplikate zu entfernen (die Tabellen sind über mehrere Slices verteilt).

Um die Anzahl der Blöcke zu bestimmen, die jede Spalte in der VENUE-Tabelle enthält, geben Sie die folgende Abfrage ein:

```
select col, count(*)
from stv_blocklist, stv_tbl_perm
where stv_blocklist.tbl = stv_tbl_perm.id
and stv_blocklist.slice = stv_tbl_perm.slice
and stv_tbl_perm.name = 'venue'
group by col
order by col;

 col | count
-----+-------
   0 |     8
   1 |     8
   2 |     8
   3 |     8
   4 |     8
   5 |     8
   6 |     8
   7 |     8
```

```
(8 rows)
```

Nutzungshinweise

Die Spalte ROWS enthält die Anzahl der gelöschten Zeilen, die nicht bereinigt wurden (oder die bereinigt wurden, jedoch mit der Option SORT ONLY). Daher kann es sein, dass der Wert SUM der Spalte ROWS in der Tabelle STV_TBL_PERM nicht dem COUNT(*)-Ergebnis entspricht, wenn Sie eine bestimmte Tabelle direkt abfragen. Zum Beispiel: Wenn zwei Zeilen von VENUE gelöscht werden, ist das COUNT(*)-Ergebnis 200, das SUM(ROWS)-Ergebnis ist aber nach wie vor 202:

```
delete from venue
where venueid in (1,2);

select count(*) from venue;
count
-------
200
(1 row)

select trim(name) tablename, sum(rows)
from stv_tbl_perm where name='venue' group by name;

tablename | sum
-----------+-----
venue     | 202
(1 row)
```

Um die Daten in STV_TBL_PERM zu synchronisieren, führen Sie eine vollständige Bereinigung in der VENUE-Tabelle durch.

```
vacuum venue;

select trim(name) tablename, sum(rows)
from stv_tbl_perm
where name='venue'
group by name;

tablename | sum
-----------+-----
venue     | 200
(1 row)
```

STV_TBL_TRANS

Verwenden Sie die Tabelle STV_TBL_TRANS, um Informationen zu den temporären Datenbanktabellen zu erhalten, die sich derzeit im Speicher befinden.

Dabei handelt es sich typischerweise um temporäre Zeilensätze, die als Zwischenergebnisse verwendet werden, während eine Abfrage ausgeführt wird. STV_TBL_TRANS unterscheidet sich von STV_TBL_PERM darin, dass STV_TBL_PERM Informationen zu permanenten Datenbanktabellen enthält.

STV_TBL_TRANS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
slice	integer	Knoten-Slice, der der Tabelle zugewiesen ist.
id	integer	Tabellen-ID.
rows	bigint	Anzahl der Datenzeilen in der Tabelle.
size	bigint	Anzahl der der Tabelle zugeordneten Bytes.
query_id	bigint	Abfrage-ID.
ref_cnt	integer	Anzahl der Referenzen.
from_suspended	integer	Ob diese Tabelle während einer jetzt angehaltenen Abfrage erstellt wurde.
prep_swap	integer	Ob diese temporäre Tabelle bei Bedarf zur Festplatte wechseln kann. (Der Wechsel geschieht nur in Situationen mit geringem Speicher.)

Beispielabfragen

Um Informationen zu temporären Tabellen für eine Abfrage mit der Abfrage-ID 90 anzuzeigen, geben Sie den folgenden Befehl ein:

```
select slice, id, rows, size, query_id, ref_cnt
from stv_tbl_trans
where query_id = 90;
```

Diese Abfrage gibt die Informationen zur temporären Tabelle für Abfrage 90 aus, wie die folgende Beispielausgabe zeigt:

```
slice | id | rows | size | query_ | ref_ | from_     | prep_
      |    |      |      | id     | cnt  | suspended | swap
------+----+------+------+--------+------+-----------+-------
 1013 | 95 |    0 |    0 |     90 |    4 |         0 |   0
    7 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   10 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   17 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   14 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    3 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 99 |    0 |    0 |     90 |    4 |         0 |   0
    9 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    5 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   19 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    2 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 98 |    0 |    0 |     90 |    4 |         0 |   0
   13 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    1 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    6 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   11 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   15 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   18 | 96 |    0 |    0 |     90 |    4 |         0 |   0
```

In diesem Beispiel sehen Sie, dass die Abfragedaten die Tabellen 95, 96 und 98 betreffen. Da dieser Tabelle Null Bytes zugeordnet sind, kann diese Abfrage im Speicher ausgeführt werden.

STV_WLM_CLASSIFICATION_CONFIG

Enthält die aktuellen Klassifizierungsregeln für WLM.

STV_WLM_CLASSIFICATION_CONFIG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
id	integer	Service-Klassen-ID. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
Bedingung	character(128)	Abfragebedingungen.
action_seq	integer	Reserviert für die Systemverwendung.
Aktion	character(64)	Reserviert für die Systemverwendung.
action_service_class	integer	Die Service-Klasse, in der die Aktion stattfindet.

Beispielabfrage

```
select * from STV_WLM_CLASSIFICATION_CONFIG;

id | condition                                    | action_seq | action |
 action_service_class
---+----------------------------------------------+------------+--------
+---------------------
 1 | (system user) and (query group: health)     |          0 | assign |
     1
 2 | (system user) and (query group: metrics)    |          0 | assign |
     2
 3 | (system user) and (query group: cmstats)    |          0 | assign |
     3
 4 | (system user)                               |          0 | assign |
     4
 5 | (super user) and (query group: superuser)   |          0 | assign |
     5
 6 | (query group: querygroup1)                  |          0 | assign |
     6
 7 | (user group: usergroup1)                    |          0 | assign |
     6
 8 | (user group: usergroup2)                    |          0 | assign |
     7
```

```
 9 | (query group: querygroup3)                  |          0 | assign |
     8
10 | (query group: querygroup4)                  |          0 | assign |
     9
11 | (user group: usergroup4)                    |          0 | assign |
     9
12 | (query group: querygroup*)                  |          0 | assign |
    10
13 | (user group: usergroup*)                    |          0 | assign |
    10
14 | (querytype: any)                            |          0 | assign |
    11
(4 rows)
```

STV_WLM_QMR_CONFIG

Zeichnet die Konfiguration für WLM-Abfrageüberwachungsregeln (QMR) auf. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

STV_WLM_QMR_CONFIG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
service_class	integer	ID für die WLM-Abfragewartschlange (Service-Klasse). Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Regeln können nur für benutzerdefinierte Warteschlangen definiert werden. Eine Liste von Servicekl assen-IDs finden Sie unter WLM-Serviceklassen-IDs.
rule_name	character(256)	Name der Abfrageüberwachungsregel.
Aktion	character(256)	Regelaktion. Mögliche Werte sind `log`, `hop`, `abort` und `change_query_priority` .
metric_name	character(256)	Name der Metrik.
metric_operator	character(256)	Der Metrik-Operator. Die möglichen Werte sind >, =, <.

Spaltenname	Datentyp	Beschreibung
metric_value	double	Der Schwellenwert für die angegebene Metrik, der eine Aktion auslöst.
action_value	character(256)	Wenn für `action` `change_query_priority` angegeben ist, sind die Werte `highest`, `high`, `normal`, `low` und `lowest` möglich. Wenn für `action` `log`, `hop`, oder `abort` angegeben ist, ist der Wert leer.

Beispielabfrage

Um die QMR-Regeldefinitionen für alle Service-Klassen über 5 (mit benutzerdefinierten Warteschlangen) anzuzeigen, führen Sie die folgende Abfrage aus. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

```
Select *
from stv_wlm_qmr_config
where service_class > 5
order by service_class;
```

STV_WLM_QUERY_QUEUE_STATE

Zeichnet den aktuellen Zustand der Abfragewarteschlangen für die Service-Klassen auf.

STV_WLM_QUERY_QUEUE_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
service_class	integer	ID für die Service-Klasse. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
position	integer	Position der Abfrage in der Warteschlange. Die Abfrage mit dem kleinsten Wert für **`position`** wird als nächste ausgeführt.
Aufgabe	integer	ID zur Verfolgung einer Abfrage durch den Workload Manager. Kann mit mehreren Abfrage-IDs verbunden werden. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
query	integer	Abfrage-ID. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
slot_count	integer	Anzahl der WLM-Abfrageslots.
start_time	timestamp	Zeitpunkt, an dem die Abfrage in die Warteschlange gesetzt wurde.
queue_time	bigint	Anzahl der Mikrosekunden seitdem die Abfrage in die Warteschlange gesetzt wurde.

Beispielabfrage

Die folgende Abfrage zeigt die Abfragen in der Warteschlange für Service-Klassen über 4.

```
select * from stv_wlm_query_queue_state
where service_class > 4
order by service_class;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück.

```
 service_class | position | task | query | slot_count |        start_time          |
 queue_time
---------------+----------+------+-------+------------+----------------------------
+------------
             5 |        0 |  455 |   476 |          5 | 2010-10-06 13:18:24.065838 |
 20937257
             6 |        1 |  456 |   478 |          5 | 2010-10-06 13:18:26.652906 |
 18350191
(2 rows)
```

STV_WLM_QUERY_STATE

Zeigt den aktuellen Status der von WLM nachverfolgten Abfragen an.

STV_WLM_QUERY_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
xid	integer	Transaktions-ID der Abfrage oder Unterabfrage,
Aufgabe	integer	ID zur Verfolgung einer Abfrage durch den Workload Manager. Kann mit mehreren Abfrage-IDs verbunden werden. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
query	integer	Abfrage-ID. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
service_class	integer	ID für die Service-Klasse. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

Spaltenname	Datentyp	Beschreibung
slot_count	integer	Anzahl der WLM-Abfrageslots.
wlm_start_time	timestamp	Zeitpunkt, an dem die Abfrage in die Systemtabellenwarteschlange oder die Kurzabfragewarteschlange gesetzt wurde.

Spaltenname	Datentyp	Beschreibung
state	character(16)	Aktueller Status der Abfrage oder Unterabfrage. Folgende Werte sind möglich: • `Classified` – Die Abfrage wurde einer Serviceklasse zugewiesen. • `Completed` – Die Ausführung der Abfrage ist abgeschlossen. Die Abfrage wurde entweder erfolgreich ausgeführt oder abgebrochen. Um den endgültigen Zustand abzurufen, prüfen Sie die Ergebnisse von STL_QUERY. • `Dequeued` – Nur zur internen Verwendung. • `Evicted` – Die Abfrage wurde für den Neustart aus der Serviceklasse bereinigt. • `Evicting` – Die Abfrage wird derzeit für den Neustart aus der Serviceklasse bereinigt. • `Initialized` – Nur zur internen Verwendung. • `Invalid` – Nur zur internen Verwendung. • `Queued` – Die Abfrage wurde an die Abfragewarteschlange gesendet, da keine Slots zur Ausführung verfügbar waren. • `QueuedWaiting` – Die Abfrage wartet in der Abfragewarteschlange. • `Rejected` – Nur zur internen Verwendung. • `Returning` – Die Abfrage gibt Ergebnisse an den Client zurück. • `Running` – Die Abfrage wird ausgeführt. • `TaskAssigned` – Nur zur internen Verwendung.
queue_time	bigint	Anzahl der Mikrosekunden, die die Abfrage in der Warteschlange verbracht hat.

Spaltenname	Datentyp	Beschreibung
exec_time	bigint	Anzahl der Mikrosekunden, die die Abfrage bisher ausgeführt wird.
query_priority	char(20)	Die Priorität der Abfrage. Mögliche Werte sind `n/a`, `lowest`, `low`, `normal`, `high` und `highest`, wobei `n/a` bedeutet, dass die Abfragepriorität nicht unterstützt wird.

Beispielabfrage

Die folgende Abfrage zeigt alle derzeit ausgeführten Abfragen in Service-Klassen über 4 an. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

```
select xid, query, trim(state) as state, queue_time, exec_time
from stv_wlm_query_state
where service_class > 4;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
xid    | query | state   | queue_time | exec_time
-------+-------+---------+------------+-----------
100813 | 25942 | Running |          0 |    1369029
100074 | 25775 | Running |          0 | 2221589242
```

STV_WLM_QUERY_TASK_STATE

Enthält den derzeitigen Status von Service-Klassen-Abfrageaufgaben.

STV_WLM_QUERY_TASK_STATE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
service_class	integer	ID für die Service-Klasse. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
Aufgabe	integer	ID zur Verfolgung einer Abfrage durch den Workload Manager. Kann mit mehreren Abfrage-IDs verbunden werden. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
query	integer	Abfrage-ID. Wenn eine Abfrage neu gestartet wird, erhält sie eine neue Abfrage-ID, jedoch keine neue Task-ID.
slot_count	integer	Anzahl der WLM-Abfrageslots.
start_time	timestamp	Zeitpunkt, an dem die Ausführung der Abfrage begann.
exec_time	bigint	Anzahl der Mikrosekunden, für die die Abfrage ausgeführt wurde.

Beispielabfrage

Die folgende Abfrage zeigt den derzeitigen Status von Abfragen in Service-Klassen über 4 an. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

```
select * from stv_wlm_query_task_state
where service_class > 4;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
service_class | task | query |         start_time         | exec_time
--------------+------+-------+----------------------------+-----------
    5         |  466 |   491 | 2010-10-06 13:29:23.063787 | 357618748
(1 row)
```

STV_WLM_SERVICE_CLASS_CONFIG

Zeichnet die Service-Klassen-Konfigurationen für WLM auf.

STV_WLM_SERVICE_CLASS_CONFIG ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
service_class	integer	ID für die Service-Klasse. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
queueing_strategy	character(32)	Reserviert für die Systemverwendung.
num_query_tasks	integer	Aktuelle tatsächliche Gleichzeitigkeitsstufe der Service-Klasse. Wenn `num_query_tasks` und `target_num_query_tasks` unterschiedlich sind, findet ein dynamischer WLM-Übergang statt. Der Wert `-1` gibt an, dass das Auto WLM (automatische WLM konfiguriert ist.
target_num_query_tasks	integer	Die von der letzten WLM-Konfigurationsänderung eingerichtete Gleichzeitigkeitsstufe.
evictable	character(8)	Reserviert für die Systemverwendung.
eviction_threshold	bigint	Reserviert für die Systemverwendung.
query_working_mem	integer	Derzeitige tatsächliche Größe des Arbeitsspeichers, in MB pro Slot und Knoten, der der Service-Klasse zugewiesen ist. Wenn `query_working_mem` und `target_query_working_mem` unterschiedlich sind, findet ein dynamischer WLM-Übergang statt. Der Wert `-1` gibt an, dass das Auto WLM (Automatische WLM konfiguriert ist.
target_query_working_mem	integer	Die Größe des Arbeitsspeichers, in MB pro Slot und Knoten, die von der letzten WLM-Konfigurationsänderung eingerichtet wurde.

Spaltenname	Datentyp	Beschreibung
min_step_mem	integer	Reserviert für die Systemverwendung.
Name	character(64)	Der Name der Serviceklasse.
max_execution_time	bigint	Anzahl der Millisekunden, die die Abfrage ausgeführt werden kann, bevor sie beendet wird.
user_group_wild_card	Boolesch	Bei TRUE behandelt die WLM-Warteschlange ein Sternchen (*) als Platzhalterzeichen in Benutzergruppenzeichenfolgen in der WLM-Konfiguration.
query_group_wild_card	Boolesch	Bei TRUE behandelt die WLM-Warteschlange ein Sternchen (*) als Platzhalterzeichen in Abfragegruppenzeichenfolgen in der WLM-Konfiguration.
concurrency_scaling	character(20)	Beschreibt, ob die Nebenläufigkeitsskalierung on oder off ist.
query_priority	character(20)	Der Wert der Abfragepriorität.
user_role_wild_card	Boolesch	Bei TRUE behandelt die WLM-Warteschlange ein Sternchen (*) als Platzhalterzeichen in Benutzerzeichenfolgen der WLM-Konfiguration.

Beispielabfrage

Die erste benutzerdefinierte Service-Klasse ist Service-Klasse 6, die als Service-Klasse Nr. 1 bezeichnet wird. Die folgende Abfrage zeigt die aktuelle Konfiguration für Service-Klassen über 4 an. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

```
select rtrim(name) as name,
num_query_tasks as slots,
query_working_mem as mem,
max_execution_time as max_time,
user_group_wild_card as user_wildcard,
query_group_wild_card as query_wildcard
from stv_wlm_service_class_config
where service_class > 4;
```

```
name                          | slots | mem | max_time | user_wildcard | query_wildcard
------------------------------+-------+-----+----------+---------------+---------------
Service class for super user |     1 | 535 |        0 | false         | false
Queue 1                       |     5 | 125 |        0 | false         | false
Queue 2                       |     5 | 125 |        0 | false         | false
Queue 3                       |     5 | 125 |        0 | false         | false
Queue 4                       |     5 | 627 |        0 | false         | false
Queue 5                       |     5 | 125 |        0 | true          | true
Default queue                 |     5 | 125 |        0 | false         | false
```

Die folgende Abfrage zeigt den Status eines dynamischen WLM-Übergangs an. Während der Übergang läuft, werden `num_query_tasks` und `target_query_working_mem` aktualisiert, bis sie den Zielwerten entsprechen. Weitere Informationen finden Sie unter Dynamische und statische WLM-Konfigurationseigenschaften.

```
select rtrim(name) as name,
num_query_tasks as slots,
target_num_query_tasks as target_slots,
query_working_mem as memory,
target_query_working_mem as target_memory
from stv_wlm_service_class_config
where num_query_tasks > target_num_query_tasks
or query_working_mem > target_query_working_mem
and service_class > 5;

 name             | slots | target_slots | memory | target_mem
------------------+-------+--------------+--------+------------
 Queue 3          |     5 |           15 |    125 |       375
 Queue 5          |    10 |            5 |    250 |       125
 (2 rows)
```

STV_WLM_SERVICE_CLASS_STATE

Enthält den derzeitigen Status der Service-Klassen.

STV_WLM_SERVICE_CLASS_STATE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
service_class	integer	ID für die Service-Klasse. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
num_queued_queries	integer	Anzahl der derzeit in der Warteschlange befindlichen Abfragen.
num_executing_queries	integer	Anzahl der derzeit ausgeführten Abfragen.
num_serviced_queries	integer	Anzahl der Abfragen, die sich jemals in der Service-Klasse befunden haben.
num_executed_queries	integer	Anzahl der Abfragen, die seit dem Neustart von Amazon Redshift ausgeführt wurden.
num_evicted_queries	integer	Anzahl der Abfragen, die seit dem Neustart von Amazon Redshift bereinigt wurden. Einige Ursachen für eine bereinigte Abfrage sind ein WLM-Timeout, eine QMR-Hopping-Aktion und eine Abfrage, die auf einem Nebenläufigkeitsskalierungs-Cluster fehlschlägt.
num_concurrency_scaling_queries	integer	Anzahl der Abfragen, die seit dem Neustart von Amazon Redshift auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt wurden.

Beispielabfrage

Die folgende Abfrage zeigt den Status für Service-Klassen über 5 an. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.

```
select service_class, num_executing_queries,
num_executed_queries
from stv_wlm_service_class_state
where service_class > 5
order by service_class;
```

```
 service_class | num_executing_queries | num_executed_queries
---------------+-----------------------+----------------------
             6 |                     1 |                  222
             7 |                     0 |                  135
             8 |                     1 |                   39
(3 rows)
```

STV_XRESTORE_ALTER_QUEUE_STATE

Verwenden Sie STV_XRESTORE_ALTER_QUEUE_STATE, um den Migrationsfortschritt jeder Tabelle während einer klassischen Größenänderung zu überwachen. Dies ist insbesondere dann der Fall, wenn der Zielknotentyp RA3 ist. Weitere Informationen zur klassischen Größenänderung auf RA3-Knoten finden Sie unter Klassische Größenänderung.

STV_XRESTORE_ALTER_QUEUE_STATE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_RESTORE_STATE zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	Ganzzahl	Die ID des Benutzers, der die Größenänderung initiiert hat.
db_id	Ganzzahl	Die ID der Datenbank.
schema	char(128)	Der Name des Schemas.
table_name	char(128)	Der Name der Tabelle.
tbl	Ganzzahl	Die ID der Tabelle.
Status	char(64)	Der Status des Migrationsfortschritts der Tabelle. Mögliche Werte können sein wie folgt. • `Waiting`: Wartet auf den Beginn der Umverteilung

Spaltenname	Datentyp	Beschreibung
		• `Applying`: Wird gerade neu verteilt • `Finished`: Die Umverteilung ist abgeschlossen
task_type	Ganzzahl	Der Umverteilungstyp für die Tabelle. Mögliche Werte können sein wie folgt. • 1: SCHLÜSSEL • 2: SOGAR Weitere Informationen zu Verteilungsstilen finden Sie unterVerteilungsstile.

Beispielabfrage

Die folgende Abfrage zeigt die Anzahl der Tabellen in einer Datenbank, deren Größe noch nicht geändert werden muss, deren Größe gerade geändert wird und deren Größenänderung abgeschlossen ist.

```
select db_id, status, count(*)
from stv_xrestore_alter_queue_state
group by 1,2 order by 3 desc

db_id  |   status    | count
-------+-------------+------
694325 | Waiting     |   323
694325 | Finished    |    60
694325 | Applying    |     1
```

SVCS-Ansichten für Haupt- und Nebenläufigkeitsskalierungs-Cluster

SVCS-Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix STL vergleichbar, außer dass die STL-Tabellen nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

Themen

SVCS_ALERT_EVENT_LOG

Zeichnet eine Warnung auf, wenn der Abfrageoptimierer Bedingungen identifiziert, die auf Leistungsprobleme hinweisen können. Diese Ansicht wird aus der STL_ALERT_EVENT_LOG-Systemtabelle erzeugt, zeigt aber keine Slice-Ebene für Abfragen an, die auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden. Verwenden Sie die Tabelle SVCS_ALERT_EVENT_LOG, um Möglichkeiten zur Verbesserung der Abfrageleistung zu identifizieren.

Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Weitere Informationen finden Sie unter Verarbeitung von Abfragen.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix STL vergleichbar, außer dass die STL-Tabellen nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_ALERT_EVENT_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
pid	integer	Die mit der Anweisung und dem Slice verbundene Prozess-ID. Eine Abfrage kann mehrere PIDs haben, wenn sie für mehrere Slices ausgeführt wird.
xid	bigint	Mit der Anweisung verbundene Transaktions-ID.
event	character (1024)	Beschreibung des Warnungsereignisses.
solution	character (1024)	Empfohlene Lösung.
event_time	Zeitstempel	Zeitpunkt des Beginns der Abfrage, nach UTC. Die Gesamtzeit umfasst die Zeit in der Warteschlange und Zeit für die Ausführung mit einer Genauigkeit von 6 Nachkommastellen für Sekundenbruchteile. Zum Beispiel , .: **`2009-06-12 11:29:19.131358`** .

Nutzungshinweise

Sie können mit SVCS_ALERT_EVENT_LOG potenzielle Probleme in Ihren Abfragen identifizieren und dann den Vorgehensweisen in Optimieren der Abfrageleistung folgen, um Ihr Datenbankdesign

zu optimieren und Ihre Abfragen neu zu gestalten. SVCS_ALERT_EVENT_LOG zeichnet die folgenden Warnungen auf:

- Fehlende Statistik

  Es fehlen statistische Daten. Führen Sie nach Datenladevorgängen oder größeren Aktualisierungen ANALYZE aus, und verwenden Sie STATUPDATE mit COPY-Operationen. Weitere Informationen finden Sie unter Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift.

- Verschachtelte Schleife

  Eine verschachtelte Schleife ist normalerweise ein cartesisches Produkt. Prüfen Sie Ihre Abfrage, um sicherzustellen, dass alle beteiligten Tabellen effizient verbunden sind.

- Sehr selektiver Filter

  Das Verhältnis der zurückgegebenen zu den gescannten Zeilen ist kleiner als 0,05. Bei den gescannten Zeilen handelt es sich um den Wert von `rows_pre_user_filter` und bei den zurückgegebenen Zeilen um den Wert der Zeilen in der Systemtabelle STL_SCAN. Zeigt an, dass die Abfrage eine ungewöhnlich große Zahl von Zeilen scannt, um den Ergebnissatz zu bestimmen. Die Ursache können fehlende oder inkorrekte Sortierschlüssel sein. Weitere Informationen finden Sie unter Arbeiten mit Sortierschlüsseln.

- Übermäßig viele Geisterzeilen

  Ein Scan übersprang eine sehr große Zahl von Zeilen, die als gelöscht markiert sind, jedoch nicht bereinigt wurden, oder es wurden Zeilen eingefügt, aber nicht bestätigt (Commit). Weitere Informationen finden Sie unter Bereinigen von Tabellen.

- Large-Verteilung

  Es wurden mehr als 1.000.000 Zeilen für Hash-Join oder Aggregierung neu verteilt. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

- Large-Broadcast

  Für mehr als 1.000.000 Zeilen wurde ein Broadcast für Hash-Join durchgeführt. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

- Serielle Ausführung

In dem Abfrageplan wird ein Umverteilungsstil von DS_DIST_ALL_INNER angezeigt, der die serielle Ausführung erzwingt, da die gesamte innere Tabelle an einen einzelnen Knoten umverteilt wurde. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

Beispielabfragen

Die folgende Abfrage zeigt Warnungsereignisse für vier Abfragen an.

```
SELECT query, substring(event,0,25) as event,
substring(solution,0,25) as solution,
trim(event_time) as event_time from svcs_alert_event_log order by query;

 query |             event             |          solution            |     event_time

-------+-------------------------------+-------------------------------
+---------------------
  6567 | Missing query planner statist | Run the ANALYZE command       | 2014-01-03
 18:20:58
  7450 | Scanned a large number of del | Run the VACUUM command to rec| 2014-01-03
 21:19:31
  8406 | Nested Loop Join in the query | Review the join predicates to| 2014-01-04
 00:34:22
 29512 | Very selective query filter:r | Review the choice of sort key| 2014-01-06
 22:00:00

(4 rows)
```

SVCS_COMPILE

Zeichnet die Kompilierungszeit und den Ort für jedes Abfragesegment von Abfragen auf, sowohl für auf dem Nebenläufigkeitsskalierungs-Cluster als auch auf dem Haupt-Cluster ausgeführte Abfragen.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix SVL vergleichbar, nur dass die SVL-Ansichten nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_COMPILE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Weitere Informationen zu SCL_COMPILE finden Sie unter SVL_COMPILE.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Die mit der Anweisung verbundene Transaktions-ID.
pid	integer	Die mit der Anweisung verbundene Prozess-ID.
query	integer	Die Abfrage-ID. Sie können diese ID verwenden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
segment	integer	Das zu kompilierende Abfragesegment.
locus	integer	Der Ort, an dem das Segment ausgeführt wird: **1**, wenn auf einem Datenverarbeitungsknoten, und **2**, wenn auf dem Führungsknoten.
starttime	timestamp	Der Zeitpunkt, zu dem der Kompiliervorgang startete, in Universal Coordinated Time (UTC).
endtime	timestamp	Der Zeitpunkt des Endes des Kompiliervorgangs, in UTC.
compile	integer	Ein Wert, der **0** ist, wenn die Kompilierung wiederverwendet wurde, und **1**, wenn das Segment kompiliert wurde.

Beispielabfragen

In diesem Beispiel führten die Abfragen 35878 und 35879 die gleiche SQL-Anweisung aus. Die Kompilierungsspalte für Abfrage 35878 zeigt 1 für vier Abfragesegmente; dies zeigt an, dass die Segmente kompiliert wurden. Abfrage 35879 zeigt `0` in der Kompilierungsspalte für jedes Segment; dies zeigt an, dass die Segmente nicht erneut kompiliert werden müssen.

```
select userid, xid,  pid, query, segment, locus,
datediff(ms, starttime, endtime) as duration, compile
from svcs_compile
where query = 35878 or query = 35879
order by query, segment;

 userid |  xid   |  pid  | query | segment | locus | duration | compile
--------+--------+-------+-------+---------+-------+----------+---------
    100 | 112780 | 23028 | 35878 |       0 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       1 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       2 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       3 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       4 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       5 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       6 |     1 |     1380 |       1
    100 | 112780 | 23028 | 35878 |       7 |     1 |     1085 |       1
    100 | 112780 | 23028 | 35878 |       8 |     1 |     1197 |       1
    100 | 112780 | 23028 | 35878 |       9 |     2 |      905 |       1
    100 | 112782 | 23028 | 35879 |       0 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       1 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       2 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       3 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       4 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       5 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       6 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       7 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       8 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       9 |     2 |        0 |       0
(20 rows)
```

SVCS_CONCURRENCY_SCALING_USAGE

Zeichnet die Nutzungszeiträume für die Nebenläufigkeitsskalierung auf. Jeder Nutzungszeitraum ist ein laufender Zeitraum, in dem ein Nebenläufigkeitsskalierungs-Cluster aktiv Abfragen verarbeitet.

SVCS_CONCURRENCY_SCALING_USAGE Diese Tabelle ist für Superuser sichtbar. Der Datenbank-Superuser kann die Ansicht für alle Benutzer zugänglich machen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
start_time	Timestamp ohne Zeitzone	Wenn der Nutzungszeitraum startet.
end_time	Timestamp ohne Zeitzone	Wenn der Nutzungszeitraum endet.
queries	bigint	Anzahl der Abfragen, die in diesem Nutzungszeitraum ausgeführt wurden.
usage_in_seconds	numeric(27,0)	Gesamtsekunden in diesem Nutzungszeitraum.

Beispielabfragen

Geben Sie die folgende Abfrage ein, um sich die Nutzungsdauer in Sekunden für einen bestimmten Zeitraum anzeigen zu lassen:

```
select * from svcs_concurrency_scaling_usage order by start_time;

start_time | end_time | queries | usage_in_seconds
---------------------------+---------------------------+---------+-----------------
2019-02-14 18:43:53.01063 | 2019-02-14 19:16:49.781649 | 48 | 1977
```

SVCS_EXPLAIN

Zeigt den EXPLAIN-Plan für eine Abfrage an, die zur Ausführung übermittelt wurde.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix STL

vergleichbar, außer dass die STL-Tabellen nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_EXPLAIN ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
nodeid	integer	Plan-Knoten-ID, wo ein Knoten mit einem oder mehreren Schritten in der Ausführung der Abfrage verwunden ist.
parentid	integer	Plan-Knoten-ID für einen übergeordneten Knoten. Zu einem übergeordneten Knoten gehört eine bestimmte Anzahl untergeordneter Knoten. So ist beispielsweise ein Zusammenführungsknoten der übergeordnete Knoten der Scans in den verbundenen Tabellen.
plannode	character(400)	Der Knotentext aus der EXPLAIN-Ausgabe. Planknoten, die sich auf die Ausführung auf Datenverarbeitungsknoten beziehen, haben das Präfix **XN** in der EXPLAIN-Ausgabe.
info	character(400)	Qualifizierer- und Filterinformationen für den Plan-Knoten. Diese Spalte enthält zum Beispiel Join-Bedingungen und Einschränkungen für WHERE-Klauseln.

Beispielabfragen

Sehen Sie sich die folgende EXPLAIN-Ausgabe für eine aggregierte Join-Abfrage an:

```
explain select avg(datediff(day, listtime, saletime)) as avgwait
from sales, listing where sales.listid = listing.listid;
                                  QUERY PLAN

------------------------------------------------------------------------------
 XN Aggregate  (cost=6350.30..6350.31 rows=1 width=16)
  ->  XN Hash Join DS_DIST_NONE  (cost=47.08..6340.89 rows=3766 width=16)
        Hash Cond: ("outer".listid = "inner".listid)
        -> XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=12)
        -> XN Hash  (cost=37.66..37.66 rows=3766 width=12)
              -> XN Seq Scan on sales  (cost=0.00..37.66 rows=3766 width=12)
(6 rows)
```

Wenn Sie diese Abfrage ausführen und die Abfrage-ID 10 ist, können Sie anhand der Tabelle SVCS_EXPLAIN die gleichen Arten von Informationen sehen, die der Befehl EXPLAIN ausgibt:

```
select query,nodeid,parentid,substring(plannode from 1 for 30),
substring(info from 1 for 20) from svcs_explain
where query=10 order by 1,2;

query| nodeid |parentid|           substring            |    substring
-----+--------+--------+--------------------------------+------------------
10   |      1 |      0 |XN Aggregate  (cost=6717.61..6  |
10   |      2 |      1 |  -> XN Merge Join DS_DIST_NO| Merge Cond:("outer"
10   |      3 |      2 |       -> XN Seq Scan on lis |
10   |      4 |      2 |       -> XN Seq Scan on sal |
(4 rows)
```

Betrachten Sie folgende Abfrage:

```
select event.eventid, sum(pricepaid)
from event, sales
where event.eventid=sales.eventid
group by event.eventid order by 2 desc;

eventid |   sum
--------+----------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
```

```
...
```

Wenn die ID dieser Abfrage 15 ist, gibt die folgende Systemtabellenabfrage die ausgeführten Plan-Knoten aus. In diesem Fall ist die Reihenfolge der Knoten umgekehrt, um die tatsächliche Ausführungsreihenfolge zu zeigen:

```
select query,nodeid,parentid,substring(plannode from 1 for 56)
from svcs_explain where query=15 order by 1, 2 desc;

query|nodeid|parentid|                            substring
-----+------+--------+-----------------------------------------------------
15   |    8 |      7 |                                  -> XN Seq Scan on eve
15   |    7 |      5 |                            -> XN Hash(cost=87.98..87.9
15   |    6 |      5 |                            -> XN Seq Scan on sales(cos
15   |    5 |      4 |                       -> XN Hash Join DS_DIST_OUTER(cos
15   |    4 |      3 |                 -> XN HashAggregate(cost=862286577.07..
15   |    3 |      2 |           -> XN Sort(cost=1000862287175.47..10008622871
15   |    2 |      1 |  -> XN Network(cost=1000862287175.47..1000862287197.
15   |    1 |      0 |XN Merge(cost=1000862287175.47..1000862287197.46 rows=87
(8 rows)
```

Die folgende Abfrage ruft die Abfrage-IDs für alle Abfrage-Pläne ab, die eine Fensterfunktion beinhalten:

```
select query, trim(plannode) from svcs_explain
where plannode like '%Window%';

query|                                    btrim
-----+-------------------------------------------------------------------------
26   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
27   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
(2 rows)
```

SVCS_PLAN_INFO

Verwenden Sie die Tabelle SVCS_PLAN_INFO zur Betrachtung der EXPLAIN-Ausgabe für eine Abfrage als Satz von Zeilen. Diese ist eine alternative Betrachtungsweise von Abfrageplänen.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix STL vergleichbar, außer dass die STL-Tabellen nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_PLAN_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
nodeid	integer	Plan-Knoten-ID, wo ein Knoten mit einem oder mehreren Schritten in der Ausführung der Abfrage verwunden ist.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.
Schritt	integer	Zahl, mit der der Abfrageschritt identifiziert wird.
locus	integer	Ort, an dem der Schritt ausgeführt wird. 0, wenn auf einem Datenverarbeitungsknoten, und 1, wenn auf dem Führungsknoten.
plannode	integer	Enumerationswert des Planknotens. Vgl. die folgende Tabelle für Enum-Werte für Plannode. (Die Spalte PLANNODE in SVCS_EXPLAIN enthält den Planknotentext.)
startupcost	double precision	Die geschätzten relativen Kosten für die Rückgabe der ersten Zeile für diesen Schritt.

Spaltenname	Datentyp	Beschreibung
totalcost	double precision	Die geschätzten relativen Kosten für die Ausführung des Schrittes.
rows	bigint	Die geschätzte Anzahl der Zeilen, die durch den Schritt erstellt werden.
bytes	bigint	Die geschätzte Anzahl der Bytes, die durch den Schritt erstellt werden.

Beispielabfragen

Die folgenden Beispiele vergleichen die Abfragepläne für eine einfache SELECT-Abfrage mit dem Befehl EXPLAIN und Abfrage der Tabelle SVCS_PLAN_INFO.

```
explain select * from category;
QUERY PLAN
------------------------------------------------------------
XN Seq Scan on category (cost=0.00..0.11 rows=11 width=49)
(1 row)

select * from category;
catid | catgroup | catname | catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports | MLB | Major League Baseball
3 | Sports | NFL | National Football League
5 | Sports | MLS | Major League Soccer
...

select * from svcs_plan_info where query=256;

query | nodeid | segment | step | locus | plannode | startupcost | totalcost
| rows | bytes
-------+--------+---------+------+-------+----------+-------------+-----------+------
+-------
256 | 1 | 0 | 1 | 0 | 104 | 0 | 0.11 | 11 | 539
256 | 1 | 0 | 0 | 0 | 104 | 0 | 0.11 | 11 | 539
(2 rows)
```

In diesem Beispiel bezieht sich PLANNODE 104 auf den sequenziellen Scan der Tabelle CATEGORY.

```
select distinct eventname from event order by 1;

eventname
----------------------------------------------------------------------
.38 Special
3 Doors Down
70s Soul Jam
A Bronx Tale
...

explain select distinct eventname from event order by 1;

QUERY PLAN
------------------------------------------------------------------------------------------
XN Merge (cost=1000000000136.38..1000000000137.82 rows=576 width=17)
Merge Key: eventname
-> XN Network (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Send to leader
-> XN Sort (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Sort Key: eventname
-> XN Unique (cost=0.00..109.98 rows=576 width=17)
-> XN Seq Scan on event (cost=0.00..87.98 rows=8798
width=17)
(8 rows)

select * from svcs_plan_info where query=240 order by nodeid desc;

query | nodeid | segment | step | locus | plannode | startupcost |
totalcost | rows | bytes
-------+--------+---------+------+-------+----------+-----------------
+------------------+------+--------
240 | 5 | 0 | 0 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 5 | 0 | 1 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 4 | 0 | 2 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 0 | 3 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 0 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 1 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 3 | 1 | 2 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
```

```
240 | 3 | 2 | 0 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 2 | 2 | 1 | 0 | 123 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 1 | 3 | 0 | 0 | 122 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
(10 rows)
```

SVCS_QUERY_SUMMARY

Verwenden Sie die Ansicht SVCS_QUERY_SUMMARY, um allgemeine Informationen zur Ausführung einer Abfrage zu erhalten.

Beachten Sie, dass die Informationen in SVCS_QUERY_SUMMARY von allen Knoten aggregiert sind.

> **Note**
>
> Die Ansicht SVCS_QUERY_SUMMARY enthält nur Informationen zu von Amazon Redshift ausgeführten Abfragen, nicht zu anderen Utility- und DDL-Befehlen. Für eine vollständige Liste und Informationen zu allen von Amazon Redshift ausgeführten Anweisungen, einschließlich DDL- und Utility-Befehlen, können Sie die Ansicht SVL_STATEMENTTEXT abfragen.
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix SVL vergleichbar, nur dass die SVL-Ansichten nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_QUERY_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Informationen zu SVL_QUERY_SUMMARY finden Sie unter SVL_QUERY_SUMMARY.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
stm	integer	Stream: Ein Satz gleichzeitiger Segmente einer Abfrage. Eine Abfrage enthält einen oder mehrere Streams.
seg	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten , und jedes Segment besteht aus einem oder mehreren Schritten . Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
maxtime	bigint	Maximale Zeit für die Ausführung des Schritts (in Mikrosekunden).
avgtime	bigint	Durchschnittliche Zeit für die Ausführung des Schritts (in Mikrosekunden).
rows	bigint	Anzahl der an dem Abfrageschritt beteiligten Datenzeilen.
bytes	bigint	Anzahl der an dem Abfrageschritt beteiligten Bytes.
rate_row	double precision	Ausführungsrate der Abfrage pro Zeile.
rate_byte	double precision	Ausführungsrate der Abfrage pro Byte.
label	Text	Schritt-Etikett, bestehend aus dem Namen eines Abfrageschritts und, falls anwendbar, der Tabellen-ID und dem Tabellennamen (zum Beispiel scan tbl = 100448). Dreistellige Tabellen-IDs beziehen sich normalerweise auf Scans temporärer Tabellen.

Spaltenname	Datentyp	Beschreibung
		Wenn Sie `tbl=0` sehen, bezieht sich dies auf einen Scan eines konstanten Werts.
is_diskbased	character(1)	Ob dieser Schritt der Abfrage als datenträgerbasierte Operation auf einem Knoten im Cluster ausgeführt wurde: true (**t**) oder false (**f**). Nur bestimmte Schritte, wie etwa Hash-, Sortierungs- oder Aggregierungsschritte, können auf die Festplatte übertragen werden. Viele Arten von Schritten werden immer im Arbeitsspeicher ausgeführt.
workmem	bigint	Größe des Arbeitsspeichers (in Byte), der dem Abfrageschritt zugewiesen wurde.
is_rrscan	character(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein Scan mit Bereichse inschränkung durchgeführt wurde. Der Standardwert ist „false“ (**f**).
is_delayed_scan	character(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein verzögerter Scan durchgeführt wurde. Der Standardwert ist „false“ (**f**).
rows_pre_filter	bigint	Für Scans permanenter Tabellen die Gesamtzahl der Zeilen, die vor der Filterung der zur Löschung markierten Zeilen (Geisterzeilen) ausgegeben wurden.

Beispielabfragen

Anzeige von Verarbeitungsinformationen für einen Abfrageschritt

Die folgende Abfrage zeigt allgemeine Verarbeitungsinformationen für jeden Schritt von Abfrage 87:

```
select query, stm, seg, step, rows, bytes
from svcs_query_summary
where query = 87
order by query, seg, step;
```

Diese Abfrage ruft die Verarbeitungsinformationen zu Abfrage 87 aus, wie die folgende Beispielausgabe zeigt:

```
 query | stm | seg | step |  rows  |  bytes
-------+-----+-----+------+--------+---------
87     |   0 |   0 |    0 |     90 |    1890
87     |   0 |   0 |    2 |     90 |     360
87     |   0 |   1 |    0 |     90 |     360
87     |   0 |   1 |    2 |     90 |    1440
87     |   1 |   2 |    0 | 210494 | 4209880
87     |   1 |   2 |    3 |  89500 |       0
87     |   1 |   2 |    6 |      4 |      96
87     |   2 |   3 |    0 |      4 |      96
87     |   2 |   3 |    1 |      4 |      96
87     |   2 |   4 |    0 |      4 |      96
87     |   2 |   4 |    1 |      1 |      24
87     |   3 |   5 |    0 |      1 |      24
87     |   3 |   5 |    4 |      0 |       0
(13 rows)
```

Ermittlung, ob Abfrageschritte auf den Datenträger übertragen wurden

Die folgende Abfrage zeigt, ob einer oder mehrere der Schritte für die Abfrage mit Abfrage-ID 1025 (vgl. die Ansicht SVL_QLOG zum Abruf der Abfrage-ID für eine Abfrage) auf die Festplatte übertragen wurde, oder ob die Abfrage vollständig im Speicher ausgeführt wurde:

```
select query, step, rows, workmem, label, is_diskbased
from svcs_query_summary
where query = 1025
order by workmem desc;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
query| step|  rows  |  workmem   |  label        | is_diskbased
-----+-----+--------+------------+---------------+--------------
1025 |  0  |16000000|  141557760 |scan tbl=9     | f
1025 |  2  |16000000|  135266304 |hash tbl=142   | t
1025 |  0  |16000000|  128974848 |scan tbl=116536| f
1025 |  2  |16000000|  122683392 |dist           | f
(4 rows)
```

Durch das Scannen der Werte für IS_DISKBASED sehen Sie, welche Abfrageschritte auf die Festplatte übertragen wurden. Bei Abfrage 1025 wurde der Hash-Schritt auf der Festplatte ausgeführt. Schritte, die auf der Festplatte ausgeführt werden können, sind u. a. Hash-,

Aggregierungs- und Sortierungsschritte. Um ausschließlich datenträgerbasierte Abfrageschritte anzuzeigen, fügen Sie der SQL-Anweisung im Beispiel oben die Klausel **`and is_diskbased = 't'`** hinzu.

SVCS_S3LIST

Verwenden Sie die Ansicht SVCS_S3LIST, um Details zu Amazon-Redshift-Spectrum-Abfragen auf Segment-Ebene zu erhalten. Ein Segment kann einen Scan einer externen Tabelle durchführen. Diese Ansicht wird aus der SVL_S3LIST-Systemansicht abgeleitet, zeigt aber keine Slice-Ebene für Abfragen an, die auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix SVL vergleichbar, nur dass die SVL-Ansichten nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_S3LIST ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu SVL_S3LIST finden Sie unter SVL_S3LIST.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten.
Knoten	integer	Die Knotennummer.
eventtime	Zeitstempel	Der Zeitpunkt nach UTC, an dem das Ereignis aufgezeichnet wurde.

Spaltenname	Datentyp	Beschreibung
bucket	char(256)	Der Name des Amazon S3-Buckets.
prefix	char(256)	Das Präfix des Amazon-S3-Bucket-Speicherortes.
recursive	char(1)	Ob ein rekursiver Scan für Unterordner vorhanden ist.
retrieved_files	integer	Die Anzahl der aufgelisteten Dateien.
max_file_size	bigint	Die maximale Dateigröße unter aufgelisteten Dateien.
avg_file_size	double precision	Die durchschnittliche Dateigröße unter aufgelisteten Dateien.
generated_splits	integer	Die Anzahl der Dateiaufteilungen.
avg_split_length	double precision	Die durchschnittliche Länge von Dateiaufteilungen in Byte.
duration	bigint	Die Zeitdauer der Dateiauflistung in Mikrosekunden.

Beispielabfrage

Im folgenden Beispiel wird SVCS_S3LIST nach der letzten ausgeführten Abfrage abgefragt.

```
select *
from svcs_s3list
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3LOG

Verwenden Sie die Ansicht SVCS_S3LOG, um Fehlerbehebungsdetails zu Redshift-Spectrum-Abfragen auf Segment-Ebene zu erhalten. Ein Segment kann einen Scan einer externen Tabelle

durchführen. Diese Ansicht wird aus der SVL_S3LOG-Systemansicht abgeleitet, zeigt aber keine Slice-Ebene für Abfragen an, die auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix SVL vergleichbar, nur dass die SVL-Ansichten nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_S3LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu SVL_S3LOG finden Sie unter SVL_S3LOG.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
pid	integer	Die Prozess-ID.
query	integer	Die Abfrage-ID.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
Schritt	integer	Der Abfrage-Schritt, der ausgeführt wurde.
Knoten	integer	Die Knotennummer.
eventtime	Zeitstempel	Der Zeitpunkt nach UTC, an dem das Ereignis aufgezeichnet wurde.
Nachricht	char(512)	Die Meldung für den Protokolleintrag.

Beispielabfrage

Im folgenden Beispiel wird SVCS_S3LOG nach der letzten ausgeführten Abfrage abgefragt.

```
select *
from svcs_s3log
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3PARTITION_SUMMARY

Verwenden Sie die Ansicht SVCS_S3PARTITION_SUMMARY, um eine Zusammenfassung der Redshift-Spectrum-Abfragepartition zu erhalten, die auf Segment-Ebene ausgeführt wird. Ein Segment kann einen Scan einer externen Tabelle durchführen.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix SVL vergleichbar, nur dass die SVL-Ansichten nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_S3PARTITION_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu SVL_S3PARTITION finden Sie unter SVL_S3PARTITION.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID. Mit diesem Wert können Sie verschiedene andere Systemtabellen und Ansichten verbinden.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten.

Spaltenname	Datentyp	Beschreibung
assignment	char(1)	Der Typ von Partitionszuweisung über Knoten hinweg.
min_starttime	timestamp	Der Zeitpunkt nach UTC, an dem die Partitionsverarbeitung gestartet wurde.
max_endtime	timestamp	Der Zeitpunkt nach UTC, an dem die Partitionsverarbeitung abgeschlossen wurde.
min_duration	bigint	Die von einem Knoten für diese Abfrage verwendete minimale Partitionsverarbeitungszeit (in Mikrosekunden).
max_duration	bigint	Die von einem Knoten für diese Abfrage verwendete maximale Partitionsverarbeitungszeit (in Mikrosekunden).
avg_duration	bigint	Die von einem Knoten für diese Abfrage verwendete durchschnittliche Partitionsverarbeitungszeit (in Mikrosekunden).
total_partitions	integer	Die Gesamtzahl von Partitionen in einer externen Tabelle.
qualified_partitions	integer	Die Gesamtzahl der qualifizierten Partitionen.
min_assigned_partitions	integer	Die einem Knoten zugewiesene minimale Anzahl von Partitionen.
max_assigned_partitions	integer	Die einem Knoten zugewiesene maximale Anzahl von Partitionen.
avg_assigned_partitions	bigint	Die einem Knoten zugewiesene durchschnittliche Anzahl von Partitionen.

Beispielabfrage

Das folgende Beispiel ruft die Partitions-Scandetails für die letzte ausgeführte Abfrage ab.

```
select query, segment, assignment, min_starttime, max_endtime, min_duration,
 avg_duration
from svcs_s3partition_summary
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3QUERY_SUMMARY

Verwenden Sie die Ansicht SVCS_S3QUERY_SUMMARY zum Erhalt einer Zusammenfassung aller Redshift-Spectrum-Abfragen (S3-Abfragen), die auf dem System ausgeführt wurden. Ein Segment kann einen Scan einer externen Tabelle durchführen.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix SVL vergleichbar, nur dass die SVL-Ansichten nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_S3QUERY_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu SVL_S3QUERY finden Sie unter SVL_S3QUERY.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der den jeweiligen Eintrag generiert hat.
query	integer	Die Abfrage-ID. Mit diesem Wert können Sie verschiedene andere Systemtabellen und Ansichten verbinden.

Spaltenname	Datentyp	Beschreibung
xid	bigint	Die Transaktions-ID.
pid	integer	Die Prozess-ID.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
Schritt	integer	Der Abfrage-Schritt, der ausgeführt wurde.
starttime	timestamp	Der Zeitpunkt nach UTC, an dem die Ausführung der Redshift-Spectrum-Abfrage in diesem Segment gestartet wurde. Ein Segment kann nur einen Scan einer externen Tabelle enthalten.
endtime	timestamp	Der Zeitpunkt nach UTC, an dem die Redshift-Spectrum-Abfrage in diesem Segment abgeschlossen wurde. Ein Segment kann nur einen Scan einer externen Tabelle enthalten.
elapsed	integer	Die Zeitdauer der Ausführung der Redshift-Spectrum-Abfrage in diesem Segment (in Mikrosekunden).
aborted	integer	Wenn eine Abfrage vom System oder einem Benutzer beendet wurde, enthält diese Spalte den Wert **1**. Wenn die Abfrage abgeschlossen wurde, enthält diese Spalte **0**.
external_table_name	char(136)	Das interne Format des Namens der externen Tabelle für den Scan einer externen Tabelle.
file_format	character(16)	Das Dateiformat der Daten einer externen Tabelle.
is_partitioned	char(1)	Wenn „true“ (**t**) zeigt dieser Spaltenwert an, dass die externe Tabelle partitioniert ist.

Spaltenname	Datentyp	Beschreibung
is_rrscan	char(1)	Wenn „true“ (**t**) zeigt dieser Spaltenwert an, dass ein Scan mit Bereichseinschränkung durchgeführt wurde.
is_nested	varchar(1)	Bei „true“ (**t**) gibt dieser Spaltenwert an, dass auf den verschachtelten Spaltendatentyp zugegriffen wird.
s3_scanned_rows	bigint	Die Anzahl der von Amazon S3 gescannten und an die Redshift-Spectrum-Ebene gesendeten Zeilen.
s3_scanned_bytes	bigint	Die Anzahl der von Amazon S3 gescannten und an die Redshift-Spectrum-Ebene gesendeten Bytes, basierend auf komprimierten Daten.
s3query_returned_rows	bigint	Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Zeilen.
s3query_returned_bytes	bigint	Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Bytes. Eine große Menge an Amazon Redshift zurückgegebener Daten kann sich auf die Systemleistung auswirken.
files	integer	Die Anzahl der Dateien, die für diese Redshift Spectrum-Abfrage verarbeitet wurden. Eine kleine Anzahl von Dateien schränkt die Vorteile der parallelen Verarbeitung ein.
files_max	integer	Die maximale Anzahl der auf einem Slice verarbeiteten Dateien.
files_avg	integer	Die durchschnittliche Anzahl der auf einem Slice verarbeiteten Dateien.

Spaltenname	Datentyp	Beschreibung
splits	bigint	Die Anzahl der Aufteilungen, die für dieses Segment verarbeitet wurden. Die Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden. Bei großen aufteilbaren Datendateien versucht Redshift Spectrum beispielsweise bei Dateigrößen über 512 MB, die Dateien zur parallelen Verarbeitung auf mehrere S3-Anforderungen aufzuteilen.
splits_max	integer	Die maximale Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden
splits_avg	bigint	Die durchschnittliche Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden
total_split_size	bigint	Die Gesamtgröße aller verarbeiteten Aufteilungen
max_split_size	bigint	Die maximale verarbeitete Aufteilungsgröße (in Bytes)
avg_split_size	bigint	Die durchschnittliche verarbeitete Aufteilungsgröße (in Bytes)
total_retries	bigint	Die Gesamtzahl der erneuten Versuche für die Amazon-Redshift-Abfrage in diesem Segment.
max_retries	integer	Die maximale Zahl erneuter Versuche für eine einzelne verarbeitete Datei.
max_request_duration	bigint	Die maximale Dauer einer einzelnen Dateianfrage (in Mikrosekunden). Länger dauernde Anfragen können Anzeichen für einen Engpass sein.
avg_request_duration	bigint	Die durchschnittliche Dauer der Dateianfragen (in Mikrosekunden)

Spaltenname	Datentyp	Beschreibung
max_request_parallelism	integer	Die maximale Anzahl paralleler Anforderungen auf einem Slice für diese Amazon-Redshift-Abfrage.
avg_request_parallelism	double precision	Die durchschnittliche Anzahl paralleler Anforderungen auf einem Slice für diese Amazon-Redshift-Abfrage.
total_slowdown_count	bigint	Die Gesamtzahl von Amazon-S3-Anforderungen mit einem Verlangsamungsfehler, der beim San einer externen Tabelle auftrat.
max_slowdown_count	integer	Die maximale Anzahl von Amazon-S3-Anforderungen mit einem Verlangsamungsfehler, der während des Sans einer externen Tabelle auf einem Slice auftrat.

Beispielabfrage

Das folgende Beispiel ruft die Can-Schritt-Details für die letzte ausgeführte Abfrage ab.

```
select query, segment, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svcs_s3query_summary
where query = pg_last_query_id()
order by query,segment;
```

```
query | segment | elapsed | s3_scanned_rows | s3_scanned_bytes | s3query_returned_rows
 | s3query_returned_bytes | files
------+---------+---------+-----------------+------------------+----------------------
+------------------------+------
 4587 |       2 |   67811 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  591568 |          172462 |         11260097 |                 8513
 |                 170260 |     1
 4587 |       2 |  216849 |               0 |                0 |                    0
 |                      0 |     0
```

```
 4587 |       2 |  216671 |               0 |                0 |                      0
 |                     0 |     0
```

SVCS_STREAM_SEGS

Führt das Verhältnis zwischen Streams und gleichzeitigen Segmenten auf.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix STL vergleichbar, außer dass die STL-Tabellen nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_STREAM_SEGS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
stream	integer	Der Satz gleichzeitiger Segmente einer Abfrage.
segment	integer	Zahl, mit der das Abfrage-Segment identifiziert wird.

Beispielabfragen

Geben Sie die folgende Abfrage ein, um das Verhältnis zwischen Streams und gleichzeitigen Segmenten für die letzte Abfrage anzuzeigen:

```
select *
```

```
from svcs_stream_segs
where query = pg_last_query_id();

 query | stream | segment
-------+--------+---------
    10 |      1 |       2
    10 |      0 |       0
    10 |      2 |       4
    10 |      1 |       3
    10 |      0 |       1
(5 rows)
```

SVCS_UNLOAD_LOG

Verwenden Sie SVCS_UNLOAD_LOG, um Details zu UNLOAD-Operationen zu erhalten.

SVCS_UNLOAD_LOG zeichnet für jede von einer UNLOAD-Anweisung erstellte Datei eine Zeile auf. Zum Beispiel: Wenn ein UNLOAD-Vorgang 12 Dateien erstellt, enthält SVCS_UNLOAD_LOG 12 entsprechende Zeilen. Diese Ansicht wird aus der STL_UNLOAD_LOG-Systemtabelle abgeleitet, zeigt aber keine Slice-Ebene für Abfragen an, die auf einem Nebenläufigkeitsskalierungs-Cluster ausgeführt werden.

> **Note**
>
> Systemansichten mit dem Präfix SVCS enthalten Details zu Abfragen auf den Haupt- und Nebenläufigkeitsskalierungs-Clustern. Die Ansichten sind mit denen mit dem Präfix STL vergleichbar, außer dass die STL-Tabellen nur Informationen für Abfragen bereitstellen, die auf dem Haupt-Cluster ausgeführt werden.

SVCS_UNLOAD_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID.
pid	integer	Die mit der Abfrageanweisung verbundene Prozess-ID.
path (Pfad)	character(1280)	Der vollständige Amazon-S3-Objektpfad für die Datei.
start_time	timestamp	Die Startzeit für die UNLOAD-Operation.
end_time	timestamp	Die Endzeit für die UNLOAD-Operation.
line_count	bigint	Die Anzahl der in die Datei entladenen Zeilen.
transfer_size	bigint	Die Anzahl der übertragenen Bytes.
file_format	character(10)	Das Format der entladenen Datei.

Beispielabfrage

Um eine Liste der Dateien zu erhalten, die von einem UNLOAD-Befehl zu Amazon S3 geschrieben wurden, können Sie eine Amazon-S3-Listenoperation aufrufen, nachdem UNLOAD abgeschlossen ist. Je nachdem, wie schnell Sie diesen Aufruf starten, kann es jedoch sein, dass die Liste unvollständig ist, da es sich bei einer Amazon-S3-Listenoperation um einen Eventually-Consistent-Vorgang handelt. Fragen Sie SVCS_UNLOAD_LOG ab, um sofort eine vollständige und autoritative Liste zu erhalten.

Die folgende Abfrage gibt den Pfadnamen für Dateien aus, die durch einen UNLOAD-Befehl für die zuletzt ausgeführte Abfrage erstellt wurden:

```
select query, substring(path,0,40) as path
from svcs_unload_log
where query = pg_last_query_id()
order by path;
```

Dieser Befehl gibt die folgende Beispielausgabe aus:

```
 query |             path
```

```
 ------+---------------------------------
  2320 | s3://my-bucket/venue0000_part_00
  2320 | s3://my-bucket/venue0001_part_00
  2320 | s3://my-bucket/venue0002_part_00
  2320 | s3://my-bucket/venue0003_part_00
(4 rows)
```

SVL-Ansichten für den Haupt-Cluster

SVL-Ansichten sind Systemansichten in Amazon Redshift, die Verweise auf STL-Tabellen und -Protokolle für detailliertere Informationen enthalten.

Diese Ansichten ermöglichen den schnelleren und einfacheren Zugriff auf häufig abgefragte Daten in diesen Tabellen.

> **Note**
>
> Die Ansicht SVL_QUERY_SUMMARY enthält nur Informationen zu von Amazon Redshift ausgeführten Abfragen, nicht zu anderen Utility- und DDL-Befehlen. Für eine vollständige Liste und Informationen zu allen Anweisungen, die von Amazon Redshift ausgeführt werden, einschließlich DDL- und Utility-Befehlen, können Sie die SVL_STATEMENTTEXT-Ansicht abfragen.

Themen

- SVL_AUTO_WORKER_ACTION
- SVL_COMPILE
- SVL_DATASHARE_CHANGE_LOG
- SVL_DATASHARE_CROSS_REGION_USAGE
- SVL_DATASHARE_USAGE_CONSUMER
- SVL_DATASHARE_USAGE_PRODUCER
- SVL_FEDERATED_QUERY
- SVL_MULTI_STATEMENT_VIOLATIONS
- SVL_MV_REFRESH_STATUS
- SVL_QERROR

- SVL_QLOG
- SVL_QUERY_METRICS
- SVL_QUERY_METRICS_SUMMARY
- SVL_QUERY_QUEUE_INFO
- SVL_QUERY_REPORT
- SVL_QUERY_SUMMARY
- SVL_RESTORE_ALTER_TABLE_PROGRESS
- SVL_S3LIST
- SVL_S3LOG
- SVL_S3PARTITION
- SVL_S3PARTITION_SUMMARY
- SVL_S3QUERY
- SVL_S3QUERY_SUMMARY
- SVL_S3RETRIES
- SVL_SPATIAL_SIMPLIFY
- SVL_SPECTRUM_SCAN_ERROR
- SVL_STATEMENTTEXT
- SVL_STORED_PROC_CALL
- SVL_STORED_PROC_MESSAGES
- SVL_TERMINATE
- SVL_UDF_LOG
- SVL_USER_INFO
- SVL_VACUUM_PERCENTAGE

SVL_AUTO_WORKER_ACTION

Zeichnet automatisierte Aktionen von Amazon Redshift in Tabellen auf, die für die automatische Optimierung definiert sind.

SVL_AUTO_WORKER_ACTION ist nur für Superusers sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
table_id	integer	Die Tabellenkennung.
type	character (32)	Die Art der Empfehlung. Mögliche Werte sind distkey und sortkey.
status	character (128)	Der Abschlussstatus der Empfehlung. Mögliche Werte sind „Start“, „Complete“ (Abgeschlossen), „Skipped“ (Übersprungen), „Abort“ (Abbrechen), „Checkpoint“ (Prüfpunkt) und „Failed“ (Fehlgeschlagen).
eventtime	Zeitstempel	Der Zeitstempel der Spalte `status`.
sequence	integer	Die Sequenznummer eines abgeschnittenen `previous_state` -Werts. Wenn eine einzelner `previous_state` -Wert mehr als 200 Zeichen enthält, werden weitere Zeilen für diesen Wert protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite usw.
previous_state	character (200)	Der vorherige Verteilungsstil und die Sortierschlüssel der Tabelle, bevor die Empfehlung angewendet wird. Der Wert wird in Schritten von je 200 Zeichen abgeschnitten.

Einige Beispiele für Werte der Spalte `status` lauten wie folgt:

- Skipped:Table not found.
- Skipped:Recommendation is empty.
- Skipped:Apply sortkey recommendation is disabled.
- Skipped:Retry exceeds the maximum limit for a table.
- Skipped:Table column has changed.
- Abort:This table is not AUTO.
- Abort:This table has been recently converted.

- Abort:This table exceeds table size threshold.
- Abort:This table is already the recommended style.
- Checkpoint: progress *21.9963*%.

Beispielabfragen

Im folgenden Beispiel zeigen die Zeilen im Ergebnis Aktionen an, die von Amazon Redshift ausgeführt wurden.

```
select table_id, type, status, eventtime, sequence, previous_state
from SVL_AUTO_WORKER_ACTION;
```

```
 table_id |  type   |                        status                         |
 eventtime          | sequence | previous_state
----------+---------+-------------------------------------------------------
+----------------------------+----------+----------------
   118082 | sortkey | Start                                                 | 2020-08-22
 19:42:20.727049 | 0        |
   118078 | sortkey | Start                                                 | 2020-08-22
 19:43:54.728819 | 0        |
   118082 | sortkey | Start                                                 | 2020-08-22
 19:42:52.690264 | 0        |
   118072 | sortkey | Start                                                 | 2020-08-22
 19:44:14.793572 | 0        |
   118082 | sortkey | Failed                                                | 2020-08-22
 19:42:20.728917 | 0        |
   118078 | sortkey | Complete                                              | 2020-08-22
 19:43:54.792705 | 0        | SORTKEY: None;
   118086 | sortkey | Complete                                              | 2020-08-22
 19:42:00.72635  | 0        | SORTKEY: None;
   118082 | sortkey | Complete                                              | 2020-08-22
 19:43:34.728144 | 0        | SORTKEY: None;
   118072 | sortkey | Skipped:Retry exceeds the maximum limit for a table.  | 2020-08-22
 19:44:46.706155 | 0        |
   118086 | sortkey | Start                                                 | 2020-08-22
 19:42:00.685255 | 0        |
   118082 | sortkey | Start                                                 | 2020-08-22
 19:43:34.69531  | 0        |
   118072 | sortkey | Start                                                 | 2020-08-22
 19:44:46.703331 | 0        |
```

```
  118082 | sortkey | Checkpoint: progress 14.755079%                    | 2020-08-22
19:42:52.692828 | 0       |
  118072 | sortkey | Failed                                             | 2020-08-22
19:44:14.796071 | 0       |
  116723 | sortkey | Abort:This table is not AUTO.                      | 2020-10-28
05:12:58.479233 | 0       |
  110203 | distkey | Abort:This table is not AUTO.                      | 2020-10-28
05:45:54.67259  | 0       |
```

SVL_COMPILE

Zeichnet Kompilationszeit und Ort für jedes Abfragesegment auf.

SVL_COMPILE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> SVL_COMPILE enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_QUERY_HISTORY zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Weitere Informationen zu SVCS_COMPILE finden Sie unter SVCS_COMPILE.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Mit der Anweisung verbundene Transaktions-ID.
pid	integer	Die mit der Anweisung verbundene Prozess-ID.

Spaltenname	Datentyp	Beschreibung
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
segment	integer	Das zu kompilierende Abfragesegment.
locus	integer	Ort, an dem das Segment ausgeführt wird: **1**, wenn auf einem Datenverarbeitungsknoten, und **2**, wenn auf dem Führungsknoten.
starttime	timestamp	Zeitpunkt des Beginns der Kompilierung, nach UTC.
endtime	timestamp	Zeitpunkt des Endes der Kompilierung, nach UTC.
compile	integer	**0**, wenn die Kompilierung wiederverwendet wurde, **1**, wenn das Segment kompiliert wurde.

Beispielabfragen

In diesem Beispiel führten die Abfragen 35878 und 35879 die gleiche SQL-Anweisung aus. Die Kompilierungsspalte für Abfrage 35878 zeigt 1 für vier Abfragesegmente; dies zeigt an, dass die Segmente kompiliert wurden. Abfrage 35879 zeigt `0` in der Kompilierungsspalte für jedes Segment; dies zeigt an, dass die Segmente nicht erneut kompiliert werden müssen.

```
select userid, xid,  pid, query, segment, locus,
datediff(ms, starttime, endtime) as duration, compile
from svl_compile
where query = 35878 or query = 35879
order by query, segment;

 userid |  xid   |  pid  | query | segment | locus | duration | compile
--------+--------+-------+-------+---------+-------+----------+---------
    100 | 112780 | 23028 | 35878 |       0 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       1 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       2 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       3 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       4 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       5 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       6 |     1 |     1380 |       1
```

```
    100 | 112780 | 23028 | 35878 |       7 |     1 |     1085 |       1
    100 | 112780 | 23028 | 35878 |       8 |     1 |     1197 |       1
    100 | 112780 | 23028 | 35878 |       9 |     2 |      905 |       1
    100 | 112782 | 23028 | 35879 |       0 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       1 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       2 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       3 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       4 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       5 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       6 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       7 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       8 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       9 |     2 |        0 |       0
(20 rows)
```

SVL_DATASHARE_CHANGE_LOG

Zeichnet die konsolidierte Ansicht auf, um Änderungen an Datashares sowohl in Produzenten- als auch Konsumenten-Clustern nachzuverfolgen.

SVL_DATASHARE_CHANGE_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_DATESHARE_CHANGE_LOG zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der die Aktion ausführt.
username	varchar(128)	Der Name des Benutzers, der die Aktion ausführt.
pid	integer	Die ID des Prozesses.

Spaltenname	Datentyp	Beschreibung
xid	bigint	Die ID der Transaktion.
share_id	integer	Die ID des betroffenen Datashares.
share_name	varchar(128)	Der Name des Datashares.
source_database_id	integer	Die ID der Datenbank, zu der der Datashare gehört.
source_database_name	varchar(128)	Der Name der Datenbank, zu der der Datashare gehört.
consumer_database_id	integer	Die ID der Datenbank, die aus dem Datashare importiert wurde.
consumer_database_name	varchar(128)	Der Name der Datenbank, die aus dem Datashare importiert wurde.
arn	varchar(192)	Der ARN der Ressource, die die importierte Datenbank unterstützt.
recordtime	timestamp	Der Zeitstempel der Aktion.
Aktion	varchar(128)	Die Aktion, die ausgeführt wird. Mögliche Werte sind CREATE DATASHARE, DROP DATASHARE, GRANT ALTER, REVOKE ALTER, GRANT SHARE, REVOKE SHARE, ALTER ADD, ALTER REMOVE, ALTER SET, GRANT USAGE, REVOKE USAGE, CREATE DATABASE, GRANT oder REVOKE USAGE für eine gemeinsam genutzte Datenbank, DROP SHARED DATABASE, ALTER SHARED DATABASE.
Status	integer	Der Status der Aktion. Mögliche Werte sind SUCCESS und ERROR-ERROR CODE

Spaltenname	Datentyp	Beschreibung
share_object_type	varchar(64)	Der Typ des Datenbankobjekts, das dem Datashare hinzugefügt oder daraus entfernt wurde. Mögliche Werte sind Schemata, Tabellen, Spalten, Funktionen und Ansichten. Dies ist ein Feld für den Produzenten-Cluster.
share_object_id	integer	Die ID des Datenbankobjekts, das dem Datashare hinzugefügt oder daraus entfernt wurde. Dies ist ein Feld für den Produzenten-Cluster.
share_object_name	varchar(128)	Der Name des Datenbankobjekts, das dem Datashare hinzugefügt oder daraus entfernt wurde. Dies ist ein Feld für den Produzenten-Cluster.
target_user_type	varchar(16)	Der Typ der Benutzer oder Gruppen, denen eine Berechtigung erteilt wurde. Dies ist ein Feld sowohl für den Produzenten- als auch den Konsumenten-Cluster.
target_userid	integer	Die ID der Benutzer oder Gruppen, denen eine Berechtigung erteilt wurde. Dies ist ein Feld sowohl für den Produzenten- als auch den Konsumenten-Cluster.
target_username	varchar(128)	Der Name der Benutzer oder Gruppen, denen eine Berechtigung erteilt wurde. Dies ist ein Feld sowohl für den Produzenten- als auch den Konsumenten-Cluster.
consumer_account	varchar(16)	Die Konto-ID des Datenverbrauchers. Dies ist ein Feld für den Produzenten-Cluster.
consumer_namespace	varchar(64)	Der Namespace des Datenverbraucherkontos. Dies ist ein Feld für den Produzenten-Cluster.
producer_account	varchar(16)	Die Konto-ID des Produzentenkontos, zu dem der Datashare gehört. Dies ist ein Feld für den Konsumenten-Cluster.
producer_namespace	varchar(64)	Der Namespace des Produktkontos, zu dem der Datashare gehört. Dies ist ein Feld für den Konsumenten-Cluster.

Spaltenname	Datentyp	Beschreibung
attribute_name	varchar(64)	Der Name eines Attributs des Datashares oder der gemeinsamen Datenbank.
attribute_value	varchar(128)	Der Wert eines Attributs des Datashares oder der gemeinsamen Datenbank.
Nachricht	varchar(512)	Die Fehlermeldung, wenn eine Aktion fehlschlägt.

Beispielabfragen

Das folgende Beispiel zeigt eine Ansicht SVL_DATASHARE_CHANGE_LOG.

```
SELECT DISTINCT action
FROM svl_datashare_change_log
WHERE share_object_name LIKE 'tickit%';

         action
 ----------------------
  "ALTER DATASHARE ADD"
```

SVL_DATASHARE_CROSS_REGION_USAGE

Verwenden Sie die Ansicht SVL_DATASHARE_CROSS_REGION_USAGE, um einen Überblick über die regionsübergreifende Datenübertragungsnutzung zu erhalten, die durch eine regionsübergreifende Datasharing-Abfrage ausgelöst wurde. SVL_DATASHARE_CROSS_REGION_USAGE aggregiert Details auf Segmentebene.

SVL_DATASHARE_CROSS_REGION_USAGE ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_DATASHARE_CROSS_REGION_USAGE zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	Ganzzahl	Die ID der Abfrage. Verwenden Sie diesen Wert, um andere Systemtabellen und Ansichten zu verbinden.
segment	bigint	Die Nummer des Segments. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
start_time	variieren	Die Uhrzeit in UTC, zu der die Übertragung der Daten begann.
end_time	time	Die Uhrzeit in UTC, zu der die Übertragung der Daten endete.
transferred_data	bigint	Die Anzahl der Bytes an Daten, die von einer Produzentenregion an eine Konsumentenregion übertragen wurden.
source_region	char(25)	Die Produzentenregion, aus der die Abfrage Daten übertragen hat.
recordtime	timestamp	Die Zeit, zu der die Aktion aufgezeichnet wird.

Beispielabfragen

Das folgende Beispiel zeigt eine Ansicht von SVL_DATASHARE_CROSS_REGION_USAGE.

```
SELECT query, segment, transferred_data, source_region
from svl_datashare_cross_region_usage
where query = pg_last_query_id()
order by query,segment;

  query | segment | transferred_data | source_region
--------+---------+------------------+---------------
 200048 |       2 |          4194304 |    us-west-1
 200048 |       2 |          4194304 |    us-east-2
```

SVL_DATASHARE_USAGE_CONSUMER

Zeichnet die Aktivität und Verwendung von Datashares auf. Diese Ansicht ist nur für den Konsumenten-Cluster relevant.

SVL_DATASHARE_USAGE_CONSUMER ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_DATASHARE_USAGE_CONSUMER zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der die Anforderung ausgibt.
pid	integer	Die ID des Leader-Prozesses, der die Abfrage ausführt.
xid	bigint	Der Kontext der aktuellen Transaktion.
request_id	varchar(50)	Die eindeutige ID des angeforderten API-Aufrufs.
request_type	varchar(25)	Der Anforderungstyp, der an das Producer-Cluster getätigt wurde.
transaction_uid	varchar(50)	Die eindeutige ID der Transaktion.
recordtime	timestamp	Die Zeit, zu der die Aktion aufgezeichnet wird.
status	integer	Der Status des angeforderten API-Aufrufs.
error	varchar(512)	Die Meldung eines Fehlers.

Beispielabfragen

Das folgende Beispiel zeigt eine Ansicht SVL_DATASHARE_USAGE_CONSUMER.

```
SELECT request_type, status, trim(error) AS error
FROM svl_datashare_usage_consumer

  request_type  | status | error
----------------+--------+--------
 "GET RELATION" |   0    |
```

SVL_DATASHARE_USAGE_PRODUCER

Zeichnet die Aktivität und Verwendung von Datashares auf. Diese Ansicht ist nur für den Produzenten-Cluster relevant.

SVL_DATASHARE_USAGE_PRODUCER ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_DATASHARE_USAGE_PRODUCER zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
share_id	integer	Die Objekt-ID (OID) des Datenspeichers.
share_name	varchar(128)	Der Name des Datashares.
request_id	varchar(50)	Die eindeutige ID des angeforderten API-Aufrufs.
request_type	varchar(25)	Der Anforderungstyp, der an das Producer-Cluster getätigt wurde.

Spaltenname	Datentyp	Beschreibung
object_type	varchar(64)	Der Typ des Objekts, das aus dem Datashare freigegeben wird. Mögliche Werte sind Schemata, Tabellen, Spalten, Funktionen und Ansichten.
object_oid	integer	Die ID des Objekts, das aus dem Datashare freigegeben wird.
object_name	varchar(128)	Der Name des Objekts, das aus dem Datashare freigegeben wird.
consumer_account	varchar(16)	Das Konto des Konsumentenkontos, für das der Datashare freigegeben wird.
consumer_namespace	varchar(64)	Der Namespace des Konsumentenkontos, für das der Datashare freigegeben wird.
consumer_transaction_uid	varchar(50)	Die eindeutige Transaktions-ID der Anweisung im Konsumenten-Cluster.
recordtime	timestamp	Die Zeit, zu der die Aktion aufgezeichnet wird.
status	integer	Der Status des Datashares.
error	varchar(512)	Die Meldung eines Fehlers.
consumer_region	char(64)	Die Region, in der sich der Konsumenten-Cluster befindet.

Beispielabfragen

Das folgende Beispiel zeigt eine SVL_DATASHARE_USAGE_PRODUCER-Ansicht.

```
SELECT DISTINCT request_type
FROM svl_datashare_usage_producer
WHERE object_name LIKE 'tickit%';
```

```
  request_type
------------------
  "GET RELATION"
```

SVL_FEDERATED_QUERY

Verwenden Sie die Ansicht SVL_FEDERATED_QUERY, um Informationen zu einem Verbundabfragenaufruf anzuzeigen.

SVL_FEDERATED_QUERY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der die Abfrage ausführt.
xid	bigint	Die Transaktions-ID.
pid	integer	Die ID des Leader-Prozesses, der die Abfrage ausführt.
query	integer	Die Abfrage-ID eines Verbundaufrufs.
sourcetype	Zeichen (32)	Der Quelltyp für den Verbundaufruf, z.B. „PG“.
recordtime	timestamp	Der Zeitpunkt, zu dem eine Abfrage für den Verbund gesendet wird. UTC wird verwendet.

Spaltenname	Datentyp	Beschreibung
querytext	Zeichen (4000)	Die Abfragezeichenfolge, die zur Ausführung an die entfernte PostgreSQL-Engine gesendet wird.
num_rows	bigint	Die Anzahl der von der Verbundabfrage ausgegebenen Zeilen.
num_bytes	bigint	Die Anzahl der von der Verbundabfrage ausgegebenen Bytes.
duration	bigint	Die Zeit (Mikrosekunden), die mit dem Abrufen von Zeilen von Cursoraufrufen aufgewendet wurde. Dies ist die Zeit, die für die Ausführung der Verbundabfrage benötigt wurde und in der Ergebnisse zurückgegeben wurden.

Beispielabfragen

Führen Sie die folgende Abfrage aus, um Informationen zu Verbundabfragenaufrufen anzuzeigen.

```
select query, trim(sourcetype) as type, recordtime, trim(querytext) as "PG Subquery"
 from svl_federated_query where query = 4292;

 query | type |         recordtime         |                         pg subquery
-------+------+----------------------------
+-------------------------------------------------------------
  4292 | PG   | 2020-03-27 04:29:58.485126 | SELECT "level" FROM functional.employees
 WHERE ("level" >= 6)
(1 row)
```

SVL_MULTI_STATEMENT_VIOLATIONS

Verwenden Sie die Ansicht SVL_MULTI_STATEMENT_VIOLATIONS, um einen vollständigen Satz aller auf dem System ausgeführten SQL-Befehle zu erhalten, die gegen Einschränkungen von Transaktionsblöcken verstoßen.

Verstöße treten auf, wenn Sie einen der folgenden SQL-Befehle ausführen, die Amazon Redshift innerhalb eines Transaktionsblocks oder auf Anforderungen mit mehreren Anweisungen einschränkt:

- CREATE DATABASE
- DROP DATABASE
- ALTER TABLE APPEND
- CREATE EXTERNAL TABLE
- DROP EXTERNAL TABLE
- RENAME EXTERNAL TABLE
- ALTER EXTERNAL TABLE
- CREATE TABLESPACE
- DROP TABLESPACE
- CREATE LIBRARY
- DROP LIBRARY
- REBUILDCAT
- INDEXCAT
- REINDEX DATABASE
- VACUUM
- GRANT
- COPY

> **Note**
>
> Wenn in dieser Ansicht Einträge vorhanden sind, ändern Sie die entsprechenden Anwendungen und SQL-Skripte. Wir empfehlen, Ihren Anwendungscode zu ändern, um die Verwendung dieser eingeschränkten SQL-Befehle außerhalb des Transaktionsblocks

zu verschieben. Wenn Sie weitere Unterstützung benötigen, wenden Sie sich an den AWS Support.

SVL_MULTI_STATEMENT_VIOLATIONS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der die Verletzung verursacht hat.
Datenbank	character(32)	Der Name der Datenbank, mit der der Benutzer verbunden war.
cmdname	character(20)	Der Name des Befehls, der nicht innerhalb eines Transaktionsblocks oder einer Anforderung mit mehreren Anweisungen ausgeführt werden kann. Beispiel: CREATE DATABASE, DROP DATABASE, ALTER TABLE APPEND, CREATE EXTERNAL TABLE, DROP EXTERNAL TABLE, RENAME EXTERNAL TABLE, ALTER EXTERNAL TABLE, CREATE LIBRARY, DROP LIBRARY, REBUILDCAT, INDEXCAT, REINDEX DATABASE, VACUUM, GRANT auf externen Ressourcen, CLUSTER, COPY, CREATE TABLESPACE und DROP TABLESPACE.
xid	bigint	Die mit der Anweisung verbundene Transaktions-ID.
pid	integer	Die Prozess-ID für die Anweisung.

Spaltenname	Datentyp	Beschreibung
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist dieses Feld leer.
starttime	timestamp	Die genaue Uhrzeit, zu der die Ausführung der Anweisung begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile, zum Beispiel: **2009-06-12 11:29:19.131358**
endtime	timestamp	Die genaue Uhrzeit, zu der die Ausführung der Anweisung abgeschlossen wurde, mit 6 Nachkommastellen für Sekundenbruchteile, zum Beispiel: **2009-06-12 11:29:19.193640**
sequence	integer	Wenn eine einzelne Anweisung mehr als 200 Zeichen enthält, werden weitere Zeilen für diese Anweisung protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite usw.
type	varchar(10)	Der Typ der SQL-Anweisung: **QUERY**, **DDL** oder **UTILITY**.
Text	character(200)	Der SQL-Text, in Schritten von je 200 Zeichen. Diese Feld kann Sonderzeichen wie Backslash (\\) und Zeilenumbruch (\n) enthalten.

Beispielabfrage

Die folgende Abfrage gibt mehrere Anweisungen zurück, die Verstöße aufweisen.

```
select * from svl_multi_statement_violations order by starttime asc;

userid | database | cmdname | xid | pid | label | starttime | endtime | sequence | type
 | text
================================================================================================
```

```
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ********* | ******* | 0 | DDL |
 create table c(b int);
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ********* | ******* | 0 | UTILITY |
 create database b;
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ********* | ******* | 0 | UTILITY |
 COMMIT
...
```

SVL_MV_REFRESH_STATUS

Die Ansicht SVL_MV_REFRESH_STATUS enthält eine Zeile für die Aktualisierungsaktivität materialisierter Ansichten.

Weitere Hinweise zu materialisierten Ansichten finden Sie unter Erstellen von materialisierten Ansichten in Amazon Redshift.

SVL_MV_REFRESH_STATUS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_MV_REFRESH_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
db_name	char(128)	Die Datenbank, die die materialisierte Ansicht enthält.
userid	bigint	Die ID des Benutzers, der die Aktualisierung durchgeführt hat.
schema_name	char(128)	Das Schema der materialisierten Ansicht.
mv_name	char(128)	Der Name der materialisierten Ansicht.
xid	bigint	Die Transaktions-ID der Aktualisierung.

Spaltenname	Datentyp	Beschreibung
starttime	timestamp	Die Anfangszeit der Aktualisierung.
endtime	timestamp	Die Endzeit der Aktualisierung.

Spaltenname	Datentyp	Beschreibung
status	Text	Der Status der Aktualisierung. Zu den Beispielw erten gehören: • MV durch Aktualisierung erfolgreich inkrement ell aktualisiert Wenn es sich um eine materialisierte Ansicht für das Streaming handelt, kann die Meldung zusätzliche Qualifizierer in Bezug auf die Anzahl der Datensätze enthalten. Diese umfassen u. a. folgende: • Stream hat keine neuen Daten zurückgeg eben – Es wurden keine Datensätze abgerufen. • Alle aus dem Stream empfangenen Datensätze wurden übersprungen. – Datensätze wurden abgerufen, aufgrund von Fehlern wurden jedoch alle übersprungen. • Einige Stream-Datensätze wurden übersprun gen. – Datensätze wurden abgerufen, aufgrund von Fehlern wurden einige jedoch übersprungen. Wenn es keine Qualifizierer gibt, wurde mindestens ein Datensatz abgerufen und alle Datensätze sind in der materialisierten Ansicht verfügbar. Es gibt noch einen möglichen Qualifizierer: • Der Stream enthält möglicherweise weitere Daten – Die Aktualisierung wurde beendet, bevor Amazon Redshift festgestellt hat, dass keine weiteren Datensätze zum Aufnehmen vorhanden waren. Der Stream kann aktuell

Spaltenname	Datentyp	Beschreibung
		sein, wurde von Amazon Redshift jedoch nicht bestätigt. • MV durch Aktualisierung erfolgreich von Grund auf neu berechnet • MV durch Aktualisierung teilweise bis zu einer aktiven Transaktion aktualisiert • MV wurde bereits aktualisiert. • Aktualisierung fehlgeschlagen. Eine Basistabellenspalte wurde umbenannt. • Aktualisierung fehlgeschlagen. Ein Basistabellenspaltentyp wurde geändert. • Aktualisierung fehlgeschlagen. Eine Basistabelle wurde umbenannt. • Die Aktualisierung ist aufgrund eines internen Fehlers fehlgeschlagen. • Aktualisierung fehlgeschlagen. Eine Basistabellenspalte wurde gelöscht. • Aktualisierung fehlgeschlagen. Schema von MV wurde umbenannt • Aktualisierung fehlgeschlagen. MV wurde nicht gefunden. • Automatische Aktualisierung aufgrund übermäßiger Benutzer-Workload abgebrochen • Aktualisierung fehlgeschlagen. Serialisierbare Isolationsverletzung
refresh_type	char(32)	Die Definition des Aktualisierungstyps. Beispielwerte umfassen Folgendes: Manual und Auto.

Beispielabfrage

Führen Sie die folgende Abfrage aus, um den Aktualisierungsstatus von materialisierten Ansichten anzuzeigen.

```
select * from svl_mv_refresh_status;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
 db_name | userid |  schema   |  name   |  xid  |         starttime          |
       endtime           |                        status                        |
 refresh_type
---------+--------+-----------+---------+-------+----------------------------
+----------------------------+------------------------------------------------------
+-----------------
 dev     |    169 | mv_schema | mv_test |  6640 | 2020-02-14 02:26:53.497935 |
 2020-02-14 02:26:53.556156 | Refresh successfully recomputed MV from scratch     |
 Manual
 dev     |    166 | mv_schema | mv_test |  6517 | 2020-02-14 02:26:39.287438 |
 2020-02-14 02:26:39.349539 | Refresh successfully updated MV incrementally       |
 Auto
 dev     |    162 | mv_schema | mv_test |  6388 | 2020-02-14 02:26:27.863426 |
 2020-02-14 02:26:27.918307 | Refresh successfully recomputed MV from scratch     |
 Manual
 dev     |    161 | mv_schema | mv_test |  6323 | 2020-02-14 02:26:20.020717 |
 2020-02-14 02:26:20.080002 | Refresh successfully updated MV incrementally       |
 Auto
 dev     |    161 | mv_schema | mv_test |  6301 | 2020-02-14 02:26:05.796146 |
 2020-02-14 02:26:07.853986 | Refresh successfully recomputed MV from scratch     |
 Manual
 dev     |    153 | mv_schema | mv_test |  6024 | 2020-02-14 02:25:18.762335 |
 2020-02-14 02:25:20.043462 | MV was already updated                              |
 Manual
 dev     |    143 | mv_schema | mv_test |  5557 | 2020-02-14 02:24:23.100601 |
 2020-02-14 02:24:23.100633 | MV was already updated                              |
 Manual
 dev     |    141 | mv_schema | mv_test |  5447 | 2020-02-14 02:23:54.102837 |
 2020-02-14 02:24:00.310166 | Refresh successfully updated MV incrementally       |
 Auto
 dev     |      1 | mv_schema | mv_test |  5329 | 2020-02-14 02:22:26.328481 |
 2020-02-14 02:22:28.369217 | Refresh successfully recomputed MV from scratch     |
 Auto
```

```
dev     |    138 | mv_schema | mv_test |  5290 | 2020-02-14 02:21:56.885093 |
2020-02-14 02:21:56.885098 | Refresh failed. MV was not found                    |
Manual
```

SVL_QERROR

Die Ansicht SVL-QERROR ist veraltet.

SVL_QLOG

Die Ansicht SVL_QLOG enthält ein Protokoll aller gegen die Datenbank ausgeführter Abfragen.

Amazon Redshift erstellt die SVL_QLOG-Ansicht als lesbare Teilmenge der Informationen aus der Tabelle STL_QUERY. Verwenden Sie diese Tabelle zum Finden der Abfrage-ID für eine kürzlich ausgeführte Abfrage finden, oder um zu sehen, wie lange eine Abfrage in Anspruch nahm.

SVL_QLOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Sie können diese ID verwenden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
xid	bigint	Transaktions-ID.
pid	integer	Die mit der Abfrage verbundene Prozess-ID.

Spaltenname	Datentyp	Beschreibung
starttime	timestamp	Die genaue Uhrzeit, zu der die Ausführung der Anweisung begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile – zum Beispiel: **2009-06-12 11:29:19.131358**
endtime	timestamp	Die genaue Uhrzeit, zu der die Ausführung der Anweisung abgeschlossen wurde, mit 6 Nachkommastellen für Sekundenbruchteile – zum Beispiel: **2009-06-12 11:29:19.193640**
elapsed	bigint	Zeitdauer der Ausführung der Abfrage (in Mikrosekunden).
aborted	integer	Wenn eine Abfrage vom System oder einem Benutzer beendet wurde, enthält diese Spalte den Wert **1**. Wenn die Abfrage abgeschlossen wurde, enthält diese Spalte **0**. Abfragen, die zu Workload-Verwaltungszwecken abgebrochen und anschließend erneut gestartet wurden, haben ebenfalls den Wert **1** in dieser Spalte.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist der Wert in diesem Feld `default`.
substring	character(60)	Gekürzter Abfragetext.
source_query	integer	Wenn für die Abfrage Ergebniszwischenspeicherung verwendet wird, ist dies die Abfrage-ID der Abfrage, anhand derer die zwischengespeicherten Ergebnisse erstellt wurden. Wenn keine Ergebniszwischenspeicherung verwendet wird, ist dieser Feldwert NULL.

Spaltenname	Datentyp	Beschreibung
concurrency_scaling_status_txt	Text	Eine Beschreibung, ob die Abfrage auf dem Haupt-Cluster oder Nebenläufigkeitsskalierungs-Cluster ausgeführt wurde.
from_sp_call	integer	Wenn die Abfrage von einer gespeicherten Prozedur aufgerufen wurde, die Abfrage-ID des Prozeduraufrufs. Wenn die Abfrage nicht als Teil einer gespeicherten Prozedur ausgeführt wurde, ist dieses Feld `NULL`.

Beispielabfragen

Das folgende Beispiel gibt die Abfrage-ID, die Ausführungszeit und den gekürzten Abfragetext für die fünf letzten Datenbankabfragen aus, die der Benutzer mit `userid = 100` ausgeführt hat.

```
select query, pid, elapsed, substring from svl_qlog
where userid = 100
order by starttime desc
limit 5;

 query  |  pid  | elapsed  |           substring
--------+-------+----------+-----------------------------------------------
 187752 | 18921 | 18465685 | select query, elapsed, substring from svl_...
 204168 |  5117 |    59603 | insert into testtable values (100);
 187561 | 17046 |  1003052 | select * from pg_table_def where tablename...
 187549 | 17046 |  1108584 | select * from STV_WLM_SERVICE_CLASS_CONFIG
 187468 | 17046 |  5670661 | select * from pg_table_def where schemaname...
(5 rows)
```

Das folgende Beispiel gibt den Namen des SQL-Scripts (Spalte LABEL) und die verstrichene Zeit für eine Abfrage zurück, die abgebrochen wurde (**`aborted=1`**):

```
select query, elapsed, trim(label) querylabel
from svl_qlog where aborted=1;

 query | elapsed  |       querylabel
-------+----------+-------------------------
```

```
    16 |  6935292 | alltickittablesjoin.sql
(1 row)
```

SVL_QUERY_METRICS

Die Ansicht SVL_QUERY_METRICS zeigt die Metriken für abgeschlossene Abfragen an. Diese Ansicht ist aus der Systemtabelle STL_QUERY_METRICS abgeleitet. Verwenden Sie die Werte in dieser Ansicht als Hilfe bei der Festlegung von Schwellenwerten zur Definition von Abfrageüberwachungsregeln. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.

SVL_QUERY_METRICS ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der die Abfrage ausgeführt hat, die den Eintrag generierte.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
service_class	integer	ID für die WLM-Abfragewartschlange (Service-Klasse). Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Nur für benutzerdefinierte Abfragen werden Metriken gemeldet. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
dimension	varchar(24)	Dimension, für die die Metriken gemeldet werden. Mögliche Werte sind: „query“, „segment“ und „step“.

Spaltenname	Datentyp	Beschreibung
segment	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt. Wenn der Segmentwert 0 ist, werden Segment-Metrikwerte in der Abfrageebene zusammengeführt.
Schritt	integer	ID für die Art des ausgeführten Schritts. Die Beschreibung für den Schritt wird in der Spalte `step_label` angezeigt.
step_label	varchar(30)	Art des ausgeführten Schritts.
query_cpu_time	bigint	Von der Abfrage verwendete CPU-Zeit, in Sekunden. Die CPU-Zeit unterscheidet sich von der Ausführungszeit der Abfrage.
query_blocks_read	bigint	Anzahl der von der Abfrage gelesenen 1 MB-Blöcke.
query_execution_time	bigint	Verstrichene Ausführungszeit für eine Abfrage, in Sekunden. Die Ausführungszeit enthält nicht die in einer Warteschlange verbrachte Zeit. Die Zeit in der Warteschlange finden Sie unter `query_queue_time` .
query_cpu_usage_percent	bigint	Prozentsatz der von der Abfrage verwendeten CPU-Kapazität.
query_temp_blocks_to_disk	bigint	Der Festplattenspeicherplatz, der von einer Abfrage zum Schreiben von Zwischenergebnissen verwendet wird, in MB.
segment_execution_time	bigint	Verstrichene Ausführungszeit für ein einzelnes Segment, in Sekunden.
cpu_skew	numeric(38,2)	Das Verhältnis der maximalen CPU-Nutzung für einen Slice zur durchschnittlichen CPU-Nutzung für alle Slices. Diese Metrik wird auf Segmentebene definiert.

Spaltenname	Datentyp	Beschreibung
io_skew	numeric(38,2)	Das Verhältnis der maximalen Zahl der gelesenen Blöcke (I/O) für einen Slice zur durchschnittlichen Zahl der gelesenen Blöcke für alle Slices.
scan_row_count	bigint	Die Anzahl der Zeilen in einem Scan-Schritt. Die Anzahl der Zeilen ist die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter ausgegeben wurden.
join_row_count	bigint	Die Anzahl der in einem Join-Schritt verarbeiteten Zeilen.
nested_loop_join_row_count	bigint	Die Anzahl der Zeilen in einem eingebetteten Loop-Join.
return_row_count	bigint	Die Anzahl der von der Abfrage ausgegebenen Zeilen.
spectrum_scan_row_count	bigint	Die Anzahl der Zeilen, die von einer Amazon-Redshift-Spectrum-Abfrage in Amazon S3 gescannt wurden.
spectrum_scan_size_mb	bigint	Die Datenmenge in MB, die von einer Amazon-Redshift-Spectrum-Abfrage in Amazon S3 gescannt wurde.
query_queue_time	bigint	Die Zeitspanne in Sekunden, über die hinweg sich die Abfrage in der Warteschlange befand.

SVL_QUERY_METRICS_SUMMARY

Die Ansicht SVL_QUERY_METRICS_SUMMARY zeigt die Maximalwerte für Metriken für abgeschlossene Abfragen. Diese Ansicht ist aus der Systemtabelle STL_QUERY_METRICS abgeleitet. Verwenden Sie die Werte in dieser Ansicht als Hilfe bei der Festlegung von Schwellenwerten zur Definition von Abfrageüberwachungsregeln. Weitere Informationen zu Regeln und Metriken für die Abfrageüberwachung für Amazon Redshift finden Sie unter WLM-Abfrageüberwachungsregeln.

SVL_QUERY_METRICS_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der die Abfrage ausgeführt hat, die den Eintrag generierte.
query	integer	Abfrage-ID. Die Abfrage-Spalte kann verwendet werden, um andere Systemtabellen und Anzeigen anzufügen.
service_class	integer	ID für die WLM-Abfragewartschlange (Service-Klasse). Abfragewarteschlangen werden in der WLM-Konfiguration definiert. Nur für benutzerdefinierte Abfragen werden Metriken gemeldet. Eine Liste von Serviceklassen-IDs finden Sie unter WLM-Serviceklassen-IDs.
query_cpu_time	bigint	Von der Abfrage verwendete CPU-Zeit, in Sekunden. Die CPU-Zeit unterscheidet sich von der Ausführungszeit der Abfrage.
query_blocks_read	bigint	Anzahl der von der Abfrage gelesenen 1 MB-Blöcke.
query_execution_time	bigint	Verstrichene Ausführungszeit für eine Abfrage, in Sekunden. Die Ausführungszeit enthält nicht die in einer Warteschlange verbrachte Zeit.
query_cpu_usage_percent	numeric(38,2)	Prozentsatz der von der Abfrage verwendeten CPU-Kapazität.

Spaltenname	Datentyp	Beschreibung
query_temp_blocks_to_disk	bigint	Der Festplattenspeicherplatz, der von einer Abfrage zum Schreiben von Zwischenergebnissen verwendet wird, in MB.
segment_execution_time	bigint	Verstrichene Ausführungszeit für ein einzelnes Segment, in Sekunden.
cpu_skew	numeric(38,2)	Das Verhältnis der maximalen CPU-Nutzung für einen Slice zur durchschnittlichen CPU-Nutzung für alle Slices. Diese Metrik wird auf Segmentebene definiert.
io_skew	numeric(38,2)	Das Verhältnis der maximalen Zahl der gelesenen Blöcke (I/O) für einen Slice zur durchschnittlichen Zahl der gelesenen Blöcke für alle Slices.
scan_row_count	bigint	Die Anzahl der Zeilen in einem Scan-Schritt. Die Anzahl der Zeilen ist die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter ausgegeben wurden.
join_row_count	bigint	Die Anzahl der in einem Join-Schritt verarbeiteten Zeilen.
nested_loop_join_row_count	bigint	Die Anzahl der Zeilen in einem eingebetteten Loop-Join.
return_row_count	bigint	Die Anzahl der von der Abfrage ausgegebenen Zeilen.
spectrum_scan_row_count	bigint	Die Anzahl der Zeilen, die von einer Amazon-Redshift-Spectrum-Abfrage in Amazon S3 gescannt wurden.
spectrum_scan_size_mb	bigint	Die Datenmenge in MB, die von einer Amazon-Redshift-Spectrum-Abfrage in Amazon S3 gescannt wurde.
query_queue_time	bigint	Die Zeitspanne in Sekunden, über die hinweg sich die Abfrage in der Warteschlange befand.

SVL_QUERY_QUEUE_INFO

Fasst die Details für Abfragen zusammen, die in einer Workload Management (WLM)-Abfragewarteschlange oder Commit-Warteschlange Zeit verbracht haben.

Die Ansicht SVL_QUERY_QUEUE_INFO filtert die vom System ausgeführten Abfragen aus und zeigt nur Abfragen an, die von einem Benutzer ausgeführt wurden.

Die Ansicht SVL_QUERY_QUEUE_INFO summiert Informationen aus den Systemtabellen STL_QUERY, STL_WLM_QUERY und STL_COMMIT_STATS.

SVL_QUERY_QUEUE_INFO ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
Datenbank	Text	Der Name der Datenbank, mit der der Benutzer verbunden war, als die Abfrage ausgegeben wurde.
query	integer	Abfrage-ID.
xid	bigint	Transaktions-ID.
userid	integer	ID des Benutzers, der die Abfrage generiert hat.
querytxt	Text	Die ersten 100 Zeichen des Abfragetexts.
queue_start_time	timestamp	Zeitpunkt, an dem die Abfrage in die WLM-Warteschlange gesetzt wurde, nach UTC.
exec_start_time	timestamp	Zeitpunkt, an dem die Ausführung der Abfrage gestartet wurde, nach UTC.
service_class	integer	ID für die Service-Klasse. Service-Klassen sind in der WLM-Konfigurationsdatei definiert.
slots	integer	Anzahl der WLM-Abfrageslots.

Spaltenname	Datentyp	Beschreibung
queue_elapsed	bigint	Zeitraum, für den die Abfrage in einer WLM-Warteschlange wartete (in Sekunden).
exec_elapsed	bigint	Zeitraum der Ausführung der Abfrage (in Sekunden).
wlm_total_elapsed	bigint	Zeitraum, den die Abfrage in einer WLM-Warteschlange verbrachte (queue_elapsed), zuzüglich des Zeitraums für die Ausführung der Abfrage (exec_elapsed).
commit_queue_elapsed	bigint	Zeitraum, für den die Abfrage in einer Commit-Warteschlange wartete (in Sekunden).
commit_exec_time	bigint	Zeitraum, für den die Abfrage in einer Commit-Operation verbrachte (in Sekunden).
service_class_name	character(64)	Der Name der Serviceklasse.

Beispielabfragen

Das folgende Beispiel zeigt die Zeit an, die Abfragen in WLM-Warteschlangen verbrachten.

```
select query, service_class, queue_elapsed, exec_elapsed, wlm_total_elapsed
from svl_query_queue_info
where wlm_total_elapsed > 0;

  query  | service_class | queue_elapsed | exec_elapsed | wlm_total_elapsed
---------+---------------+---------------+--------------+-------------------
 2742669 |             6 |             2 |          916 |               918
 2742668 |             6 |             4 |          197 |               201
(2 rows)
```

SVL_QUERY_REPORT

Amazon Redshift erstellt die Ansicht SVL_QUERY_REPORT aus einer UNION einer Reihe von Amazon-Redshift-STL-Systemtabellen, um Informationen zu ausgeführten Abfrageschritten anzuzeigen.

Diese Ansicht schlüsselt die Informationen zu ausgeführten Abfragen nach Slice und Schritt auf; dies ist nützlich für die Behebung von Slice- und Knoten-Problemen im Amazon-Redshift-Cluster.

SVL_QUERY_REPORT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
slice	integer	Daten-Slice der Ausführung des Schritts.
segment	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt.
Schritt	integer	Abfrageschritt, der abgeschlossen wurde.

Spaltenname	Datentyp	Beschreibung
start_time	timestamp	Genaue Uhrzeit in UTC, zu der die Ausführung des Segments begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile. Beispiel: **2012-12-12 11:29:19.131358**
end_time	timestamp	Genaue Uhrzeit in UTC, zu der die Ausführung des Segments abgeschlossen wurde, mit 6 Nachkommastellen für Sekundenbruchteile. Beispiel: **2012-12-12 11:29:19.131467**
elapsed_time	bigint	Zeit (in Mikrosekunden) für die Ausführung des Segments.
rows	bigint	Anzahl der von dem Schritt (pro Slice) produzierten Zeilen. Diese Zahl steht für die Anzahl der Zeilen für den Slice, die aus der Ausführung des Schritts resultieren, nicht für die Anzahl der Zeilen, die von dem Schritt empfangen oder verarbeitet wurden. Anders ausgedrückt ist dies die Anzahl der Zeilen, die den Schritt überstehen und an den nächsten Schritt weitergegeben werden.
bytes	bigint	Anzahl der von dem Schritt (pro Slice) produzierten Bytes.
label	char(256)	Schritt-Etikett, bestehend aus dem Namen eines Abfrageschritts und, falls anwendbar, der Tabellen-ID und dem Tabellennamen (zum Beispiel)., `scan tbl=100448 name =user`). Dreistellige Tabellen-IDs beziehen sich normalerweise auf Scans temporärer Tabellen. Wenn Sie `tbl=0` sehen, bezieht sich dies auf einen Scan eines konstanten Werts.
is_diskbased	character (1)	Ob dieser Schritt der Abfrage als festplattenbasierte Operation ausgeführt wurde: true (**t**) oder false (**f**). Nur bestimmte Schritte, wie etwa Hash-, Sortierungs- oder Aggregierungsschritte, können auf die Festplatte übertragen werden. Viele Arten von Schritten werden immer im Arbeitsspeicher ausgeführt.

Spaltenname	Datentyp	Beschreibung
workmem	bigint	Größe des Arbeitsspeichers (in Byte), der dem Abfrageschritt zugewiesen wurde. Dies ist der query_working_mem-Schwellenwert, der während der Ausführung zur Verwendung zugewiesen wird, nicht die Speicherm enge, die tatsächlich genutzt wurde.
is_rrscan	character (1)	„true“ (`t`) zeigt an, dass für diesen Schritt ein Scan mit Bereichse inschränkung durchgeführt wurde.
is_delayed_scan	character (1)	„true“ (`t`) zeigt an, dass für diesen Schritt ein verzögerter Scan durchgeführt wurde.
rows_pre_filter	bigint	Bei Scans permanenter Tabellen die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen ausgegeben wurden (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefi lter.

Beispielabfragen

Die folgende Abfrage zeigt die Datenverzerrung der von der Abfrage mit der Abfrage-ID 279 ausgegebenen Zeilen. Verwenden Sie diese Abfrage, um festzustellen, ob die Datenbank gleichmäßig über die Slices in dem Data Warehouse-Cluster verteilt sind:

```
select query, segment, step, max(rows), min(rows),
case when sum(rows) > 0
then ((cast(max(rows) -min(rows) as float)*count(rows))/sum(rows))
else 0 end
from svl_query_report
where query = 279
group by query, segment, step
order by segment, step;
```

Diese Abfrage sollte der folgenden Beispielausgabe ähnliche Daten ausgeben:

```
query | segment | step |   max    |   min    |         case
------+---------+------+----------+----------+----------------------
279 |       0 |    0 | 19721687 | 19721687 |                    0
```

```
279 |       0 |    1 | 19721687 | 19721687 |                    0
279 |       1 |    0 |   986085 |   986084 | 1.01411202804304e-06
279 |       1 |    1 |   986085 |   986084 | 1.01411202804304e-06
279 |       1 |    4 |   986085 |   986084 | 1.01411202804304e-06
279 |       2 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       2 |    2 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    2 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    3 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    1 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    2 |        1 |        1 |                    0
279 |       5 |    0 |        1 |        1 |                    0
279 |       5 |    1 |        1 |        1 |                    0
279 |       6 |    0 |       20 |       20 |                    0
279 |       6 |    1 |        1 |        1 |                    0
279 |       7 |    0 |        1 |        1 |                    0
279 |       7 |    1 |        0 |        0 |                    0
(19 rows)
```

SVL_QUERY_SUMMARY

Verwenden Sie die Ansicht SVL_QUERY_SUMMARY, um allgemeine Informationen zur Ausführung einer Abfrage zu erhalten.

Die Ansicht SVL_QUERY_SUMMARY enthält einen Teil der Daten der Ansicht SVL_QUERY_REPORT. Beachten Sie, dass die Informationen in SVL_QUERY_SUMMARY von allen Knoten aggregiert sind.

> **Note**
>
> Die Ansicht SVL_QUERY_SUMMARY enthält nur Informationen zu von Amazon Redshift ausgeführten Abfragen, nicht zu anderen Utility- und DDL-Befehlen. Für eine vollständige Liste und Informationen zu allen von Amazon Redshift ausgeführten Anweisungen, einschließlich DDL- und Utility-Befehlen, können Sie die Ansicht SVL_STATEMENTTEXT abfragen.

SVL_QUERY_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Informationen zu SVCS_QUERY_SUMMARY finden Sie unter SVCS_QUERY_SUMMARY.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
query	integer	Abfrage-ID. Kann verwendet werden, um verschiedene andere Systemtabellen und Anzeigen anzufügen.
stm	integer	Stream: Ein Satz gleichzeitiger Segmente einer Abfrage. Eine Abfrage enthält einen oder mehrere Streams.
seg	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten , und jedes Segment besteht aus einem oder mehreren Schritten . Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt.
Schritt	integer	Abfrageschritt, der ausgeführt wurde.
maxtime	bigint	Maximale Zeit für die Ausführung des Schritts (in Mikrosekunden).
avgtime	bigint	Durchschnittliche Zeit für die Ausführung des Schritts (in Mikrosekunden).
rows	bigint	Anzahl der an dem Abfrageschritt beteiligten Datenzeilen.
bytes	bigint	Anzahl der an dem Abfrageschritt beteiligten Bytes.
rate_row	double precision	Ausführungsrate der Abfrage pro Zeile.

Spaltenname	Datentyp	Beschreibung
rate_byte	double precision	Ausführungsrate der Abfrage pro Byte.
label	Text	Schritt-Etikett, bestehend aus dem Namen eines Abfragesc hritts und, falls anwendbar, der Tabellen-ID und dem Tabellenn amen (zum Beispiel scan tbl = 100448). Dreistellige Tabellen-IDs beziehen sich normalerweise auf Scans temporärer Tabellen. Wenn Sie `tbl=0` sehen, bezieht sich dies auf einen Scan eines konstanten Werts.
is_diskba sed	character(1)	Ob dieser Schritt der Abfrage als datenträgerbasierte Operation auf einem Knoten im Cluster ausgeführt wurde: true (**t**) oder false (**f**). Nur bestimmte Schritte, wie etwa Hash-, Sortierungs- oder Aggregierungsschritte, können auf die Festplatte übertragen werden. Viele Arten von Schritten werden immer im Arbeitsspeicher ausgeführt.
workmem	bigint	Größe des Arbeitsspeichers (in Byte), der dem Abfrageschritt zugewiesen wurde.
is_rrscan	character(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein Scan mit Bereichse inschränkung durchgeführt wurde. Der Standardwert ist „false“ (**f**).
is_delaye d_scan	character(1)	„true“ (**t**) zeigt an, dass für diesen Schritt ein verzögerter Scan durchgeführt wurde. Der Standardwert ist „false“ (**f**).
rows_pre_ filter	bigint	Für Scans permanenter Tabellen die Gesamtzahl der Zeilen, die vor der Filterung der zur Löschung markierten Zeilen (Geisterzeilen) ausgegeben wurden.

Beispielabfragen

Anzeige von Verarbeitungsinformationen für einen Abfrageschritt

Die folgende Abfrage zeigt allgemeine Verarbeitungsinformationen für jeden Schritt von Abfrage 87:

```
select query, stm, seg, step, rows, bytes
from svl_query_summary
where query = 87
order by query, seg, step;
```

Diese Abfrage ruft die Verarbeitungsinformationen zu Abfrage 87 aus, wie die folgende Beispielausgabe zeigt:

```
 query | stm | seg | step |  rows  |  bytes
-------+-----+-----+------+--------+---------
87     |   0 |   0 |    0 |     90 |    1890
87     |   0 |   0 |    2 |     90 |     360
87     |   0 |   1 |    0 |     90 |     360
87     |   0 |   1 |    2 |     90 |    1440
87     |   1 |   2 |    0 | 210494 | 4209880
87     |   1 |   2 |    3 |  89500 |       0
87     |   1 |   2 |    6 |      4 |      96
87     |   2 |   3 |    0 |      4 |      96
87     |   2 |   3 |    1 |      4 |      96
87     |   2 |   4 |    0 |      4 |      96
87     |   2 |   4 |    1 |      1 |      24
87     |   3 |   5 |    0 |      1 |      24
87     |   3 |   5 |    4 |      0 |       0
(13 rows)
```

Ermittlung, ob Abfrageschritte auf den Datenträger übertragen wurden

Die folgende Abfrage zeigt, ob einer oder mehrere der Schritte für die Abfrage mit Abfrage-ID 1025 (vgl. die Ansicht SVL_QLOG zum Abruf der Abfrage-ID für eine Abfrage) auf die Festplatte übertragen wurde, oder ob die Abfrage vollständig im Speicher ausgeführt wurde:

```
select query, step, rows, workmem, label, is_diskbased
from svl_query_summary
where query = 1025
order by workmem desc;
```

Diese Abfrage gibt die folgende Beispielausgabe zurück:

```
query| step|  rows  |  workmem   |  label        | is_diskbased
-----+-----+--------+------------+---------------+--------------
```

```
1025 |  0  |16000000|  141557760 |scan tbl=9     | f
1025 |  2  |16000000|  135266304 |hash tbl=142   | t
1025 |  0  |16000000|  128974848 |scan tbl=116536| f
1025 |  2  |16000000|  122683392 |dist           | f
(4 rows)
```

Durch das Scannen der Werte für IS_DISKBASED sehen Sie, welche Abfrageschritte auf die Festplatte übertragen wurden. Bei Abfrage 1025 wurde der Hash-Schritt auf der Festplatte ausgeführt. Schritte, die auf der Festplatte ausgeführt werden können, sind u. a. Hash-, Aggregierungs- und Sortierungsschritte. Um ausschließlich datenträgerbasierte Abfrageschritte anzuzeigen, fügen Sie der SQL-Anweisung im Beispiel oben die Klausel **`and is_diskbased = 't'`** hinzu.

SVL_RESTORE_ALTER_TABLE_PROGRESS

Verwenden Sie SVL_RESTORE_ALTER_TABLE_PROGRESS, um den Migrationsfortschritt jeder Tabelle im Cluster während einer klassischen Größenänderung auf RA3-Knoten zu überwachen. Dies erfasst den historischen Durchsatz der Datenmigration während der Größenänderung. Weitere Informationen zur klassischen Größenänderung auf RA3-Knoten finden Sie unter Klassische Größenänderung.

SVL_RESTORE_ALTER_TABLE_PROGRESS ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_RESTORE_LOG zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

> **Note**
>
> Zeilen mit einem Fortschritt von oder werden nach 7 Tagen gelöscht. `100.00%` `ABORTED` Zeilen für Tabellen, die während oder nach einer klassischen Größenänderung gelöscht wurden, können weiterhin in SVL_RESTORE_ALTER_TABLE_PROGRESS erscheinen.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
tbl	Ganzzahl	Die ID der Tabelle.
progress	char(32)	Der Status des Weiterverteilungsfortschritts der Tabelle. Mögliche Werte `0.00%` `100.00%` sind `ABORTED` Prozentsätze von bis und die Nachricht. `ABORTED`bedeutet, dass die Weiterverteilung gestoppt wurde, ohne sie zu beenden, wobei der Grund in der `message` Spalte erklärt wird.
Nachricht	char(256)	Die Nachricht, die mit dem Weiterverteilungsfortschritt der Tabelle verknüpft ist.

Beispielabfrage

Die folgende Abfrage gibt laufende und in der Warteschlange stehende Abfragen zurück.

```
select * from svl_restore_alter_table_progress;

tbl     | progress |                           message
--------+----------+-------------------------------------------------------------
105614  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105610  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105594  | 0.00%    | Table waiting for alter diststyle conversion.
105602  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105606  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105598  | 100.00%  | Restored to distkey successfully.
```

SVL_S3LIST

Verwenden Sie die Ansicht SVL_S3LIST, um Details zu Amazon-Redshift-Spectrum-Abfragen auf Segment-Ebene zu erhalten.

SVL_S3LIST ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> SVL_S3LIST enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten.
Knoten	integer	Die Knotennummer.
slice	integer	Der Daten-Slice, für den ein bestimmtes Segment ausgeführt wurde.
eventtime	Zeitstempel	Der Zeitpunkt nach UTC, an dem das Ereignis aufgezeichnet wurde.
bucket	Text	Der Name des Amazon S3-Buckets.
prefix	Text	Das Präfix des Amazon-S3-Bucket-Speicherortes.
recursive	char(1)	Ob ein rekursiver Scan für Unterordner vorhanden ist.
retrieved_files	integer	Die Anzahl der aufgelisteten Dateien.

Spaltenname	Datentyp	Beschreibung
max_file_size	bigint	Die maximale Dateigröße unter aufgelisteten Dateien.
avg_file_size	double precision	Die durchschnittliche Dateigröße unter aufgelisteten Dateien.
generated_splits	integer	Die Anzahl der Dateiaufteilungen.
avg_split_length	double precision	Die durchschnittliche Länge von Dateiaufteilungen in Byte.
duration	bigint	Die Zeitdauer der Dateiauflistung in Mikrosekunden.

Beispielabfrage

Im folgenden Beispiel wird SVL_S3LIST nach der letzten ausgeführten Abfrage abgefragt.

```
select *
from svl_s3list
where query = pg_last_query_id()
order by query,segment;
```

SVL_S3LOG

Verwenden Sie die Ansicht SVL_S3LOG, um Details zu Amazon-Redshift-Spectrum-Abfragen auf Segment- und Knotenslice-Ebene zu erhalten.

SVL_S3LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> SVL_S3LOG enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht

enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
pid	integer	Die Prozess-ID.
query	integer	Die Abfrage-ID.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
Schritt	integer	Der Abfrage-Schritt, der ausgeführt wurde.
Knoten	integer	Die Knotennummer.
slice	integer	Der Daten-Slice, für den ein bestimmtes Segment ausgeführt wurde.
eventtime	Zeitstempel	Zeitpunkt nach UTC, an dem die Ausführung des Schrittes gestartet wurde.
Nachricht	Text	Meldung für den Protokolleintrag.

Beispielabfrage

Im folgenden Beispiel wird SVL_S3LOG nach der letzten ausgeführten Abfrage abgefragt.

```
select *
from svl_s3log
where query = pg_last_query_id()
```

```
order by query,segment,slice;
```

SVL_S3PARTITION

Verwenden Sie die Ansicht SVL_S3PARTITION um Details zu Amazon-Redshift-Spectrum-Partitionen auf Segment- und Knotenslice-Ebene zu erhalten.

SVL_S3PARTITION ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> SVL_S3PARTITION enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID.
segment	integer	Eine Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
Knoten	integer	Die Knotennummer.
slice	integer	Der Daten-Slice, für den ein bestimmtes Segment ausgeführt wurde.

Spaltenname	Datentyp	Beschreibung
starttime	Timestamp ohne Zeitzone	Zeitpunkt nach UTC, an dem die Partitionsbereinigung gestartet wurde.
endtime	Timestamp ohne Zeitzone	Zeitpunkt nach UTC, an dem die Partitionsbereinigung beendet wurde.
duration	bigint	Verstrichene Zeit (in Mikrosekunden)
total_partitions	integer	Gesamtanzahl der Partitionen
qualified_partitions	integer	Anzahl der qualifizierten Partitionen.
assigned_partitions	integer	Anzahl der zugewiesenen Partitionen auf dem Slice.
assignment	character	Art der Zuweisung.

Beispielabfrage

Das folgende Beispiel ruft die Partitionsdetails für die letzte ausgeführte Abfrage ab.

```
SELECT query, segment,
       MIN(starttime) AS starttime,
       MAX(endtime) AS endtime,
       datediff(ms,MIN(starttime),MAX(endtime)) AS dur_ms,
       MAX(total_partitions) AS total_partitions,
       MAX(qualified_partitions) AS qualified_partitions,
       MAX(assignment) as assignment_type
FROM svl_s3partition
WHERE query=pg_last_query_id()
GROUP BY query, segment
```

```
query | segment |           starttime           |           endtime           | dur_ms|
 total_partitions | qualified_partitions | assignment_type
------+---------+-------------------------------+-----------------------------+-------
+-----------------+----------------------+---------------
99232 |       0 | 2018-04-17 22:43:50.201515    | 2018-04-17 22:43:54.674595  |  4473 |
       2526       |        334           | p
```

SVL_S3PARTITION_SUMMARY

Verwenden Sie die Ansicht SVL_S3PARTITION_SUMMARY, um eine Zusammenfassung der Redshift-Spectrum-Abfragepartition zu erhalten, die auf Segment-Ebene ausgeführt wird.

SVL_S3PARTITION_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Informationen zu SVCS_S3PARTITION finden Sie unter SVCS_S3PARTITION_SUMMARY.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID. Mit diesem Wert können Sie verschiedene andere Systemtabellen und Ansichten verbinden.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten.
assignment	char(1)	Der Typ von Partitionszuweisung über Knoten hinweg.
min_starttime	timestamp	Der Zeitpunkt nach UTC, an dem die Partitionsverarbeitung gestartet wurde.
max_endtime	timestamp	Der Zeitpunkt nach UTC, an dem die Partitionsverarbeitung abgeschlossen wurde.
min_duration	bigint	Die von einem Knoten für diese Abfrage verwendete minimale Partitionsverarbeitungszeit (in Mikrosekunden).

Spaltenname	Datentyp	Beschreibung
max_duration	bigint	Die von einem Knoten für diese Abfrage verwendete maximale Partitionsverarbeitungszeit (in Mikrosekunden).
avg_duration	bigint	Die von einem Knoten für diese Abfrage verwendete durchschnittliche Partitionsverarbeitungszeit (in Mikrosekunden).
total_partitions	integer	Die Gesamtzahl von Partitionen in einer externen Tabelle.
qualified_partitions	integer	Die Gesamtzahl der qualifizierten Partitionen.
min_assigned_partitions	integer	Die einem Knoten zugewiesene minimale Anzahl von Partitionen.
max_assigned_partitions	integer	Die einem Knoten zugewiesene maximale Anzahl von Partitionen.
avg_assigned_partitions	bigint	Die einem Knoten zugewiesene durchschnittliche Anzahl von Partitionen.

Beispielabfrage

Das folgende Beispiel ruft die Partitions-Scandetails für die letzte ausgeführte Abfrage ab.

```
select query, segment, assignment, min_starttime, max_endtime, min_duration,
 avg_duration
from svl_s3partition_summary
where query = pg_last_query_id()
order by query,segment;
```

SVL_S3QUERY

Verwenden Sie die Ansicht SVL_S3QUERY, um Details zu Amazon-Redshift-Spectrum-Abfragen auf Segment- und Knotenslice-Ebene zu erhalten.

SVL_S3QUERY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

> **Note**
>
> SVL_S3QUERY enthält nur Abfragen, die auf Haupt-Clustern ausgeführt werden. Abfragen, die auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, sind nicht enthalten. Um auf Abfragen zuzugreifen, die sowohl auf Haupt-Clustern als auch auf Nebenläufigkeitsskalierungs-Clustern ausgeführt werden, empfehlen wir, die SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_DETAIL zu verwenden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der einen bestimmten Eintrag generiert hat.
query	integer	Die Abfrage-ID.
segment	integer	Eine Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
Schritt	integer	Der Abfrage-Schritt, der ausgeführt wurde.
Knoten	integer	Die Knotennummer.

Spaltenname	Datentyp	Beschreibung
slice	integer	Der Daten-Slice, für den ein bestimmtes Segment ausgeführt wurde.
starttime	timestamp	Zeitpunkt nach UTC, an dem die Ausführung der Abfrage gestartet wurde.
endtime	timestamp	Zeitpunkt nach UTC, an dem die Ausführung der Abfrage abgeschlossen wurde.
elapsed	integer	Verstrichene Zeit (in Mikrosekunden)
external_table_name	char(136)	Internes Format des Namens der externen Tabelle für den s3-scan-Schritt.
is_partitioned	char(1)	Wenn „true“ (`t`) zeigt dieser Spaltenwert an, dass die externe Tabelle partitioniert ist.
is_rrscan	char(1)	Wenn „true“ (`t`) zeigt dieser Spaltenwert an, dass ein Scan mit Bereichseinschränkung durchgeführt wurde.
s3_scanned_rows	bigint	Die Anzahl der von Amazon S3 gescannten und an die Redshift-Spectrum-Ebene gesendeten Zeilen.
s3_scanned_bytes	bigint	Die Anzahl der von Amazon S3 gescannten und an die Redshift-Spectrum-Ebene gesendeten Bytes.
s3query_returned_rows	bigint	Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Zeilen.
s3query_returned_bytes	bigint	Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Bytes.

Spaltenname	Datentyp	Beschreibung
files	integer	Die Anzahl der Dateien, die für diesen S3-Scan-Schritt auf diesem Slice verarbeitet wurden.
splits	int	Die Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden. Bei großen aufteilbaren Datendateien versucht Redshift Spectrum beispielsweise bei Dateigrößen über 512 MB, die Dateien zur parallelen Verarbeitung auf mehrere S3-Anforderungen aufzuteilen.
total_split_size	bigint	Die Gesamtgröße aller Aufteilungen, die in diesem Slice verarbeitet wurden (in Bytes)
max_split_size	bigint	Die maximale Aufteilungsgröße, die in diesem Slice verarbeitet wurden (in Bytes)
total_retries	integer	Die Gesamtzahl der erneuten Versuche für die verarbeiteten Dateien.
max_retries	integer	Die maximale Zahl erneuter Versuche für eine einzelne verarbeitete Datei.
max_request_duration	integer	Die maximale Dauer einer einzelnen Redshift Spectrum-Anfrage (in Mikrosekunden)
avg_request_duration	double precision	Die durchschnittliche Dauer der Redshift Spectrum-Anfragen (in Mikrosekunden)
max_request_parallelism	integer	Die maximale Anzahl ausstehender Redshift Spectrum-Anfragen auf diesem Slice für diesen S3-Scan-Schritt.
avg_request_parallelism	double precision	Die durchschnittliche Anzahl paralleler Redshift Spectrum-Anfragen auf diesem Slice für diesen S3-Scan-Schritt.

Beispielabfrage

Das folgende Beispiel ruft die Scanschritt-Details für die letzte ausgeführte Abfrage ab.

```
select query, segment, slice, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svl_s3query
where query = pg_last_query_id()
order by query,segment,slice;
```

```
query | segment | slice | elapsed | s3_scanned_rows | s3_scanned_bytes |
 s3query_returned_rows | s3query_returned_bytes | files
------+---------+-------+---------+-----------------+------------------
+----------------------+------------------------+------
 4587 |       2 |     0 |   67811 |               0 |                0 |
      0 |                      0 |     0
 4587 |       2 |     1 |  591568 |          172462 |         11260097 |
   8513 |                  170260 |     1
 4587 |       2 |     2 |  216849 |               0 |                0 |
      0 |                      0 |     0
 4587 |       2 |     3 |  216671 |               0 |                0 |
      0 |                      0 |     0
```

SVL_S3QUERY_SUMMARY

Verwenden Sie die Ansicht SVL_S3QUERY_SUMMARY zum Erhalt einer Zusammenfassung aller Amazon-Redshift-Spectrum-Abfragen (S3-Abfragen), die auf dem System ausgeführt wurden. SVL_S3QUERY_SUMMARY aggregiert Details aus SVL_S3QUERY auf Segmentebene.

SVL_S3QUERY_SUMMARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_DETAIL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Informationen zu SVCS_S3QUERY_SUMMARY finden Sie unter SVCS_S3QUERY_SUMMARY.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, der den jeweiligen Eintrag generiert hat.
query	integer	Die Abfrage-ID. Mit diesem Wert können Sie verschiedene andere Systemtabellen und Ansichten verbinden.
xid	bigint	Die Transaktions-ID.
pid	integer	Die Prozess-ID.
segment	integer	Die Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten.
Schritt	integer	Der Abfrage-Schritt, der ausgeführt wurde.
starttime	timestamp	Zeitpunkt nach UTC, an dem die Ausführung der Abfrage gestartet wurde.
endtime	timestamp	Zeitpunkt nach UTC, an dem die Abfrage abgeschlossen wurde.
elapsed	integer	Die Dauer der Ausführung der Abfrage (in Mikrosekunden).
aborted	integer	Wenn eine Abfrage vom System oder einem Benutzer beendet wurde, enthält diese Spalte den Wert **1**. Wenn die Abfrage abgeschlossen wurde, enthält diese Spalte **0**.
external_table_name	char(136)	Das interne Format des Namens der externen Tabelle für den Scan einer externen Tabelle.
file_format	character(16)	Das Dateiformat der Daten einer externen Tabelle.

Spaltenname	Datentyp	Beschreibung
is_partitioned	char(1)	Wenn „true“ (**t**) zeigt dieser Spaltenwert an, dass die externe Tabelle partitioniert ist.
is_rrscan	char(1)	Wenn „true“ (**t**) zeigt dieser Spaltenwert an, dass ein Scan mit Bereichseinschränkung durchgeführt wurde.
is_nested	char(1)	Bei „true“ (**t**) gibt dieser Spaltenwert an, dass auf den verschachtelten Spaltendatentyp zugegriffen wird.
s3_scanned_rows	bigint	Die Anzahl der von Amazon S3 gescannten und an die Redshift-Spectrum-Ebene gesendeten Zeilen.
s3_scanned_bytes	bigint	Die Anzahl der von Amazon S3 gescannten und an die Redshift-Spectrum-Ebene gesendeten Bytes, basierend auf komprimierten Daten.
s3query_returned_rows	bigint	Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Zeilen.
s3query_returned_bytes	bigint	Die Anzahl der von der Redshift Spectrum-Ebene an den Cluster zurückgegebenen Bytes. Eine große Menge an Amazon Redshift zurückgegebener Daten kann sich auf die Systemleistung auswirken.
files	integer	Die Anzahl der Dateien, die für diese Redshift Spectrum-Abfrage verarbeitet wurden. Eine kleine Anzahl von Dateien schränkt die Vorteile der parallelen Verarbeitung ein.
files_max	integer	Die maximale Anzahl der auf einem Slice verarbeiteten Dateien.
files_avg	integer	Die durchschnittliche Anzahl der auf einem Slice verarbeiteten Dateien.

Spaltenname	Datentyp	Beschreibung
splits	int	Die Anzahl der Aufteilungen, die für dieses Segment verarbeitet wurden. Die Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden. Bei großen aufteilbaren Datendateien versucht Redshift Spectrum beispielsweise bei Dateigrößen über 512 MB, die Dateien zur parallelen Verarbeitung auf mehrere S3-Anforderungen aufzuteilen.
splits_max	int	Die maximale Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden
splits_avg	int	Die durchschnittliche Anzahl der Aufteilungen, die in diesem Slice verarbeitet wurden
total_split_size	bigint	Die Gesamtgröße aller verarbeiteten Aufteilungen
max_split_size	bigint	Die maximale verarbeitete Aufteilungsgröße (in Bytes)
avg_split_size	bigint	Die durchschnittliche verarbeitete Aufteilungsgröße (in Bytes)
total_retries	integer	Die Gesamtzahl erneuter Versuche für eine einzelne verarbeitete Datei.
max_retries	integer	Die maximale Zahl erneuter Versuche für die einzelnen verarbeiteten Dateien.
max_request_duration	integer	Die maximale Dauer einer einzelnen Dateianfrage (in Mikrosekunden). Länger dauernde Anfragen können Anzeichen für einen Engpass sein.
avg_request_duration	double precision	Die durchschnittliche Dauer der Dateianfragen (in Mikrosekunden)

Spaltenname	Datentyp	Beschreibung
max_request_parallelism	integer	Die maximale Anzahl paralleler Anforderungen auf einem Slice für diese Amazon-Redshift-Abfrage.
avg_request_parallelism	double precision	Die durchschnittliche Anzahl paralleler Anforderungen auf einem Slice für diese Amazon-Redshift-Abfrage.
total_slowdown_count	bigint	Die Gesamtzahl von Amazon-S3-Anforderungen mit einem Verlangsamungsfehler, der beim San einer externen Tabelle auftrat.
max_slowdown_count	integer	Die maximale Anzahl von Amazon-S3-Anforderungen mit einem Verlangsamungsfehler, der während des Sans einer externen Tabelle auf einem Slice auftrat.

Beispielabfrage

Das folgende Beispiel ruft die Scanschritt-Details für die letzte ausgeführte Abfrage ab.

```
select query, segment, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svl_s3query_summary
where query = pg_last_query_id()
order by query,segment;
```

```
query | segment | elapsed | s3_scanned_rows | s3_scanned_bytes | s3query_returned_rows
 | s3query_returned_bytes | files
------+---------+---------+-----------------+------------------+----------------------
+------------------------+------
 4587 |       2 |   67811 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  591568 |          172462 |         11260097 |                 8513
 |                 170260 |     1
 4587 |       2 |  216849 |               0 |                0 |                    0
 |                      0 |     0
```

```
 4587 |       2 |  216671 |               0 |                0 |                     0
 |                     0 |     0
```

SVL_S3RETRIES

In der Ansicht SVL_S3RETRIES erhalten Sie Informationen dazu, warum eine Amazon-Redshift-Spectrum-Abfrage auf der Basis von Amazon S3 nicht erfolgreich war.

SVL_S3RETRIES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die Abfrage-ID.
segment	integer	Segmentnummer. Eine Abfrage besteht aus mehreren Segmenten, und jedes Segment besteht aus einem oder mehreren Schritten. Abfragesegmente können parallel ausgeführt werden. Jedes Segment wird in einem einzelnen Prozess ausgeführt.
Knoten	integer	Die Knotennummer.
slice	integer	Der Daten-Slice, für den ein bestimmtes Segment ausgeführt wurde.

Spaltenname	Datentyp	Beschreibung		
eventtime	Timestamp ohne Zeitzone	Zeitpunkt nach UTC, an dem die Ausführung des Schrittes gestartet wurde.		
retries	integer	Die Anzahl der Wiederholungen für die Abfrage.		
successful_fetches	integer	Die Häufigkeit, mit der Daten zurückgegeben wurden.		
file_size	bigint	Die Größe der Datei in Bytes.		
location	Text	Der Speicherort der Tabelle.		
Nachricht	Text	Die Fehlermeldung.		

Beispielabfrage

Das folgende Beispiel ruft Daten zu erfolglosen S3-Abfragen ab.

```
SELECT svl_s3retries.query, svl_s3retries.segment, svl_s3retries.node,
 svl_s3retries.slice, svl_s3retries.eventtime, svl_s3retries.retries,
svl_s3retries.successful_fetches, svl_s3retries.file_size,
 btrim((svl_s3retries."location")::text) AS "location",
 btrim((svl_s3retries.message)::text)
AS message FROM svl_s3retries;
```

SVL_SPATIAL_SIMPLIFY

Sie können die Systemansicht SVL_SPATIAL_SIMPLIFY mit dem Befehl COPY abfragen, um Informationen zu vereinfachten räumlichen Geometrieobjekten zu erhalten. Wenn Sie COPY für ein Shapefile verwenden, können Sie zur Erfassung die Optionen SIMPLIFY `tolerance`, SIMPLIFY AUTO und SIMPLIFY AUTO `max_tolerance` verwenden. Das Ergebnis der Vereinfachung wird in der Systemansicht SVL_SPATIAL_SIMPLIFY zusammengefasst.

Wenn SIMPLIFY AUTO `max_tolerance` festgelegt ist, enthält diese Ansicht eine Zeile für jede Geometrie, die die maximale Größe überschritten hat. Wenn SIMPLIFY `tolerance` festgelegt ist, wird eine Zeile für den gesamten COPY-Vorgang gespeichert. Diese Zeile verweist auf die Abfrage-ID von COPY und die angegebene Vereinfachungstoleranz.

SVL_SPATIAL_SIMPLIFY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_SPATIAL_SIMPLIFY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	integer	Die ID der Abfrage (Befehl COPY), die diese Zeile generiert hat.
line_number	integer	Wenn COPY `SIMPLIFY AUTO` angegeben ist, ist dieser Wert die Datensatznummer des vereinfachten Datensatzes im Shapefile.
maximum_tolerance	double	Der im Befehl COPY angegebene Abstandstoleranzwert. Dies ist entweder der maximale Toleranzwert mit der Option `SIMPLIFY AUTO` oder der feste Toleranzwert mit der Option `SIMPLIFY`.

Spaltenname	Datentyp	Beschreibung
initial_size	integer	Die Größe des GEOMETRY-Datenwerts in Byte vor der Vereinfachung.
simplified	char(1)	Wenn die Option COPY SIMPLIFY AUTO angegeben ist, t wenn die Geometrie erfolgreich vereinfacht wurde, ansonsten f. Die Geometrie wird möglicherweise nicht erfolgreich vereinfacht, wenn nach der Vereinfachung mit der angegebenen maximalen Toleranz ihre Größe immer noch die maximale Geometriegröße überschreitet.
final_size	integer	Wenn die Option COPY SIMPLIFY AUTO ist, ist dies die Größe der Geometrie nach der Vereinfachung in Byte.
final_tolerance	double	

Beispielabfrage

Die folgende Abfrage gibt die Liste der Datensätze zurück, die COPY vereinfacht hat.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+----------------------
    20 |     1184704 |                -1 |      1513736 | t          |    1008808 |
 1.276386653895e-05
    20 |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

SVL_SPECTRUM_SCAN_ERROR

Sie können die Systemansicht SVL_SPECTRUM_SCAN_ERROR abfragen, um Informationen zu Redshift-Spectrum-Scanfehlern zu erhalten.

SVL_SPECTRUM_SCAN_ERROR ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_EXTERNAL_QUERY_ERROR zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Zeigt ein Beispiel für protokollierte Fehler an. Der Standardwert ist 10 Einträge pro Abfrage.

Spaltenname	Datentyp	Beschreibung
userid	Ganzzahl	Die ID des Benutzers, der die Zeile generiert hat.
query	Ganzzahl	Die ID der Abfrage, die diese Zeile generiert hat.
location	character(128)	Der Speicherort der abgefragten Daten.
rowid	character(128)	Die Fehlerstelle innerhalb der Datei. Die `rowid`-Teile werden durch einen `:` (Doppelpunkt) getrennt und weitere Teile können zu einem späteren Zeitpunkt hinzugefügt werden. `row_offset :row_group :row_id` Ein row_offset ist der Offset (in Byte) der Zeile innerhalb der Datei. Er wird für nicht unterstützte Dateiformate auf `-1` festgelegt. Eine Tabelle ist in row_groups unterteilt. Jede Gruppe verfügt über Zeilen mit unterschiedlichen row_ids.
colname	character(128)	Der Name der Spalte, die von der Abfrage zurückgegeben wird.
original_value	character(128)	Ursprünglicher Wert wurde abgefragt.

Spaltenname	Datentyp	Beschreibung
modified_value	character(128)	Der geänderte Wert wurde basierend auf der in der Abfrage angegebenen Konfigurationsoption für die Datenverarbeitung zurückgegeben.
trigger	character(128)	In der Abfrage angegebene Datenverarbeitungsoption.
action	character(128)	Aktion, die mit der in der Abfrage angegebenen Datenverarbeitungsoption verknüpft ist.
action_value	character(128)	Wert des Aktionsparameters, der mit der in der Abfrage angegebenen Datenverarbeitungsoption verbunden ist.
error_code	Ganzzahl	Ergebniscode der in der Abfrage angegebenen Datenverarbeitungsoption.

Beispielabfrage

Die folgende Abfrage gibt die Liste der Zeilen zurück, für die Datenbehandlungsvorgänge ausgeführt wurden.

```
SELECT * FROM svl_spectrum_scan_error;
```

Die Abfrage gibt Ergebnisse wie die folgenden zurück.

```
  userid  query     location                                        rowid   colname
         original_value             modified_value       trigger          action
                 action_valueerror_code
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                     156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0     league_nspi
          34595                      32767                UNSPECIFIED
OVERFLOW_VALUE                                  199
```

```
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:1     league_nspi
          34151                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2     league_nspi
          33223                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3     league_nspi
          32808                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:4     league_nspi
          32790                      32767               UNSPECIFIED
OVERFLOW_VALUE                                  199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:5     league_name
         Spanish Primera Division   Spanish Primera Divi UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:6     league_name
         Spanish Primera Division   Spanish Primera Divi UNSPECIFIED      TRUNCATE
                                    156
```

SVL_STATEMENTTEXT

Verwenden Sie die Ansicht SVL_STATEMENTTEXT, um einen vollständigen Satz aller SQL-Befehle, die auf dem System ausgeführt wurden, zu erhalten.

Die Ansicht SVL_STATEMENTTEXT enthält die Vereinigung aller Zeilen in den Tabellen STL_DDLTEXT, STL_QUERYTEXT und STL_UTILITYTEXT. Diese Ansicht enthält auch eine Verbindung mit der Tabelle STL_QUERY.

SVL_STATEMENTTEXT ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert,

dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	ID des Benutzers, der den Eintrag generiert hat.
xid	bigint	Mit der Anweisung verbundene Transaktions-ID.
pid	integer	Prozess-ID für die Anweisung.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist dieses Feld leer.
starttime	timestamp	Genaue Uhrzeit in UTC, zu der die Ausführung der Anweisung begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile. Beispiel: **2009-06-12 11:29:19.131358**
endtime	timestamp	Genaue Uhrzeit in UTC, zu der die Ausführung der Anweisung abgeschlossen wurde, mit 6 Nachkommastellen für Sekundenbruchteile. Beispiel: **2009-06-12 11:29:19.193640**
sequence	integer	Wenn eine einzelne Anweisung mehr als 200 Zeichen enthält, werden weitere Zeilen für diese Anweisung protokolliert. Sequenz 0 ist die erste Zeile, 1 die zweite usw.
type	varchar(10)	Typ der SQL-Anweisung: **QUERY**, **DDL** oder **UTILITY**.

Spaltenname	Datentyp	Beschreibung
Text	character(200)	SQL-Text, in Schritten von je 200 Zeichen. Diese Feld kann Sonderzeichen wie Backslash (\\) und Zeilenumbruch (\n) enthalten.

Beispielabfrage

Die folgende Abfrage gibt am 16. Juni 2009 ausgeführte DDL-Anweisungen aus:

```
select starttime, type, rtrim(text) from svl_statementtext
where starttime like '2009-06-16%' and type='DDL' order by starttime asc;

starttime                  | type |              rtrim
---------------------------|------|--------------------------------
2009-06-16 10:36:50.625097 | DDL  | create table ddltest(c1 int);
2009-06-16 15:02:16.006341 | DDL  | drop view alltickitjoin;
2009-06-16 15:02:23.65285  | DDL  | drop table sales;
2009-06-16 15:02:24.548928 | DDL  | drop table listing;
2009-06-16 15:02:25.536655 | DDL  | drop table event;
...
```

Wiederherstellen von gespeichertem SQL

Um das in der Spalte `text` von SVL_STATEMENTTEXT gespeicherte SQL wiederherzustellen, führen Sie eine SELECT-Anweisung aus, um SQL von mindestens einem Teil der Spalte `text` zu erstellen. Bevor Sie das wiederhergestellte SQL ausführen, ersetzen Sie alle Sonderzeichen (\n) durch einen Zeilenumbruch. Das Ergebnis der folgenden SELECT-Anweisung besteht aus Zeilen von wiederhergestelltem SQL im Feld `query_statement`.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS query_statement
from SVL_STATEMENTTEXT where pid=pg_backend_pid();
```

Mit der folgenden Abfrage werden beispielsweise 3 Spalten ausgewählt. Die Abfrage selbst ist länger als 200 Zeichen und wird in mehreren Teilen in SVL_STATEMENTTEXT gespeichert.

```
select
1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
```

```
2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,
3 AS b01234567890123456789012345678901234567890123456789012345678901234
FROM stl_querytext;
```

In diesem Beispiel wird die Abfrage in 2 Teilen (Zeilen) in der Spalte `text` von SVL_STATEMENTTEXT gespeichert.

```
select sequence, text from SVL_STATEMENTTEXT where pid = pg_backend_pid() order by
 starttime, sequence;
```

```
 sequence |
                text

----------
+-----------------------------------------------------------------------------------------------------
        0 | select\n1 AS
 a01234567890123456789012345678901234567890123456789012345678901234567890,\n2 AS
 b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b01234567890123456789012345678901234567890123456789012345678901234
        1 | \nFROM stl_querytext;
```

Um das in SVL_STATEMENTTEXT gespeicherte SQL wiederherzustellen, führen Sie den folgenden SQL-Code aus.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS text
from SVL_STATEMENTTEXT where pid=pg_backend_pid();
```

Um das resultierende wiederhergestellte SQL in Ihrem Client zu verwenden, ersetzen Sie alle Sonderzeichen (\n) durch einen Zeilenumbruch.

```
                     text

-----------------------------------------------------------------------------------------------------
 select\n1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
\n2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b01234567890123456789012345678901234567890123456789012345678901234\nFROM stl_querytext;
```

SVL_STORED_PROC_CALL

Sie können die Systemansicht SVL_STORED_PROC_CALL abfragen, um Informationen über Aufrufe einer gespeicherten Prozedur zu erhalten, einschließlich Start- und Endzeit sowie Informationen dazu, ob ein Aufruf abgebrochen wurde. Jeder Aufruf einer gespeicherten Prozedur erhält eine Abfrage-ID.

SVL_STORED_PROC_CALL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_PROCEDURE_CALL zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, dessen Berechtigungen zum Ausführen der Anweisung verwendet wurden. Wenn dieser Aufruf innerhalb einer gespeicherten SECURITY DEFINER-Prozedur eingebunden war, dann ist dies die Benutzer-ID des Besitzers dieser gespeicherten Prozedur.
session_userid	integer	Die ID des Benutzers, der die Sitzung erstellt hat und der der Aufrufer des Aufrufs der gespeicherten Prozedur auf oberster Ebene ist.
query	integer	Die Abfrage-ID des Prozeduraufrufs.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist der Wert in diesem Feld der Standardwert.

Spaltenname	Datentyp	Beschreibung
xid	bigint	Die Transaktions-ID.
pid	integer	Die Prozess-ID. Normalerweise werden alle Abfragen in einer Sitzung in demselben Prozess ausgeführt; dieser Wert bleibt daher konstant, wenn Sie eine Reihe von Abfragen in derselben Sitzung ausführen. Im Anschluss an bestimmte interne Ereignisse startet Amazon Redshift möglicherweise eine aktive Sitzung neu und weist einen neuen PID-Wert zu. Weitere Informationen finden Sie unter STL_RESTARTED_SESSIONS.
Datenbank	character(32)	Der Name der Datenbank, mit der der Benutzer verbunden war, als die Abfrage ausgegeben wurde.
querytxt	character(4000)	Der tatsächliche Text der Prozeduraufrufsabfrage.
starttime	timestamp	Uhrzeit in UTC, zu der die Ausführung der Abfrage begonnen wurde, mit 6 Nachkommastellen für Sekundenbruchteile, z. B.: **`2009-06-12 11:29:19.131358.`**
endtime	timestamp	Uhrzeit in UTC, zu der die Ausführung der Abfrage beendet wurde, mit 6 Nachkommastellen für Sekundenbruchteile, z. B.: **`2009-06-12 11:29:19.131358.`**
aborted	integer	Wenn eine gespeicherte Prozedur vom System beendet oder von einem Benutzer abgebrochen wurde, enthält diese Spalte den Wert 1. Wenn der Aufruf abgeschlossen wird, enthält diese Spalte den Wert 0.
from_sp_call	integer	Wenn die Prozedur von einem anderen Prozeduraufruf aufgerufen wurde, enthält diese Spalte die Abfrage-ID des externen Aufrufs. Andernfalls ist das Feld NULL.

Beispielabfrage

Die folgende Abfrage gibt die verstrichene Zeit in absteigender Reihenfolge und den Abschlussstatus für Aufrufe gespeicherter Prozeduren des vergangenen Tages zurück.

```
select query, datediff(seconds, starttime, endtime) as elapsed_time, aborted,
 trim(querytxt) as call from svl_stored_proc_call where starttime >= getdate() -
 interval '1 day' order by 2 desc;

  query | elapsed_time | aborted |                                          call
--------+--------------+---------
+-------------------------------------------------------------------------------
   4166 |            7 |       0 | call search_batch_status(35,'succeeded');
   2433 |            3 |       0 | call test_batch (123456)
   1810 |            1 |       0 | call prod_benchmark (123456)
   1836 |            1 |       0 | call prod_testing (123456)
   1808 |            1 |       0 | call prod_portfolio ('N', 123456)
   1816 |            1 |       1 | call prod_portfolio ('Y', 123456)
```

SVL_STORED_PROC_MESSAGES

Sie können die Systemansicht SVL_STORED_PROC_MESSAGES abfragen, um Informationen über Meldungen zu gespeicherten Prozeduren abzurufen. Ausgegebene Meldungen werden protokolliert, auch wenn der Aufruf der gespeicherten Prozedur abgebrochen wird. Jeder Aufruf einer gespeicherten Prozedur erhält eine Abfrage-ID. Weitere Informationen zum Festlegen der Mindestebene für protokollierte Meldungen finden Sie unter stored_proc_log_min_messages.

SVL_STORED_PROC_MESSAGES ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_PROCEDURE_MESSAGES zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
userid	integer	Die ID des Benutzers, dessen Berechtigungen zum Ausführen der Anweisung verwendet wurden. Wenn dieser Aufruf innerhalb einer gespeicherten SECURITY DEFINER-Prozedur eingebunden war, dann ist dies die Benutzer-ID des Besitzers dieser gespeicherten Prozedur.
session_userid	integer	Die ID des Benutzers, der die Sitzung erstellt hat und der der Aufrufer des Aufrufs der gespeicherten Prozedur auf oberster Ebene ist.
pid	integer	Die Prozess-ID.
xid	bigint	Die Transaktions-ID der Prozeduraufrufabfrage.
query	integer	Die Abfrage-ID des Prozeduraufrufs.
recordtime	timestamp	Die Uhrzeit in UTC, zu der die Meldung ausgegeben wurde.
loglevel	integer	Der numerische Wert der Protokollebene der ausgegebenen Meldung. Mögliche Werte: 20 – für LOG 30 – für INFO 40 – für NOTICE 50 – für WARNING 60 – für EXCEPTION
loglevel_text	character(10)	Die Protokollebene, die dem numerischen Wert in LogLevel entspricht. Mögliche Werte: LOG, INFO, NOTICE, WARNING und EXCEPTION.
Nachricht	character(1024)	Der Text der ausgegebenen Meldung.
linenum	integer	Die Zeilennummer der ausgegebenen Meldung.
querytext	Zeichen (500)	Der tatsächliche Text der Prozeduraufrufsabfrage.
label	Zeichen (320)	Entweder der Name der für die Ausführung verwendeten Datei oder eine mit dem Befehl SET QUERY_GROUP

Spaltenname	Datentyp	Beschreibung
		definierte Beschriftung. Wenn die Tabelle nicht dateibasiert ist oder der Parameter QUERY_GROUP nicht eingerichtet ist, ist der Wert in diesem Feld der Standardwert.
aborted	integer	Wenn eine gespeicherte Prozedur vom System beendet oder von einem Benutzer abgebrochen wurde, enthält diese Spalte den Wert 1. Wenn der Aufruf abgeschlossen wird, enthält diese Spalte den Wert 0.
message_xid	bigint	Die Transaktions-ID der ausgelösten Nachricht.

Beispielabfrage

Die folgenden SQL-Anweisungen zeigen, wie ausgegebene Meldungen mithilfe von SVL_STORED_PROC_MESSAGES überprüft werden können.

```
-- Create and run a stored procedure
CREATE OR REPLACE PROCEDURE test_proc1(f1 int) AS
$$
BEGIN
    RAISE INFO 'Log Level: Input f1 is %',f1;
    RAISE NOTICE 'Notice Level: Input f1 is %',f1;
    EXECUTE 'select invalid';
    RAISE NOTICE 'Should not print this';

EXCEPTION WHEN OTHERS THEN
     raise exception 'EXCEPTION level: Exception Handling';
END;
$$ LANGUAGE plpgsql;

-- Call this stored procedure
CALL test_proc1(2);

-- Show raised messages with level higher than INFO
SELECT query, recordtime, loglevel, loglevel_text, trim(message) as message, aborted
 FROM svl_stored_proc_messages
  WHERE loglevel > 30 AND query = 193 ORDER BY recordtime;
```

```
 query |         recordtime         | loglevel | loglevel_text |               message
              | aborted
-------+----------------------------+----------+---------------
+-------------------------------------+---------
   193 | 2020-03-17 23:57:18.277196 |       40 | NOTICE        | Notice Level: Input f1
 is 2         |       1
   193 | 2020-03-17 23:57:18.277987 |       60 | EXCEPTION     | EXCEPTION level:
 Exception Handling |       1
(2 rows)

-- Show raised messages at EXCEPTION level
SELECT query, recordtime, loglevel, loglevel_text, trim(message) as message, aborted
 FROM svl_stored_proc_messages
  WHERE loglevel_text = 'EXCEPTION' AND query = 193 ORDER BY recordtime;

 query |         recordtime         | loglevel | loglevel_text |               message
              | aborted
-------+----------------------------+----------+---------------
+-------------------------------------+---------
   193 | 2020-03-17 23:57:18.277987 |       60 | EXCEPTION     | EXCEPTION level:
 Exception Handling |       1
```

Die folgenden SQL-Anweisungen zeigen, wie ausgegebene Meldungen mit der SET-Option mithilfe von SVL_STORED_PROC_MESSAGES überprüft werden können, wenn eine gespeicherte Prozedur erstellt wird. Da test_proc() die Mindestprotokollebene NOTICE aufweist, werden nur Meldungen auf der Ebene NOTICE, WARNING und EXCEPTION in SVL_STORED_PROC_MESSAGES protokolliert.

```
-- Create a stored procedure with minimum log level of NOTICE
CREATE OR REPLACE PROCEDURE test_proc() AS
$$
BEGIN
    RAISE LOG 'Raise LOG messages';
    RAISE INFO 'Raise INFO messages';
    RAISE NOTICE 'Raise NOTICE messages';
    RAISE WARNING 'Raise WARNING messages';
    RAISE EXCEPTION 'Raise EXCEPTION messages';
    RAISE WARNING 'Raise WARNING messages again'; -- not reachable
END;
$$ LANGUAGE plpgsql SET stored_proc_log_min_messages = NOTICE;

-- Call this stored procedure
```

```
CALL test_proc();

-- Show the raised messages
SELECT query, recordtime, loglevel_text, trim(message) as message, aborted FROM
 svl_stored_proc_messages
  WHERE query = 149 ORDER BY recordtime;

 query |         recordtime         | loglevel_text |          message         |
 aborted
-------+----------------------------+---------------+--------------------------
+---------
   149 | 2020-03-16 21:51:54.847627 | NOTICE        | Raise NOTICE messages    |
 1
   149 | 2020-03-16 21:51:54.84766  | WARNING       | Raise WARNING messages   |
 1
   149 | 2020-03-16 21:51:54.847668 | EXCEPTION     | Raise EXCEPTION messages |
 1
(3 rows)
```

SVL_TERMINATE

Zeichnet die Zeit auf, zu der ein Benutzer einen Prozess abbricht oder beendet.

SELECT PG_TERMINATE_BACKEND(pid), SELECT PG_CANCEL_BACKEND(pid) und CANCEL pid erstellt einen Protokolleintrag in SVL_TERMINATE.

SVL_TERMINATE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_QUERY_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
pid	integer	Die Prozess-ID des abgebrochenen oder beendeten Prozesses.

Spaltenname	Datentyp	Beschreibung
eventtime	Zeitstempel	Der Zeitpunkt, zu dem der Prozess abgebrochen oder beendet wurde.
userid	integer	Die Benutzer-ID des Benutzers, der den Befehl ausführt.
type	string	Die Art der Beendigung. CANCEL oder TERMINATE.

Der folgende Befehl zeigt die zuletzt abgebrochene Abfrage.

```
select * from svl_terminate order by eventtime desc limit 1;
 pid  |         eventtime          | userid |  type
------+----------------------------+--------+--------
 8324 | 2020-03-24 09:42:07.298937 |      1 | CANCEL
(1 row)
```

SVL_UDF_LOG

Zeichnet systemdefinierte Fehlermeldungen und Warnungen auf, die während der Ausführung einer benutzerdefinierten Funktion (User-Defined Function, UDF) generiert wurden.

SVL_UDF_LOG ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_UDF_LOG zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
query	bigint	Die Abfrage-ID. Sie können diese ID verwenden, um

Spaltenname	Datentyp	Beschreibung
		verschiedene andere Systemtabellen und Anzeigen anzufügen.
Nachricht	char(4096)	Die von der Funktion generiert e Meldung.
created	timestamp	Der Zeitpunkt der Erstellung des Protokolls.
traceback	char(4096)	Falls verfügbar, bietet dieser Wert einen Stack-Tra ceback für die benutzerd efinierte Funktion. Weitere Informationen finden Sie unter traceback in der Python-St andardbibliothek.
funcname	character(256)	Der Name der ausgeführten benutzerdefinierten Funktion.
Knoten	integer	Der Knoten, auf dem die Meldung generiert wurde.
slice	integer	Der Slice, auf dem die Meldung generiert wurde.
seq	integer	Die Sequenz der Meldung auf dem Slice.

Beispielabfragen

Das folgende Beispiel zeigt, wie benutzerdefinierte Funktionen mit systemdefinierten Fehlern umgehen. Der erste Block zeigt die Definition für eine benutzerdefinierte Funktion, die die Umkehrung eines Arguments ausgibt. Wenn Sie die Funktion ausgeben und 0 als Argument angeben, wie der zweite Block zeigt, gibt die Funktion einen Fehler aus. Die dritte Anweisung liest die in SVL_UDF_LOG protokollierte Fehlermeldung.

```
-- Create a function to find the inverse of a number

CREATE OR REPLACE FUNCTION  f_udf_inv(a int)
  RETURNS float IMMUTABLE
AS $$
   return 1/a
$$ LANGUAGE plpythonu;

-- Run the function with a 0 argument to create an error
Select f_udf_inv(0) from sales;

-- Query SVL_UDF_LOG to view the message

Select query, created, message::varchar
from svl_udf_log;

 query |          created           | message
-------+----------------------------
+-----------------------------------------------------------
  2211 | 2015-08-22 00:11:12.04819  | ZeroDivisionError: long division or modulo by
 zero\nNone
```

Das folgende Beispiel fügt der benutzerdefinierten Funktion Protokollierung und eine Warnmeldung hinzu, so dass eine Division durch Null zu einer Warnmeldung und nicht zum Anhalten mit einer Fehlermeldung führt.

```
-- Create a function to find the inverse of a number and log a warning

CREATE OR REPLACE FUNCTION f_udf_inv_log(a int)
  RETURNS float IMMUTABLE
 AS $$
  import logging
  logger = logging.getLogger() #get root logger
  if a==0:
    logger.warning('You attempted to divide by zero.\nReturning zero instead of error.
\n')
    return 0
  else:
     return 1/a
$$ LANGUAGE plpythonu;
```

Das folgende Beispiel führt die Funktion aus und fragt dann SVL_UDF_LOG zur Anzeige der Meldung ab.

```
-- Run the function with a 0 argument to trigger the warning
Select f_udf_inv_log(0) from sales;

-- Query SVL_UDF_LOG to view the message

Select query, created, message::varchar
from svl_udf_log;

query |          created           | message
------+----------------------------+---------------------------------
    0 | 2015-08-22 00:11:12.04819  | You attempted to divide by zero.
                                     Returning zero instead of error.
```

SVL_USER_INFO

In der Ansicht SVL_USER_INFO können Sie Daten zu Benutzern der Amazon-Redshift-Datenbank abrufen.

SVL_USER_INFO ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
usename	Text	Der Benutzernamen für die Rolle.
usesysid	integer	Die Benutzer-ID für den Benutzer.
usecreatedb	Boolean	Ein Wert, der angibt, ob der Benutzer zum Erstellen von Datenbanken berechtigt ist.
usesuper	Boolean	Ein Wert, der angibt, ob der Benutzer ein Superuser ist.
usecatupd	Boolean	Ein Wert, der angibt, ob der Benutzer Systemkataloge aktualisieren kann.

Spaltenname	Datentyp	Beschreibung
useconnlimit	Text	Die Anzahl der Verbindungen, die der Benutzer öffnen kann.
syslogaccess	Text	Ein Wert, der angibt, ob der Benutzer Zugriff auf die Systemprotokolle hat. Die beiden möglichen Werte sind `RESTRICTED` und `UNRESTRICTED` . `RESTRICTED` bedeutet, dass Benutzer, die keine Superuser sind, ihre eigenen Datensätze anzeigen können. `UNRESTRICTED` bedeutet, dass Benutzer, die keine Superuser sind, alle Datensätze in den Systemansichten und -tabellen anzeigen können, für die sie `SELECT`-Berechtigungen besitzen.
last_ddl_ts	timestamp	Der Zeitstempel für die letzte Data Definition Language (DDL)-Ers tellungsanweisung, die vom Benutzer ausgeführt wurde.
sessiontimeout	integer	Die maximale Zeit in Sekunden, die eine Sitzung ohne Timeout inaktiv oder untätig bleibt. 0 bedeutet, dass kein Timeout eingestellt ist. Informati onen zur Einstellung für den Leerlauf oder den Timeout bei Inaktivität des Clusters finden Sie unter Kontingente und Limits in Amazon Redshift im Amazon-Redshift-Verwaltungshandbuch.
external_id	text	Die eindeutige Kennung des Benutzers beim externen Identitätsanbieter.

Beispielabfragen

Mit dem folgenden Befehl werden Benutzerinformationen aus SVL_USER_INFO abgerufen.

```
SELECT * FROM SVL_USER_INFO;
```

SVL_VACUUM_PERCENTAGE

Die Ansicht VL_VACUUM_PERCENTAGE meldet den Prozentsatz von Datenblöcken, die nach einer Bereinigung einer Tabelle zugewiesen wurden. Dieser Prozentsatz zeigt an, wie viel Festplattenspeicherplatz zurückgewonnen wurde. Vgl. den Befehl VACUUM für weitere Informationen zum Bereinigungsutility.

SVL_VACUUM_PERCENTAGE ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Einige oder alle Daten in dieser Tabelle sind auch in der SYS-Überwachungsansicht SYS_VACUUM_HISTORY zu finden. Die Daten in der SYS-Überwachungsansicht sind so formatiert, dass sie leichter verwendbar und besser verständlich sind. Wir empfehlen Ihnen, für Ihre Abfragen die SYS-Überwachungsansicht zu verwenden.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
xid	bigint	Transaktions-ID für die Bereinigungsanweisung.
table_id	integer	Tabellen-ID für die bereinigte Tabelle.
percentage	bigint	Prozentsatz der Datenblöcke nach einer Bereinigung (relativ zur Anzahl der Blöcke in der Tabelle vor der Bereinigung).

Beispielabfrage

Die folgende Abfrage zeigt den Prozentsatz für eine bestimmte Operation auf Tabelle 100238 an:

```
select * from svl_vacuum_percentage
where table_id=100238 and xid=2200;

xid  | table_id | percentage
-----+----------+------------
1337 |   100238 |         60
(1 row)
```

Nach dieser Bereinigungsoperation enthält die Tabelle noch 60 Prozent der ursprünglichen Blöcke.

Systemkatalogtabellen

Themen

- PG_ATTRIBUTE_INFO

Die Systemkataloge speichern Schema-Metadaten, wie etwa Informationen zu Tabellen und Spalten. Systemkatalogtabellen haben das Präfix PG.

Die Standard-PostgreSQL-Katalogtabellen sind für Amazon-Redshift-Benutzer zugänglich. Weitere Informationen zu PostgreSQL-Systemkatalogen finden Sie unter PostgreSQL-Systemtabellen.

PG_ATTRIBUTE_INFO

PG_ATTRIBUTE_INFO ist eine Amazon-Redshift-Systemansicht, die auf der PostgreSQL-Katalogtabelle PG_ATTRIBUTE und der internen Katalogtabelle PG_ATTRIBUTE_ACL basiert. PG_ATTRIBUTE_INFO enthält Details zu Spalten einer Tabelle oder Ansicht, einschließlich Spaltenzugriffskontrolllisten, falls vorhanden.

Tabellenspalten

PG_ATTRIBUTE_INFO zeigt zusätzlich zu den Spalten in PG_ATTRIBUTE die folgende Spalte.

Spaltenname	Datentyp	Beschreibung
attacl	aclitem[]	Die Zugriffsberechtigungen auf Spaltenebene (falls vorhanden), die speziell für diese Spalte gewährt wurden.

PG_CLASS_INFO

PG_CLASS_INFO ist eine Amazon-Redshift-Systemansicht in den PostgreSQL-Katalogtabellen PG_CLASS und PG_CLASS_EXTENDED. PG_CLASS_INFO enthält Informationen zum Zeitpunkt der Tabellenerstellung und zum aktuellen Verteilungsstil. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.

PG_CLASS_INFO ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

PG_CLASS_INFO zeigt die folgenden Spalten zusätzlich zu den Spalten in PG_CLASS an. Die `oid`-Spalte in PG_CLASS hat in der Tabelle PG_CLASS_INFO den Namen `relid`.

Spaltenname	Datentyp	Beschreibung
relcreationtime	timestamp	Zeitpunkt in UTC, an dem die Tabelle erstellt wurde
releffectivediststyle	integer	Der Verteilungsstil einer Tabelle oder, wenn die Tabelle die automatische Verteilung verwendet, der aktuelle von Amazon-Redshift zugewiesene Verteilungsstil.

Die Spalte RELEFFECTIVEDISTSTYLE in PG_CLASS_INFO zeigt den Verteilungsstil für die Tabelle an. Wenn die Tabelle die automatische Verteilung verwendet, ist RELEFFECTIVEDISTSTYLE auf 10, 11 oder 12 festgelegt. Diese Werte bezeichnen die verwendeten Verteilungsstile AUTO (ALL), AUTO (EVEN) oder AUTO (KEY). Wenn die Tabelle die automatische Verteilung verwendet, ändert sich der Verteilungsstil von anfänglich AUTO (ALL) in AUTO (EVEN), wenn die Tabelle wächst, oder AUTO (KEY), wenn eine Spalte als Verteilungsschlüssel dienen kann.

In der folgenden Tabelle wird der Verteilungsstil für die einzelnen Werte in der Spalte RELEFFECTIVEDISTSTYLE angegeben:

RELEFFECTIVEDISTSTYLE	Aktueller Verteilungsstil
0	EVEN
1	KEY
8	ALL
10	AUTO (ALL)
11	AUTO (EVEN)
12	AUTO (KEY)

Beispiel

Die folgende Abfrage gibt den aktuellen Verteilungsstil von Tabellen im Katalog zurück.

```
select reloid as tableid,trim(nspname) as schemaname,trim(relname) as
 tablename,reldiststyle,releffectivediststyle,
CASE WHEN "reldiststyle" = 0 THEN 'EVEN'::text
     WHEN "reldiststyle" = 1 THEN 'KEY'::text
     WHEN "reldiststyle" = 8 THEN 'ALL'::text
     WHEN "releffectivediststyle" = 10 THEN 'AUTO(ALL)'::text
     WHEN "releffectivediststyle" = 11 THEN 'AUTO(EVEN)'::text
     WHEN "releffectivediststyle" = 12 THEN 'AUTO(KEY)'::text ELSE '<<UNKNOWN>>'::text
 END as diststyle,relcreationtime
from pg_class_info a left join pg_namespace b on a.relnamespace=b.oid;
```

```
 tableid | schemaname | tablename | reldiststyle | releffectivediststyle | diststyle  |
      relcreationtime
---------+------------+-----------+--------------+-----------------------+------------
+----------------------------
 3638033 | public     | customer  |            0 |                     0 | EVEN       |
 2019-06-13 15:02:50.666718
 3638037 | public     | sales     |            1 |                     1 | KEY        |
 2019-06-13 15:03:29.595007
 3638035 | public     | lineitem  |            8 |                     8 | ALL        |
 2019-06-13 15:03:01.378538
```

```
 3638039 | public     | product   |           9 |                    10 | AUTO(ALL)  |
 2019-06-13 15:03:42.691611
 3638041 | public     | shipping  |           9 |                    11 | AUTO(EVEN) |
 2019-06-13 15:03:53.69192
 3638043 | public     | support   |           9 |                    12 | AUTO(KEY)  |
 2019-06-13 15:03:59.120695
(6 rows)
```

PG_DATABASE_INFO

PG_DATABASE_INFO ist eine Amazon-Redshift-Systemansicht, die die PostgreSQL-Katalogtabelle PG_DATABASE erweitert.

PG_DATABASE_INFO ist für alle Benutzer sichtbar.

Tabellenspalten

PG_DATABASE_INFO enthält neben den Spalten aus PG_DATABASE die folgenden Spalten. Die `oid`-Spalte in PG_DATABASE hat in der Tabelle PG_DATABASE_INFO den Namen `datid`. Weitere Informationen finden Sie in der PostgreSQL-Dokumentation.

Spaltenname	Datentyp	Beschreibung
datid	OID	Die Objektkennung (OID), die intern von Systemtabellen verwendet wird.
datconnlimit	text	Die maximale Anzahl der gleichzeitigen Verbindungen zu einer Datenbank. Bei -1 ist die Menge gleichzeitiger Verbindungen unbegrenzt.

PG_DEFAULT_ACL

Speichert Informationen zu Standard-Zugriffsberechtigungen. Für weitere Informationen zu Standard-Zugriffsberechtigungen vgl. ALTER DEFAULT PRIVILEGES.

PG_DEFAULT_ACL ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
defacluser	integer	ID des Benutzers, für den die aufgeführten Berechtigungen angewendet werden.
defaclnamespace	OID	Die Objekt-ID des Schemas, in dem die Standardb erechtigungen angewendet werden. Wenn kein Schema angegeben ist, lautet der Standardwert 0.
defaclobjtype	character	Der Objekttyp, für den Standardberechtigungen angewendet werden. Gültige Werte sind: • r-relation (Tabelle oder Ansicht) • f–function • p–stored procedure
defaclacl	aclitem[]	Eine Zeichenfolge, die die Standardberechtigungen für den angegebenen Benutzer oder die Benutzergruppe und den Objekttyp definiert. Wenn die Berechtigungen einem Benutzer gewährt werden, hat die Zeichenfolge die folgende Form: *{ username=privilegestring/grantor }* username (Benutzername Der Name des Benutzers, dem die Berechtigungen gewährt werden. Wenn der Benutzername fehlt, werden die Berechtigungen PUBLIC gewährt. Wenn die Berechtigungen einer Benutzergruppe gewährt werden, hat die Zeichenfolge die folgende Form: *{ "group groupname=privilegestring/grantor" }*

Spaltenname	Datentyp	Beschreibung
		privilegestring Eine Zeichenfolge, die definiert, welche Berechtigungen gewährt werden. Folgende Werte sind zulässig: • r–SELECT (lesen) • a–INSERT (anhängen) • w–UPDATE (schreiben) • d–DELETE • x – Gewährt die Berechtigung zum Erstellen einer Fremdschlüsseleinschränkung (REFERENCES). • X–EXECUTE • * – Zeigt an, dass der Benutzer, der die vorige Berechtigung erhält, diese Berechtigung seinerseits anderen gewähren kann (WITH GRANT OPTION). grantor Der Name des Benutzers, der die Berechtigungen gewährt hat. Das folgende Beispiel zeigt an, dass der Benutzer `admin` dem Benutzer `dbuser` alle Berechtigungen, einschließlich WITH GRANT OPTION, gewährt hat. `dbuser=r*a*w*d*x*X*/admin`

Beispiel

Die folgende Abfrage gibt alle für die Datenbank definierten Standardberechtigungen zurück.

```
select pg_get_userbyid(d.defacluser) as user,
n.nspname as schema,
case d.defaclobjtype when 'r' then 'tables' when 'f' then 'functions' end
as object_type,
array_to_string(d.defaclacl, ' + ')  as default_privileges
from pg_catalog.pg_default_acl d
left join pg_catalog.pg_namespace n on n.oid = d.defaclnamespace;

 user  | schema | object_type |              default_privileges
-------+--------+-------------+-----------------------------------------------------
 admin | tickit | tables      | user1=r/admin + "group group1=a/admin" + user2=w/admin
```

Das Ergebnis im vorherigen Beispiel zeigt, dass für alle vom Benutzer `admin` im Schema `tickit` erstellten neuen Tabellen `admin` SELECT-Berechtigungen für `user1`, INSERT-Berechtigungen für `group1` und UPDATE-Berechtigungen für `user2` gewährt.

PG_EXTERNAL_SCHEMA

Speichert Informationen über externe Schemata.

PG_EXTERNAL_SCHEMA ist für alle Benutzer sichtbar. Superuser können alle Zeilen anzeigen, während normale Benutzer nur die Metadaten anzeigen können, auf die sie Zugriff haben. Weitere Informationen finden Sie unter CREATE EXTERNAL SCHEMA.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
esoid	OID	ID des externen Schemas.
eskind	integer	Art des externen Schemas.
esdbname	Text	Name der externen Datenbank.
esoptions	Text	Optionen für das externe Schema.

Beispiel

Das folgende Beispiel zeigt Einzelheiten für externe Schemata.

```
select esoid, nspname as schemaname, nspowner, esdbname as external_db, esoptions
from pg_namespace a,pg_external_schema b where a.oid=b.esoid;

esoid  | schemaname      | nspowner | external_db | esoptions

-------+-----------------+----------+-------------
+-------------------------------------------------------------
100134 | spectrum_schema |      100 | spectrum_db |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
100135 | spectrum        |      100 | spectrumdb  |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
100149 | external        |      100 | external_db |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

PG_LIBRARY

Speichert Informationen zu benutzerdefinierten Bibliotheken.

PG_LIBRARY ist für alle Benutzer sichtbar. Superuser können alle Zeilen sehen; reguläre Benutzer können nur ihre eigenen Daten sehen. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
name	Name	Name der Bibliothek.
language_oid	OID	Reserviert für die Systemverwendung.
file_store_id	integer	Reserviert für die Systemverwendung.
owner	integer	Benutzer-ID des Eigentümers der Bibliothek.

Beispiel

Das folgende Beispiel gibt Informationen über von Benutzern installierte Bibliotheken zurück.

```
select * from pg_library;

name       | language_oid | file_store_id | owner
-----------+--------------+---------------+------
f_urlparse |       108254 |          2000 |   100
```

PG_PROC_INFO

PG_PROC_INFO ist eine Amazon-Redshift-Systemansicht in der PostgreSQL-Katalogtabelle PG_PROC und der internen Katalogtabelle PG_PROC_EXTENDED. PG_PROC_INFO enthält Details zu gespeicherten Prozeduren und Funktionen, einschließlich Informationen über Ausgabeargumente, falls vorhanden.

Tabellenspalten

PG_PROC_INFO zeigt die folgenden Spalten zusätzlich zu den Spalten in PG_PROC an. Die `oid`-Spalte in PG_PROC hat in der Tabelle PG_PROG_INFO den Namen `prooid`.

Spaltenname	Datentyp	Beschreibung
prooid	OID	Die Objekt-ID der Funktion oder gespeicherten Prozedur.
prokind	"char"	Ein Wert, der den Typ der Funktionen oder gespeicherten Prozeduren angibt. Dieser Wert lautet „f“ für reguläre Funktionen, „p“ für gespeicherte Prozeduren und „a“ für Aggregationsfunktionen.
proargmodes	"char"[ ]	Ein Array mit den Modi der Prozedurargumente, kodiert als „i“ für IN-Argumente, „o“ für OUT-Argumente und „b“ für INOUT-Argumente. Wenn alle Argumente IN-Argumente sind, ist das Feld NULL. Feldindizes entsprechen Positionen im proallargtypes-Array.
proallargtypes	oid[ ]	Ein Array mit den Datentypen der Prozedurargumente. Dieses Array enthält alle Argumenttypen (einschließlich OUT- und INOUT-Argumente). Wenn jedoch alle Argumente IN-Argumente sind, ist das Feld NULL. Die

Spaltenname	Datentyp	Beschreibung
		Feldindizierung ist eins-basiert. Im Gegensatz dazu wird proargtypes von null indiziert.

Das Feld proargnames in PG_PROC_INFO enthält die Namen aller Argumenttypen (einschließlich OUT und INOUT), falls vorhanden.

PG_STATISTIC_INDICATOR

Speichert Informationen zur Anzahl der seit dem letzten ANALYZE-Vorgang eingefügten oder gelöschten Zeilen. Die Tabelle PG_STATISTIC_INDICATOR wird häufig nach DML-Operationen aktualisiert, die statistischen Daten sind daher nur Näherungswerte.

PG_STATISTIC_INDICATOR ist nur für Superuser sichtbar. Weitere Informationen finden Sie unter Sichtbarkeit der Daten in Systemtabellen und Ansichten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
stairelid	OID	Tabellen-ID
stairows	float	Gesamtzahl der Zeilen in der Tabelle.
staiins	float	Anzahl der seit dem letzten ANALYZE-Vorgang eingefügten Zeilen.
staidels	float	Anzahl der seit dem letzten ANALYZE-Vorgang gelöschten oder aktualisierten Zeilen.

Beispiel

Das folgende Beispiel gibt Informationen zu den Tabellenänderungen seit dem letzten ANALYZE-Vorgang zurück.

```
select * from pg_statistic_indicator;

stairelid | stairows | staiins | staidels
----------+----------+---------+---------
   108271 |       11 |       0 |        0
   108275 |      365 |       0 |        0
   108278 |     8798 |       0 |        0
   108280 |    91865 |       0 |   100632
   108267 |    89981 |   49990 |     9999
   108269 |      808 |     606 |      374
   108282 |   152220 |   76110 |   248566
```

PG_TABLE_DEF

Speichert Informationen zu Tabellenspalten.

PG_TABLE_DEF gibt nur Informationen zu Tabellen zurück, die für den Benutzer sichtbar sind. Wenn PG_TABLE_DEF nicht die erwarteten Ergebnisse ausgibt, prüfen Sie, ob der Parameter search_path korrekt so eingestellt ist, dass die relevanten Schemata eingeschlossen sind.

Mit SVV_TABLE_INFO können Sie umfassendere Informationen zu einer Tabelle anzeigen, einschließlich der Datenverteilungsverzerrung, der Schlüsselverteilungsverzerrung, der Tabellengröße und statistischer Daten.

Tabellenspalten

Spaltenname	Datentyp	Beschreibung
schemaname	Name	Schemaname.
tablename	Name	Tabellenname.
column	Name	Spaltenname.
type	Text	Datentyp der Spalte.
encoding	character(32)	Kodierung der Spalte.

Spaltenname	Datentyp	Beschreibung
distkey	Boolean	„True“, wenn diese Spalte der Verteilungsschlüssel für die Tabelle ist.
sortkey	integer	Reihenfolge der Spalte im Sortierschlüssel. Wenn die Tabelle einen zusammengesetzten Sortierschlüssel verwendet, haben alle zu dem Sortierschlüssel gehörenden Spalten einen positiven Wert, der die Position der Spalte in dem Sortierschlüssel angibt. Wenn eine Tabelle einen überlappenden Sortierschlüssel verwendet, hat jede zu dem Sortierschlüssel gehörende Spalte einen positiven oder negativen Wert, wobei der absolute Wert die Position der Spalte in dem Sortierschlüssel angibt. Bei 0 gehört die Spalte nicht zu einem Sortierschlüssel.
notnull	Boolean	„True“, wenn für die Spalte eine NOT NULL-Einschränkung gilt.

Beispiel

Das folgende Beispiel zeigt die zusammengesetzten Sortierschlüsselspalten für die Tabelle LINEORDER_COMPOUND.

```
select "column", type, encoding, distkey, sortkey, "notnull"
from pg_table_def
where tablename = 'lineorder_compound'
and sortkey <> 0;

column       | type    | encoding | distkey | sortkey | notnull
-------------+---------+----------+---------+---------+--------
lo_orderkey  | integer | delta32k | false   |       1 | true
lo_custkey   | integer | none     | false   |       2 | true
lo_partkey   | integer | none     | true    |       3 | true
lo_suppkey   | integer | delta32k | false   |       4 | true
lo_orderdate | integer | delta    | false   |       5 | true
(5 rows)
```

Das folgende Beispiel zeigt die überlappenden Sortierschlüsselspalten für die Tabelle LINEORDER_INTERLEAVED.

```
select "column", type, encoding, distkey, sortkey, "notnull"
from pg_table_def
where tablename = 'lineorder_interleaved'
and sortkey <> 0;

column       | type    | encoding | distkey | sortkey | notnull
-------------+---------+----------+---------+---------+--------
lo_orderkey  | integer | delta32k | false   |      -1 | true
lo_custkey   | integer | none     | false   |       2 | true
lo_partkey   | integer | none     | true    |      -3 | true
lo_suppkey   | integer | delta32k | false   |       4 | true
lo_orderdate | integer | delta    | false   |      -5 | true
(5 rows)
```

PG_TABLE_DEF gibt nur Informationen für Tabellen in Schemata aus, die zum Suchpfad gehören. Weitere Informationen finden Sie unter search_path.

Nehmen Sie beispielsweise an, Sie erstellen ein neues Schema und eine neue Tabelle und fragen dann PG_TABLE_DEF ab.

```
create schema demo;
create table demo.demotable (one int);
select * from pg_table_def where tablename = 'demotable';

schemaname|tablename|column| type | encoding | distkey | sortkey | notnull
----------+---------+------+------+----------+---------+---------+--------
```

Die Abfrage gibt für die neue Tabelle keine Zeilen zurück. Prüfen Sie die Einstellung für `search_path`.

```
show search_path;

  search_path
---------------
 $user, public
(1 row)
```

Fügen Sie das Schema `demo` zum Suchpfad hinzu und führen Sie die Abfrage erneut aus.

```
set search_path to '$user', 'public', 'demo';

select * from pg_table_def where tablename = 'demotable';

schemaname| tablename |column|  type   | encoding |distkey|sortkey| notnull
----------+-----------+------+---------+----------+-------+-------+--------
demo      | demotable | one  | integer | none     | f     |     0 | f
(1 row)
```

PG_USER_INFO

PG_USER_INFO ist eine Amazon-Redshift-Systemansicht, die Benutzerinformationen wie Benutzer-ID und Ablaufzeit des Passworts anzeigt.

Nur Superuser können PG_USER_INFO sehen.

Tabellenspalten

PG_USER_INFO enthält die folgenden Spalten. Weitere Informationen finden Sie in der PostgreSQL-Dokumentation.

Spaltenname	Datentyp	Beschreibung
usename	Name	Benutzername
usesysid	Ganzzahl	Benutzer-ID
usecreatedb	boolesch	„True“, wenn der Benutzer Datenbanken erstellen kann.
usesuper	boolesch	„True“, wenn der Benutzer ein Superuser ist.
usecatupd	boolesch	„True“, wenn der Benutzer Systemkataloge aktualisieren kann.
passwd	text	Passwort
valuntil	abstime	Datum und Uhrzeit, an dem das Passwort abläuft.

Spaltenname	Datentyp	Beschreibung
useconfig	text[]	Die Sitzung wird standardmäßig für Laufzeitvariablen ausgeführt.
useconnlimit	Text	Die Anzahl der Verbindungen, die der Benutzer öffnen kann.

Abfragen der Katalogtabellen

Themen

- Beispiele für Katalogabfragen

Im Allgemeinen können Sie Katalogtabellen und Ansichten (Beziehungen, deren Namen mit **PG_** beginnen) mit Amazon-Redshift-Tabellen und -Ansichten verbinden.

Die Katalogtabellen verwenden eine Reihe von Datentypen, die Amazon Redshift nicht unterstützt. Die folgenden Datentypen werden unterstützt, wenn Abfragen Katalogtabellen mit Amazon-Redshift-Tabellen verbinden:

- bool
- „char“
- float4
- int2
- int4
- int8
- Name
- oid
- Text
- varchar

Wenn Sie eine Verbindungsabfrage schreiben, die explizit oder implizit auf eine Spalte verweist, die einen nicht unterstützten Datentyp aufweist, gibt die Abfrage einen Fehler aus. Die in einigen

Katalogtabellen verwendeten SQL-Funktionen werden ebenfalls nicht unterstützt, ausgenommen solche, die von den Tabellen PG_SETTINGS und PG_LOCKS verwendet werden.

So kann beispielsweise die Tabelle PG_STATS aufgrund nicht unterstützter Funktionen nicht in Verbindung mit einer Amazon-Redshift-Tabelle abgefragt werden.

Die folgenden Katalogtabellen und -ansichten bieten nützliche Informationen, die mit Informationen in Amazon-Redshift-Tabellen verbunden werden können. Einige dieser Tabellen erlauben aufgrund von Datentyp- und Funktionseinschränkungen nur den partiellen Zugriff. Wenn Sie die nur partiell zugänglichen Tabellen abfragen, wählen Sie deren Spalten sorgfältig aus.

Die folgenden Tabellen sind vollständig zugänglich und enthalten keine nicht unterstützten Typen oder Funktionen:

- pg_attribute
- pg_cast
- pg_depend
- pg_description
- pg_locks
- pg_opclass

Die folgenden Tabellen sind nur partiell zugänglich und enthalten nicht unterstützte Typen, Funktionen und abgeschnittene Textspalten. Werte in Textspalten werden auf varchar(256)-Werte verkürzt.

- pg_class
- pg_constraint
- pg_database
- pg_group
- pg_language
- pg_namespace
- pg_operator
- pg_proc
- pg_settings
- pg_statistic

- pg_tables
- pg_type
- pg_user
- pg_views

Die hier nicht aufgeführten Katalogtabellen sind entweder nicht zugänglich oder für Amazon-Redshift-Administratoren eher unwichtig. Sie können jedoch jede Katalogtabelle oder -ansicht offen abfragen, wenn die Abfrage keine Verbindung zu einer Amazon-Redshift-Tabelle beinhaltet.

Sie können die OID-Spalten in den Postgres-Katalogtabellen als Verbindungsspalten verwenden. Beispielsweise gleicht die Verbindungsbedingung `pg_database.oid = stv_tbl_perm.db_id` die interne Datenbankobjekt-ID für jede Zeile von PG_DATABASE mit der sichtbaren Spalte DB_ID in der Tabelle STV_TBL_PERM ab. Die OID-Spalten sind interne Primärschlüssel, die nicht sichtbar sind, wenn Sie eine Auswahl aus der Tabelle treffen. Die Katalogansichten haben keine OID-Spalten.

Einige Amazon-Redshift-Funktionen können nur in den Datenverarbeitungsknoten ausgeführt werden. Wenn eine Abfrage eine vom Benutzer erstellte Tabelle referenziert, wird SQL auf den Datenverarbeitungsknoten ausgeführt.

Eine Abfrage, die nur Katalogtabellen (Tabellen mit einem PG-Präfix wie PG_TABLE_DEF) oder keine Tabellen referenziert, wird ausschließlich auf dem Führungsknoten ausgeführt.

Wenn eine Abfrage, die eine Datenverarbeitungsknotenfunktion verwendet, auf keine benutzerdefinierte Tabelle oder Amazon-Redshift-Systemtabelle verweist, wird der folgende Fehler zurückgegeben.

```
[Amazon](500310) Invalid operation: One or more of the used functions must be applied
 on at least one user created table.
```

Die folgenden Amazon-Redshift-Funktionen sind reine Datenverarbeitungsknoten-Funktionen:

Funktionen für Systeminformationen

- LISTAGG
- MEDIAN
- PERCENTILE_CONT
- PERCENTILE_DISC und APPROXIMATE PERCENTILE_DISC

Beispiele für Katalogabfragen

Die folgenden Abfragen zeigen einige Möglichkeiten für die Abfrage von Katalogtabellen, um nützliche Informationen zu einer Amazon-Redshift-Datenbank zu erhalten.

Anzeige von Tabellen-ID, Datenbank, Schema und Tabellenname

Die folgende Ansichtsdefinition verbindet die Systemtabelle STV_TBL_PERM mit den Systemkatalogtabellen PG_CLASS, PG_NAMESPACE und PG_DATABASE zur Ausgabe von Tabellen-ID, Datenbankname, Schemaname und Tabellenname.

```
create view tables_vw as
select distinct(id) table_id
,trim(datname)   db_name
,trim(nspname)   schema_name
,trim(relname)   table_name
from stv_tbl_perm
join pg_class on pg_class.oid = stv_tbl_perm.id
join pg_namespace on pg_namespace.oid = relnamespace
join pg_database on pg_database.oid = stv_tbl_perm.db_id;
```

Das folgende Beispiel gibt die Informationen für Tabellen-ID 117855 zurück.

```
select * from tables_vw where table_id = 117855;
```

```
table_id | db_name   | schema_name | table_name
---------+-----------+-------------+-----------
  117855 |       dev | public      | customer
```

Auflisten der Anzahl der Spalten pro Amazon-Redshift-Tabelle

Die folgende Abfrage verknüpft einige Katalogtabellen, um herauszufinden, wie viele Spalten jede Amazon-Redshift-Tabelle enthält. Amazon-Redshift-Tabellennamen werden sowohl in PG_TABLES als auch in STV_TBL_PERM gespeichert. Verwenden Sie nach Möglichkeit PG_Tables, um Amazon-Redshift-Tabellennamen zurückzunehmen.

Diese Abfrage beinhaltet keine Amazon-Redshift-Tabellen.

```
select nspname, relname, max(attnum) as num_cols
from pg_attribute a, pg_namespace n, pg_class c
```

```
where n.oid = c.relnamespace and  a.attrelid = c.oid
and c.relname not like '%pkey'
and n.nspname not like 'pg%'
and n.nspname not like 'information%'
group by 1, 2
order by 1, 2;

nspname | relname  | num_cols
--------+----------+----------
public  | category |        4
public  | date     |        8
public  | event    |        6
public  | listing  |        8
public  | sales    |       10
public  | users    |       18
public  | venue    |        5
(7 rows)
```

Auflisten der Schemata und Tabellen in einer Datenbank

Die folgende Abfrage verbindet STV_TBL_PERM mit einigen PG-Tabellen zur Ausgabe einer Liste von Tabellen in der TICKIT-Datenbank und ihrer Schemanamen (Spalte NSPNAME). Dazu gibt die Abfrage die Gesamtzahl der Zeilen in jeder Tabelle aus. (Diese Abfrage ist nützlich, wenn mehrere Schemata in Ihrem System die gleichen Tabellennamen haben.)

```
select datname, nspname, relname, sum(rows) as rows
from pg_class, pg_namespace, pg_database, stv_tbl_perm
where pg_namespace.oid = relnamespace
and pg_class.oid = stv_tbl_perm.id
and pg_database.oid = stv_tbl_perm.db_id
and datname ='tickit'
group by datname, nspname, relname
order by datname, nspname, relname;

datname | nspname | relname  |  rows
--------+---------+----------+--------
tickit  | public  | category |     11
tickit  | public  | date     |    365
tickit  | public  | event    |   8798
tickit  | public  | listing  | 192497
tickit  | public  | sales    | 172456
tickit  | public  | users    |  49990
```

```
tickit  | public  | venue    |    202
(7 rows)
```

Auflisten von Tabellen-IDs, Datentypen, Spaltennamen und Tabellennamen

Die folgende Abfrage listet einige Informationen zu jeder Benutzertabelle und ihren Spalten aus: Tabellen-ID, Tabellenname, die Namen der Spalten sowie den Datentyp jeder Spalte:

```
select distinct attrelid, rtrim(name), attname, typname
from pg_attribute a, pg_type t, stv_tbl_perm p
where t.oid=a.atttypid and a.attrelid=p.id
and a.attrelid between 100100 and 110000
and typname not in('oid','xid','tid','cid')
order by a.attrelid asc, typname, attname;

attrelid |  rtrim   |    attname     |  typname
---------+----------+----------------+-----------
  100133 | users    | likebroadway   | bool
  100133 | users    | likeclassical  | bool
  100133 | users    | likeconcerts   | bool
...
  100137 | venue    | venuestate     | bpchar
  100137 | venue    | venueid        | int2
  100137 | venue    | venueseats     | int4
  100137 | venue    | venuecity      | varchar
...
```

Zählen der Anzahl der Datenblöcke für jede Spalte in einer Tabelle

Die folgende Abfrage verbindet die Tabelle STV_BLOCKLIST mit PG_CLASS zur Ausgabe von Speicherinformationen für die Spalten in der Tabelle SALES.

```
select col, count(*)
from stv_blocklist s, pg_class p
where s.tbl=p.oid and relname='sales'
group by col
order by col;

col | count
----+-------
  0 |     4
  1 |     4
```

```
  2 |     4
  3 |     4
  4 |     4
  5 |     4
  6 |     4
  7 |     4
  8 |     4
  9 |     8
 10 |     4
 12 |     4
 13 |     8
(13 rows)
```

Konfigurationsreferenz

Themen

- spectrum_enable_pseudo_columns
- enable_spectrum_oid
- spectrum_query_maxerror
- statement_timeout
- stored_proc_log_min_messages
- Zeitzone
- use_fips_ssl
- wlm_query_slot_count

Modifizieren der Serverkonfiguration

Sie können die Serverkonfiguration auf eine der folgenden Arten ändern:

- Mithilfe eines SET-Befehls zum Übergehen einer Einstellung nur für die Dauer der aktuellen Sitzung.

  Beispiel:

  ```
  set extra_float_digits to 2;
  ```

- Durch die Modifizierung der Parametergruppeneinstellungen für den Cluster. Zu den Parametergruppeneinstellungen gehören zusätzliche Parameter, die Sie konfigurieren können. Weitere Informationen finden Sie unter Amazon Redshift Parameter Groups (Amazon-Redshift-Parametergruppen) im Amazon-Redshift-Verwaltungshandbuch.
- Mithilfe des ALTER USER-Befehls zur Einrichtung eines Konfigurationsparameters auf einen neuen Wert für alle von dem angegebenen Benutzer ausgeführten Sitzungen.

  ```
  ALTER USER username SET parameter { TO | = } { value | DEFAULT }
  ```

Verwenden Sie den SHOW-Befehl zur Anzeige der aktuellen Parametereinstellungen. Verwenden Sie SHOW ALL zur Anzeige aller Einstellungen, die Sie mit dem SET-Befehl konfigurieren können.

```
SHOW ALL;

name                      | setting
```

```
--------------------------+--------------
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | 2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

> **Note**
>
> Beachten Sie, dass die Konfigurationsparameter auf die Datenbank angewendet werden, mit der Sie in Ihrem Data Warehouse verbunden sind.

analyze_threshold_percent

Werte (Standard in Fettdruck)

10, 0 bis 100,0

Beschreibung

Setzt den Schwellenwert für den Prozentsatz der geänderten Zeilen für die Analyse einer Tabelle. Zur Verkürzung der Verarbeitungszeit und zur Verbesserung der allgemeinen Systemleistung überspringt Amazon Redshift den ANALYZE-Vorgang für alle Tabellen mit einem geringeren Prozentsatz geänderter Zeilen als von `analyze_threshold_percent` angegeben. Beispiel: Wenn eine Tabelle 100 000 000 Zeilen enthält und sich seit dem letzten ANALYZE-Vorgang 9 000 000 Zeilen geändert haben, wird die Tabelle standardmäßig übersprungen, da weniger als 10 Prozent der Zeilen geändert wurden. Um Tabellen zu analysieren, wenn nur eine kleine Zahl von Zeilen geändert wurde, legen Sie `analyze_threshold_percent` auf eine beliebig kleine Zahl fest. Wenn Sie beispielsweise `analyze_threshold_percent` auf 0,01 festlegen, wird eine Tabelle mit 100.000.000 nicht übersprungen, wenn mindestens 10.000 Zeilen geändert wurden. Um alle Tabellen zu analysieren, auch wenn keine Zeilen geändert wurden, legen Sie `analyze_threshold_percent` auf 0 fest.

Sie können den Parameter `analyze_threshold_percent` mit einem SET-Befehl nur für die Dauer der aktuellen Sitzung ändern. Der Parameter kann nicht in einer Parametergruppe geändert werden.

Beispiel

```
set analyze_threshold_percent to 15;
set analyze_threshold_percent to 0.01;
set analyze_threshold_percent to 0;
```

cast_super_null_on_error

Werte (Standard in Fettdruck)

on, off

Beschreibung

Gibt an, dass Amazon Redshift einen NULL-Wert zurückgibt, wenn Ihre Abfrage im Standard-Lax-Modus erfolgt, wenn Sie versuchen, auf ein nicht vorhandenes Mitglied eines Objekts oder Elements eines Arrays zuzugreifen.

datashare_break_glass_session_var

Werte (Standard in Fettdruck)

Es ist kein Standard vorhanden. Der Wert kann eine beliebige Zeichenfolge sein. Diese wird von Amazon Redshift generiert, wenn eine nicht empfohlene Operation stattfindet, wie im Folgenden beschrieben.

Beschreibung

Wendet eine Berechtigung an, die bestimmte Operationen zulässt, die im Allgemeinen für eine AWS Data Exchange Datenfreigabe nicht empfohlen werden.

Im Allgemeinen wird empfohlen, eine Datenfreigabe nicht mit der Anweisung DROP DATASHARE oder ALTER AWS Data Exchange DATASHARE SET PUBLICACCESSIBLE zu löschen oder zu ändern. Um das Löschen oder Ändern einer AWS Data Exchange Datenfreigabe zu ermöglichen und so die Einstellung für den öffentlichen Zugriff zu deaktivieren, setzen Sie die Variable auf einen einmaligen Wert. `datashare_break_glass_session_var` Dieser einmalige Wert wird von Amazon Redshift generiert und nach dem ersten Versuch des betreffenden Vorgangs in einer Fehlermeldung angegeben.

Nachdem Sie die Variable auf den einmalig generierten Wert gesetzt haben, führen Sie die Anweisung DROP DATASHARE oder ALTER DATASHARE erneut aus.

Weitere Informationen finden Sie unter Nutzungshinweise für ALTER DATASHARE oder Nutzungshinweise für DROP DATASHARE.

Beispiel

```
set datashare_break_glass_session_var to '620c871f890c49';
```

datestyle

Werte (Standard in Fettdruck)

Formatangabe (ISO, Postgres, SQL oder Deutsch) und Sortierung nach Jahr/Monat/Tag (DMY, MDY, YMD).

- ISO – verwendet den Datumsstil JJJJ-MM-TT HH:MM:SS.
- Postgres – verwendet den Datumsstil MM-TT HH:MM:SS JJJJ.
- SQL – verwendet den Datumsstil MM-TT-JJJJ HH:MM:SS.
- Deutsch – verwendet den Datumsstil TT-MM-JJJJ HH:MM:SS.

Beschreibung

Stellt das Anzeigeformat für Datum und Uhrzeit sowie die Regeln für die Interpretation mehrdeutiger Dateneingabewerte ein. Die Zeichenfolge enthält zwei Parameter, die separat oder gemeinsam geändert werden können.

Beispiel

```
show datestyle;
DateStyle
-----------
ISO, MDY
(1 row)

set datestyle to 'SQL,DMY';
```

default_geometry_encoding

Werte (Standard in Fettdruck)

1, 2

Beschreibung

Eine Sitzungskonfiguration, die angibt, ob räumliche Geometrien, die während dieser Sitzung erstellt wurden, mit einer Bounding Box codiert werden. Wenn `default_geometry_encoding 1` ist, werden Geometrien nicht mit einer Bounding Box codiert. Wenn `default_geometry_encoding 2` ist, werden Geometrien mit einer Bounding Box codiert. Weitere Informationen zur Unterstützung von Bounding Boxes finden Sie unter Begrenzungsrahmen.

describe_field_name_in_uppercase

Werte (Standard in Fettdruck)

aus ("false"), an ("true")

Beschreibung

Gibt an, ob von SELECT-Anweisungen zurückgegebene Spaltennamen in Groß- oder Kleinbuchstaben zurückgegeben werden. Ist dieser Parameter aktiviert, werden Spaltennamen in Großbuchstaben angegeben. Ist dieser Parameter deaktiviert, werden Spaltennamen in Kleinbuchstaben angegeben. Amazon Redshift speichert Spaltennamen unabhängig von der Einstellung für in Kleinbuchstaben `describe_field_name_in_uppercase`.

Beispiel

```
set describe_field_name_in_uppercase to on;

show describe_field_name_in_uppercase;

DESCRIBE_FIELD_NAME_IN_UPPERCASE
--------------------------------
on
```

downcase_delimited_identifier

Werte (Standard in Fettdruck)

on, off

Beschreibung

Diese Konfiguration wird eingestellt. Verwenden Sie stattdessen `enable_case_sensitive_identifier`.

Ermöglicht dem Superparser das Lesen von JSON-Feldern, die in Groß- oder Großbuchstaben gehalten sind. Ermöglicht auch die Unterstützung von Verbundabfragen für unterstützte PostgreSQL-Datenbanken mit gemischten Namen von Datenbank, Schema, Tabelle und Spalte. Um Bezeichner mit Groß-/Kleinschreibung zu verwenden, setzen Sie diesen Parameter auf „off“.

Nutzungshinweise

- Wenn Sie Sicherheitsfunktionen auf Zeilenebene oder Funktionen zur dynamischen Datenmaskierung verwenden, empfehlen wir, den Wert `downcase_delimited_identifier` im Cluster oder in der Parametergruppe Ihrer Arbeitsgruppe festzulegen. Dadurch wird sichergestellt, dass `downcase_delimited_identifier` beim Erstellen und Anfügen einer Richtlinie und der anschließenden Abfrage einer Relation, auf die eine Richtlinie angewendet wurde, konstant bleibt. Weitere Informationen zur Sicherheit auf Zeilenebene finden Sie unter Sicherheit auf Zeilenebene. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.
- Wenn Sie die Option `downcase_delimited_identifier` deaktivieren und eine Tabelle erstellen, können Sie Spaltennamen festlegen, bei denen zwischen Groß- und Kleinschreibung unterschieden wird. Wenn Sie `downcase_delimited_identifier` aktivieren und die Tabelle abfragen, werden die Spaltennamen klein geschrieben. Dies kann zu anderen Abfrageergebnissen führen als mit deaktivierter Option `downcase_delimited_identifier`. Betrachten Sie das folgende Beispiel:

```
SET downcase_delimited_identifier TO off;
--Amazon Redshift preserves case for column names and other identifiers.

--Create a table with two columns that are identical except for the case.
CREATE TABLE t ("c" int, "C" int);
```

```
INSERT INTO t VALUES (1, 2);

SELECT * FROM t;

 c | C
---+---
 1 | 2
(1 row)

SET enable_downcase_delimited_identifier TO on;
--Amazon Redshift no longer preserves case for column names and other identifiers.

SELECT * FROM t;

 c | c
---+---
 1 | 1
(1 row)
```

- Wir empfehlen, dass normale Benutzer, die Tabellen mit dynamischer Datenmaskierung oder Sicherheitsrichtlinien auf Zeilenebene abfragen, die Standardeinstellung downcase_delimited_identifier verwenden. Weitere Informationen zu Sicherheit auf Zeilenebene finden Sie unter Sicherheit auf Zeilenebene. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

enable_case_sensitive_identifier

Werte (Standard in Fettdruck)

true (wahr), false (falsch)

Beschreibung

Ein Konfigurationswert, der bestimmt, ob Namensbezeichner von Datenbanken, Tabellen und Spalten die Groß- und Kleinschreibung beachten. Die Groß-/Kleinschreibung von Namensbezeichnern wird beibehalten, wenn sie in doppelte Anführungszeichen eingeschlossen werden. Wenn Sie `enable_case_sensitive_identifier` auf `true` festlegen, wird die Groß-/Kleinschreibung von Namensbezeichnern beibehalten. Wenn Sie `enable_case_sensitive_identifier` auf `false` festlegen, wird die Groß-/Kleinschreibung von Namensbezeichnern nicht beibehalten.

Die Groß-/Kleinschreibung eines in doppelte Anführungszeichen eingeschlossenen Benutzernamens bleibt unabhängig von der Einstellung der `enable_case_sensitive_identifier`-Konfigurationsoption immer erhalten.

Beispiele

Das folgende Beispiel zeigt, wie Bezeichner für Tabellen- und Spaltennamen, bei denen die Groß-/Kleinschreibung beachtet werden muss, erstellt und verwendet werden.

```
-- To create and use case sensitive identifiers
SET enable_case_sensitive_identifier TO true;

-- Create tables and columns with case sensitive identifiers
CREATE TABLE "MixedCasedTable" ("MixedCasedColumn" int);

CREATE TABLE MixedCasedTable (MixedCasedColumn int);

-- Now query with case sensitive identifiers
SELECT "MixedCasedColumn" FROM "MixedCasedTable";

MixedCasedColumn
------------------
(0 rows)


SELECT MixedCasedColumn FROM MixedCasedTable;

mixedcasedcolumn
------------------
(0 rows)
```

Das folgende Beispiel zeigt einen Fall, bei dem die Groß-/Kleinschreibung von Bezeichnern nicht beibehalten wird.

```
-- To not use case sensitive identifiers
RESET enable_case_sensitive_identifier;

-- Mixed case identifiers are lowercased
CREATE TABLE "MixedCasedTable2" ("MixedCasedColumn" int);

CREATE TABLE MixedCasedTable2 (MixedCasedColumn int);
```

```
ERROR:  Relation "mixedcasedtable2" already exists


SELECT "MixedCasedColumn" FROM "MixedCasedTable2";

 mixedcasedcolumn
------------------
(0 rows)


SELECT MixedCasedColumn FROM MixedCasedTable2;

 mixedcasedcolumn
------------------
(0 rows)
```

Nutzungshinweise

- Wenn Sie die automatische Aktualisierung für materialisierte Ansichten verwenden, empfehlen wir, den Wert `enable_case_sensitive_identifier` in Ihrem Cluster oder in der Parametergruppe Ihrer Arbeitsgruppe festzulegen. Dadurch wird sichergestellt, dass `enable_case_sensitive_identifier` konstant bleibt, wenn die materialisierten Ansichten aktualisiert werden. Informationen zur automatischen Aktualisierung materialisierter Ansichten finden Sie unter Aktualisieren einer materialisierten Ansicht. Informationen zum Festlegen von Konfigurationswerten in Parametergruppen finden Sie unter Amazon-Redshift-Parametergruppen im Amazon-Redshift-Verwaltungshandbuch.
- Wenn Sie Sicherheitsfunktionen auf Zeilenebene oder Funktionen zur dynamischen Datenmaskierung verwenden, empfehlen wir, den Wert `enable_case_sensitive_identifier` im Cluster oder in der Parametergruppe Ihrer Arbeitsgruppe festzulegen. Dadurch wird sichergestellt, dass `enable_case_sensitive_identifier` beim Erstellen und Anfügen einer Richtlinie und der anschließenden Abfrage einer Relation, auf die eine Richtlinie angewendet wurde, konstant bleibt. Weitere Informationen zur Sicherheit auf Zeilenebene finden Sie unter Sicherheit auf Zeilenebene. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.
- Wenn Sie die Option `enable_case_sensitive_identifier` aktivieren und eine Tabelle erstellen, können Sie Spaltennamen festlegen, bei denen zwischen Groß- und Kleinschreibung unterschieden wird. Wenn Sie `enable_case_sensitive_identifier` deaktivieren und die Tabelle abfragen, werden die Spaltennamen klein geschrieben. Dies kann zu anderen

Abfrageergebnissen führen als mit aktivierter Option `enable_case_sensitive_identifier`. Betrachten Sie das folgende Beispiel:

```
SET enable_case_sensitive_identifier TO on;
--Amazon Redshift preserves case for column names and other identifiers.

--Create a table with two columns that are identical except for the case.
CREATE TABLE t ("c" int, "C" int);

INSERT INTO t VALUES (1, 2);

SELECT * FROM t;

 c | C
---+---
 1 | 2
(1 row)

SET enable_case_sensitive_identifier TO off;
--Amazon Redshift no longer preserves case for column names and other identifiers.

SELECT * FROM t;

 c | c
---+---
 1 | 1
(1 row)
```

- Wir empfehlen, dass normale Benutzer, die Tabellen mit dynamischer Datenmaskierung abfragen oder angefügte Sicherheitsrichtlinien auf Zeilenebene verwenden, die Standardeinstellung downcase_delimited_identifier nutzen. Weitere Informationen zur Sicherheit auf Zeilenebene finden Sie unter Sicherheit auf Zeilenebene. Weitere Informationen zur dynamischen Datenmaskierung finden Sie unter Dynamische Datenmaskierung.

enable_case_sensitive_super_attribute

Werte (Standard in Fettdruck)

true (wahr), false (falsch)

Beschreibung

Ein Konfigurationswert, der bestimmt, ob die Groß- und Kleinschreibung bei der Navigation in SUPER-Datentypstrukturen bei Attributnamen ohne Trennzeichen beachtet wird. Wenn Sie `enable_case_sensitive_super_attribute` auf `true` festlegen, wird die Groß- und Kleinschreibung bei der Navigation in SUPER-Datentypstrukturen bei Attributnamen ohne Trennzeichen beachtet. Wenn Sie den Wert auf `false` festlegen, wird die Groß- und Kleinschreibung bei der Navigation in SUPER-Datentypstrukturen bei Attributnamen ohne Trennzeichen nicht beachtet.

Wenn Sie einen Attributnamen in doppelte Anführungszeichen setzen und `enable_case_sensitive_identifier` auf `true` festlegen, bleibt die Groß-/Kleinschreibung unabhängig von der Einstellung der Konfigurationsoption `enable_case_sensitive_super_attribute` immer erhalten.

`enable_case_sensitive_super_attribute` gilt nur für Spalten des SUPER-Datentyps. Für alle anderen Spalten sollten Sie stattdessen `enable_case_sensitive_identifier` verwenden.

Weitere Informationen zum SUPER-Datentyp finden Sie unter Typ SUPER und Erfassen und Abfragen von halbstrukturierten Daten in Amazon Redshift.

Beispiele

Das folgende Beispiel zeigt die Ergebnisse der Auswahl von SUPER-Werten, wenn `enable_case_sensitive_super_attribute` aktiviert bzw. deaktiviert ist.

```
--Create a table with a SUPER column.
CREATE TABLE tbl (col SUPER);

--Insert values.
INSERT INTO tbl VALUES (json_parse('{
 "A": "A", "a": "a"
}'));

SET enable_case_sensitive_super_attribute TO ON;

SELECT col.A FROM tbl;
  a
-----
 "A"
```

```
(1 row)

SELECT col.a FROM tbl;
  a
-----
 "a"
(1 row)

SET enable_case_sensitive_super_attribute TO OFF;

SELECT col.A FROM tbl;
  a
-----
 "a"
(1 row)

SELECT col.a FROM tbl;
  a
-----
 "a"
(1 row)
```

Nutzungshinweise

- Ansichten und materialisierte Ansichten folgen dem Wert von `enable_case_sensitive_super_attribute` zum Zeitpunkt ihrer Erstellung. Ansichten mit später Bindung, gespeicherte Prozeduren und benutzerdefinierte Funktionen folgen dem Wert von `enable_case_sensitive_super_attribute` zum Zeitpunkt der Abfrage.
- Wenn Sie die automatische Aktualisierung für materialisierte Ansichten verwenden, empfehlen wir, `enable_case_sensitive_identifier value` in der Parametergruppe Ihres Clusters oder Ihrer Arbeitsgruppe festzulegen. Dadurch wird sichergestellt, dass `enable_case_sensitive_identifier` konstant bleibt, wenn die materialisierten Ansichten aktualisiert werden. Informationen zur automatischen Aktualisierung materialisierter Ansichten finden Sie unter Aktualisieren einer materialisierten Ansicht. Informationen zum Festlegen von Konfigurationswerten in Parametergruppen finden Sie unter Amazon-Redshift-Parametergruppen im Amazon-Redshift-Verwaltungshandbuch.
- Der Spaltenname in den Anweisungsergebnissen wird unabhängig vom Wert von `enable_case_sensitive_super_attribute` immer in Kleinbuchstaben geschrieben. Damit auch beim Spaltennamen die Groß- und Kleinschreibung beachtet wird, aktivieren Sie `enable_case_sensitive_identifier`.

- Wir empfehlen, dass normale Benutzer, bei der Tabellenabfrage angefügte Sicherheitsrichtlinien auf Zeilenebene verwenden, die Standardeinstellung `enable_case_sensitive_identifier` nutzen. Weitere Informationen zu Sicherheit auf Zeilenebene finden Sie unter Sicherheit auf Zeilenebene.

enable_numeric_rounding

Werte (Standard in Fettdruck)

aktiviert („true“), deaktiviert („false“)

Beschreibung

Gibt an, ob numerisch gerundet werden soll. Wenn `enable_numeric_rounding` on ist, rundet Amazon Redshift NUMERIC-Werte, wenn sie in andere numerische Typen wie INTEGER oder DECIMAL umgewandelt werden. Wenn `enable_numeric_rounding` `off` ist, kürzt Amazon Redshift NUMERIC-Werte, wenn sie in andere numerische Typen umgewandelt werden. Weitere Informationen zu numerischen Typen finden Sie unter Numerische Typen.

Beispiel

```
--Create a table and insert the numeric value 1.5 into it.
CREATE TABLE t (a numeric(10, 2));

INSERT INTO t VALUES (1.5);

SET enable_numeric_rounding to ON;
--Amazon Redshift now rounds NUMERIC values when casting to other numeric types.

SELECT a::int FROM t;

 a
---
 2
(1 row)


SELECT a::decimal(10, 0) FROM t;

 a
```

```
---
 2
(1 row)


 SELECT a::decimal(10, 5) FROM t;

    a
---------
 1.50000
(1 row)


SET enable_numeric_rounding to OFF;
--Amazon Redshift now truncates NUMERIC values when casting to other numeric types.

SELECT a::int FROM t;

 a
---
 1
(1 row)


SELECT a::decimal(10, 0) FROM t;

 a
---
 1
(1 row)


SELECT a::decimal(10, 5) FROM t;

    a
---------
 1.50000
(1 row)
```

enable_result_cache_for_session

Werte (Standard in Fettdruck)

an ("true"), aus ("false")

Beschreibung

Gibt an, ob die Abfrageergebnisse zwischengespeichert werden sollen. Wenn für `enable_result_cache_for_session` die Option on festgelegt ist, sucht Amazon Redshift nach einer gültigen, zwischengespeicherten Kopie der Abfrageergebnisse, wenn eine Abfrage gesendet wird. Wenn im Ergebnis-Cache ein passender Datensatz gefunden wird, verwendet Amazon Redshift die zwischengespeicherten Ergebnisse und führt die Abfrage nicht aus. Wenn für `enable_result_cache_for_session` die Option `off` festgelegt ist, ignoriert Amazon Redshift den Ergebnis-Cache und führt alle abgesendeten Abfragen aus.

Beispiel

```
SET enable_result_cache_for_session TO off;
--Amazon Redshift now ignores the results cache
```

enable_vacuum_boost

Werte (Standard in Fettdruck)

false, true

Beschreibung

Gibt an, ob die Option vacuum boost für alle in einer Sitzung ausgeführten VACUUM-Befehle aktiviert werden soll. Wenn für `enable_vacuum_boost` die Option `true` festgelegt ist, führt Amazon Redshift alle VACUUM-Befehle in der Sitzung mit der Option BOOST aus. Wenn für `enable_vacuum_boost` die Option `false` festgelegt ist, wird Amazon Redshift standardmäßig nicht mit der Option BOOST ausgeführt. Weitere Informationen über die Option BOOST finden Sie unter VACUUM.

error_on_nondeterministic_update

Werte (Standard in Fettdruck)

false, true

Beschreibung

Gibt an, ob UPDATE-Abfragen mit mehreren Übereinstimmungen pro Zeile einen Fehler auslösen.

Beispiel

```
SET error_on_nondeterministic_update TO true;

CREATE TABLE t1(x1 int, y1 int);

CREATE TABLE t2(x2 int, y2 int);

INSERT INTO t1 VALUES (1,10), (2,20), (3,30);

INSERT INTO t2 VALUES (2,40), (2,50);

UPDATE t1 SET y1=y2 FROM t2 WHERE x1=x2;

ERROR: Found multiple matches to update the same tuple.
```

extra_float_digits

Werte (Standard in Fettdruck)

0, -15 bis 2

Beschreibung

Stellt die Anzahl der für Fließpunktwerte angezeigten Stellen an, einschließlich float4 und float8. Der Wert wird der Standardstellenanzahl hinzugefügt (FLT_DIG oder DBL_DIG). Der Wert kann bis auf 2 gesetzt werden, um teilweise bedeutsame Stellen einzuschließen. Dies ist besonders nützlich für die Ausgabe von Float-Daten, die exakt wiederhergestellt werden müssen. Es kann auch ein negativer Wert gewählt werden, um unerwünschte Stellen zu unterdrücken.

Beispiel

Im folgenden Beispiel wird `extra_float_digits` auf `-2` festgelegt. Zeigen Sie zunächst die aktuelle Parametereinstellung an.

```
show all;
      name                      | setting
--------------------------+----------------
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | 2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

Legen Sie dann den neuen Wert auf `-2` fest.

```
set extra_float_digits to -2;
```

Zeigen Sie abschließend die aktualisierte Parametereinstellung an.

```
show all;
      name                      | setting
--------------------------+----------------
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | -2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

interval_forbid_composite_literals

Werte (Standard in Fettdruck)

falsch, wahr

Beschreibung

Eine Sitzungskonfiguration, die den Wert eines Intervalls ändert, das sowohl die Teile JAHR BIS MONAT als auch TAG BIS SEKUNDE enthält.

`interval_forbid_composite_literals`Ist dies der `true` Fall, wird ein Fehler zurückgegeben, wenn ein Intervall mit den Teilen JAHR BIS MONAT und TAG BIS SEKUNDE gefunden wird. Die folgende SQL-Anweisung enthält beispielsweise ein INTERVALL VON TAG ZU SEKUNDE mit den Teilen JAHR BIS MONAT und TAG BIS SEKUNDE.

```
SELECT INTERVAL '1 year 1 day' DAY TO SECOND;
ERROR:  Interval Day To Second literal cannot contain year-month parts. Disable the GUC
 interval_forbid_composite_literals to suppress this error and silently discard the
 year-month part.
```

Wenn `interval_forbid_composite_literals` ja`false`, unterdrückt Amazon Redshift einen Fehler und schneidet den Wert JAHR BIS MONAT von einem INTERVALL DAY TO SECOND-Wert ab. Die folgende SQL-Anweisung enthält beispielsweise ein INTERVALL VON TAG ZU SEKUNDE mit den Teilen JAHR BIS MONAT und TAG BIS SEKUNDE.

```
SET interval_forbid_composite_literals to "false";
SELECT INTERVAL '1 year 1 day' DAY TO SECOND;


 intervald2s
------------------------------
1 days 0 hours 0 mins 0.0 secs
```

json_serialization_enable

Werte (Standard in Fettdruck)

false, true

Beschreibung

Eine Sitzungskonfiguration, die das JSON-Serialisierungsverhalten von ORC-, JSON-, Ion- und Parkett-formatierten Daten ändert. Wenn `json_serialization_enable` als `true` festgelegt ist, werden alle Sammlungen der obersten Ebene automatisch in JSON serialisiert und als

VARCHAR(65535) zurückgegeben. Nicht komplexe Spalten sind nicht betroffen oder serialisiert. Da Sammlungsspalten als VARCHAR(65535) serialisiert werden, kann auf ihre verschachtelten Unterfelder nicht direkt als Teil der Abfragesyntax zugegriffen werden (z. B. in der Filter-Klausel). Wenn `json_serialization_enable` als `false` festgelegt ist, werden Sammlungen der obersten Ebene nicht in JSON serialisiert. Weitere Informationen zur verschachtelten JSON-Serialisierung finden Sie unter Serialisieren komplexer verschachtelter JSON-Datentypen.

json_serialization_parse_nested_strings

Werte (Standard in Fettdruck)

false, true

Beschreibung

Eine Sitzungskonfiguration, die das JSON-Serialisierungsverhalten von ORC-, JSON-, Ion- und Parkett-formatierten Daten ändert. Wenn sowohl `json_serialization_parse_nested_strings` als auch `json_serialization_enable` als „true“ festgelegt sind, werden Zeichenfolgenwerte, die in komplexen Typen (wie Maps, Strukturen oder Arrays) gespeichert sind, analysiert und direkt in das Ergebnis geschrieben, wenn sie gültige JSON-Dateien sind. Wenn `json_serialization_parse_nested_strings` als „false“ festgelegt ist, werden Strings innerhalb verschachtelter komplexer Typen als Escape-JSON-Zeichenfolgen serialisiert. Weitere Informationen finden Sie unter Serialisieren komplexer Typen, die JSON-Zeichenfolgen enthalten.

max_concurrency_scaling_clusters

Werte (Standard in Fettdruck)

1, 0 bis 10

Beschreibung

Legt bei aktivierter Nebenläufigkeitsskalierung die maximale Anzahl der zulässigen Nebenläufigkeitsskalierungs-Cluster fest. Erhöhen Sie diesen Wert, wenn eine höhere Nebenläufigkeitsskalierung erforderlich ist. Verringern Sie den Wert, wenn Sie die Nutzung von Nebenläufigkeitsskalierungs-Clustern und die entstehenden Abrechnungskosten reduzieren möchten.

Die maximale Anzahl von Parallelitätsskalierungsclustern ist ein einstellbares Kontingent. Weitere Informationen finden Sie unter Amazon Redshift quotas (Amazon-Redshift-Kontingente) im Amazon-Redshift-Verwaltungshandbuch.

max_cursor_result_set_size

Werte (Standard in Fettdruck)

0 (Standard zum Maximum) - 14400000 MB

Beschreibung

Der Parameter `max_cursor_result_set_size` wird nicht mehr verwendet. Für weitere Informationen zur Cursorergebnisgröße vgl. Einschränkungen für Cursors.

mv_enable_aqmv_for_session

Werte (Standard in Fettdruck)

true (wahr), false (falsch)

Beschreibung

Gibt an, ob Amazon Redshift das automatische Umschreiben materialisierter Ansichten auf Sitzungsebene durchführen kann.

navigate_super_null_on_error

Werte (Standard in Fettdruck)

on, off

Beschreibung

Gibt an, dass Amazon Redshift einen NULL-Wert zurückgibt, wenn Ihre Abfrage im Standard-Lax-Modus erfolgt, wenn Sie versuchen, zu einem nicht vorhandenen Mitglied eines Objekts oder Elements eines Arrays zu navigieren.

parse_super_null_on_error

Werte (Standard in Fettdruck)

off, on

Beschreibung

Gibt an, dass Amazon Redshift einen NULL-Wert zurückgibt, wenn Ihre Abfrage im Strict-Modus erfolgt, wenn Amazon Redshift versucht, ein nicht vorhandenes Mitglied eines Objekts oder Elements eines Arrays zu parsen.

pg_federation_repeatable_read

Werte (Standard in Fettdruck)

true, false

Beschreibung

Gibt die Isolationsstufe der Verbundabfragetransaktion für die Ergebnisse aus der PostgreSQL-Datenbank an.

- Wenn `pg_federation_repeatable_read` auf true gesetzt wird, werden Verbundtransaktionen mit der Semantik der Isolationsstufe REPEATABLE READ verarbeitet. Dies ist die Standardeinstellung.
- Wenn `pg_federation_repeatable_read` auf false gesetzt wird, werden Verbundtransaktionen mit der Semantik der Isolationsstufe READ COMMITTED verarbeitet.

Weitere Informationen finden Sie hier:

- Einschränkungen beim Zugriff auf Verbunddaten mit Amazon Redshift.
- Verwalten gleichzeitiger Schreiboperationen.

Beispiele

Mit dem folgenden Befehl wird `pg_federation_repeatable_read` für eine Sitzung auf on gesetzt. Mit dem Befehl show wird der Wert des eingestellten Werts angezeigt.

```
set pg_federation_repeatable_read to on;
```

```
show pg_federation_repeatable_read;

pg_federation_repeatable_read
-----------------------------
on
```

query_group

Werte (Standard in Fettdruck)

Kein Standardwert; der Wert kann jede beliebige Zeichenfolge sein.

Beschreibung

Dieser Parameter verwendet eine benutzerdefinierte Beschriftung für eine Gruppe von Abfragen, die in der gleichen Sitzung ausgeführt werden. Diese Beschriftung wird in den Abfrageprotokollen erfasst. Sie kann zur Einschränkung der Ergebnisse verwendet werden, die von den Tabellen STL_QUERY und STV_INFLIGHT sowie der Ansicht SVL_QLOG zurückgegeben werden. So können Sie beispielsweise eine separate Beschriftung für jede Abfrage verwenden, die Sie zur eindeutigen Identifizierung von Abfragen ohne Suche nach ihren IDs verwenden.

Dieser Parameter ist nicht in der Serverkonfigurationsdatei vorhanden und muss zur Laufzeit mit einem SET-Befehl eingestellt werden. Obwohl Sie eine lange Zeichenfolge als Beschriftung verwenden können, wird die Beschriftung in der Spalte LABEL der Tabelle STL_QUERY und der Anzeige SVL_QLOG auf 30 Zeichen (und auf 15 Zeichen in STV_INFLIGHT) verkürzt.

Im folgenden Beispiel ist query_group auf **`Monday`** gesetzt, und dann werden mehrere Abfragen mit dieser Beschriftung durchgeführt:

```
set query_group to 'Monday';
SET
select * from category limit 1;
...
...
select query, pid, substring, elapsed, label
from svl_qlog where label ='Monday'
order by query;
```

```
query | pid  |        substring                   |  elapsed  | label
------+------+------------------------------------+-----------+--------
789   | 6084 | select * from category limit 1;    | 65468     | Monday
790   | 6084 | select query, trim(label) from ... | 1260327   | Monday
791   | 6084 | select * from svl_qlog where ..    | 2293547   | Monday
792   | 6084 | select count(*) from bigsales;     | 108235617 | Monday
...
```

search_path

Werte (Standard in Fettdruck)

'$user', public, schema_names

Eine durch Kommata getrennte Liste vorhandener Schemanamen. Wenn '$user' vorhanden ist, wird das Schema mit dem gleichen Namen wie `SESSION_USER` ersetzt, andernfalls wird es ignoriert.

Beschreibung

Gibt an, in welcher Reihenfolge die Schemata durchsucht werden, wenn ein Objekt (beispielsweise eine Tabelle oder eine Funktion) mit dem einfachen Namen ohne Schemakomponente referenziert wird:

- Externe Schemata und externe Tabellen unterstützen keine Suchpfade. Externe Tabellen müssen durch einen externen Schemanamen ausdrücklich qualifiziert werden.
- Wenn Objekte ohne spezifisches Zielschema erstellt werden, werden Sie in das erste in dem Suchpfad aufgeführte Schema gesetzt. Wenn der Suchpfad leer ist, gibt das System einen Fehler aus.
- Wenn Objekte mit unterschiedlichen Namen in unterschiedlichen Schemata existieren, wird das zuerst in dem Suchpfad stehende verwendet.
- Ein Objekt, das sich in keinem der Schemata in dem Suchpfad befindet, kann nur durch Angabe seines enthaltenden Schemas mit einem qualifizierten Namen (mit Punkten) referenziert werden.
- Das Systemkatalogschema, pg_catalog, ist immer ausgewählt. Wenn es in dem Pfad erwähnt wird, wird es in der angegebenen Reihenfolge durchsucht. Wenn dies nicht der Fall ist, wird es vor den Pfadelementen durchsucht.
- Das temporäre Tabellenschema der aktuellen Sitzung, pg_temp_nnn, wird, falls vorhanden, immer durchsucht. Es kann ausdrücklich in dem Pfad mittels des Alias pg_temp aufgeführt werden.

Wenn es nicht in dem Pfad aufgeführt wird, wird es zuerst (selbst vor pg_catalog) durchsucht. Das temporäre Schema wird jedoch nur nach Relationsnamen (Tabellen, Ansichten) durchsucht. Es wird nicht nach Funktionsnamen durchsucht.

Beispiel

Das folgende Beispiel erstellt das Schema ENTERPRISE und setzt den search_path auf ein neues Schema.

```
create schema enterprise;
set search_path to enterprise;
show search_path;

 search_path
-------------
 enterprise
(1 row)
```

Das folgende Beispiel fügt das Schema ENTERPRISE dem Standard-search_path hinzu.

```
set search_path to '$user', public, enterprise;
show search_path;

         search_path
-----------------------------
 "$user", public, enterprise
(1 row)
```

Das folgende Beispiel fügt die Tabelle FRONTIER dem Schema ENTERPRISE hinzu.

```
create table enterprise.frontier (c1 int);
```

Wenn die Tabelle PUBLIC.FRONTIER in derselben Datenbank erstellt wird und der Benutzer in einer Abfrage den Schemanamen nicht angibt, hat PUBLIC.FRONTIER Vorrang gegenüber ENTERPRISE.FRONTIER.

```
create table public.frontier(c1 int);
insert into enterprise.frontier values(1);
select * from frontier;
```

```
frontier
----
(0 rows)

select * from enterprise.frontier;

c1
----
1
(1 row)
```

spectrum_enable_pseudo_columns

Werte (Standard in Fettdruck)

true (wahr), false (falsch)

Beschreibung

Sie können die Erstellung von Pseudospalten für eine Sitzung deaktivieren, indem Sie den Konfigurationsparameter `spectrum_enable_pseudo_columns` auf `false` festlegen.

Beispiel

Mit dem folgenden Befehl wird die Erstellung von Pseudospalten für eine Sitzung deaktiviert.

```
set spectrum_enable_pseudo_columns to false;
```

enable_spectrum_oid

Werte (Standard in Fettdruck)

true (wahr), false (falsch)

Beschreibung

Sie können auch nur die Pseudospalte `$spectrum_oid` deaktivieren, indem Sie den Konfigurationsparameter `enable_spectrum_oid` auf `false` setzen.

Beispiel

Mit dem folgenden Befehl wird die Pseudospalte `$spectrum_oid` deaktiviert, indem der Konfigurationsparameter `enable_spectrum_oid` auf `false` gesetzt wird.

```
set enable_spectrum_oid to false;
```

spectrum_query_maxerror

Werte (Standard in Fettdruck)

-1, Ganzzahl

Beschreibung

Sie können eine Ganzzahl angeben, um die maximale Anzahl von zu akzeptierenden Fehlern anzugeben, bevor Sie die Abfrage abbrechen. Ein negativer Wert schaltet die maximale Fehlerdatenbehandlung aus. Die Ergebnisse werden protokolliert in SVL_SPECTRUM_SCAN_ERROR.

Beispiel

Im folgenden Beispiel werden ORC-Daten vorausgesetzt, die überschüssige Zeichen und ungültige Zeichen enthalten. Die Spaltendefinition für `my_string` gibt eine Länge von 3 Zeichen an. Im Folgenden finden Sie Beispieldaten für dieses Beispiel:

```
my_string
---------------
abcdef
gh
ab
```

Die folgenden Befehle legen die maximale Anzahl von Fehlern auf 1 fest und führen die Abfrage aus.

```
set spectrum_query_maxerror to 1;
SELECT my_string FROM orc_data;
```

Die Abfrage stoppt und die Ergebnisse werden protokolliert in SVL_SPECTRUM_SCAN_ERROR.

statement_timeout

Werte (Standard in Fettdruck)

0 (keine Begrenzung), x Millisekunden

Beschreibung

Bricht alle Anweisungen ab, die mehr als die angegebene Zahl von Millisekunden benötigen.

Der `statement_timeout`-Wert ist die maximale Zeitspanne, für die eine Abfrage laufen kann, bevor sie von Amazon Redshift beendet wird. Dies beinhaltet auch die Planung, Warteschlangen in Workload Management (WLM) und die Ausführungszeit. Vergleichen Sie diese Zeit mit dem WLM-Timeout (max_execution_time) und einem QMR-Wert (query_execution_time), die nur die Ausführungszeit beinhalten.

Wenn das WLM-Timeout (max_execution_time) auch als Teil einer WLM-Konfiguration angegeben wird, wird die untere der Anweisungen statement_timeout und max_execution_time verwendet. Weitere Informationen finden Sie unter WLM-Timeout.

Beispiel

Da die folgende Abfrage länger als 1 Millisekunde dauert, wird Sie wegen Zeitüberschreitung abgebrochen.

```
set statement_timeout = 1;

select * from listing where listid>5000;
ERROR:  Query (150) canceled on user's request
```

stored_proc_log_min_messages

Werte (Standard in Fettdruck)

LOG, INFO, NOTICE, WARNING, EXCEPTION

Beschreibung

Gibt die Mindestprotokollierungsebene für ausgegebene gespeicherte Prozedurmeldungen an. Nachrichten auf oder über der angegebenen Ebene werden protokolliert. Der Standardwert ist LOG

(alle Meldungen werden protokolliert). Die Reihenfolge der Protokollierungsebenen von der höchsten zur niedrigsten ist:

1. EXCEPTION
2. WARNING
3. NOTICE
4. INFO
5. LOG

Wenn Sie z. B. den Wert NOTICE angeben, werden Meldungen nur für NOTICE, WARNING und EXCEPTION protokolliert.

Zeitzone

Werte (Standard in Fettdruck)

UTC, Zeitzone

Syntax

```
SET timezone { TO | = } [ time_zone | DEFAULT ]

SET time zone [ time_zone | DEFAULT ]
```

Beschreibung

Legt die Zeitzone für die aktuelle Sitzung fest. Die Zeitzone kann ein Offset von UTC (Universal Coordinated Time) oder einem Zeitzonennamen sein.

> **Note**
>
> Der Konfigurationsparameter `timezone` kann nicht mit einer Cluster-Parametergruppe eingerichtet werden. Die Zeitzone kann nur für die aktuelle Sitzung mit einem SET-Befehl eingerichtet werden. Um die Zeitzone für alle von einem bestimmten Datenbankbenutzer ausgeführten Sitzungen einzurichten, verwenden Sie den Befehl ALTER USER. ALTER USER … SET TIMEZONE ändert die Zeitzone für aufeinander folgende Sitzungen, nicht für die aktuelle Sitzung.

Wenn Sie die Zeitzone mit dem Befehl `SET timezone` (ein Wort) mit `TO` oder = einrichten, können Sie time_zone als Zeitzonenname, ein Offset im POSIX-Stil-Format oder im ISO-8601-Format angeben, wie nachfolgend gezeigt.

```
SET timezone { TO | = } time_zone
```

Wenn Sie die Zeitzone mit dem Befehl SET einrichten, ohne `TO` oder =, können Sie time_zone mit einem INTERVAL und einem Zeitzonennamen, einem Offset im POSIX-Stil-Format oder im ISO-8601-Format angeben, wie nachfolgend gezeigt.

```
SET time zone time_zone
```

Zeitzonenformate

Amazon Redshift unterstützt die folgenden Zeitzonenformate:

- Name der Zeitzone
- INTERVAL
- Zeitzonenangabe im POSIX-Stil
- ISO-8601-Offset

Da Zeitzonenabkürzungen, wie PST oder PDT, als feste Offsets von UTC definiert sind und Sommerzeitregeln nicht berücksichtigt werden, unterstützt der SET-Befehl keine Zeitzonenabkürzungen.

Für weitere Einzelheiten zu Zeitzonenformaten siehe unten.

Time zone name (Zeitzonenname) – der vollständige Zeitzonenname, etwa Amerika/New_York. Vollständige Zeitzonennamen können Sommerzeitregeln enthalten.

Es folgen einige Beispiele für Zeitzonennamen:

- Etc/Greenwich
- Amerika/New_York
- CST6CDT
- GB

> **Note**
>
> Viele Zeitzonennamen, etwa EST, MST, NZ oder UCT, sind gleichzeitig Abkürzungen.

Führen Sie den folgenden Befehl aus, um eine Liste aller gültigen Zeitzonen anzuzeigen.

```
select pg_timezone_names();
```

INTERVAL – ein Offset von UTC. Zum Beispiel: PST ist 8:00 oder 8 Stunden.

Es folgen einige Beispiele für INTERVAL-Zeitzonenoffsets::

- 8:00
- 8 Stunden
- 30 Minuten

POSIX-Stil-Format – eine Zeitzonenangabe in der Form STDoffset oder STDoffsetDST, wobei STD eine Zeitzonenabkürzung, offset der numerische Offset in Stunden westlich von UTC und DST eine optionale Abkürzung für die Sommerzeit ist. Für die Sommerzeit wird angenommen, dass Sie eine Stunde vor der angegebenen Verschiebung liegt.

Zeitzonenformate im POSIX-Stil verwenden positive Verschiebungen westlich von Greenwich; im Gegensatz dazu verwendet die ISO-8601-Konvention östlich von Greenwich positive Werte.

Es folgen einige Beispiele für Zeitzonen im POSIX-Stil:

- PST8
- PST8PDT
- EST5
- EST5EDT

> **Note**
>
> Amazon Redshift validiert Zeitzonenangaben im POSIX-Stil-Format nicht, es kann daher sein, dass die Zeitzone auf einen ungültigen Wert gesetzt wird. Beispielsweise führt der folgende

Befehl nicht zu einem Fehler, obwohl dadurch die Zeitzone auf einen ungültigen Wert gesetzt wird.

```
set timezone to ‘xxx36’;
```

ISO-8601-Offset – der Offset von UTC in der Form `±[hh]:[mm]`.

Es folgen Beispiele für ISO-8601-Offsets:

- -8:00
- +7:30

Beispiele

Im folgenden Beispiel wird die Zeitzone für die aktuelle Sitzung auf New York festgelegt.

```
set timezone = 'America/New_York';
```

Im folgenden Beispiel wird die Zeitzone für die aktuelle Sitzung auf UTC–8 (PST) festgelegt.

```
set timezone to '-8:00';
```

Das folgende Beispiel verwendet INTERVAL, um die Zeitzone auf PST festzusetzen.

```
set timezone interval '-8 hours'
```

Das folgende Beispiel setzt die Zeitzone für die aktuelle Sitzung auf die Standardzeitzone des Systems (UTC) zurück.

```
set timezone to default;
```

Um die Zeitzone für den Datenbankbenutzer einzurichten, verwenden Sie eine ALTER USER ... SET-Anweisung. Im folgenden Beispiel wird die Zeitzone für dbuser auf New York festgelegt. Der neue Wert bleibt für den Benutzer für alle folgenden Sitzungen erhalten.

```
ALTER USER dbuser SET timezone to 'America/New_York';
```

use_fips_ssl

Werte (Standard in Fettdruck)

true (wahr), false (falsch)

Beschreibung

Ein Parametergruppenwert, der angibt, ob der FIPS-konforme SSL-Modus verwendet wird. Wenn j`true`, `use_fips_ssl` wird der FIPS-konforme SSL-Modus verwendet. Wenn j`false`, `use_fips_ssl` wird der FIPS-konforme SSL-Modus nicht verwendet. Weitere Informationen finden Sie unter Konfiguration von Sicherheitsoptionen für Verbindungen im Amazon Redshift Management Guide.

Informationen zur Konfiguration von Parametern für einen von Amazon Redshift bereitgestellten Cluster finden Sie unter Über Parametergruppen im Amazon Redshift Management Guide. Informationen zur Konfiguration von Parametern für Redshift Serverless finden Sie unter Konfiguration einer FIPS-konformen SSL-Verbindung zu Amazon Redshift Serverless im Amazon Redshift Management Guide und/oder oder in der Redshift Serverless API-Referenz. CreateWorkgroupUpdateWorkgroup

wlm_query_slot_count

Werte (Standard in Fettdruck)

1, 1 bis 50 (nicht mehr als die Anzahl der verfügbaren Slots (Gleichzeitigkeitsniveau) für die Serviceklasse)

Beschreibung

Legt die Anzahl der Abfrageslots fest, die eine Abfrage verwendet.

Workload Management (WLM) reserviert die Plätze in einer Abfragewarteschlange entsprechend der für die Warteschlange festgelegten Gleichzeitigkeitsstufe. Ist die Gleichzeitigkeitsstufe beispielsweise auf 5 gesetzt, hat die Serviceklasse 5 Slots. WLM weist den verfügbaren Speicher für eine Serviceklasse gleichzeitig jedem Slot zu. Weitere Informationen finden Sie unter Implementierung von Workload Management.

> **Note**
>
> Wenn der Wert von wlm_query_slot_count größer als die Anzahl der verfügbaren Slots (Gleichzeitigkeitsniveau) für die Serviceklasse ist, schlägt die Abfrage fehl. Wenn ein Fehler auftritt, verringern Sie wlm_query_slot_count auf einen zulässigen Wert.

Bei Operationen, bei denen die Leistung von der Größe des zugewiesenen Speichers stark beeinflusst wird, etwa bei der Bereinigung, kann die Erhöhung von wlm_query_slot_count die Leistung verbessern. Inspizieren Sie insbesondere für langsame Bereinigungsbefehle den entsprechenden Datensatz in der Ansicht SVV_VACUUM_SUMMARY. Wenn Sie hohe Werte (nahe bei oder über 100) für sort_partitions und merge_increments in der Ansicht SVV_VACUUM_SUMMARY sehen, sollten Sie den Wert für wlm_query_slot_count bei der nächsten Bereinigung dieser Tabelle erhöhen.

Die Erhöhung des Werts für wlm_query_slot_count begrenzt die Anzahl der gleichzeitigen Abfragen, die ausgeführt werden können. Angenommen, die Serviceklasse hat ein Gleichzeitigkeitsniveau von 5 und wlm_query_slot_count ist auf 3 gesetzt. Während eine Abfrage in der Sitzung mit einem wlm_query_slot_count-Wert von 3 läuft, können maximal 2 weitere gleichzeitige Abfragen in derselben Serviceklasse ausgeführt werden. Nachfolgende Abfragen warten in der Warteschlange, bis die derzeit ausgeführten Abfragen abgeschlossen und die Slots freigegeben sind.

Beispiele

Verwenden Sie den SET-Befehl. um den Wert von wlm_query_slot_count für die Dauer der aktuellen Sitzung einzustellen.

```
set wlm_query_slot_count to 3;
```

Dokumentverlauf

> **Note**
>
> Eine Beschreibung der neuen Funktionen in Amazon Redshift finden Sie unter Was ist neu?.

In der folgenden Tabelle werden die wichtigen Dokumentationsänderungen am Datenbankentwicklerhandbuch zu Amazon Redshift nach Mai 2018 beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

API-Version: 2012-12-01

Eine Liste der Änderungen am Amazon-Redshift-Clusterverwaltungshandbuch finden Sie unter Dokumentverlauf für das Amazon-Redshift-Verwaltungshandbuch.

Weitere Informationen zu neuen Funktionen, einschließlich einer Liste von Korrekturen und der zugehörigen Cluster-Versionsnummern, finden Sie im Cluster-Versionsverlauf.

Änderung	Beschreibung	Datum
Support für räumliche 3D- und 4D-Geometrien und neue räumliche Funktionen	Sie können jetzt zusätzliche räumliche Funktionen verwenden, und einige Funktionen werden um 3D- und 4D-Geometrieunterstützung ergänzt.	19. August 2021
Support für die Codierung der Spaltenkomprimierung zur automatischen Tabelleno ptimierung	Sie können die Option ENCODE AUTO für eine Tabelle angeben, damit die Komprimierungskodierung für alle Spalten in der Tabelle automatisch verwaltet werden kann.	3. August 2021

Support für mehrere SQL-Anweisungen oder eine SQL-Anweisung mit Parametern, die die Amazon Redshift Data API verwenden	Sie können jetzt mehrere SQL-Anweisungen oder eine Anweisung mit Parametern mit der Amazon Redshift Data API ausführen.	28. Juli 2021
Support für Kollation ohne Berücksichtigung der Groß- und Kleinschreibung mit Überschreibung auf Spaltenebene	Sie können jetzt die Klausel COLLATE in einer Anweisung CREATE DATABASE verwenden, um die Standardkollation anzugeben.	24. Juni 2021
Support für die Freigabe von Daten über mehrere Konten	Sie können jetzt Daten übergreifend über AWS-Konten freigeben.	30. April 2021
Support für hierarchische Datenabfragen mit rekursivem CTE	Sie können jetzt einen rekursiven gemeinsamen Tabellenausdruck (CTE) in SQL verwenden.	29. April 2021
Support von datenbank übergreifenden Abfragen	Sie können jetzt Daten über Datenbanken in einem Cluster abfragen.	10. März 2021
Support der fein abgestuften Zugriffskontrolle auf COPY- und UNLOAD-Befehlen	Sie können jetzt bestimmten Benutzern und Gruppen in Ihrem Amazon-Redshift-Cluster die Berechtigung zum Ausführen von COPY- und UNLOAD-Befehlen erteilen, um eine feiner abgestufte Richtilinie für die Zugriffssteuerung zu erstellen.	12. Januar 2021
Support für native JSON und teilweise strukturierte Daten	Sie können jetzt den SUPER-Datentyp definieren.	9. Dezember 2020

Unterstützung für Verbundabfragen in MySQL	Sie können jetzt eine Verbundabfrage in eine unterstützte MySQL Engine schreiben.	9. Dezember 2020
Support für die gemeinsame Nutzung von Daten	Sie können jetzt Daten über Amazon-Redshift-Cluster hinweg freigeben.	9. Dezember 2020
Support für die automatische Tabellenoptimierung	Sie können jetzt automatische Verteilung und Sortierschlüssel definieren.	9. Dezember 2020
Support für Amazon Redshift ML	Sie können jetzt Machine Learning (ML)-Modelle erstellen, trainieren und bereitstellen.	08. Dezember 2020
Support für automatisches Aktualisieren und Umschreiben von Abfragen von materialisierten Ansichten	Sie können jetzt materialisierte Ansichten up-to-date mit automatischer Aktualisierung beibehalten und die Abfrageleistung durch automatisches Umschreiben verbessern.	11. November 2020
Support für TIME- und TIMETZ-Datentypen	Sie können nun Tabellen mit den Datentypen TIME und TIMEZ erstellen. Der Datentyp TIME speichert die Tageszeit ohne Zeitzoneninformationen und TIMEZ speichert die Tageszeit einschließlich Zeitzoneninformationen	11. November 2020
Unterstützung für Lambda-UDFs und Aufgliederung in Token	Sie können jetzt Lambda-UDFs schreiben, um eine externe Aufgliederung von Daten in Token zu ermöglichen.	26. Oktober 2020

Support für das Ändern einer Tabellenspaltencodierung	Sie können nun eine Tabellenspaltencodierung ändern.	20. Oktober 2020
Support für datenbank übergreifende Abfragen	Amazon Redshift kann jetzt Daten datenbankübergreifend in einem Cluster abfragen.	15. Oktober 2020
Unterstützung für HyperLogLog -Sketches	Amazon Redshift kann jetzt speichern und verarbeiten HyperLogLogSketches.	2. Oktober 2020
Support für Apache Hudi und Delta Lake	Verbesserungen beim Erstellen externer Tabellen für Redshift Spectrum.	24. September 2020
Support für Verbesserungen beim Abfragen von räumlichen Daten	Zu den Verbesserungen gehören das Laden eines Shapefile und mehrere neue Geo-SQL-Funktionen.	15. September 2020
Die materialisierte Ansicht unterstützt externe Tabellen	Sie können materialisierte Ansichten in Amazon Redshift erstellen, die auf externe Datenquellen verweisen.	19. Juni 2020
Unterstützung für das Schreiben in eine externe Tabelle	Sie können in externe Tabellen schreiben, indem Sie CREATE EXTERNAL TABLE AS SELECT ausführen, um in eine neue externe Tabelle zu schreiben, oder INSERT INTO, um Daten in eine vorhandene externe Tabelle einzufügen.	08. Juni 2020

Support für Speichers teuerelemente für Schemas	Aktualisiert Befehle und Ansichten, die Speichers teuerelemente für Schemas verwalten.	2. Juni 2020
Unterstützung für allgemeine Verfügbarkeit von Verbundab fragen	Informationen zum Abfragen von Daten mit Verbundab fragen aktualisiert.	16. April 2020
Unterstützung zusätzlicher räumlicher Funktionen	Beschreibungen zusätzlic her räumlicher Funktionen hinzugefügt.	2. April 2020
Unterstützung für die allgemeine Verfügbarkeit materialisierte Ansichten	Materialisierte Ansichten sind ab der Clusterversion 1.0.13059 allgemein verfügbar .	19. Februar 2020
Unterstützung von Berechtig ungen auf Spaltenebene	Berechtigungen auf Spalteneb ene sind ab der Clusterversion 1.0.13059 verfügbar.	19. Februar 2020
ALTER TABLE	Sie können einen ALTER TABLE-Befehl mit der ALTER DISTSTYLE ALL-Klausel verwenden, um den Verteilun gsstil einer Tabelle zu ändern.	11. Februar 2020
Unterstützung für Verbundab fragen	Das Handbuch enthält nun Informationen zur Verbundab frage mit einem aktualisi erten CREATE EXTERNAL SCHEMA.	3. Dezember 2019
Unterstützung für den Data-Lake-Export	Das Handbuch enthält nun Informationen zu neuen Parametern des UNLOAD-Be fehls.	3. Dezember 2019

Unterstützung von Geodaten	Der Leitfaden wurde aktualisiert, um die Unterstützung für Geodaten zu beschreiben.	21. November 2019
Unterstützung für die neue Konsole	Das Handbuch enthält nun eine Beschreibung der neuen Amazon-Redshift-Konsole.	11. November 2019
Unterstützung für die automatische Tabellensortierung	Amazon Redshift kann Tabellendaten automatisch sortieren.	7. November 2019
Unterstützung der Option VACUUM BOOST	Sie können die BOOST-Option beim Vacuum von Tabellen verwenden.	7. November 2019
Unterstützung für Standard-IDENTITY-Spalten	Sie können Tabellen mit Standard-IDENTITY-Spalten erstellen.	19. September 2019
Unterstützung für AZ64-Komprimierungscodierung	Sie können einige Spalten mit der AZ64-Komprimierungscodierung codieren.	19. September 2019
Unterstützung für Abfragepriioritäten	Sie können die Abfragepriorität einer automatischen WLM-Warteschlange festlegen.	22. August 2019
Unterstützung fü AWS Lake Formation	Sie können einen Lake Formation Data Catalog mit Amazon Redshift Spectrum verwenden.	8. August 2019

COMPUPDATE PRESET	Sie können einen COPY-Befehl mit COMPUPDATE PRESET verwenden, um Amazon Redshift das Auswählen der Komprimierungscodierung zu ermöglichen.	13. Juni 2019
ALTER COLUMN	Sie können einen ALTER TABLE-Befehl mit ALTER COLUMN verwenden, um die Größe einer VARCHAR-Spalte zu erhöhen.	22. Mai 2019
Support für gespeicherte Prozeduren	Sie können PL/pgSQL-gespeicherte Prozesse in Amazon Redshift definieren.	24. April 2019
Unterstützung für eine automatische Workload Management (WLM)-Konfiguration	Sie können festlegen, dass Amazon Redshift mit dem automatischen WLM ausgeführt wird.	24. April 2019
UNLOAD zu Zstandard	Verwenden Sie den UNLOAD-Befehl, um die Zstandard-Komprimierung auf in Amazon S3 entladene Text- und CSV-Dateien anzuwenden.	3. April 2019
Nebenläufigkeitsskalierung	Bei aktivierter Nebenläufigkeitsskalierung fügt Amazon Redshift automatisch zusätzliche Cluster-Kapazität hinzu, wenn diese benötigt wird, um eine gestiegene Zahl von gleichzeitigen Leseabfragen zu verarbeiten.	21. März 2019

UNLOAD zu CSV	Sie können den UNLOAD-Befehl zum Entladen in eine als CSV-Text formatierte Datei verwenden.	13. März 2019
Verteilungsstil AUTO	Um die automatische Verteilung zu aktivieren, können Sie den AUTO-Verteilungsstil mit einer CREATE TABLE-Anweisung festlegen. Wenn Sie die automatische Verteilung aktivieren, weist Amazon Redshift auf der Grundlage der Größe der Tabellendaten den optimalen Verteilungsstil zu. Die Änderung der Verteilung wird im Hintergrund durchgeführt und dauert nur wenige Sekunden.	23. Januar 2019
COPY aus Parquet unterstützt SMALLINT	COPY unterstützt jetzt das Laden aus Parquet-formatierten Dateien in Spalten, die den SMALLINT-Datentyp verwenden. Weitere Informationen finden Sie unter COPY aus spaltenbasierten Datenformaten	2. Januar 2019
DROP EXTERNAL DATABASE	Sie können externe Datenbanken löschen, indem Sie in einem DROP SCHEMA-Befehl eine DROP EXTERNAL DATABASE-Klausel verwenden.	3. Dezember 2018

Regionenübergreifende UNLOAD-Operation	Sie können einen UNLOAD-Befehl in einen Amazon-S3-Bucket in einer anderen AWS-Region durchführen, indem Sie den Parameter REGION angeben.	31. Oktober 2018
Automatisches Aufrufen von VACUUM DELETE	Amazon Redshift führt automatisch eine VACCUM DELETE-Option im Hintergrund aus, sodass Sie nur selten (wenn überhaupt) eine DELETE ONLY-Bereinigung ausführen müssen. Amazon Redshift legt für VACUUM DELETE eine zeitplangesteuerte Ausführung so fest, dass diese Ausführung in Zeiten fällt, in denen der Workload gering ist, und hält die Ausführung bei hoher Auslastung an.	31. Oktober 2018
Automatische Verteilung	Wenn Sie eine CREATE TABLE-Anweisung ohne Verteilungsstil angeben, ordnet Amazon Redshift auf der Grundlage der Tabellendaten den optimalen Verteilungsstil zu. Die Änderung der Verteilung wird im Hintergrund durchgeführt und dauert nur wenige Sekunden.	31. Oktober 2018

Differenzierte Zugriffskontrolle für den AWS Glue Data Catalog	Sie können jetzt verschied ene Zugangsstufen zu Daten in festlegen AWS Glue Data Catalog.	15. Oktober 2018
UNLOAD mit Datentypen	Sie können für einen UNLOAD-Befehl die Option MANIFEST VERBOSE angeben, um der Manifestd atei Metadaten hinzuzufügen, beispielsweise Spaltennamen und -datentypen, Dateigrößen und Zeilenanzahl.	10. Oktober 2018
Hinzufügen mehrerer Partition en mit einer einzelnen ALTER TABLE-Anweisung	Sie können bei externen Redshift Spectrum-Tabellen mehrere PARTITION-Klauseln zu einer ALTER TABLE ADD-Anweisung zusammenbinden. Weitere Informationen finden Sie unter Beispiele für das Ändern einer externen Tabelle.	10. Oktober 2018
UNLOAD mit Kopfzeile	Sie können für einen UNLOAD-Befehl die Option HEADER angeben, um zu Beginn jeder Ausgabedatei eine Kopfzeile hinzuzufügen, die Spaltennamen enthält.	19. September 2018
Neue Systemtabellen und Ansichten	Hinzufügung der Dokumenta tion fürSVL_S3Retries, SVL_USER_INFO und STL_DISK_FULL_DIAG.	31. August 2018

Unterstützung verschachtelter Daten in Amazon Redshift Spectrum	Sie können nun verschachtelte Daten abfragen, die in Amazon Redshift Spectrum-Tabellen gespeichert sind. Weitere Informationen finden Sie unter Tutorial: Abfragen verschachtelter Daten mit Amazon Redshift Spectrum.	8. August 2018
SQA standardmäßig aktiviert	Short Query Acceleration (SQA) ist jetzt standardmäßig für alle neuen Cluster aktiviert. SQA bietet mithilfe von maschinellem Lernen eine höhere Leistung, schnellere Ergebnisse und eine bessere Vorhersehbarkeit der Anfrageausführungszeiten. Weitere Informationen finden Sie unter Short Query Acceleration.	8. August 2018
Amazon Redshift Advisor	Sie können jetzt individuelle Empfehlungen zur Verbesserung der Clusterleistung und zur Reduzierung der Betriebskosten von Amazon Redshift Advisor erhalten. Weitere Informationen dazu finden Sie unter Amazon Redshift Advisor.	26. Juli 2018
Sofortige Alias-Referenz	Sie können jetzt sofort nach seiner Definition auf einen Alias-Ausdruck verweisen. Weitere Informationen finden Sie unter SELECT-Liste.	18. Juli 2018

Angeben des Komprimierungstyps beim Erstellen einer externen Tabelle	Sie können beim Erstellen einer externen Tabelle mit Amazon Redshift Spectrum jetzt den Komprimierungstyp angeben. Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.	27. Juni 2018
PG_LAST_UNLOAD_ID	Es wurde eine Dokumentation für eine neue Systeminformationsfunktion hinzufügt: PG_LAST_UNLOAD_ID. Weitere Informationen finden Sie unter PG_LAST_UNLOAD_ID.	27. Juni 2018
ALTER TABLE RENAME COLUMN	ALTER TABLE unterstützt jetzt das Umbenennen von Spalten für externe Tabellen. Weitere Informationen finden Sie unter Beispiele für das Ändern einer externen Tabelle.	7. Juni 2018

Frühere Updates

In der folgenden Tabelle sind die wichtigen Änderungen in jeder Version des Entwicklerhandbuchs zu Amazon Redshift Database vor Juni 2018 beschrieben.

Änderung	Beschreibung	Datum geändert
COPY aus Parquet umfasst SMALLINT	COPY unterstützt jetzt das Laden aus Parquet-formatierten Dateien in Spalten, die den SMALLINT-Datentyp verwenden. Weitere Informationen finden Sie unter COPY aus spaltenbasierten Datenformaten	2. Januar 2019

Änderung	Beschreibung	Datum geändert
COPY aus spaltenbasierten Formaten	COPY unterstützt jetzt das Laden von Dateien im spaltenbasierten Datenformat Parquet und ORC in Amazon S3. Weitere Informationen finden Sie unter COPY aus spaltenbasierten Datenformaten	17. Mai 2018
Dynamische maximale Laufzeit für SQA	Standardmäßig weist das Workload Management (WLM) nun dynamisch einen Wert für die maximale SQA-Laufzeit basierend auf der Analyse Ihres Cluster-Workloads zu. Weitere Informationen finden Sie unter Maximale Ausführungszeit für kurze Abfragen.	17. Mai 2018
Neue Spalte in STL_LOAD_COMMITS	Die Systemtabelle STL_LOAD_COMMITS hat eine neue Spalte, `file_format` .	10. Mai 2018
Neue Spalten in STL_HASHJOIN und anderen Systemprotokolltabellen	Die Systemtabelle STL_HASHJOIN hat drei neue Spalten `hash_segment` , `hash_step` und `checksum`. Darüber hinaus wurde eine `checksum` zu STL_MERGEJOIN, STL_NESTLOOP, STL_HASH, STL_SCAN, STL_SORT, STL_LIMIT und STL_PROJECT hinzugefügt.	17. Mai 2018
Neue Spalten in STL_AGGR	Die Systemtabelle STL_AGGR hat zwei neue Spalten: `resizes` und `flushable` .	19. April 2018
Neue Optionen für REGEX-Funktionen	Sie können nun für die Funktionen REGEXP_INSTR und REGEXP_SUBSTR die zu verwendende Übereinstimmung festlegen und angeben, ob bei der Übereinstimmung die Groß-/Kleinschreibung beachtet werden soll. Mit REGEXP_INSTR können Sie auch angeben, ob die Position des ersten Zeichens der Übereinstimmung oder die Position des ersten Zeichens nach dem Ende der Übereinstimmung zurückgegeben werden soll.	22. März 2018

Änderung	Beschreibung	Datum geändert
Neue Spalten in Systemtabellen	Die Spalten tombstonedblocks, tossedblocks und batched_by wurden zur STL_COMMIT_STATS Systemtabelle hinzugefügt. Die Spalte localslice wurde zur STV_SLICES Systemansicht hinzugefügt.	22. März 2018
Hinzufügen und entfernen von Spalten in externen Tabellen	ALTER TABLE unterstützt nun ADD COLUMN und DROP COLUMN für externe Amazon-Redshift-Spectrum-Tabellen.	22. März 2018
Redshift Spectrum – neue AWS-Regionen	Redshift Spectrum ist jetzt in den Regionen Mumbai und São Paulo verfügbar. Eine Liste der unterstützten -Regionen finden Sie unter Amazon-Redshift-Spectrum-Regionen.	22. März 2018
Tabellenlimits wurden auf 20.000 angehoben.	Die maximale Anzahl an Tabellen beträgt für den Clusterknotentyp "8xlarge" jetzt 20 000. Das Limit für die Knotentypen "large" und "xlarge" beträgt 9.900. Weitere Informationen finden Sie unter Limits und Kontingente.	13. März 2018
Redshift-Spectrum-Support für JSON und ION	Mithilfe von Redshift Spectrum können Sie auf Dateien mit skalaren Daten im JSON- oder ION-Format verweisen. Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.	26. Februar 2018
Verketten von IAM-Rollen für Redshift Spectrum	Sie können AWS Identity and Access Management (IAM-Rolle)-Rollen so verketten, dass Ihr Cluster andere, dem Cluster nicht angefügte Rollen annehmen kann, einschließlich Rollen, die zu einem anderen AWS-Konto gehören. Weitere Informationen finden Sie unter Verketten von IAM-Rollen in Amazon Redshift Spectrum.	1. Februar 2018

Änderung	Beschreibung	Datum geändert
ADD PARTITION unterstützt IF NOT EXISTS	Die ADD PARTITION-Klausel für ALTER TABLE unterstützt jetzt die Option IF NOT EXISTS. Weitere Informationen finden Sie unter ALTER TABLE.	11. Januar 2018
DATE-Daten für externe Tabellen	Externe Tabellen von Redshift Spectrum unterstützen jetzt den Datentyp DATE. Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.	11. Januar 2018
Redshift Spectrum – neue AWS-Regionen	Redshift Spectrum ist jetzt in den Regionen Singapur, Sydney, Seoul und Frankfurt verfügbar. Eine Liste der unterstützten AWS-Regionen finden Sie unter Amazon-Redshift-Spectrum-Regionen.	16. November 2017
Beschleunigungen kurzer Abfragen in Amazon Redshift Workload Management (WLM)	Wenn Sie Short Query Acceleration (SQA) verwenden , werden ausgewählte, kurze Abfragen gegenüber Abfragen mit einer höheren Dauer priorisiert. SQA führt kurze Abfragen an einer dedizierten Stelle aus, sodass SQA-Abfragen nicht hinter längeren Abfragen in Warteschlangen eingereiht werden. Mit SQA werden kürzere Abfragen schneller ausgeführt und der Benutzer sieht schneller Ergebnisse. Weitere Informationen finden Sie unter Arbeiten mit Short Query Acceleration.	16. November 2017
WLM weist gewechselte Abfragen neu zu	Statt eine gewechselte Abfrage abzubrechen und neu zu starten, weist Amazon Redshift Workload Management (WLM) berechtigte Abfragen jetzt einer neuen Warteschlange zu. Wenn WLM eine Abfrage neu zuweist, wird die Abfrage in die neue Warteschl ange verschoben und dort weiter ausgeführt. Dies spart Zeit und Systemressourcen. Gewechselte Abfragen, die nicht neu zugewiesen werden können, werden neu gestartet oder abgebrochen. Weitere Informationen finden Sie unter WLM-Abfragewartesc hlangen-Hopping.	16. November 2017

Änderung	Beschreibung	Datum geändert
Systempro tokollzugriff für Benutzer	In den meisten Systemprotokolltabellen, die für Benutzer sichtbar sind, sind von anderen Benutzern erstellte Zeilen für normale Benutzer standardmäßig nicht sichtbar. Um normalen Benutzern Zugriff auf alle Zeilen von für Benutzer sichtbaren Tabellen einschließlich der von anderen Benutzern generierten Zeilen zu gewähren, führen Sie ALTER USER oder CREATE USERaus und legen Sie den Parameter SYSLOG ACCESS auf UNRESTRICTED fest.	16. November 2017
Ergebnis-Zwischenspeicherung	Mit Ergebnis-Zwischenspeicherung wird das Ergebnis von Amazon Redshift zwischengespeichert, wenn Sie eine Abfrage ausführen. Wenn Sie die Abfrage erneut ausführen, sucht Amazon Redshift nach einer gültigen, zwischengespeicherten Kopie des Abfrageergebnisses. Wenn im Ergebnis-Cache ein passender Datensatz gefunden wird, verwendet Amazon Redshift das zwischengespeicherte Ergebnis und führt die Abfrage nicht aus. Das Zwischenspeichern des Ergebnisses ist standardmäßig aktiviert. Um die Zwischenspeicherung des Ergebnisses zu deaktivieren, legen Sie für den Konfigurationsparameter enable_result_cache_for_session die Option `off` fest.	16. November 2017
Spaltenmetadatenfunktionen	PG_GET_COLS und PG_GET_LATE_BINDING_VIEW_COLS geben Spaltenmetadaten für Amazon-Redshift-Tabellen, -Ansichten und Ansichten mit später Bindung zurück.	16. November 2017
WLM-Warteschlangen-Hopping für CTAS	Amazon Redshift Workload Management (WLM) unterstützt nun Abfragewarteschlangen-Hopping für CREATE TABLE AS (CTAS)-Anweisungen und schreibgeschützte Abfragen, beispielsweise SELECT-Anweisungen. Weitere Informationen finden Sie unter WLM-Abfragewarteschlangen-Hopping.	19. Oktober 2017

Änderung	Beschreibung	Datum geändert
Manifestdateien für Amazon Redshift Spectrum	Wenn Sie eine externe Redshift-Spectrum-Tabelle erstellen, können Sie eine Manifestdatei angeben, in der die Speicherorte der Datendateien in Amazon S3 aufgeführt sind. Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.	19. Oktober 2017
Amazon Redshift Spectrum – neue AWS-Regionen	Redshift Spectrum ist jetzt in den Regionen Europa (Irland) und Asien-Pazifik (Tokio) verfügbar. Eine Liste der unterstützten AWS-Regionen finden Sie unter Überlegungen zu Amazon Redshift Spectrum.	19. Oktober 2017
Amazon Redshift Spectrum hat Dateiformate hinzugefügt	Sie können nun externe Redshift Spectrum-Tabellen basierend auf den Datendateiformaten Regex, OpenCSV und Avro erstellen. Weitere Informationen finden Sie unter CREATE EXTERNAL TABLE.	5. Oktober 2017
Pseudospalten für externe Amazon-Redshift-Spectrum-Tabellen	Sie können die Pseudospalten `$path` und `$size` in einer externen Redshift-Spectrum-Tabelle auswählen , um den Speicherort und die Größe der referenzierten Datendateien in Amazon S3 anzuzeigen. Weitere Informationen finden Sie unter Pseudospalten.	5. Oktober 2017
Funktionen zur Validierung von JSON	Sie können mithilfe der Funktionen IS_VALID_JSON und IS_VALID_JSON_ARRAY die Formatierung von JSON überprüfen. Für die anderen JSON-Funktionen gibt es jetzt das optionale Argument `null_if_invalid` .	5. Oktober 2017
LISTAGG DISTINCT	Verwenden Sie die DISTINCT-Bedingung mit der Aggregationsfunktion LISTAGG und der Fensterfunktion LISTAGG, um duplizierte Werte aus dem angegebenen Ausdruck zu eliminieren, bevor die Werte verkettet werden.	5. Oktober 2017

Änderung	Beschreibung	Datum geändert
Anzeigen von Spaltennamen in Großbuchstaben	Wenn Sie Spaltennamen in SELECT-Ergebnissen in Großbuchstaben anzeigen möchten, legen Sie für den Konfigurationsparameter describe_field_name_in_uppercase den Wert `true` fest.	5. Oktober 2017
Überspringen von Headerzeilen in externen Tabellen	Verwenden Sie die Eigenschaft `skip.header.line.count` im Befehl CREATE EXTERNAL TABLE, um Headerzeilen am Anfang von Redshift Spectrum-Datendateien zu überspringen.	5. Oktober 2017
Anzahl der Scan-Zeilen	Die WLM-Abfragemonitorregeln verwenden die Metrik scan_row_count, um die Anzahl der Zeilen in einem Scanschritt zurückzugeben. Die Anzahl der Zeilen ist die Gesamtzahl der Zeilen, die nach der Filterung der zur Löschung markierten Zeilen (Geisterzeilen), jedoch vor der Anwendung benutzerdefinierter Abfragefilter ausgegeben wurden. Weitere Informationen finden Sie unter Abfrageüberwachungsmetriken für Amazon Redshift wurden bereitgestellt.	21. September 2017
Benutzerdefinierte SQL-Funktionen	Eine skalare benutzerdefinierte SQL-Funktion (User-Defined Function; UDF) enthält eine SQL-SELECT-Klausel, die ausgeführt wird, wenn die Funktion aufgerufen wird, und einen einzelnen Wert zurückgibt. Weitere Informationen finden Sie unter Erstellung einer skalaren SQL-UDF.	31. August 2017

Änderung	Beschreibung	Datum geändert
Ansichten mit später Bindung	Eine Ansicht mit später Bindung ist nicht an die zugrunde liegenden Datenbankobjekte gebunden, wie etwa Tabellen und benutzerdefinierte Funktione n. Deshalb besteht keine Abhängigkeit zwischen der Ansicht und den Objekten, auf die sie verweist. Sie können sogar eine Ansicht erstellen, wenn die referenzierten Objekte nicht vorhanden sind. Da keine Abhängigkeit besteht, können Sie ein referenzi ertes Objekt entfernen oder ändern, ohne die Ansicht zu beeinflussen. Amazon Redshift sucht erst nach Abhängigkeiten, wenn die Ansicht abgefragt wird. Zum Erstellen einer Ansicht mit später Bindung legen Sie die Klausel WITH NO SCHEMA BINDING mit Ihrer CREATE VIEW-Aussage fest. Weitere Informationen finden Sie unter CREATE VIEW.	31. August 2017
Die OCTET_LEN GTH-Funktion	OCTET_LENGTH gibt die Länge der angegebenen Zeichenfolge durch die Anzahl der Bytes an.	18. August 2017
Dateitypen ORC und Grok unterstüt zt	Amazon Redshift Spectrum unterstützt jetzt die Datenformate ORC und Grok für Redshift-Spectrum-Datendateien. Weitere Informationen finden Sie unter Erstellen von Datendateien für Abfragen in Amazon Redshift Spectrum.	18. August 2017
RegexSerDe wird jetzt unterstützt	Amazon Redshift Spectrum unterstützt jetzt das RegexSerDe Datenformat . Weitere Informationen finden Sie unter Erstellen von Datendateien für Abfragen in Amazon Redshift Spectrum.	19. Juli 2017

Änderung	Beschreibung	Datum geändert
Den Tabellen SVV_TABLES und SVV_COLUMNS wurden neue Spalten hinzugefügt	Die Spalten `domain_name` und `remarks` wurden zu SVV_COLUMNS hinzugefügt. Eine Spalte für Anmerkungen wurde zu hinzugefügt SVV_TABLES.	19. Juli 2017
SVV_TABLES- und SVV_COLUMNS-Systemansichten	Die Systemansichten SVV_TABLES und SVV_COLUMNS bieten Informationen zu Spalten und weitere Details zu lokalen und externen Tabellen und Ansichten.	7. Juli 2017
VPC für Amazon Redshift Spectrum mit Amazon EMR Hive Metastore nicht mehr erforderlich	Die Redshift-Spectrum-Anforderung, dass sich der Amazon-Redshift-Cluster und der Amazon-EMR-Cluster im selben VPC und Subnetz befinden müssen, wenn ein Amazon EMR Hive Metastore verwendet wird, wurde entfernt. Weitere Informationen finden Sie unter Arbeiten mit externen Katalogen in Amazon Redshift Spectrum.	7. Juli 2017
UNLOAD zu geringeren Dateigrößen	Standardmäßig erstellt UNLOAD mehrere Dateien auf Amazon S3 mit einer maximalen Dateigröße von 6,2 GB. Um kleinere Dateien zu erstellen, geben Sie den Parameter MAXFILESIZE mit dem UNLOAD-Befehl an. Sie können eine maximale Dateigröße zwischen 5 MB und 6,2 GB angeben. Weitere Informationen finden Sie unter UNLOAD.	7. Juli 2017
TABLE PROPERTIES	Sie können jetzt den Parameter TABLE PROPERTIES numRows für CREATE EXTERNAL TABLE oder ALTER TABLE einstellen, um die Tabellenstatistik so zu aktualisieren, dass die Anzahl der Zeilen in der Tabelle reflektiert wird.	6. Juni 2017

Änderung	Beschreibung	Datum geändert
ANALYZE PREDICATE COLUMNS	Um Zeit und Cluster-Ressourcen zu sparen, können Sie wählen, dass nur die Spalten analysiert werden, die wahrscheinlich als Prädikate verwendet werden. Wenn Sie ANALYZE mit der Klausel PREDICATE COLUMNS ausführen, berücksichtigt die Analyseop eration nur Spalten, die in einem Join, einer Filterbed ingung oder einer Gruppierungsklausel bzw. als Sortierungs- oder Verteilungsschlüssel verwendet wurden. Weitere Informationen finden Sie unter Analysieren von Tabellen.	25. Mai 2017
IAM-Richtlinien für Amazon Redshift Spectrum	Um den Zugriff auf einen Amazon-S3-Bucket nur mit der Verwendung von Redshift Spectrum zu gewähren, können Sie eine Bedingung einfügen, die den Zugriff für den Benutzeragenten gewährt `AWS Redshi ft/Spectrum` . Weitere Informationen finden Sie unter IAM-Richtlinien für Amazon Redshift Spectrum.	25. Mai 2017
Amazon Redshift Spectrum Recursive Scan	Redshift Spectrum scannt jetzt auch Dateien in Unterordnern sowie im angegebenen Ordner in Amazon S3. Weitere Informationen finden Sie unter Erstellen externer Tabellen für Redshift Spectrum.	25. Mai 2017
Abfrageüb erwachungsregeln	Mithilfe der WLM-Abfrageüberwachungsregeln definieren Sie auf Metriken basierende Leistungs grenzen für WLM-Warteschlangen und geben an, welche Aktion durchgeführt werden soll, wenn eine Abfrage diese Grenzwerte überschreitet (log, hop, abort). Sie definieren die Abfrageüberwachungsregeln im Rahmen Ihrer Workload Management (WLM)-Kon figuration. Weitere Informationen finden Sie unter WLM-Abfrageüberwachungsregeln.	21. April 2017

Änderung	Beschreibung	Datum geändert
Amazon Redshift Spectrum	Mit Redshift Spectrum können Sie effektiv Daten aus Dateien in Amazon S3 abfragen und abrufen, ohne die Daten in Tabellen laden zu müssen. Redshift-Spectrum-Abfragen werden auch für sehr große Datensätze sehr schnell durchgeführt, da Redshift Spectrum die Datendateien direkt in Amazon S3 scannt. Ein großer Teil der Verarbeitung findet auf der Amazon-Redshift-Spectrum-Ebene statt, und die meisten Daten bleiben in Amazon S3. Mehrere Cluster können den selben Datensatz auf Amazon S3 gleichzeitig abfragen, ohne dass Kopien der Daten für jeden Cluster erstellt werden müssen. Weitere Informationen finden Sie unter Abfrage externer Daten mit Amazon Redshift Spectrum	19. April 2017
neue Systemtabellen zur Unterstützung von Redshift Spectrum	Die folgenden neuen Systemansichten wurden zur Unterstützung von Redshift Spectrum hinzugefügt: • SVL_S3QUERY • SVL_S3QUERY_SUMMARY • SVV_EXTERNAL_COLUMNS • SVV_EXTERNAL_DATABASES • SVV_EXTERNAL_PARTITIONS • SVV_EXTERNAL_TABLES • PG_EXTERNAL_SCHEMA	19. April 2017

Änderung	Beschreibung	Datum geändert
Die Zusammenfassungsfunktion APPROXIMATE PERCENTILE_DISC	Die Zusammenfassungsfunktion APPROXIMATE PERCENTILE_DISC ist jetzt verfügbar.	4. April 2017
Serverseitige Verschlüsselung mit KMS	Sie können jetzt Daten zu Amazon S3 unter Verwendung einer serverseitigen Verschlüsselung mit einem AWS Key Management Service-Schlüssel (SSE-KMS) entladen. Zusätzlich lädt COPY jetzt KMS-verschlüsselte Datendateien aus Amazon S3 auf transparente Weise. Weitere Informationen finden Sie unter UNLOAD.	9. Februar 2017
Neue Autorisierungssyntax	Sie können jetzt die Parameter IAM_ROLE, MASTER_SYMMETRIC_KEY, ACCESS_KEY_ID, SECRET_ACCESS_KEY und SESSION_TOKEN verwenden, um Autorisierungs- und Zugriffsinformationen für die Befehle COPY, UNLOAD und CREATE LIBRARY bereitzustellen. Die neue Autorisierungssyntax bietet eine flexiblere Alternative zur Bereitstellung eines Einzelzeichenfolgenarguments für den Parameter CREDENTIALS. Weitere Informationen finden Sie unter Autorisierungsparameter.	9. Februar 2017
Erhöhung des Schema-Limits	Sie können jetzt bis zu 9.900 Schemata pro Cluster erstellen. Weitere Informationen finden Sie unter CREATE SCHEMA.	9. Februar 2017

Änderung	Beschreibung	Datum geändert
Standard-Tabellenkodierung	CREATE TABLE und ALTER TABLE weisen den meisten Spalten jetzt LZO-Kompressionskodierung zu. Als Sortierschlüssel definierten Spalten, Spalten mit den Datentypen BOOLEAN, REAL oder DOUBLE PRECISION und temporären Tabellen wird Standardmäßig die RAW-Kodierung zugewiesen. Weitere Informationen finden Sie unter ENCODE.	6. Februar 2017
ZSTD-Kompressionskodierung	Amazon Redshift unterstützt nun ZSTD-Spaltenkomprimierungskodierung.	19. Januar 2017
Zusammenfassungsfunktionen PERCENTILE_CONT und MEDIAN	PERCENTILE_CONT und MEDIAN stehen jetzt als Zusammenfassungsfunktionen und als Fensterfunktionen zur Verfügung.	19. Januar 2017
Benutzerdefinierte Funktion (UDF) Benutzerprotokollierung	Die können das Protokollierungsmodul von Python verwenden, um in Ihren UDFs benutzerdefinierte Fehler- und Warnmeldungen zu erstellen. Nach der Abfrageausführung können Sie die Systemansicht SVL_UDF_LOG abfragen, um protokollierte Meldungen abzurufen. Weitere Informationen zu benutzerdefinierten Meldungen finden Sie unter . Protokollieren von Fehlern und Warnungen in UDFs	8. Dezember 2016
Geschätzte Reduzierung bei ANALYZE COMPRESSION	Der Befehl ANALYZE COMPRESSION meldet jetzt eine Schätzung der prozentualen Reduzierung des Festplattenplatzes für jede Spalte. Weitere Informationen finden Sie unter ANALYZE COMPRESSION.	10. November 2016

Änderung	Beschreibung	Datum geändert
Verbindungsgrenzen	Sie können jetzt einen Grenzwert für die maximale Zahl von Datenbankverbindungen einrichten, die der Benutzer gleichzeitig geöffnet haben darf. Sie können auch die Anzahl der gleichzeitigen Verbindungen für eine Datenbank begrenzen. Weitere Informationen erhalten Sie unter CREATE USER und CREATE DATABASE.	10. November 2016
Erweiterung der COPY-Sortierreihenfolge	COPY fügt jetzt automatisch der sortierten Region einer Tabelle neue Zeilen hinzu, wenn Sie Ihre Daten in Sortierungsschlüsselreihenfolge laden. Für die spezifischen Anforderungen für die Aktivierung dieser Erweiterung vgl. Laden von Daten in der Reihenfolge des Sortierschlüssels	10. November 2016
CTAS mit Kompression	CREATE TABLE AS (CTAS) weist jetzt neuen Tabellen auf der Grundlage ihres jeweiligen Datentyps automatisch Kompressionskodierungen zu. Weitere Informationen finden Sie unter Vererbung von Spalten- und Tabellenattributen.	28. Oktober 2016
Zeitstempel beim Datentyp Zeitzone	Amazon Redshift unterstützt nun Zeitstempel mit Zeitzonen-Datentyp (TIMESTAMPTZ). Weiterhin wurden verschiedene neue Funktionen zur Unterstützung des neuen Datentyps hinzugefügt. Weitere Informationen finden Sie unter Datums- und Zeitfunktionen.	29. September 2016

Änderung	Beschreibung	Datum geändert
Analyse-Schwellenwert	Zur Reduzierung der Verarbeitungszeit und zur Verbesserung der allgemeinen Systemleistung für ANALYZE-Operationen überspringt Amazon Redshift die Analyse einer Tabelle, wenn der Prozentsatz der Zeilen, die seit der letzten Ausführung des Befehls ANALYZE geändert wurden, unter dem vom Parameter analyze_threshold_percent angegebenen Analyse-Schwellenwert liegt. Der Standardwert für `analyze_threshold_percent` ist 10.	9. August 2016
Neue Systemtabelle STL_RESTARTED_SESSIONS	Wenn Amazon Redshift eine Sitzung neu startet, zeichnet STL_RESTARTED_SESSIONS die neue Prozess-ID (PID) und die alte PID auf.	9. August 2016
Aktualisierung der Dokumentation für die Datum- und Uhrzeit-Funktionen	Hinzufügung einer Zusammenfassung der Funktionen mit Links zum Datums- und Zeitfunktionen sowie Aktualisierung der Funktionsreferenzen aus Konsistenzgründen.	24. Juni 2016
Neue Spalten in STL_CONNECTION_LOG	Die Systemtabelle STL_CONNECTION_LOG hat jetzt zwei neue Spalten zur Nachverfolgung der SSL-Verbindungen. Wenn Sie routinemäßig Audit-Protokolle in eine Amazon Redshift-Tabelle laden, müssen Sie der Zieltabelle die folgenden neuen Spalten hinzufügen: sslcompression und sslexpansion.	5. Mai 2016
MD5-Hash-Passwort	Sie können ein Passwort für einen CREATE USER- oder ALTER USER-Befehl hinzufügen, indem Sie die MD5-Hash-Zeichenfolge von Passwort und Benutzername angeben.	21. April 2016
Neue Spalte in STV_TBL_PERM	Die Spalte `backup` in der Systemansicht STV_TBL_PERM zeigt an, ob die Tabelle in Cluster-Snapshots eingeschlossen ist. Weitere Informationen finden Sie unter BACKUP.	21. April 2016

Änderung	Beschreibung	Datum geändert
Keine Sicherung stabellen	Für Tabellen wie beispielsweise Staging-Tabellen, die keine kritischen Daten enthalten, können Sie BACKUP NO in Ihrer CREATE TABLE- oder CREATE TABLE AS-Anweisung angeben, um zu verhindern, dass Amazon Redshift die Tabelle in automatisierte oder manuelle Snapshots einschließt. Die Verwendung von Tabellen ohne Sicherung spart Verarbeitungszeit bei der Erstellung von Snapshots sowie bei der Wiederher stellung von Snapshots und verringert die Speicherp latznutzung auf Amazon S3.	7. April 2016
VACUUM-Lö schschwellenwert	Standardmäßig gewinnt der Befehl VACUUM jetzt Festplattenplatz zurück, wenn mindestens 95 Prozent der verbleibenden Zeilen nicht zur Löschung markiert sind. Das Ergebnis ist, dass VACUUM normalerweise viel weniger Zeit für die Löschphase benötigt, als wenn 100 Prozent der gelöschten Zeilen zurückgewonnen werden müssten. Sie können den Standardschwellenw ert für eine einzelne Tabelle ändern, indem Sie den Parameter TO threshold PERCENT einschließen, wenn Sie den Befehl VACUUM ausführen.	7. April 2016
Systemtabelle SVV_TRANS ACTIONS	Die Systemansicht SVV_TRANSACTIONS zeichnet Informationen zu Transaktionen auf, die derzeit Tabellen in der Datenbank sperren.	7. April 2016

Änderung	Beschreibung	Datum geändert
Verwendung von IAM-Rollen zum Zugriff auf andere AWS-Ressourcen	Um Daten zwischen Ihrem Cluster und einer anderen AWS-Ressource, etwa Amazon S3, DynamoDB, Amazon EMR oder Amazon EC2 zu verschieben, muss Ihr Cluster über die Berechtigung zum Zugriff auf die Ressource und zur Durchführung der erforderlichen Aktionen verfügen. Als sicherere Alternative zur Bereitstellung eines Zugriffsschlüsselpaares mit den Befehlen COPY, UNLOAD oder CREATE LIBRARY können Sie jetzt eine IAM-Rolle angeben, die Ihr Cluster für Authentifizierung und Autorisierung verwendet. Weitere Informationen finden Sie unter Rollenbasierte Zugriffskontrolle.	29. März 2016
VACUUM-Sortierungsschwellenwert	Der Befehl VACUUM überspringt jetzt die Sortierungsphase für alle Tabellen, in denen mehr als 95 Prozent der Zeilen bereits sortiert sind. Sie können den Standard-Sortierungsschwellenwert für eine einzelne Tabelle ändern, indem Sie den Parameter TO threshold PERCENT einschließen, wenn Sie den Befehl VACUUM ausführen.	17. März 2016
Neue Spalten in STL_CONNECTION_LOG	Die Systemtabelle STL_CONNECTION_LOG hat drei neue Spalten. Wenn Sie routinemäßig Audit-Protokolle in eine Amazon-Redshift-Tabelle laden, müssen Sie der Zieltabelle die folgenden neuen Spalten hinzufügen: sslversion, sslcipher und mtu.	17. März 2016
UNLOAD mit bzip2-Kompression	Sie können jetzt UNLOAD mit bzip2-Kompression verwenden.	8. Februar 2016

Änderung	Beschreibung	Datum geändert
ALTER TABLE APPEND	ALTER TABLE APPEND fügt einer Zieltabelle Zeilen hinzu, indem Daten aus einer vorhandenen Quelltabelle verschoben werden. ALTER TABLE APPEND ist in der Regel sehr viel schneller als die vergleichbaren Operationen CREATE TABLE AS oder INSERT, da die Daten verschoben und nicht dupliziert werden.	8. Februar 2016
WLM-Abfragewartesch langen-Hopping	Wenn Workload Management (WLM) eine schreibgeschützte Abfrage (etwa eine SELECT-Anweisung) aufgrund eines WLM-Timeouts abbricht, versucht WLM, die Abfrage an die nächste passende Warteschlange weiterzuleiten. Weitere Informationen finden Sie unter WLM-Abfragewarteschlangen-Hopping	7. Januar 2016
ALTER DEFAULT PRIVILEGES	Sie können den Befehl ALTER DEFAULT PRIVILEGES verwenden, um den Standardsatz von Zugriffrechten zu definieren, die auf Objekte angewendet werden sollen, die vom angegebenen Benutzer in der Zukunft erstellt werden.	10. Dezember 2015
bzip2-Dateikompression	Der Befehl COPY unterstützt das Laden von Daten aus Dateien, die mit bzip2 komprimiert wurden.	10. Dezember 2015
NULLS FIRST und NULLS LAST	Sie können angeben, ob eine ORDER BY-Klausel NULLEN im Ergebnissatz zuerst oder zuletzt anordnen soll. Weitere Informationen erhalten Sie unter ORDER BY-Klausel und Übersicht über die Syntax von Fensterfunktionen.	19. November 2015
REGION-Schlüsselwort für CREATE LIBRARY	Wenn sich der Amazon-S3-Bucket, der die UDF-Bibliotheksdateien enthält, nicht in der gleichen AWS-Region wie Ihr Amazon-Redshift-Cluster befindet, können Sie die Option REGION verwenden, um die Region anzugeben, in der sich die Daten befinden. Weitere Informationen finden Sie unter CREATE LIBRARY.	19. November 2015

Änderung	Beschreibung	Datum geändert
Benutzerdefinierte skalare Funktionen (UDFs)	Sie können jetzt benutzerdefinierte skalare Funktionen erstellen, um Nicht-SQL-Verarbeitungsfunktionen zu implementieren, die entweder von Amazon Redshift unterstützten Modulen in der Python-2.7-Standardbibliothek bereitgestellt werden, oder von Ihren eigenen benutzerdefinierten UDF auf der Grundlage der Programmiersprache Python. Weitere Informationen finden Sie unter Erstellung benutzerdefinierter Funktionen.	11. September 2015
Dynamische Eigenschaften in der WLM-Konfiguration	Der WLM-Konfigurationsparameter unterstützt jetzt die dynamische Anwendung einiger Eigenschaften. Andere Eigenschaften bleiben weiterhin statische Änderungen und erfordern, dass die verbundenen Cluster neu gestartet werden, damit die Konfigurationsänderungen wirksam werden. Weitere Informationen erhalten Sie unter Dynamische und statische WLM-Konfigurationseigenschaften und Implementierung von Workload Management.	3. August 2015
Die Funktion LISTAGG	Die Funktion LISTAGG und Die Fensterfunktion LISTAGG geben eine Zeichenfolge zurück, die durch die Verkettung einer Reihe von Spaltenwerten entsteht.	30. Juli 2015
Veralteter Parameter	Der Konfigurationsparameter max_cursor_result_set_size ist veraltet. Die Größe der Cursor-Ergebnissätze wird jetzt auf der Grundlage des Knotentyps des Clusters beschränkt. Weitere Informationen finden Sie unter Einschränkungen für Cursors.	24. Juli 2015
Revision der Dokumentation des Befehls COPY	Die Referenz zu dem Befehl COPY wurde umfassend revidiert, um das Material benutzerfreundlicher und zugänglicher zu gestalten.	15. Juli 2015

Änderung	Beschreibung	Datum geändert
COPY von Avro-Format	Der Befehl COPY unterstützt das Laden von Daten im Avro-Format aus Datendateien in Amazon S3, Amazon EMR und Remote-Hosts mittels SSH. Weitere Informationen erhalten Sie unter AVRO und Beispiele für die Kopie aus Avro.	8. Juli 2015
STV_STARTUP_RECOVERY_STATE	Die STV_STARTUP_RECOVERY_STATE-Systemtabelle zeichnet den Status von Tabellen auf, die während Cluster-Neustartoperationen vorübergehend gesperrt sind. Amazon Redshift sperrt Tabellen vorübergehend, während sie bearbeitet werden, um angehaltene Transaktionen nach einem Cluster-Neustart aufzulösen.	25. Mai 2015
ORDER BY optional für Rangfunktionen	Die Klausel ORDER BY ist jetzt für bestimmte Fensterrangfunktionen optional. Weitere Informationen finden Sie unter CUME_DIST-Fensterfunktion, Die Fensterfunktion DENSE_RANK, Die Fensterfunktion RANK, Die Fensterfunktion NTILE, Die Fensterfunktion PERCENT_RANK und Die Fensterfunktion ROW_NUMBER.	25. Mai 2015
Überlappende Sortierungsschlüssel	Überlappende Sortierungsschlüssel geben allen Spalten in dem Sortierungsschlüssel das gleiche Gewicht. Die Verwendung überlappender Sortierungsschlüssel anstelle der Standard-Kompositionsschlüssel erhöht die Leistung von Abfragen deutlich, die restriktive Prädikate auf sekundären Sortierspalten verwenden, besonders für sehr große Tabellen. Die überlappende Sortierung verbessert darüber hinaus die allgemeine Leistung, wenn mehrere Abfragen Filter für verschiedene Spalten in der selben Tabelle verwenden. Weitere Informationen erhalten Sie unter Arbeiten mit Sortierschlüsseln und CREATE TABLE.	11. Mai 2015

Änderung	Beschreibung	Datum geändert
Überarbeitung des Themas „Optimieren der Abfrageleistung“	Optimieren der Abfrageleistung wurde erweitert und enthält jetzt neue Abfragen zur Analyse der Abfrageleistung sowie weitere Beispiele. Das Thema wurde darüber hinaus überarbeitet, um es klarer und umfassender zu gestalten. Bewährte Methoden für die Gestaltung von Abfragen mit Amazon Redshift enthält weitere Informationen zur Verbesserung der Leistung von Abfragen.	23. März 2015
SVL_QUERY_QUEUE_INFO	Die Ansicht SVL_QUERY_QUEUE_INFO fasst die Details für Abfragen zusammen, die in einer WLM-Abfragewarteschlange oder Commit-Warteschlange Zeit verbracht haben.	19. Februar 2015
SVV_TABLE_INFO	Sie können jetzt die Ansicht SVV_TABLE_INFO für die Diagnose und die Behandlung von Problemen mit dem Tabellendesign verwenden, die die Abfrageleistung beeinträchtigen können; dazu gehören auch Probleme mit der Kompressionskodierung, mit Verteilungsschlüsseln, dem Sortierungsstil, der Datenverteilungsverzögerung, der Tabellengröße und der Statistik.	19. Februar 2015
UNLOAD verwendet serverseitige Dateiverschlüsselung	Der Befehl UNLOAD verwendet nun automatisch serverseitige Verschlüsselung (SSE) in Amazon S3 zur Verschlüsselung aller entladener Datendateien. Die serverseitige Verschlüsselung fügt eine weitere Datensicherheitsebene hinzu, ohne dabei die Leistung deutlich zu beeinträchtigen.	31. Oktober 2014
CUME_DIST-Fensterfunktion	CUME_DIST-Fensterfunktion berechnet die kumulative Verteilung eines Wertes in einem Fenster oder einer Partition.	31. Oktober 2014

Änderung	Beschreibung	Datum geändert
Funktion MONTHS_BETWEEN	Funktion MONTHS_BETWEEN bestimmt die Anzahl der Monate zwischen zwei Daten.	31. Oktober 2014
Funktion NEXT_DAY	Funktion NEXT_DAY gibt das Datum der ersten Instance eines bestimmten Tages aus, der nach dem angegebenen Datum liegt.	31. Oktober 2014
Die Fensterfunktion PERCENT_RANK	Die Fensterfunktion PERCENT_RANK berechnet den prozentualen Rang einer bestimmten Zeile.	31. Oktober 2014
Die Fensterfunktion RATIO_TO_REPORT	Die Fensterfunktion RATIO_TO_REPORT berechnet das Verhältnis eines Wertes zur Summe der Werte in einem Fenster oder einer Partition.	31. Oktober 2014
Die Funktion TRANSLATE	Die Funktion TRANSLATE ersetzt alle Vorkommen bestimmter Zeichen in einem bestimmten Ausdruck durch angegebene Ersatzzeichen.	31. Oktober 2014
Funktion NVL2	Funktion NVL2 gibt einen von zwei Werten aus, je nachdem, ob ein angegebener Ausdruck zu NULL oder zu NOT NULL aufgelöst wird.	16. Oktober 2014
Die Fensterfunktion MEDIAN	Die Fensterfunktion MEDIAN berechnet den Medianwert für den Wertebereich in einem Fenster oder einer Partition.	16. Oktober 2014
ON ALL TABLES IN SCHEMA schema_name-Klausel für die Befehle GRANT und REVOKE	Die Befehle GRANT und REVOKE wurden mit einer ON ALL TABLES IN SCHEMA schema_name-Klausel aktualisiert. Diese Klausel ermöglicht die Verwendung eines einzelnen Befehls zum Ändern der Berechtigungen für alle Tabellen in einem Schema.	16. Oktober 2014

Änderung	Beschreibung	Datum geändert
IF EXISTS-Klausel für die Befehle DROP SCHEMA, DROP TABLE, DROP USER und DROP VIEW	Die Befehle DROP SCHEMA, DROP TABLE, DROP USER und DROP VIEW wurden mit einer IF EXISTS-Klausel aktualisiert. Diese Klausel führt dazu, dass der Befehl keine Änderungen vornimmt und eine Meldung ausgibt, anstatt mit einem Fehler abzubrechen, wenn das angegebene Objekt nicht existiert.	16. Oktober 2014
IF NOT EXISTS-Klausel für die Befehle CREATE SCHEMA und CREATE TABLE	Die Befehle CREATE SCHEMA und CREATE TABLE wurden mit einer IF NOT EXISTS-Klausel aktualisi ert. Diese Klausel führt dazu, dass der Befehl keine Änderungen vornimmt und eine Meldung ausgibt, anstatt mit einem Fehler abzubrechen, wenn das angegebene Objekt bereits existiert.	16. Oktober 2014
COPY-Unte rstützung für UTF-16-Kodierung	Der COPY-Befehl unterstützt jetzt das Laden aus Datendateien, die UTF-16 oder die UTF-8-Kodierung verwenden. Weitere Informationen finden Sie unter ENCODING.	29. September 2014
Neues Tutorial zum Workload-Management	Tutorial: Konfigurieren von manuellen Workload-Management(WLM)-Warteschlangen erläutert den Prozess der Konfiguration von Workload Managemen t (WLM)-Warteschlangen zur Verbesserung der Abfrageverarbeitung und der Ressourcenzuweisung.	25. September 2014
AES 128-Bit-V erschlüsselung	Der COPY-Befehl unterstützt jetzt die AES 128-Bit- und die AES 256-Bit-Verschlüsselung beim Laden aus mit clientseitiger Amazon-S3-Verschlüsselung verschlüsselten Datendateien. Weitere Informationen finden Sie unter Laden verschlüsselter Datendateien aus Amazon S3.	29. September 2014
Funktion PG_LAST_U NLOAD_COUNT	Die Funktion PG_LAST_UNLOAD_COUNT gibt die Anzahl der Zeilen aus, die in der letzten UNLOAD-Operation verarbeitet wurden. Weitere Informationen finden Sie unter PG_LAST_UNLOAD_COUNT.	15. September 2014

Änderung	Beschreibung	Datum geändert
Neuer Abschnitt zur Fehlerbehebung bei Abfragen	Fehlerbehebung bei Abfragen bietet eine kurze Übersicht über die Identifizierung und Behandlung einiger der häufigsten und schwerwiegendsten Probleme, die bei Amazon-Redshift-Abfragen auftreten können.	7. Juli 2014
Neues Tutorial zum Laden von Daten	Tutorial: So laden Sie Daten aus Amazon S3 führt Sie durch den kompletten Prozess des Ladens von Daten in Ihre Amazon-Redshift-Datenbanktabellen aus Datendateien in einem Amazon-S3-Bucket.	1. Juli 2014
Fensterfunktion PERCENTILE_CONT	Fensterfunktion PERCENTILE_CONT ist eine Funktion für die inverse Verteilung, die ein kontinuierliches Verteilungsmodell annimmt. Sie empfängt einen Perzentilwert und eine Sortierspezifikation und gibt einen interpolierten Wert zurück, der in Bezug auf die Sortierspezifikation in den angegebenen Perzentilwert fällt.	30. Juni 2014
Die Fensterfunktion PERCENTILE_DISC	Die Fensterfunktion PERCENTILE_DISC ist eine Funktion für die inverse Verteilung, die ein diskretes Verteilungsmodell annimmt. Sie empfängt einen Perzentilwert und eine Sortierspezifikation und gibt ein Element aus dem Satz zurück.	30. Juni 2014
Funktionen GREATEST und LEAST	Die Funktionen Funktionen GREATEST und LEAST geben den größten oder den kleinsten Wert aus einer Liste von Ausdrücken aus.	30. Juni 2014
Regionsübergreifedner COPY-Befehl	Der Befehl COPY unterstützt das Laden von Daten aus Amazon-S3-Buckets oder Amazon-DynamoDB-Tabellen, die sich nicht in der gleichen Region wie der Amazon-Redshift-Cluster befinden. Weitere Informationen finden Sie unter REGION in der Referenz zum COPY-Befehl.	30. Juni 2014

Änderung	Beschreibung	Datum geändert
Erweiterung der Bewährten Verfahren	Bewährte Methoden für Amazon Redshift wurde erweitert, umorganisiert und an die Spitze der Navigationshierarchie verschoben, um diesen Abschnitt leichter auffindbar zu machen.	28. Mai 2014
UNLOAD zu einer einzelnen Datei	Der Befehl UNLOAD kann optional Tabellendaten seriell zu einer einzelnen Datei in Amazon S3 entladen, indem die Option PARALLEL OFF hinzugefügt wird. Wenn der Umfang der Daten die maximale Dateigröße von 6,2 GB überschreitet, erstellt UNLOAD zusätzliche Dateien.	6. Mai 2014
REGEXP-Funktionen	Die Funktionen REGEXP_COUNT, REGEXP_INSTR und REGEXP_REPLACE manipulieren Zeichenfolgen auf der Grundlage der Musterzuordnung regulärer Ausdrücke.	6. Mai 2014
COPY aus Amazon EMR	Der Befehl COPY unterstützt das direkte Laden von Daten aus Amazon-EMR-Clustern. Weitere Informationen finden Sie unter So laden Sie Daten aus Amazon EMR:.	18. April 2014
Erhöhung der WLM-Gleichzeitigkeitsbeschränkung	Sie können jetzt Workload Management (WLM) so konfigurieren, dass in benutzerdefinierten Abfragewarteschlangen bis zu 50 Abfragen gleichzeitig ausgeführt werden. Diese Erhöhung sorgt für mehr Flexibilität bei der Verwaltung der Systemleistung durch die Modifizierung von WLM-Konfigurationen. Weitere Informationen finden Sie unter Implementieren von manuellem WLM	18. April 2014

Änderung	Beschreibung	Datum geändert
Neuer Konfigurationsparameter für die Verwaltung der Cursorgröße	Der Konfigurationsparameter `max_cursor_result_set_size` definiert den maximalen Umfang der Daten, in Megabyte, die pro Cursor-Ergebnissatz bei einer größeren Abfrage ausgegeben werden können. Dieser Parameterwert beeinflusst auch die Anzahl der gleichzeitigen Cursors für den Cluster, so dass Sie einen Wert konfigurieren können, der die Anzahl der Cursors für Ihren Cluster erhöht oder verringert. Weitere Informationen finden Sie unter DECLARE in diesem Leitfaden und unter Konfigurieren der maximalen Größe eines Cursor-Ergebnissatzes im Amazon-Redshift-Verwaltungshandbuch.	28. März 2014
COPY von JSON-Format	Der Befehl COPY unterstützt das Laden von Daten im JSON-Format aus Datendateien in Amazon S3 sowie aus Remote-Hosts mittels SSH. Weitere Informationen finden Sie in den COPY von JSON-Format-Anwendungshinweisen.	25. März 2014
Neue Systemtabelle STL_PLAN_INFO	Die Tabelle STL_PLAN_INFO ergänzt den Befehl EXPLAIN als weitere Möglichkeit zur Betrachtung von Abfrageplänen.	25. März 2014
Neue Funktion REGEXP_SUBSTR	Die Funktion REGEXP_SUBSTR gibt die durch die Suche nach einem regulären Ausdrucksmuster aus einer Zeichenfolge extrahierten Zeichen aus.	25. März 2014
Neue Spalten für STL_COMMIT_STATS	Die Tabelle STL_COMMIT_STATS hat zwei neue Spalten: `numxids` und `oldestxid` .	6. März 2014
Unterstützung für COPY von SSH für gzip und lzop	Der Befehl COPY unterstützt die gzip- und die lzop-Kompression beim Laden von Daten über eine SSH-Verbindung.	13. Februar 2014

Änderung	Beschreibung	Datum geändert
Neue Funktionen	Die Fensterfunktion ROW_NUMBER gibt die Nummer der aktuellen Zeile aus. Die Funktion STRTOL konvertiert den Zeichenfolgenausdruck einer Zahl der angegebenen Basis in den entsprechenden Ganzzahlwert. PG_CANCEL_BACKEND und PG_TERMINATE_BACKEND ermöglichen Benutzern das Abbrechen von Abfragen und Sitzungsverbindung en. Die Funktion LAST_DAY wurde hinzugefügt, um die Kompatibilität mit Oracle zu gewährleisten.	13. Februar 2014
Neue Systemtab elle	Die Systemtabelle STL_COMMIT_STATS bietet Metriken zur Commit-Leistung, einschließlich des Timings der verschiedenen Commit-Phasen und der Anzahl der Commit-Blöcke.	13. Februar 2014
FETCH mit Einzelknoten-Clust ern	Bei der Verwendung eines Cursors auf einem Einzelknoten-Cluster ist die maximale Anzahl von Zeilen, die mit dem Befehl FETCH abgerufen werden können, 1000. FETCH FORWARD ALL wird für Einzelknoten-Cluster nicht unterstützt.	13. Februar 2014
DS_DIST_A LL_INNER-Umverteilungsstrat egie	DS_DIST_ALL_INNER in der Explain plan-Ausgabe zeigt an, dass die gesamte innere Tabelle neu zu einem einzelnen Slice verteilt wurde, da die äußere Tabelle DISTSTYLE ALL verwendet. Weitere Informati onen erhalten Sie unter Beispiele für Join-Varianten und Auswerten des Abfrageplans.	13. Januar 2014
Neue Systemtab ellen für Abfragen	Amazon Redshift hat neue Systemtabellen hinzugefü gt, mit denen Benutzer die Abfrageausführung für Optimierungs- und Fehlerbehebungszwecke evaluieren können. Weitere Informationen finden Sie unter SVL_COMPILE, STL_SCAN, STL_RETURN, STL_SAVE STL_ALERT_EVENT_LOG.	13. Januar 2014

Änderung	Beschreibung	Datum geändert
Einzelnoten-Cursors	Cursors werden jetzt auch für Einzelknoten-Cluster unterstützt. Ein Einzelknoten-Cluster kann jetzt zwei Cursors gleichzeitig geöffnet haben, bei einem maximalen Ergebnissatz von 32 GB. Auf einem Einzelknoten-Cluster empfehlen wir die Einstellung des Parameters für die ODBC-Cache-Größe auf 1.000. Weitere Informationen finden Sie unter DECLARE.	13. Dezember 2013
Verteilungsstil ALL	Die ALL-Verteilung kann die Ausführungszeiten für bestimmte Arten von Abfragen dramatisch verkürzen. Wenn eine Tabelle den Verteilungsstil ALL verwendet, wird eine Kopie der Tabelle auf jeden Knoten verteilt. Da die Tabelle effektiv mit allen anderen Tabellen zusammen platziert wird, ist keine Neuverteilung während der Ausführung der Abfrage erforderlich. Die ALL-Verteilung ist nicht für alle Tabellen geeignet, da sie die Speicherplatzanforderungen und die Ladezeit erhöht. Weitere Informationen finden Sie unter Arbeiten mit Datenverteilungsstilen.	11. November 2013
COPY von Remote-Hosts	Zusätzlich zum Laden von Tabellen aus Datendateien in Amazon S3 und aus Amazon DynamoDB-Tabellen kann der Befehl COPY Textdaten aus Amazon-EMR-Clustern, Amazon-EC2-Instances und anderen Remote-Hosts über SSH-Verbindungen laden. Amazon Redshift verwendet mehrere gleichzeitige SSH-Verbindungen, um Daten parallel zu lesen und zu laden. Weitere Informationen finden Sie unter Laden von Daten aus Remote-Hosts.	11. November 2013

Änderung	Beschreibung	Datum geändert
Verwendeter WLM-Speicherplatz in Prozent	Sie können den Workload gleichmäßig verteilen, indem Sie einen bestimmten Prozentsatz des Speichers für jede Warteschlange in Ihrer Workload Management (WLM)-Konfiguration anweisen. Weitere Informationen finden Sie unter Implementieren von manuellem WLM.	11. November 2013
APPROXIMATE COUNT(DISTINCT)	Abfragen, die APPROXIMATE COUNT(DISTINCT) werden viel schneller ausgeführt, bei einer relativen Fehlerquote von etwa 2 %. Die Funktion APPROXIMATE COUNT(DISTINCT) verwendet einen HyperLogLog Algorithmus. Weitere Informationen hierzu finden Sie unter Die Funktion COUNT.	11. November 2013
Neue SQL-Funktionen für den Abruf von Details zu aktuellen Abfragen	Vier neue SQL-Funktionen rufen Details zu aktuellen Abfragen und COPY-Befehlen ab. Die neuen Funktionen erleichtert die Abfrage von Systemprotokolltabellen und bieten in vielen Fällen die benötigten Details, ohne dass ein Zugriff auf die Systemtabellen erforderlich ist. Weitere Informationen finden Sie unter PG_BACKEND_PID, PG_LAST_COPY_ID, PG_LAST_COPY_COUNT, PG_LAST_QUERY_ID.	1. November 2013
MANIFEST-Option für UNLOAD	Die MANIFEST-Option für den Befehl UNLOAD ergänzt die MANIFEST-Option für den Befehl COPY. Die Verwendung der MANIFEST-Option mit UNLOAD erstellt automatisch eine Manifest-Datei, die die Datendateien ausführlich auflistet, die von der Entladeoperation auf Amazon S3 erstellt wurden. Sie können dann diese Manifestdatei mit einem COPY-Befehl zum Entladen der Daten verwenden. Weitere Informationen erhalten Sie unter Entladen von Daten aus Amazon S3 und UNLOAD-Beispiele.	1. November 2013

Änderung	Beschreibung	Datum geändert
MANIFEST-Option für COPY	Sie können die Option MANIFEST mit dem Befehl COPY verwenden, um die Datendateien explizit aufzulisten, die aus Amazon S3 geladen werden sollen.	18. Oktober 2013
Systemtabellen für die Behebung von Fehlern für Abfragen	Zusätzliche Dokumentation für Systemtabellen, die für die Fehlerbehebung bei Abfragen verwendet werden. Der Abschnitt STL-Ansichten für die Protokollierung enthält jetzt Dokumentation zu den folgenden Systemtabellen: STL_AGGR, STL_BCAST , STL_DIST, STL_DELETE, STL_HASH, STL_HASHJ OIN, STL_INSERT, STL_LIMIT, STL_MERGE, STL_MERGEJOIN, STL_NESTLOOP, STL_PARSE , STL_PROJECT, STL_SCAN, STL_SORT, STL_UNIQUE, STL_WINDOW.	3. Oktober 2013
Funktion CONVERT_T IMEZONE	Funktion CONVERT_TIMEZONE konvertiert einen Zeitstempel von einer Zeitzone zu einer anderen, mit der Option für die automatische Anpassung der Sommerzeit.	3. Oktober 2013
Die Funktion SPLIT_PART	Die Funktion SPLIT_PART trennt eine Zeichenfolge am angegebenen Trennzeichen und gibt den Teil an der angegebenen Position aus.	3. Oktober 2013
Systemtabelle STL_USERLOG	STL_USERLOG zeichnet Einzelheiten für Änderunge n auf, die auftreten, wenn ein Datenbankbenutzer erstellt, geändert oder gelöscht wird.	3. Oktober 2013
LZO-Spalt enkodierung und LZOP-Date ikompression.	LZODie -Spaltenkompression kombiniert ein sehr hohes Kompressionsverhältnis mit hoher Leistung. COPY aus Amazon S3 unterstützt das Laden aus Dateien, die mittels LZOP-Komprimierung komprimiert wurden.	19. September 2013

Änderung	Beschreibung	Datum geändert
JSON, reguläre Ausdrücke und Cursors	Zusätzliche Unterstützung für das Parsing von JSON-Zeichenfolgen, die Musterzuordnung mit regulären Ausdrücken sowie die Verwendung von Cursors zum Abruf großer Datensätze über eine ODBC-Verbindung. Weitere Informationen finden Sie unter JSON-Funktionen, Patternmatching-Bedingungen und DECLARE.	10. September 2013
ACCEPTINVCHAR-Option für COPY	Sie können Daten erfolgreich laden, die ungültige UTF-8-Zeichen enthalten, indem Sie die Option ACCEPTINVCHAR mit dem Befehl COPY angeben.	29. August 2013
CSV-Option für COPY	Der Befehl COPY unterstützt jetzt das Laden aus CSV-formatierten Eingabedateien.	9. August 2013
CRC32	Die Funktion CRC32 führt zyklische Redundanz prüfungen durch.	9. August 2013
WLM-Platzhalter	Workload Management (WLM) unterstützt Platzhalter zum Hinzufügen von Benutzergruppen und Abfragegruppen zu Warteschlangen. Weitere Informationen finden Sie unter Platzhalter.	1. August 2013
WLM-Timeout	Um die Zeit zu begrenzen, die Abfragen in einer WLM-Warteschlange nutzen können, können Sie den WLM-Timeout-Wert für jede Warteschlange einstellen. Weitere Informationen finden Sie unter WLM-Timeout.	1. August 2013
Neue COPY-Optionen „auto“ und „epochsecs“	Der Befehl COPY führt die automatische Erkennung von Datums- und Uhrzeitformaten durch. Die neuen Uhrzeitformate „epochsecs“ und „epochmillisecs“ ermöglichen COPY das Laden der Daten im Epochenformat.	25. Juli 2013
Funktion CONVERT_TIMEZONE	Funktion CONVERT_TIMEZONE konvertiert einen Zeitstempel von einer Zeitzone zu einer anderen.	25. Juli 2013

Änderung	Beschreibung	Datum geändert
Funktion FUNC_SHA1	Funktion FUNC_SHA1 konvertiert eine Zeichenfolge mit dem SHA1-Algorithmus.	15. Juli 2013
max_execution_time	Zur Begrenzung der Zeit, die Abfragen in Anspruch nehmen dürfen, können Sie den Parameter max_execution_time im Rahmen der WLM-Konfiguration einstellen. Weitere Informationen finden Sie unter Modifizieren der WLM-Konfiguration.	22. Juli 2013
Vier-Byte-UTF-8-Zeichen	Der Datentyp VARCHAR unterstützt jetzt Vier-Byte-UTF-8-Zeichen. Fünf-Byte- oder längere UTF-8-Zeichen werden nicht unterstützt. Weitere Informationen finden Sie unter Speicherung und Bereiche.	18. Juli 2013
SVL_QERROR	Die Systemansicht SVL_QERROR wurde außer Betrieb genommen.	12. Juli 2013
Revidierter Dokumentverlauf	Die Seite „Dokumentverlauf“ zeigt jetzt das Datum der Aktualisierung des Dokuments.	12. Juli 2013
STL_UNLOAD_LOG	STL_UNLOAD_LOG zeichnet die Details für eine Entladeoperation auf.	5. Juli 2013
Parameter JDBC-Abrufgröße	Um Speicherplatzfehler auf der Clientseite beim Abruf von großen Datensätzen mit JDBC zu vermeiden, können Sie Ihren Client so einrichten, dass er die Daten in Batches abruft; dazu dient der Parameter JDBC-Abrufgröße. Weitere Informationen finden Sie unter Festlegen des JDBC-Parameters für die Abrufgröße.	27. Juni 2013
UNLOAD für verschlüsselte Dateien	UNLOAD unterstützt nun das Entladen von Tabellendaten zu verschlüsselten Dateien in Amazon S3.	22. Mai 2013

Änderung	Beschreibung	Datum geändert
Temporäre Anmeldeinformationen	COPY und UNLOAD unterstützen jetzt die Verwendung temporärer Anmeldeinformationen.	11. April 2013
Hinzugefügte Klarstellungen	Klarstellung und Erweiterung der Besprechung des Tabellendesigns und des Ladens von Daten.	14. Februar 2013
Hinzufügung bewährter Methoden	Bewährte Methoden für die Gestaltung von Tabellen mit Amazon Redshift und Bewährte Methoden für Amazon Redshift zum Laden von Daten hinzugefügt.	14. Februar 2013
Klarstellung von Passworteinschränkungen	Klarstellung von Passworteinschränkungen für CREATE USER und ALTER USER, verschiedene kleinere Revisionen.	14. Februar 2013
Neues Handbuch	Dies ist die erste Veröffentlichung des Amazon-Redshift-Entwicklerhandbuchs.	14. Februar 2013

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.