# Table of Contents

- What Is Amazon Redshift? ................................................................. 1
- Are you a first-time Amazon Redshift user? ........................................... 1
- Amazon Redshift management overview ................................................ 1
  - Cluster management ............................................................................. 2
  - Cluster access and security ................................................................. 2
  - Monitoring clusters ............................................................................. 3
  - Databases .......................................................................................... 4
- Amazon Redshift Serverless (preview) .................................................... 5
  - What is Amazon Redshift Serverless? .................................................. 5
    - Overview of Amazon Redshift Serverless features ............................... 6
    - Comparing Amazon Redshift Serverless to a provisioned cluster .......... 7
    - Amazon Redshift Serverless console .................................................. 8
    - Known issues for Amazon Redshift Serverless .................................... 13
    - Quotas and limits for Amazon Redshift Serverless ............................... 13
    - Capacity for Amazon Redshift Serverless ........................................... 14
    - Billing for Amazon Redshift Serverless .............................................. 14
  - Get started with Amazon Redshift Serverless ........................................ 17
    - Permissions required to use Amazon Redshift Serverless ..................... 17
  - Connecting to Amazon Redshift Serverless ........................................... 18
    - Connecting to serverless endpoint ...................................................... 18
    - Connecting to Amazon Redshift Serverless through JDBC drivers .......... 18
    - Connecting to the serverless endpoint with the Data API ....................... 19
    - Connecting with SSL to the serverless endpoint .................................... 21
    - Connecting to Amazon Redshift Serverless from other VPC endpoints ...... 21
  - Working with Amazon Redshift query editor v2 and the serverless endpoint ........................................................................ 23
  - Data sharing in Amazon Redshift Serverless ........................................... 24
    - Data sharing in Amazon Redshift Serverless ........................................ 24
    - Security and connections in Amazon Redshift Serverless ..................... 24
    - Identity and access management in Amazon Redshift Serverless ........... 25
  - Monitoring queries and workload with Amazon Redshift Serverless .......... 26
    - Granting access to monitor queries ..................................................... 26
    - Monitoring views ............................................................................... 28
  - Working with snapshots and recovery points ......................................... 29
    - Working with snapshots and recovery points ....................................... 29
  - Audit logging for Amazon Redshift Serverless ........................................ 31
    - Exporting logs .................................................................................. 31
- Clusters .............................................................................................. 34
  - Overview of Amazon Redshift clusters ................................................ 34
    - Clusters and nodes ............................................................................ 34
  - Use EC2-VPC when you create your cluster ........................................... 38
    - EC2-VPC .......................................................................................... 38
    - EC2-Classic ....................................................................................... 38
    - Launch a cluster ............................................................................... 39
  - Overview of RA3 node types ............................................................... 39
    - Working with Amazon Redshift managed storage ................................. 40
    - Managing RA3 node types ................................................................. 40
    - RA3 node type availability in AWS Regions ........................................ 40
  - Upgrading to RA3 node types ............................................................. 41
    - Upgrade DS2 reserved nodes to RA3 reserved nodes during elastic resize or snapshot restore ..... 43
    - Upgrading from DC1 node types to DC2 node types ............................... 44
    - Upgrading a DS2 cluster on EC2-Classic to EC2-VPC ......................... 45
    - Region and Availability Zone considerations ....................................... 45
  - Cluster maintenance ........................................................................... 45
    - Maintenance windows ....................................................................... 46
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting up the Amazon Redshift CLI</td>
<td>453</td>
</tr>
<tr>
<td>Installation instructions</td>
<td>453</td>
</tr>
<tr>
<td>Getting started with the AWS Command Line Interface</td>
<td>454</td>
</tr>
<tr>
<td>Monitoring cluster performance</td>
<td>458</td>
</tr>
<tr>
<td>Overview</td>
<td>458</td>
</tr>
<tr>
<td>Performance data</td>
<td>459</td>
</tr>
<tr>
<td>Amazon Redshift metrics</td>
<td>459</td>
</tr>
<tr>
<td>Dimensions for Amazon Redshift metrics</td>
<td>464</td>
</tr>
<tr>
<td>Amazon Redshift query and load performance data</td>
<td>465</td>
</tr>
<tr>
<td>Working with performance data</td>
<td>466</td>
</tr>
<tr>
<td>Viewing cluster performance data</td>
<td>467</td>
</tr>
<tr>
<td>Viewing query history data</td>
<td>473</td>
</tr>
<tr>
<td>Viewing database performance data</td>
<td>476</td>
</tr>
<tr>
<td>Viewing workload concurrency and concurrency scaling data</td>
<td>479</td>
</tr>
<tr>
<td>Viewing queries and loads</td>
<td>481</td>
</tr>
<tr>
<td>Viewing cluster metrics during load operations</td>
<td>485</td>
</tr>
<tr>
<td>Analyzing workload performance</td>
<td>485</td>
</tr>
<tr>
<td>Managing alarms</td>
<td>486</td>
</tr>
<tr>
<td>Working with performance metrics in the CloudWatch console</td>
<td>487</td>
</tr>
<tr>
<td>Events</td>
<td>489</td>
</tr>
<tr>
<td>Overview</td>
<td>489</td>
</tr>
<tr>
<td>Viewing events using the console</td>
<td>489</td>
</tr>
<tr>
<td>Viewing events using the AWS SDK for Java</td>
<td>489</td>
</tr>
<tr>
<td>Viewing events using the AWS CLI and Amazon Redshift API</td>
<td>491</td>
</tr>
<tr>
<td>Event notifications</td>
<td>491</td>
</tr>
<tr>
<td>Overview</td>
<td>491</td>
</tr>
<tr>
<td>Amazon Redshift event categories and event messages</td>
<td>493</td>
</tr>
<tr>
<td>Managing event notifications using the Amazon Redshift console</td>
<td>503</td>
</tr>
<tr>
<td>Managing event notifications using the AWS CLI and Amazon Redshift API</td>
<td>504</td>
</tr>
<tr>
<td>Quotas and limits</td>
<td>505</td>
</tr>
<tr>
<td>Amazon Redshift quotas</td>
<td>505</td>
</tr>
<tr>
<td>Amazon Redshift Spectrum quotas and limits</td>
<td>509</td>
</tr>
<tr>
<td>Naming constraints</td>
<td>510</td>
</tr>
<tr>
<td>Tagging</td>
<td>512</td>
</tr>
<tr>
<td>Tagging overview</td>
<td>512</td>
</tr>
<tr>
<td>Tagging requirements</td>
<td>513</td>
</tr>
<tr>
<td>Managing resource tags using the console</td>
<td>513</td>
</tr>
<tr>
<td>Managing tags using the Amazon Redshift API</td>
<td>513</td>
</tr>
<tr>
<td>Document history</td>
<td>515</td>
</tr>
</tbody>
</table>
What is Amazon Redshift?

Welcome to the Amazon Redshift Cluster Management Guide. Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. You can start with just a few hundred gigabytes of data and scale to a petabyte or more. This enables you to use your data to acquire new insights for your business and customers.

The first step to create a data warehouse is to launch a set of nodes, called an Amazon Redshift cluster. After you provision your cluster, you can upload your data set and then perform data analysis queries. Regardless of the size of the data set, Amazon Redshift offers fast query performance using the same SQL-based tools and business intelligence applications that you use today.

Are you a first-time Amazon Redshift user?

If you are a first-time user of Amazon Redshift, we recommend that you begin by reading the following sections:

- Amazon Redshift management overview (p. 1) – This topic provides an overview of Amazon Redshift.
- Service Highlights and Pricing – This product detail page provides the Amazon Redshift value proposition, service highlights, and pricing.
- Amazon Redshift Getting Started Guide – This guide walks you through the process of creating a cluster, creating database tables, uploading data, and testing queries.
- Amazon Redshift Cluster Management Guide (this guide) – This guide shows you how to create and manage Amazon Redshift clusters.
- Amazon Redshift Database Developer Guide – If you are a database developer, this guide explains how to design, build, query, and maintain the databases that make up your data warehouse.

There are several ways to manage clusters. If you prefer a more interactive way of managing clusters, you can use the Amazon Redshift console or the AWS Command Line Interface (AWS CLI). If you are an application developer, you can use the Amazon Redshift Query API or the AWS Software Development Kit (SDK) libraries to manage clusters programmatically. If you use the Amazon Redshift Query API, you must authenticate every HTTP or HTTPS request to the API by signing it. For more information about signing requests, go to Signing an HTTP request (p. 451).

For information about the CLI, API, and SDKs, go to the following links:

- AWS CLI Command Reference
- Amazon Redshift API Reference
- SDK References in Tools for Amazon Web Services.

Amazon Redshift management overview

The Amazon Redshift service manages all of the work of setting up, operating, and scaling a data warehouse. These tasks include provisioning capacity, monitoring and backing up the cluster, and applying patches and upgrades to the Amazon Redshift engine.
Cluster management

An Amazon Redshift cluster is a set of nodes, which consists of a leader node and one or more compute nodes. The type and number of compute nodes that you need depends on the size of your data, the number of queries you will run, and the query execution performance that you need.

Creating and managing clusters

Depending on your data warehousing needs, you can start with a small, single-node cluster and easily scale up to a larger, multi-node cluster as your requirements change. You can add or remove compute nodes to the cluster without any interruption to the service. For more information, see Amazon Redshift clusters (p. 34).

Reserving compute nodes

If you intend to keep your cluster running for a year or longer, you can save money by reserving compute nodes for a one-year or three-year period. Reserving compute nodes offers significant savings compared to the hourly rates that you pay when you provision compute nodes on demand. For more information, see Purchasing Amazon Redshift reserved nodes (p. 312).

Creating cluster snapshots

Snapshots are point-in-time backups of a cluster. There are two types of snapshots: automated and manual. Amazon Redshift stores these snapshots internally in Amazon Simple Storage Service (Amazon S3) by using an encrypted Secure Sockets Layer (SSL) connection. If you need to restore from a snapshot, Amazon Redshift creates a new cluster and imports data from the snapshot that you specify. For more information about snapshots, see Amazon Redshift snapshots (p. 292).

Cluster access and security

There are several features related to cluster access and security in Amazon Redshift. These features help you to control access to your cluster, define connectivity rules, and encrypt data and connections. These features are in addition to features related to database access and security in Amazon Redshift. For more information about database security, see Managing Database Security in the Amazon Redshift Database Developer Guide.

AWS accounts and IAM credentials

By default, an Amazon Redshift cluster is only accessible to the AWS account that creates the cluster. The cluster is locked down so that no one else has access. Within your AWS account, you use the AWS Identity and Access Management (IAM) service to create user accounts and manage permissions for those accounts to control cluster operations. For more information, see Security in Amazon Redshift (p. 321).

Security groups

By default, any cluster that you create is closed to everyone. IAM credentials only control access to the Amazon Redshift API-related resources: the Amazon Redshift console, command line interface (CLI), API, and SDK. To enable access to the cluster from SQL client tools via JDBC or ODBC, you use security groups:

- If you are using the EC2-VPC platform for your Amazon Redshift cluster, you must use VPC security groups. We recommend that you launch your cluster in an EC2-VPC platform.

You cannot move a cluster to a VPC after it has been launched with EC2-Classic. However, you can restore an EC2-Classic snapshot to an EC2-VPC cluster using the Amazon Redshift console. For more information, see Restoring a cluster from a snapshot (p. 303).
• If you are using the EC2-Classic platform for your Amazon Redshift cluster, you must use Amazon Redshift security groups.

In either case, you add rules to the security group to grant explicit inbound access to a specific range of CIDR/IP addresses or to an Amazon Elastic Compute Cloud (Amazon EC2) security group if your SQL client runs on an Amazon EC2 instance. For more information, see Amazon Redshift cluster security groups (p. 438).

In addition to the inbound access rules, you create database users to provide credentials to authenticate to the database within the cluster itself. For more information, see Databases (p. 4) in this topic.

Encryption

When you provision the cluster, you can optionally choose to encrypt the cluster for additional security. When you enable encryption, Amazon Redshift stores all data in user-created tables in an encrypted format. You can use AWS Key Management Service (AWS KMS) to manage your Amazon Redshift encryption keys.

Encryption is an immutable property of the cluster. The only way to switch from an encrypted cluster to a cluster that is not encrypted is to unload the data and reload it into a new cluster. Encryption applies to the cluster and any backups. When you restore a cluster from an encrypted snapshot, the new cluster is encrypted as well.

For more information about encryption, keys, and hardware security modules, see Amazon Redshift database encryption (p. 323).

SSL connections

You can use Secure Sockets Layer (SSL) encryption to encrypt the connection between your SQL client and your cluster. For more information, see Configuring security options for connections (p. 193).

Monitoring clusters

There are several features related to monitoring in Amazon Redshift. You can use database audit logging to generate activity logs, configure events and notification subscriptions to track information of interest. Use the metrics in Amazon Redshift and Amazon CloudWatch to learn about the health and performance of your clusters and databases.

Database audit logging

You can use the database audit logging feature to track information about authentication attempts, connections, disconnections, changes to database user definitions, and queries run in the database. This information is useful for security and troubleshooting purposes in Amazon Redshift. The logs are stored in Amazon S3 buckets. For more information, see Database audit logging (p. 423).

Events and notifications

Amazon Redshift tracks events and retains information about them for a period of several weeks in your AWS account. For each event, Amazon Redshift reports information such as the date the event occurred, a description, the event source (for example, a cluster, a parameter group, or a snapshot), and the source ID. You can create Amazon Redshift event notification subscriptions that specify a set of event filters. When an event occurs that matches the filter criteria, Amazon Redshift uses Amazon Simple Notification Service to actively inform you that the event has occurred. For more information about events and notifications, see Amazon Redshift events (p. 489).
Performance

Amazon Redshift provides performance metrics and data so that you can track the health and performance of your clusters and databases. Amazon Redshift uses Amazon CloudWatch metrics to monitor the physical aspects of the cluster, such as CPU utilization, latency, and throughput. Amazon Redshift also provides query and load performance data to help you monitor the database activity in your cluster. For more information about performance metrics and monitoring, see Monitoring Amazon Redshift cluster performance (p. 458).

Databases

Amazon Redshift creates one database when you provision a cluster. This is the database you use to load data and run queries on your data. You can create additional databases as needed by running a SQL command. For more information about creating additional databases, go to Step 1: Create a database in the Amazon Redshift Database Developer Guide.

When you provision a cluster, you specify an admin user who has access to all of the databases that are created within the cluster. This admin user is a superuser who is the only user with access to the database initially, though this user can create additional superusers and users. For more information, go to Superusers and Users in the Amazon Redshift Database Developer Guide.

Amazon Redshift uses parameter groups to define the behavior of all databases in a cluster, such as date presentation style and floating-point precision. If you don't specify a parameter group when you provision your cluster, Amazon Redshift associates a default parameter group with the cluster. For more information, see Amazon Redshift parameter groups (p. 271).

For more information about databases in Amazon Redshift, go to the Amazon Redshift Database Developer Guide.
Amazon Redshift Serverless (preview)

This is prerelease documentation for Amazon Redshift Serverless, which is in preview release. The documentation and the feature are both subject to change. We recommend that you use this feature only in test environments, and not in production environments. For preview terms and conditions, see Beta Service Participation in AWS Service Terms.

Amazon Redshift Serverless makes it convenient for you to run and scale analytics without having to provision and manage data warehouse clusters. With Amazon Redshift Serverless, data analysts, developers, and data scientists can now use Amazon Redshift to get insights from data in seconds by loading and querying data into the data warehouse. Amazon Redshift automatically provisions and scales data warehouse capacity to deliver fast performance for demanding and unpredictable workloads. You only pay for the capacity that you use. You can benefit from this simplicity without changing your existing analytics and business intelligence applications.

Topics
- What is Amazon Redshift Serverless? (p. 5)
- Get started with Amazon Redshift Serverless (p. 17)
- Connecting to Amazon Redshift Serverless (p. 18)
- Working with Amazon Redshift query editor v2 and the serverless endpoint (p. 23)
- Data sharing in Amazon Redshift Serverless (p. 24)
- Security and connections in Amazon Redshift Serverless (p. 24)
- Monitoring queries and workload with Amazon Redshift Serverless (p. 26)
- Working with snapshots and recovery points (p. 29)
- Audit logging for Amazon Redshift Serverless (p. 31)

What is Amazon Redshift Serverless?

Amazon Redshift Serverless automatically provisions data warehouse capacity and intelligently scales the underlying resources. The serverless endpoint adjusts capacity in seconds to deliver consistently high performance and simplified operations for even the most demanding and volatile workloads.

With the serverless endpoint, you can benefit from the following features:
- Access and analyze data without the need to set up, tune, and manage Amazon Redshift provisioned clusters.
- Use the superior Amazon Redshift SQL capabilities, industry-leading performance, and lake house architecture to seamlessly query across a data warehouse, a data lake, and operational data sources.
- Deliver consistently high performance and simplified operations for even the most demanding and volatile workloads with intelligent and automatic scaling in seconds.
- Pay only when the data warehouse is in use.
With the serverless endpoint, you use a console interface to reach a serverless data warehouse. Through the data warehouse, you can access your Amazon Redshift managed storage and your Amazon S3 data lake.

In the following sections, you can learn the basics of Amazon Redshift Serverless.

If you have any questions about this preview, contact the Amazon Redshift Serverless team by email at redshift-preview-serverless@amazon.com. For service issues, contact AWS Support.

Overview of Amazon Redshift Serverless features

Most of the features supported by an Amazon Redshift provisioned cluster are also supported on a serverless endpoint. The following lists some of the key Amazon Redshift capabilities you can use with a serverless endpoint.

- Snapshots – you can restore a snapshot of a serverless endpoint or a provisioned cluster to your serverless endpoint. For more information, see Working with snapshots and recovery points (p. 29).
- Recovery points – Amazon Redshift Serverless automatically creates a point of recovery every 30 minutes. These recovery points are kept for 24 hours. You can use them to restore your serverless endpoint after accidental writes or deletes. When you restore from a recovery point all the data in the databases of your serverless endpoint is restored to an earlier point in time. You can also create a snapshot from a recovery point if you need to keep a point of recovery for a longer period. For more information, see Working with snapshots and recovery points (p. 29).
- Base RPU capacity – You can set a base capacity in Redshift Processing Units (RPUs). One RPU provides 16 GiBs of memory. Amazon Redshift uses this measurement to limit the resources (thus cost) used for your workload. You can increase this value to improve query performance. The default is 128 RPUs.
- Usage limits of cross-Region data sharing – You can limit the amount of data that is transferred from a producer Region to a consumer Region. Data transfer costs differ by AWS Region.
- User-defined functions (UDFs) – you can run user-defined functions (UDFs) on your serverless endpoint. For more information, see Creating user-defined functions in the Amazon Redshift Database Developer Guide.
- Stored procedures – you can run stored procedures on your serverless endpoint. For more information, see Creating stored procedures in the Amazon Redshift Database Developer Guide.
- Materialized views – you can create materialized views on your serverless endpoint. For more information, see Creating materialized views in the Amazon Redshift Database Developer Guide.
- Spatial functions – you can run spatial functions on your serverless endpoint. For more information, see Querying spatial data in the Amazon Redshift Database Developer Guide.
- Federated queries – you can run queries to join data with Aurora and Amazon RDS databases from with your serverless endpoint. For more information, see Querying data with federated queries in the Amazon Redshift Database Developer Guide.
- Data lake queries – you can run queries to join data with your Amazon S3 data lake with your serverless endpoint.
- HyperLogLog – you can run HyperLogLog functions on your serverless endpoint. For more information, see Using HyperLogLog sketches in the Amazon Redshift Database Developer Guide.
- Querying data across databases – you can query data across databases on your serverless endpoint. For more information, see Querying data across databases in the Amazon Redshift Database Developer Guide.
- Data sharing – you can access datashares on provisioned clusters with your serverless endpoint. For more information, see Sharing data across clusters in the Amazon Redshift Database Developer Guide.
- Semistructured data querying – you can ingest and store semistructured data with the SUPER data type on your serverless endpoint. For more information, see Ingesting and querying semistructured data in the Amazon Redshift Database Developer Guide.
• Amazon Redshift machine learning – you can use Amazon Redshift machine learning with your serverless endpoint. For more information, see Using machine learning in the Amazon Redshift Database Developer Guide.

• SQL commands and functions – with a few exceptions (such as REBOOT_CLUSTER), you can use Amazon Redshift SQL commands and functions on your serverless endpoint. For more information, see SQL reference in the Amazon Redshift Database Developer Guide.

Comparing Amazon Redshift Serverless to a provisioned cluster

The following list describes features and behavior in Amazon Redshift Serverless and explains how they differ from a provisioned cluster.

• **Node types** - When you work with a serverless endpoint, you don't configure or interact with specific nodes or node types, like you do with a provisioned Redshift cluster, where you can resize and choose a specific node type. For your serverless endpoint, you can control RPU capacity, for example, when you want your endpoint to handle additional query load. This includes configuration settings for base capacity and maximum RPU hours. For more information, see Understanding Amazon Redshift Serverless capacity (p. 14).

• **Workload management and concurrency scaling** - With a provisioned cluster, you enable concurrency scaling on your cluster to handle periods of heavy load. With a serverless endpoint, concurrency scaling is turned on automatically and occurs without any configuration.

• **Port** - With a provisioned cluster, you can choose any port to connect. With a serverless endpoint, your endpoint port, 5439, is static.

• **Resizing** - With a provisioned cluster, you perform an elastic resize to add nodes or to upgrade node types. You can also perform a classic resize if elastic resize isn't available for the type of resize you want to do. With a serverless endpoint, you choose a higher or lower RPU level in the console to specify your compute capacity. You can scale up or down. You don't have to resize a cluster to change capacity. It's simply a setting on the AWS console.

• **Pausing and resuming** - With a provisioned cluster, you pause and resume your cluster manually, based on an assessment of your workload at various times. With a serverless endpoint, compute resources are deprovisioned automatically after a workload is run. Note that pausing or resuming a serverless endpoint has no bearing on billing or cost, because cost accrues according to when queries are processed, rather than when compute resources are active or idle.

• **Spectrum queries** - With a provisioned cluster, Amazon Redshift Spectrum capacity exists on separate servers that are queried from the Amazon Redshift cluster. Billing accrues if compute resources aren't idle. With a serverless endpoint, billing accrues as Redshift Spectrum data is queried, like any other transaction. (Where data is queried from doesn't affect billing. The metric is whether compute resources are processing SQL workloads.)

• **Compute-resource billing** - With a provisioned cluster, billing occurs per second when compute resources aren't idle. Specifically, when cluster nodes aren't paused. With a serverless endpoint, you are charged on a per-second basis as transactions are run. When transactions aren't running, there is no charge.

• **Maintenance window** - With a provisioned cluster, you specify a maintenance window when patching occurs on your cluster. (Typically, you choose a recurring time when use is low.) With a serverless endpoint, there is no maintenance window. The system handles patching automatically without disrupting workloads.

• **Encryption** - A provisioned cluster's data can be encrypted or unencrypted with AWS KMS (with AWS managed or customer managed keys), or HSM. A serverless endpoint is always encrypted with AWS KMS, with AWS managed or customer managed keys.

• **Storage billing** - For both a provisioned cluster and a serverless endpoint, storage is billed apart from compute resources at a rate of GB per month.
• **User management** - For both a provisioned cluster and a serverless endpoint, users are IAM or Redshift users. For information regarding setting up users and roles, see Overview of managing access permissions to your Amazon Redshift resources (p. 335).

• **Encryption** - A provisioned cluster's data can be encrypted or unencrypted with AWS KMS (AWS managed or customer managed keys), or HSM. A serverless endpoint is always encrypted with AWS KMS, with AWS managed or customer managed keys.

• **JDBC and ODBC tools and compatibility** - Both provisioned clusters and Amazon Redshift Serverless are compatible with any JDBC or ODBC compliant tool or client application.

• **Requirement for credentials on sign in** - Access to Amazon Redshift requires user name and password credentials, from an IAM user with specific permissions, for a provisioned cluster. (You can also associate a user to an IAM role.) Once authenticated, the user can connect seamlessly to the database, to the Redshift console, and to query editor v2. For a serverless endpoint, credentials are optional. For more information, see Connecting to serverless endpoint (p. 18).

• **Data API** - Both provisioned clusters and Amazon Redshift Serverless support the Amazon Redshift Data API. The only difference is that with a serverless endpoint you don't include the `cluster-identity` parameter value when you run a command. For more information about calling the Data API, see Using the Amazon Redshift Data API (p. 238).

• **Snapshots** - Provisioned clusters support both system and user snapshots. Amazon Redshift Serverless supports snapshots and recovery points. For more information about snapshots and recovery points for a serverless endpoint, see Working with snapshots and recovery points (p. 29).

• **Data Sharing** - Provisioned clusters support cross database, cross account, cross-region, and AWS Data Exchange data sharing. Amazon Redshift Serverless supports all of the data sharing possibilities that provisioned clusters do, and it also supports data sharing between a serverless endpoint and a provisioned cluster, tool, or client application.

• **Tracks** - Provisioned clusters support switching between current and trailing tracks. Amazon Redshift Serverless has no concept of a track.

• **System tables and views** - Provisioned clusters support the existing set of system tables and views for cluster monitoring and other tasks that require system metadata. Amazon Redshift Serverless supports new system tables and views. For more information about system tables, see Monitoring views (p. 28).

• **Parameter groups** - Provisioned clusters support parameter groups. Amazon Redshift Serverless does not have the concept of a parameter group. For more information about parameter groups for a provisioned cluster, see Amazon Redshift parameter groups (p. 271).

• **Query monitoring** - Query monitoring in provisioned clusters does not show all data in system tables, because it depends on a different API. Query monitoring for a serverless endpoint requires users to connect to the database to use system tables. Thus, query monitoring and system tables are in sync. Queries of system tables for a serverless endpoint use the database user mapped to the IAM user for using query monitoring. For additional operators, the administrator must manually grant the `sys:operator` role to the database user. For more information, see Monitoring queries and workload with Amazon Redshift Serverless (p. 26).

# Amazon Redshift Serverless console

The Amazon Redshift Serverless console navigation menu contains the following pages and links:

• **Serverless dashboard** to see a summary of your resources and activity.

• **Query editor** link to open the Amazon Redshift query editor v2 to manage and query the data in your serverless endpoint. The query editor v2 is an SQL client where you can run queries and create and load databases.

• **Serverless configuration** to update your serverless endpoint settings.

• **Query and database monitoring** to review and analyze your query activity.

• **Resource monitoring** to review your capacity and compute usage.
• **Datashares** to manage account level data sharing. This page is where you can manage the datashares available to the serverless endpoint. For more information, see Data sharing in Amazon Redshift Serverless (p. 24).

• **Provisioned clusters dashboard** to see a summary of your provisioned clusters and open the Amazon Redshift console.

• **Documentation** link to open the documentation landing page.

On the **Serverless dashboard** page, you can view a summary of your resources and graphs of your usage.

• **Resource summary** – This section shows the number of databases and snapshots in your serverless endpoint.

• **Query summary** – This section shows query activity for the last one hour.

• **RPU capacity used** – This section shows capacity used for the last one hour.

• **Datashares** – This section shows the number of datashares in this account or from another account that are available or require authorization.

On the **Serverless configuration** page, you can view your serverless endpoint environment settings. This page displays information about your serverless endpoint settings such as **General information, Data backup, Data access, and Limits**.

The **General information** section displays the following:

• **Serverless namespace** – Is an identifier of your serverless endpoint.

• **Date created** – Is the date and time that your serverless endpoint was created.

• **Status** – Your serverless endpoint must be available to query or change some configuration settings.

• **Change admin password** – You can change the password of the Admin user. For more information on changing the admin password, on the console, see Configuring the serverless endpoint using the console (p. 12).

• **Database name** – Is the name of the initial database created with the serverless endpoint. This name can be used in the connection string when you connect to the serverless endpoint.

• **Serverless credit remaining** – Is the amount of credit remaining for the account.

• **Endpoint** – Is the serverless endpoint used for some connections.

• **JDBC URL** – Is the connection string to connect from JDBC tools.

• **ODBC URL** – Is the connection string to connect from ODBC tools.

On the **Data backup** tab you can work with the following:

• **Snapshots** – You can create, delete, and manage snapshots of your serverless endpoint data. Currently, the retention period for snapshots is indefinitely. You can authorize AWS accounts for serverless to restore from a specific snapshot.

• **Recovery points** – Displays the recovery points that are automatically created so you can recover from an accidental write or delete within the last 24 hours. You can create a snapshot from a recovery point if you want to keep a point of recovery for a longer time period. Currently, the retention period for snapshots is indefinitely.

On the **Data access** tab you can work with the following:

• **Network and security** settings – You can view VPC-related values, AWS KMS encryption values, and audit logging values. You can update only audit logging. For more information on setting network and security settings on the console, see Configuring the serverless endpoint using the console (p. 12).
• **AWS KMS key** – The AWS KMS key used to encrypt resources in the serverless endpoint.

• **Permissions** – You can manage the IAM roles that Amazon Redshift Serverless can assume to use resources on your behalf. For more information, see Identity and access management in Amazon Redshift Serverless (p. 25).

• **Redshift-managed VPC endpoints** – You can access your serverless endpoint from another VPC or subnet. For more information, see Connecting to the Amazon Redshift serverless endpoint from a Redshift-managed VPC endpoint (p. 22).

On the **Limits** tab you can work with the following:

• **Base capacity in Redshift processing units (RPUs)** settings – You can set the base capacity used to process your workload. To improve query performance, increase your RPU value.

• **Usage limits** – The maximum compute resources that your serverless endpoint can use in a time period before an action is initiated. You limit the amount of resource your serverless endpoint uses to run your workload. Usage is measured in Redshift Processing Units (RPUs) seconds. An RPU second is the number of RPUs used in one second. You determine an action when a threshold that you set is reached, as follows:
  • Send an alert.
  • Log an entry to a system table.
  • Turn off user queries.

For more information, see Understanding Amazon Redshift Serverless capacity (p. 14).

On the **Datashares** tab you can work with the following:

• **Datashares created in my namespace** settings – You can create a datashare and share it with other namespaces and AWS accounts.

• **Datashares from other namespaces and AWS accounts** – You can create a database from a datashare from other namespace and AWS accounts.

For more information about data sharing, see Data sharing in Amazon Redshift Serverless (p. 24).

On the **Query and database monitoring** page, you can view graphs of your **Query history** and **Database performance**. You can filter the data based on several dimensions.

On the **Query history** tab, you see the following graphs (you can choose between **Query list** and **Resource metrics**):

• **Query runtime** – This graph shows which queries are running in the same timeframe. Choose a bar in the graph to view more query execution details.

• **Queries and loads** – This section lists queries and loads by **Query ID**.

• **RPU capacity used** – This graph shows overall capacity in Redshift processing units (RPUs).

• **Database connections** – This graph shows the number of active database connections.

On the **Database performance** tab, you see the following graphs:

• **Queries completed per second** – This graph shows the average number of queries completed per second.

• **Queries duration** – This graph shows the average amount of time to complete a query.

• **Database connections** – This graph shows the number of active database connections.

• **Running queries** – This graph shows the total number of running queries at a given time.
• **Queued queries** – This graph shows the total number of queries queued at a given time.

• **Query run time breakdown** – This graph shows the total time queries spent running by query type.

On the **Resource monitoring** page, you can view graphs of your consumed resources. You can filter the data based on several facets.

• **RPU capacity used** – This graph shows the overall capacity in Redshift processing units (RPUs).

• **Compute usage** – This graph shows the accumulative usage of Amazon Redshift Serverless by period for the selected time range.

On the **Datashares** page, you can manage datashares **In my account** and **From other accounts**. For more information about data sharing, see Data sharing in Amazon Redshift Serverless (p. 24).

### Setting up the serverless endpoint using the console

The first time you select the **Serverless dashboard**, in the Redshift console, you walk through the steps to set up your serverless endpoint. Under **Get started with Amazon Redshift Serverless**, complete the following steps.

1. If you have never had a serverless endpoint, you can choose to receive a serverless credit. Select **Choose starter base configuration** if you want to use a configuration with a lower base RPU level, which conserves resources used and extends the life of the credit.

2. Under **Configuration**, choose either **Use default settings** or **Customize settings**, to specify your own. If you choose default settings, the following are set to default values:

   • **Database name and password**
   • **Admin user credentials**
   • **Network and security settings**
   • **Permissions**

   If you choose the default settings, skip to the last step in this procedure to save. If you choose **Customize settings**, complete the remaining steps prior to saving.

3. Specify the database name. You can optionally select **Customize admin user credentials** if you don’t want to use the default IAM admin credentials, per your requirements. For more information, see Identity and access management in Amazon Redshift Serverless (p. 25).

4. Specify settings for **Network and security**. These include the following:

   • **Virtual private cloud (VPC)** - The VPC to contain the serverless endpoint.
   • **VPC security groups** - These security groups define which subnets and IP ranges can be used in the VPC.
   • **Subnet** - The subnets in the VPC that is associated with the specified database.
   • The AWS-owned KMS key is used by default to encrypt your data. Instead of using the AWS-owned KMS key, you can **Customize encryption settings** to choose a **KMS key** you manage – The AWS KMS key is used to encrypt resources in the serverless endpoint.
   • **Audit logging** - You choose logs to export for audit logging. Audit logging helps with monitoring and troubleshooting.

5. For **Permissions**, you can create an IAM role to associate with the serverless endpoint, or associate your own existing IAM role. The IAM role you associate with your serverless endpoint must include a trust relationship with redshift-serverless.amazonaws.com and redshift.amazonaws.com. For more information, see Permissions required to use Amazon Redshift Serverless (p. 17).

6. Choose **Save configuration** to complete setup.
Configuring the serverless endpoint using the console

Use **Serverless configuration** to update settings for your serverless endpoint.

Access to particular console settings depends on your IAM role and the permissions that you have assigned.

Change your admin password

1. Choose **Change admin password**. A dialog appears.
2. You can specify a **New admin username** and a **New admin user password**.
3. Choose **Save**.

Edit network and security settings

1. Choose the **Data access** tab. Choose **Edit** on the **Network and security** panel. A dialog appears.
2. For **Audit logging**, choose the logs to export. Each log type specifies different metadata.
3. To complete the configuration update, choose **Save changes**.

Checking the serverless endpoint using the dashboard

The Amazon Redshift Serverless dashboard contains a collection of panels that show at a glance metrics and other information about the serverless endpoint. These panels include the following:

- **Resources summary** - Displays high-level information about the serverless endpoint, such as the count of databases, the storage used, and other metrics.
- **Query summary** - Displays information about queries, including completed queries and running queries. Choose **View details** to go to a screen that has additional filters.
- **RPU capacity used** - Displays the overall capacity used over a given time period, like the previous ten hours, for instance.
- **Datashares** - Shows the count of datashares, which are used to share data between, for example, AWS accounts. The metrics show which datashares require authorization, and other information.

From the dashboard you can quickly dive into these available metrics to check a detail regarding your serverless endpoint, or review queries, or track work items.

Filtering queries

You can use the filters available on the serverless dashboard. To filter queries, perform the following steps.

1. On the left of the **Query summary** panel, select the drop-down list to filter by completed queries, failed queries, or both.
2. On the right of the **Query summary** panel, select the drop-down list to filter by running queries, queued queries, or both.

Deleting the serverless endpoint using the console

Complete the following steps:

1. On **Serverless configuration**, choose **Actions** and select **Delete serverless endpoint**.
2. A dialogue box opens. When you choose to delete the serverless endpoint, all recovery points are deleted, but manual snapshots are not deleted. Access to data in datashares is removed for consumers.

Type `delete` and select **Delete serverless endpoint** to confirm.

After you complete the steps, the status of the endpoint is `Deleting` and the banner indicates that the endpoint is being deleted. While the delete process is in progress, tabs under **Serverless configuration** are disabled. But you can configure provisioned clusters on the **Provisioned clusters dashboard**.

### Known issues for Amazon Redshift Serverless

The following issues are open:

- After a long period of inactivity using Amazon Redshift query editor v2, running a query can show a spinning progress wheel without a query result pane appearing. Refreshing the web browser can clear this condition.
- Canceling multiple queries in the query editor v2 can sometimes result in a spinning progress wheel. Refreshing the web browser can clear this condition.
- You can have only one serverless endpoint for each AWS account.
- Public endpoints aren't supported.

### Quotas and limits for Amazon Redshift Serverless

#### Limits for Amazon Redshift Serverless objects

Amazon Redshift has quotas that limit the use of several object types in your serverless endpoint. There is a default value for each.

<table>
<thead>
<tr>
<th>Quota name</th>
<th>AWS default value</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of databases</td>
<td>100</td>
<td>No</td>
<td>The maximum allowed count of databases for a serverless endpoint.</td>
</tr>
<tr>
<td>Number of schemas</td>
<td>9,900</td>
<td>No</td>
<td>The maximum allowed count of schemas for a serverless endpoint.</td>
</tr>
<tr>
<td>Number of tables</td>
<td>100,000</td>
<td>No</td>
<td>The maximum allowed count of tables for a serverless endpoint.</td>
</tr>
</tbody>
</table>

#### Cursor constraints

<table>
<thead>
<tr>
<th>Endpoint type</th>
<th>Maximum result set (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Redshift Serverless endpoint</td>
<td>150000</td>
</tr>
</tbody>
</table>

To see a table that lists result-set size specifications for a provisioned cluster, see **DECLARE**.
Additional constraints

*Maintenance window* - There is no maintenance window with Amazon Redshift Serverless. Software version updates are automatically applied. Any ongoing connections are dropped at the point in time when Amazon Redshift switches versions. Clients need to reestablish connections and Amazon Redshift Serverless works instantly.

*Availability zones* - A serverless endpoint is supported only in Availability Zones with the following IDs:

- use1-az2
- use1-az4
- use1-az6
- use2-az1
- use2-az2

When you configure your serverless endpoint, open Additional considerations, and make sure that the subnet IDs provided in Subnet contain at least one of the supported Availability Zone IDs. To see the subnet to Availability Zone ID mapping, go to the VPC console and choose Subnets to see the list of subnet IDs with their Availability Zone IDs. Verify that your subnet is mapped to a supported Availability Zone ID. To create a subnet, see Create a subnet in your VPC in the Amazon VPC User Guide.

Capacity for Amazon Redshift Serverless

Understanding Amazon Redshift Serverless capacity

RPUs

Amazon Redshift Serverless measures data warehouse compute capacity in Redshift Processing Units, or RPUs. Each RPU, which is a unit of compute resources, includes 16 GB of memory. A serverless endpoint that's allocated more resources has a higher number of RPUs.

Base capacity

Each serverless endpoint has a *base capacity* setting. It's specified in RPUs, and represents the minimum data warehouse capacity that Amazon Redshift Serverless uses to process queries. Base capacity can be adjusted down or up, based on your performance and cost requirements. Setting a higher base capacity can improve query performance, especially for data processing and ETL jobs that process large amounts of data. Setting a higher base capacity also results in higher cost for the serverless endpoint.

The default base capacity for a serverless endpoint is 128 RPUs. You can adjust the *Base capacity* setting from 32 RPUs to 512 RPUs in units of 8 (32,40,48...512) on the serverless endpoint management console.

Billing for Amazon Redshift Serverless

Understanding Amazon Redshift Serverless billing

Billing for compute capacity

*Base capacity*

When queries run, you're billed according to RPUs used in a time interval. Specifically, RPUs per second. When no queries are running, you aren't billed. You aren't charged for idle resources. Your *Base capacity* setting establishes a minimum per-second billing amount for the serverless endpoint, which accrues as
the endpoint processes database queries. Amazon Redshift Serverless scales above the base capacity to handle spikes in the number of queries.

**Maximum RPU hours**

To keep pricing consistent, you can set your **maximum RPU hours**, which is a setting on the console, used per day, per week, or per month. You can be alerted when the limit is reached, or you can set the serverless endpoint to stop processing queries. Setting the maximum RPU hours helps keep your cost under control and consistent because of the actions it can take if you reach a usage threshold. These actions don’t limit the performance of the serverless endpoint. You can adjust the maximum RPU hours at any time without an interruption in query processing. For more information, see [Understanding Amazon Redshift Serverless capacity](p. 14).

Setting the base capacity and maximum RPU hours can help you meet your performance requirements for query processing while maintaining consistent costs per week or per month. For more information about the base capacity setting, see [Understanding Amazon Redshift Serverless capacity](p. 14). For more information about serverless billing, see [Amazon Redshift pricing](#).

**Illustrating cost with a billing scenario**

A sample scenario: You run a data-processing job every hour between 7:00am and 7:00pm on your Amazon Redshift data warehouse in the US East (N. Virginia) Region. Assume that each time the job runs, it takes 10 minutes and 30 seconds to complete, which doesn't change. And assume Amazon Redshift runs at 128 RPU capacity during the job. The following results show the day’s total usage and cost:

- **Query duration** - The job runs 13 times between 7:00am-7:00pm, with each run taking 10 minutes and 30 seconds. This adds up to 136 minutes and 30 seconds, or 7830 seconds.
- **Capacity used** - 128 RPUs
- **Daily charges** - $139.20 ((7830 seconds x 128 RPU * $0.50 per RPU-hour for the region) / 3600 seconds in an hour)

For the most up-to-date pricing information, see [Amazon Redshift pricing](#).

**Monitoring usage and cost**

There are several ways you can estimate usage and billing for a serverless endpoint. System views can be helpful because the system metadata, including query and usage data, is timely and you don't have to do any setup to query it. CloudWatch can also be useful for monitoring usage for your serverless endpoint, and has additional features to provide insights and set actions.

**Visualizing usage by querying a system view**

You can query the `sys_serverless_usage` system table to track usage. Query it to retrieve an approximation of the duration that queries were processed:

```sql
select
  trunc(start_time) "Day",
  sum(compute_seconds)/60/60 * <Price for 1 RPU>
from sys_serverless_usage
group by trunc(start_time)
order by 1
```

This query approximates the cost per day incurred for a serverless endpoint. Variations like how Amazon Redshift Serverless reports to AWS billing, differences in aggregation frequency, and rounding can affect the billing. It may differ slightly from the results from `sys_serverless_usage`. Additionally, `sys_serverless_usage` shows your usage in near real time. The billing report is created after queries complete. There may be usage logged in `sys_serverless_usage` that isn't reflected in the billing...
Billing for Amazon Redshift Serverless report. Thus, the information in `sys_serverless_usage` should be used to approximate to your end-of-month billing. For more information about monitoring tables and views, see Monitoring queries and workload with Amazon Redshift Serverless (p. 26).

**Visualizing usage with CloudWatch**

You can use the metrics available in CloudWatch to track usage. The metrics generated for CloudWatch are `ComputeSeconds`, indicating the total RPU seconds used in the current minute and `ComputeCapacity`, indicating the total compute capacity for that minute. Usage metrics can also be found on the Redshift console on the Redshift Serverless dashboard. For more information about CloudWatch, see What is Amazon CloudWatch?

**Billing for storage**

Primary storage capacity is billed as Redshift Managed Storage (RMS). Storage is billed by GB / month. Storage billing is separate from billing for compute resources. Storage used for user snapshots is billed at the standard backup billing rate.

Data transfer costs and machine learning (ML) costs apply separately, the same as provisioned clusters. Snapshot replication and data sharing across AWS regions are billed at the transfer rates outlined on the pricing page. For more information, see Amazon Redshift pricing.

**Billing usage notes**

- **Recording usage** - A query or transaction is only metered and recorded after the transaction completes, is rolled back, or stopped. For instance, if a transaction runs for two days, RPU usage is recorded after it completes. You can monitor ongoing use in real time by querying `sys_serverless_usage`. Transaction recording may reflect as RPU usage variation and affect costs for specific hours and for daily use.

- **Session timeout** - Amazon Redshift Serverless has a mechanism in place to end idle sessions. Idle sessions are terminated after 3600 seconds. This timeout mechanism can be over written by using SESSION TIMEOUT limits of CREATE USER. We recommend that you don't change the timeout, unless you have a specific reason to do so, because it can result in unexpected RPU usage.

- **Writing explicit transactions** - It's important as a best practice to end transactions. If you don't end or roll back an open transaction, Amazon Redshift Serverless continues to use RPUs. For example, if you write an explicit BEGIN TRAN, it's important to have corresponding COMMIT and ROLLBACK statements.

- **Cancelled queries** - If you run a query and cancel it before it finishes, you are still billed for the time the query ran.

- **Scaling** - The serverless endpoint may initiate scaling for handling periods of higher load, in order to maintain consistent performance. Your Amazon Redshift Serverless endpoint billing includes both base compute and scaled capacity at the same RPU rate.

- **System tables** - When you query a system table, the query time is billed. But automatic optimizations, such as automatic sort and vacuum, are not charged for.

- **Redshift Spectrum** - When you have a serverless endpoint and run queries, there isn’t a separate charge for data-lake queries. For queries on data stored in Amazon S3, the charge is the same, by transaction time, as queries on local data.

- **Federated queries** - Federated queries are charged in terms of RPUs used over a specific time interval, in the same manner as queries on the data warehouse or data lake.

- **Storage** - Storage is billed separately, by GB / month.

- **Minimum charge** - The minimum charge is for 60 seconds. Beyond 60 seconds, it is metered on a per-second basis.

- **Snapshot billing** - Snapshot billing doesn't change. It's charged according to storage, billed at a rate of GB / month. You can restore your data warehouse to specific points in the last 24 hours at a 30 minute granularity, free of charge. For more information, see Amazon Redshift pricing.
Get started with Amazon Redshift Serverless

To use Amazon Redshift Serverless, first set up a serverless endpoint. To set up a serverless endpoint using the Amazon Redshift console, you need IAM permission as described in Using identity-based policies (IAM policies) for Amazon Redshift in the Amazon Redshift Cluster Management Guide. In addition, to use Amazon Redshift Serverless, attach a policy similar to the following policy to your IAM role or IAM user.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "redshift-serverless:*",
         "Resource": "*"
      }
   ]
}
```

To get started, open the AWS Management Console, choose the Amazon Redshift console, and then choose Try Amazon Redshift Serverless.

If you have the correct AWS Identity and Access Management (IAM) permissions, the first time you access the serverless endpoint console you view the Get started with Amazon Redshift Serverless page. Your organization might be eligible for Serverless credits for your serverless endpoint. For more information, see Amazon Redshift free trial.

Here is where you can Use default settings or Customize settings to create your serverless endpoint and a database. When you customize, you see database settings, network settings, and settings for permissions. For more information on the customization steps, see Setting up the serverless endpoint using the console (p. 11).

Amazon Redshift Serverless initializes the resources for your AWS account in the current AWS Region. The initialization process can take a few minutes to set up the environment. The Amazon Redshift query editor v2 is opened in a new tab for you to start using your serverless endpoint.

Permissions required to use Amazon Redshift Serverless

When you use Amazon Redshift Serverless, the IAM role you associate to your serverless endpoint needs a trust relationship with both redshift.amazonaws.com and redshift-serverless.amazonaws.com to allow Amazon Redshift to assume permissions on your behalf.

The following example shows the policy document in JSON format to set up a trust relationship with Amazon Redshift Serverless.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "Service": ["redshift-serverless.amazonaws.com", "redshift.amazonaws.com"
         ]
      }
   ]
}
```
Connecting to Amazon Redshift Serverless

Amazon Redshift Serverless provides a serverless endpoint for your AWS account. If you have multiple teams or projects and want to manage costs separately, you can use separate AWS accounts.

The serverless endpoint connects to the serverless environment in your AWS account in the current AWS Region. The serverless endpoint runs in a VPC and is not publicly accessible.

Amazon Redshift Serverless is available in the following AWS Regions:

- US East (N. Virginia) Region (us-east-1)
- US East (Ohio) Region (us-east-2)
- US West (Oregon) Region (us-west-2)
- Europe (Ireland) Region (eu-west-1)
- Europe (Frankfurt) Region (eu-central-1)
- Asia Pacific (Tokyo) Region (ap-northeast-1)

Connecting to serverless endpoint

You can connect to a database (named dev) on the endpoint with the following syntax.

```
account-number.aws-region.redshift-serverless.amazonaws.com:port/dev
```

For example, the following connection string specifies region us-east-1.

```
123456789012.us-east-1.redshift-serverless.amazonaws.com:5439/dev
```

Connecting to Amazon Redshift Serverless through JDBC drivers

You can use one of the following methods to connect to your serverless endpoint with your preferred SQL client using the Amazon Redshift-provided JDBC driver version 2 driver.

To connect with IAM using JDBC driver version 2.x, use the following syntax.

```
jdbc:redshift:iam://redshift-serverless-default:aws-region/dev
```

For example, the following connection string specifies Region us-east-1.

```
jdbc:redshift:iam://redshift-serverless-workspace:us-east-1/dev
```
To connect with a user name and password for database authentication using JDBC driver version 2.0 or later, use the following syntax.

```java
jdbc:redshift://account-number.aws-region.redshift-serverless.amazonaws.com:5439/dev
```

For example, the following connection string specifies account ID 123456789012 in Region us-east-2.

```java
jdbc:redshift://123456789012.us-east-2.redshift-serverless.amazonaws.com:5439/dev
```

To connect with IAM using the JDBC driver version 2.1.0.3, which is available by request during the preview period, use the following syntax.

```java
jdbc:redshift:iam://account-number.aws-region.redshift-serverless.amazonaws.com:5439/dev
```

For example, the following connection string specifies account ID 123456789012 in Region us-east-2.

```java
```

For more information about drivers, see Configuring connections in the Amazon Redshift Cluster Management Guide.

**Connecting to the serverless endpoint with the Data API**

You can also use the Amazon Redshift Data API to connect to serverless endpoint. Leave off the `cluster-identifier` parameter in your AWS CLI calls to route your query to serverless endpoint.

The following example runs a SQL statement to retrieve data from serverless endpoint. This example uses the temporary credentials authentication method.

```
aws redshift-data execute-statement --sql "select 1;"
```

```json
{
    "CreatedAt": 1636062665.587,
    "Database": "dev",
    "Id": "ad30c9a1-be92-4534-9edf-7d9ae4ea6a3"
}
```

The following example describes a SQL statement that was submitted to serverless endpoint.

```
aws redshift-data describe-statement --id 1d222b16-6470-467f-a1f8-7f38103dab11
```

```json
{
    "CreatedAt": 1636064662.659,
    "Duration": 4358742,
    "HasResultSet": true,
    "Id": "1d222b16-6470-467f-a1f8-7f38103dab11",
    "QueryString": "select 1;",
    "RedshiftPid": 1073881246,
    "RedshiftQueryId": 0,
    "ResultRows": 1,
    "ResultSets": []
}
```
The following example retrieves the results from a SQL statement that ran against serverless endpoint.

```
aws redshift-data get-statement-result --id 1d222b16-6470-467f-a1f8-7f38103dab11
```

```json
{
   "Records": [
      {
         "longValue": 1
      }
   ],
   "ColumnMetadata": [
      {
         "isCaseSensitive": false,
         "isCurrency": false,
         "isSigned": true,
         "label": "?column?",
         "length": 0,
         "name": "?column?",
         "nullable": 1,
         "precision": 10,
         "scale": 0,
         "schemaName": "",
         "tableName": "",
         "typeName": "int4"
      }
   ],
   "TotalNumRows": 1
}
```

The following example runs a SQL statement to retrieve data from serverless endpoint. This example uses the AWS Secrets Manager authentication method.

First, create a secret in AWS Secrets Manager.

```
aws secretsmanager create-secret --name serverless-test --secret-string '{
   "password": "Testing12345",
   "engine": "redshift",
   "host": "123456789012.us-east-1.redshift-serverless-dev.amazonaws.com",
   "port": 5439,
   "username": "testUser"
}'
```

```json
{
   "ARN": "arn:aws:secretsmanager:us-east-1:123456789012:secret:serverless-test-YY3nMG",
   "Name": "serverless-test",
   "VersionId": "961a01eb-a30f-4d56-ab05-3708fd60d728"
}
```

Then, run a SQL statement using the secret for authentication.

```
aws redshift-data execute-statement --sql "select * from sys_query_history;"
   --database dev
   --secret-arn arn:aws:secretsmanager:us-east-1:123456789012:secret:serverless-test-YY3nMG
```
Connecting with SSL to the serverless endpoint

You can also run a describe of a SQL statement to see the associated secret.

```
aws redshift-data describe-statement --id 56662da2-5691-4a8d-b1c1-cb73f577f08d
```

For more information about the Data API, see Using the Amazon Redshift Data API in the Amazon Redshift Cluster Management Guide.

## Connecting with SSL to the serverless endpoint

### Configuring a secure connection to Amazon Redshift Serverless

Amazon Redshift supports Secure Sockets Layer (SSL) connections to encrypt queries and data. To set up a secure connection, you can use the same configuration you use to set up a connection to a provisioned Redshift cluster. Follow the steps in Configuring security options for connections, which describes how to download and install the available SSL certificate bundle. The bundle works for a connection to both a serverless Redshift instance and a provisioned cluster.

### Connecting to Amazon Redshift Serverless from other VPC endpoints

You can connect to Amazon Redshift Serverless from other VPC endpoints, including on-premises and public VPC endpoints.

#### Connecting from the public subnet to the Amazon Redshift Serverless endpoint using Network Load Balancer

To enable public access to Amazon Redshift Serverless endpoint, configure the Network Load Balancer in the VPC to listen to the new target group, which is configured to route the traffic to the Amazon Redshift Serverless managed VPC endpoint in the account. For more information, see AWS PrivateLink Guide.
and User Guide for Network Load Balancers. Use AWS CloudFormation to install the redshift-nlb.yml template to create the Network Load Balancer and the Listener Target group.

1. Choose **Serverless configuration**.
2. Choose **Data access**.
3. Get the **Virtual private cloud (VPC)** ID. For example, vpc-def12345.
4. Get the **Subnet** IDs. For example, subnet-abc12345.
5. Choose the **VPC endpoint ID** link to display the VPC Dashboard.
6. Choose **Endpoints** and get the **Endpoint ID**. For example, vpce-0123456789.
7. Choose the **Subnets** tab and get the **IPv4 Address**. For example, 123.12.3.12.
8. Choose **Launch CFN stack**.

(Optional) You can also download and customize the redshift-nlb.yml CloudFormation CFN template, then open AWS CloudFormation and install the customized template.

The CloudFormation console opens with the redshift-nlb.yml template selected.

9. Choose **Next**. The Specify stack details options display.
10. Enter the **SubnetIds**. For example, subnet-abc12345.
11. Enter the **VpcId**. For example, vpc-def12345.
12. Enter the **VpcePrivateIpAddress**. For example, 123.12.3.12.
13. Choose **Next**. The Configure stack options display.
14. Choose **Next** and review the settings.
15. Choose **Create stack**. AWS CloudFormation creates the redshift-nlb stack.
16. Choose **Outputs** and get the Network Load Balancer (NLB) DNS name. For example, redshift-serverless-123456789abcdf.elb.region.amazonaws.com.

### Connecting to Amazon Redshift from a public subnet

You use the Network Load Balancer DNS to connect to Amazon Redshift from the public subnet.

In PSQL, you use syntax similar to the following.

```sql
psql "host=redshift-serverless-dns-name.region.amazonaws.com dbname=dev port=5439 user=admin"
```

To use your preferred query editor and the JDBC driver version 2, be sure that SSL is enabled and server certificate verification is disabled. You use syntax similar to the following.

```java
jdbc:redshift://dns-name.region.amazonaws.com:5439/dev?
useSSL=true&verifyServerCertificate=false
```

### Connecting to the Amazon Redshift serverless endpoint from a Redshift-managed VPC endpoint

Your serverless endpoint is provisioned in a VPC. By creating a Redshift-managed VPC endpoint, you privately access your serverless endpoint from client applications in another VPC. When you do this, the traffic doesn't pass through the Internet and you don't use public IP addresses. This provides for improved communication privacy and security.

**Create a Redshift-managed VPC endpoint on the console**
1. On the console, choose **Serverless configuration**.
2. In **Redshift-managed VPC endpoints**, choose **Create endpoint**.
3. Enter the endpoint name. Create a name that is meaningful for your organization.
4. Choose the AWS account ID. This is your 12-digit account ID, or your account alias.
5. Choose the AWS VPC where the endpoint is located. Then choose a subnet ID. In the most common use case, this is a subnet where you have a client that you want to connect to your serverless endpoint.
6. You can choose VPC security groups to add. Each acts as a virtual firewall to control inbound and outbound traffic to specific virtual-desktop instances, for instance.
7. Choose **Create endpoint**.

**Edit a Redshift-managed VPC endpoint on the console**

1. On the console, choose **Serverless configuration**.
2. In **Redshift-managed VPC endpoints**, choose **Edit**.
3. Add or remove VPC security groups. This is the only setting you can change after creating a Redshift-managed VPC endpoint.
4. Choose **Save changes**.

**Delete a Redshift-managed VPC endpoint on the console**

1. On the console, choose **Serverless configuration**.
2. In **Redshift-managed VPC endpoints**, select the VPC endpoint to delete.
3. Choose **Delete**.

---

**Working with Amazon Redshift query editor v2 and the serverless endpoint**

You can manage and query your databases using query editor v2. The query editor v2 is a full feature web-based SQL client tool to connect to your Amazon Redshift data. To get set up to use the Amazon Redshift query editor v2, including what permissions are needed, see [Configuring your AWS account](#) in the **Amazon Redshift Cluster Management Guide**.

Look for the **Query data** button to query data in your serverless endpoint with query editor v2. When you invoke query editor v2 from the Amazon Redshift Serverless console, a new browser tab opens with the query editor. The query editor v2 connects from your client machine to the serverless endpoint environment. You can use all Amazon Redshift SQL functionality with Amazon Redshift Serverless including semistructured data support, data sharing, machine learning functions, queries to an Amazon S3 data lake, and federated query. With query editor v2, you can do the following tasks:

- Query data in your **Serverless** and provisioned clusters.
- Load sample data into the **sample_data_dev** database.
- Create databases, schemas, tables, and functions.
- Save and share queries.
- Save charts.

For information about query editor v2, see **Querying a database using the Amazon Redshift query editor v2** in the **Amazon Redshift Cluster Management Guide**.
Data sharing in Amazon Redshift Serverless

Use data sharing to share the most up-to-date and consistent information as it’s updated in a serverless endpoint.

Data sharing in Amazon Redshift Serverless

With data sharing, you have live access to data so that your users can see the most up-to-date and consistent information as it’s updated in a serverless endpoint.

Getting started with data sharing in Amazon Redshift Serverless

You can share data for read purposes across different Amazon Redshift Serverless endpoints within or across AWS accounts.

You can get started with data sharing by using either the SQL interface or the Amazon Redshift console. For more information, see either Getting started data sharing using the SQL interface or Getting started data sharing using the console in the Amazon Redshift Database Developer Guide.

Within an AWS account, you can share data for read purposes from a provisioned cluster to a serverless endpoint, or from a serverless endpoint to a provisioned cluster. For more information about sharing data within an AWS account using the SQL interface, see Sharing data within an AWS account in the Amazon Redshift Database Developer Guide.

Across AWS accounts, you can share data for read purposes from a serverless endpoint to another serverless endpoint, from a provisioned cluster to a serverless endpoint, or from a serverless endpoint to a provisioned cluster. For more information about sharing data across AWS accounts, see Sharing data across AWS accounts in the Amazon Redshift Database Developer Guide.

To get started sharing data within an AWS account, open the AWS Management Console, and then choose the Amazon Redshift console. Choose Serverless configuration and then Datashares. Follow the procedures in Getting started data sharing using the console in the Amazon Redshift Database Developer Guide.

To get started sharing data across AWS accounts, open the AWS Management Console, and then choose the Amazon Redshift console. Choose Datashares. Follow the procedures in Getting started data sharing using the console in the Amazon Redshift Database Developer Guide.

Data sharing considerations in Amazon Redshift Serverless

Following are considerations for working with data sharing in Amazon Redshift Serverless:

- When sharing data with provisioned clusters, Amazon Redshift only supports data sharing on the ra3.16xlarge, ra3.4xlarge, and ra3.xlplus instance types for producer and consumer clusters.
- Serverless endpoints are encrypted by default.

For a list of datasharing limitations, see Limitations for data sharing in the Amazon Redshift Database Developer Guide.

Security and connections in Amazon Redshift Serverless

Access to Amazon Redshift requires credentials that AWS can use to authenticate your requests.
Identity and access management in Amazon Redshift Serverless

Access to Amazon Redshift requires credentials that AWS can use to authenticate your requests. Those credentials must have permissions to access AWS resources, such as a serverless endpoint.

The following sections provide details about how you can use AWS Identity and Access Management (IAM) and Amazon Redshift to help secure your resources by controlling who can access them. For more information, see Identity and access management in Amazon Redshift (p. 334).

Granting permissions to Amazon Redshift Serverless

To access other AWS services, Amazon Redshift Serverless requires permissions.

Authorizing Amazon Redshift Serverless to access other AWS services for you

Some Amazon Redshift features require Amazon Redshift to access other AWS services on your behalf. For your Amazon Redshift Serverless endpoints to act for you, supply security credentials to your endpoints. The preferred method to supply security credentials is to specify an AWS Identity and Access Management (IAM) role. You can also create an IAM role through the Amazon Redshift console and set it as the default. For more information, see Creating an IAM role as default for Amazon Redshift (p. 25).

To access other AWS services, create an IAM role with the appropriate permissions. You also need to associate the role with your serverless endpoint. In addition, either specify the Amazon Resource Name (ARN) of the role when you run the Amazon Redshift command or specify the default keyword.

When changing the trust relationship for the IAM role in the https://console.aws.amazon.com/iam/, make sure that you use redshift.amazonaws.com as the service name. For information about how to manage IAM roles to access other AWS services on your behalf, see Authorizing Amazon Redshift to access other AWS services on your behalf (p. 405).

Creating an IAM role as default for Amazon Redshift

When you create IAM roles through the Amazon Redshift console, Amazon Redshift programmatically creates the roles in your AWS account. Amazon Redshift also automatically attaches existing AWS managed policies to them. This approach means that you can stay within the Amazon Redshift console and don't have to switch to the IAM console for role creation.

The IAM role that you create through the console for your cluster has the AmazonRedshiftAllCommandsFullAccess managed policy automatically attached. This IAM role allows Amazon Redshift to copy, unload, query, and analyze data for AWS resources in your IAM account. The related commands include COPY, UNLOAD, CREATE EXTERNAL FUNCTION, CREATE EXTERNAL TABLE, CREATE EXTERNAL SCHEMA, CREATE MODEL, and CREATE LIBRARY. For more information about how to create an IAM role as default for Amazon Redshift, see Creating an IAM role as default for Amazon Redshift (p. 25).

To get started creating an IAM role as default for Amazon Redshift, open the AWS Management Console, choose the Amazon Redshift console, and then choose Try Amazon Redshift Serverless. On the Amazon Redshift Serverless console, choose Customize settings. Under Permissions, follow the procedures in Using the console to manage IAM role associations (p. 412).

When you already have a serverless endpoint and want to configure IAM roles for your endpoint, open the AWS Management Console. Choose the Amazon Redshift console, and then choose Go to serverless. On the Amazon Redshift Serverless console, choose Serverless configuration and then Data access. Under Permissions, follow the procedures in Using the console to manage IAM role associations (p. 412).
Getting started with IAM credentials for Amazon Redshift

When you sign in to the Amazon Redshift console for the first time and first try out Amazon Redshift Serverless, you can either log on as an IAM user or an IAM role. After you get started creating a serverless endpoint, Amazon Redshift records your IAM user name or role name that you used when you signed in. You can use the same IAM credentials to sign in to the Amazon Redshift console and the Amazon Redshift Serverless console.

While creating a serverless endpoint, you can create a database in your serverless endpoint. Use the query editor v2 to connect to the database with the temporary credentials option.

To add a new admin user name and password that persist for the database, choose Customize admin user credentials and enter a new admin user name and admin user password.

To get started using Amazon Redshift Serverless and create a serverless endpoint in the serverless endpoint console for the first time, use an IAM user or IAM role. Make sure that this user or role has either the administrator permission arn:aws:iam::aws:policy/AdministratorAccess or the full Amazon Redshift permission arn:aws:iam::aws:policy/AmazonRedshiftFullAccess attached to the IAM policy that you used.

The following scenarios outline how your IAM credentials are used by Amazon Redshift Serverless when you get started on the Amazon Redshift Serverless console:

- If you choose Use default settings – Amazon Redshift Serverless translates your current IAM identity to a database superuser. You can use the same IAM identity with the Amazon Redshift Serverless console to perform superuser actions in your database in the serverless endpoint.
- If you choose Customize settings without specifying Admin user name and password Amazon Redshift Serverless – Your current IAM credentials are used as your default admin user credentials. This is basically the same as Use default settings.
- If you choose Customize settings and specify Admin user name and password Amazon Redshift Serverless – Amazon Redshift Serverless translates your current IAM identity to a database superuser. Amazon Redshift Serverless also creates another long-term login username and password pair also as a superuser. You can either use your current IAM identity or the created username and password pair to login in to your database as a superuser.

Monitoring queries and workload with Amazon Redshift Serverless

You can monitor your serverless endpoint queries and workload with the provided system views.

Granting access to monitor queries

A superuser can provide access to users who aren’t superusers so that they can perform query monitoring for all users. First, you add a policy for a user or a role to provide query monitoring access. Then, you grant query monitoring permission to the user or role.

To add the query monitoring policy

3. Choose Create Policy.
4. Choose JSON and paste the following policy definition.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "redshift-data:ExecuteStatement",
        "redshift-data:DescribeStatement",
        "redshift-data:GetStatementResult",
        "redshift-data:ListDatabases"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "redshift-serverless:GetCredentials",
      "Resource": "*"
    }
  ]
}
```

Ignore the following error message, if it appears.

*Invalid Service In Action: The service redshift-serverless:* specified in the action does not exist.*

5. Choose Review policy.
6. For Name, enter a name for the policy, such as query-monitoring.
7. Choose Create policy.
8. Under Policies, choose the policy you created.
10. Choose the users or roles to which you want to attach the policy and choose Attach policy.

**To grant query monitoring permission for a user**

Users with sys:monitor permission can view all queries. In addition, users with sys:operator permission can cancel queries, analyze query history, and perform vacuum operations.

1. Enter the following command to provide system monitor access, where *user-name* is the name of the user for whom you want to provide access.

```
grant "sys:monitor" to "IAM:user-name";
```

2. (Optional) Enter the following command to provide system operator access, where *user-name* is the name of the user for whom you want to provide access.

```
grant "sys:operator" to "IAM:user-name";
```

**To grant query monitoring permission for a role**

Users with a role that has sys:monitor permission can view all queries. In addition, users with a role that has sys:operator permission can cancel queries, analyze query history, and perform vacuum operations.
Monitoring views

Monitoring views are system views in Amazon Redshift Serverless that are used to monitor query and workload usage. These views are located in the `pg_catalog` schema. The system views available have been designed to give you the information needed to monitor the serverless endpoint, which is much simpler than needed for provisioned clusters. The SYS system views have been designed to work with a serverless endpoint. To display the information provided by these views, run SQL SELECT statements.

System views are defined to support the following monitoring objectives.

Workload monitoring

You can monitor your query activities over time to:

- Understand workload patterns, so you know what is normal (baseline) and what is within business service level agreements (SLAs).
- Rapidly identify deviation from normal, which might be a transient issue or something that warrants further action.

Data load and unload monitoring

Data movement in and out of a serverless endpoint is a critical function. You use COPY and UNLOAD to load or unload data, and you must monitor progress closely in terms of bytes/rows transferred and files completed to track adherence to business SLAs. This is normally done by running system table queries frequently (that is, every minute) to track progress and raise alerts for investigation/corrective action if significant deviations are detected.

Failure and problem diagnostics

There are cases where you must take action for query or runtime failures. Developers rely on system tables to self-diagnose issues and determine correct remedies.

Performance tuning

You might need to tune queries that are not meeting SLA requirements either from the start or have degraded over time. To tune, you need to have runtime details including run plan, statistics, duration, and resource consumption. You need baseline data for offending queries to determine the cause for deviation and to guide you on how to make improvements.

User objects event monitoring

You need to monitor actions and activities on user objects like refreshing materialized views, vacuum, and analyze. This includes system-managed events like auto-refresh for materialized views. You want to monitor when an event ends if it is user initiated, or the last successful execution if system initiated.

Usage tracking for billing

You can monitor your usage trends over time to:

- Inform budget planning and business expansion estimates.
- Identify potential cost-saving opportunities like removing cold data.
You can't query STL, STV, SVCS, SVL, and some SVV system tables and views with Amazon Redshift Serverless, except the following:

- SVV_ALL_COLUMNS
- SVV_ALL_SCHEMAS
- SVV_ALL_TABLES
- SVV_TABLES
- SVV_DATASHARES
- SVV_DATASHARE_CONSUMERS
- SVV_DATASHARE_OBJECTS
- SVV_EXTERNAL_COLUMNS
- SVV_EXTERNAL_DATABASES
- SVV_EXTERNAL_PARTITIONS
- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_REDSHIFT_COLUMNS
- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES
- SVV_TABLE_INFO
- SVV_TRANSACTIONS

You can query the following SYS system views to monitor a serverless endpoint.

- SYS_QUERY_HISTORY
- SYS_QUERYDETAIL
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_LOAD_HISTORY
- SYS_LOAD_ERROR_DETAIL
- SYS_UNLOAD_HISTORY
- SYS_SERVERLESS_USAGE
might notice minor performance degradation during this phase. The second phase can last from a few hours to several days and in some cases a couple of weeks. The length of time depends on the size of data, but performance progressively improves. At the end of this phase, your serverless endpoint is fully tuned.

### Snapshots

You can restore a snapshot that you created on the Amazon Redshift Serverless console to replace the data in your serverless endpoint. You can restore a snapshot encrypted with an AWS managed KMS key to a serverless endpoint.

**To create a snapshot**

1. On the Amazon Redshift Serverless console, choose **Data backup**.
2. Choose **Create snapshot**.
3. Enter a snapshot identifier.
4. (Optional) Choose a retention period. If you choose **Custom value**, choose the number of days.
5. Choose **Create**.

**To manage access to a snapshot**

1. On the Amazon Redshift Serverless console, choose **Data backup**.
2. Choose **Action**, **Manage access**.
3. Enter an **AWS account ID** to authorize it to use the snapshot.

**To restore a snapshot from your serverless account**

1. On the Amazon Redshift Serverless console, choose **Data backup**.
2. Choose a snapshot to use.
3. Choose **Action**, **Restore from snapshot**.

**To restore a snapshot from your provisioned cluster to your serverless endpoint**

1. Start on the Amazon Redshift provisioned cluster console, navigate to the **Clusters, Snapshots** page.
2. Choose a snapshot to use.
3. Choose **Restore from snapshot**, **Restore to serverless endpoint**.
4. Confirm you want to restore from your snapshot. This action replaces all the databases in your serverless endpoint with the data from your provisioned cluster.

For more information about snapshots on provisioned clusters, see [Amazon Redshift snapshots](https://docs.aws.amazon.com/redshift/latest/gsg/index.html) in the *Amazon Redshift Cluster Management Guide*.

### Recovery points

Recovery points are created for your serverless endpoint every 30 minutes and saved for 24 hours.

On the Amazon Redshift Serverless console, choose **Data backup** to manage recovery points. You can do the following operations:

- Restore a serverless endpoint to a recovery point.
- Convert a recovery point to a snapshot.
To restore a serverless endpoint to a recovery point

1. On the Amazon Redshift Serverless console, choose **Data backup**.
2. Under **Recovery points**, choose the **Creation time** of the recovery point that you want to restore.
3. Choose **Restore**.
4. Enter `restore` in the text input field and choose **Restore**.

To convert a recovery point to a snapshot

1. On the Amazon Redshift Serverless console, choose **Data backup**.
2. Under **Recovery points**, choose the **Creation time** of the recovery point that you want to convert to a snapshot.
3. Choose **Create snapshot from recovery point**.
4. Enter a **Snapshot identifier**.
5. Choose **Create**.

Audit logging for Amazon Redshift Serverless

Exporting logs

You can configure Amazon Redshift Serverless to export connection, user, and user-activity log data to a log group in Amazon CloudWatch Logs. With Amazon CloudWatch Logs, you can perform real-time analysis of the log data and use CloudWatch to create alarms and view metrics. You can use CloudWatch Logs to store your log records in durable storage.

To export generated log data to Amazon CloudWatch Logs, the respective logs must be selected for export in the serverless endpoint configuration settings, on the console.

Monitoring log events in CloudWatch

After selecting which Redshift logs to export, you can monitor events in Amazon CloudWatch Logs. A new log group is automatically created for Amazon Redshift Serverless under the following prefix, in which `log_type` represents the log type.

```
/aws/redshift/serverless/<log_type>
```

For example, if you choose to export the connection log, log data is stored in the following log group.

```
/aws/redshift/serverless/connectionlog
```

Log events are exported to a log group using the serverless log stream. The behavior depends on which of the following conditions are true:

- **A log group with the specified name exists.** Redshift exports log data for the serverless endpoint using the existing log group. To create log groups with predefined log-retention periods, metric filters, and customer access, you can use automated configuration, such as that provided by [AWS CloudFormation](https://aws.amazon.com/cloudformation/).

- **A log group with the specified name doesn’t exist.** When a matching log entry is detected in the log for the instance, Amazon Redshift Serverless creates a new log group in Amazon CloudWatch Logs automatically. The log group uses the default log-retention period of **Never Expire**. To change the log-retention period, use the Amazon CloudWatch Logs console, the AWS CLI, or the Amazon CloudWatch Logs API.
Logs API. For more information about changing log-retention periods in CloudWatch Logs, see *Change log data retention* in *Working with log groups and log streams*.

To search for information within log events for your serverless endpoint, use the Amazon CloudWatch Logs console, the AWS CLI, or the Amazon CloudWatch Logs API. For more information about searching and filtering log data, see *Searching and filtering log data*.

**Amazon Redshift Serverless metrics**

CloudWatch metrics are the following:

<table>
<thead>
<tr>
<th>Metric name</th>
<th>Units</th>
<th>Description</th>
<th>Dimension sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueriesCompletedPerSecond</td>
<td>Number of queries</td>
<td>The number of queries completed each second.</td>
<td>{DatabaseName, Latency}, {DatabaseName}, {Latency}, {}</td>
</tr>
<tr>
<td>QueryDuration</td>
<td>Microseconds</td>
<td>The average amount of time to complete a query.</td>
<td>{DatabaseName, Latency}, {DatabaseName}, {Latency}, {}</td>
</tr>
<tr>
<td>QueriesRunning</td>
<td>Number of queries</td>
<td>The number of running queries at a point in time.</td>
<td>{DatabaseName, QueryType}, {DatabaseName}, {QueryType}, {}</td>
</tr>
<tr>
<td>QueriesQueued</td>
<td>Number of queries</td>
<td>The number of queries in the queue at a point in time.</td>
<td>{DatabaseName, QueryType}, {DatabaseName}, {QueryType}, {}</td>
</tr>
<tr>
<td>DatabaseConnections</td>
<td>Number of connections</td>
<td>The number of connections to a database at a point in time.</td>
<td>{DatabaseName}, {}</td>
</tr>
<tr>
<td>QueryRuntimeBreakdown</td>
<td>Milliseconds</td>
<td>The total time queries ran, by query stage.</td>
<td>{DatabaseName, Stage}, {DatabaseName}, {Stage}, {}</td>
</tr>
<tr>
<td>ComputeCapacity</td>
<td>RPU</td>
<td>Average number of compute units allocated during the past 30 minutes, rounded up to the nearest integer.</td>
<td>{}</td>
</tr>
<tr>
<td>ComputeSeconds</td>
<td>RPU-seconds</td>
<td>Accumulated compute-unit seconds used in the last 30 minutes.</td>
<td>{}</td>
</tr>
</tbody>
</table>
### Metric name

<table>
<thead>
<tr>
<th>Metric name</th>
<th>Units</th>
<th>Description</th>
<th>Dimension sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueriesSucceeded</td>
<td>Number of queries</td>
<td>The number of queries that succeeded in the last 5 minutes.</td>
<td>{DatabaseName, QueryType}, {DatabaseName}, {QueryType}, {}</td>
</tr>
<tr>
<td>QueriesFailed</td>
<td>Number of queries</td>
<td>The number of queries that failed in the last 5 minutes.</td>
<td>{DatabaseName, QueryType}, {DatabaseName}, {QueryType}, {}</td>
</tr>
</tbody>
</table>

Dimension sets are the grouping dimensions applied to your metrics. You can use these dimension groups to specify how your statistics are retrieved.

The following table details dimensions and dimension values for specific metrics:

#### CloudWatch dimensions and dimension values

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description and values</th>
</tr>
</thead>
<tbody>
<tr>
<td>DatabaseName</td>
<td>The name of the database. A custom value.</td>
</tr>
<tr>
<td>latency</td>
<td>Possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• short — under 10 seconds</td>
</tr>
<tr>
<td></td>
<td>• medium — between 10 seconds and 10 minutes</td>
</tr>
<tr>
<td></td>
<td>• long — over 10 minutes</td>
</tr>
<tr>
<td>QueryType</td>
<td>Possible values are INSERT, DELETE, UPDATE, UNLOAD, LOAD, SELECT, CTAS, and OTHER.</td>
</tr>
<tr>
<td>stage</td>
<td>The execution stages for a query. Possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• QueryPlanning: Time spent parsing and optimizing SQL statements.</td>
</tr>
<tr>
<td></td>
<td>• QueryWaiting: Time spent waiting in the WLM queue.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingRead: Time spent executing read queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingInsert: Time spent executing insert queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingDelete: Time spent executing delete queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingUpdate: Time spent executing update queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingCtas: Time spent executing create table as queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingUnload: Time spent executing unload queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryExecutingCopy: Time spent executing copy queries.</td>
</tr>
<tr>
<td></td>
<td>• QueryCommit: Time spent committing.</td>
</tr>
</tbody>
</table>
Amazon Redshift clusters

In the following sections, you can learn the basics of creating a data warehouse by launching a set of compute nodes, called an Amazon Redshift cluster.

**Topics**
- Overview of Amazon Redshift clusters (p. 34)
- Use EC2-VPC when you create your cluster (p. 38)
- Overview of RA3 node types (p. 39)
- Upgrading to RA3 node types (p. 41)
- Upgrading from DC1 node types to DC2 node types (p. 44)
- Upgrading a DS2 cluster on EC2-Classic to EC2-VPC (p. 45)
- Region and Availability Zone considerations (p. 45)
- Cluster maintenance (p. 45)
- Default disk space alarm (p. 49)
- Cluster status (p. 50)
- Overview of managing clusters in Amazon Redshift (p. 51)
- Working with AQUA (Advanced Query Accelerator) (p. 59)
- Managing usage limits in Amazon Redshift (p. 63)
- Managing cluster relocation in Amazon Redshift (p. 64)
- Working with Redshift-managed VPC endpoints in Amazon Redshift (p. 67)
- Managing clusters using the console (p. 70)
- Managing clusters using the AWS CLI and Amazon Redshift API (p. 76)
- Managing clusters using the AWS SDK for Java (p. 77)
- Managing clusters in a VPC (p. 79)
- Cluster version history (p. 86)

**Overview of Amazon Redshift clusters**

An Amazon Redshift data warehouse is a collection of computing resources called nodes, which are organized into a group called a cluster. Each cluster runs an Amazon Redshift engine and contains one or more databases.

**Note**
At this time, Amazon Redshift version 1.0 engine is available. However, as the engine is updated, multiple Amazon Redshift engine versions might be available for selection.

**Clusters and nodes in Amazon Redshift**

An Amazon Redshift cluster consists of nodes. Each cluster has a leader node and one or more compute nodes. The leader node receives queries from client applications, parses the queries, and develops query execution plans. The leader node then coordinates the parallel execution of these plans with the compute nodes and aggregates the intermediate results from these nodes. It then finally returns the results back to the client applications.

Compute nodes run the query execution plans and transmit data among themselves to serve these queries. The intermediate results are sent to the leader node for aggregation before being sent back to the client applications. For more information about leader nodes and compute nodes, see Data warehouse system architecture in the Amazon Redshift Database Developer Guide.
When you launch a cluster, one option that you specify is the node type. The node type determines the CPU, RAM, storage capacity, and storage drive type for each node.

Amazon Redshift offers different node types to accommodate your workloads, and we recommend choosing RA3 or DC2 depending on the required performance, data size, and expected data growth.

RA3 nodes with managed storage enable you to optimize your data warehouse by scaling and paying for compute and managed storage independently. With RA3, you choose the number of nodes based on your performance requirements and only pay for the managed storage that you use. Size your RA3 cluster based on the amount of data you process daily. You launch clusters that use the RA3 node types in a virtual private cloud (VPC). You can’t launch RA3 clusters in EC2-Classic. For more information, see Creating a cluster in a VPC (p. 81).

Amazon Redshift managed storage uses large, high-performance SSDs in each RA3 node for fast local storage and Amazon S3 for longer-term durable storage. If the data in a node grows beyond the size of the large local SSDs, Amazon Redshift managed storage automatically offloads that data to Amazon S3. You pay the same low rate for Amazon Redshift managed storage regardless of whether the data sits in high-performance SSDs or Amazon S3. For workloads that require ever-growing storage, managed storage lets you automatically scale your data warehouse storage capacity without adding and paying for additional nodes.

DC2 nodes enable you to have compute-intensive data warehouses with local SSD storage included. You choose the number of nodes you need based on data size and performance requirements. DC2 nodes store your data locally for high performance, and as the data size grows, you can add more compute nodes to increase the storage capacity of the cluster. For datasets under 1 TB (compressed), we recommend DC2 node types for the best performance at the lowest price. If you expect your data to grow, we recommend using RA3 nodes so you can size compute and storage independently to achieve improved price and performance. You launch clusters that use the DC2 node types in a virtual private cloud (VPC). You can’t launch DC2 clusters in EC2-Classic. For more information, see Creating a cluster in a VPC (p. 81).

DS2 nodes enable you to create large data warehouses using hard disk drives (HDDs), and we recommend using RA3 nodes instead. If you are using DS2 nodes, see Upgrading to RA3 node types (p. 41) for upgrade guidelines. If you are using eight or more nodes of ds2.xlarge, or any number of ds2.8xlarge nodes, you can now upgrade to RA3 to get 2x more storage and improved performance for the same on-demand cost.

Node types are available in different sizes. Node size and the number of nodes determine the total storage for a cluster. For more information, see Node type details (p. 36).

Some node types allow one node (single-node) or two or more nodes (multi-node). The minimum number of nodes for clusters of some node types is two nodes. On a single-node cluster, the node is shared for leader and compute functionality. Single-node clusters are not recommended for running production workloads. On a multi-node cluster, the leader node is separate from the compute nodes. The leader node is the same node type as the compute nodes. You only pay for compute nodes.

Amazon Redshift applies quotas to resources for each AWS account in each AWS Region. A quota restricts the number of resources that your account can create for a given resource type, such as nodes or snapshots, within an AWS Region. For more information about the default quotas that apply to Amazon Redshift resources, see Amazon Redshift Limits in the Amazon Web Services General Reference. To request an increase, submit an Amazon Redshift Limit Increase Form.
The cost of your cluster depends on the AWS Region, node type, number of nodes, and whether the nodes are reserved in advance. For more information about the cost of nodes, see the Amazon Redshift pricing page.

**Node type details**

The following tables summarize the node specifications for each node type and size. The headings in the tables have these meanings:

- **vCPU** is the number of virtual CPUs for each node.
- **RAM** is the amount of memory in gibibytes (GiB) for each node.
- **Default slices per node** is the number of slices into which a compute node is partitioned when a cluster is created or resized with classic resize.

The number of slices per node might change if the cluster is resized using elastic resize. However, the total number of slices on all the compute nodes in the cluster remains the same after elastic resize.

When you create a cluster with the restore from snapshot operation, the number of slices of the resulting cluster might change from the original cluster if you change the node type.

- **Storage** is the capacity and type of storage for each node.
- **Node range** is the minimum and maximum number of nodes that Amazon Redshift supports for the node type and size.

**Note**

You might be restricted to fewer nodes depending on the quota that is applied to your AWS account in the selected AWS Region. To request an increase, submit an Amazon Redshift Limit Increase Form.

- **Total capacity** is the total storage capacity for the cluster if you deploy the maximum number of nodes that is specified in the node range.

**RA3 node types**

<table>
<thead>
<tr>
<th>Node size</th>
<th>vCPU</th>
<th>RAM (GiB)</th>
<th>Default slices per node</th>
<th>Managed storage quota per node</th>
<th>Node range with create cluster</th>
<th>Total managed storage capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ra3.xlplus</td>
<td>4</td>
<td>32</td>
<td>2</td>
<td>32 TB$^{1,5}$</td>
<td>1–16$^2$</td>
<td>1024 TB$^{2,4}$</td>
</tr>
<tr>
<td>ra3.4xlarge</td>
<td>12</td>
<td>96</td>
<td>4</td>
<td>128 TB$^1$</td>
<td>2–32$^3$</td>
<td>8192 TB$^{3,4}$</td>
</tr>
<tr>
<td>ra3.16xlarge</td>
<td>48</td>
<td>384</td>
<td>16</td>
<td>128 TB$^1$</td>
<td>2–128</td>
<td>16,384 TB$^4$</td>
</tr>
</tbody>
</table>

$^1$ The storage quota for Amazon Redshift managed storage.

$^2$ You can create a cluster with the ra3.xlplus node type that has up to 16 nodes. For single-node clusters, only classic resize is supported. For multiple-node clusters, you can resize with elastic resize to a maximum of 32 nodes.

$^3$ You can create clusters with the ra3.4xlarge or ra3.16xlarge node type with up to 16 nodes. You can resize it with elastic resize to a maximum of 64 nodes.

$^4$ Total managed storage quota is the maximum number of nodes times the managed storage quota per node.

$^5$ Total managed storage quota for a one-node ra3.xlplus cluster is 4TB.
### Dense storage node types

<table>
<thead>
<tr>
<th>Node size</th>
<th>vCPU</th>
<th>RAM (GiB)</th>
<th>Default slices per node</th>
<th>Storage per node</th>
<th>Node range</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ds2.xlarge</td>
<td>4</td>
<td>31</td>
<td>2</td>
<td>2 TB HDD</td>
<td>1–32</td>
<td>64 TB</td>
</tr>
<tr>
<td>ds2.8xlarge</td>
<td>36</td>
<td>244</td>
<td>16</td>
<td>16 TB HDD</td>
<td>2–128</td>
<td>2 PB</td>
</tr>
</tbody>
</table>

### Dense compute node types

<table>
<thead>
<tr>
<th>Node size</th>
<th>vCPU</th>
<th>RAM (GiB)</th>
<th>Default slices per node</th>
<th>Storage per node</th>
<th>Node range</th>
<th>Total capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>dc2.large</td>
<td>2</td>
<td>15</td>
<td>2</td>
<td>160 GB NVMe-SSD</td>
<td>1–32</td>
<td>5.12 TB</td>
</tr>
<tr>
<td>dc2.8xlarge</td>
<td>32</td>
<td>244</td>
<td>16</td>
<td>2.56 TB NVMe-SSD</td>
<td>2–128</td>
<td>326 TB</td>
</tr>
<tr>
<td>dc1.large</td>
<td>2</td>
<td>15</td>
<td>2</td>
<td>160 GB SSD</td>
<td>1–32</td>
<td>5.12 TB</td>
</tr>
<tr>
<td>dc1.8xlarge</td>
<td>32</td>
<td>244</td>
<td>32</td>
<td>2.56 TB SSD</td>
<td>2–128</td>
<td>326 TB</td>
</tr>
</tbody>
</table>

1 We recommend DC2 node types over DC1 node types. For more information on how to upgrade, see Upgrading from DC1 node types to DC2 node types (p. 44).

### Previous node type names

In previous releases of Amazon Redshift, certain node types had different names. You can use the previous names in the Amazon Redshift API and AWS CLI. However, we recommend that you update any scripts that reference those names to use the current names instead. The current and previous names are as follows.

<table>
<thead>
<tr>
<th>Current name</th>
<th>Previous names</th>
</tr>
</thead>
<tbody>
<tr>
<td>ds2.xlarge</td>
<td>ds1.xlarge, dw.hs1.xlarge, dw1.xlarge</td>
</tr>
<tr>
<td>ds2.8xlarge</td>
<td>ds1.8xlarge, dw.hs1.8xlarge, dw1.8xlarge</td>
</tr>
<tr>
<td>dc1.large</td>
<td>dw2.large</td>
</tr>
<tr>
<td>dc1.8xlarge</td>
<td>dw2.8xlarge</td>
</tr>
</tbody>
</table>

### Determining the number of nodes

Because Amazon Redshift distributes and runs queries in parallel across all of a cluster's compute nodes, you can increase query performance by adding nodes to your cluster. When you run a cluster with at least two compute nodes, data on each node is mirrored on disks of another node to reduce the risk of incurring data loss.

You can monitor query performance in the Amazon Redshift console and with Amazon CloudWatch metrics. You can also add or remove nodes as needed to achieve the balance between price and...
performance for your cluster. When you request an additional node, Amazon Redshift takes care of all the details of deployment, load balancing, and data maintenance. For more information about cluster performance, see Monitoring Amazon Redshift cluster performance (p. 458).

Reserved nodes are appropriate for steady-state production workloads, and offer significant discounts over on-demand nodes. You can purchase reserved nodes after running experiments and proof-of-concepts to validate your production configuration. For more information, see Purchasing Amazon Redshift reserved nodes (p. 312).

When you pause a cluster, you suspend on-demand billing during the time the cluster is paused. During this paused time, you only pay for backup storage. This frees you from planning and purchasing data warehouse capacity ahead of your needs, and enables you to cost-effectively manage environments for development or test purposes.

For information about pricing of on-demand and reserved nodes, see Amazon Redshift pricing.

**Use EC2-VPC when you create your cluster**

Amazon Redshift clusters run in Amazon EC2 instances that are configured for the Amazon Redshift node type and size that you select. Create your cluster using EC2-VPC. If you are still using EC2-Classic, we recommend you use EC2-VPC to get improved performance and security. For more information about these networking platforms, see Supported Platforms in the Amazon EC2 User Guide for Linux Instances. Your AWS account settings determine whether EC2-VPC or EC2-Classic are available to you.

**Note**

To prevent connection issues between SQL client tools and the Amazon Redshift database, we recommend doing one of two things. You can configure an inbound rule that enables the hosts to negotiate packet size. Alternatively, you can disable TCP/IP jumbo frames by setting the maximum transmission unit (MTU) to 1500 on the network interface (NIC) of your Amazon EC2 instances. For more information about these approaches, see Queries appear to hang and sometimes fail to reach the cluster (p. 236).

**EC2-VPC**

When using EC2-VPC, your cluster runs in a virtual private cloud (VPC) that is logically isolated to your AWS account. If you provision your cluster in the EC2-VPC, you control access to your cluster by associating one or more VPC security groups with the cluster. For more information, see Security Groups for Your VPC in the Amazon VPC User Guide.

To create a cluster in a VPC, you must first create an Amazon Redshift cluster subnet group by providing subnet information of your VPC, and then provide the subnet group when launching the cluster. For more information, see Amazon Redshift cluster subnet groups (p. 82).

For more information about Amazon Virtual Private Cloud (Amazon VPC), see the Amazon VPC product detail page.

**EC2-Classic**

The EC2-Classic platform is retiring on August 15, 2022. We recommend that you migrate your clusters from EC2-Classic platform to a EC2-VPC platform. For more information, see Upgrading a DS2 cluster on EC2-Classic to EC2-VPC (p. 45) and EC2-Classic Networking is Retiring – Here’s How to Prepare.

In EC2-Classic, your cluster runs in a single, flat network that you share with other AWS customers. If you provision your cluster in the EC2-Classic, you control access to your cluster by associating one or more
Amazon Redshift cluster security groups with the cluster. For more information, see Amazon Redshift cluster security groups (p. 438).

Launch a cluster

Your AWS account can either launch instances of both EC2-VPC and EC2-Classic, or only EC2-VPC, on a region-by-region basis. To determine which networking platform your account supports, and then launch a cluster, do the following:

1. Decide which AWS Region you want to deploy a cluster. For a list of AWS Regions in which Amazon Redshift is available, see Amazon Redshift endpoints in the Amazon Web Services General Reference.
2. Find out which Amazon EC2 platforms your account supports in the chosen AWS Region. You can find this information in the Amazon EC2 console. For step-by-step instructions, see Supported Platforms in the Amazon EC2 User Guide for Linux Instances.
3. If your account supports both of the platforms, we recommend EC2-VPC. If your account supports only EC2-VPC, you must deploy your cluster in VPC.
4. Launch your Amazon Redshift cluster. You can create a cluster using the Amazon Redshift console, or by using the Amazon Redshift API, AWS CLI, or SDK libraries. For more information about these options and links to the related documentation, see What is Amazon Redshift? (p. 1).

Overview of RA3 node types

We recommend that you upgrade existing workloads running on DS2 node type clusters to RA3 node types to take advantage of improved performance and to get more storage capacity. RA3 nodes provide the following advantages:

- They are flexible to grow your compute capacity without increasing your storage costs. And they scale your storage without over-provisioning compute capacity.
- They use high performance SSDs for your hot data and Amazon S3 for cold data. Thus they provide ease of use, cost-effective storage, and high query performance.
- They use high bandwidth networking built on the AWS Nitro System to further reduce the time taken for data to be offloaded to and retrieved from Amazon S3.

Consider choosing RA3 node types in these cases:

- You need the flexibility to scale and pay for compute separate from storage.
- You query a fraction of your total data.
- Your data volume is growing rapidly or is expected to grow rapidly.
- You want the flexibility to size the cluster based only on your performance needs.

To use RA3 node types, your AWS Region must support RA3. For more information, see RA3 node type availability in AWS Regions (p. 40).

Important

You can use ra3.xlplus node types only with cluster version 1.0.21262 or later. You can view the version of an existing cluster with the Amazon Redshift console. For more information, see Determining the cluster maintenance version (p. 49).

Make sure that you use the new Amazon Redshift console when working with RA3 node types.

In addition, to use RA3 node types with Amazon Redshift operations that use the maintenance track, the maintenance track value must be set to a cluster version that supports RA3. For more information about maintenance tracks, see Choosing cluster maintenance tracks (p. 47).

Consider the following when using single-node RA3 node types.
• AQUA is supported.
• Datasharing producers and consumers are supported.
• To change node types, only classic resize is supported. Changing the node type with elastic resize or snapshot restore is not supported. The following scenarios are supported:
  • Classic resize of a 1-node ds2.xlarge to a 1-node ra3.xlplus, and vice versa.
  • Classic resize of a 1-node ds2.xlarge to a multiple-node ra3.xlplus, and vice versa.
  • Classic resize of a multiple-node ds2.xlarge to a 1-node ra3.xlplus, and vice versa.
  • Classic resize of a 1-node dc2.xlarge to a 1-node ra3.xlplus, and vice versa.
  • Classic resize of a 1-node dc2.xlarge to a multiple-node ra3.xlplus, and vice versa.
  • Classic resize of a multiple-node dc2.xlarge to a 1-node ra3.xlplus, and vice versa.

Working with Amazon Redshift managed storage

With Amazon Redshift managed storage, you can store and process all your data in Amazon Redshift while getting more flexibility to scale compute and storage capacity separately. You continue to ingest data with the COPY or INSERT command. To optimize performance and manage automatic data placement across tiers of storage, Amazon Redshift takes advantage of optimizations such as data block temperature, data block age, and workload patterns. When needed, Amazon Redshift scales storage automatically to Amazon S3 without requiring any manual action.

For information about storage costs, see Amazon Redshift pricing.

Managing RA3 node types

To take advantage of separating compute from storage, you can create or upgrade your cluster with the RA3 node type. To use the RA3 node types, create your clusters in a virtual private cloud (EC2-VPC).

To change the number of nodes of Amazon Redshift cluster with an RA3 node type, do one of the following:

• Add or remove nodes with the elastic resize operation. In some situations, removing nodes from a RA3 cluster isn't allowed with elastic resize. For example, when a 2:1 node count upgrade puts the number of slices per node at 32. For more information, see Resizing clusters (p. 51). If elastic resize isn't available, use classic resize.
• Add or remove nodes with the classic resize operation. Choose this option when you are resizing to a configuration that isn't available through elastic resize. Elastic resize is quicker than classic resize. For more information, see Resizing clusters (p. 51).

RA3 node type availability in AWS Regions

The RA3 node types are available only in the following AWS Regions:

• US East (N. Virginia) Region (us-east-1)
• US East (Ohio) Region (us-east-2)
• US West (N. California) Region (us-west-1)
• US West (Oregon) Region (us-west-2)
• Asia Pacific (Hong Kong) Region (ap-east-1)
• Asia Pacific (Jakarta) Region (ap-southeast-3) – only ra3.4xlarge and ra3.16xlarge node types are supported
• Asia Pacific (Mumbai) Region (ap-south-1)
• Asia Pacific (Seoul) Region (ap-northeast-2)
Upgrading to RA3 node types

To upgrade your existing node type to RA3, you have the following options to change the node type:

- Restore from a snapshot – Amazon Redshift uses the most recent snapshot of your DS2 or DC2 cluster and restores it to create a new RA3 cluster. As soon as the cluster creation is complete (usually within minutes), RA3 nodes are ready to run your full production workload. As compute is separate from storage, hot data is brought in to the local cache at fast speeds thanks to a large networking bandwidth. If you restore from the latest DS2 or DC2 snapshot, RA3 preserves hot block information of the DS2 or DC2 workload and populates its local cache with the hottest blocks. For more information, see Restoring a cluster from a snapshot (p. 296).

To keep the same endpoint for your applications and users, you can rename the new RA3 cluster with the same name as the original DS2 or DC2 cluster. To rename the cluster, modify the cluster in the Amazon Redshift console or ModifyCluster API operation. For more information, see Renaming clusters (p. 58) or ModifyCluster API operation in the Amazon Redshift API Reference.

- Elastic resize – resize the cluster using elastic resize. When you use elastic resize to change node type, Amazon Redshift automatically creates a snapshot, creates a new cluster, deletes the old cluster, and renames the new cluster. The elastic resize operation can be run on-demand or can be scheduled to run at a future time. You can quickly upgrade your existing DS2 or DC2 node type clusters to RA3 with elastic resize. For more information, see Elastic resize (p. 52).

The following table shows recommendations when upgrading to RA3 node types. (These recommendations also apply to reserved nodes.)

<table>
<thead>
<tr>
<th>Existing node type</th>
<th>Existing number of nodes</th>
<th>Recommended new node type</th>
<th>Upgrade action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ds2.xlarge</td>
<td>1</td>
<td>ra3.xlplus</td>
<td>Create a 1-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>2</td>
<td>ra3.xlplus</td>
<td>Create a 2-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>3</td>
<td>ra3.xlplus</td>
<td>Create a 2-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>Existing node type</td>
<td>Existing number of nodes</td>
<td>Recommended new node type</td>
<td>Upgrade action</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------</td>
<td>---------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>4</td>
<td>ra3.xlplus</td>
<td>Create a 3-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>5</td>
<td>ra3.xlplus</td>
<td>Create a 4-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>6</td>
<td>ra3.xlplus</td>
<td>Create a 4-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>7</td>
<td>ra3.xlplus</td>
<td>Create a 5-node ra3.xlplus cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>8</td>
<td>ra3.4xlarge</td>
<td>Create a 2-node ra3.4xlarge cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>9</td>
<td>ra3.4xlarge</td>
<td>Create a 3-node ra3.4xlarge cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>10</td>
<td>ra3.4xlarge</td>
<td>Create a 3-node ra3.4xlarge cluster.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>11–128</td>
<td>ra3.4xlarge</td>
<td>Create 1 node of ra3.4xlarge for every 4 nodes of ds2.xlarge.</td>
</tr>
<tr>
<td>ds2.8xlarge</td>
<td>2–15</td>
<td>ra3.4xlarge</td>
<td>Create 2 nodes of ra3.4xlarge for every 1 node of ds2.8xlarge.</td>
</tr>
<tr>
<td>ds2.8xlarge</td>
<td>16–128</td>
<td>ra3.16xlarge</td>
<td>Create 1 node of ra3.16xlarge for every 2 nodes of ds2.8xlarge.</td>
</tr>
<tr>
<td>dc2.8xlarge</td>
<td>2–15</td>
<td>ra3.4xlarge</td>
<td>Create 2 nodes of ra3.4xlarge for every 1 node of dc2.8xlarge.</td>
</tr>
<tr>
<td>dc2.8xlarge</td>
<td>16–128</td>
<td>ra3.16xlarge</td>
<td>Create 1 node of ra3.16xlarge for every 2 nodes of dc2.8xlarge.</td>
</tr>
<tr>
<td>dc2.large</td>
<td>1–4</td>
<td>none</td>
<td>Keep existing dc2.large cluster.</td>
</tr>
<tr>
<td>dc2.large</td>
<td>4–15</td>
<td>ra3.xlplus</td>
<td>Create 3 nodes of ra3.xlplus for every 8 nodes of dc2.large.</td>
</tr>
</tbody>
</table>
Upgrade DS2 reserved nodes to RA3 reserved nodes during elastic resize or snapshot restore

If you have DS2 reserved nodes, you can upgrade them with the RA3 reserved-node upgrade feature, using the Amazon Redshift console or the AWS CLI. On the console, there are a couple of ways you can do this.

One way is to upgrade DS2 reserved nodes to RA3 during an elastic resize. If you have reserved nodes and select RA3 nodes, the console walks you through the reserved-node upgrade process. From a technical standpoint, elastic resize works the same for both reserved nodes and nodes that aren't reserved.

If you change the cluster size from the recommended size, when you configure the elastic resize, RA3 reserved-node upgrade isn't available and doesn't appear on the console. (You can still upgrade DS2 reserved nodes to RA3, but the resize doesn't include RA3 reserved-node upgrade as part of the process.) Also note that the cluster size you want may not be available because of cluster-size limitations for elastic resize. For instance, if you have a 4-node DS2 reserved-node cluster, you may not be able to select a 3-node RA3 cluster. In this case, you can perform a classic resize to get the cluster size you want.

A couple of steps occur after the cluster resize. First, data is migrated to the RA3 cluster. Then, the DS2 reserved-node lease is converted to an RA3 reserved-node lease. Note that the time for data migration can vary, depending on the size of the cluster and whether the resize is elastic or classic. In the case of a classic resize, it's common for data migration to take several hours.

After you start the resize, track progress by viewing messages in Events, available on the Amazon Redshift dashboard. There is an event notification for the resize, and another for the reserved-node upgrade. For information about working with events, see Amazon Redshift events (p. 489). After the resize, the active resized cluster appears on the AWS Management Console. You can also view the converted RA3 reserved-node lease. The source DS2 reserved node(s) may still appear on the console for about a day. You aren't billed for them. Don't delete the source DS2 reserved node until you verify that the RA3 cluster is active and the converted reserved-node lease is generated.

The other way that you can use the RA3 reserved-node upgrade feature is when you restore from a snapshot. If you select the RA3 node type and you have DS2 reserved nodes, you can select the RA3 reserved-node upgrade feature at that point. When you restore from a snapshot, it's restored to an RA3 reserved-node cluster. As mentioned previously, if you choose a cluster size other than the recommended size, the RA3 reserved-node upgrade selection isn't available on the console.

For more information about resizing your cluster and upgrading nodes, see Details of resizing a cluster (p. 56). There, you can find a detailed description of the process and also answers about what

<table>
<thead>
<tr>
<th>Existing node type</th>
<th>Existing number of nodes</th>
<th>Recommended new node type</th>
<th>Upgrade action</th>
</tr>
</thead>
<tbody>
<tr>
<td>dc2.large</td>
<td>16–32</td>
<td>ra3.4xlarge</td>
<td>Create 1 node of ra3.4xlarge for every 8 nodes of dc2.large</td>
</tr>
</tbody>
</table>

1 Extra nodes might be needed depending on workload requirements. Add or remove nodes based on the compute requirements of your required query performance.

2 Clusters with the dc2.large node type are limited to 32 nodes.

The minimum number of nodes for some RA3 node types is 2 nodes. Take this into consideration when creating an RA3 cluster.
happens to the cluster and to the data when you resize. For more detail about the steps in the elastic-
resize process, see Elastic resize (p. 52). For more information about restoring from a snapshot, see
Restoring a cluster from a snapshot (p. 296).

If you have more questions about upgrading reserved nodes to RA3, for instance about upgrading DC2
reserved nodes to RA3, contact AWS Support. For information about pricing of on-demand and reserved
nodes, see Amazon Redshift pricing.

If you have already purchased DS2 reserved nodes, contact AWS for help with converting DS2 reserved
nodes to RA3 reserved nodes. To contact AWS for more information, see Amazon Redshift RA3 instances
with managed storage.

Upgrading from DC1 node types to DC2 node
types

To take advantage of performance improvements, you can upgrade your DC1 clusters to DC2 node types.
Clusters that use the DC2 node type must be launched in a virtual private cloud (EC2-VPC).

If your DC1 cluster is not in a VPC:

1. Create a snapshot of your DC1 cluster. For more information, see Amazon Redshift
   snapshots (p. 292).
2. Create a VPC, or choose an existing VPC in your account. For more information, see Managing clusters
   in a VPC (p. 79).
3. Restore your snapshot to a new DC2 cluster in the VPC. For more information, see Restoring a cluster
   from a snapshot (p. 296).

If your DC1 cluster is already in a VPC, choose one of the following methods:

- Resize your DC1 cluster and change the node type to DC2 as part of the operation. Your cluster is not
  available for a period of time during the resize operation. For more information, see Resizing clusters
  in Amazon Redshift (p. 51).
- Create a snapshot of your DC1 cluster, then restore your snapshot to a DC2 cluster in the VPC. For
  more information, see Restoring a cluster from a snapshot (p. 296).

Consider the following when upgrading from DC1 to DC2 node types.

- DC1 clusters that are 100% full might not upgrade to an equivalent number of DC2 nodes. If more
disk space is needed, you can:
  - Resize to a configuration with more available disk space.
  - Clean up unneeded data by truncating tables or deleting rows.
- DC2 clusters don’t support EC2-Classic networking. If your DC1 cluster is not running in a VPC, create
  one for your DC2 migration. For more information, see Managing clusters in Amazon Redshift (p. 79).
- If you resize the cluster, it might be put into read-only mode for the duration of the operation. For
  more information, see Resizing clusters in Amazon Redshift (p. 51).
- If you have purchased DC1 reserved nodes, you can upgrade your DC1 reserved nodes to DC2 nodes
  for the remainder of your term. For more information about how to change your reservation with the
  AWS CLI, see Upgrading reserved nodes with the AWS CLI (p. 316).
- If you use restore to upgrade from dc1.large to dc2.large, and change the number of nodes, then the
  snapshot must have been created at cluster version 1.0.10013 or later.
Upgrading a DS2 cluster on EC2-Classic to EC2-VPC

Amazon Redshift clusters run in Amazon EC2 instances that are configured for the Amazon Redshift node type and size that you choose. We recommend that you upgrade your cluster on EC2-Classic to launch in a VPC using EC2-VPC for improved performance and security.

To upgrade your DS2 cluster on EC2-Classic to EC2-VPC

1. Create a snapshot of your DS2 cluster. For more information, see Amazon Redshift snapshots (p. 292).
2. Create a VPC, or choose an existing VPC in your account. For more information, see Managing clusters in a VPC (p. 79).
3. Restore your snapshot to a new DS2 cluster in the VPC. For more information, see Restoring a cluster from a snapshot (p. 296).

Region and Availability Zone considerations

Amazon Redshift is available in several AWS Regions. By default, Amazon Redshift provisions your cluster in a randomly selected Availability Zone (AZ) within the AWS Region that you choose. All the cluster nodes are provisioned in the same Availability Zone.

You can optionally request a specific Availability Zone if Amazon Redshift is available in that zone. For example, if you already have an Amazon EC2 instance running in one Availability Zone, you might want to create your Amazon Redshift cluster in the same zone to reduce latency. On the other hand, you might want to choose another Availability Zone for higher availability. Amazon Redshift might not be available in all Availability Zones within an AWS Region.

For a list of supported AWS Regions where you can provision an Amazon Redshift cluster, see Amazon Redshift endpoints in the Amazon Web Services General Reference.

Cluster maintenance

Amazon Redshift periodically performs maintenance to apply upgrades to your cluster. During these updates, your Amazon Redshift cluster isn’t available for normal operations. You have several ways to control how we maintain your cluster. For example, you can control when we deploy updates to your clusters. You can also choose whether your cluster runs the most recently released version, or the version
released previously to the most recently released version. Finally, you have the option to defer non-mandatory maintenance updates for a period of time.

Topics

- Maintenance windows (p. 46)
- Deferring maintenance (p. 47)
- Choosing cluster maintenance tracks (p. 47)
- Managing cluster versions (p. 48)
- Rolling back the cluster version (p. 48)
- Determining the cluster maintenance version (p. 49)

Maintenance windows

Amazon Redshift assigns a 30-minute maintenance window at random from an 8-hour block of time per AWS Region, occurring on a random day of the week (Monday through Sunday, inclusive).

Default maintenance windows

The following list shows the time blocks for each AWS Region from which the default maintenance windows are assigned:

- US East (N. Virginia) Region: 03:00–11:00 UTC
- US East (Ohio) Region: 03:00–11:00 UTC
- US West (N. California) Region: 06:00–14:00 UTC
- US West (Oregon) Region: 06:00–14:00 UTC
- Africa (Cape Town) Region: 20:00–04:00 UTC
- Asia Pacific (Hong Kong) Region: 13:00–21:00 UTC
- Asia Pacific (Jakarta) Region: 15:00–23:00 UTC
- Asia Pacific (Mumbai) Region: 16:30–00:30 UTC
- Asia Pacific (Osaka) Region: 13:00–21:00 UTC
- Asia Pacific (Seoul) Region: 13:00–21:00 UTC
- Asia Pacific (Singapore) Region: 14:00–22:00 UTC
- Asia Pacific (Sydney) Region: 12:00–20:00 UTC
- Asia Pacific (Tokyo) Region: 13:00–21:00 UTC
- Canada (Central) Region: 03:00–11:00 UTC
- China (Beijing) Region: 13:00–21:00 UTC
- China (Ningxia) Region: 13:00–21:00 UTC
- Europe (Frankfurt) Region: 06:00–14:00 UTC
- Europe (Ireland) Region: 22:00–06:00 UTC
- Europe (London) Region: 22:00–06:00 UTC
- Europe (Milan) Region: 21:00–05:00 UTC
- Europe (Paris) Region: 23:00–07:00 UTC
- Europe (Stockholm) Region: 23:00–07:00 UTC
- Middle East (Bahrain) Region: 13:00–21:00 UTC
• South America (São Paulo) Region: 19:00–03:00 UTC

If a maintenance event is scheduled for a given week, it starts during the assigned 30-minute maintenance window. While Amazon Redshift is performing maintenance, it terminates any queries or other operations that are in progress. Most maintenance completes during the 30-minute maintenance window, but some maintenance tasks might continue running after the window closes. If there are no maintenance tasks to perform during the scheduled maintenance window, your cluster continues to operate normally until the next scheduled maintenance window.

You can change the scheduled maintenance window by modifying the cluster, either programmatically or by using the Amazon Redshift console. The window must be at least 30 minutes and not longer than 24 hours. For more information, see Managing clusters using the console (p. 70).

Deferring maintenance

If you need to reschedule your cluster’s maintenance window, you have the option to defer maintenance by up to 45 days. For example, if your cluster’s maintenance window is set to Wednesday 8:30 – 9:00 UTC and you need to have access to your cluster at that time, you can defer the maintenance to a later time period. We will not perform any maintenance on your cluster when you have specified a deferment, unless we need to update hardware.

If we need to update hardware or make other mandatory updates during your period of deferment, we notify you and make the required changes. Your cluster isn’t available during these updates.

If you defer your cluster’s maintenance, the maintenance window following your period of deferment is mandatory. It can’t be deferred.

**Note**

You can’t defer maintenance after it has started.

For more information, see Modifying a cluster (p. 73).

Choosing cluster maintenance tracks

When Amazon Redshift releases a new cluster version, your cluster is updated during its maintenance window. You can control whether your cluster is updated to the most recent approved release or to the previous release.

The maintenance track controls which cluster version is applied during a maintenance window. When Amazon Redshift releases a new cluster version, that version is assigned to the *current* track, and the previous version is assigned to the *trailing* track. To set the maintenance track for the cluster, specify one of the following values:

- **Current** – Use the most current approved cluster version.
- **Trailing** – Use the cluster version before the current version.
- **Preview** – Use the cluster version that contains new features available for preview.

For example, suppose that your cluster is currently running version 1.0.2762 and the Amazon Redshift current version is 1.0.3072. If you set the maintenance track value to **Current**, your cluster is updated to version 1.0.3072 (the next approved release) during the next maintenance window. If you set the maintenance track value to **Trailing**, your cluster isn’t updated until there is a new release after 1.0.3072.

**Preview tracks**

A **Preview** track might not always be available to choose. When you choose a **Preview** track, a track name must also be selected. Preview tracks and its related resources are temporary, have functional
limitations, and might not contain all current Amazon Redshift features available in other tracks. When working with preview tracks:

- Use the new Amazon Redshift console when working with preview tracks. For example, when you create a cluster to use with preview features.
- You can't switch a cluster from one preview track to another.
- You can't switch a cluster to a preview track from a current or trailing track.
- You can't restore from a snapshot created from a different preview track.
- You can only use the preview track when creating a new cluster, or when restoring from a snapshot.
- You can't restore from a snapshot created from a different preview track, or with a cluster maintenance version later than the preview track cluster version. For example, when you restore a cluster to a preview track, you can only use a snapshot created from an earlier cluster maintenance version than that of the preview track.

Switching between maintenance tracks

Changing tracks for a cluster is generally a one-time decision. You should exercise caution in changing tracks. If you change the maintenance track from Trailing to Current, we will update the cluster to the Current track release version during the next maintenance window. However, if you change the cluster's maintenance track to Trailing we won't update your cluster until there is a new release after the Current track release version.

Maintenance tracks and restore

A snapshot inherits the source cluster's maintenance track. If you change the source cluster's maintenance track after you take a snapshot, the snapshot and the source cluster are on different tracks. When you restore from the snapshot, the new cluster will be on the maintenance track that was inherited from the source cluster. You can change the maintenance track after the restore operation completes. Resizing a cluster doesn't affect the cluster's maintenance track.

Managing cluster versions

A maintenance track is a series of releases. You can decide if your cluster is on the Current track or the Trailing track. If you put your cluster on the Current track, it will always be upgraded to the most recent cluster release version during its maintenance window. If you put your cluster on the Trailing track, it will always run the cluster release version that was released immediately before the most recently released version.

The Release status column in the Amazon Redshift console list of clusters indicates whether one of your clusters is available for upgrade.

Rolling back the cluster version

If your cluster is up to date with the latest cluster version, you can choose to roll it back to the previous version.

For detailed information about features and improvements included with each cluster version, see Cluster version history (p. 86).

To roll back to a previous cluster version

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster to roll back.
4. For Actions, choose Roll back cluster version. The Roll back cluster version page appears.
5. If there is a version available for roll back, follow the instructions on the page.
6. Choose Roll back now.

Determining the cluster maintenance version

You can determine the Amazon Redshift engine and database version with the Amazon Redshift console.

To find the version of a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include Cluster performance, Query monitoring, Databases, Datashares, Schedules, Maintenance, and Properties tabs.
3. Choose the Maintenance tab for more details.
4. In the Maintenance section, find Current cluster version.

Note
Although the console displays this information in one field, it’s two parameters in the Amazon Redshift API, ClusterVersion and ClusterRevisionNumber. For more information, see Cluster in the Amazon Redshift API Reference.

Default disk space alarm

When you create an Amazon Redshift cluster, you can optionally configure an Amazon CloudWatch alarm to monitor the average percentage of disk space that is used across all of the nodes in your cluster. We’ll refer to this alarm as the default disk space alarm.

The purpose of default disk space alarm is to help you monitor the storage capacity of your cluster. You can configure this alarm based on the needs of your data warehouse. For example, you can use the warning as an indicator that you might need to resize your cluster. You might resize either to a different node type or to add nodes, or perhaps to purchase reserved nodes for future expansion.

The default disk space alarm triggers when disk usage reaches or exceeds a specified percentage for a certain number of times and at a specified duration. By default, this alarm triggers when the percentage that you specify is reached, and stays at or above that percentage for five minutes or longer. You can edit the default values after you launch the cluster.

When the CloudWatch alarm triggers, Amazon Simple Notification Service (Amazon SNS) sends a notification to specified recipients to warn them that the percentage threshold is reached. Amazon SNS uses a topic to specify the recipients and message that are sent in a notification. You can use an existing Amazon SNS topic; otherwise, a topic is created based on the settings that you specify when you launch the cluster. You can edit the topic for this alarm after you launch the cluster. For more information about creating Amazon SNS topics, see Getting Started with Amazon Simple Notification Service.

After you launch the cluster, you can view and edit the alarm from the cluster’s Status window under CloudWatch Alarms. The name is percentage-disk-space-used-default-<string>. You can open the alarm to view the Amazon SNS topic that it is associated with and edit alarm settings. If you did not select an existing Amazon SNS topic to use, the one created for you is named <clustername>-default-alarms (<recipient>); for example, examplecluster-default-alarms (notify@example.com).
For more information about configuring and editing the default disk space alarm, see Creating a cluster (p. 71) and Creating or editing a disk space alarm (p. 76).

**Note**
If you delete your cluster, the alarm associated with the cluster will not be deleted but it will not trigger. You can delete the alarm from the CloudWatch console if you no longer need it.

## Cluster status

The cluster status displays the current state of the cluster. The following table provides a description for each cluster status.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>available</td>
<td>The cluster is running and available.</td>
</tr>
<tr>
<td>available, prep-for-resize</td>
<td>The cluster is being prepared for elastic resize. The cluster is running and available for read and write queries, but cluster operations, such as creating a snapshot, are not available.</td>
</tr>
<tr>
<td>available, resize-cleanup</td>
<td>An elastic resize operation is completing data transfer to the new cluster nodes. The cluster is running and available for read and write queries, but cluster operations, such as creating a snapshot, are not available.</td>
</tr>
<tr>
<td>cancelling-resize</td>
<td>The resize operation is being cancelled.</td>
</tr>
<tr>
<td>creating</td>
<td>Amazon Redshift is creating the cluster. For more information, see Creating a cluster (p. 71).</td>
</tr>
<tr>
<td>deleting</td>
<td>Amazon Redshift is deleting the cluster. For more information, see Deleting a cluster (p. 73).</td>
</tr>
<tr>
<td>final-snapshot</td>
<td>Amazon Redshift is taking a final snapshot of the cluster before deleting it. For more information, see Deleting a cluster (p. 73).</td>
</tr>
<tr>
<td>hardware-failure</td>
<td>The cluster suffered a hardware failure. If you have a single-node cluster, the node cannot be replaced. To recover your cluster, restore a snapshot. For more information, see Amazon Redshift snapshots (p. 292).</td>
</tr>
<tr>
<td>incompatible-hsm</td>
<td>Amazon Redshift cannot connect to the hardware security module (HSM). Check the HSM configuration between the cluster and HSM. For more information, see Encryption for Amazon Redshift using hardware security modules (p. 326).</td>
</tr>
<tr>
<td>incompatible-network</td>
<td>There is an issue with the underlying network configuration. Make sure that the VPC in which you launched the cluster exists and its settings are correct. For more information, see Managing clusters in a VPC (p. 79).</td>
</tr>
<tr>
<td>incompatible-parameters</td>
<td>There is an issue with one or more parameter values in the associated parameter group, and the parameter value or values cannot be applied. Modify the parameter group and update any invalid values. For more information, see Amazon Redshift parameter groups (p. 271).</td>
</tr>
<tr>
<td>incompatible-restore</td>
<td>There was an issue restoring the cluster from the snapshot. Try restoring the cluster again with a different snapshot. For more information, see Amazon Redshift snapshots (p. 292).</td>
</tr>
</tbody>
</table>
Overview of managing clusters in Amazon Redshift

After your cluster is created, there are several operations you can perform on it. The operations include resizing, pausing, resuming, renaming, and deleting.

Resizing clusters in Amazon Redshift

As your data warehousing capacity and performance needs change or grow, you can resize your cluster to make the best use of the computing and storage options that Amazon Redshift provides.

You can use elastic resize to scale your cluster by changing the node type and number of nodes. We recommend using elastic resize, which typically completes in minutes. If your new node configuration isn't available through elastic resize, you can use classic resize. Classic resize takes more time to complete, because it involves provisioning a new cluster and copying data blocks to it. In contrast, elastic resize redistributes data slices, which requires fewer resources.

To resize your cluster, use one of the following approaches:

- **Elastic resize** – Use it to change the node type, number of nodes, or both. Elastic resize works quickly by changing or adding nodes to your existing cluster. If you change only the number of nodes, queries are temporarily paused and connections are held open, if possible. Typically, elastic resize takes 10–15 minutes. During the resize operation, the cluster is read-only.

  We recommend using elastic resize whenever possible, because it completes much more quickly than classic resize.

- **Classic resize** – Use it to change the node type, number of nodes, or both. Classic resize provisions a new cluster and copies the data from the source cluster to the new cluster. Choose this option only when you are resizing to a configuration that isn't available through elastic resize, because it takes considerably more time to complete. An example of when to use it is when resizing to or from a single-node cluster. During the resize operation, the cluster is read-only. Classic resize can take several hours.

---

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modifying</td>
<td>Amazon Redshift is applying changes to the cluster. For more information, see Modifying a cluster (p. 73).</td>
</tr>
<tr>
<td>paused</td>
<td>The cluster is paused. For more information, see Pausing and resuming clusters (p. 57).</td>
</tr>
<tr>
<td>rebooting</td>
<td>Amazon Redshift is rebooting the cluster. For more information, see Rebooting a cluster (p. 74).</td>
</tr>
<tr>
<td>renaming</td>
<td>Amazon Redshift is applying a new name to the cluster. For more information, see Renaming clusters (p. 58).</td>
</tr>
<tr>
<td>resizing</td>
<td>Amazon Redshift is resizing the cluster. For more information, see Resizing a cluster (p. 74).</td>
</tr>
<tr>
<td>rotating-keys</td>
<td>Amazon Redshift is rotating encryption keys for the cluster. For more information, see Encryption key rotation in Amazon Redshift (p. 327).</td>
</tr>
<tr>
<td>storage-full</td>
<td>The cluster has reached its storage capacity. Resize the cluster to add nodes or to choose a different node size. For more information, see Resizing a cluster (p. 74).</td>
</tr>
<tr>
<td>updating-hsm</td>
<td>Amazon Redshift is updating the HSM configuration.</td>
</tr>
</tbody>
</table>
to several days, or longer, depending on the amount of data to transfer and the difference in cluster size and computing resources.

If your configuration warrants performing a classic resize, you can minimize production impact by making a copy of your existing cluster and subsequently resizing the copy. *Snapshot, restore, and resize* (p. 55) describes the steps.

You can resize (both elastic resize and classic resize) your cluster on a schedule. When you use the new Amazon Redshift console, you can set up a schedule to resize your cluster. For more information, see *Resizing a cluster* (p. 74). You can also use the AWS CLI or Amazon Redshift API operations to schedule a resize. For more information, see `create-scheduled-action` in the *AWS CLI Command Reference* or `CreateScheduledAction` in the *Amazon Redshift API Reference*.

**Topics**
- Elastic resize (p. 52)
- Classic resize (p. 54)
- Snapshot, restore, and resize (p. 55)
- Details of resizing a cluster (p. 56)

**Elastic resize**

Elastic resize is the fastest method to resize a cluster. You can use elastic resize to add or remove nodes and change node types.

When a cluster is resized using elastic resize with the same node type, it automatically redistributes the data to the new nodes. Because it doesn't create a new cluster in this scenario, the elastic resize operation completes quickly, usually in a few minutes. You might notice a slight increase in execution time for some queries while the data is redistributed in the background. An elastic resize operation occurs in the following stages:

1. Elastic resize takes a cluster snapshot.

   The snapshot that elastic resize creates includes no-backup tables (p. 295). If your cluster doesn't have a recent snapshot because you disabled automated snapshots, the backup operation takes longer. To minimize the time before the resize operation begins, we recommend that you enable automated snapshots or create a manual snapshot before starting an elastic resize. When you start an elastic resize and a snapshot operation is currently in progress, then elastic resize might fail if the snapshot operation doesn't complete within a few minutes. For more information, see *Amazon Redshift snapshots* (p. 292).

2. The cluster is temporarily unavailable while elastic resize migrates cluster metadata.

   This stage is very short, just a few minutes at most. Amazon Redshift holds session connections and queries remain queued. Some sessions and queries might time out.

3. Session connections are reinstated and queries resume.

4. Elastic resize redistributes data to the node slices in the background.

   The cluster is available for read and write operations, but some queries might take longer to run.

When a cluster is resized using elastic resize to change the node type, a snapshot is created. A new cluster is provisioned for you with the latest data from the snapshot. The cluster is temporarily unavailable for writes when the data is transferred to the new cluster. It is available for reads. The new cluster is populated in the background. After the new cluster is fully populated, queries should reach optimal performance. When the resize process nears completion, Amazon Redshift updates the endpoint of the new cluster, and all connections to the original cluster are terminated.
After the resize completes, Amazon Redshift sends an event notification. You can connect to the new cluster and resume running read and write queries.

If you have reserved nodes, for example DS2 reserved nodes, you can upgrade to RA3 reserved nodes. You can do this when you use the console to restore from a snapshot or to perform an elastic resize. You can use the console to guide you through this process. For more information about upgrading to RA3 nodes, see Upgrading to RA3 node types.

To monitor the progress of a resize operation using the Amazon Redshift console, choose CLUSTERS, then choose the cluster being resized to see the details.

You can't use elastic resize on single-node clusters.

To run an elastic resize on a cluster that is transferring data from a shared snapshot, at least one backup must be available for the cluster. You can view your backups on the Amazon Redshift console snapshots list, the describe-cluster-snapshots CLI command, or the DescribeClusterSnapshots API operation.

Elastic resize doesn't sort tables or reclaim disk space, so it isn't a substitute for a vacuum operation. A classic resize copies tables to a new cluster, so it can reduce the need to vacuum. For more information, see Vacuuming tables.

Elastic resize has the following constraints:

- Elastic resize is available only for clusters that use the EC2-VPC platform. For more information, see Use EC2-VPC when you create your cluster (p. 38).
- Make sure that your new node configuration has enough storage for existing data. Even when you add nodes, your new configuration might not have enough storage because of the way that data is redistributed.
- The possible configurations of node number and type that you can resize to is determined by the number of nodes in the original cluster and the target node type of the resized cluster. To determine the possible configurations available, you can use the console. Or you can use the describe-node-configuration-options AWS CLI command with the action-type resize-cluster option. For more information about the resizing using the Amazon Redshift console, see Resizing a cluster (p. 74).

The following example CLI command describes the configuration options available. In this example, the cluster named mycluster is a dc2.large 8-node cluster.

```bash
aws redshift describe-node-configuration-options --cluster-identifier mycluster --region eu-west-1 --action-type resize-cluster
```

This command returns an option list with recommended node types, number of nodes, and disk utilization for each option. The configurations returned can vary based on the specific input cluster. You can choose one of the returned configurations when you specify the options of the resize-cluster CLI command.

- Elastic resize has limits on the nodes that you can add to a cluster. For example, a dc2 cluster supports elastic resize up to double the number of nodes. To illustrate, you can add a node to a 4-node dc2.8xlarge cluster to make it a 5-node cluster, or add more nodes until you reach 8.

  With some ra3 node types, you can increase the number of nodes up to four times the existing count. Specifically, suppose that your cluster consists of ra3.4xlarge or ra3.16xlarge nodes. You can then use elastic resize to increase the number of nodes in an 8-node cluster to 32. Or you can pick a value below the limit. If your cluster has ra3.xlplus nodes, the limit is double.

  All ra3 node types support a decrease in the number of nodes to a quarter of the existing count. For example, you can decrease the size of a cluster with ra3.4xlarge nodes from 12 nodes to 3, or to a number above the minimum.
The following table lists growth and reduction limits for each node type that supports elastic resize.

<table>
<thead>
<tr>
<th>Node type</th>
<th>Growth limit</th>
<th>Reduction limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ra3.16xlarge</td>
<td>4x (from 4 to 16 nodes, for example)</td>
<td>To one-quarter of the number (from 16 to 4 nodes, for example)</td>
</tr>
<tr>
<td>ra3.4xlarge</td>
<td>4x</td>
<td>To one-quarter of the number</td>
</tr>
<tr>
<td>ra3.xlplus</td>
<td>2x (from 4 to 8 nodes, for example)</td>
<td>To one-quarter of the number</td>
</tr>
<tr>
<td>dc2.8xlarge</td>
<td>2x</td>
<td>To one-half the number (from 16 to 8 nodes, for example)</td>
</tr>
<tr>
<td>dc2.large</td>
<td>2x</td>
<td>To one-half the number</td>
</tr>
<tr>
<td>ds2.8xlarge</td>
<td>2x</td>
<td>To one-half the number</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>2x</td>
<td>To one-half the number</td>
</tr>
</tbody>
</table>

**Classic resize**

With the classic resize operation, your data is copied in parallel from the compute node or nodes in your source cluster to the compute node or nodes in the target cluster. The time that it takes to resize depends on the amount of data and the number of nodes in the smaller cluster. It can take anywhere from a couple of hours to a couple of days or longer.

The duration of a classic resize varies based several factors, including:

- The workload on the source cluster.
- The number and size of the tables being transferred.
- How evenly data is distributed across the compute nodes and slices.
- The node configuration in the source and target clusters.

When you start the resize operation, Amazon Redshift puts the existing cluster into read-only mode until the resize finishes. During this time, you can only run queries that read from the database. You can't run any queries that write to the database, including read-write queries. For more information, see [Write and read-write operations](https://docs.aws.amazon.com/redshift/latest/dg/). For more information, see [Write and read-write operations](https://docs.aws.amazon.com/redshift/latest/dg/).

**Note**

To resize with minimal production impact, you can use the steps in the following section, **Snapshot, restore, and resize (p. 55)**. You can use these steps to create a copy of your cluster, resize the copy, and then switch the connection endpoint to the resized cluster when the resize is complete.

Both the classic resize approach and the snapshot and restore approach copy user tables and data to the new cluster; they don't retain system tables and data. With either classic resize or snapshot and restore, if you have enabled audit logging in your source cluster, you can continue to access the logs in Amazon S3. With these approaches, you can still access the logs after you delete the source cluster. You can keep or delete these logs as your data policies specify. Elastic resize retains the system log tables.

After Amazon Redshift puts the source cluster into read-only mode, it provisions a new cluster, the target cluster. It does so using the information that you specify for the node type, cluster type, and number of nodes. Then Amazon Redshift copies the data from the source cluster to the target cluster. When this is
complete, all connections switch to use the target cluster. If you have any queries in progress at the time this switch happens, your connection is lost and you must restart the query on the target cluster. You can view the resize progress on the Amazon Redshift console.

Amazon Redshift doesn't sort tables during a resize operation, so the existing sort order is maintained. When you resize a cluster, Amazon Redshift distributes the database tables to the new nodes based on their distribution styles and runs an ANALYZE command to update statistics. Rows that are marked for deletion aren't transferred, so you need to run only a VACUUM command if your tables need to be resorted. For more information, see Vacuuming tables in the Amazon Redshift Database Developer Guide.

You can cancel a classic resize operation before it completes by choosing Cancel resize from the cluster details in the Amazon Redshift console. The amount of time it takes to cancel a resize depends on the stage of the resize operation when you cancel. The cluster isn't available until the cancel resize operation completes. If the resize operation is in the final stage, you can't cancel the operation.

**Snapshot, restore, and resize**

As described in the preceding section, the time it takes to resize a cluster with the classic resize operation depends heavily on the amount of data in the cluster.

Elastic resize (p. 52) is the fastest method to resize an Amazon Redshift cluster. If elastic resize isn't an option for you and you require near-constant write access to your cluster, use the snapshot and restore operations with classic resize as described in the following section. This approach requires that any data that is written to the source cluster after the snapshot is taken must be copied manually to the target cluster after the switch. Depending on how long the copy takes, you might need to repeat this several times until you have the same data in both clusters. Then you can make the switch to the target cluster. This process might have a negative impact on existing queries until the full set of data is available in the target cluster. However, it minimizes the amount of time that you can't write to the database.

The snapshot, restore, and classic resize approach uses the following process:

1. Take a snapshot of your existing cluster. The existing cluster is the source cluster.
2. Note the time that the snapshot was taken. Doing this means that you can later identify the point when you need to rerun extract, transact, load (ETL) processes to load any post-snapshot data into the target database.
3. Restore the snapshot into a new cluster. This new cluster is the target cluster. Verify that the sample data exists in the target cluster.
4. Resize the target cluster. Choose the new node type, number of nodes, and other settings for the target cluster.
5. Review the loads from your ETL processes that occurred after you took a snapshot of the source cluster. Be sure to reload the same data in the same order into the target cluster. If you have ongoing data loads, repeat this process several times until the data is the same in both the source and target clusters.
6. Stop all queries running on the source cluster. To do this, you can reboot the cluster, or you can log on as a superuser and use the `PG_CANCEL_BACKEND` and the `PG_TERMINATE_BACKEND` commands. Rebooting the cluster is the easiest way to make sure that the cluster is unavailable.
7. Rename the source cluster. For example, rename it from `examplecluster` to `examplecluster-source`.
8. Rename the target cluster to use the name of the source cluster before the rename. For example, rename the target cluster from preceding to `examplecluster`. From this point on, any applications that use the endpoint containing `examplecluster` connect to the target cluster.
9. Delete the source cluster after you switch to the target cluster, and verify that all processes work as expected.

Alternatively, you can rename the source and target clusters before reloading data into the target cluster. This approach works if you don't have a requirement that any dependent systems and reports
be immediately up to date with those for the target cluster. In this case, step 6 moves to the end of the process described preceding.

The rename process is only required if you want applications to continue using the same endpoint to connect to the cluster. If you don’t require this, you can instead update any applications that connect to the cluster to use the endpoint of the target cluster without renaming the cluster.

There are a couple of benefits to reusing a cluster name. First, you don’t need to update application connection strings because the endpoint doesn’t change, even though the underlying cluster changes. Second, related items such as Amazon CloudWatch alarms and Amazon Simple Notification Service (Amazon SNS) notifications are tied to the cluster name. This tie means that you can continue using the same alarms and notifications that you set up for the cluster. This continued use is primarily a concern in production environments where you want the flexibility to resize the cluster without reconfiguring related items, such as alarms and notifications.

**Details of resizing a cluster**

If your storage and performance needs change after you initially provision your cluster, you can resize your cluster. You can scale the cluster in or out by adding or removing nodes. Additionally, you can scale the cluster up or down by specifying a different node type.

For example, you can add more nodes, change node types, change a single-node cluster to a multi-node cluster, or change a multi-node cluster to a single-node cluster. However, you must ensure that the resulting cluster is large enough to hold the data that you currently have or else the resize will fail. When using the API, you have to specify the node type, node size, and the number of nodes even if you only change one of the properties.

The following describes the resize process:

1. When you initiate the resize process, Amazon Redshift sends an event notification that acknowledges the resize request and starts to provision the new (target) cluster.
2. When the new (target) cluster is provisioned, Amazon Redshift sends an event notification that the resize has started, then restarts your existing (source) cluster in read-only mode. The restart terminates all existing connections to the cluster. All uncommitted transactions (including COPY) are rolled back. While the cluster is in read-only mode, you can run read queries but not write queries.
3. Amazon Redshift starts to copy data from the source cluster to the target cluster.
4. When the resize process nears completion, Amazon Redshift updates the endpoint of the target cluster, and all connections to the source cluster are terminated.
5. After the resize completes, Amazon Redshift sends an event notification that the resize has completed. You can connect to the target cluster and resume running read and write queries.

When you resize your cluster, it will remain in read-only mode until the resize completes. You can view the resize progress on the Amazon Redshift console. The time it takes to resize a cluster depends on the amount of data in each node. Typically, the resize process varies from a couple of hours to a day, although clusters with larger amounts of data might take even longer. This is because the data is copied in parallel from each node on the source cluster to the nodes in the target cluster. For more information about resizing clusters, see Resizing a cluster (p. 74).

Amazon Redshift distributes the database tables to the new compute nodes based on their distribution styles and runs an **ANALYZE** to update statistics. Rows that are marked for deletion aren’t transferred, so you need to run a **VACUUM** only if your tables need to be resorted. For more information, see *Vacuuming tables* in the Amazon Redshift Database Developer Guide.

If your cluster is public and is in a VPC, it keeps the same Elastic IP address (EIP) for the leader node after resizing. If your cluster is private and is in a VPC, it keeps the same private IP address for the leader node.
after resizing. If your cluster isn't in a VPC, a new public IP address is assigned for the leader node as part of the resize operation.

To get the leader node IP address for a cluster, use the dig utility, as shown following.

```bash
dig mycluster.abcd1234.us-west-2.redshift.amazonaws.com
```

The leader node IP address is at the end of the ANSWER SECTION in the results, as shown following.

---

**Pausing and resuming clusters**

If you have a cluster that only needs to be available at specific times, you can pause the cluster and later resume it. While the cluster is paused, on-demand billing is suspended. Only the cluster's storage incurs charges. For more information about pricing, see the Amazon Redshift pricing page.

When you pause a cluster, Amazon Redshift creates a snapshot, begins terminating queries, and puts the cluster in a pausing state. If you delete a paused cluster without requesting a final snapshot, then you can't restore the cluster. You can't cancel or roll back a pause or resume operation after it's initiated.

You can pause and resume a cluster on the Amazon Redshift console, with the AWS CLI, or with Amazon Redshift API operations.

You can schedule actions to pause and resume a cluster. When you use the new Amazon Redshift console to create a recurring schedule to pause and resume, then two scheduled actions are created for the date range that you choose. The scheduled action names are suffixed with `-pause` and `-resume`. The total length of the name must fit within the maximum size of a scheduled action name.

You can't pause the following types of clusters:

- EC2-Classic clusters.
• Clusters that are not active, for example a cluster that is currently modifying.
• Hardware security module (HSM) clusters.
• Clusters that have automated snapshots disabled.

When deciding to pause a cluster, consider the following:

• Connections or queries to the cluster aren't available.
• You can't see query monitoring information of a paused cluster on the Amazon Redshift console.
• You can't modify a paused cluster. Any scheduled actions on the cluster aren't done. These include creating snapshots, resizing clusters, and cluster maintenance operations.
• Hardware metrics aren't created. Update your CloudWatch alarms if you have alarms set on missing metrics.
• You can't copy the latest automated snapshots of a paused cluster to manual snapshots.
• While a cluster is pausing, it can't be resumed until the pause operation is complete.
• When you pause a cluster, billing is suspended. However, the pause operation typically completes within 15 minutes, depending upon the size of the cluster.
• Audit logs are archived and not restored on resume.
• No-backup tables on the cluster are not restored on resume. For more information about no-backup tables, see Excluding tables from snapshots (p. 295).

When you resume a cluster, consider the following:

• The cluster version of the resumed cluster is updated to the maintenance version based on the maintenance window of the cluster.
• If you delete the subnet associated with a paused cluster, you might have an incompatible network. In this case, restore your cluster from the latest snapshot.
• If you delete an Elastic IP address while the cluster is paused, then a new Elastic IP address is requested.
• If Amazon Redshift can't resume the cluster with its previous elastic network interface, then Amazon Redshift tries to allocate a new one.
• When you resume a cluster, your node IP addresses might change. You might need to update your VPC settings to support these new IP addresses for features like COPY from Secure Shell (SSH) or COPY from Amazon EMR.
• If you try to resume a cluster that isn’t paused, the resume operation returns an error. If the resume operation is part of a scheduled action, modify or delete the scheduled action to prevent future errors.
• Depending upon the size of the cluster, it can take several minutes to resume a cluster before queries can be processed. In addition, query performance can be impacted for some period of time while the cluster is being re-hydrated after resume completes.

Renaming clusters

You can rename a cluster if you want the cluster to use a different name. Because the endpoint to your cluster includes the cluster name (also referred to as the *cluster identifier*), the endpoint changes to use the new name after the rename finishes. For example, if you have a cluster named `examplecluster` and rename it to `newcluster`, the endpoint changes to use the `newcluster` identifier. Any applications that connect to the cluster must be updated with the new endpoint.

You might rename a cluster if you want to change the cluster to which your applications connect without having to change the endpoint in those applications. In this case, you must first rename the original cluster and then change the second cluster to reuse the name of the original cluster before the rename. Doing this is necessary because the cluster identifier must be unique within your account and
region, so the original cluster and second cluster cannot have the same name. You might do this if you restore a cluster from a snapshot and don’t want to change the connection properties of any dependent applications.

**Note**
- If you delete the original cluster, you are responsible for deleting any unwanted cluster snapshots.

When you rename a cluster, the cluster status changes to *renaming* until the process finishes. The old DNS name that was used by the cluster is immediately deleted, although it could remain cached for a few minutes. The new DNS name for the renamed cluster becomes effective within about 10 minutes. The renamed cluster is not available until the new name becomes effective. The cluster will be rebooted and any existing connections to the cluster will be dropped. After this completes, the endpoint will change to use the new name. For this reason, you should stop queries from running before you start the rename and restart them after the rename finishes.

Cluster snapshots are retained, and all snapshots associated with a cluster remain associated with that cluster after it is renamed. For example, suppose that you have a cluster that serves your production database and the cluster has several snapshots. If you rename the cluster and then replace it in the production environment with a snapshot, the cluster that you renamed still has those existing snapshots associated with it.

Amazon CloudWatch alarms and Amazon Simple Notification Service (Amazon SNS) event notifications are associated with the name of the cluster. If you rename the cluster, you need to update these accordingly. You can update the CloudWatch alarms in the CloudWatch console, and you can update the Amazon SNS event notifications in the Amazon Redshift console on the **Events** pane. The load and query data for the cluster continues to display data from before the rename and after the rename. However, performance data is reset after the rename process finishes.

For more information, see [Modifying a cluster](p. 73).

### Shutting down and deleting clusters

You can shut down your cluster if you want to stop it from running and incurring charges. When you shut it down, you can optionally create a final snapshot. If you create a final snapshot, Amazon Redshift will create a manual snapshot of your cluster before shutting it down. You can later restore that snapshot if you want to resume running the cluster and querying data.

If you no longer need your cluster and its data, you can shut it down without creating a final snapshot. In this case, the cluster and data are deleted permanently. For more information about shutting down and deleting clusters, see [Deleting a cluster](p. 73).

Regardless of whether you shut down your cluster with a final manual snapshot, all automated snapshots associated with the cluster will be deleted after the cluster is shut down. Any manual snapshots associated with the cluster are retained. Any manual snapshots that are retained, including the optional final snapshot, are charged at the Amazon Simple Storage Service storage rate if you have no other clusters running when you shut down the cluster, or if you exceed the available free storage that is provided for your running Amazon Redshift clusters. For more information about snapshot storage charges, see the Amazon Redshift pricing page.

### Working with AQUA (Advanced Query Accelerator)

AQUA (Advanced Query Accelerator) is an analytics query accelerator for Amazon Redshift that uses custom-designed hardware to speed up queries that scan large datasets.

AQUA is a cost-effective addition to Amazon Redshift–managed storage that is optimized for secure, transactional, multitenant access and high-throughput analytic queries. It uses high speed Non-Volatile
Memory Express (NVMe) solid-state storage elements and Nitro-based acceleration for compression and encryption. It uses acceleration based on a field-programmable gate array (FPGA) to push as much computation as possible into the storage layer. These components are connected together in a unique way, allowing data scanning without using a traditional CPU. At the same time, these components allow intermediate results to be aggregated in high-speed memory. AQUA is a data cache and maintains high-speed connections to Redshift managed storage.

You don't need to change your databases or applications to use AQUA. Amazon Redshift identifies the scan portions of queries that can benefit from acceleration and push them to AQUA for processing. AQUA automatically optimizes query performance on subsets of the data that require extensive scans, filters, and aggregation. With this approach, you can use AQUA to run queries that scan, filter, and aggregate large datasets. AQUA excels at queries that require processing-intensive scans, filters, and aggregation such as those that contain LIKE and SIMILAR TO predicates.

AQUA supports authentication, encryption, isolation, and compliance to keep your data at rest and data in transit secure. For more information about data security and AQUA, see the following:

- Data protection in Amazon Redshift (p. 322)
- Encryption at rest (p. 323)
- Encryption in transit (p. 331)

For more information about AQUA, see When does Amazon Redshift use AQUA to run queries? (p. 61)

AQUA is available on clusters with ra3.xlplus, ra3.4xlarge, and ra3.16xlarge node types.

AQUA is available with release version 1.0.24421 or later in the following AWS Regions:

- US East (N. Virginia) Region (us-east-1)
- US East (Ohio) Region (us-east-2)
- US West (N. California) Region (us-west-1)
- US West (Oregon) Region (us-west-2)
- Asia Pacific (Mumbai) Region (ap-south-1)
- Asia Pacific (Seoul) Region (ap-northeast-2)
- Asia Pacific (Singapore) Region (ap-southeast-1)
- Asia Pacific (Sydney) Region (ap-southeast-2)
- Asia Pacific (Tokyo) Region (ap-northeast-1)
- Europe (Frankfurt) Region (eu-central-1)
- Europe (Ireland) Region (eu-west-1)
- Europe (London) Region (eu-west-2)
- Europe (Stockholm) Region (eu-north-1)

You can activate and manage AQUA for Amazon Redshift clusters on the Amazon Redshift console, with the AWS CLI, or with Amazon Redshift API operations. You can do so when you create a cluster, restore a cluster from a snapshot, or modify an existing cluster. When you activate an existing cluster, make sure to reboot your cluster for the change to take effect.

To activate AQUA on the Amazon Redshift console, navigate to your cluster and choose Configure AQUA for Actions. To view a previously defined AQUA configuration for your cluster, navigate to your cluster, and view the General information section, AQUA information.

You have the following choices when you configure AQUA:

- Turn on – You choose to activate AQUA. AQUA can only be activated in certain AWS Regions and for ra3.xlplus, ra3.4xlarge, and ra3.16xlarge node types.
• **Turn off** – You choose not to activate AQUA.
• **Automatic** – Amazon Redshift determines whether to use AQUA. This is the default. Currently, AQUA isn't activated with this option, but this behavior is subject to change.

You can also activate AQUA for the duration of a session. For more information, see [activate_aqua](#) in the [Amazon Redshift Database Developer Guide](#).

**Managing AQUA using the AWS CLI**

You can use the following AWS CLI commands to activate and manage AQUA. For more information, see the [AWS CLI Command Reference](#).

- `create-cluster`
- `restore-from-cluster-snapshot`
- `modify-aqua-configuration`

**Managing AQUA using Amazon Redshift API operations**

You can use the following Amazon Redshift API operations to activate and manage AQUA. For more information, see the [Amazon Redshift API Reference](#).

- `CreateCluster`
- `RestoreFromClusterSnapshot`
- `ModifyAquaConfiguration`

**When does Amazon Redshift use AQUA to run queries?**

For each query that scans a table, Amazon Redshift determines whether the scan operation is sent to AQUA or run locally on the Amazon Redshift cluster. Scan and aggregation operations are sent to AQUA when they contain at least one predicate that contains a LIKE or SIMILAR TO expression that AQUA supports. If a scan operation is sent to AQUA, then the entire operation runs on AQUA (not only the LIKE or SIMILAR TO processing).

For example, the following query includes the predicates `str1 LIKE 'a%'` and `num1 > 10` that are sent to AQUA.

```sql
select num2 from tbl where str1 LIKE 'a%' and num1 > 10 GROUP BY num2;
```

In this case, AQUA scans the table `tbl`, filters by the predicates, groups the results by `num2`, and returns the results to Amazon Redshift.

In the following example, Amazon Redshift determines which portions of the query are better run in AQUA and which to run locally on the Amazon Redshift cluster. In this case, there is a LIKE predicate on the `customer_name` column that is used to filter the rows returned. The scan operation is sent to AQUA to run that part of the query. Amazon Redshift receives only the results of the scan, which Amazon Redshift then uses to perform a join locally and complete the query.

```sql
select c.customer_name, p.prod_name, sum(revenue) from orders o
```
join customers c ON c.id = o.customer_id
join products p ON p.id = o.product_id
where c.customer_name LIKE '%Amazon%Web%'
order by sum(revenue);

For more information about LIKE and SIMILAR TO, see LIKE and SIMILAR TO in the Amazon Redshift Database Developer Guide.

Considerations for AQUA

The following types of SQL queries are currently not supported by AQUA. Queries that include them run locally on your Amazon Redshift cluster.

- Queries that perform writes, such as, INSERT, UPDATE, DELETE, CREATE TABLE AS, COPY, and UNLOAD. For more information, see SQL Commands in the Amazon Redshift Database Developer Guide.
- SELECT queries without a predicate, for example the following.

```sql
select * from tbl1
```

- Python user-defined functions (UDFs) in scans, including scans on complex views that are defined by Python UDFs.
- Queries that use the following metacharacters in LIKE and SIMILAR TO predicates:
  - A nonconstant regular expression literal in a LIKE expression.
  - Either of two alternatives, for example the following.

```sql
SIMILAR TO '%(cat|dog)%'
```

- Multiple instances of single-character wildcards in VARCHAR columns, for example the following.

```sql
SIMILAR TO 'ab.cd.ef'
```

- Unicode escape sequence, hex-characters, and octal-characters in the match string, for example the following.

```sql
SIMILAR TO '%ab\uc382'

SIMILAR TO '%ab\0000c382'

SIMILAR TO '%ab\x88'

SIMILAR TO '%ab\127'
```

- Repetition metacharacters such as "\*", "\+", "\?", \{m,n}\ applied on a pattern within () and on multibyte characters, for example the following.

```sql
SIMILAR TO 'abc(def)*'

SIMILAR TO 'abc#+'
```

- Multibyte characters within a bracket expression [...], for example the following.

```sql
SIMILAR TO 'abc[#de]'.
• Queries where string functions, such as LOWER, UPPER, LEFT, RIGHT, are applied to the output of a LIKE or SIMILAR TO scan. For example, the following excerpt shows a LOWER function.

```
LOWER(a) LIKE "%cat%"
```

Instead, try to rewrite the query to use case-insensitive ILIKE without LOWER.

```
ILIKE "%cat%"
```

**How to determine if AQUA was used**

The following SQL shows segments of query execution from `SVL_QUERY_SUMMARY` that ran on AQUA.

```
select * from svl_query_summary where label ~ 'Aqua' limit 100;
```

If you have superuser access, you can also join the `STL_QUERY` and `STL_QUERYTEXT` views to see the SQL statements that correspond to queries run by AQUA. The explain plan that is generated by the `EXPLAIN` statement doesn't show if a query used AQUA.

**Managing usage limits in Amazon Redshift**

You can define limits to monitor and control your usage and associated cost of some Amazon Redshift features. You can create daily, weekly, and monthly usage limits, and define actions that Amazon Redshift automatically takes if those limits are reached. Actions include such things as logging an event to a system table to record usage exceeding your defined limits. Other possible actions include raising alerts with Amazon SNS and Amazon CloudWatch to notify an administrator and disabling further usage to control costs.

You can define usage limits for each cluster. After your cluster is created, you can define usage limits for the following features:

• Amazon Redshift Spectrum
• Amazon Redshift Concurrency Scaling
• Amazon Redshift cross-Region datasharing

Usage limits are available with release version 1.0.14677 or later in the AWS Regions where Amazon Redshift Spectrum and Amazon Redshift Concurrency Scaling are available.

A Redshift Spectrum limit specifies the threshold of the total amount of data scanned in 1-TB increments. A concurrency scaling limit specifies the threshold of the total amount of time used by concurrency scaling in 1-minute increments. A cross-Region datasharing limit specifies the threshold of the total amount of data scanned in 1-TB increments.

A limit can be specified for a daily, weekly, or monthly period (using UTC to determine the start and end of the period). If you create a limit in the middle of a period, then the limit is measured from that point to the end of the period. For example, if you create a monthly limit on March 15, then the first monthly period is measured from March 15 through March 31.

You can define multiple usage limits for each feature. Each limit can have a different action. Possible actions include the following:

• **Log to system table** – This is the default action. Information is logged to the `STL_USAGE_CONTROL` table. Logging is helpful when evaluating past usage and in deciding on future usage limits. For
more information about what is logged, see STL_USAGE_CONTROL in the Amazon Redshift Database Developer Guide.

- **Alert** – Amazon Redshift emits CloudWatch metrics for available and consumed usage. You can define up to three usage limits for each feature. If you enable the alert action using the Amazon Redshift console, a CloudWatch alarm is automatically created on these metrics. You can optionally attach an Amazon SNS subscription to that alarm. If you are using an AWS CLI or API operation, make sure that you create the CloudWatch alarm manually. When the threshold is reached, events are also logged to a system table.

- **Disable feature** – When the threshold is reached, Amazon Redshift disables the feature until the quota is refreshed for the next time period (daily, weekly, or monthly). Only one limit for each feature can have the disable action. Events are also logged to a system table, and alerts can be emitted.

Usage limits persist until the usage limit definition itself or the cluster is deleted.

You can define and manage usage limits with the new Amazon Redshift console, the AWS CLI, or with Amazon Redshift API operations. To define a limit on the Amazon Redshift console, navigate to your cluster and choose Configure usage limit for Actions. To view previously defined usage limits for your cluster, navigate to your cluster, and choose the Maintenance tab, Usage limits section. To view the amount of usage available and consumed for your cluster, navigate to your cluster. Choose the Cluster performance tab, then view the graphs for the usage consumed for a feature.

You can use the following Amazon Redshift CLI operations to manage usage limits. For more information, see the AWS CLI Command Reference.

- create-usage-limit
- describe-usage-limits
- modify-usage-limit
- delete-usage-limit

You can use the following Amazon Redshift API operations to manage usage limits. For more information, see the Amazon Redshift API Reference.

- CreateUsageLimit
- DescribeUsageLimits
- ModifyUsageLimit
- DeleteUsageLimit

Watch the following video to learn how to create and monitor usage limits using the Amazon Redshift console: Cost Controls for Amazon Redshift Spectrum and Concurrency Scaling.

### Managing cluster relocation in Amazon Redshift

By using relocation in Amazon Redshift, you enable Amazon Redshift to move a cluster to another Availability Zone (AZ) without any loss of data or changes to your applications. With relocation, you can continue operations when there is an interruption of service on your cluster with minimal impact.

When cluster relocation is enabled, Amazon Redshift might choose to relocate clusters in some situations. In particular, this happens where issues in the current Availability Zone prevent optimal cluster operation or to improve service availability. You can also invoke the relocation function in cases where resource constraints in a given Availability Zone are disrupting cluster operations. An example is the ability to resume or resize a cluster. Amazon Redshift offers the relocation feature at no extra charge.
When an Amazon Redshift cluster is relocated to a new Availability Zone, the new cluster has the same endpoint as the original cluster. Your applications can reconnect to the endpoint and continue operations without modifications or loss of data. However, relocation might not always be possible due to potential resource constraints in a given Availability Zone.

Amazon Redshift cluster relocation is supported for the RA3 instance types only, such as ra3.16xlarge, ra3.4xlarge, and ra3.xlplus. RA3 instance types use Redshift Managed Storage (RMS) as a durable storage layer. The latest copy of a cluster's data is always available in other Availability Zones in an AWS Region. In other words, you can relocate an Amazon Redshift cluster to another Availability Zone without any loss of data.

When you enable your cluster for relocation, Amazon Redshift migrates your cluster to be behind a proxy. Doing this helps implement location-independent access to a cluster's compute resources. The migration causes the cluster to be rebooted. When a cluster is relocated to another Availability Zone, an outage occurs while the new cluster is brought back online in the new Availability Zone. However, you don't have to make any changes to your applications because the cluster endpoint remains unchanged even after the cluster is relocated to the new Availability Zone.

If you enable relocation and you currently use the leader node IP address to access your cluster, make sure to change that access. Instead, use the IP address associated with the cluster's virtual private cloud (VPC) endpoint. To find this cluster IP address, find and use the VPC endpoint in the Network and security section of the cluster details page. To get more details on the VPC endpoint, sign in to the Amazon VPC console.

You can also use the AWS Command Line Interface (AWS CLI) command `describe-vpc-endpoints` to get the elastic network interface associated with the endpoint. You can use the `describe-network-interfaces` command to get the associated IP address. For more information on Amazon Redshift AWS CLI commands, see Available commands in the AWS CLI Command Reference.

### Enabling cluster relocation

You can enable and manage cluster relocation from the Amazon Redshift console, AWS CLI, and Amazon Redshift API.

To enable cluster relocation, define a subnet group that includes multiple Availability Zones. If Amazon Redshift identifies more than one accessible Availability Zone, Amazon Redshift automatically chooses from the list of accessible Availability Zones to relocate the cluster.

After relocation is complete, you use the same endpoint to access the cluster. Amazon Redshift deletes the original cluster's compute resources and returns them to the resource pool.

### Limitations

When using Amazon Redshift relocation, be aware of the following limitations:

- Cluster relocation might not be possible in all scenarios due to potential resource limitations in a given Availability Zone. If this happens, Amazon Redshift doesn't change the original cluster.
- Relocation isn't supported on DC1, DC2, or the DS2 instance families of products.
- Relocation isn't available for a publicly accessible Amazon Redshift cluster.
- You can't perform a relocation across AWS Regions.
- You can enable relocation only for a cluster that uses the default port setting (5439). Otherwise, enabling relocation fails.
- If you have enabled relocation successfully and later attempt to modify the default port setting, the modify operation fails.
Managing relocation using the console

You can manage the settings for cluster relocation using the Amazon Redshift console.

Enabling relocation when creating a new cluster

Use the following procedure to enable relocation when creating a new cluster.

To enable relocation for a new cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose Create cluster to create a new cluster. For more information on how to create a cluster, see Create a sample Amazon Redshift cluster in Amazon Redshift Getting Started Guide.
4. Under Backup, for Cluster relocation, choose Enable. Relocation is disabled by default.
5. Under Network and security, for Publicly accessible, accept the default Disable. If you choose Enable, Amazon Redshift returns an error.
6. Choose Create cluster.

Modifying relocation for an existing cluster

Use the following procedure to change the relocation setting for an existing cluster.

To modify the relocation setting for an existing cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.
3. Choose the name of the cluster that you want to modify from the list. The cluster details page appears.
4. Choose the Maintenance tab, then in the Backup details section choose Edit.
5. Under Backup, choose Enable. Relocation is disabled by default.
6. Choose the Properties tab, then in the Network and security section make sure to choose Disable for the Publicly accessible option.
7. In the Network and security section, make sure to choose Disable for the Publicly accessible option.
8. Choose Modify cluster.

Relocating a cluster

Use the following procedure to manually relocate a cluster to another Availability Zone. This is especially useful when you want to test your network setup in secondary Availability Zones or when you are running into resource constraints in the current Availability Zone.

To relocate a cluster to another Availability Zone

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.

3. Choose the name of the cluster that you want to move from the list. The cluster details page appears.

4. For Actions, choose Relocate. The Relocate cluster page appears.

5. (Optional) Choose an Availability Zone. If you don't choose an Availability Zone, Amazon Redshift chooses one for you.

Amazon Redshift starts the relocation and displays the cluster as relocating. After the relocation completes, the cluster status changes to available.

Managing relocation using the Amazon Redshift CLI

You can manage the settings for cluster relocation using the AWS Command Line Interface (CLI).

With the AWS CLI, the following example command creates an Amazon Redshift cluster named mycluster that has relocation enabled.

```
aws redshift create-cluster --cluster-identifier mycluster --number-of-nodes 2 --master-username adminuser --master-user-password TopSecret1 --node-type ra3.4xlarge --port 5439 --no-publicly-accessible --availability-zone-relocation
```

If your current cluster is using a different port, you have to modify it to use 5439 before modifying it to enable relocation. The following example command modify the port in case your cluster doesn't use 5439.

```
aws redshift modify-cluster --cluster-identifier mycluster --port 5439
```

The following example command enables the availability-zone-relocation parameter on the Amazon Redshift cluster.

```
aws redshift modify-cluster --cluster-identifier mycluster --availability-zone-relocation
```

The following example command disables the availability-zone-relocation parameter on the Amazon Redshift cluster.

```
aws redshift modify-cluster --cluster-identifier mycluster --no-availability-zone-relocation
```

The following example command invokes relocation on the Amazon Redshift cluster.

```
aws redshift modify-cluster --cluster-identifier mycluster --availability-zone us-east-1b
```

Working with Redshift-managed VPC endpoints in Amazon Redshift

By default, an Amazon Redshift cluster is provisioned in a virtual private cloud (VPC). It can be accessed from another VPC or subnet when you either allow public access or set up an internet gateway, a NAT
device, or an AWS Direct Connect connection to route traffic to the cluster. Or you can access a cluster by setting up a Redshift-managed VPC endpoint (powered by AWS PrivateLink).

You set up a Redshift-managed VPC endpoint as a private connection between a VPC that contains a cluster and a VPC that is running a client tool. If the cluster is in another account, then the account owner (grantor) needs to grant access to the account (grantee) that wants to establish a connection. With this approach, you can access the data warehouse without using public IP addresses or routing traffic across the internet.

The following scenarios describe common reasons to allow access to a cluster using a Redshift-managed VPC endpoint:

- AWS account A wants to allow a VPC in AWS account B to have access to a cluster.
- AWS account A wants to allow a VPC that is also in AWS account A to have access to a cluster.
- AWS account A wants to allow a different subnet in the cluster's VPC within AWS account A to have access to a cluster.

The general workflow to set up a Redshift-managed VPC endpoint to access a cluster in another account is as follows:

1. The owner account of the cluster grants access authorization to another account and specifies the AWS account ID and VPC identifier (or all VPCs) of the grantee.
2. The grantee account is notified that they have permission to create a Redshift-managed VPC endpoint.
3. The grantee account creates a Redshift-managed VPC endpoint.
4. The grantee account can now access the cluster of the owner account using the Redshift-managed VPC endpoint.

You can manage this process with the Amazon Redshift console, the AWS CLI, or the Amazon Redshift API.

**Considerations when using Redshift-managed VPC endpoints**

When using Redshift-managed VPC endpoints, keep the following in mind:

- Make sure that the cluster to access is an RA3 node type.
- Make sure that the cluster to access has cluster relocation turned on. For information about requirements to turn on cluster relocation, see Managing cluster relocation in Amazon Redshift (p. 64).
- Make sure that the cluster to access is available through port 5439.
- You can modify the VPC security groups associated with an existing Redshift-managed VPC endpoint. To modify other settings, delete the current Redshift-managed VPC endpoint and create a new one.
- The number of Redshift-managed VPC endpoints that you can create is limited to your VPC endpoint quota.
- The Redshift-managed VPC endpoints aren't accessible from the internet. A Redshift-managed VPC endpoint is accessible only within the VPC where the endpoint is provisioned or any VPCs peered with the VPC where the endpoint is provisioned as permitted by the route tables and security groups.
- You can't use the Amazon VPC console to manage Redshift-managed VPC endpoints.

For information about quotas and naming constraints, see Quotas and limits in Amazon Redshift (p. 505).
Managing Redshift-managed VPC endpoints using the Amazon Redshift console

You can configure the use of Redshift-managed VPC endpoints by using the Amazon Redshift console.

Granting access to a cluster

If the VPC that you want to access your cluster is in another AWS account, make sure to authorize it from the owner's (grantor's) account.

To allow a VPC in another AWS account to have access to your cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. For the cluster that you want to allow access, view the cluster details by choosing the cluster name. Choose the Properties tab of the cluster.
   
   The Granted accounts section displays the accounts and corresponding VPCs that have access to your cluster.
4. Choose Grant access to display a form to enter Grantee information to add an account.
5. For AWS account ID, enter the ID of the account you are granting access. You can grant access to specific VPCs or all VPCs in the specified account.
6. Choose Grant access to grant access.

Creating a Redshift-managed VPC endpoint

If you own a cluster or you have been granted access to it, you can create a Redshift-managed VPC endpoint for the cluster.

To create a Redshift-managed VPC endpoint

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations.
   
   The Configurations page displays the Redshift-managed VPC endpoints that have been created. To view details for an endpoint, choose its name.
3. Choose Create endpoint to display a form to enter information about the endpoint to add.
4. Enter values for Endpoint name, AWS account ID, Cluster identifier, Virtual private cloud (VPC), Subnet group, and other properties of the endpoint.
   
   The subnet group in Subnet group defines the subnets and IP addresses where Amazon Redshift deploys the endpoint. Amazon Redshift chooses a subnet that has IP addresses available for the network interface associated with the endpoint.
   
   The optional security group in Security group defines the ports, protocols, and sources for inbound traffic that you are authorizing for your endpoint. Commonly, you allow access to port 5439 to the security group or the CIDR range where your workloads run.
5. Choose Create endpoint to create the endpoint.
After your endpoint is created, you can access the cluster through the URL shown in Endpoint URL in the configuration settings for your Redshift-managed VPC endpoint.

Managing Redshift-managed VPC endpoints using the AWS CLI

You can use the following Amazon Redshift CLI operations to work with Redshift-managed VPC endpoints. For more information, see the AWS CLI Command Reference.

- authorize-endpoint-access
- revoke-endpoint-access
- create-endpoint-access
- modify-endpoint-access
- delete-endpoint-access
- describe-endpoint-access
- describe-endpoint-authorization

Managing Redshift-managed VPC endpoints using Amazon Redshift API operations

You can use the following Amazon Redshift API operations to work with Redshift-managed VPC endpoints. For more information, see the Amazon Redshift API Reference.

- AuthorizeEndpointAccess
- RevokeEndpointAccess
- CreateEndpointAccess
- ModifyEndpointAccess
- DeleteEndpointAccess
- DescribeEndpointAccess
- DescribeEndpointAuthorization

Managing clusters using the console

To create, modify, resize, delete, reboot, and back up clusters, use the Clusters section in the Amazon Redshift console.

To view clusters

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list. If you don’t have any clusters, choose Create cluster to create one.
3. Choose the cluster name in the list to view more details about a cluster.

Topics

- Creating a cluster (p. 71)
Creating a cluster

Before you create a cluster, read Overview of Amazon Redshift clusters (p. 34) and Clusters and nodes in Amazon Redshift (p. 34).

To create a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.
3. Choose Create cluster to create a cluster.
4. Follow the instructions on the console page to enter the properties for Cluster configuration.

Choose one of the following methods to size your cluster:

- If your AWS Region supports RA3 node types, choose either Production or Free trial to answer the question What are you planning to use this cluster for?

  If your organization is eligible, you might be able to create a cluster under the Amazon Redshift free trial program. To do this, choose Free trial to create a configuration with the dc2.large node type. For more information about choosing a free trial, see Amazon Redshift free trial.

- If you don't know how large to size your cluster, choose Help me choose. Doing this starts a sizing calculator that asks you questions about the size and query characteristics of the data that you plan to store in your data warehouse.

  If you know the required size of your cluster (that is, the node type and number of nodes), choose I'll choose. Then choose the Node type and number of Nodes to size your cluster for the proof of concept.

5. Follow the instructions on the console page to enter the properties for Cluster details.

  If you are behind a firewall, the database port must be an open port that accepts inbound connections.

6. (Optional) Follow the instructions on the console page to enter properties for Cluster permissions. Provide cluster permissions if your cluster needs to access other AWS services for you, for example to load data from Amazon S3.
7. Choose **Create cluster** to create the cluster. The cluster might take several minutes to be ready to use.

### Additional configurations

When you create a cluster, you can specify additional properties to customize it. You can find more details about some of these properties in the following list.

**Virtual private cloud (VPC)**

Choose a VPC that has a subnet group. After the cluster is created, the subnet group can't be changed.

**Parameter groups**

Choose a cluster parameter group to associate with the cluster. If you don't choose one, the cluster uses the default parameter group.

**Encryption**

Choose whether you want to encrypt all data within the cluster and its snapshots. If you leave the default setting, **None**, encryption is not enabled. If you want to enable encryption, choose whether you want to use AWS Key Management Service (AWS KMS) or a hardware security module (HSM), and then configure the related settings. For more information about encryption in Amazon Redshift, see Amazon Redshift database encryption (p. 323).

- **KMS**

  Choose **Use AWS Key Management Service (AWS KMS)** if you want to enable encryption and use AWS KMS to manage your encryption key. Also, choose the key to use. You can choose a default key, a key from the current account, or a key from a different account.

  **Note**

  If you want to use a key from another AWS account, then enter the Amazon Resource Name (ARN) for the key to use. You must have permission to use the key. For more information about access to keys in AWS KMS, see Controlling access to your keys in the AWS Key Management Service Developer Guide.

  For more information about using AWS KMS encryption keys in Amazon Redshift, see Database encryption for Amazon Redshift using AWS KMS (p. 324).

- **HSM**

  Choose **HSM** if you want to enable encryption and use a hardware security module (HSM) to manage your encryption key.

  If you choose **HSM**, choose values from **HSM Connection** and **HSM Client Certificate**. These values are required for Amazon Redshift and the HSM to form a trusted connection over which the cluster key can be passed. The HSM connection and client certificate must be set up in Amazon Redshift before you launch a cluster. For more information about setting up HSM connections and client certificates, see Encryption for Amazon Redshift using hardware security modules (p. 326).

**Maintenance track**

You can choose whether the cluster version used is the **Current**, **Trailing**, or sometimes **Preview** track.

**Monitoring**

You can choose whether to create CloudWatch alarms.

**Configure cross-region snapshot**

You can choose whether to enable cross-Region snapshots.
Modifying a cluster

When you modify a cluster, changes to the following options are applied immediately:

- VPC security groups
- Publicly accessible
- Admin user password
- HSM Connection
- HSM Client Certificate
- Maintenance detail
- Snapshot preferences

Changes to the following options take effect only after the cluster is restarted:

- Cluster identifier
  Amazon Redshift restarts the cluster automatically when you change Cluster identifier.
- Enhanced VPC routing
  Amazon Redshift restarts the cluster automatically when you change Enhanced VPC routing.
- Cluster parameter group

If you decrease the automated snapshot retention period, existing automated snapshots whose settings fall outside of the new retention period are deleted. For more information, see Amazon Redshift snapshots (p. 292).

To modify a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster to modify.
5. Update the cluster properties. Some of the properties you can modify are:
   - Cluster identifier
   - Snapshot retention
   - Cluster relocation

To edit settings for Network and security, Maintenance, and Database configurations, the console provides links to the appropriate cluster details tab.
6. Choose Save changes.

Deleting a cluster

If you no longer need your cluster, you can delete it. If you plan to provision a new cluster with the same data and configuration as the one you are deleting, you need a manual snapshot. By using a manual snapshot, you can restore the snapshot later and resume using the cluster. If you delete your cluster but you don’t create a final manual snapshot, the cluster data is deleted. In either case, automated snapshots are deleted after the cluster is deleted, but any manual snapshots are retained until you delete them.
You might be charged Amazon Simple Storage Service storage rates for manual snapshots, depending on the amount of storage you have available for Amazon Redshift snapshots for your clusters. For more information, see Shutting down and deleting clusters (p. 59).

To delete a cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster to delete.
4. For Actions, choose Delete. The Delete cluster page appears.
5. Choose Delete cluster.

Rebooting a cluster

When you reboot a cluster, the cluster status is set to rebooting and a cluster event is created when the reboot is completed. Any pending cluster modifications are applied at this reboot.

To reboot a cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster to reboot.
4. For Actions, choose Reboot cluster. The Reboot cluster page appears.
5. Choose Reboot cluster.

Resizing a cluster

When you resize a cluster, you specify a number of nodes or node type that is different from the current configuration of the cluster. While the cluster is in the process of resizing, you cannot run any write or read/write queries on the cluster; you can run only read queries.

For more information about resizing clusters, including walking through the process of resizing clusters using different approaches, see Resizing clusters in Amazon Redshift (p. 51).

To resize a cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster to resize.
4. For Actions, choose Resize. The Resize cluster page appears.
5. Follow the instructions on the page. You can resize the cluster now, once at a specific time, or increase and decrease the size of your cluster on a schedule.
6. Depending on your choices, choose Resize now or Schedule resize.

If you have reserved nodes, for example DS2 reserved nodes, you can upgrade to RA3 reserved nodes. You can do this when you use the console to restore from a snapshot or to perform an elastic resize. You can use the console to guide you through this process. For more information about upgrading to RA3 nodes, see Upgrading to RA3 node types.
Upgrading the release version of a cluster

You can upgrade the release maintenance version of a cluster that has a Release Status value of New release available. When you upgrade the maintenance version, you can choose to upgrade immediately or upgrade in the next maintenance window.

**Important**
If you upgrade immediately, your cluster is offline until the upgrade completes.

**To upgrade a cluster to a new release version**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster to upgrade.
4. For Actions, choose Upgrade cluster version. The Upgrade cluster version page appears.
5. Follow the instructions on the page.

Getting information about cluster configuration

**To display information about a cluster**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include Cluster performance, Query monitoring, Databases, Datashares, Schedules, Maintenance, and Properties tabs.
3. Choose each tab to view more details.

Getting an overview of cluster status

**To view the status of a cluster**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. View the status of the cluster in the Status column.

Creating a snapshot of a cluster

**To create a snapshot of a cluster**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters.
3. Choose the cluster for which to create a snapshot.
4. For Actions, choose Create snapshot. The Create snapshot page appears.
5. Follow the instructions on the page.
6. Choose **Create snapshot**.

**Creating or editing a disk space alarm**

**To create a disk space usage alarm for a cluster**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Alarms**.
3. For **Actions**, choose **Create alarm**. The **Create alarm** page appears.
4. Follow the instructions on the page.
5. Choose **Create alarm**.

**Working with cluster performance data**

In the console, you can work with cluster performance on the **Cluster performance** tab of the cluster details page.

**Managing clusters using the AWS CLI and Amazon Redshift API**

You can use the following AWS CLI operations to manage clusters in Amazon Redshift.

- cancel-resize
- create-cluster
- delete-cluster
- describe-clusters
- describe-cluster-versions
- describe-node-configuration-options
- describe-orderable-cluster-options
- describe-resize
- modify-cluster
- pause-cluster
- reboot-cluster
- resize-cluster
- resume-cluster

You can use the following Amazon Redshift API operations to manage clusters.

- CancelResize
- CreateCluster
- DeleteCluster
- DescribeClusters
- DescribeClusterVersions
Managing clusters using the AWS SDK for Java

The following Java code example demonstrates common cluster management operations including:

- Creating a cluster.
- Listing metadata about a cluster.
- Modifying configuration options.

After you initiate the request for the cluster to be created, you must wait until the cluster is in the `available` state before you can modify it. This example uses a loop to periodically check the status of the cluster using the `describeClusters` method. When the cluster is available, the preferred maintenance window for the cluster is changed.

For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). You need to update the code and specify a cluster identifier.

Example

```java
/**
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * This file is licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License. A copy of
 * the License is located at
 * http://aws.amazon.com/apache2.0/
 *
 * This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
 * CONDITIONS OF ANY KIND, either express or implied. See the License for the
 * specific language governing permissions and limitations under the License.
 */

// snippet-sourcedescription:[CreateAndModifyCluster demonstrates how to create and modify
// an Amazon Redshift cluster.]
// snippet-service:[redshift]
// snippet-keyword:[Java]
// snippet-keyword:[Amazon Redshift]
// snippet-keyword:[Code Sample]
// snippet-keyword:[CreateCluster]
// snippet-keyword:[DescribeClusters]
// snippet-keyword:[ModifyCluster]
// snippet-sourcetype:[full-example]
// snippet-sourcedate:[2019-02-01]
// snippet-sourceauthor:[AWS]
// snippet-start:[redshift.java.CreateAndModifyCluster.complete]
```
package com.amazonaws.services.redshift;

import java.io.IOException;
import com.amazonaws.services.redshift.AmazonRedshift;
import com.amazonaws.services.redshift.AmazonRedshiftClientBuilder;
import com.amazonaws.services.redshift.model.*;

public class CreateAndModifyCluster {
    public static AmazonRedshift client = AmazonRedshiftClientBuilder.defaultClient();
    public static String clusterIdentifier = "provide a cluster identifier";
    public static long sleepTime = 20;

    public static void main(String[] args) throws IOException {
        try {
            createCluster();
            waitForClusterReady();
            describeClusters();
            modifyCluster();
            describeClusters();
        } catch (Exception e) {
            System.err.println("Operation failed: " + e.getMessage());
        }
    }

    private static void createCluster() {
        CreateClusterRequest request = new CreateClusterRequest()
                .withClusterIdentifier(clusterIdentifier)
                .withMasterUsername("masteruser")
                .withMasterUserPassword("12345678Aa")
                .withNodeType("ds2.xlarge")
                .withNumberOfNodes(2)
                .withClusterSubnetGroupName("subnetgroup1");

        Cluster createResponse = client.createCluster(request);
        System.out.println("Created cluster "+ createResponse.getClusterIdentifier());
    }

    private static void describeClusters() {
        DescribeClustersRequest request = new DescribeClustersRequest()
                .withClusterIdentifier(clusterIdentifier);

        DescribeClustersResult result = client.describeClusters(request);
        printResult(result);
    }

    private static void modifyCluster() {
        ModifyClusterRequest request = new ModifyClusterRequest()
                .withClusterIdentifier(clusterIdentifier)
                .withPreferredMaintenanceWindow("wed:07:30-wed:08:00");

        client.modifyCluster(request);
        System.out.println("Modified cluster "+ clusterIdentifier);
    }

    private static void printResult(DescribeClustersResult result) {
    }
}
Managing clusters in a VPC

Topics

- Overview (p. 79)
- Creating a cluster in a VPC (p. 81)
- Managing VPC security groups for a cluster (p. 81)
- Amazon Redshift cluster subnet groups (p. 82)

Overview

Amazon Redshift supports both the EC2-VPC and EC2-Classic platforms to launch a cluster in a virtual private cloud (VPC) based on the Amazon VPC service. For more information, see Use EC2-VPC when you create your cluster (p. 38).

Note
Amazon Redshift supports launching clusters into dedicated tenancy VPCs. For more information, see Dedicated instances in the Amazon VPC User Guide.

When provisioning a cluster in VPC, you need to do the following:

- Provide VPC information.

When you request Amazon Redshift to create a cluster in your VPC, you must provide your VPC information by creating a cluster subnet group. This information includes the VPC ID and a list of...
subnets in your VPC. When you launch a cluster, you provide the cluster subnet group so that Amazon Redshift can provision your cluster in one of the subnets in the VPC. For more information about creating subnet groups in Amazon Redshift, see Amazon Redshift cluster subnet groups (p. 82). For more information about setting up VPC, see Getting started with Amazon VPC in the Amazon VPC Getting Started Guide.

- **Optionally, configure the publicly accessible options.**

If you configure your cluster to be publicly accessible, you can optionally select an elastic IP address to use for the external IP address. An elastic IP address is a static IP address that is associated with your AWS account. You can use an elastic IP address to connect to your cluster from outside the VPC. An elastic IP address gives you the ability to change your underlying configuration without affecting the IP address that clients use to connect to your cluster. This approach can be helpful for situations such as recovery after a failure.

If you want to use an elastic IP address associated with your own AWS account, you must create it in Amazon EC2 prior to launching your Amazon Redshift cluster. Otherwise, it will not be available during the launch process. You can also have Amazon Redshift configure an elastic IP address to use for the VPC. However, the assigned elastic IP address is managed by the Amazon Redshift service and isn't associated with your AWS account. For more information, see Elastic IP addresses in the Amazon EC2 User Guide for Linux Instances.

In some cases, you might have a publicly accessible cluster in a VPC that you want to connect to it by using the private IP address from within the VPC. If so, set the following VPC parameters to true:

- DNS resolution
- DNS hostnames

Suppose that you have a publicly accessible cluster in a VPC but don't set those parameters to true in the VPC. In these cases, connections made from within the VPC resolve to the elastic IP address of the cluster instead of the private IP address. We recommend that you set these parameters to true and use the private IP address for a publicly accessible cluster when connecting from within the VPC. For more information, see Using DNS with your VPC in the Amazon VPC User Guide.

**Note**

If you have an existing publicly accessible cluster in a VPC, connections from within the VPC continue to use the elastic IP address to connect to the cluster until you resize the cluster. This occurs even with the preceding parameters set. Any new clusters follow the new behavior of using the private IP address when connecting to the publicly accessible cluster from within the same VPC.

The elastic IP address is an external IP address for accessing the cluster outside of a VPC. It's not related to the cluster node public IP addresses and private IP addresses that are displayed in the Amazon Redshift console under Connection details. The public and private cluster node IP addresses appear regardless of whether the cluster is publicly accessible or not. They are used only in certain circumstances to configure ingress rules on the remote host. These circumstances occur when you load data from an Amazon EC2 instance or other remote host using a Secure Shell (SSH) connection. For more information, see Step 1: Retrieve the cluster public key and cluster node IP addresses in the Amazon Redshift Database Developer Guide.

The option to associate a cluster with an elastic IP address is available when you create the cluster or restore the cluster from a snapshot. In some cases, you might want to associate the cluster with an elastic IP address or change an elastic IP address that is associated with the cluster. To attach an elastic IP address after the cluster is created, first update the cluster so that it is not publicly accessible, then make it both publicly accessible and add an Elastic IP address in the same operation.

- **Associate a VPC security group.**

You then grant inbound access using a VPC security group. This VPC security group must allow access over the database port for the cluster so that you can connect by using SQL client tools. You can configure this in advance, or add rules to it after you launch the cluster. For more information, see
Security in your VPC in the Amazon VPC User Guide. You cannot use the Amazon Redshift cluster security groups to grant inbound access to the cluster.

For more information about working with clusters in a VPC, see Creating a cluster in a VPC (p. 81).

### Restoring a snapshot of a cluster in VPC

A snapshot of a cluster in VPC can only be restored in a VPC, not outside the VPC. You can restore it in the same VPC or another VPC in your account. For more information about snapshots, see Amazon Redshift snapshots (p. 292).

### Creating a cluster in a VPC

The following are the general steps how you can deploy a cluster in your virtual private cloud (VPC).

#### To create a cluster in a VPC

1. **Set up a VPC.**
   
   You can create your cluster either in the default VPC for your account, if your account has one, or a VPC that you have created. For more information, see Use EC2-VP when you create your cluster (p. 38). To create a VPC, see Getting started with Amazon VPC in the Amazon VPC User Guide. Make a note of the VPC identifier, subnet, and subnet's Availability Zone. You need this information when you launch your cluster.

   **Note**

   You must have at least one subnet defined in your VPC so you can add it to the cluster subnet group in the next step. If you use the VPC wizard, a subnet for your VPC is automatically created for you. For more information about adding a subnet to your VPC, see Adding a subnet to your VPC in the Amazon VPC User Guide.

2. **Create an Amazon Redshift cluster subnet group to specify which subnet your Amazon Redshift cluster can use in the VPC.**
   
   You can create a cluster subnet group using either the Amazon Redshift console or programmatically. For more information, see Amazon Redshift cluster subnet groups (p. 82).

3. **Authorize access for inbound connections in a VPC security group that you associate with the cluster.**
   
   You can enable a client outside the VPC (on the public internet) to connect to the cluster. To do this, you associate the cluster with a VPC security group that grants inbound access to the port that you used when you launched the cluster. For examples of security group rules, see Security group rules in the Amazon VPC User Guide.

4. **Follow the steps in Getting started with Amazon Redshift in the Amazon Redshift Getting Started Guide to create a cluster.** Make the following modifications when creating your cluster:
   
   - To display the Additional configurations section, switch off Use defaults.
   - In the Network and security section, specify the Virtual private cloud (VPC), Cluster subnet group, and VPC security group that you set up.

Now you are ready to use the cluster. You can follow the Getting Started steps to test the cluster by uploading sample data and trying example queries.

### Managing VPC security groups for a cluster

When you provision an Amazon Redshift cluster, it is locked down by default so nobody has access to it. To grant other users inbound access to an Amazon Redshift cluster, you associate the cluster with a
Cluster subnet groups

Security group. If you are on the EC2-VPC platform, you can either use an existing Amazon VPC security group or define a new one. You then associate it with a cluster as described following. If you are on the EC2-Classic platform, you define a cluster security group and associate it with a cluster. For more information on using cluster security groups on the EC2-Classic platform, see Amazon Redshift cluster security groups (p. 438).

A VPC security group consists of a set of rules that control access to an instance on the VPC, such as your cluster. Individual rules set access based either on ranges of IP addresses or on other VPC security groups. When you associate a VPC security group with a cluster, the rules that are defined in the VPC security group control access to the cluster.

Each cluster you provision on the EC2-VPC platform has one or more Amazon VPC security groups associated with it. Amazon VPC provides a VPC security group called default, which is created automatically when you create the VPC. Each cluster that you launch in the VPC is automatically associated with the default VPC security group if you don’t specify a different VPC security group when you create the cluster. You can associate a VPC security group with a cluster when you create the cluster, or you can associate a VPC security group later by modifying the cluster.

The following table describes the default rules for the default VPC security group.

<table>
<thead>
<tr>
<th>Source</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>The security group ID (sg-xxxxxxxx)</td>
<td>All</td>
<td>All</td>
<td>Allow inbound traffic from instances assigned to the same security group</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>All</td>
<td>All</td>
<td>Allow all outbound traffic</td>
</tr>
</tbody>
</table>

You can change the rules for the default VPC security group as needed for your Amazon Redshift cluster.

If the default VPC security group is enough for you, you don’t need to create more. However, you can optionally create additional VPC security groups to better manage inbound access to your cluster. For example, suppose that you are running a service on an Amazon Redshift cluster, and you have several different service levels you provide to your customers. If you don’t want to provide the same access at all service levels, you might want to create separate VPC security groups, one for each service level. You can then associate these VPC security groups with your cluster.

You can create up to 100 VPC security groups for a VPC and associate a VPC security group with many clusters. However, you can only associate up to five VPC security groups with a given cluster.

Amazon Redshift applies changes to a VPC security group immediately. So if you have associated the VPC security group with a cluster, inbound cluster access rules in the updated VPC security group apply immediately.

You can create and modify VPC security groups in the https://console.aws.amazon.com/vpc/. You can also manage VPC security groups programmatically by using the AWS CLI, the Amazon EC2 CLI, and the AWS Tools for Windows PowerShell. For more information about working with VPC security groups, see Security groups for your VPC in the Amazon VPC User Guide.

Amazon Redshift cluster subnet groups

Overview

You create a cluster subnet group if you are provisioning your cluster in your virtual private cloud (VPC). For more information about VPC, see the Amazon VPC product detail page.

Your VPC can have one or more subnets, a subset of IP addresses within your VPC, that enable you to group your resources based on your security and operation needs. A cluster subnet group allows you
to specify a set of subnets in your VPC. When provisioning a cluster you provide the subnet group and
Amazon Redshift creates the cluster on one of the subnets in the group.

For more information about creating a VPC, go to Amazon VPC User Guide Documentation.

After creating a subnet group, you can remove subnets you previously added or add more subnets.
Amazon Redshift provides API operations for you to create, modify or delete a cluster subnet group. You
can also perform these operations in the console.

Managing cluster subnet groups using the console

You can manage your cluster subnet groups using the Amazon Redshift console. You can create a cluster
subnet group, manage an existing one, or delete one. All of these tasks start from the cluster subnet
group list. You must select a cluster subnet group to manage it.

You can provision a cluster on one of the subnets that you provide the subnet group. A cluster subnet
group enables you to specify a set of subnets in your virtual private cloud (VPC).

Creating a cluster subnet group

You must have at least one cluster subnet group defined to provision a cluster in a VPC.

To create a cluster subnet group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://
   console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Subnet groups. The list of subnet
groups is displayed.
3. Choose Create cluster subnet group to display the create page.
4. Enter information for the subnet group, including which subnets to add.
5. Choose Create cluster subnet group to create the group with the subnets that you chose.

Modifying a cluster subnet group

To modify a cluster subnet group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://
   console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Subnet groups. The list of subnet
groups is displayed.
3. Choose the subnet group to modify.
4. For Actions, choose Modify to display the details of the subnet group.
5. Update information for the subnet group.
6. Choose Save to modify the group.

Deleting a cluster subnet group

You can't delete a cluster subnet group that is used by a cluster.

To delete a cluster subnet group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://
   console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Configurations**, then choose **Subnet groups**. The list of subnet groups is displayed.
3. Choose the subnet group to delete, then choose **Delete**.

### Managing cluster subnet groups using the AWS SDK for Java

The following Java code example demonstrates common cluster subnet operations including:

- Creating a cluster subnet group.
- Listing metadata about a cluster subnet group.
- Modifying a cluster subnet group.

For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). You need to update the code and provide a cluster subnet group name and two subnet identifiers.

#### Example

```java
/**
   * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
   * This file is licensed under the Apache License, Version 2.0 (the "License").
   * You may not use this file except in compliance with the License. A copy of
   * the License is located at
   * http://aws.amazon.com/apache2.0/
   * This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
   * CONDITIONS OF ANY KIND, either express or implied. See the License for the
   * specific language governing permissions and limitations under the License.
   */

// CreateAndModifyClusterSubnetGroup demonstrates how to create and modify an Amazon Redshift subnet group.
// snippet-service:[redshift]
// snippet-keyword:[Java]
// snippet-keyword:[Amazon Redshift]
// snippet-keyword:[Code Sample]
// snippet-keyword:[CreateClusterSubnetGroup]
// snippet-keyword:[DescribeClusterSubnetGroups]
// snippet-keyword:[full-example]
// snippet-sourcedate:[2019-02-01]
// snippet-sourceauthor:[AWS]

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import com.amazonaws.services.redshift.model.

public class CreateAndModifyClusterSubnetGroup {

    public static AmazonRedshift client;
    public static String clusterSubnetGroupName = "subnet-group-name";
    // You can use the VPC console to find subnet IDs to use.
    public static String subnetId1 = "***provide a subnet ID****";
    public static String subnetId2 = "***provide a subnet ID****";
```
public static void main(String[] args) throws IOException {

    // Default client using the com.amazonaws.auth.DefaultAWSCredentialsProviderChain
    client = AmazonRedshiftClientBuilder.defaultClient();

    try {
        createClusterSubnetGroup();
        describeClusterSubnetGroups();
        modifyClusterSubnetGroup();
    } catch (Exception e) {
        System.err.println("Operation failed: " + e.getMessage());
    }
}

private static void createClusterSubnetGroup() {
    CreateClusterSubnetGroupRequest request = new CreateClusterSubnetGroupRequest()
        .withClusterSubnetGroupName(clusterSubnetGroupName)
        .withDescription("my cluster subnet group")
        .withSubnetIds(subnetId1);
    client.createClusterSubnetGroup(request);
    System.out.println("Created cluster subnet group: " + clusterSubnetGroupName);
}

private static void modifyClusterSubnetGroup() {
    // Get existing subnet list.
    DescribeClusterSubnetGroupsRequest request1 = new DescribeClusterSubnetGroupsRequest()
        .withClusterSubnetGroupName(clusterSubnetGroupName);
    DescribeClusterSubnetGroupsResult result1 = client.describeClusterSubnetGroups(request1);
    List<String> subnetNames = new ArrayList<String>();
    // We can work with just the first group returned since we requested info about one group.
    for (Subnet subnet : result1.getClusterSubnetGroups().get(0).getSubnets()) {
        subnetNames.add(subnet.getSubnetIdentifier());
    }
    // Add to existing subnet list.
    subnetNames.add(subnetId2);
    ModifyClusterSubnetGroupRequest request = new ModifyClusterSubnetGroupRequest()
        .withClusterSubnetGroupName(clusterSubnetGroupName)
        .withSubnetIds(subnetNames);
    ClusterSubnetGroup result2 = client.modifyClusterSubnetGroup(request);
    System.out.println("Subnet group modified.");
    printResultSubnetGroup(result2);
}

private static void describeClusterSubnetGroups() {
    DescribeClusterSubnetGroupsRequest request = new DescribeClusterSubnetGroupsRequest()
        .withClusterSubnetGroupName(clusterSubnetGroupName);
    DescribeClusterSubnetGroupsResult result = client.describeClusterSubnetGroups(request);
    printResultSubnetGroups(result);
}

private static void printResultSubnetGroups(DescribeClusterSubnetGroupsResult result) {
    if (result == null) {
        System.out.println("Describe cluster subnet groups result is null.");
        return;
    }
}
for (ClusterSubnetGroup group : result.getClusterSubnetGroups())
{
    printResultSubnetGroup(group);
}

private static void printResultSubnetGroup(ClusterSubnetGroup group) {
    System.out.format("Name: %s, Description: %s\n", group.getClusterSubnetGroupName(),
            group.getDescription());
    for (Subnet subnet : group.getSubnets()) {
        System.out.format("  Subnet: %s, %s, %s\n", subnet.getSubnetIdentifier(),
            subnet.getSubnetAvailabilityZone().getName(),
            subnet.getSubnetStatus());
    }
}
// snippet-end:[redshift.java.CreateAndModifyClusterSubnetGroup.complete]
Querying a database

To query databases hosted by your Amazon Redshift cluster, you have two options:

- Connect to your cluster and run queries on the AWS Management Console with the query editor.  
  If you use the query editor on the Amazon Redshift console, you don't have to download and set up a SQL client application.
- Connect to your cluster through a SQL client tool, such as SQL Workbench/J.

Amazon Redshift supports SQL client tools connecting through Java Database Connectivity (JDBC) and Open Database Connectivity (ODBC). Amazon Redshift doesn't provide or install any SQL client tools or libraries, so you must install them on your client computer or Amazon EC2 instance to use them. You can use most SQL client tools that support JDBC or ODBC drivers.

Topics
- Querying a database using the Amazon Redshift query editor v2 (p. 87)
- Querying a database using the query editor (p. 114)
- Scheduling a query on the Amazon Redshift console (p. 117)
- Connecting to an Amazon Redshift cluster using SQL client tools (p. 121)
- Using the Amazon Redshift Data API (p. 238)

Querying a database using the Amazon Redshift query editor v2

The query editor v2 is a separate web-based SQL client application that you use to author and run queries on your Amazon Redshift data warehouse. You can visualize your results in charts and collaborate by sharing your queries with others on your team. The query editor v2 is a replacement for the previous query editor.

Note
The query editor v2 is available in commercial AWS Regions where Amazon Redshift is available.

For a demo of query editor v2, watch the following video. Amazon Redshift query editor v2.

For a demo of data analysis, watch the following video. Data analysis using Amazon Redshift query editor v2.

The query editor v2 has a rich set of features to manage and run your SQL statements. The topics in the following sections get you started with many of these features. Explore the query editor v2 on your own to familiarize yourself with its capabilities.

Topics
- Configuring your AWS account (p. 88)
- Working with query editor v2 (p. 92)
Configuring your AWS account

When you choose the query editor v2 from the Amazon Redshift console, a new tab in your browser opens with the query editor v2 interface. With the proper permissions, you can access data in an Amazon Redshift cluster owned by your AWS account that is in the current AWS Region.

The first time an administrator configures query editor v2 for your AWS account, they choose the AWS KMS key that is used to encrypt query editor v2 resources. By default, an AWS owned key is used to encrypt resources. Or an administrator can use a customer managed key by choosing the Amazon Resource Name (ARN) for the key in the configuration page. After configuring an account, AWS KMS encryption settings can't be changed.

For more information about creating and using a customer managed key with query editor v2, see Creating an AWS KMS customer managed key to use with query editor v2 (p. 88).

Amazon Redshift query editor v2 supports authentication, encryption, isolation, and compliance to keep your data at rest and data in transit secure. For more information about data security and query editor v2, see the following:

- Encryption at rest (p. 323)
- Encryption in transit (p. 331)
- Configuration and vulnerability analysis in Amazon Redshift (p. 447)

The query editor v2 has adjustable quotas for some of its resources. For more information, see Amazon Redshift quotas (p. 505).

Resources created with query editor v2

Within query editor v2, you can create resources such as saved queries and charts. All resources in query editor v2 are associated with an IAM role or IAM user.

In the query editor v2, you can add and remove tags for saved queries and charts. You can use these tags when setting up custom IAM policies or to search for resources. You can also manage tags by using the AWS Resource Groups Tag Editor.

You can set up IAM roles and IAM users with IAM policies to share queries with others in your same AWS account in the AWS Region.

Creating an AWS KMS customer managed key to use with query editor v2

To create a symmetric customer managed key:

You can create a symmetric customer managed key to encrypt query editor v2 resources using the AWS KMS console or AWS KMS API operations. For instructions about creating a key, see Creating symmetric AWS KMS key in the AWS Key Management Service Developer Guide.
Key policy

Key policies control access to your customer managed key. Every customer managed key must have exactly one key policy, which contains statements that determine who can use the key and how they can use it. When you create your customer managed key, you can specify a key policy. For more information, see Managing access to AWS KMS keys in the AWS Key Management Service Developer Guide.

To use your customer managed key with Amazon Redshift query editor v2, the following API operations must be allowed in the key policy:

- **kms:GenerateDataKey** – Generates a unique symmetric data key to encrypt your data.
- **kms:Decrypt** – Decrypts data that was encrypted with the customer managed key.
- **kms:DescribeKey** – Provides the customer managed key details to allow the service to validate the key.

The following is a sample AWS KMS policy for AWS account 111122223333. In the first section, the **kms:ViaService** limits use of the key to the query editor v2 service (which is named sqlworkbench.region.amazonaws.com in the policy). The AWS account using the key must be 111122223333. In the second section, the root user and key administrators of AWS account 111122223333 can access to the key.

```
{
  "Version": "2012-10-17",
  "Id": "key-consolepolicy",
  "Statement": [
    {
      "Sid": "Allow access to principals authorized to use Amazon Redshift Query Editor V2",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "sqlworkbench.region.amazonaws.com",
          "kms:CallerAccount": "111122223333"
        }
      }
    },
    {
      "Sid": "Allow access for key administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "kms:*"
      ],
    }
  ]
}
```

The following resources provide more information about AWS KMS keys:
For more information about AWS KMS policies, see Specifying permissions in a policy in the AWS Key Management Service Developer Guide.

For information about troubleshooting AWS KMS policies, see Troubleshooting key access in the AWS Key Management Service Developer Guide.

For more information about keys, see AWS KMS keys in the AWS Key Management Service Developer Guide.

Accessing the query editor v2

To access the query editor v2, you need permission. An administrator can attach one of the following AWS managed policies to the IAM user or role to grant permission. These AWS managed policies are written with different options that control how tagging resources allows sharing of queries. You can use the IAM console (https://console.aws.amazon.com/iam/) to attach IAM policies.

- **AmazonRedshiftQueryEditorV2FullAccess** – Grants full access to the Amazon Redshift query editor v2 operations and resources. This policy also grants access to other required services.
- **AmazonRedshiftQueryEditorV2NoSharing** – Grants the ability to work with Amazon Redshift query editor v2 without sharing resources. This policy also grants access to other required services.
- **AmazonRedshiftQueryEditorV2ReadSharing** – Grants the ability to work with Amazon Redshift query editor v2 with limited sharing of resources. The granted principal can read the resources shared with its team but can't update them. This policy also grants access to other required services.
- **AmazonRedshiftQueryEditorV2ReadWriteSharing** – Grants the ability to work with Amazon Redshift query editor v2 with sharing of resources. The granted principal can read and update the resources shared with its team. This policy also grants access to other required services.

You can also create your own policy based on the permissions allowed and denied in the provided managed policies. If you use the IAM console policy editor to create your own policy, choose SQL Workbench as the service for which you create the policy in the visual editor. The query editor v2 uses the service name AWS SQL Workbench in the visual editor and IAM Policy Simulator.

For a principal (an IAM user or IAM role) to connect to an Amazon Redshift cluster, they need the permissions in one of the query editor v2 managed policies. They also need the redshift:GetClusterCredentials permission to the cluster. To get this permission, someone with administrative permission can attach a policy to the IAM users or IAM roles that need to connect to the cluster by using temporary credentials. You can scope the policy to specific clusters or be more general. For more information about permission to use temporary credentials, see Create an IAM role or user with permissions to call GetClusterCredentials (p. 373).

For a principal (an IAM user or IAM role) to use the SQL Notebooks (preview) feature, you must add the following policy to one of the principal's existing query editor v2 managed policies.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AmazonRedshiftQueryEditorV2NonResourceLevelPermissions",
            "Effect": "Allow",
            "Action": [
                "sqlworkbench:ListNotebooks"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AmazonRedshiftQueryEditorV2CreateOwnedResourcePermissions",
            "Effect": "Allow",
            "Action": [
```
For more information about managed policies in Amazon Redshift, see AWS-managed (predefined) policies for Amazon Redshift (p. 341).

You can use the IAM console to attach IAM policies to an IAM user or an IAM role. After you attach a policy to a role, you can attach the role to an IAM user.

**To attach the IAM policies to an IAM user**

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. Choose Users.
3. Choose the user that needs access to the query editor v2.
4. Choose Add permissions.
5. Choose **Attach existing policies directly**.
6. For **Policy names**, choose the proper policies as previously described.
7. Choose **Next: Review**.
8. Choose **Add permissions**.

**To attach the IAM policies to an IAM role**

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. Choose **Roles**.
3. Choose the role that needs access to the query editor v2.
4. Choose **Attach policies**.
5. For **Policy names**, choose the proper policies as previously described.
6. Choose **Attach policy**.

For more information about using IAM to manage users and roles, see Changing permissions for an IAM user in the IAM User Guide.

**Working with query editor v2**

The query editor v2 is primarily used to edit and run queries, visualize results, and share your work with your team. With query editor v2, you can create databases, schemas, tables, and user-defined functions (UDFs). In a tree-view panel, for each of your clusters, you can view its schemas. For each schema, you can view its tables, views, UDFs, and stored procedures.

**Topics**

- Opening query editor v2 (p. 92)
- Connecting to an Amazon Redshift database (p. 93)
- Browsing an Amazon Redshift database (p. 94)
- Creating database objects (p. 96)
- Loading sample data (p. 97)
- Loading data from Amazon S3 (p. 98)
- Working with SQL notebooks (preview) (p. 99)

**Opening query editor v2**

**To open the query editor v2**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. From the navigator menu, choose **Editor**, then **Query editor V2**.

The query editor v2 opens in a new tab.

The query editor page has a navigator menu where you choose **Database** to work with data in your cluster, **Queries** to work with saved queries, and **Charts** to work with saved charts. The navigator menu is similar to the following.
When working in the **Database** view, you have the following controls:

- The **Cluster** field displays the name of the cluster you are currently connected to. The **Database** field displays the databases within the cluster. The actions that you perform in the **Database** view default to act on the database you have selected.
- A tree-view hierarchical view of your clusters, databases, and schemas. Under schemas, you can work with your tables, views, functions, and stored procedures. Each object in the tree view supports a context menu to perform associated actions, such as **Refresh** or **Drop**, for the object.
  - A **Create** action to create databases, schemas, tables, and functions.
  - A **Load data** action to load data from Amazon S3 into your databases.
  - A preferences icon to edit your preferences.
  - A **Save** icon to save your query.
  - A **Shortcuts** icon to display keyboard shortcuts for the editor.
- An editor area where you can enter and run your query.
- After you run a query, a **Result** tab appears with the results. Here is where you can turn on **Chart** to visualize your results.

### Connecting to an Amazon Redshift database

To connect to a database, choose the cluster name in the tree-view panel. If prompted, enter the connection parameters.

When you connect to a cluster and its databases, you provide a **Database** name and **User name**. You also provide parameters required for one of the following authentication methods:

**Database user name and password**

With this method, also provide a **Password** for the database that you are connecting to. The query editor v2 creates a secret on your behalf stored in AWS Secrets Manager. This secret contains credentials to connect to your database.
Temporary credentials

With this method, query editor v2, generates a temporary password to connect to the database.

When you select a cluster with query editor v2, depending on the context, you can create, edit, and delete connections using the context (right-click) menu.

Browsing an Amazon Redshift database

Within a database, you can manage schemas, tables, views, functions, and stored procedures in the tree-view panel. Each object in the view has actions associated with it in a context (right-click) menu.

After you choose a table, you can do the following:

• To start a query in the editor with a SELECT statement that queries all columns in the table, use Select table.

• To see the attributes or a table, use Show table definition. Use this to see column names, column types, encoding, distribution keys, sort keys, and whether a column can contain null values. For more information about table attributes, see CREATE TABLE in the Amazon Redshift Database Developer Guide.

• To delete a table, use Delete. You can either use Truncate table to delete all rows from the table or Drop table to remove the table from the database. For more information, see TRUNCATE and DROP TABLE in the Amazon Redshift Database Developer Guide.

Choose a schema to Refresh or Drop schema.

Choose a view to Show view definition or Drop view.

Choose a function to Show function definition or Drop function.

Choose a stored procedure to Show procedure definition or Drop procedure.

The hierarchical tree-view panel is similar to the following. Open the context (right-click) menu for an icon to see what actions you can perform.
Creating database objects

You can create database objects, including databases, schemas, tables, and user-defined functions (UDFs).

To create a database

For information about databases, see CREATE DATABASE in the Amazon Redshift Database Developer Guide.

1. Choose Create, and then choose Database.
2. Enter a Database name.
3. (Optional) Select Users and groups, and choose a Database user.
4. Choose Create database.
   
    The new database displays in the tree-view panel.

To create a schema

For information about schemas, see Schemas in the Amazon Redshift Database Developer Guide.

1. Choose Create, and then choose Schema.
2. Enter a Schema name.
3. (Optional) Select Authorize user, and choose a Database user.
4. Choose Create schema.
   
    The new schema appears in the tree-view panel.

To create a table

You can create a table based on a comma-separated value (CSV) file that you specify or define each column of the table. For information about tables, see Designing tables and CREATE TABLE in the Amazon Redshift Database Developer Guide.

Choose Open query in editor to view and edit the CREATE TABLE statement before you run the query to create the table.

1. Choose Create and select Table.
2. Choose a schema.
3. Enter a table name.
4. Choose Add field to add a column.
5. Use a CSV file as a template for the table definition:
   
    a. Choose Load from CSV.
    b. Browse to the file location.
    
       If you use a CSV file, be sure that the first row of the file contains the column headings.
    c. Choose the file and choose Open. Confirm that the column names and data types are what you intend.
6. For each column, choose the column and choose the options that you want:
   • Choose a value for Encoding.
   • Choose a Default value.
   • Turn on Automatically increment if you want the column values to increment. Then specify a value for Auto increment seed and Auto increment step.
   • Turn on Not NULL if the column should always contain a value.
   • Enter a Size value for the column.
   • Turn on Primary key if you want the column to be a primary key.
   • Turn on Unique key if you want the column to be a unique key.

7. (Optional) Choose Table details and choose any of the following options:
   • Distribution key column and style.
   • Sort key column and sort type.
   • Turn on Backup to include the table in cluster snapshots.
   • Turn on Temporary table to create the table as a temporary table.

8. Choose Open query in editor to continue specifying options to define the table or choose Create table to create the table.

To create a function

1. Choose Create, and choose Function.
2. For Type, choose SQL or Python.
4. Enter a value for Name for the function.
5. Enter a value for Volatility for the function.
6. Choose Parameters by their data types in the order of the input parameters.
7. For Returns, choose a data type.
8. Enter the SQL program code for the function.
9. Choose Create.

For more information about user-defined functions (UDFs), see Creating user-defined functions in the Amazon Redshift Database Developer Guide.

Loading sample data

The query editor v2 comes with sample data and queries available to be loaded into a sample database and corresponding schema.

To load sample data, choose the icon associated with the sample data you want to load. The query editor v2 then loads the data into a schema in database sample_data_dev and creates a folder of saved queries in your Queries folder.

The following sample datasets are available.

ticket

Most of the examples in the Amazon Redshift documentation use a sample data called ticket. This data consists of seven tables: two fact tables and five dimensions. When you load this data the
schema tickit is updated with sample data. For more information about the tickit data, see Sample database in the Amazon Redshift Database Developer Guide.

tpch

This data is used for a decision support benchmark. When you load this data the schema tpcch is updated with sample data. For more information about the tpcch data, see TPC-H.

tpcds

This data is used for a decision support benchmark. When you load this data the schema tpcds is updated with sample data. For more information about the tpcds data, see TPC-DS.

Loading data from Amazon S3

You can load data into an existing table from Amazon S3.

To load data into an existing table

The COPY command is used by query editor v2 to load data from Amazon S3. The COPY command generated and used in the query editor v2 Load data wizard supports all the parameters available to the COPY command syntax to copy from Amazon S3. For information about the COPY command and its options used to load data from Amazon S3, see COPY from Amazon Simple Storage Service in the Amazon Redshift Database Developer Guide.

1. Confirm that the table is already created in the database where you want to load data. The query editor v2 can only load data into an existing table.
2. Choose Load data.
3. In S3 URIs, choose Browse S3 to look for the Amazon S3 bucket that contains the data to load.
4. If the specified Amazon S3 bucket isn't in the same AWS Region with the target Amazon Redshift cluster, then choose the S3 file location for the AWS Region where the data is located.
5. Choose This file is a manifest file if the Amazon S3 file is actually a manifest containing multiple Amazon S3 bucket URIs.
6. Choose an IAM role that has the required permissions to load data from Amazon S3.
7. Choose the File format for the file to be uploaded. The supported data formats are CSV, JSON, DELIMITER, FIXEDWIDTH, SHAPEFILE, AVRO, PARQUET, and ORC. Depending on the specified file format, you can choose the respective File options. You can also select Data is encrypted if the data is encrypted and enter the Amazon Resource Name (ARN) of the KMS key used to encrypt the data.

For PARQUET and ORC, there isn't any file option to configure.
8. Choose a compression method to compress your file. The default is no compression.
9. (Optional) The Advanced settings support various Data conversion parameters and Load operations. Enter this information as needed for your file.

For more information about data conversion and data load parameters, see Data conversion parameters and Data load operations in the Amazon Redshift Database Developer Guide.
10. Confirm or choose the location of the Target table including database, schema, and table name where the data is loaded.
11. Choose Load data to start the data load.

When the load completes, the query editor displays with the generated COPY command that was used to load your data. The Result of the COPY is shown. If successful, you can now use SQL to select data from the loaded table. When there is an error, query the system view STL_LOAD_ERRORS to get more details. For information about COPY command errors, see STL_LOAD_ERRORS in the Amazon Redshift Database Developer Guide.
Working with SQL notebooks (preview)

This is prerelease documentation for query editor v2 notebooks, which is in preview release. The documentation and the feature are both subject to change. We recommend that you use this feature only in test environments, and not in production environments. For preview terms and conditions, see Beta Service Participation in AWS Service Terms.

You can use SQL notebooks to organize, annotate, and share multiple SQL queries in a single document. You can add multiple SQL query and Markdown cells to an SQL notebook. SQL notebooks provide a way to group queries and explanations associated with a data analysis in a single document by using multiple query and Markdown cells. You can add text and format the appearance using Markdown syntax to provide context and additional information for your data analysis tasks. You can share your SQL notebooks with team members.

To use the SQL notebook feature, you must add a policy for the SQL notebook (preview) feature to a principal (an IAM user or IAM role) that already has one of the query editor v2 managed policies. For more information, see Accessing the query editor v2 (p. 90).

For a demo of notebooks, watch the following video. Introduction to SQL Notebooks.

To create an SQL notebook

1. Choose + and then choose Notebook.

   By default, an SQL query cell appears in the new SQL notebook.

2. (Optionally) Choose Rename and enter a name for the SQL notebook.

3. In the SQL query cell, do one of the following:
   - Enter a query.
   - Paste a query that you copied.

4. (Optionally) Choose Add markdown to add a Markdown cell where you can provide descriptive or explanatory text using standard Markdown syntax.

5. (Optionally) Choose Add SQL and Add markdown to insert additional SQL and Markdown text cells.

To open a notebook

1. From the navigator menu, choose Notebooks.

2. Choose the SQL notebook that you want to open and double-click it.

To share an SQL notebook with your team

- Choose Share to team-name.

Authoring and running queries

You can enter a query in the editor or select a saved query from the Queries list and choose Run.

By default, Limit 100 is set to limit the results to 100 rows. You can turn off this option to return a larger result set. If you turn off this option, you can include the LIMIT option in your SQL statement if you want to avoid very large result sets. For more information, see ORDER BY clause in the Amazon Redshift Database Developer Guide.
To display a query plan in the results area, turn on **Explain**.

To save a query to the **Queries** folder, choose **Save**.

For a successful query, a success message appears. If the query returns information, the results display in the **Results** section. If the number of results exceeds the display area, numbers appear at the top of the results area. You can choose the numbers to display successive pages of results.

You can filter and sort **Result** for each column. To enter filter criteria in the result column header, hover over the column to see a menu (≡) where you can enter criteria to filter the column.

If the query contains an error, the query editor v2 displays an error message in the results area. The message provides information on how to correct the query.

You can export the query results on the current page to a file in JSON or comma-separated value (CSV) format. To save the file in either format, open the context (right-click) menu in the results area, then choose **Export current page** and either **JSON** or **CSV**. You can also select rows and export the results for specific rows.

To add a new query tab, choose the + tab, which appears in the row with the query tabs.

**To run a query**

1. In the query area, do one of the following:
   - Enter a query.
   - Paste a query that you copied.
   - Choose the **Queries** folder, open the context menu (right-click) a saved query, and choose **Open query**.
2. Confirm that you chose the correct **Cluster** and **Database** value for the SQL you plan to run.
   - Change your **Cluster** value by choosing another cluster in the tree view. Change your database by choosing a different **Database** in the interface.
3. Choose **Run**.

   The **Result** area opens and displays the query results.

**To display the explain plan for a query**

1. Select the query.
2. Turn on **Explain**.
3. Choose **Run**.

   By default, the **Explain graph** is also on.

3. Choose **Run**.

   The query runs and the explain plan displays in the query **Result** area.

The query editor v2 supports the following features:

- You can author queries with multiple SQL statements in one query tab. The queries are run in serially and multiple results tabs open for each query.
- You can run queries in parallel by opening a query tab, choose a database, run the query. While that query is running, you can choose a different database (and different cluster if you want), and run another query.
• You can author queries with session variables and temporary tables.
• You can author queries with replaceable parameters designated by \( \$\{\text{parameter}\} \). You can author your SQL query with multiple replaceable parameters and use the same parameter in multiple places in your SQL statement.

When the query runs, a window is presented to enter the value of the parameter. Each time you run the query, the window is presented to enter your parameter values.

For an example, see Example: Sales greater than a specific parameter (p. 102).
• Queries are versioned automatically. You can choose an earlier version of a query to run.
• You don't need to wait for a query to complete before continuing on with your workflow. Your queries continue to run even if you close the query editor.
• When authoring queries, automatic completion of schema, table, and column names is supported using a shortcut.

Choose the `Shortcuts` icon to find the `Auto complete` shortcut.

The following examples demonstrate some of these features.

Query examples

Following, you can find descriptions of the various types of queries that you can run.

The data used in many of these queries is from the `tickit` sample schema. For more information about the `tickit` data, see Loading sample data (p. 97).

When you run these example queries, confirm that you choose the correct database in the editor, such as `sample_data_dev`.

Topics
• Example: Setting session variables (p. 101)
• Example: Top event by total sales (p. 101)
• Example: Sales greater than a specific parameter (p. 102)
• Example: Create a temporary table (p. 102)
• Example: Selecting from a temporary table (p. 102)

Example: Setting session variables

The following command sets the `search_path` server configuration parameter to `public` for the session. For more information, see `SET` and `search_path` in the Amazon Redshift Database Developer Guide.

```sql
set search_path to public;
```

Example: Top event by total sales

The following query finds the event with the most sales.

```sql
select eventname, count(salesid) totalorders, sum(pricepaid) totalsales
from sales, event
where sales.eventid=event.eventid
group by eventname
```
order by 3;

Following is a partial list of the results.

<table>
<thead>
<tr>
<th>eventname</th>
<th>totalorders</th>
<th>totalsales</th>
</tr>
</thead>
<tbody>
<tr>
<td>White Christmas</td>
<td>20</td>
<td>9352</td>
</tr>
<tr>
<td>Joshua Radin</td>
<td>38</td>
<td>23469</td>
</tr>
<tr>
<td>Beach Boys</td>
<td>58</td>
<td>30383</td>
</tr>
<tr>
<td>Linda Ronstadt</td>
<td>56</td>
<td>35043</td>
</tr>
<tr>
<td>Rascal Flatts</td>
<td>76</td>
<td>38214</td>
</tr>
<tr>
<td>Billy Idol</td>
<td>67</td>
<td>40101</td>
</tr>
<tr>
<td>Stephenie Meyer</td>
<td>72</td>
<td>41509</td>
</tr>
<tr>
<td>Indigo Girls</td>
<td>57</td>
<td>45399</td>
</tr>
</tbody>
</table>

**Example: Sales greater than a specific parameter**

The following query finds sales where the quantity sold is greater than the parameter specified by \${numberoforders}. When the parameter value is 7, the result is 60 rows. When you run the query, the query editor v2 displays a Run query form window to gather the value of parameters in the SQL statement.

```sql
select salesid, qtysold
from sales
where qtysold > \${numberoforders}
order by 2;
```

Following is a partial list of the results.

<table>
<thead>
<tr>
<th>salesid qtysold</th>
</tr>
</thead>
<tbody>
<tr>
<td>20005 8</td>
</tr>
<tr>
<td>21279 8</td>
</tr>
<tr>
<td>130232 8</td>
</tr>
<tr>
<td>42737 8</td>
</tr>
<tr>
<td>74681 8</td>
</tr>
<tr>
<td>67103 8</td>
</tr>
<tr>
<td>105533 8</td>
</tr>
<tr>
<td>91620 8</td>
</tr>
<tr>
<td>121552 8</td>
</tr>
</tbody>
</table>

**Example: Create a temporary table**

The following statement creates the temporary table `eventsalestemp` by selecting information from the `sales` and `event` tables.

```sql
create temporary table eventsalestemp as
select eventname, count(salesid) totalorders, sum(pricepaid) totalsales
from sales, event
where sales.eventid=event.eventid
group by eventname;
```

**Example: Selecting from a temporary table**

The following statement selects events, total orders, and total sales from the temporary table `eventsalestemp`, ordered by total orders.

```sql
select eventname, totalorders, totalsales
```
from eventsalestemp
order by 2;

Following is a partial list of results.

<table>
<thead>
<tr>
<th>eventname</th>
<th>totalorders</th>
<th>totalsales</th>
</tr>
</thead>
<tbody>
<tr>
<td>White Christmas</td>
<td>20</td>
<td>9352</td>
</tr>
<tr>
<td>Joshua Radin</td>
<td>38</td>
<td>23469</td>
</tr>
<tr>
<td>Martina McBride</td>
<td>50</td>
<td>52932</td>
</tr>
<tr>
<td>Linda Ronstadt</td>
<td>56</td>
<td>35043</td>
</tr>
<tr>
<td>Indigo Girls</td>
<td>57</td>
<td>45399</td>
</tr>
<tr>
<td>Beach Boys</td>
<td>58</td>
<td>30383</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Considerations for querying

Consider the following information about working with queries when you use the query editor:

- The maximum query result size is the smaller of 5 MB or 100,000 rows.
- You can export only the current page of results or selected result rows to a JSON or CSV file.
- You can run a query up to 300,000 characters long.
- You can save a query up to 30,000 characters long.
- You can't use transactions in the editor. For more information about transactions, see BEGIN in the Amazon Redshift Database Developer Guide.

Querying a data lake

You can query data in an Amazon S3 data lake. First, you create an external schema to reference the external database in the AWS Glue Data Catalog. Then, you can query data in the Amazon S3 data lake.

Demo: Query a data lake

For a demo of how to query a data lake, watch the following video. Query your data lake from Amazon Redshift query editor v2.

Prerequisites

Creating an IAM role

You create an IAM role to access an AWS Glue Data Catalog enabled for AWS Lake Formation.

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Policies.
   
   If this is your first time choosing Policies, the Welcome to Managed Policies page appears. Choose Get Started.
3. Choose Create policy.
4. Choose to create the policy on the JSON tab.
5. Paste in the following JSON policy document, which grants access to the Data Catalog but denies the administrator permissions for Lake Formation.

```json
{

```
"Version": "2012-10-17",
"Statement": [
{
"Sid": "RedshiftPolicyForLF",
"Effect": "Allow",
"Action": [
"glue:*",
"lakeformation:GetDataAccess"
],
"Resource": "*"
}
]

6. When you are finished, choose Review to review the policy. The policy validator reports any syntax errors.

7. On the Review policy page, for Name enter a name for the policy that you are creating, for example, mydatalake_policy. Enter a Description (optional). Review the policy Summary to see the permissions that are granted by your policy. Then choose Create policy to save your work.

After you create a policy, you can create a role and apply the policy.

8. In the navigation pane of the IAM console, choose Roles, and then choose Create role.

9. For Select type of trusted entity, choose AWS service.

10. Choose the Amazon Redshift service to assume this role.

11. Choose the Redshift Customizable use case for your service. Then choose Next: Permissions.

12. Choose the permissions policy that you created, mydatalake_policy, to attach to the role.


15. For Role name, enter a name for the role, for example, mydatalake_role.

16. (Optional) For Role description, enter a description for the new role.

17. Review the role, and then choose Create role.

Granting permissions

You grant SELECT permissions on the table to query in the Lake Formation database.


2. In the navigation pane, choose Permissions, and then choose Grant.

3. Provide the following information:

   - For IAM role, choose the IAM role you created, myspectrum_role. When you run the Amazon Redshift Query Editor, it uses this IAM role for permission to the data.

     **Note**
     To grant SELECT permission on the table in a Lake Formation–enabled Data Catalog to query, do the following:
     - Register the path for the data in Lake Formation.
     - Grant users permission to that path in Lake Formation.
     - Created tables can be found in the path registered in Lake Formation.

   - For Database, choose your Lake Formation database.
   - For Table, choose a table within the database to query.
   - For Columns, choose All Columns.
   - Choose the Select permission.

4. Choose Save.
Important
As a best practice, allow access only to the underlying Amazon S3 objects through Lake Formation permissions. To prevent unapproved access, remove any permission granted to Amazon S3 objects outside of Lake Formation. If you previously accessed Amazon S3 objects before setting up Lake Formation, remove any IAM policies or bucket permissions that previously were set up. For more information, see Upgrading AWS Glue Data Permissions to the AWS Lake Formation Model and Lake Formation Permissions.

Creating the external schema

To query data in an Amazon S3 data lake, you create an external schema. The external schema references the external database in the AWS Glue Data Catalog.

1. Choose Create, and then choose Schema.
2. Enter a schema name.
3. To grant ownership of the database to a user, choose Authorize user and choose a user.
4. Choose External.
5. Under AWS Glue Data Catalog details, Region defaults to the Region where your Redshift database is located.
6. Choose the AWS Glue database that the external schema will map to.
7. Choose an IAM role that has the required permissions to query data on Amazon S3.
8. Choose Create schema.

The schema appears in the database browser.

Querying data in your Amazon S3 data lake

You use the schema that you created in the previous procedure.

1. In the database browser, choose the schema.
2. To view a table definition, choose a table.

   The table columns and data types display.
3. To query a table, choose the table and use the context menu (right-click) to choose Select table.

Working with datashares

You can create a datashare so that users on another cluster can query the data. The cluster containing the data that you want to share is called the producer cluster. You create a datashare on the producer cluster for the database objects that you want to share. You can share schemas, tables, views, and SQL user-defined functions (UDFs). The cluster that you want to share the data to is called the consumer cluster. On the consumer cluster, you create a database from the datashare. Then, users on the consumer cluster can query the data. For more information, see Getting started data sharing in the Amazon Redshift Database Developer Guide.

Creating datashares

You create a datashare on the cluster that you want to use as the producer cluster. To learn more about datashare considerations, see Data sharing considerations in Amazon Redshift in the Amazon Redshift Database Developer Guide.

1. Choose the database on the producer cluster that you want to use.
2. Create the datashare. For example:

```
create datashare mysourse;
```

3. Set permissions on the datashare. For example:

```
grant alter, share on datashare mysourse to admin;
```

4. Set permissions on the database objects that you want to share. For example:

```
alter datashare mysourse add schema public;
alter datashare mysourse add table public.event;
```

5. Set permissions on the consumer cluster namespace to access the datashare. For example:

```
grant usage on datashare mysourse to namespace '2b12345-1234-5678-9012-bb1234567890';
```

### Showing datashares

You can show the datashares that you've created on the producer cluster.

1. Choose the producer cluster.
2. Show the datashares. For example:

```
show datashares;
```

<table>
<thead>
<tr>
<th>share_name</th>
<th>share_owner</th>
<th>source_database</th>
<th>consumer_database</th>
<th>share_type</th>
<th>createdate</th>
<th>is_publicaccessible</th>
<th>share_acl</th>
<th>producer_account</th>
<th>producer_namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_datashare</td>
<td>100</td>
<td>db_producer</td>
<td>NULL</td>
<td>OUTBOUND</td>
<td>2/15/2022</td>
<td>FALSE</td>
<td>admin</td>
<td>123456789012</td>
<td>p1234567-8765-4321</td>
</tr>
</tbody>
</table>

### Creating the consumer database

On the consumer cluster, you create a database from the datashare. These steps describe how to share data between two clusters in the same account. For information on sharing data across AWS accounts, see [Sharing data across AWS accounts](#) in the [Amazon Redshift Database Developer Guide](#).

You can use SQL commands or the query editor v2 tree-view panel to create the database.

**To use SQL**

1. Create a database from the datashare for your account and the namespace of the producer cluster. For example:

```
create database share_db from datashare mysourse of account '123456789012' namespace 'p1234567-8765-4321-p10987654321';
```

2. Set permissions so that users can access the database and the schema. For example:

```
grant usage on database share_db to usernames;
```
grant usage on schema public to usernames;

To use the query editor v2 tree-view panel

1. Choose Create, and then choose Database.
2. Enter a Database name.
3. (Optional) Select Users and groups, and choose a Database user.
4. Choose Create using a datashare.
5. Choose the datashare.
6. Choose Create database.

The new datashare database displays in the query editor v2 tree-view panel.
7. Set permissions so that users can access the database and the schema. For example:

grant usage on database share_db to usernames;

grant usage on schema public to usernames;

Querying datashare objects

On the consumer cluster, you can query datashare objects using fully qualified object names expressed with the three-part notation: database name, schema, and name of the object.

1. In the query editor v2 tree-view panel, choose the schema.
2. To view a table definition, choose a table.

The table columns and data types display.
3. To query a table, choose the table and use the context menu (right-click) to choose Select table.
4. Query tables using SELECT commands. For example:

```sql
select top 10 * from test_db.public.event;
```

Visualizing query results

After you run a query and the results display, you can turn on Chart to display a graphic visualization of the results. You can use the following controls to define the content, structure, and appearance of your chart:

+ Trace

Represents a set of related graphical marks in a chart. You can define multiple traces in a chart.
Type

You can define the trace type to represent data as one of the following:
• Scatter chart for a scatter plot or bubble chart.
• Bar chart to represent categories of data with vertical or horizontal bars.
• Area chart to define filled areas.
• Histogram that uses bars to represent frequency distribution.
• Pie chart for a circular representation of data where each slice represents a percentage of the whole.
• Funnel or Funnel Area chart to represent data through various stages of a process.
• OHLC (open-high-low-close) chart often used for financial data to represent open, high, low, and close values along the x-axis, which usually represents intervals of time.
• Candlestick chart to represent a range of values for a category over a timeline.
• Waterfall chart to represent how an initial value increases or decreases through a series of intermediate values. Values can represent time intervals or categories.
• Line chart to represent changes in value over time.

X axis
You specify a table column that contains values to plot along the X axis. Columns that contain descriptive values usually represent dimensional data. Columns that contain quantitative values usually represent factual data.

Y axis
You specify a table column that contains values to plot along the Y axis. Columns that contain descriptive values usually represent dimensional data. Columns that contain quantitative values usually represent factual data.

Subplots
You can define additional presentations of chart data.

Transforms
You can define transforms to filter trace data. You use a split transform to display multiple traces from a single source trace. You use an aggregate transform to present a trace as an average or minimum. You use a sort transform to sort a trace.

General appearance
You can set defaults for background color, margin color, color scales to design palettes, text style and sizes, title style and size, and mode bar. You can define interactions for drag, click, and hover. You can define meta text. You can define default appearances for traces, axes, legends, and annotations.

Choose **Traces** to display the results as a chart. For **Type**, choose the style of chart as **Bar, Line**, and so on. For **Orientation**, you can choose **Vertical** or **Horizontal**. For **X**, choose the table column that you want to use for the horizontal axis. For **Y**, choose the table column that you want to use for the vertical axis.

Choose **Refresh** to update the chart display. Choose **Full screen** to expand the chart display.

**To create a chart**

1. Run a query and get results.
2. Turn on **Charts**.
3. Choose **Trace** and start to visualize your data.
4. Choose a chart style from one of the following:
   • Scatter
   • Bar
   • Area
   • Histogram
• Pie
• Funnel
• Funnel Area
• OHLC (open-high-low-close)
• Candlestick
• Waterfall
• Line

5. Choose **Style** to customize the appearance, including colors, axes, legend, and annotations. You can add text, shapes, and images.

6. Choose **Annotations** to add text, shapes, and images.

**To save a chart**

1. Choose **Save Chart**.
2. Enter a name for your chart.
3. Choose **Save**.

**To export a chart**

1. Choose **Export**.
2. Choose **PNG** or **JPEG**.
3. Set the width and height for your chart.
4. Choose **Export**.
5. Choose to open the file in your default graphic application or save the file with the default name.

**To browse for and open a saved chart**

1. Choose the **Charts** tab.
2. Open the chart that you want.

**To organize your charts into folders**

1. Choose **Charts** from the navigation pane.
2. Choose **New folder** and name the folder.
3. Choose **Create** to create the folder in the **Charts** tab.

You can move charts in and out of the folder using drag-and-drop.

**Example: Create a pie chart to visualize query results**

The following example uses the `Sales` table of the sample database. For more information, see **Sample database** in the **Amazon Redshift Database Developer Guide**.

Following is the query that you run to provide the data for the pie chart.

```sql
select top 5 eventname, count(salesid) totalorders, sum(pricepaid) totalsales
from sales, event
where sales.eventid=event.eventid group by eventname
order by 3;
```
To create a pie chart for the top event by total sales

1. Run the query.
2. In the query results area, turn on Chart.
3. Choose Trace.
4. For Type, choose Pie.
5. For Values, choose totalsales.
6. For Labels, choose eventname.
7. Choose Style and then General.
8. Under Colorscales, choose Categorical and then Pastel2.

Example: Create a combination chart for comparing revenue and sales

Perform the steps in this example to create a chart that combines a bar chart for revenue data and a line graph for sales data. The following example uses the Sales table of the tickit sample database. For more information, see Sample database in the Amazon Redshift Database Developer Guide.

Following is the query that you run to provide the data for the chart.

```sql
select eventname, total_price, total_qty_sold
from (select eventid, total_price, total_qty_sold, ntile(1000) over(order by total_price desc) as percentile
from (select eventid, sum(pricepaid) total_price, sum(qtysold) total_qty_sold
from tickit.sales
group by eventid)) Q, tickit.event E
where Q.eventid = E.eventid
and percentile = 1
order by total_price desc;
```

To create a combination chart for comparing revenue and sales

1. Run the query.
2. In the query results area, turn on Chart.
3. Under trace o, for Type, choose Bar.
4. For X, choose eventname.
5. For Y, choose total_price.
The bar chart displays with event names along the X axis.

6. Under **Style**, choose **Traces**.
7. For **Name**, enter *Revenue*.
8. Under **Style**, choose **Axes**.
9. For **Titles**, choose Y and enter *Revenue*.

   The label *Revenue* displays on the left Y axis.

10. Under **Structure**, choose **Traces**.
11. Choose **Trace**.

   The trace 1 options display.

12. For **Type**, choose **Line**.
13. For X, choose *eventname*.
14. For Y, choose *total_qty_sold*.
15. Under **Axes To Use**, for Y Axis choose **+**.

   The Y Axis displays Y2.

16. Under **Style**, choose **Axes**.
18. For **Name**, enter *Sales*.
20. Under **Axis Line**, choose **Show** and for **Position**, choose **Right**.
Demo: Build visualizations using Amazon Redshift query editor v2

For a demo of how to build visualizations, watch the following video. Build visualizations using Amazon Redshift query editor v2.

Collaborating and sharing as a team

You can share queries with your team.

Sharing queries and collaborating with your team

You can share queries with others on your team. A team is set up by an administrator and is based on the IAM policy associated with an IAM user or IAM role. For information about how to set up a team within query editor v2, see Permissions required to use the query editor v2 (p. 346).

Saving, browsing for, and deleting queries

Before you can share your query with your team, save your query. You can view and delete saved queries.
To save a query
1. Prepare your query and choose **Save**.
2. Enter a title for your query.
3. Choose **Save**.

To browse for saved queries
1. Choose **Queries** from the navigation pane.
2. You can view queries that are **My queries**, **Shared by me**, or **Shared to my team**. These queries can appear as individual queries or within folders you created.

To delete a saved query
1. Open the context (right-click) menu for a saved query.
2. Choose **Delete** and confirm the action.

To organize your saved queries into folders
1. Choose **Queries** from the navigation pane.
2. Choose **New folder** and name the folder.
3. Choose **Create** to create the folder in the **Queries** tab.
   You can now move queries in and out of the folder using drag-and-drop.

Sharing a query
You can share your queries with your team. You can also view the history of saved queries and manage query versions.

To share a query with your team, make sure that you have the principal tag `sqlworkbench-team` set to the same value as the rest of your team members in your account. For example, an administrator might set the value to `accounting-team` for everyone in the accounting department.

To share a query with a team
1. Choose **Queries** from the navigation pane.
2. Open the context (right-click) menu of the query that you want to share and choose **Share with my team**.
3. Choose the team or teams that you want to share the query with and then choose **Save sharing options**.

Every time you save an SQL query, the query editor v2 saves it as a new version. You can browse earlier query versions, save a copy of a query, or restore a query.

To manage query versions
1. Choose **Queries** from the navigation pane.
2. Open the context (right-click) menu for the query that you want to work with.
3. Choose **Version history** to open a list of versions of the query.
4. On the **Version history** page, you can do the following:
Querying a database using the query editor

Using the query editor is an easy way to run queries on databases hosted by your Amazon Redshift cluster. After creating your cluster, you can immediately run queries by using the query editor on the Amazon Redshift console.

In February 2021, an updated query editor was deployed and authorization permissions to use the query editor changed. The new query editor uses the Amazon Redshift Data API to run queries. The AmazonRedshiftQueryEditor policy, which is an AWS-managed AWS Identity and Access Management (IAM) policy, was updated to include the necessary permissions. If you have a custom IAM policy, make sure that you update it. Use AmazonRedshiftQueryEditor as a guide. The changes to AmazonRedshiftQueryEditor include the following:

- Permission to manage query editor statement results requires the statement owner user.
- Permission to use Secrets Manager to connect to a database has been added.

For more information, see Permissions required to use the Amazon Redshift console query editor (p. 346).

When you connect to your cluster from the new query editor, you can use one of two authentication methods, as described in Connecting with the query editor (p. 116).

Using the query editor, you can do the following:

- Run single SQL statement queries.
- Download result sets as large as 100 MB to a comma-separated value (CSV) file.
- Save queries for reuse. You can't save queries in the Europe (Paris) Region, the Asia Pacific (Osaka) Region, the Asia Pacific (Hong Kong) Region, or the Middle East (Bahrain) Region.
- View query runtime details for user-defined tables.
- Schedule queries to run at a future time.
- View a history of queries that you created in the query editor.
- Run queries against clusters using enhanced VPC routing.

Query editor considerations

Consider the following about working with queries when you use the query editor:

- The maximum duration of a query is 24 hours.
- The maximum query result size is 100 MB. If a call returns more than 100 MB of response data, the call is terminated.
- The maximum retention time for query results is 24 hours.
- The maximum query statement size is 100 KB.
- The cluster must be in a virtual private cloud (VPC) based on the Amazon VPC service.
- You can't use transactions in the query editor. For more information about transactions, see BEGIN in the Amazon Redshift Database Developer Guide.
You can save a query up to 3,000 characters long.

Enabling access to the query editor

To access the query editor, you need permission. To enable access, attach the `AmazonRedshiftQueryEditor` and `AmazonRedshiftReadOnlyAccess` AWS-managed policies for IAM permissions to the IAM user that you use to access your cluster. You can use the IAM console (https://console.aws.amazon.com/iam/) to attach IAM policies.

If you have already created an IAM user to access Amazon Redshift, you can attach the `AmazonRedshiftQueryEditor` and `AmazonRedshiftReadOnlyAccess` AWS-managed policies to that user. If you haven't created an IAM user yet, create one and attach the policy to the IAM user.

The AWS-managed policy `AmazonRedshiftQueryEditor` allows the action `redshift:GetClusterCredentials`, which by default gives a database user superuser access to the database. To restrict access, you can do one of the following:

- Create a custom policy that allows calling `redshift:GetClusterCredentials` and restricts the resource to a given value for `DbUser`.
- Add a policy to the user that denies permission to `redshift:GetClusterCredentials` and then requires users of the query editor to sign in with temporary credentials. For example, a denial policy might be similar to the following policy.

```json
{
   "Version": "2012-10-17",
   "Statement": {
      "Effect": "Deny",
      "Action": "redshift:GetClusterCredentials",
      "Resource": "*"
   }
}
```

For more information, see Create an IAM role or user role or user with permissions to call GetClusterCredentials (p. 373).

If you are granted access to Amazon Redshift query editor by attaching the AWS-managed policy `AmazonRedshiftQueryEditor`, then you can list all secrets. However, you can create and retrieve only secrets that are tagged with the key `RedshiftQueryOwner` and the value `$(aws:userid)`. If you create the key from the Amazon Redshift query editor, then the key is automatically tagged. To use a secret that wasn't created with the Amazon Redshift query editor, confirm that the secret is tagged with the key `RedshiftQueryOwner` and a value of your unique IAM user identifier, for example AIDACKCEVSQ6C2EXAMPLE.

To attach the required IAM policies for the query editor

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. Choose Users.
3. Choose the user that needs access to the query editor.
4. Choose Add permissions.
5. Choose Attach existing policies directly.
6. For Policy names, choose `AmazonRedshiftQueryEditor` and `AmazonRedshiftReadOnlyAccess`.
7. Choose Next: Review.
8. Choose Add permissions.
Connecting with the query editor

When you connect to a cluster with the query editor, you use one of the following authentication methods. Each method requires a different combination of input from the Amazon Redshift console.

AWS Secrets Manager

With this method, provide a secret value for secret-arn that is stored in AWS Secrets Manager. This secret contains credentials to connect to your database.

Temporary credentials

With this method, provide your database and db-user values.

Storing database credentials in AWS Secrets Manager

When you call the query editor, you can pass credentials for the cluster by using a secret in AWS Secrets Manager. To pass credentials in this way, you specify the name or the Amazon Resource Name (ARN) of the secret.

For more information about the minimum permissions, see Creating and Managing Secrets with AWS Secrets Manager in the AWS Secrets Manager User Guide.

To store your credentials in a secret for an Amazon Redshift cluster

1. Use AWS Secrets Manager to create a secret that contains credentials for the cluster. When you choose Store a new secret, choose Credentials for Redshift cluster. Store a value for User name (the database user), Password, and DB cluster (cluster identifier) in your secret.

   For instructions, see Creating a Basic Secret in the AWS Secrets Manager User Guide.

2. Use the AWS Secrets Manager console to view the details for the secret you created, or run the aws secretsmanager describe-secret AWS CLI command.

Using the query editor

In the following example, you use the query editor to perform the following tasks:

- Run SQL commands.
- View query execution details.
- Save a query.
- Download a query result set.

To complete the following example, you need an existing Amazon Redshift cluster. If you don't have a cluster, create one by following the procedure described in Creating a cluster (p. 71).

To use the query editor on the Amazon Redshift console

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.

2. On the navigation menu, choose Query editor, then connect to a database in your cluster.

3. For Schema, choose public to create a new table based on that schema.

4. Enter the following in the query editor window and choose Run to create a new table.
create table shoes(
    shoetype varchar (10),
    color varchar(10));

5. Choose **Clear**.
6. Enter the following command in the query editor window, and choose **Run** to add rows to the table.

```
insert into shoes values
    ('loafers', 'brown'),
    ('sandals', 'black');
```

7. Choose **Clear**.
8. Enter the following command in the query editor window and choose **Run** to query the new table.

```
select * from shoes;
```

The **Query results** displays the results.

<table>
<thead>
<tr>
<th>Shoe type</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>sandals</td>
<td>black</td>
</tr>
<tr>
<td>loafers</td>
<td>brown</td>
</tr>
</tbody>
</table>

9. Choose **Execution** to view the run details.
10. Choose **Data**, then **Export** to download the query results as a file.

---

### Scheduling a query on the Amazon Redshift console

To create a schedule to run a SQL statement, you can use the query editor on the Amazon Redshift console. You can create a schedule to run your SQL statement at the time intervals that match your business needs. When it’s time for the scheduled query to run, Amazon EventBridge initiates the query.

**To create a schedule to run a SQL statement on the console**

1. Open the console and query editor as described in [Using the query editor](p. 116).
2. Choose **Schedule** to create a schedule to run an SQL statement.

When you define the schedule, you provide the following information:

- An IAM role that is used to assume the required permissions to run the query. For more information, see [Setting up permissions to schedule a query on the Amazon Redshift console](p. 118).
- The authentication values for either AWS Secrets Manager or temporary credentials to authorize access your cluster. For more information, see [Authenticating a scheduled query](p. 119).
- The name of the scheduled query and a single SQL statement to be run.
- The schedule frequency and repeat options or a cron formatted value.
- Optionally, you can enable Amazon SNS notifications to monitor the scheduled query. If your query is being run but you don’t see messages published in your SNS topic, see [My rule is being triggered but I don’t see any messages published into my Amazon SNS topic](in the Amazon EventBridge User Guide).
You can also manage and update scheduled queries using the Amazon Redshift console. Depending on your version of the console, scheduled queries might be listed in the following places:

- On the **Schedules** tab of the details page of your cluster.
- On the scheduled queries list that you can reach from the navigation pane. To see the list, on the navigation menu, choose **Queries and loads, Schedule query list**.
- On the **Scheduled queries** tab of the query editor.

If you choose **Schedule name** from one of these locations, you can view and edit your scheduled query's definition.

### Setting up permissions to schedule a query on the Amazon Redshift console

To schedule queries, the AWS Identity and Access Management (IAM) user defining the schedule and the IAM role associated with the schedule must be configured as follows.

For the IAM user logged into the Amazon Redshift console, do the following:

- Attach the **AmazonEventBridgeFullAccess** AWS-managed policy.
- Attach a policy with the **sts:AssumeRole** permission of the IAM role that you specify when you define the scheduled SQL statement.

The following example shows a policy that assumes a specified IAM role.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeIAMRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::account-id:role/sql-statement-iam-role"
    }
  ]
}
```

For the IAM role that you specify to enable the scheduler to run a query, do the following:

- Ensure that this IAM role specifies the EventBridge service principal (**events.amazonaws.com**). The following is an example trust relationship.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": ["events.amazonaws.com"
      ],
      "Action": "sts:AssumeRole"
    }
  ]
}
```
For more information about how to create an IAM role for EventBridge events, see Permissions required to use the Amazon EventBridge scheduler (p. 348).

- Attach the AmazonRedshiftDataFullAccess AWS-managed policy to the IAM role.
- To allow IAM users to view schedule history, edit the IAM role to add the sts:AssumeRole permission.

The following is an example of the definition of an IAM role.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": ["events.amazonaws.com"]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Sid": "AssumeRole",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::account-id:user/user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

The following example snippet allows specific IAM users to view schedule history.

```json
{
    "Sid": "AssumeRole",
    "Effect": "Allow",
    "Principal": {
        "AWS": ["arn:aws:iam::account-id:user/testRedshiftUser",
                 "arn:aws:iam::account-id:user/myusername"
        ],
        "Action": "sts:AssumeRole"
    }
}
```

**Authenticating a scheduled query**

When you schedule a query, you use one of the following authentication methods when the query SQL runs. Each method requires a different combination of input from the Amazon Redshift console.

**AWS Secrets Manager**

With this method, provide a secret value for secret-arn that is stored in AWS Secrets Manager. This secret contains credentials to connect to your database. The secret must be tagged with the key RedshiftDataFullAccess.

For more information about the minimum permissions, see Creating and Managing Secrets with AWS Secrets Manager in the AWS Secrets Manager User Guide.
Temporary credentials

With this method, provide your database and db-user values.

The AmazonRedshiftDataFullAccess policy allows the database user named redshift_data_api_user permission for redshift:GetClusterCredentials. If you want to use a different database user to run the SQL statement, then add a policy to the IAM role to allow redshift:GetClusterCredentials. The following example policy allows database users awsuser and myuser.

```json
{
"Version": "2012-10-17",
"Statement": [
  {
    "Sid": "UseTemporaryCredentialsForAllDbUsers",
    "Effect": "Allow",
    "Action": "redshift:GetClusterCredentials",
    "Resource": [
      "arn:aws:redshift:*:*:dbuser:*:awsuser",
      "arn:aws:redshift:*:*:dbuser:*:myuser"
    ]
  }
]
}
```

Create an Amazon EventBridge rule that runs when a query finishes

You can create an event rule to send a notification when a query finishes. For the procedure using the Amazon EventBridge console, see Creating Amazon EventBridge rules that react to events in the Amazon EventBridge User Guide. For more information about event patterns, see Amazon EventBridge event patterns in the Amazon EventBridge User Guide.

For example, the following sample event is sent when a query is FINISHED.

```json
{
"version": "0",
"id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
"detail-type": "Redshift Data Statement Status Change",
"source": "aws.redshift-data",
"account": "123456789012",
"time": "2020-12-22T17:00:00Z",
"region": "us-west-1",
"resources": [
  "arn:aws:redshift:us-east-2:123456789:cluster:t1"
],
"detail": {
  "statementId": "01bdaca2-8967-4e34-ae3f-41d9728d5644",
  "clusterId": "test-dataapi",
  "statementName": "awsome query",
  "state": "FINISHED",
  "pages": 5,
  "expireAt": "2020-12-22T18:43:48Z",
  "principal": "arn:aws:sts::123456789012:assumed-role/any",
  "queryId": 123456
}
}
```

You can create an event pattern rule to filter the event.
Connecting to an Amazon Redshift cluster using SQL client tools

You can connect to Amazon Redshift clusters from SQL client tools over Java Database Connectivity (JDBC), Python, and Open Database Connectivity (ODBC) connections. Amazon Redshift doesn't provide or install any SQL client tools or libraries. To use these tools or libraries to work with data in your clusters, install them on your client computer or Amazon EC2 instance. You can use most SQL client tools that support JDBC, Python, or ODBC drivers.

You can use the following sections to walk through the process of configuring your client computer or Amazon EC2 instance to use a JDBC, Python, or ODBC connection. They also discuss related security options for the client connection to the server. Additionally, following you can find information about setting up and connecting from two example third-party SQL client tools, SQL Workbench/J and psql. You can try these tools if you don't have a business intelligence tool to use yet. You can also use this section to learn about connecting to your cluster programatically. Finally, if you encounter issues when attempting to connect to your cluster, you can review the troubleshooting information in this section to identify possible solutions.

**Topics**
- Configuring connections in Amazon Redshift (p. 121)
- Configuring security options for connections (p. 193)
- Connecting to clusters from client tools and code (p. 198)
- Troubleshooting connection issues in Amazon Redshift (p. 233)

**Configuring connections in Amazon Redshift**

In the following section, you can find how to configure JDBC, Python, and ODBC connections to connect to your cluster from SQL client tools. This section describes how to set up JDBC, Python, and ODBC connections. It also describes how to use Secure Sockets Layer (SSL) and server certificates to encrypt communication between the client and server.

**JDBC, Python, and ODBC drivers for Amazon Redshift**

To work with data in your cluster, you need JDBC, Python, or ODBC drivers for connectivity from your client computer or instance. Code your applications to use JDBC, Python, or ODBC data access API operations, and use SQL client tools that support either JDBC, Python, or ODBC.

Amazon Redshift offers JDBC, Python, and ODBC drivers for download. These drivers are supported by AWS Support. PostgreSQL drivers are not tested and not supported by the Amazon Redshift team.
Use the Amazon Redshift–specific drivers when connecting to an Amazon Redshift cluster. The Amazon Redshift drivers have the following advantages:

- Support for IAM, SSO, and federated authentication.
- Support for new Amazon Redshift data types.
- Support for authentication profiles.
- Improved performance in conjunction with Amazon Redshift enhancements.

For more information about how to download the JDBC and ODBC drivers and configure connections to your cluster, see Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 122), Configuring the Amazon Redshift Python connector (p. 154), and Configuring an ODBC connection (p. 179).

For JDBC driver version 1.2.47.1071 or later and ODBC driver version 1.4.16.1000 or later, Amazon Redshift must be version 1.0.17708 or later. To verify the Amazon Redshift version, see VERSION in the Amazon Redshift Database Developer Guide.

Finding your cluster connection string

To connect to your cluster with your SQL client tool, you need the cluster connection string. You can find the cluster connection string in the Amazon Redshift console, on a cluster's details page.

To find the connection string for a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details.
3. The JDBC URL and ODBC URL connection strings are available, along with additional details, in the General information section. Each string is based on the AWS Region where the cluster runs. Click the icon next to the appropriate connection string to copy it.

Configuring a connection for JDBC driver version 2.1 for Amazon Redshift

You can use a JDBC driver version 2.1 connection to connect to your Amazon Redshift cluster from many third-party SQL client tools. The Amazon Redshift Python connector provides an open source solution. You can browse the source code, request enhancements, report issues, and provide contributions.

To use a JDBC connection, refer to the following sections.

Topics
- Download the Amazon Redshift JDBC driver, version 2.1 (p. 123)
- Installing the Amazon Redshift JDBC driver, version 2.1 (p. 123)
- Getting the JDBC URL (p. 125)
- Building the connection URL (p. 125)
- Configuring TCP keepalives for your JDBC connection (p. 126)
- Configuring your JDBC connection with Apache Maven (p. 126)
- Configuring authentication and SSL (p. 128)
- Configuring logging (p. 131)
- Converting data types (p. 133)
- Using prepared statement support (p. 134)
Download the Amazon Redshift JDBC driver, version 2.1

Amazon Redshift offers drivers for tools that are compatible with the JDBC 4.2 API. The class name for this driver is `com.amazon.redshift.jdbc42.Driver`.

For detailed information about how to install the JDBC driver, reference the JDBC driver libraries, and register the driver class, see the following topics.

For each computer where you use the Amazon Redshift JDBC driver version 2.1, make sure that the Java Runtime Environment (JRE) 8.0 is installed.

If you use the Amazon Redshift JDBC driver for database authentication, make sure that you have AWS SDK for Java 1.11.118 or later in your Java class path. If you don't have AWS SDK for Java installed, download the ZIP file with JDBC 4.2–compatible driver and driver dependent libraries for the AWS SDK:

- **JDBC 4.2–compatible driver version 2.1 and AWS SDK driver–dependent libraries.**

  This ZIP file contains the JDBC 4.2–compatible driver version 2.1 and AWS SDK for Java 1.x driver–dependent library files. Unzip the dependent jar files to the same location as the JDBC driver. Only the JDBC driver needs to be in CLASSPATH.

  This ZIP file doesn't include the complete AWS SDK for Java 1.x. However, it includes the AWS SDK for Java 1.x driver–dependent libraries that are required for AWS Identity and Access Management (IAM) database authentication.

  Use this Amazon Redshift JDBC driver with the AWS SDK that is required for IAM database authentication.

  To install the complete AWS SDK for Java 1.x, see AWS SDK for Java 1.x in the AWS SDK for Java Developer Guide.

  - **JDBC 4.2–compatible driver version 2.1 (without the AWS SDK).**

Review the JDBC driver version 2.1 software license and change log file:

- **JDBC driver version 2.1 license**
- **JDBC driver version 2.1 change log**

JDBC drivers version 1.2.27.1051 and later support Amazon Redshift stored procedures. For more information, see Creating stored procedures in Amazon Redshift in the Amazon Redshift Database Developer Guide.

If your tool requires a specific previous version of a JDBC driver version 2.1, see Previous versions of JDBC driver version 2.1 (p. 153).

Installing the Amazon Redshift JDBC driver, version 2.1

To install the Amazon Redshift JDBC 4.2–compatible driver version 2.1 and driver–dependent libraries for AWS SDK, extract the files from the ZIP archive to the directory of your choice.

To install the Amazon Redshift JDBC 4.2–compatible driver version 2.1 (without the AWS SDK), copy the JAR file to the directory of your choice.
To access an Amazon Redshift data store using the Amazon Redshift JDBC driver, you need to perform configuration as described following.

**Topics**
- Referencing the JDBC driver libraries (p. 124)
- Registering the driver class (p. 124)

**Referencing the JDBC driver libraries**

The JDBC application or Java code that you use to connect to your data must access the driver JAR files. In the application or code, specify all the JAR files that you extracted from the ZIP archive.

**Using the driver in a JDBC application**

JDBC applications usually provide a set of configuration options for adding a list of driver library files. Use the provided options to include all the JAR files from the ZIP archive as part of the driver configuration in the application. For more information, see the documentation for your JDBC application.

**Using the driver in Java code**

You must include all the driver library files in the class path. This is the path that the Java Runtime Environment searches for classes and other resource files. For more information, see the appropriate Java SE documentation to set the class path for your operating system.

- Windows: https://docs.oracle.com/javase/7/docs/technotes/tools/windows/classpath.html
- Linux and Solaris: https://docs.oracle.com/javase/7/docs/technotes/tools/solaris/classpath.html

**Registering the driver class**

Make sure that you register the appropriate class for your application. You use following classes to connect the Amazon Redshift JDBC driver to Amazon Redshift data stores:

- **Driver classes** extend java.sql.Driver.
- **DataSource classes** extend javax.sql.DataSource and javax.sql.ConnectionPoolDataSource.

The driver supports the following fully qualified class names that are independent of the JDBC version:

- com.amazon.redshift.jdbc.Driver
- com.amazon.redshift.jdbc.DataSource

The following example shows how to use the DriverManager class to establish a connection for JDBC 4.2.

```java
private static Connection connectViaDM() throws Exception {
    Connection connection = null;
    connection = DriverManager.getConnection(CONNECTION_URL);
    return connection;
}
```

The following example shows how to use the DataSource class to establish a connection.

```java
```
private static Connection connectViaDS() throws Exception {
    Connection connection = null;
    Amazon Redshift JDBC Driver Installation and Configuration Guide
DataSource ds = new com.amazon.redshift.jdbc.DataSource();
ds.setURL(CONNECTION_URL);
connection = ds.getConnection();
return connection;
}

Getting the JDBC URL

Before you can connect to your Amazon Redshift cluster from a SQL client tool, you need to know the JDBC URL of your cluster. The JDBC URL has the following format: jdbc:redshift://endpoint:port/database.

The fields of the preceding format have the following values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>jdbc</td>
<td>The protocol for the connection.</td>
</tr>
<tr>
<td>redshift</td>
<td>The subprotocol that specifies to use the Amazon Redshift driver to connect to the database.</td>
</tr>
<tr>
<td>endpoint</td>
<td>The endpoint of the Amazon Redshift cluster.</td>
</tr>
<tr>
<td>port</td>
<td>The port number that you specified when you launched the cluster. If you have a firewall, make sure that this port is open for you to use.</td>
</tr>
<tr>
<td>database</td>
<td>The database that you created for your cluster.</td>
</tr>
</tbody>
</table>

The following is an example JDBC URL: jdbc:redshift://examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com:5439/dev

Be sure to enter the URL values, for example SessionToken values, in URL encoded format.

For information about how to get your JDBC connection, see Finding your cluster connection string (p. 122).

If the client computer fails to connect to the database, you can troubleshoot possible issues. For more information, see Troubleshooting connection issues in Amazon Redshift (p. 233).

Building the connection URL

Use the connection URL to supply connection information to the data store that you are accessing. The following is the format of the connection URL for the Amazon Redshift JDBC driver version 2.1. Here, [Host] the endpoint of the Amazon Redshift server and [Port] is the number of the Transmission Control Protocol (TCP) port that the server uses to listen for client requests.

jdbc:redshift://[Host]:[Port]

By default, Amazon Redshift uses port 5439.

The following is the format of a connection URL that specifies some optional settings.
jdbc:redshift://[Host]:[Port]/[Schema];[Property1]=[Value];
[Property2]=[Value];

For example, suppose that you want to connect to port 9000 on an Amazon Redshift cluster in the US
West (N. California) Region on AWS. You also want to access the schema named Default and authenticate
the connection using a user name and password. In this case, you use the following connection URL.

jdbc:redshift://redshift.company.us-west-
1.redshift.amazonaws.com:9000/Default;UID=amazon;PWD=amazon

You can use the following characters to separate configuration options in the URL string:
• ;
• ?
• &

For example, the following URL strings are equivalent:

jdbc:redshift://my_host:5439/dev;ssl=false;defaultRowFetchSize=100

jdbc:redshift://my_host:5439/dev?ssl=false?defaultRowFetchSize=100

The following URL example specifies a log level of 6 and the path for the logs.

jdbc:redshift://redshift.amazonaws.com:5439/dev;DSILogLevel=6;LogPath=/home/user/logs;

Don't duplicate properties in the connection URL.

For a complete list of the configuration options that you can specify, see Options for JDBC driver version
2.1 configuration (p. 136).

**Configuring TCP keepalives for your JDBC connection**

By default, the Amazon Redshift JDBC driver is configured to use TCP keepalives to prevent connections
from timing out. You can specify when the driver starts sending keepalive packets or turn off the feature
by setting the relevant properties in the connection URL. For more information about the syntax of the
connection URL, see Building the connection URL (p. 125).

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCPKeepAliveMinutes</td>
<td>To specify when the driver sends keepalive packets, set the number of minutes of inactivity before the driver starts sending TCP keepalive packets.</td>
</tr>
<tr>
<td>TCPKeepAlive</td>
<td>To turn off TCP keepalives, set this property to FALSE.</td>
</tr>
</tbody>
</table>

**Configuring your JDBC connection with Apache Maven**

Apache Maven is a software project management and comprehension tool. The AWS SDK for Java
supports Apache Maven projects. For more information, see Using the SDK with Apache Maven in the
AWS SDK for Java Developer Guide.
If you use Apache Maven, you can configure and build your projects to use an Amazon Redshift JDBC driver to connect to your Amazon Redshift cluster. To do this, add the JDBC driver as a dependency in your project’s pom.xml file. If you use Maven to build your project and want to use a JDBC connection, take the steps in the following section.

**Configuring the JDBC driver as a Maven dependency**

**To configure the JDBC driver as a Maven dependency**

1. Add either the Amazon repository or the Maven Central repository to the repositories section of your pom.xml file.
   
   **Note**
   The URL in the following code example returns an error if used in a browser. Use this URL only in the context of a Maven project.

   For an Amazon Maven repository, use the following.

   ```xml
   <repositories>
   <repository>
       <id>redshift</id>
       <url>http://redshift-maven-repository.s3-website-us-east-1.amazonaws.com/release</url>
   </repository>
   </repositories>
   
   To connect using Secure Sockets Layer (SSL), add the following repository to your pom.xml file.

   ```xml
   <repositories>
   <repository>
       <id>redshift</id>
       <url>https://s3.amazonaws.com/redshift-maven-repository/release</url>
   </repository>
   </repositories>
   
   For a Maven Central repository, add the following to your pom.xml file.

   ```xml
   <repositories>
   <repository>
       <id>redshift</id>
       <url>https://repo1.maven.org/maven2</url>
   </repository>
   </repositories>
   
2. Declare the version of the driver that you want to use in the dependencies section of your pom.xml file.

   Amazon Redshift offers drivers for tools that are compatible with the JDBC 4.2 API. For information about the functionality supported by these drivers, see Download the Amazon Redshift JDBC driver (p. 170).

   Add a dependency for the driver as shown following.

   Replace `driver-version` in the following example with your driver version, for example 2.1.0.1.

   For a JDBC 4.2–compatible driver, use the following.

   ```xml
   <dependency>
       <groupId>com.amazon.redshift</groupId>
       <artifactId>redshift-jdbc42</artifactId>
   </dependency>
   ```
The class name for this driver is `com.amazon.redshift.jdbc42.Driver`.

The Amazon Redshift Maven drivers need the following optional dependencies when you use IAM database authentication.

```xml
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-core</artifactId>
  <version>1.12.23</version>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-redshift</artifactId>
  <version>1.12.23</version>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-sts</artifactId>
  <version>1.12.23</version>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
```

If your tool requires a specific previous version of a driver, see Use previous JDBC driver versions with Maven (p. 178).

Upgrading the driver to the latest version

To upgrade or change the Amazon Redshift JDBC driver to the latest version, first modify the version section of the dependency to the latest version of the driver. Then clean your project with the Maven Clean Plugin, as shown following.

```bash
mvn clean
```

Configuring authentication and SSL

To protect data from unauthorized access, Amazon Redshift data stores require all connections to be authenticated using user credentials. Some data stores also require connections to be made over the Secure Sockets Layer (SSL) protocol, either with or without one-way authentication.

The Amazon Redshift JDBC driver version 2.1 provides full support for these authentication protocols.

The SSL version that the driver supports depends on the JVM version that you are using. For information about the SSL versions that are supported by each version of Java, see Diagnosing TLS, SSL, and HTTPS on the Java Platform Group Product Management Blog.

The SSL version used for the connection is the highest version that is supported by both the driver and the server, which is determined at connection time.

Configure the Amazon Redshift JDBC driver version 2.1 to authenticate your connection according to the security requirements of the Amazon Redshift server that you are connecting to.
You must always provide your Amazon Redshift user name and password to authenticate the connection. Depending on whether SSL is enabled and required on the server, you might also need to configure the driver to connect through SSL. Or you might use one-way SSL authentication so that the client (the driver itself) verifies the identity of the server.

You provide the configuration information to the driver in the connection URL. For more information about the syntax of the connection URL, see Building the connection URL (p. 125).

SSL indicates TLS/SSL, both Transport Layer Security and Secure Sockets Layer. The driver supports industry-standard versions of TLS/SSL.

**Using user name and password only**

If the server you are connecting to doesn't use SSL, then you only need to provide your user name and password to authenticate the connection.

**To configure authentication using your user name and password only**

1. Set the **UID** property to your user name for accessing the Amazon Redshift server.
2. Set the **PWD** property to the password corresponding to your user name.

**Using SSL without identity verification**

If the server you are connecting to uses SSL but doesn't require identity verification, then you can configure the driver to use a non-validating SSL factory.

**To configure an SSL connection without identity verification**

1. Set the **UID** property to your user name for accessing the Amazon Redshift server.
2. Set the **PWD** property to the password corresponding to your user name.

**Using one-way SSL authentication**

If the server you are connecting to uses SSL and has a certificate, then you can configure the driver to verify the identity of the server using one-way authentication.

One-way authentication requires a signed, trusted SSL certificate for verifying the identity of the server. You can configure the driver to use a specific certificate or access a TrustStore that contains the appropriate certificate. If you don't specify a certificate or TrustStore, then the driver uses the default Java TrustStore (typically either `jssecacerts` or `cacerts`).

**To configure one-way SSL authentication**

1. Set the **UID** property to your user name for accessing the Amazon Redshift server.
2. Set the **PWD** property to the password corresponding to your user name.
3. Set the **SSL** property to true.
4. Set the **SSLRootCert** property to the location of your root CA certificate.
5. If you aren't using one of the default Java TrustStores, then do one of the following:
   a. To specify a server certificate, set the **SSLRootCert** property to the full path of the certificate.
   b. To specify a TrustStore, do the following:
      a. Use the keytool program to add the server certificate to the TrustStore that you want to use.
      b. Specify the TrustStore and password to use when starting the Java application using the driver. For example:
6. Choose one:
   • To validate the certificate, set the SSLMode property to verify-ca.
   • To validate the certificate and verify the host name in the certificate, set the SSLMode property to verify-full.

**Configuring IAM authentication**

If you are connecting to a Amazon Redshift server using IAM authentication, set the following properties as part of your data source connection string.

For more information on IAM authentication, see *Identity and access management in Amazon Redshift* (p. 334).

To use IAM authentication, use one of the following connection string formats:

<table>
<thead>
<tr>
<th>Connection string</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jdbc:redshift:iam:// [host]:[port]/ [db]</td>
<td>A regular connection string. The driver infers the ClusterID and Region from the host.</td>
</tr>
<tr>
<td>jdbc:redshift:iam:// [cluster-id]: [region]/[db]</td>
<td>The driver retrieves host information, given the ClusterID and Region.</td>
</tr>
<tr>
<td>jdbc:redshift:iam:// [host]/[db]</td>
<td>The driver defaults to port 5439, and infers ClusterID and Region from the host.</td>
</tr>
</tbody>
</table>

**Specifying profiles**

If you are using IAM authentication, you can specify any additional required or optional connection properties under a profile name. By doing this, you can avoid putting certain information directly in the connection string. You specify the profile name in your connection string using the Profile property.

Profiles can be added to the AWS credentials file. The default location for this file is: `~/.aws/credentials`.

You can change the default value by setting the path in the following environment variable:

`AWS_CREDENTIAL_PROFILES_FILE`.

For more information about profiles, see *Working with AWS Credentials* in the *AWS SDK for Java*.

**Using instance profile credentials**

If you are running an application on an Amazon EC2 instance that is associated with an IAM role, you can connect using the instance profile credentials.

To do this, use one of the IAM connection string formats in the preceding table, and set the dbuser connection property to the Amazon Redshift user name that you are connecting as.

For more information about instance profiles, see *Access Management* in the *IAM User Guide*.

**Using credential providers**

The driver also supports credential provider plugins from the following services:
• Active Directory Federation Service (ADFS)
• JSON Web Tokens (JWT) Service
• Microsoft Azure Active Directory (AD) Service and Browser Microsoft Azure Active Directory (AD) Service
• Okta Service
• PingFederate Service
• Browser SAML for SAML services such as Okta, Ping, or ADFS

If you use one of these services, the connection URL needs to specify the following properties:

• **Plugin_Name** – The fully-qualified class path for your credentials provider plugin class.
• **IdP_Host** – The host for the service that you are using to authenticate into Amazon Redshift.
• **IdP_Port** – The port that the host for the authentication service listens at. Not required for Okta.
• **User** – The user name for the idp_host server.
• **Password** – The password associated with the idp_host user name.
• **DbUser** – The Amazon Redshift user name you are connecting as.
• **SSL_Insecure** – Indicates whether the IDP server certificate should be verified.
• **Client_ID** – The client ID associated with the user name in the Azure AD portal. Only used for Azure AD.
• **Client_Secret** – The client secret associated with the client ID in the Azure AD portal. Only used for Azure AD.
• **IdP_Tenant** – The Azure AD tenant ID for your Amazon Redshift application. Only used for Azure AD.
• **App_ID** – The Okta app ID for your Amazon Redshift application. Only used for Okta.
• **App_Name** – The optional Okta app name for your Amazon Redshift application. Only used for Okta.
• **Partner_SPID** – The optional partner SPID (service provider ID) value. Only used for PingFederate.

If you are using a browser plugin for one of these services, the connection URL can also include:

• **Login_URL** – The URL for the resource on the identity provider’s website when using the Security Assertion Markup Language (SAML) or Azure AD services through a browser plugin. This parameter is required if you are using a browser plugin.

• **Listen_Port** – The port that the driver uses to get the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.

• **IdP_Response_Timeout** – The amount of time, in seconds, that the driver waits for the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.

For information on additional connection string properties, see Options for JDBC driver version 2.1 configuration (p. 136).

**Configuring logging**

You can turn on logging in the driver to assist in diagnosing issues.

You can log driver information by using the following methods:

• To save logged information in .log files, see Using log files (p. 132).
• To send logged information to the LogStream or LogWriter specified in the DriverManager, see Using LogStream or LogWriter (p. 132).
You provide the configuration information to the driver in the connection URL. For more information about the syntax of the connection URL, see Building the connection URL (p. 125).

Using log files

Only turn on logging long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.

Set the LogLevel key in your connection URL to turn on logging and specify the amount of detail included in log files. The following table lists the logging levels provided by the Amazon Redshift JDBC driver version 2.1, in order from least verbose to most verbose.

<table>
<thead>
<tr>
<th>LogLevel value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Log severe error events that will lead the driver to abort.</td>
</tr>
<tr>
<td>2</td>
<td>Log error events that might allow the driver to continue running.</td>
</tr>
<tr>
<td>3</td>
<td>Log events that might result in an error if action is not taken.</td>
</tr>
<tr>
<td>4</td>
<td>Log general information that describes the progress of the driver.</td>
</tr>
<tr>
<td>5</td>
<td>Log detailed information that is useful for debugging the driver.</td>
</tr>
<tr>
<td>6</td>
<td>Log all driver activity.</td>
</tr>
</tbody>
</table>

To set up logging that uses log files

1. Set the LogLevel property to the desired level of information to include in log files.
2. Set the LogPath property to the full path to the folder where you want to save log files.

   For example, the following connection URL enables logging level 3 and saves the log files in the C:\temp folder: 
   `jdbc:redshift://redshift.company.us-west-1.redshift.amazonaws.com:9000/Default;DSILogLevel=3; LogPath=C:\temp`

3. To make sure that the new settings take effect, restart your JDBC application and reconnect to the server.

   The Amazon Redshift JDBC driver produces the following log files in the location specified in the LogPath property:
   - `redshift_jdbc.log` file that logs driver activity that is not specific to a connection.
   - `redshift_jdbc_connection_[Number].log` file for each connection made to the database, where `[Number]` is a number that identifies each log file. This file logs driver activity that is specific to the connection.

   If the LogPath value is invalid, then the driver sends the logged information to the standard output stream (`System.out`)

Using LogStream or LogWriter

Only turn on logging long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.
Set the LogLevel key in your connection URL to turn on logging and specify the amount of detail sent to the LogStream or LogWriter specified in the DriverManager.

To turn on logging that uses the LogStream or LogWriter:

1. To configure the driver to log general information that describes the progress of the driver, set the LogLevel property to 1 or INFO.
2. To make sure that the new settings take effect, restart your JDBC application and reconnect to the server.

To turn off logging that uses the LogStream or LogWriter:

1. Remove the LogLevel property from the connection URL.
2. To make sure that the new settings take effect, restart your JDBC application and reconnect to the server.

Converting data types

The Amazon Redshift JDBC driver version 2.1 supports many common data formats, converting between Amazon Redshift, SQL, and Java data types.

The following table lists the supported data type mappings.

<table>
<thead>
<tr>
<th>Amazon Redshift type</th>
<th>SQL type</th>
<th>Java type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
<td>Long</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
<td>Boolean</td>
</tr>
<tr>
<td>CHAR</td>
<td>SQL_CHAR</td>
<td>String</td>
</tr>
<tr>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
<td>java.sql.Date</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>SQL_NUMERIC</td>
<td>BigDecimal</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>SQL_DOUBLE</td>
<td>Double</td>
</tr>
<tr>
<td>GEOMETRY</td>
<td>SQL_LONGVARBINARY</td>
<td>byte[]</td>
</tr>
<tr>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
<td>Integer</td>
</tr>
<tr>
<td>OID</td>
<td>SQL_BIGINT</td>
<td>Long</td>
</tr>
<tr>
<td>SUPER</td>
<td>SQL_LONGVARCHAR</td>
<td>String</td>
</tr>
<tr>
<td>REAL</td>
<td>SQL_REAL</td>
<td>Float</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SQL_SMALLINT</td>
<td>Short</td>
</tr>
<tr>
<td>TEXT</td>
<td>SQL_VARCHAR</td>
<td>String</td>
</tr>
<tr>
<td>TIME</td>
<td>SQL_TYPE_TIME</td>
<td>java.sql.Time</td>
</tr>
<tr>
<td>TIMETZ</td>
<td>SQL_TYPE_TIME</td>
<td>java.sql.Time</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
<td>java.sql.Timestamp</td>
</tr>
<tr>
<td>TIMESTAMPTZ</td>
<td>SQL_TYPE_TIMESTAMP</td>
<td>java.sql.Timestamp</td>
</tr>
</tbody>
</table>
### Using prepared statement support

The Amazon Redshift JDBC driver supports prepared statements. You can use prepared statements to improve the performance of parameterized queries that need to be run multiple times during the same connection.

A prepared statement is a SQL statement that is compiled on the server side but not run immediately. The compiled statement is stored on the server as a PreparedStatement object until you close the object or the connection. While that object exists, you can run the prepared statement as many times as needed using different parameter values, without having to compile the statement again. This reduced overhead enables the set of queries to be run more quickly.

For more information about prepared statements, see "Using Prepared Statements" in JDBC Basics tutorial from Oracle.

You can prepare a statement that contains multiple queries. For example, the following prepared statement contains two INSERT queries:

```java
PreparedStatement pstmt = conn.prepareStatement("INSERT INTO MyTable VALUES (1, 'abc'); INSERT INTO CompanyTable VALUES (1, 'abc');");
```

Take care that these queries don't depend on the results of other queries that are specified within the same prepared statement. Because queries don't run during the prepare step, the results have not been returned yet, and aren't available to other queries in the same prepared statement.

For example, the following prepared statement, which creates a table and then inserts values into that newly-created table, is not allowed:

```java
PreparedStatement pstmt = conn.prepareStatement("CREATE TABLE MyTable(col1 int, col2 varchar); INSERT INTO myTable VALUES (1, 'abc');");
```

If you try to prepare this statement, the server returns an error stating that the destination table (myTable) doesn't exist yet. The CREATE query must be run before the INSERT query can be prepared.

### Differences between the 2.1 and 1.x versions of the JDBC driver

This section describes the differences in the information returned by the 2.1 and 1.x versions of the JDBC driver. The JDBC driver version 1.x is discontinued.

The following table lists the DatabaseMetadata information returned by the getDatabaseProductName() and getDatabaseProductVersion() functions for each version of the JDBC driver. JDBC driver version 2.1 obtains the values while establishing the connection. JDBC driver version 1.x obtains the values as a result of a query.

<table>
<thead>
<tr>
<th>JDBC driver version</th>
<th>getDatabaseProductName() result</th>
<th>getDatabaseProductVersion() result</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Redshift</td>
<td>8.0.2</td>
</tr>
<tr>
<td>1.x</td>
<td>PostgreSQL</td>
<td>08.00.0002</td>
</tr>
</tbody>
</table>
The following table lists the DatabaseMetadata information returned by the getTypeInfo function for each version of the JDBC driver.

<table>
<thead>
<tr>
<th>JDBC driver version</th>
<th>getTypeInfo result</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Consistent with Redshift datatypes</td>
</tr>
<tr>
<td>1.x</td>
<td>Consistent with PostgreSQL datatypes</td>
</tr>
</tbody>
</table>

Creating initialization (.ini) files for JDBC driver version 2.1

By using initialization (.ini) files for Amazon Redshift JDBC driver version 2.1, you can specify system level configuration parameters. For example, federated IdP authentication parameters can vary for each application. The .ini file provides a common location for SQL clients to get the required configuration parameters.

You can create an JDBC driver version 2.1 initialization (.ini) file that contains configuration options for SQL clients. The default name of the file is rsjdbc.ini. The JDBC driver version 2.1 checks for the .ini file in the following locations, listed in order of precedence:

- IniFile parameter in the connection URL or in the connection property dialog box of the SQL client. Be sure that the IniFile parameter contains the full path to the .ini file, including the file name. For information about the IniFile parameter, see IniFile (p. 144). If the IniFile parameter incorrectly specifies the location of the .ini file, an error displays.
- Environment variables such as AMAZON_REDSHIFT_JDBC_INI_FILE with the full path, including the file name. You can use rsjdbc.ini or specify a file name. If the AMAZON_REDSHIFT_JDBC_INI_FILE environment variable incorrectly specifies the location of the .ini file, an error displays.
- Directory where the driver JAR file is located.
- User home directory.
- Temp directory of the system.

You can organize the .ini file into sections, for example [DRIVER]. Each section contains key-value pairs that specify various connection parameters. You can use the IniSection parameter to specify a section in the .ini file. For information about the IniSection parameter, see IniSection (p. 145).

Following is an example of the .ini file format, with sections for [DRIVER], [DEV], [QA], and [PROD]. The [DRIVER] section can apply to any connection.

```
[DRIVER]
key1=val1
key2=val2

[DEV]
key1=val1
key2=val2

[QA]
key1=val1
key2=val2

[PROD]
key1=val1
key2=val2
```

The JDBC driver version 2.1 loads configuration parameters from the following locations, listed in order of precedence:
• Default configuration parameters in the application code.
• [DRIVER] section properties from the .ini file, if included.
• Custom section configuration parameters, if the IniSection option is provided in the connection URL or in the connection property dialog box of the SQL client.
• Properties from the connection property object specified in the getConnection call.
• Configuration parameters specified in the connection URL.

Options for JDBC driver version 2.1 configuration

Following, you can find descriptions for the options that you can specify for version 2.1 of the Amazon Redshift JDBC driver.

You can set configuration properties using the connection URL. For more information, see Building the connection URL (p. 125).

Topics
• AccessKeyID (p. 137)
• AllowDBUserOverride (p. 138)
• App_ID (p. 138)
• App_Name (p. 138)
• ApplicationName (p. 138)
• AuthProfile (p. 139)
• AutoCreate (p. 139)
• Client_ID (p. 139)
• Client_Secret (p. 139)
• ClusterID (p. 139)
• connectTimeout (p. 140)
• databaseMetadataCurrentDbOnly (p. 140)
• DbUser (p. 140)
• DbGroups (p. 140)
• DBNAME (p. 140)
• defaultRowFetchSize (p. 141)
• DisableisValidQuery (p. 141)
• enableFetchReadAndProcessBuffers (p. 141)
• enableFetchRingBuffer (p. 142)
• enableMultiSqlSupport (p. 142)
• fetchRingBufferSize (p. 142)
• ForceLowercase (p. 142)
• groupFederation (p. 143)
• HOST (p. 143)
• IAMDisableCache (p. 143)
• IAMDuration (p. 143)
• IdP_Host (p. 144)
• IdP_Port (p. 144)
• IdP_Tenant (p. 144)
• IdP_Response_Timeout (p. 144)
AccessKeyId

- **Default Value** – None
- **Data Type** – String
You can specify this parameter to enter the IAM access key for the IAM user or role. You can usually locate the key by looking at an existing string or user profile. If you specify this parameter, you must also specify the SecretAccessKey parameter.

This parameter is optional.

**AllowDBUserOverride**

- **Default Value** – 0
- **Data Type** – String

This option specifies whether the driver uses the `DbUser` value from the SAML assertion or the value that is specified in the `DbUser` connection property in the connection URL.

This parameter is optional.

1

The driver uses the `DbUser` value from the SAML assertion.

If the SAML assertion doesn't specify a value for `DbUser`, the driver uses the value specified in the `DbUser` connection property. If the connection property also doesn't specify a value, the driver uses the value specified in the connection profile.

0

The driver uses the `DbUser` value specified in the `DbUser` connection property.

If the `DbUser` connection property doesn't specify a value, the driver uses the value specified in the connection profile. If the connection profile also doesn't specify a value, the driver uses the value from the SAML assertion.

**App_ID**

- **Default Value** – None
- **Data Type** – String

The Okta-provided unique ID associated with your Amazon Redshift application.

This parameter is required if authenticating through the Okta service.

**App_Name**

- **Default Value** – None
- **Data Type** – String

The name of the Okta application that you use to authenticate the connection to Amazon Redshift.

This parameter is optional.

**ApplicationName**

- **Default Value** – null
- **Data Type** – String

The name of the application to pass to Amazon Redshift for audit purposes.
This parameter is optional.

**AuthProfile**

- **Default Value** – None
- **Data Type** – String

The name of the authentication profile to use for connecting to Amazon Redshift.

This parameter is optional.

**AutoCreate**

- **Default Value** – false
- **Data Type** – Boolean

This option specifies whether the driver causes a new user to be created when the specified user doesn't exist.

This parameter is optional.

**true**

If the user specified by either `DBUser` or unique ID (UID) doesn't exist, a new user with that name is created.

**false**

The driver doesn't cause new users to be created. If the specified user doesn't exist, the authentication fails.

**Client_ID**

- **Default Value** – None
- **Data Type** – String

The client ID to use when authenticating the connection using the Azure AD service.

This parameter is required if authenticating through the Azure AD service.

**Client_Secret**

- **Default Value** – None
- **Data Type** – String

The Client Secret to use when authenticating the connection using the Azure AD service.

This parameter is required if authenticating through the Azure AD service.

**ClusterID**

- **Default Value** – None
- **Data Type** – String

The name of the Amazon Redshift cluster that you want to connect to.
This parameter is optional.

**connectTimeout**

- **Default Value** – 10
- **Data Type** – Integer

The timeout value to use for socket connect operations. If the time required to establish an Amazon Redshift connection exceeds this value, the connection is considered unavailable. The timeout is specified in seconds. A value of 0 means that no timeout is specified.

This parameter is optional.

**databaseMetadataCurrentDbOnly**

- **Default Value** – true
- **Data Type** – Boolean

This option specifies whether the metadata API retrieves data from all accessible databases or only from the connected database.

This parameter is optional.

You can specify the following values:

**true**

The application retrieves metadata from a single database.

**false**

The application retrieves metadata from all accessible databases.

**DbUser**

- **Default Value** – None
- **Data Type** – String

The user ID to use with your Amazon Redshift account. You can use an ID that doesn't currently exist if you have enabled the AutoCreate property.

This parameter is optional.

**DbGroups**

- **Default Value** – PUBLIC
- **Data Type** – String

A comma-separated list of existing database group names that **DbUser** joins for the current session.

This parameter is optional.

**DBNAME**

- **Default Value** – null
• **Data Type** – String

The name of the database to connect to. You can use this option to specify the database name in the JDBC connection URL.

This parameter is required. You must specify the database name, either in the connection URL or in the connection properties of the client application.

**defaultRowFetchSize**

• **Default Value** – 0
• **Data Type** – Integer

This option specifies a default value for getFetchSize.

This parameter is optional.

You can specify the following values:

0

Fetch all rows in a single operation.

**Positive integer**

Number of rows to fetch from the database for each fetch iteration of the ResultSet.

**DisableIsValidQuery**

• **Default Value** – False
• **Data Type** – Boolean

This option specifies whether the driver submits a new database query when using the Connection.isValid() method to determine whether the database connection is active.

This parameter is optional.

**true**

The driver doesn't submit a query when using Connection.isValid() to determine whether the database connection is active. This may cause the driver to incorrectly identify the database connection as active if the database server has shut down unexpectedly.

**false**

The driver submits a query when using Connection.isValid() to determine whether the database connection is active.

**enableFetchReadAndProcessBuffers**

• **Default Value** – true
• **Data Type** – Boolean

This option enables Amazon Redshift to fetch rows using a ring buffer and a processing buffer on separate threads to improve performance.

This parameter is optional.
enableFetchRingBuffer

- **Default Value** – true
- **Data Type** – Boolean

This option specifies that the driver fetches rows using a ring buffer on a separate thread. The fetchRingBufferSize parameter specifies the ring buffer size.

This parameter is optional.

enableMultiSqlSupport

- **Default Value** – true
- **Data Type** – Boolean

This option specifies whether to process multiple SQL commands separated by semicolons in a Statement.

This parameter is optional.

You can specify the following values:

- **true**
  - The driver processes multiple SQL commands, separated by semicolons, in a Statement object.

- **false**
  - The driver returns an error for multiple SQL commands in a single Statement.

fetchRingBufferSize

- **Default Value** – 1G
- **Data Type** – String

This option specifies the size of the ring buffer used while fetching the result set. You can specify a size in bytes, for example 1K for 1 KB, 5000 for 5,000 bytes, 1M for 1 MB, 1G for 1 GB, and so on. You can also specify a percentage of heap memory. The driver stops fetching rows upon reaching the limit. Fetching resumes when the application reads rows and frees space in the ring buffer.

This parameter is optional.

ForceLowercase

- **Default Value** – false
- **Data Type** – Boolean

This option specifies whether the driver lowercases all database groups (DbGroups) sent from the identity provider to Amazon Redshift when using SSO authentication.

This parameter is optional.

- **true**
  - The driver lowercases all database groups that are sent from the identity provider.
false

The driver doesn't alter database groups.

groupFederation

- **Default Value** – false
- **Data Type** – Boolean

This option specifies whether to use Amazon Redshift IDP groups. This is supported by the GetClusterCredentialsV2 API.
This parameter is optional.

true

Use Amazon Redshift Identity Provider (IDP) groups.
false

Use STS API and GetClusterCredentials for user federation and explicitly specify DbGroups for the connection.

HOST

- **Default Value** – null
- **Data Type** – String

The host name of the Amazon Redshift server to connect to. You can use this option to specify the host name in the JDBC connection URL.

This parameter is required. You must specify the host name, either in the connection URL or in the connection properties of the client application.

IAMDisableCache

- **Default Value** – false
- **Data Type** – Boolean

This option specifies whether the IAM credentials are cached.
This parameter is optional.

true

The IAM credentials aren't cached.
false

The IAM credentials are cached. This improves performance when requests to the API gateway are throttled, for instance.

IAMDuration

- **Default Value** – 900
- **Data Type** – Integer
The length of time, in seconds, until the temporary IAM credentials expire.

- **Minimum value** – 900
- **Maximum value** – 3,600

This parameter is optional.

**IdP_Host**

- **Default Value** – None
- **Data Type** – String

The IdP (identity provider) host you are using to authenticate into Amazon Redshift. This can be specified in either the connection string or in a profile.

This parameter is optional.

**IdP_Port**

- **Default Value** – None
- **Data Type** – String

The port used by an IdP (identity provider). You can specify the port in either the connection string or in a profile. The default port is 5439.

This parameter is optional.

**IdP_Tenant**

- **Default Value** – None
- **Data Type** – String

The Azure AD tenant ID for your Amazon Redshift application.

This parameter is required if authenticating through the Azure AD service.

**IdP_Response_Timeout**

- **Default Value** – 120
- **Data Type** – Integer

The amount of time, in seconds, that the driver waits for the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.

This parameter is optional.

**IniFile**

- **Default Value** – None
- **Data Type** – String

The full path of the .ini file, including file name. For example:

```
IniFile="C:\tools\rsjdbc.ini"
```
For information about the .ini file, see Creating initialization (.ini) files for JDBC driver version 2.1 (p. 135).

This parameter is optional.

**IniSection**

- **Default Value** – None
- **Data Type** – String

The name of a section in the .ini file containing the configuration options. For information about the .ini file, see Creating initialization (.ini) files for JDBC driver version 2.1 (p. 135).

The following example specifies the [Prod] section of the .ini file:

```
IniSection="Prod"
```

This parameter is optional.

**Login_URL**

- **Default Value** – None
- **Data Type** – String

The URL for the resource on the identity provider's website when using the SAML or Azure AD services through a browser plugin.

This parameter is required if authenticating with the SAML or Azure AD services through a browser plugin.

**loginTimeout**

- **Default Value** – 0
- **Data Type** – Integer

The number of seconds to wait before timing out when connecting and authenticating to the server. If establishing the connection takes longer than this threshold, then the connection is aborted.

When this property is set to 0, connections don't time out.

This parameter is optional.

**loginToRp**

- **Default Value** – `urn:amazon:webservices`
- **Data Type** – String

The relying party trust that you want to use for the AD FS authentication type.

This parameter is optional.

**LogLevel**

- **Default Value** – 0
- **Data Type** – Integer
Use this property to turn on or turn off logging in the driver and to specify the amount of detail included in log files.

Enable logging only long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.

This parameter is optional.

Set the parameter to one of the following values:

0
  - Disable all logging.
1
  - Enable logging on the FATAL level, which logs very severe error events that will lead the driver to abort.
2
  - Enable logging on the ERROR level, which logs error events that might still allow the driver to continue running.
3
  - Enable logging on the WARNING level, which logs events that might result in an error if action is not taken.
4
  - Enable logging on the INFO level, which logs general information that describes the progress of the driver.
5
  - Enable logging on the DEBUG level, which logs detailed information that is useful for debugging the driver.
6
  - Enable logging on the TRACE level, which logs all driver activity.

When logging is enabled, the driver produces the following log files in the location specified in the LogPath property:

- `redshift_jdbc.log` – File that logs driver activity that is not specific to a connection.
- `redshift_jdbc_connection_[Number].log` – File for each connection made to the database, where `[Number]` is a number that distinguishes each log file from the others. This file logs driver activity that is specific to the connection.

If the LogPath value is invalid, the driver sends the logged information to the standard output stream, `System.out`.

**LogPath**

- **Default Value** – The current working directory.
- **Data Type** – String

The full path to the folder where the driver saves log files when the DSILogLevel property is enabled.

To be sure that the connection URL is compatible with all JDBC applications, we recommend that you escape the backslashes (`\`) in your file path by typing another backslash.
This parameter is optional.

**Partner_SPID**

- **Default Value** – None
- **Data Type** – String

The partner SPID (service provider ID) value to use when authenticating the connection using the PingFederate service.

This parameter is optional.

**Password**

- **Default Value** – None
- **Data Type** – String

When connecting using IAM authentication through an IDP, this is the password for the IDP.Host server. When using standard authentication, this can be used for the Amazon Redshift database password instead of PWD.

This parameter is optional.

**Plugin_Name**

- **Default Value** – None
- **Data Type** – String

The fully qualified class name to implement a specific credentials provider plugin.

This parameter is optional.

The following provider options are supported:

- **AdfsCredentialsProvider** – Active Directory Federation Service
- **AzureCredentialsProvider** – Microsoft Azure Active Directory (AD) Service
- **BasicJwtCredentialsProvider** – JSON Web Tokens (JWT) Service
- **BasicSamlCredentialsProvider** – Security Assertion Markup Language (SAML) credentials which you can use with many SAML service providers.
- **BrowserAzureCredentialsProvider** – Browser Microsoft Azure Active Directory (AD) Service
- **BrowserAzureOAuth2CredentialsProvider** – Browser Microsoft Azure Active Directory (AD) Service for Native Authentication
- **BrowserSamlCredentialsProvider** – Browser SAML for SAML services such as Okta, Ping, or ADFS
- **OktaCredentialsProvider** – Okta Service
- **PingCredentialsProvider** – PingFederate Service

**PORT**

- **Default Value** – null
- **Data Type** – Integer

The port of the Amazon Redshift server to connect to. You can use this option to specify the port in the JDBC connection URL.
This parameter is optional.

**Preferred_Role**

- **Default Value** – None
- **Data Type** – String

The IAM role that you want to assume during the connection to Amazon Redshift.

This parameter is optional.

**Profile**

- **Default Value** – None
- **Data Type** – String

The name of the profile to use for IAM authentication. This profile contains any additional connection properties not specified in the connection string.

This parameter is optional.

**PWD**

- **Default Value** – None
- **Data Type** – String

The password corresponding to the Amazon Redshift user name that you provided using the property UID.

This parameter is optional.

**queryGroup**

- **Default Value** – null
- **Data Type** – String

This option assigns a query to a queue at runtime by assigning your query to the appropriate query group. The query group is set for the session. All queries that run on the connection belong to this query group.

This parameter is optional.

**readOnly**

- **Default Value** – false
- **Data Type** – Boolean

This property specifies whether the driver is in read-only mode.

This parameter is optional.

**true**

The connection is in read-only mode and cannot write to the data store.
false

The connection is not in read-only mode and can write to the data store.

Region

- **Default Value** – null
- **Data Type** – String

This option specifies the AWS Region where the cluster is located. If you specify the StsEndPoint option, the Region option is ignored. The Redshift GetClusterCredentials API operation also uses the Region option.

This parameter is optional.

reWriteBatchedInserts

- **Default Value** – false
- **Data Type** – Boolean

This option enables optimization to rewrite and combine compatible INSERT statements into batches.

This parameter is optional.

reWriteBatchedInsertsSize

- **Default Value** – 128
- **Data Type** – Integer

This option enables optimization to rewrite and combine compatible INSERT statements into batches. This value must increase exponentially by the power of 2.

This parameter is optional.

roleArn

- **Default Value** – None
- **Data Type** – String

The Amazon Resource Name (ARN) of role. Make sure to specify this parameter when you specify BasicJwtCredentialsProvider for the Plugin_Name option. You specify the ARN in the following format:

```
arn:partition:service:region:account-id:resource-id
```

This parameter is required if you specify BasicJwtCredentialsProvider for the Plugin_Name option.

roleSessionName

- **Default Value** – jwt_redshift_session
- **Data Type** – String

An identifier for the assumed role session. Typically, you pass the name or identifier that is associated with the user of your application. The temporary security credentials that your application uses are associated with that user. You can specify this parameter when you specify BasicJwtCredentialsProvider for the Plugin_Name option.
This parameter is optional.

**scope**

- **Default Value** – None
- **Data Type** – String

A space-separated list of scopes to which the user can consent. You specify this parameter so that your Microsoft Azure application can get consent for APIs that you want to call. You can specify this parameter when you specify BrowserAzureOAuth2CredentialsProvider for the Plugin_Name option.

This parameter is required for the BrowserAzureOAuth2CredentialsProvider plug-in.

**SecretAccessKey**

- **Default Value** – None
- **Data Type** – String

The IAM access key for the user or role. If this is specified, then AccessKeyId must also be specified.

This parameter is optional.

**SessionToken**

- **Default Value** – None
- **Data Type** – String

The temporary IAM session token associated with the IAM role you are using to authenticate.

This parameter is optional.

**socketFactory**

- **Default Value** – null
- **Data Type** – String

This option specifies a socket factory for socket creation.

This parameter is optional.

**socketTimeout**

- **Default Value** – 0
- **Data Type** – Integer

The number of seconds to wait during socket read operations before timing out. If the operation takes longer than this threshold, then the connection is closed. When this property is set to 0, the connection doesn’t time out.

This parameter is optional.

**SSL**

- **Default Value** – TRUE
- **Data Type** – String
Use this property to turn on or turn off SSL for the connection.
This parameter is optional.
You can specify the following values:

**TRUE**

The driver connects to the server through SSL.

**FALSE**

The driver connects to the server without using SSL. This option is not supported with IAM authentication.

Alternatively, you can configure the AuthMech property.

**SSL_Insecure**

- **Default Value** – true
- **Data Type** – String

This property indicates whether the IDP hosts server certificate should be verified.
This parameter is optional.
You can specify the following values:

**true**

The driver doesn't check the authenticity of the IDP server certificate.

**false**

The driver checks the authenticity of the IDP server certificate.

**SSLCert**

- **Default Value** – None
- **Data Type** – String

The full path of a .pem or .crt file containing additional trusted CA certificates for verifying the Amazon Redshift server instance when using SSL.
This parameter is required if SSLKey is specified.

**SSLFactory**

- **Default Value** – None
- **Data Type** – String

The SSL factory to use when connecting to the server through TLS/SSL without using a server certificate.

**SSLKey**

- **Default Value** – None
- **Data Type** – String
The full path of the .der file containing the PKCS8 key file for verifying the certificates specified in SSLCert.

This parameter is required if SSLCert is specified.

**SSLMODE**

- **Default Value** – verify-ca
- **Data Type** – String

Use this property to specify how the driver validates certificates when TLS/SSL is enabled.

This parameter is optional.

You can specify the following values:

- **verify-ca**
  - The driver verifies that the certificate comes from a trusted certificate authority (CA).

- **verify-full**
  - The driver verifies that the certificate comes from a trusted CA and that the host name in the certificate matches the host name specified in the connection URL.

**SSLPassword**

- **Default Value** – 0
- **Data Type** – String

The password for the encrypted key file specified in SSLKey.

This parameter is required if SSLKey is specified and the key file is encrypted.

**SSLRootCert**

- **Default Value** – None
- **Data Type** – String

The full path of a .pem or .crt file containing the root CA certificate for verifying the Amazon Redshift Server instance when using SSL.

**StsEndpointUrl**

- **Default Value** – Null
- **Data Type** – String

You can specify an AWS Security Token Service (AWS STS) endpoint. If you specify this option, the Region option is ignored. You can only specify a secure protocol (HTTPS) for this endpoint.

**tcpKeepAlive**

- **Default Value** – TRUE
- **Data Type** – String

Use this property to turn on or turn off TCP keepalives.
This parameter is optional.

You can specify the following values:

TRUE

The driver uses TCP keepalives to prevent connections from timing out.

FALSE

The driver doesn’t use TCP keepalives.

UID

- **Default Value** – None
- **Data Type** – String

The user name that you use to access the database.

This parameter is required.

User

- **Default Value** – None
- **Data Type** – String

When connecting using IAM authentication through an IDP, this is the user name for the idp_host server. When using standard authentication this can be used for the Amazon Redshift database user name.

This parameter is optional.

webIdentityToken

- **Default Value** – None
- **Data Type** – String

The OAuth 2.1 access token or OpenID Connect ID token that is provided by the identity provider. Your application must get this token by authenticating the user of your application with a web identity provider. Make sure to specify this parameter when you specify BasicJwtCredentialsProvider for the Plugin_Name option.

This parameter is required if you specify BasicJwtCredentialsProvider for the Plugin_Name option.

**Previous versions of JDBC driver version 2.1**

Download a previous version of the Amazon Redshift JDBC driver version 2.1 only if your tool requires a specific version of the driver.

These are JDBC 4.2–compatible JDBC driver version 2.1 drivers:

Configuring the Amazon Redshift Python connector

By using the Amazon Redshift connector for Python, you can integrate work with the AWS SDK for Python (Boto3), and also pandas and Numerical Python (NumPy). For more information on pandas, see the pandas GitHub repository. For more information on NumPy, see the NumPy GitHub repository.

The Amazon Redshift Python connector provides an open source solution. You can browse the source code, request enhancements, report issues, and provide contributions.

To use the Amazon Redshift Python connector, make sure that you have Python version 3.6 or later. For more information, see the Amazon Redshift Python driver license agreement.

The Amazon Redshift Python connector provides the following:

- AWS Identity and Access Management (IAM) authentication. For more information, see Identity and access management in Amazon Redshift (p. 334).
- Identity provider authentication using federated API access. Federated API access is supported for corporate identity providers such as the following:
  - Azure AD. For more information, see the AWS Big Data blog post Federate Amazon Redshift access with Microsoft Azure AD single sign-on.
  - Active Directory Federation Services. For more information, see the AWS Big Data blog post Federate access to your Amazon Redshift cluster with Active Directory Federation Services (AD FS): Part 1.
  - Okta. For more information, see the AWS Big Data blog post Federate Amazon Redshift access with Okta as an identity provider.
  - PingFederate. For more information, see the PingFederate site.
  - JumpCloud. For more information, see the JumpCloud site.
- Amazon Redshift data types.
The Amazon Redshift Python connector implements Python Database API Specification 2.0. For more information, see PEP 249—Python Database API Specification v2.0 on the Python website.

Topics
- Installing the Amazon Redshift Python connector (p. 155)
- Configuration options for the Amazon Redshift Python connector (p. 156)
- Importing the Python connector (p. 163)
- Integrating the Python connector with NumPy (p. 164)
- Integrating the Python connector with pandas (p. 164)
- Using identity provider plugins (p. 164)
- Examples of using the Amazon Redshift Python connector (p. 166)
- API reference for the Amazon Redshift Python connector (p. 167)

Installing the Amazon Redshift Python connector

You can use any of the following methods to install the Amazon Redshift Python connector:

- Python Package Index (PyPI)
- Conda
- Cloning the GitHub repository

Installing the Python connector from PyPI

To install the Python connector from the Python Package Index (PyPI), you can use pip. To do this, run the following command.

```python
>>> pip install redshift_connector
```

You can install the connector within a virtual environment. To do this, run the following command.

```python
>>> pip install redshift_connector
```

Optionally, you can install pandas and NumPy with the connector.

```python
>>> pip install "redshift_connector[full]"
```

For more information on pip, see the pip site.

Installing the Python connector from Conda

You can install the Python connector from Anaconda.org.

```bash
>>> conda install -c conda-forge redshift_connector
```

Installing the Python connector by cloning the GitHub repository from AWS

To install the Python connector from source, clone the GitHub repository from AWS. After you install Python and virtualenv, set up your environment and install the required dependencies by running the following commands.

```bash
$ git clone https://github.com/aws/amazon-redshift-python-driver.git
$ cd RedshiftPythonDriver
$ virtualenv venv
```
Configuration options for the Amazon Redshift Python connector

Following, you can find descriptions for the options that you can specify for the Amazon Redshift Python connector.

access_key_id

- Default value – None
- Data type – String

The access key for the IAM role or IAM user configured for IAM database authentication. This parameter is optional.

allow_db_user_override

- Default value – False
- Data type – Boolean

True

Specifies that the connector uses the DbUser value from the Security Assertion Markup Language (SAML) assertion.

False

Specifies that the value in the DbUser connection parameter is used.

This parameter is optional.

app_name

- Default value – None
- Data type – String

The name of the identity provider (IdP) application used for authentication.

This parameter is optional.

auth_profile

- Default value – None
- Data type – String

The name of an Amazon Redshift authentication profile having connection properties as JSON. For more information about naming connection parameters, see the RedshiftProperty class. The RedshiftProperty class stores connection parameters provided by the end user and, if applicable, generated during the IAM authentication process (for example, temporary IAM credentials). For more information, see the RedshiftProperty class.

This parameter is optional.
auto_create

- **Default value** – False
- **Data type** – Boolean

A value that indicates whether to create the user if the user doesn't exist.

This parameter is optional.

client_id

- **Default value** – None
- **Data type** – String

The client ID from Azure IdP.

This parameter is optional.

client_secret

- **Default value** – None
- **Data type** – String

The client secret from Azure IdP.

This parameter is optional.

cluster_identifier

- **Default value** – None
- **Data type** – String

The cluster identifier of the Amazon Redshift cluster.

This parameter is optional.

credentials_provider

- **Default value** – None
- **Data type** – String

The IdP that is used for authenticating with Amazon Redshift. Following are valid values:

- OktaCredentialsProvider
- AzureCredentialsProvider
- BrowserAzureCredentialsProvider
- PingCredentialsProvider
- BrowserSamlCredentialsProvider
• **AdfsCredentialsProvider**

This parameter is optional.

**database**

• **Default value** – None
• **Data type** – String

The name of the database to which you want to connect.

This parameter is optional.

**database_metadata_current_db_only**

• **Default value** – True
• **Data type** – Boolean

A value that indicates whether an application supports multidatabase datashare catalogs. The default value of True indicates that the application doesn't support multidatabase datashare catalogs for backward compatibility.

This parameter is optional.

**db_groups**

• **Default value** – None
• **Data type** – String

A comma-separated list of existing database group names that the user indicated by DbUser joins for the current session.

This parameter is optional.

**db_user**

• **Default value** – None
• **Data type** – String

The user ID to use with Amazon Redshift.

This parameter is optional.

**endpoint_url**

• **Default value** – None
• **Data type** – String

The Amazon Redshift endpoint URL. This option is only for AWS internal use.

This parameter is required.

**host**

• **Default value** – None
• **Data type** – String

The hostname of Amazon Redshift cluster.
This parameter is optional.

iam

• **Default value** – False
• **Data type** – Boolean

IAM authentication is enabled.
This parameter is required.

iam_disable_cache

• **Default value** – False
• **Data type** – Boolean

This option specifies whether the IAM credentials are cached. By default, the IAM credentials are cached.
This improves performance when requests to the API gateway are throttled.
This parameter is optional.

idpPort

• **Default value** – 7890
• **Data type** – Integer

The listen port to which IdP sends the SAML assertion.
This parameter is required.

idp_response_timeout

• **Default value** – 120
• **Data type** – Integer

The timeout for retrieving SAML assertion from IdP.
This parameter is required.

idp_tenant

• **Default value** – None
• **Data type** – String

The IdP tenant.
This parameter is optional.

listen_port

• **Default value** – 7890
• **Data type** – Integer

The listen port to which the IdP sends the SAML assertion.
This parameter is optional.

```plaintext
login_url
```

• **Default value** – None
• **Data type** – String

The SSO Url for the IdP.
This parameter is optional.

```plaintext
max_prepared_statements
```

• **Default value** – 1000
• **Data type** – Integer

The maximum number of prepared statements that can be open concurrently.
This parameter is required.

```plaintext
partner_sp_id
```

• **Default value** – None
• **Data type** – String

The Partner SP ID used for authentication with Ping.
This parameter is optional.

```plaintext
password
```

• **Default value** – None
• **Data type** – String

The password to use for authentication.
This parameter is optional.

```plaintext
port
```

• **Default value** – 5439
• **Data type** – Integer

The port number of the Amazon Redshift cluster.
This parameter is required.

```plaintext
preferred_role
```

• **Default value** – None
• **Data type** – String
The IAM role preferred for the current connection. This parameter is optional.

**principal_arn**
- **Default value** – None
- **Data type** – String

The Amazon Resource Name (ARN) of the IAM user or role for which you are generating a policy. This parameter is optional.

**profile**
- **Default value** – None
- **Data type** – String

The name of a profile in an AWS credentials file that contains AWS credentials. This parameter is optional.

**provider_name**
- **Default value** – None
- **Data type** – String

The name of the Redshift Native Authentication Provider. This parameter is optional.

**region**
- **Default value** – None
- **Data type** – String

The AWS Region where the cluster is located. This parameter is optional.

**role_arn**
- **Default value** – None
- **Data type** – String

The Amazon Resource Name (ARN) of the role that the caller is assuming. This parameter is used by the provider indicated by JwtCredentialsProvider.

For the JwtCredentialsProvider provider, this parameter is mandatory. Otherwise, this parameter is optional.

**role_session_name**
- **Default value** – jwt_redshift_session
- **Data type** – String
An identifier for the assumed role session. Typically, you pass the name or identifier that is associated with the user who is using your application. The temporary security credentials that your application uses are associated with that user. This parameter is used by the provider indicated by JwtCredentialsProvider.

This parameter is optional.

**scope**

- **Default value** – None
- **Data type** – String

A space-separated list of scopes to which the user can consent. You specify this parameter so that your application can get consent for APIs that you want to call. You can specify this parameter when you specify BrowserAzureOAuth2CredentialsProvider for the Plugin_Name option.

This parameter is required for the BrowserAzureOAuth2CredentialsProvider plug-in.

**secret_access_key_id**

- **Default value** – None
- **Data type** – String

The secret access key for the IAM role or user configured for IAM database authentication.

This parameter is optional.

**session_token**

- **Default value** – None
- **Data type** – String

The access key for the IAM role or user configured for IAM database authentication. This parameter is required if temporary AWS credentials are being used.

This parameter is optional.

**ssl**

- **Default value** – True
- **Data type** – Boolean

Secure Sockets Layer (SSL) is enabled.

This parameter is required.

**ssl_insecure**

- **Default value** – True
- **Data type** – Boolean

A value that specifies whether the IdP hosts server certificate is to be verified.

This parameter is optional.
sslmode
- Default value – verify-ca
- Data type – String

The security of the connection to Amazon Redshift. You can specify either of the following:
- verify-ca
- verify-full

This parameter is required.

user
- Default value – None
- Data type – String

The user name to use for authentication.
This parameter is optional.

web_identity_token
- Default value – None
- Data type – String

The OAuth 2.0 access token or OpenID Connect ID token that is provided by the identity provider. Make sure that your application gets this token by authenticating the user who is using your application with a web identity provider. The provider indicated by JwtCredentialsProvider uses this parameter.

For the JwtCredentialsProvider provider, this parameter is mandatory. Otherwise, this parameter is optional.

Importing the Python connector

To import the Python connector, run the following command.

```python
>>> import redshift_connector
```

Importing NumPy and connecting to Amazon Redshift

To import the Amazon Redshift Python connector and Numerical Python (NumPy), run the following commands.

```python
import redshift_connector
import numpy
```

To connect to an Amazon Redshift cluster using AWS credentials, run the following command.

```python
conn = redshift_connector.connect(
    host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
    port=5439,
    database='dev',
    user='awsuser',
    password='my_password'
)```
Integrating the Python connector with NumPy

Following is an example of integrating the Python connector with NumPy.

```python
>>> import numpy
cursor.execute("select * from book")
result: numpy.ndarray = cursor.fetch_numpy_array()
print(result)
```

Following is the result.

```
[['One Hundred Years of Solitude' 'Gabriel García Márquez']
 ['A Brief History of Time' 'Stephen Hawking']]```

Integrating the Python connector with pandas

Following is an example of integrating the Python connector with pandas.

```python
>>> import pandas
cursor.execute("select * from book")
result: pandas.DataFrame = cursor.fetch_dataframe()
print(result)
```

Using identity provider plugins

For general information on how to use identity provider plugins, see Options for providing IAM credentials (p. 380).

Authentication using the ADFS identity provider plugin

Following is an example of using the Active Directory Federation Service (ADFS) identity provider plugin to authenticate a user connecting to an Amazon Redshift database.

```python
>>> con = redshift_connector.connect(
 iam=True,
 database='dev',
 host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
 cluster_identifier='my-testing-cluster',
 credentials_provider='AdfsCredentialsProvider',
 user='brooke@myadfshostname.com',
 password='Hunter2',
 idp_host='myadfshostname.com'
)
```

Authentication using the Azure identity provider plugin

Following is an example of authentication using the Azure identity provider plugin. You can create values for a client_id and client_secret for an Azure Enterprise application as shown following.

```python
>>> con = redshift_connector.connect(
 iam=True,
 database='dev',
 host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
 cluster_identifier='my-testing-cluster',
...
Credentials Provider

```python
credentials_provider='AzureCredentialsProvider',
user='brooke@myazure.org',
password='Hunter2',
idp_tenant='my_idp_tenant',
client_id='my_client_id',
client_secret='my_client_secret',
preferred_role='arn:aws:iam:123:role/DataScientist'
```

**Authentication using Azure Browser identity provider plugin**

Following is an example of using the Azure Browser identity provider plugin to authenticate a user connecting to an Amazon Redshift database.

Multi-factor authentication occurs in the browser, where the username and password are provided by the user.

```python
>>> con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='BrowserAzureCredentialsProvider',
    idp_tenant='my_idp_tenant',
    client_id='my_client_id',
)
```

**Authentication using the Okta identity provider plugin**

Following is an example of authentication using the Okta identity provider plugin. You can obtain the values for `idp_host`, `app_id` and `app_name` through the Okta application.

```python
>>> con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='OktaCredentialsProvider',
    user='brooke@myazure.org',
    password='hunter2',
    idp_host='my_idp_host',
    app_id='my_first_appetizer',
    app_name='dinner_party'
)
```

**Authentication using JumpCloud with a generic SAML browser identity provider plugin**

Following is an example of using JumpCloud with a generic SAML browser identity provider plugin for authentication.

The password parameter is required. However, you don't have to enter this parameter because multi-factor authentication occurs in the browser.

```python
>>> con = redshift_connector.connect(
    iam=True,
    database='dev',
    host='my-testing-cluster.abc.us-east-2.redshift.amazonaws.com',
    cluster_identifier='my-testing-cluster',
    credentials_provider='BrowserSamlCredentialsProvider',
    user='brooke@myjumpcloud.org',
    password='',
)
Examples of using the Amazon Redshift Python connector

Following are examples of how to use the Amazon Redshift Python connector.

Topics
- Connecting to an Amazon Redshift cluster using AWS credentials (p. 166)
- Querying a table (p. 166)
- Retrieving the query result set (p. 166)
- Enabling autocommit (p. 166)
- Using COPY to copy data from and UNLOAD to write data to an Amazon S3 bucket (p. 166)

Connecting to an Amazon Redshift cluster using AWS credentials

To connect to an Amazon Redshift cluster using your AWS credentials, run the following command.

```python
>>> conn = redshift_connector.connect(  
    host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',  
    database='dev',  
    user='awsuser',  
    password='my_password'
)
```

Querying a table

To select all rows from the table `book`, run the following command.

```python
>>> cursor.execute("select * from book")
```

Retrieving the query result set

To retrieve the query result set, run the following command.

```python
>>> result: tuple = cursor.fetchall()  
print(result)  
>> [('One Hundred Years of Solitude', 'Gabriel García Márquez'), ('A Brief History of Time', 'Stephen Hawking')]
```

Enabling autocommit

The autocommit property is off by default, following the Python Database API Specification. You can use the following commands to turn on the autocommit property of the connection. First, you perform a rollback command to make sure that a transaction is not in progress.

```python
>>> con.rollback()  
con.autocommit = True  
con.run("VACUUM")  
con.autocommit = False
```

Using COPY to copy data from and UNLOAD to write data to an Amazon S3 bucket

The following example shows how to copy data from an Amazon S3 bucket into a table and then unload from the table into the S3 bucket.
A text file named `category_csv.txt` containing the following data is uploaded to an S3 bucket.

```plaintext
>>> 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"
```

Following is an example of the Python code, which first connects to the Amazon Redshift database. It then creates a table called `category` and copies the CSV data from the S3 bucket into the table.

```python
>>> with redshift_connector.connect(...) as conn:
    with conn.cursor() as cursor:
        cursor.execute("create table category (catid int, cargroup varchar, catname varchar, catdesc varchar)"
        cursor.execute("copy category from 's3://testing/category_csv.txt' iam_role 'arn:aws:iam::123:role/RedshiftCopyUnload' csv;"
        cursor.execute("select * from category")
        print(cursor.fetchall())
        cursor.execute("unload ('select * from category') to 's3://testing/unloaded_category_csv.txt' iam_role 'arn:aws:iam::123:role/RedshiftCopyUnload' csv;"
        print('done')
```

The data is unloaded into the file `unloaded_category_csv.text0000_part00` in the S3 bucket.

```plaintext
>>> 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"
```

API reference for the Amazon Redshift Python connector

Following, you can find a description of the Amazon Redshift Python connector API operations.

**redshift_connector**

Following, you can find a description of the `redshift_connector` API operation.

**connect(user, database, password[, port, ...])**

Establishes a connection to an Amazon Redshift cluster. This function validates user input, optionally authenticates using an identity provider plugin, and then constructs a connection object.

**apilevel**

The DBAPI level supported, currently "2.0".

**paramstyle**

str(object=’’) -> str  str(bytes_or_buffer[, encoding[, errors]]) -> str

The database API parameter style to use globally.

**Connection**

Following, you can find a description of the connection API operations for the Amazon Redshift Python connector.
__init__(user, password, database[, host, _])

Initializes a raw connection object.
cursor

Creates a cursor object bound to this connection.
commit

Commits the current database transaction.
rollback

Rolls back the current database transaction.
close

Closes the database connection.
execute(cursor, operation, vals)

Runs the specified SQL command. You can provide the parameters as a sequence or as a mapping, depending upon the value of redshift_connector.paramstyle.
run(sql[, stream])

Runs the specified SQL command. Optionally, you can provide a stream for use with the COPY command.
xid(format_id, global_transaction_id, _)

Create a transaction ID. Only the global_transaction_id parameter is used in postgres. format_id and branch_qualifier are not used in postgres. The global_transaction_id can be any string identifier supported by postgres that returns a tuple (format_id, global_transaction_id, branch_qualifier).

tpc_begin(xid)

Begins a TPC transaction with a transaction ID xid consisting of a a format ID, global transaction ID, and branch qualifier.

tpc_prepare

Performs the first phase of a transaction started with .tpc_begin.

tpc_commit([xid])

When called with no arguments, .tpc_commit commits a TPC transaction previously prepared with .tpc_prepare().

tpc_rollback([xid])

When called with no arguments, .tpc_rollback rolls back a TPC transaction.

tpc_recover

Returns a list of pending transaction IDs suitable for use with .tpc_commit(xid) or .tpc_rollback(xid).

**Cursor**

Following, you can find a description of the cursor API operation.

__init__(connection[, paramstyle])

Initializes a raw cursor object.
execute(operation[, args, stream, …])

Runs a database operation.
executemany(operation, param_sets)

Prepares a database operation, and then runs it for all parameter sequences or mappings provided.

cursor

Fetched the next row of a query result set.

cursor.fetchmany([num])

Fetched the next set of rows of a query result.

cursor.fetchall

Fetched all remaining rows of a query result.

close

Close the cursor now.

__iter__

A cursor object can be iterated to retrieve the rows from a query.

cursor.fetch_dataframe([num])

Returns a dataframe of the last query results.

cursor.write_dataframe(df, table)

Writes the same structure dataframe into an Amazon Redshift database.

cursor.fetch_numpy_array([num])

Returns a NumPy array of the last query results.

get_catalogs

Amazon Redshift doesn't support multiple catalogs from a single connection. Amazon Redshift only returns the current catalog.

cursor.get_tables([catalog, schema_pattern, ...])

Returns the unique public tables which are user-defined within the system.

cursor.get_columns([catalog, schema_pattern, ...])

Returns a list of all columns in a specific table in an Amazon Redshift database.

AdfsCredentialsProvider plugin

Following is the syntax for the AdfsCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```python
redshift_connector.plugin.AdfsCredentialsProvider()
```

AzureCredentialsProvider plugin

Following is the syntax for the AzureCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```python
redshift_connector.plugin.AzureCredentialsProvider()
```
**BrowserAzureCredentialsProvider plugin**

Following is the syntax for the BrowserAzureCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```
redshift_connector.plugin.BrowserAzureCredentialsProvider()
```

**BrowserSamlCredentialsProvider plugin**

Following is the syntax for the BrowserSamlCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```
redshift_connector.plugin.BrowserSamlCredentialsProvider()
```

**OktaCredentialsProvider plugin**

Following is the syntax for the OktaCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```
redshift_connector.plugin.OktaCredentialsProvider()
```

**PingCredentialsProvider plugin**

Following is the syntax for the PingCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```
redshift_connector.plugin.PingCredentialsProvider()
```

**SamlCredentialsProvider plugin**

Following is the syntax for the SamlCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```
redshift_connector.plugin.SamlCredentialsProvider()
```

---

**Configuring a JDBC driver version 1.x connection**

**Important**

The JDBC driver version 1.x is discontinued. Further updates will not be released, except for urgent security patches. Support for JDBC driver version 1.x branch will be discontinued on October 1, 2022.
We recommend that you use the open source Amazon Redshift JDBC driver version 2.1, which is actively maintained. For more information, see Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 122).

**Download the Amazon Redshift JDBC driver**

Amazon Redshift offers drivers for tools that are compatible with the JDBC 4.2 API. For information about the functionality supported by these drivers, see the Amazon Redshift JDBC driver release notes.
For detailed information about how to install the JDBC driver version 1.x, reference the JDBC driver libraries, and register the driver class, see Amazon Redshift JDBC driver installation and configuration guide.

For each computer where you use the Amazon Redshift JDBC driver, make sure that Java Runtime Environment (JRE) 8.0 is installed.

If you use the Amazon Redshift JDBC driver for database authentication, make sure that you have AWS SDK for Java 1.11.118 or later in your Java class path. If you don't have AWS SDK for Java installed, download the ZIP file with JDBC 4.2–compatible driver (without the AWS SDK) and driver dependent libraries for the AWS SDK:

- **JDBC 4.2–compatible driver (without the AWS SDK) and driver dependent libraries for AWS SDK files version 1.2.55.**

  The class name for this driver is `com.amazon.redshift.jdbc42.Driver`.

  This ZIP file contains the JDBC 4.2–compatible driver (without the AWS SDK) and its dependent library files. Unzip the dependent jar files to the same location as the JDBC driver. Only the JDBC driver needs to be in the CLASSPATH because the driver manifest file contains all dependent library file names which are located in the same directory as the JDBC driver. For more information about how to install the JDBC driver, see Amazon Redshift JDBC driver installation and configuration guide.

  Use this Amazon Redshift JDBC driver with the AWS SDK that is required for IAM database authentication.

- **JDBC 4.2–compatible driver (without the AWS SDK) version 1.2.55.**

  The class name for this driver is `com.amazon.redshift.jdbc42.Driver`.

  Be sure to use ANTLR version 4.8.1. The antlr4-runtime-4.8-1.jar is included in the ZIP download link above with the JDBC 4.2–compatible driver (without the AWS SDK) and driver dependent libraries for the AWS SDK.

  For more information about previous driver versions, see Use previous JDBC driver versions with the AWS SDK for Java (p. 177).

Then download and review the Amazon Redshift ODBC and JDBC driver license agreement.

If your tool requires a specific previous version of a driver, see Use previous JDBC driver 1.x versions in certain cases (p. 175).

**Getting the JDBC URL**

Before you can connect to your Amazon Redshift cluster from a SQL client tool, you need to know the JDBC URL of your cluster. The JDBC URL has the following format: `jdbc:redshift://endpoint:port/database`.

**Note**

A JDBC URL specified with the former format of `jdbc:postgresql://endpoint:port/database` still works.

The fields of the format shown preceding have the following values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>jdbc</td>
<td>The protocol for the connection.</td>
</tr>
<tr>
<td>redshift</td>
<td>The subprotocol that specifies to use the Amazon Redshift driver to connect to the database.</td>
</tr>
</tbody>
</table>
Configuring connections in Amazon Redshift

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>endpoint</strong></td>
<td>The endpoint of the Amazon Redshift cluster.</td>
</tr>
<tr>
<td><strong>port</strong></td>
<td>The port number that you specified when you launched the cluster. If you have a firewall, make sure that this port is open for you to use.</td>
</tr>
<tr>
<td><strong>database</strong></td>
<td>The database that you created for your cluster.</td>
</tr>
</tbody>
</table>

The following is an example JDBC URL: `jdbc:redshift://examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com:5439/dev`

For information about how to get your JDBC connection, see [Finding your cluster connection string](#) (p. 122).

If the client computer fails to connect to the database, you can troubleshoot possible issues. For more information, see [Troubleshooting connection issues in Amazon Redshift](#) (p. 233).

**Configure authentication and SSL for JDBC connection**

Configure the Amazon Redshift JDBC driver to authenticate your connection according to the security requirements of the Amazon Redshift server that you are connecting to.

To authenticate the connection, always provide your Amazon Redshift user name and password. The password is transmitted using a salted MD5 hash of the password. Depending on whether SSL is enabled and required on the server, you might also need to configure the driver to connect through SSL. You might need to use one-way SSL authentication so that the client (the driver itself) verifies the identity of the server.

For information about configuring the JDBC driver to authenticate the connection, see [Amazon Redshift JDBC driver installation and configuration guide](#).

**Configure TCP keepalives for JDBC connection**

By default, the Amazon Redshift JDBC driver is configured to use TCP keepalives to prevent connections from timing out. You can specify when the driver starts sending keepalive packets or disable the feature by setting the relevant properties in the connection URL.

For information about configuring TCP keepalives for the JDBC driver, see [Amazon Redshift JDBC driver installation and configuration guide](#).

**Configure logging for JDBC connection**

To help troubleshoot issues, you can enable logging in the JDBC driver.

For information about configuring logging for JDBC connection, see [Amazon Redshift JDBC driver installation and configuration guide](#).

**Configure JDBC connection with Apache Maven**

Apache Maven is a software project management and comprehension tool. The AWS SDK for Java supports Apache Maven projects. For more information, see [Using the SDK with Apache Maven](#) in the [AWS SDK for Java Developer Guide](#).

If you use Apache Maven, you can configure and build your projects to use an Amazon Redshift JDBC driver to connect to your Amazon Redshift cluster. To do this, add the JDBC driver as a dependency in your project's `pom.xml` file. If you use Maven to build your project and want to use a JDBC connection, take the steps in the following section.
Configuring the JDBC driver as a Maven dependency

To configure the JDBC driver as a Maven dependency

1. Add the following repository to the repositories section of your pom.xml file.

   ```xml
   <repositories>
     <repository>
       <id>redshift</id>
       <url>http://redshift-maven-repository.s3-website-us-east-1.amazonaws.com/release</url>
     </repository>
   </repositories>
   
   To connect using SSL, add the following repository to your pom.xml file.

   ```xml
   <repositories>
     <repository>
       <id>redshift</id>
       <url>https://s3.amazonaws.com/redshift-maven-repository/release</url>
     </repository>
   </repositories>
   
2. Declare the version of the driver that you want to use in the dependencies section of your pom.xml file.

   Amazon Redshift offers drivers for tools that are compatible with the JDBC 4.2 API. For information about the functionality supported by these drivers, see Download the Amazon Redshift JDBC driver (p. 170).

   Add a dependency for the driver from the following list.

   ```xml
   <dependency>
     <groupId>com.amazon.redshift</groupId>
     <artifactId>redshift-jdbc42</artifactId>
     <version>driver-version</version>
   </dependency>
   
   The class name for this driver is com.amazon.redshift.jdbc42.Driver.

3. Download and review the Amazon Redshift ODBC and JDBC driver license agreement.

   The standard Amazon Redshift JDBC drivers include the AWS SDK that is required to use IAM database authentication. We recommend using the standard drivers unless the size of the driver files is an issue for your application. If you need smaller driver files and you do not use IAM database authentication, or if
you already have AWS SDK for Java 1.11.118 or later in your Java class path, then add a dependency for the driver from the following list.

Replace the driver-version in the following example with your driver version. For example, 1.2.55.1083.

- JDBC 4.2–compatible driver:

```xml
<dependency>
  <groupId>com.amazon.redshift</groupId>
  <artifactId>redshift-jdbc42-no-awssdk</artifactId>
  <version>driver-version</version>
</dependency>
```

The class name for this driver is com.amazon.redshift.jdbc42.Driver.

The Amazon Redshift Maven drivers with no SDKs include the following required dependency that you must include in your project.

```xml
<dependency>
  <groupId>org.antlr</groupId>
  <artifactId>antlr4</artifactId>
  <version>4.5.3</version>
  <scope>runtime</scope>
  <optional>false</optional>
</dependency>
```

The Amazon Redshift Maven drivers with no SDKs include the following optional dependencies that you can include in your project as needed.

```xml
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-core</artifactId>
  <version>1.11.118</version>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-redshift</artifactId>
  <version>1.11.118</version>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-sts</artifactId>
  <version>1.11.118</version>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
```

If your tool requires a specific previous version of a driver, see Use previous JDBC driver versions with Maven (p. 178).

**Upgrading the driver to the latest version**

To upgrade or change the Amazon Redshift JDBC driver to the latest version, first modify the version section of the dependency to the latest version of the driver. Then clean your project with the Maven Clean Plugin, as shown following.
mvn clean

**Configure JDBC driver options**

To control the behavior of the Amazon Redshift JDBC driver, you can append configuration options to the JDBC URL. For example, the following JDBC URL connects to your cluster using Secure Socket Layer (SSL), user (UID), and password (PWD).

```
jdbc:redshift://examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com:5439/dev?
ssl=true&UID=your_username&PWD=your_password
```

For more information about SSL options, see Connect using SSL (p. 193).

For information about how to set up JDBC driver configuration options, see Amazon Redshift JDBC driver installation and configuration guide.

**Use previous JDBC driver 1.x versions in certain cases**

Download a previous version of the Amazon Redshift JDBC driver version 1.x only if your tool requires a specific version of the driver. For information about the functionality supported in these versions of the drivers, see Download the Amazon Redshift JDBC driver (p. 170).

Then download and review the Amazon Redshift ODBC and JDBC driver license agreement.

For authentication using AWS Identity and Access Management (IAM) credentials or identity provider (IdP) credentials, use Amazon Redshift JDBC driver version 1.2.8.1005 or later.

**Important**
Amazon Redshift has changed the way that SSL certificates are managed. If you must use a driver version earlier than 1.2.8.1005, you might need to update your current trust root CA certificates to continue to connect to your clusters using SSL. For more information, see Transitioning to ACM certificates for SSL connections (p. 196).

If you use the Amazon Redshift JDBC driver for database authentication, make sure that you have AWS SDK for Java 1.11.118 or later in your Java class path. If you don't have AWS SDK for Java installed, you can use a driver that includes the AWS SDK. For more information, see Use previous JDBC driver versions with the AWS SDK for Java (p. 177).

These are JDBC 4.2–compatible drivers:

- [https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.55.1083/RedshiftJDBC42-no-awssdk-1.2.55.1083.jar](https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.55.1083/RedshiftJDBC42-no-awssdk-1.2.55.1083.jar).
- [https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.54.1082/RedshiftJDBC42-no-awssdk-1.2.54.1082.jar](https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.54.1082/RedshiftJDBC42-no-awssdk-1.2.54.1082.jar).
- [https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.50.1077/RedshiftJDBC42-no-awssdk-1.2.50.1077.jar](https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.50.1077/RedshiftJDBC42-no-awssdk-1.2.50.1077.jar).
- [https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.47.1071/RedshiftJDBC42-no-awssdk-1.2.47.1071.jar](https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.47.1071/RedshiftJDBC42-no-awssdk-1.2.47.1071.jar).
- [https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.45.1069/RedshiftJDBC42-no-awssdk-1.2.45.1069.jar](https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.45.1069/RedshiftJDBC42-no-awssdk-1.2.45.1069.jar).
- [https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.43.1067/RedshiftJDBC42-no-awssdk-1.2.43.1067.jar](https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.43.1067/RedshiftJDBC42-no-awssdk-1.2.43.1067.jar).
These ZIP files contain JDBC 4.2–compatible drivers and driver–dependent AWS SDK library files. Unzip the dependent jar files to the same location as the JDBC driver. Only the JDBC driver needs to be in the CLASSPATH. For more information about how to install the JDBC driver, see Amazon Redshift JDBC driver installation and configuration guide.

These are previous JDBC 4.1–compatible drivers:
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.34.1058/RedshiftJDBC41-no-awssdk-1.2.34.1058.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.32.1056/RedshiftJDBC41-no-awssdk-1.2.32.1056.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.27.1051/RedshiftJDBC41-no-awssdk-1.2.27.1051.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.16.1027/RedshiftJDBC41-no-awssdk-1.2.16.1027.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.15.1025/RedshiftJDBC41-no-awssdk-1.2.15.1025.jar.

These are previous JDBC 4.0–compatible drivers:
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.43.1067/RedshiftJDBC4-no-awssdk-1.2.43.1067.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.41.1065/RedshiftJDBC4-no-awssdk-1.2.41.1065.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.37.1061/RedshiftJDBC4-no-awssdk-1.2.37.1061.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.36.1060/RedshiftJDBC4-no-awssdk-1.2.36.1060.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.34.1058/RedshiftJDBC4-no-awssdk-1.2.34.1058.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.32.1056/RedshiftJDBC4-no-awssdk-1.2.32.1056.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.27.1051/RedshiftJDBC4-no-awssdk-1.2.27.1051.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.20.1043/RedshiftJDBC4-no-awssdk-1.2.20.1043.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.16.1027/RedshiftJDBC4-no-awssdk-1.2.16.1027.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.15.1025/RedshiftJDBC4-no-awssdk-1.2.15.1025.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.12.1017/RedshiftJDBC4-no-awssdk-1.2.12.1017.jar.

Use previous JDBC driver versions with the AWS SDK for Java
If you use the JDBC driver for database authentication, make sure that you have AWS SDK for Java 1.11.118 or later in your Java class path. If you don't have AWS SDK for Java installed, you can use one of the following drivers that include the AWS SDK.

• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.43.1067/RedshiftJDBC42-1.2.43.1067.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.41.1065/RedshiftJDBC42-1.2.41.1065.jar.
• https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/1.2.37.1061/RedshiftJDBC42-1.2.37.1061.jar.
Use previous JDBC driver versions with Maven

Add a previous version of the Amazon Redshift JDBC driver to your project only if your tool requires a specific version of the driver. For information about the functionality supported in these driver versions,
see Download the Amazon Redshift JDBC driver (p. 170). For information about configuring with Maven, see Configure JDBC connection with Apache Maven (p. 172).

Configuring an ODBC connection

You can use an ODBC connection to connect to your Amazon Redshift cluster from many third-party SQL client tools and applications. To do this, set up the connection on your client computer or Amazon EC2 instance. If your client tool supports JDBC, you might choose to use that type of connection rather than ODBC due to the ease of configuration that JDBC provides. However, if your client tool doesn’t support JDBC, follow the steps in this section to configure an ODBC connection.

Amazon Redshift provides 64-bit ODBC drivers for Linux, Windows, and macOS X operating systems. The 32-bit ODBC drivers are discontinued. Further updates will not be released, except for urgent security patches.

For the latest information about ODBC driver functionality and prerequisites, see Amazon Redshift ODBC driver release notes.

For installation and configuration information for Amazon Redshift ODBC drivers, see Amazon Redshift ODBC connector installation and configuration guide.

If you want to use an ODBC connection, take the following steps.

Topics
- Obtain the ODBC URL for your cluster (p. 179)
- Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180)
- Install the Amazon Redshift ODBC driver on Linux (p. 182)
- Install the Amazon Redshift ODBC driver on macOS X (p. 183)
- Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 184)
- Configure ODBC driver options (p. 188)
- Use previous ODBC driver versions in certain cases (p. 188)

Obtain the ODBC URL for your cluster

Amazon Redshift displays the ODBC URL for your cluster in the Amazon Redshift console. This URL contains the information to set up the connection between your client computer and the database.

An ODBC URL has the following format:
```
Driver={driver};Server=endpoint;Database=database_name;UID=user_name;PWD=password;Port=port
```

The fields of the format shown preceding have the following values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver</td>
<td>The name of the 64-bit ODBC driver to use: Amazon Redshift (x64). The name of the 32-bit ODBC driver: Amazon Redshift (x86).</td>
</tr>
<tr>
<td>Server</td>
<td>The endpoint of the Amazon Redshift cluster.</td>
</tr>
<tr>
<td>Database</td>
<td>The database that you created for your cluster.</td>
</tr>
<tr>
<td>UID</td>
<td>The user name of a user account that has permission to connect to the database. This value is a database permission, not an Amazon Redshift permission, although you can use the admin user account that you set up when you launched the cluster.</td>
</tr>
</tbody>
</table>
Configuring connections in Amazon Redshift

### Field | Value
---|---
PWD | The password for the user account to connect to the database.
Port | The port number that you specified when you launched the cluster. If you have a firewall, ensure that this port is open for you to use.

The following is an example ODBC URL:

```
Driver={Amazon Redshift (x64)};
Server=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com;
Database=dev; UID=adminuser; PWD=insert_your_admin_user_password_here;
Port=5439
```

For information about how to get your ODBC connection, see Finding your cluster connection string (p. 122).

### Install and configure the Amazon Redshift ODBC driver on Microsoft Windows

#### System requirements

You install the Amazon Redshift ODBC driver on client computers accessing an Amazon Redshift data warehouse. Each computer where you install the driver must meet a list of minimum system requirements. For information about minimum system requirements, see Amazon Redshift ODBC connector installation and configuration guide.

#### Installing the Amazon Redshift driver on Windows operating systems

Use the following procedure to download the Amazon Redshift ODBC drivers for Windows operating systems. Only use a driver other than these if you're running a third-party application that is certified for use with Amazon Redshift and that requires a specific driver.

#### To install the ODBC driver

1. Download one of the following, depending on the system architecture of your SQL client tool or application:
   - 64-bit ODBC driver version 1.4.49
     The name for this driver is Amazon Redshift (x64).
   - 32-bit ODBC driver version 1.4.34
     The name for this driver is Amazon Redshift (x86). The 32-bit ODBC drivers are discontinued. Further updates will not be released, except for urgent security patches.

   **Note**
   Download the MSI package that corresponds to the system architecture of your SQL client tool or application. For example, if your SQL client tool is 64-bit, install the 64-bit driver.

   Then download and review the Amazon Redshift ODBC and JDBC driver license agreement.

2. Double-click the .msi file, and then follow the steps in the wizard to install the driver.

#### Creating a system DSN entry for an ODBC connection on Microsoft Windows

After you download and install the ODBC driver, add a data source name (DSN) entry to the client computer or Amazon EC2 instance. SQL client tools use this data source to connect to the Amazon Redshift database.
We recommend that you create a system DSN instead of a user DSN. Some applications load the data using a different user account. These applications might not be able to detect user DSNs that are created under another user account.

**Note**
For authentication using AWS Identity and Access Management (IAM) credentials or identity provider (IdP) credentials, additional steps are required. For more information, see Configure a JDBC or ODBC connection to use IAM credentials (p. 375).

For information about how to create a system DSN entry, see Amazon Redshift ODBC connector installation and configuration guide.

**To create a system DSN entry for an ODBC connection on Windows**

1. In the **Start** menu, open **ODBC Data Sources**.

   Make sure that you choose the ODBC Data Source Administrator that has the same bitness as the client application that you are using to connect to Amazon Redshift.

2. In the **ODBC Data Source Administrator**, choose the **Driver** tab and locate the driver folder:

   - Amazon Redshift ODBC Driver (64-bit)
   - Amazon Redshift ODBC Driver (32-bit)

3. Choose the **System DSN** tab to configure the driver for all users on the computer, or the **User DSN** tab to configure the driver for your user account only.

4. Choose **Add**. The **Create New Data Source** window opens.

5. Choose the **Amazon Redshift** ODBC driver, and then choose **Finish**. The **Amazon Redshift ODBC Driver DSN Setup** window opens.

6. Under **Connection Settings**, enter the following information:

   **Data source name**

   Enter a name for the data source. You can use any name that you want to identify the data source later when you create the connection to the cluster. For example, if you followed the Amazon Redshift Getting Started Guide, you might type exampleclusterdns to make it easy to remember the cluster that you associate with this DSN.

   **Server**

   Specify the endpoint for your Amazon Redshift cluster. You can find this information in the Amazon Redshift console on the cluster's details page. For more information, see Configuring connections in Amazon Redshift (p. 121).

   **Port**

   Enter the port number that the database uses. By default, Amazon Redshift uses 5439, but use the port that the cluster was configured to use when it was launched.

   **Database**

   Enter the name of the Amazon Redshift database. If you launched your cluster without specifying a database name, enter **dev**. Otherwise, use the name that you chose during the launch process. If you followed the Amazon Redshift Getting Started Guide, enter **dev**.

7. Under **Authentication**, specify the configuration options to configure standard or IAM authentication. For information about authentication options, see "Configuring Authentication on Windows" in Amazon Redshift ODBC Connector Installation and Configuration Guide.

8. Under **SSL Settings**, specify a value for the following:

   **SSL authentication**
Choose a mode for handling Secure Sockets Layer (SSL). In a test environment, you might use prefer. However, for production environments and when secure data exchange is required, use verify-ca or verify-full. For more information about using SSL on Windows, see "Configuring SSL Verification on Windows" in Amazon Redshift ODBC Connector Installation and Configuration Guide.

9. Under Additional Options, specify options on how to return query results to your SQL client tool or application. For more information, see "Configuring Additional Options on Windows" in Amazon Redshift ODBC Connector Installation and Configuration Guide.

10. In Logging Options, specify values for the logging option. For more information, see "Configuring Logging Options on Windows" in Amazon Redshift ODBC Connector Installation and Configuration Guide.

Then choose OK.

11. Under Data Type Options, specify values for data types. For more information, see "Configuring Data Type Options on Windows" in Amazon Redshift ODBC Connector Installation and Configuration Guide.

Then choose OK.

12. Choose Test. If the client computer can connect to the Amazon Redshift database, you see the following message: Connection successful.

If the client computer fails to connect to the database, you can troubleshoot possible issues. For more information, see Troubleshooting connection issues in Amazon Redshift (p. 233).

13. Configure TCP keepalives on Windows to prevent connections from timing out. For information about how to configure TCP keepalives on Windows, see Amazon Redshift ODBC Connector Installation and Configuration Guide.

14. To help troubleshooting, configure logging. For information about how to configure logging on Windows, see Amazon Redshift ODBC Connector Installation and Configuration Guide.

### Install the Amazon Redshift ODBC driver on Linux

#### System requirements

You install the Amazon Redshift ODBC driver on client computers accessing an Amazon Redshift data warehouse. Each computer where you install the driver must meet a list of minimum system requirements. For information about minimum system requirements, see Amazon Redshift ODBC connector installation and configuration guide.

#### Installing the Amazon Redshift driver on Linux operating systems

Use the steps in this section to download and install the Amazon Redshift ODBC drivers on a supported Linux distribution. The installation process installs the driver files in the following directories:

- /opt/amazon/redshiftodbc/lib/64 (for the 64-bit driver)
- /opt/amazon/redshiftodbc/ErrorMessages
- /opt/amazon/redshiftodbc/Setup
- /opt/amazon/redshiftodbc/lib/32 (for the 32-bit driver)

**To install the Amazon Redshift ODBC driver**

1. Download one of the following, depending on the system architecture of your SQL client tool or application:

   - 64-bit RPM driver version 1.4.49
• 64-bit Debian driver version 1.4.49
• 32-bit RPM driver version 1.4.34
• 32-bit Debian driver version 1.4.34

The name for each of these drivers is Amazon Redshift ODBC driver. The 32-bit ODBC drivers are discontinued. Further updates will not be released, except for urgent security patches.

Note
Download the package that corresponds to the system architecture of your SQL client tool or application. For example, if your client tool is 64-bit, install a 64-bit driver.

Then download and review the Amazon Redshift ODBC and JDBC driver license agreement.

2. Go to the location where you downloaded the package, and then run one of the following commands. Use the command that corresponds to your Linux distribution.

• On RHEL and CentOS operating systems, run the following command.

```
yum --nogpgcheck localinstall RPMFileName
```

Replace `RPMFileName` with the RPM package file name. For example, the following command demonstrates installing the 64-bit driver.

```
yum --nogpgcheck localinstall AmazonRedshiftODBC-64-bit-1.x.xxxxx-x.x86_64.deb
```

• On SLES, run the following command.

```
zypper install RPMFileName
```

Replace `RPMFileName` with the RPM package file name. For example, the following command demonstrates installing the 64-bit driver.

```
zypper install AmazonRedshiftODBC-1.x.xxxxx-x.x86_64.rpm
```

• On Debian, run the following command.

```
sudo apt install ./DEBFileName.deb
```

Replace `DEBFileName.deb` with the Debian package file name. For example, the following command demonstrates installing the 64-bit driver.

```
sudo apt install ./AmazonRedshiftODBC-1.x.xxxxx-x.x86_64.deb
```

Important
When you have finished installing the drivers, configure them for use on your system. For more information on driver configuration, see Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 184).

Install the Amazon Redshift ODBC driver on macOS X

System requirements
You install the driver on client computers accessing an Amazon Redshift data warehouse. Each computer where you install the driver must meet a list of minimum system requirements. For information about
Installing the Amazon Redshift ODBC driver on macOS X

Use the steps in this section to download and install the Amazon Redshift ODBC driver on a supported version of macOS X. The installation process installs the driver files in the following directories:

- /opt/amazon/redshift/lib/universal
- /opt/amazon/redshift/ErrorMessages
- /opt/amazon/redshift/Setup

To install the Amazon Redshift ODBC driver on macOS X

1. Download the macOS X driver version 1.4.49. The name for this driver is Amazon Redshift ODBC driver.

   Important
   After certificate rotation on September 4, 2020, ODBC Driver version 1.4.8.1000 or earlier on macOS will not be able to establish connections to Amazon Redshift clusters. For more information, see Driver update required for Amazon Redshift ODBC Driver earlier than 1.4.10 on Apple macOS.

   Then download and review the Amazon Redshift ODBC and JDBC driver license agreement.

2. Double-click AmazonRedshiftODBC.dmg to mount the disk image.

3. Double-click AmazonRedshiftODBC.pkg to run the installer.

4. Follow the steps in the installer to complete the driver installation process. To perform the installation, agree to the terms of the license agreement.

   Important
   When you have finished installing the driver, configure it for use on your system. For more information on driver configuration, see Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 184).

Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems

On Linux and macOS X operating systems, you use an ODBC driver manager to configure the ODBC connection settings. ODBC driver managers use configuration files to define and configure ODBC data sources and drivers. The ODBC driver manager that you use depends on the operating system that you use:

- unixODBC driver manager (for Linux operating systems)
- iODBC driver manager (for macOS X operating system)

For more information about the supported ODBC driver managers to configure the Amazon Redshift ODBC drivers, see System requirements (p. 182) for Linux operating systems and System requirements (p. 183) for macOS X operating systems. Also, see "Specifying ODBC Driver Managers on Non- Windows Machines" in Amazon Redshift ODBC connector installation and configuration guide.

Three files are required for configuring the Amazon Redshift ODBC driver: amazon.redshiftodbc.ini, odbc.ini, and odbcinst.ini.

If you installed to the default location, the amazon.redshiftodbc.ini configuration file is located in one of the following directories:
• /opt/amazon/redshiftodbc/lib/64 (for the 64-bit driver on Linux operating systems)
• /opt/amazon/redshiftodbc/lib/32 (for the 32-bit driver on Linux operating systems)
• /opt/amazon/redshift/lib (for the driver on macOS X)

Additionally, under /opt/amazon/redshiftodbc/Setup on Linux or /opt/amazon/redshift/Setup on macOS X, there are sample odbc.ini and odbcinst.ini files. You can use these files as examples for configuring the Amazon Redshift ODBC driver and the data source name (DSN).

We don't recommend using the Amazon Redshift ODBC driver installation directory for the configuration files. The sample files in the Setup directory are for example purposes only. If you reinstall the Amazon Redshift ODBC driver at a later time, or upgrade to a newer version, the installation directory is overwritten. You then lose any changes that you might have made to those files.

To avoid this, copy the amazon.redshiftodbc.ini file to a directory other than the installation directory. If you copy this file to the user's home directory, add a period (.) to the beginning of the file name to make it a hidden file.

For the odbc.ini and odbcinst.ini files, either use the configuration files in the user's home directory or create new versions in another directory. By default, your Linux or macOS X operating system should have an odbc.ini file and an odbcinst.ini file in the user's home directory (/home/ #USER or ~/.). These default files are hidden files, which is indicated by the dot (.) in front of each file name. These files display only when you use the –a flag to list the directory contents.

Whichever option you choose for the odbc.ini and odbcinst.ini files, modify the files to add driver and DSN configuration information. If you create new files, you also need to set environment variables to specify where these configuration files are located.

By default, ODBC driver managers are configured to use hidden versions of the odbc.ini and odbcinst.ini configuration files (named .odbc.ini and .odbcinst.ini) located in the home directory. They also are configured to use the amazon.redshiftodbc.ini file in the /lib subfolder of the driver installation directory. If you store these configuration files elsewhere, set the environment variables described following so that the driver manager can locate the files. For more information, see "Specifying the Locations of the Driver Configuration Files" in Amazon Redshift ODBC connector installation and configuration guide.

Creating a data source name on Linux and macOS X operating systems

When connecting to your data store using a data source name (DSN), configure the odbc.ini file to define DSNs. Set the properties in the odbc.ini file to create a DSN that specifies the connection information for your data store.

For information about how to configure the odbc.ini file, see "Creating a Data Source Name on a Non-Windows Machine" in Amazon Redshift ODBC connector installation and configuration guide.

Use the following format on Linux operating systems.

```
[ODBC Data Sources]

driver_name=dsn_name

[dsn_name]
Driver=path/driver_file
Host=cluster_endpoint
Port=port_number
Database=database_name
locale=locale
```

The following example shows the configuration for odbc.ini with the 64-bit ODBC driver on Linux operating systems.
Configuring connections in Amazon Redshift

Configuring a DSN-Less Connection

```ini
[ODBC Data Sources]
Amazon_Redshift_x64=Amazon Redshift (x64)

[Amazon Redshift (x64)]
Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

The following example shows the configuration for odbc.ini with the 32-bit ODBC driver on Linux operating systems.

```ini
[ODBC Data Sources]
Amazon_Redshift_x32=Amazon Redshift (x86)

[Amazon Redshift (x86)]
Driver=/opt/amazon/redshiftodbc/lib/32/libamazonredshiftodbc32.so
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

Use the following format on macOS X operating systems.

```ini
[ODBC Data Sources]
driver_name=dsn_name

[dsn_name]
Driver=/path/lib/amazonredshiftodbc.dylib
Host=cluster_endpoint
Port=port_number
Database=database_name
locale=locale
```

The following example shows the configuration for odbc.ini on macOS X operating systems.

```ini
[ODBC Data Sources]
Amazon_Redshift_dylib=Amazon Redshift DSN for macOS X

[Amazon Redshift DSN for macOS X]
Driver=/opt/amazon/redshift/lib/amazonredshiftodbc.dylib
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US
```

Configuring a connection without a DSN on Linux and macOS X operating systems

To connect to your data store through a connection that doesn't have a DSN, define the driver in the odbcinst.ini file. Then provide a DSN-less connection string in your application.

For information about how to configure the odbcinst.ini file in this case, see "Configuring a DSN-Less Connection on a Non-Windows Machine" in Amazon Redshift ODBC connector installation and configuration guide.

Use the following format on Linux operating systems.
Configuring connections in Amazon Redshift

The following example shows the odbcinst.ini configuration for the 64-bit driver installed in the default directories on Linux operating systems.

```
[ODBC Drivers]
Amazon Redshift (x64)=Installed

[Amazon Redshift (x64)]
Description=Amazon Redshift ODBC Driver (64-bit)
Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
```

The following example shows the odbcinst.ini configuration for the 32-bit driver installed in the default directories on Linux operating systems.

```
[ODBC Drivers]
Amazon Redshift (x86)=Installed

[Amazon Redshift (x86)]
Description=Amazon Redshift ODBC Driver (32-bit)
Driver=/opt/amazon/redshiftodbc/lib/32/libamazonredshiftodbc32.so
```

Use the following format on macOS X operating systems.

```
[ODBC Drivers]
driver_name=Installed

[driver_name]
Description=driver_description
Driver=path/lib/amazonredshiftodbc.dlib
```

The following example shows the odbcinst.ini configuration for the driver installed in the default directory on macOS X operating systems.

```
[ODBC Drivers]
Amazon RedshiftODBC DSN=Installed

[Amazon RedshiftODBC DSN]
Description=Amazon Redshift ODBC Driver for macOS X
Driver=/opt/amazon/redshift/lib/amazonredshiftodbc.dlib
```

Configuring environment variables

Use the correct ODBC driver manager to load the correct driver. To do this, set the library path environment variable. For more information, see "Specifying ODBC Driver Managers on Non-Windows Machines" in Amazon Redshift ODBC connector installation and configuration guide.
By default, ODBC driver managers are configured to use hidden versions of the odbc.ini and odbcinst.ini configuration files (named .odbc.ini and .odbcinst.ini) located in the home directory. They also are configured to use the amazon.redshiftodbc.ini file in the /lib subfolder of the driver installation directory. If you store these configuration files elsewhere, the environment variables so that the driver manager can locate the files. For more information, see "Specifying the Locations of the Driver Configuration Files" in Amazon Redshift ODBC Connector Installation and Configuration Guide.

Configuring connection features

You can configure the following connection features for your ODBC setting:

- Configure the ODBC driver to provide credentials and authenticate the connection to the Amazon Redshift database.
- Configure the ODBC driver to connect to a socket enabled with Secure Sockets Layer (SSL), if you are connecting to an Amazon Redshift server that has SSL enabled.
- Configure the ODBC driver to connect to Amazon Redshift through a proxy server.
- Configure the ODBC driver to use a query processing mode to prevent queries from consuming too much memory.
- Configure the ODBC driver to pass IAM authentication processes through a proxy server.
- Configure the ODBC driver to use TCP keepalives to prevent connections from timing out.

For information about these connection features, see Amazon Redshift ODBC connector installation and configuration guide.

Configure ODBC driver options

You can use configuration options to control the behavior of the Amazon Redshift ODBC driver.

In Microsoft Windows, you typically set driver options when you configure a data source name (DSN). You can also set driver options in the connection string when you connect programmatically, or by adding or changing registry keys in HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBC.INI\your_DSN. For more information about configuring a DSN, see Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180). For an example of setting driver options in a connection string, see Connect to your cluster programmatically (p. 228).

In Linux and macOS X, you set driver configuration options in your odbc.ini and amazon.redshiftodbc.ini files, as described in Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 184). Configuration options set in an amazon.redshiftodbc.ini file apply to all connections. In contrast, configuration options set in an odbc.ini file are specific to a connection. Configuration options set in odbc.ini take precedence over configuration options set in amazon.redshiftodbc.ini.

For information about how to set up ODBC driver configuration options, see Amazon Redshift ODBC connector installation and configuration guide.

Use previous ODBC driver versions in certain cases

Download a previous version of the Amazon Redshift ODBC driver only if your tool requires a specific version of the driver.

For authentication using AWS Identity and Access Management (IAM) credentials or identity provider (IdP) credentials, use Amazon Redshift ODBC driver version 1.3.6.1000 or later.

Important
Amazon Redshift has changed the way that SSL certificates are managed. If you must use a driver version earlier than 1.3.7.1000, you might need to update your current trust root
CA certificates to continue to connect to your clusters using SSL. For more information, see Transitioning to ACM certificates for SSL connections (p. 196).

Use previous ODBC driver versions for Windows

The following are the 64-bit drivers:

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.49.1000/AmazonRedshiftODBC64-1.4.49.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.45.1000/AmazonRedshiftODBC64-1.4.45.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.40.1000/AmazonRedshiftODBC64-1.4.40.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC64-1.4.34.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.30.1000/AmazonRedshiftODBC64-1.4.30.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC64-1.4.27.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.17.1000/AmazonRedshiftODBC64-1.4.17.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.16.1000/AmazonRedshiftODBC64-1.4.16.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.11.1000/AmazonRedshiftODBC64-1.4.11.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.10.1000/AmazonRedshiftODBC64-1.4.10.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.8.1000/AmazonRedshiftODBC64-1.4.8.1000.msi.

The following are the 32-bit drivers:

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC32-1.4.34.1000.msi.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC32-1.4.27.1000.msi.

189
Use previous ODBC driver versions for Linux

The following are the versions of the 64-bit driver:

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.49.1000/AmazonRedshiftODBC-64-bit-1.4.49.1000-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.49.1000/AmazonRedshiftODBC-64-bit-1.4.49.1000-1.x86_64.deb.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.45.1000/AmazonRedshiftODBC-64-bit-1.4.45.1000-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.45.1000/AmazonRedshiftODBC-64-bit-1.4.45.1000-1.x86_64.deb.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.40.1000/AmazonRedshiftODBC-64-bit-1.4.40.1000-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.40.1000/AmazonRedshiftODBC-64-bit-1.4.40.1000-1.x86_64.deb.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC-64-bit-1.4.34.1000-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC-64-bit-1.4.34.1000-1.x86_64.deb.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.30.1000/AmazonRedshiftODBC-64-bit-1.4.30.1000-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.30.1000/AmazonRedshiftODBC-64-bit-1.4.30.1000-1.x86_64.deb.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC-64-bit-1.4.27.1000-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC-64-bit-1.4.27.1000-1.x86_64.deb.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.20.1001/AmazonRedshiftODBC-64-bit-1.4.20.1001-1.x86_64.rpm.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.20.1001/AmazonRedshiftODBC-64-bit-1.4.20.1001-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.18.1001/AmazonRedshiftODBC-64-bit-1.4.18.1001-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.18.1001/AmazonRedshiftODBC-64-bit-1.4.18.1001-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.17.1000/AmazonRedshiftODBC-64-bit-1.4.17.1000-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.17.1000/AmazonRedshiftODBC-64-bit-1.4.17.1000-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.16.1000/AmazonRedshiftODBC-64-bit-1.4.16.1000-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.16.1000/AmazonRedshiftODBC-64-bit-1.4.16.1000-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.13.1000/AmazonRedshiftODBC-64-bit-1.4.13.1000-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.13.1000/AmazonRedshiftODBC-64-bit-1.4.13.1000-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.11.1000/AmazonRedshiftODBC-64-bit-1.4.11.1000-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.11.1000/AmazonRedshiftODBC-64-bit-1.4.11.1000-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.10.1000/AmazonRedshiftODBC-64-bit-1.4.10.1000-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.10.1000/AmazonRedshiftODBC-64-bit-1.4.10.1000-1.x86_64.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.8.1000/AmazonRedshiftODBC-64-bit-1.4.8.1000-1.x86_64.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.8.1000/AmazonRedshiftODBC-64-bit-1.4.8.1000-1.x86_64.deb.

The following are versions of the 32-bit driver:

• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC-32-bit-1.4.34.1000-1.i686.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC-32-bit-1.4.34.1000-1.i686.deb.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC-32-bit-1.4.27.1000-1.i686.rpm.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC-32-bit-1.4.27.1000-1.i686.deb.
Use previous ODBC driver versions for macOS X

The following are the versions of the Amazon Redshift ODBC driver for macOS X:

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.49.1000/AmazonRedshiftODBC-1.4.49.1000.dmg.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.45.1000/AmazonRedshiftODBC-1.4.45.1000.dmg.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.34.1000/AmazonRedshiftODBC-1.4.34.1000.dmg.
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.27.1000/AmazonRedshiftODBC-1.4.27.1000.dmg.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.17.1000/AmazonRedshiftODBC-1.4.17.1000.dmg.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.16.1000/AmazonRedshiftODBC-1.4.16.1000.dmg.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.13.1000/AmazonRedshiftODBC-1.4.13.1000.dmg.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.11.1000/AmazonRedshiftODBC-1.4.11.1000.dmg.
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.10.1000/AmazonRedshiftODBC-1.4.10.1000.dmg.

**Important**
After certificate rotation on September 4, 2020, ODBC Driver version 1.4.8.1000 or earlier on macOS will not be able to establish connections to Amazon Redshift clusters. For more information, see Driver update required for Amazon Redshift ODBC Driver earlier than 1.4.10 on Apple macOS.

Configuring security options for connections

Amazon Redshift supports Secure Sockets Layer (SSL) connections to encrypt data and server certificates to validate the server certificate that the client connects to.

**Connect using SSL**

To support SSL connections, Amazon Redshift creates and installs an AWS Certificate Manager (ACM) issued SSL certificate on each cluster. ACM certificates are publicly trusted by most operating systems, web browsers, and clients. You might need to download a certificate bundle if your SQL clients or applications connect to Amazon Redshift using SSL with the `sslMode` connection option set to `require`, `verify-ca`, or `verify-full`. If your client needs a certificate, Amazon Redshift provides a bundle certificate as follows:

  - The expected MD5 checksum number is 418dea9b6d5d5de7a8f1ac42e164cdcf.
  - The sha256 checksum number is 36dba8eb041cd14b9d6015893963301bcb92e1c456847784de2acb5bd550.

  Don't use the previous certificate bundle that was located at https://s3.amazonaws.com/redshift-downloads/redshift-ca-bundle.crt.

  - The expected MD5 checksum number is 418dea9b6d5d5de7a8f1ac42e164cdcf.
  - The sha256 checksum number is 36dba8eb041cd14b9d6015893963301bcb92e1c456847784de2acb5bd550.

  Don't use the previous certificate bundles that were located at https://s3-cn-north-1.amazonaws.com.cn/redshift-downloads-cn/redshift-ca-bundle.crt and
By default, cluster databases accept a connection whether it uses SSL or not. To configure your cluster to require an SSL connection, set the `require_SSL` parameter to `true` in the parameter group that is associated with the cluster.

Amazon Redshift supports an SSL mode that is compliant with Federal Information Processing Standard (FIPS) 140-2. FIPS-compliant SSL mode is disabled by default.

Important
Enable FIPS-compliant SSL mode only if your system is required to be FIPS-compliant.

To enable FIPS-compliant SSL mode, set both the `use_fips_ssl` parameter and the `require_SSL` parameter to `true` in the parameter group that is associated with the cluster. For information about modifying a parameter group, see Amazon Redshift parameter groups (p. 271).

Amazon Redshift supports the Elliptic Curve Diffie—Hellman Ephemeral (ECDHE) key agreement protocol. With ECDHE, the client and server each have an elliptic curve public-private key pair that is used to establish a shared secret over an insecure channel. You don't need to configure anything in Amazon Redshift to enable ECDHE. If you connect from a SQL client tool that uses ECDHE to encrypt communication between the client and server, Amazon Redshift uses the provided cipher list to make the appropriate connection. For more information, see Elliptic curve diffie—hellman on Wikipedia and Ciphers on the OpenSSL website.

Using SSL and trust CA certificates in ODBC

If you connect using the latest Amazon Redshift ODBC drivers (version 1.3.7.1000 or later), you can skip this section. To download the latest drivers, see Configuring an ODBC connection (p. 179).

You might need to update your current trust root CA certificates to continue to connect to your clusters using SSL. For more information, see Connect using SSL (p. 193).

You can verify that the certificate that you downloaded matches the expected MD5 checksum number. To do this, you can use the `Md5sum` program on Linux operating systems, or another tool on Windows and macOS X operating systems.

ODBC DSNs contain an `sslmode` setting that determines how to handle encryption for client connections and server certificate verification. Amazon Redshift supports the following `sslmode` values from the client connection:

- **disable**
  
  SSL is disabled and the connection is not encrypted.

- **allow**
  
  SSL is used if the server requires it.

- **prefer**
  
  SSL is used if the server supports it. Amazon Redshift supports SSL, so SSL is used when you set `sslmode` to `prefer`.

- **require**
SSL is required.
  • verify-ca

SSL must be used and the server certificate must be verified.
  • verify-full

SSL must be used. The server certificate must be verified and the server hostname must match the hostname attribute on the certificate.

You can determine whether SSL is used and server certificates are verified in a connection between the client and the server. To do this, you need to review the sslmode setting for your ODBC DSN on the client and the require_SSL setting for the Amazon Redshift cluster on the server. The following table describes the encryption result for the various client and server setting combinations:

<table>
<thead>
<tr>
<th>Sslmode (client)</th>
<th>require_SSL (server)</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>disable</td>
<td>false</td>
<td>The connection is not encrypted.</td>
</tr>
<tr>
<td>disable</td>
<td>true</td>
<td>The connection can't be made because the server requires SSL and the client has SSL disabled for the connection.</td>
</tr>
<tr>
<td>allow</td>
<td>true</td>
<td>The connection is encrypted.</td>
</tr>
<tr>
<td>allow</td>
<td>false</td>
<td>The connection is not encrypted.</td>
</tr>
<tr>
<td>prefer or require</td>
<td>true</td>
<td>The connection is encrypted.</td>
</tr>
<tr>
<td>prefer or require</td>
<td>false</td>
<td>The connection is encrypted.</td>
</tr>
<tr>
<td>verify-ca</td>
<td>true</td>
<td>The connection is encrypted and the server certificate is verified.</td>
</tr>
<tr>
<td>verify-ca</td>
<td>false</td>
<td>The connection is encrypted and the server certificate is verified.</td>
</tr>
<tr>
<td>verify-full</td>
<td>true</td>
<td>The connection is encrypted and the server certificate and hostname are verified.</td>
</tr>
<tr>
<td>verify-full</td>
<td>false</td>
<td>The connection is encrypted and the server certificate and hostname are verified.</td>
</tr>
</tbody>
</table>

**Connect using the server certificate with ODBC on Microsoft Windows**

If you want to connect to your cluster using SSL and the server certificate, first download the certificate to your client computer or Amazon EC2 instance. Then configure the ODBC DSN.

1. Download the Amazon Redshift certificate authority bundle to your client computer at the `lib` folder in your driver installation directory, and save the file as `root.crt`. For download information, see Connect using SSL (p. 193).
2. Open ODBC Data Source Administrator, and add or edit the system DSN entry for your ODBC connection. For SSL Mode, select verify-full unless you use a DNS alias. If you use a DNS alias, select verify-ca. Then choose Save.

For more information about configuring the ODBC DSN, see Configuring an ODBC connection (p. 179).
Using SSL and server certificates in Java

SSL provides one layer of security by encrypting data that moves between your client and cluster. Using a server certificate provides an extra layer of security by validating that the cluster is an Amazon Redshift cluster. It does so by checking the server certificate that is automatically installed on all clusters that you provision. For more information about using server certificates with JDBC, go to Configuring the client in the PostgreSQL documentation.

Connect using trust CA certificates in Java

If you connect using the latest Amazon Redshift JDBC drivers (version 1.2.8.1005 or later), you can skip this section. To download the latest drivers, see Configure a JDBC connection.

**Important**

Amazon Redshift has changed the way that SSL certificates are managed. You might need to update your current trust root CA certificates to continue to connect to your clusters using SSL. For more information, see Connect using SSL (p. 193).

To connect using trust CA certificates

You can use the `redshift-keytool.jar` file to import CA certificates in the Amazon Redshift Certificate Authority bundle into a Java TrustStore or your private TrustStore.

1. If you use the Java command line `-Djavax.net.ssl.trustStore` option, remove it from command line, if possible.
2. Download `redshift-keytool.jar`.
3. Do one of the following:
   - To import the Amazon Redshift Certificate Authority bundle into a Java TrustStore, run the following command:

   ```
   java -jar redshift-keytool.jar -s
   ```
   - To import the Amazon Redshift Certificate Authority bundle into your private TrustStore, run the following command:

   ```
   java -jar redshift-keytool.jar -k <your_private_trust_store> -p <keystore_password>
   ```

Transitioning to ACM certificates for SSL connections

Amazon Redshift is replacing the SSL certificates on your clusters with AWS Certificate Manager (ACM) issued certificates. ACM is a trusted public certificate authority (CA) that is trusted by most current systems. You might need to update your current trust root CA certificates to continue to connect to your clusters using SSL.

This change affects you only if all of the following apply:

- Your SQL clients or applications connect to Amazon Redshift clusters using SSL with the `sslMode` connection option set to `require`, `verify-ca`, or `verify-full` configuration option.
- You aren't using the Amazon Redshift ODBC or JDBC drivers, or you use Amazon Redshift drivers before ODBC version 1.3.7.1000 or JDBC version 1.2.8.1005.

If this change affects you on commercial Amazon Redshift Regions, then you must update your current trust root CA certificates before October 23, 2017. Amazon Redshift will transition your clusters to use
ACM certificates between now and October 23, 2017. The change should have very little or no effect on your cluster's performance or availability.

If this change affects you on AWS GovCloud (US) (US) Regions, then you must update your current trust root CA certificates before April 1, 2020 to avoid service interruption. Beginning on this date, clients connecting to Amazon Redshift clusters using SSL encrypted connections need an additional trusted certificate authority (CA). Clients use trusted certificate authorities to confirm the identity of the Amazon Redshift cluster when they connect to it. Your action is required to update your SQL clients and applications to use an updated certificate bundle that includes the new trusted CA.

**Important**
In the China Regions on January 5, 2021, Amazon Redshift is replacing the SSL certificates on your clusters with AWS Certificate Manager (ACM) issued certificates. If this change affects you on China (Beijing) Region or China (Ningxia) Region, then you must update your current trust root CA certificates before January 5, 2021 to avoid service interruption. Beginning on this date, clients connecting to Amazon Redshift clusters using SSL encrypted connections need an additional trusted certificate authority (CA). Clients use trusted certificate authorities to confirm the identity of the Amazon Redshift cluster when they connect to it. Your action is required to update your SQL clients and applications to use an updated certificate bundle that includes the new trusted CA.

- Using the latest Amazon Redshift ODBC or JDBC drivers (p. 197)
- Using earlier Amazon Redshift ODBC or JDBC drivers (p. 197)
- Using other SSL connection types (p. 198)

### Using the latest Amazon Redshift ODBC or JDBC drivers

The preferred method is to use the latest Amazon Redshift ODBC or JDBC drivers. Amazon Redshift drivers beginning with ODBC version 1.3.7.1000 and JDBC version 1.2.8.1005 automatically manage the transition from an Amazon Redshift self-signed certificate to an ACM certificate. To download the latest drivers, see Configuring an ODBC connection (p. 179) or Configuring a JDBC driver version 1.x connection (p. 170).

If you use the latest Amazon Redshift JDBC driver, it's best not to use `-Djavax.net.ssl.trustStore` in JVM options. If you must use `-Djavax.net.ssl.trustStore`, import the Redshift certificate authority bundle into the truststore it points to. For download information, see Connect using SSL (p. 193). For more information, see Importing the Amazon Redshift certificate authority bundle into a TrustStore (p. 198).

### Using earlier Amazon Redshift ODBC or JDBC drivers

- If your ODBC DSN is configured with `SSLCertPath`, overwrite the certificate file in the specified path.
- If `SSLCertPath` is not set, then overwrite the certificate file named `root.crt` in the driver DLL location.

If you must use an Amazon Redshift JDBC driver before version 1.2.8.1005, then do one of the following:

- If your JDBC connection string uses the `sslCert` option, remove the `sslCert` option. Then import the Redshift certificate authority bundle to your Java TrustStore. For download information, see Connect using SSL (p. 193). For more information, see Importing the Amazon Redshift certificate authority bundle into a TrustStore (p. 198).
- If you use the Java command line `-Djavax.net.ssl.trustStore` option, remove it from command line, if possible. Then import the Redshift certificate authority bundle to your Java TrustStore. For download information, see Connect using SSL (p. 193). For more information, see Importing the Amazon Redshift certificate authority bundle into a TrustStore (p. 198).
**Importing the Amazon Redshift certificate authority bundle into a TrustStore**

You can use `redshift-keytool.jar` to import CA certificates in the Amazon Redshift Certificate Authority bundle into a Java TrustStore or your private truststore.

**To import the Amazon Redshift certificate authority bundle into a TrustStore**

1. Download `redshift-keytool.jar`.
2. Do one of the following:
   - To import the Amazon Redshift Certificate Authority bundle into a Java TrustStore, run the following command:
     
```
java -jar redshift-keytool.jar -s
```
   - To import the Amazon Redshift Certificate Authority bundle into your private TrustStore, run the following command:
     
```
java -jar redshift-keytool.jar -k <your_private_trust_store> -p <keystore_password>
```

**Using other SSL connection types**

Follow the steps in this section if you connect using any of the following:

- Open source ODBC driver
- Open source JDBC driver
- The `psql` command line interface
- Any language bindings based on `libpq`, such as `psycopg2` (Python) and `ruby-pg` (Ruby)

**To use ACM certificates with other SSL connection types:**

1. Download the Amazon Redshift certificate authority bundle. For download information, see Connect using SSL (p. 193).
2. Place the certificates from the bundle in your `root.crt` file.
   - On Linux and macOS X operating systems, the file is `~/.postgresql/root.crt`.
   - On Microsoft Windows, the file is `%APPDATA%\postgresql\root.crt`.

**Connecting to clusters from client tools and code**

This section provides some options for third-party tools to connect to the cluster if you do not already have a business intelligence tool to do so. Additionally, it describes how to connect to your cluster programmatically.

**Topics**

- Amazon Redshift RSQL (p. 199)
- Connect to your cluster by using SQL Workbench/J (p. 221)
- Connect to your cluster by using the `psql` tool (p. 224)
- Connect to your cluster programmatically (p. 228)
- Using an authentication profile to connect to Amazon Redshift (p. 231)
Amazon Redshift RSQL

Amazon Redshift RSQL is a command-line client for interacting with Amazon Redshift clusters and databases. You can connect to an Amazon Redshift cluster, describe database objects, query data, and view query results in various output formats.

Amazon Redshift RSQL supports the capabilities of the PostgreSQL psql command-line tool with an additional set of capabilities specific to Amazon Redshift. These include the following:

- You can use single sign-on (SSO) authentication using ADFS, PingIdentity, Okta, Azure ADm or other SAML/JWT based identity providers. You can also use browser-based SAML identity providers for multi-factor authentication (MFA).
- You can describe properties or attributes of Amazon Redshift objects such as table distribution keys, table sort keys, late-binding views (LBVs), and materialized views. You can also describe properties or attributes of external tables in an AWS Glue catalog or Apache Hive Metastore, external databases in Amazon RDS for PostgreSQL, Amazon Aurora PostgreSQL-Compatible Edition, RDS for MySQL (preview) and Amazon Aurora MySQL-Compatible Edition (preview), and tables shared by using Amazon Redshift data sharing.
- You can also use enhanced control flow commands such as IF (ELSEIF, ELSE, ENDIF), GOTO and LABEL.

With Amazon Redshift RSQL batch mode, which runs a script passed as an input parameter, you can run scripts that include both SQL and complex business logic. If you have existing self-managed, on-premises data warehouses, you can use Amazon Redshift RSQL to replace existing extract, transfer, load (ETL) and automation scripts, such as Teradata BTEQ scripts. Using RSQL helps you to avoid manually reimplementing scripts in a procedural language.

Amazon Redshift RSQL is available for Linux, Windows, and macOS X operating systems.

To report issues for Amazon Redshift RSQL, write to <redshift-rsql-support@amazon.com>.

Getting started with Amazon Redshift RSQL

Install Amazon Redshift RSQL on a computer with a Linux, macOS, or Microsoft Windows operating system.

Prerequisites

Linux:

1. Install the driver manager with the following command:

```
sudo yum install unixODBC openssl
```

OpenSSL is required for Linux distributions. The OpenSSL library is located in the Linux OpenSSL Github repository. For more information about OpenSSL, see OpenSSL.

2. Install the ODBC driver: Installing the Amazon Redshift driver on Linux operating systems (p. 182).

3. Copy the ini file to your home directory:

```
cp /opt/amazon/redshiftodbc/Setup/odbc.ini ~/.odbc.ini
```

4. Set the environment variables to point to the location of the file:

```
export ODBCINI=~/.odbc.ini
export ODBCSYSINI=/opt/amazon/redshiftodbc/Setup
export AMAZONREDSHIFTODBCINI=/opt/amazon/redshiftodbc/lib/64/amazon.redshiftodbc.ini
```
Connecting to clusters from client tools and code

For more information about configuring environment variables, see Configuring environment variables (p. 187).

Mac OS:

1. Install the driver manager with the following command:

   ```bash
   brew install unixodbc openssl --build-from-source
   ```

2. Install the ODBC driver: Install the Amazon Redshift ODBC driver on macOS X (p. 183).

3. Copy the ini file to your home directory:

   ```bash
   cp /opt/amazon/redshift/Setup/odbc.ini ~/.odbc.ini
   ```

4. Set the environment variables to point to the location of the file:

   ```bash
   export ODBCINI=~/.odbc.ini
   export ODBCSYSINI=/opt/amazon/redshift/Setup
   export AMAZONREDSHIFTODBCINI=/opt/amazon/redshift/lib/amazon.redshiftodbc.ini
   ```

   For more information about configuring environment variables, see Configuring environment variables (p. 187).

5. Set `DYLD_LIBRARY_PATH` to location of your libodbc.dylib if its not in /usr/local/lib.

   ```bash
   export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/usr/local/lib
   ```

Windows:

Follow the instructions at Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180) to install the driver. Windows doesn't require a driver manager.

OpenSSL is required for Amazon Redshift RSQL on Windows. The Windows OpenSSL library is located in the Windows OpenSSL GitHub repository. For more information about OpenSSL, see OpenSSL.

Download RSQL

- Linux 64-bit RPM: RSQL Version 1.0.4
- Mac OS 64-bit DMG: RSQL Version 1.0.4
- Windows 64-bit MSI: RSQL Version 1.0.4

See the change log and downloads for previous versions at Amazon Redshift RSQL change log (p. 201).

Install RSQL

Perform the steps for your operating system.

- **Linux:**

  ```bash
  sudo rpm -i AmazonRedshiftRsql-<version>-1.x86_64.rpm
  ```

- **Mac OS:**

  1. Double-click the dmg file to mount the disk image.
  2. Double-click the pkg file to run the installer.
3. Follow the steps in the installer to complete the installation. Agree to the terms of the license agreement.

- **Windows:**
  1. Double-click the file to run the installer.
  2. Follow the prompts to complete the installation.

**Amazon Redshift RSQL change log**

1.0.4 (2022-03-19)

- Add support for RSPASSWORD environment variable. Set a password to connect to Amazon Redshift. For example, `export RSPASSWORD=TestPassword`.

1.0.3 (2021-12-08)

**Bug Fixes**

- Fixed dialogue pop up when using `\c` or `\logon` to switch between databases in Windows OS.
- Fixed crash when checking ssl information.

1.0.2 (2021-11-26)

**Bug Fixes**

- Fixed issue with prompt for superuser when using IAM authentication.
- Fixed incorrect 'Failed' status message when running queries that return no rows.
- Fixed crash when using SingleRowMode in DSN file.

**New**

- Added DSN connection option to the help content.

**Amazon Redshift RSQL previous versions**

Choose one of the links to download the version of Amazon Redshift RSQL you need, based on your operating system.

**Linux 64-bit RPM**

- RSQl Version 1.0.3
- RSQl Version 1.0.2
- RSQl Version 1.0.1

**Mac OS 64-bit DMG**

- RSQl Version 1.0.3
- RSQl Version 1.0.2
- RSQl Version 1.0.1

**Windows 64-bit MSI**
• RSQl Version 1.0.3
• RSQl Version 1.0.2
• RSQl Version 1.0.1

Connect to a cluster with Amazon Redshift RSQl

Connecting without a DSN

1. On the Amazon Redshift console, choose the cluster you want to connect to and note the endpoint, database, and port.
2. At a command prompt, specify the connection information by using command-line parameters.

```
rsql -h <endpoint> -U <username> -d <databasename> -p <port>
```

Here, the following apply:

• <endpoint> is the Endpoint you recorded in the previous step.
• <username> is the name of a user with permissions to connect to the cluster.
• <databasename> is the Database Name you recorded in the previous step.
• <port> is the Port you recorded in the previous step. <port> is an optional parameter which has a default value of 5439.

An example follows.

```
rsql -h testcluster.example.amazonaws.com -U user1 -d dev -p 5439
```

3. At the password prompt, enter the password for the <username> user.

A successful connection response looks like the following.

```
% rsql -h testcluster.example.com -d dev -U user1 -p 5349
Password for user user1:
DSN-less Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.
(testcluster) user1@dev=#
```

The command to connect has the same parameters on Linux, Mac OS, and Windows.

Connecting using a DSN

You can connect RSQl to Amazon Redshift by using a data source name (DSN) to simplify the organization of connection properties. For more information, see Configuring connection features (p. 188). This topic includes instructions for ODBC-driver installation and descriptions for DSN properties.

Using a DSN connection with a password

The following shows an example of a DSN-connection configuration that uses a password.
The default <path to driver> for Mac OSX is /opt/amazon/redshift/lib/
libamazonredshiftodbc.dylib and for Linux is /opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so.

```
[testuser]
Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
SSLMode=verify-ca
Min_TLS=1.2
boolaschar=0
Host=<server endpoint>
Port=<database port>
Database=<dbname>
UID=<username>
PWD=<password>
sslmode=prefer
```

The following output results from a successful connection.

```
% rsql -D testuser
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.
(testcluster) user1@dev=#
```

Using Single sign-on DSN

You can configure a DSN for single sign-on (SSO) authentication. The following shows an example of a DSN-connection configuration that uses Okta SSO.

```
[testokta]
Driver=<path to driver>
SSLMode=verify-ca
Min_TLS=1.2
boolaschar=0
Host=<server endpoint>
clusterid=<cluster id>
region=<region name>
Database=<dbname>
locale=en-US
iam=1
plugin_name=<plugin name>
uid=<okta username>
pwd=<okta password>
idp_host=<idp endpoint>
app_id=<app id>
app_name=<app name>
preferred_role=<role arn>
```

Sample output from a successful connection.

```
% rsql -D testokta
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
```
The following example shows an example of a DSN-connection configuration that uses Azure SSO.

```
[testazure]
Driver=<path to driver>
SSLMode=verify-ca
Min_TLS=1.2
boolaschar=0
Host=<server endpoint>
Port=<cluster port>
clusterid=<cluster id>
region=<region name>
Database=<dbname>
locale=en-us
iam=1
plugin_name=<plugin name>
uid=<azure username>
pwd=<azure password>
idp_tenant=<Azure idp tenant uuid>
client_id=<Azure idp client uuid>
client_secret=<Azure idp client secret>
```

### Using a DSN connection with an IAM profile

You can connect to Amazon Redshift using your configured IAM profile. The IAM profile must have privileges to call `GetClusterCredentials`. The following example shows the DSN properties to use. The `ClusterID` and `Region` parameters are required only if the `Host` is not an Amazon provided endpoint like `examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com`.

```
[testiam]
Driver=Default
Host=testcluster.example.com
Database=dev
DbUser=testuser
ClusterID=rsqltestcluster
Region=us-east-1
IAM=1
Profile=default
```

The value for the `Profile` key is the named profile you choose from your AWS CLI credentials. This example shows the credentials for the profile named `default`.

```
$ cat .aws/credentials
[default]
aws_access_key_id = ASIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUttnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

The following shows the connection response.

```
$ rsql -D testiam
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.
```
Using a DSN connection with an Instance profile

You can connect to Amazon Redshift using your Amazon EC2 instance profile. The instance profile must have privileges to call GetClusterCredentials. See example below for the DSN properties to use. The ClusterID and Region parameters are required only if the Host is not an Amazon provided endpoint like examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com.

```
[testinstanceprofile]
Driver=Default
Host=testcluster.example.com
Database=dev
DbUser=testuser
ClusterID=rsqltestcluster
Region=us-east-1
IAM=1
Instanceprofile=1
```

The following shows the connection response.

```
$ rsql -D testinstanceprofile
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.27.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.
(testcluster) testuser@dev=>
```

Using a DSN connection with the default credential provider chain

To connect using the default credential provider chain, specify only the IAM property, and Amazon Redshift RSQL will attempt to acquire credentials in the order described in Working with AWS Credentials in the AWS SDK for Java. At least one of the providers in the chain must have GetClusterCredentials permission. This is useful for connecting from ECS containers, for example.

```
[iamcredentials]
Driver=Default
Host=testcluster.example.com
Database=dev
DbUser=testuser
ClusterID=rsqltestcluster
Region=us-east-1
IAM=1
```

Amazon Redshift RSQL meta commands

Amazon Redshift RSQL meta commands return informational records about databases or specific database objects. Results can include various columns and metadata. Other commands perform specific actions. These commands are preceded with a backslash.

```
\d[5+]
```

Lists local user created tables, regular views, late-binding views and materialized views. `\dS` also lists tables and views, like `\d`, but system objects are included in the returned records. The `+` results in the
additional metadata column description for all listed objects. The following shows sample records returned as a result of the command.

<table>
<thead>
<tr>
<th>List of relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>schema</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>public</td>
</tr>
</tbody>
</table>

(7 rows)

\d[S+] NAME

Describes a table, view, or index. Includes the column names and types. It also provides the diststyle, backup configuration, create date (tables created after October 2018), and constraints. For example, \dS+ sample returns object properties. Appending S+ results in additional columns included in the returned records.

<table>
<thead>
<tr>
<th>Table &quot;public.sample&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>Encoding</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>t</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

Diststyle: KEY
Backup: YES
Created: 2021-07-20 19:47:27.997045
Unique Constraints:
- "sample_pkey" PRIMARY KEY (col1)
- "sample_col2_key" UNIQUE (col2)
Foreign-key constraints:
- "sample_col2_fkey" FOREIGN KEY (col2) REFERENCES lineitem(l_orderkey)

The distribution style, or Diststyle, of the table can be KEY, AUTO, EVEN or ALL.

Backup indicates if the table is backed up when a snapshot is taken. Valid values are YES or NO.

Created is the timestamp for when the table is created. The creation date isn't available for Amazon Redshift tables created before November 2018. Tables created before this date display n/a (Not Available).

Unique Constraints lists unique and primary key constraints on the table.

Foreign-key constraints lists foreign-key constraints on the table.
Lists casts. Includes the source type, target type, and whether the cast is implicit.

The following shows a subset of results from `\dC+`.

<table>
<thead>
<tr>
<th>source type</th>
<th>target type</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>implicit?</td>
<td>description</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>&quot;char&quot;</td>
<td>character</td>
<td>bpchar</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>&quot;char&quot;</td>
<td>character varying</td>
<td>text</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>&quot;char&quot;</td>
<td>integer</td>
<td>int4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>&quot;char&quot;</td>
<td>text</td>
<td>text</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>&quot;path&quot;</td>
<td>point</td>
<td>point</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>&quot;path&quot;</td>
<td>polygon</td>
<td>polygon</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>abstime</td>
<td>date</td>
<td>date</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>abstime</td>
<td>integer</td>
<td>(binary coercible)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>abstime</td>
<td>time without time zone</td>
<td>time</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>abstime</td>
<td>timestamp with time zone</td>
<td>timestamp with time zone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>abstime</td>
<td>timestamp without time zone</td>
<td>timestamp without time zone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>bit</td>
<td>bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>boolean</td>
<td>bool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>character</td>
<td>bpchar</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>character varying</td>
<td>text</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>double precision</td>
<td>float8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>integer</td>
<td>int4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>numeric</td>
<td>numeric</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>oid</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>real</td>
<td>float4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>regclass</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>regoper</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>regoperator</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>regproc</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>regprocedure</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>regtype</td>
<td>oid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>smallint</td>
<td>int2</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td>assignment</td>
</tr>
<tr>
<td>bigint</td>
<td>super</td>
<td>int@partiql</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>assignment</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\dd[S] [PATTERN]

Shows object descriptions not displayed elsewhere.

\de

Lists external tables. This includes tables in the AWS Glue data catalog, Hive Metastore and federated tables from Amazon RDS/Aurora MySQL, Amazon RDS/Aurora PostgreSQL and Amazon Redshift datashare tables.

\de NAME

Describes an external table.

The following example shows an AWS Glue external table.

```
\de spectrum.lineitem

Glue External table "spectrum.lineitem"

<table>
<thead>
<tr>
<th>Column</th>
<th>External Type</th>
<th>Redshift Type</th>
<th>Position</th>
<th>Partition Key</th>
<th>Nullable</th>
</tr>
</thead>
<tbody>
<tr>
<td>l_orderkey</td>
<td>bigint</td>
<td>bigint</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_partkey</td>
<td>bigint</td>
<td>bigint</td>
<td>2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_suppkey</td>
<td>int</td>
<td>int</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_linenumber</td>
<td>int</td>
<td>int</td>
<td>4</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_quantity</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_extendedprice</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>6</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_discount</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>7</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_tax</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>8</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_returnflag</td>
<td>char(1)</td>
<td>char(1)</td>
<td>9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_linenumber</td>
<td>char(1)</td>
<td>char(1)</td>
<td>10</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_commitdate</td>
<td>date</td>
<td>date</td>
<td>11</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_receiptdate</td>
<td>date</td>
<td>date</td>
<td>12</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_shipdate</td>
<td>date</td>
<td>date</td>
<td>13</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_shipinstruct</td>
<td>char(25)</td>
<td>char(25)</td>
<td>14</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_shipmode</td>
<td>char(10)</td>
<td>char(10)</td>
<td>15</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_comment</td>
<td>varchar(44)</td>
<td>varchar(44)</td>
<td>16</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Location: s3://redshiftbucket/kfhose2019/12/31
Input_format: org.apache.hadoop.mapred.TextInputFormat
Output_format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Serialization_lib: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Serde_parameters: {"field.delim":"|","serialization.format":"|"}
Parameters:
{"EXTERNAL":"TRUE","numRows":"178196721475","transient_lastDdlTime":"1577771873"}
```

A Hive Metastore table.

```
\de emr.lineitem

Hive Metastore External Table "emr.lineitem"

<table>
<thead>
<tr>
<th>Column</th>
<th>External Type</th>
<th>Redshift Type</th>
<th>Position</th>
<th>Partition Key</th>
<th>Nullable</th>
</tr>
</thead>
<tbody>
<tr>
<td>l_orderkey</td>
<td>bigint</td>
<td>bigint</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_partkey</td>
<td>bigint</td>
<td>bigint</td>
<td>2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_suppkey</td>
<td>int</td>
<td>int</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_linenumber</td>
<td>int</td>
<td>int</td>
<td>4</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_quantity</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_extendedprice</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>6</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_shipdate</td>
<td>date</td>
<td>date</td>
<td>7</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_shipinstruct</td>
<td>char(25)</td>
<td>char(25)</td>
<td>8</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_shipmode</td>
<td>char(10)</td>
<td>char(10)</td>
<td>9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>l_comment</td>
<td>varchar(44)</td>
<td>varchar(44)</td>
<td>10</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
```

208
Amazon Redshift Cluster Management Guide
Connecting to clusters from client tools and code

```
<table>
<thead>
<tr>
<th>Column</th>
<th>External Type</th>
<th>Redshift Type</th>
<th>Position</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>l_discount</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>l_tax</td>
<td>decimal(12,2)</td>
<td>decimal(12,2)</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>l_returnflag</td>
<td>char(1)</td>
<td>char(1)</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>l_linestatus</td>
<td>char(1)</td>
<td>char(1)</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>l_commitdate</td>
<td>date</td>
<td>date</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>l_receiptdate</td>
<td>date</td>
<td>date</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>l_shipinstruct</td>
<td>char(25)</td>
<td>char(25)</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>l_shipmode</td>
<td>char(10)</td>
<td>char(10)</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>l_comment</td>
<td>varchar(44)</td>
<td>varchar(44)</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>l_shipdate</td>
<td>date</td>
<td>date</td>
<td>16</td>
<td>1</td>
</tr>
</tbody>
</table>
```

Location: s3://redshiftbucket/cetas
Serde_parameters: {"serialization.format":"1"}
Parameters: {"EXTERNAL":"TRUE", "numRows":"4307207", "transient_lastDdlTime":"1626990007"}

PostgreSQL external table.

```
\# \de pgrsql.alltypes

<table>
<thead>
<tr>
<th>Column</th>
<th>External Type</th>
<th>Redshift Type</th>
<th>Position</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>col1</td>
<td>bigint</td>
<td>bigint</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>col2</td>
<td>bigint</td>
<td>bigint</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>col5</td>
<td>boolean</td>
<td>boolean</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>col6</td>
<td>box</td>
<td>varchar(65535)</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>col7</td>
<td>bytea</td>
<td>varchar(65535)</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>col8</td>
<td>character(10)</td>
<td>character(10)</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>col9</td>
<td>character varying(10)</td>
<td>character varying(10)</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>col10</td>
<td>cidr</td>
<td>varchar(65535)</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>col11</td>
<td>circle</td>
<td>varchar(65535)</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>col12</td>
<td>date</td>
<td>date</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>col13</td>
<td>double precision</td>
<td>double precision</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>col14</td>
<td>inet</td>
<td>varchar(65535)</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>col15</td>
<td>integer</td>
<td>integer</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>col16</td>
<td>interval</td>
<td>varchar(65535)</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>col17</td>
<td>json</td>
<td>varchar(65535)</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>col18</td>
<td>jsonb</td>
<td>varchar(65535)</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>col19</td>
<td>line</td>
<td>varchar(65535)</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>col20</td>
<td>lseg</td>
<td>varchar(65535)</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td>col21</td>
<td>macaddr</td>
<td>varchar(65535)</td>
<td>19</td>
<td>0</td>
</tr>
</tbody>
</table>
```
Lists functions of various types. The command `\df`, for example, returns a list of functions. Results include properties like name, data-type returned, access privileges, and additional metadata. Function types can include triggers, stored procedures, window functions and other types. When you append `S` to the command, for example `\dfantS`, additional metadata columns are included, such as `owner`, `security`, and `access privileges`.

Lists data about procedural languages associated with the database. Information includes the name, such as `plpgsql`, and additional metadata, which includes whether it is trusted, access privileges, and description. Sample call is, for example, `\dL`, which lists languages and their properties. When you append `S` to the command, additional metadata columns are included, such as `call handler` and `access privileges`.

Sample results:

```
List of languages
```
<table>
<thead>
<tr>
<th>name</th>
<th>trusted</th>
<th>internal language</th>
<th>call handler</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+------------+---------+-------------------+---------------------+------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>f</td>
<td>t</td>
<td>fmgr_c_validator(oid)</td>
<td>-</td>
</tr>
<tr>
<td>exfunc</td>
<td>f</td>
<td>f</td>
<td>exfunc_call_handler()</td>
<td>-</td>
</tr>
<tr>
<td>internal</td>
<td>f</td>
<td>t</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>plpython_u</td>
<td>f</td>
<td>f</td>
<td>plpython_call_handler()</td>
<td>-</td>
</tr>
</tbody>
</table>

| rdsdb=U/rdsdb | rdsdb=U/rdsdb | rdsdb=U/rdsdb | rdsdb=U/rdsdb |
|-----------------------------------------------|
| Built-in functions                          |
|                      |

\**dm*[S+] [PATTERN]**

Lists materialized views. For example, \*dm*[S+] lists materialized views and their properties. When you append S+ to the command, additional metadata columns are included.

\**dn*[S+] [PATTERN]**

Lists schemas. When you append S+ to the command, for example \*dn*[S+], additional metadata columns are included, such as description and access privileges.

\**dp** [PATTERN]

Lists table, view, and sequence access privileges.

\**dt*[S+] [PATTERN]**

Lists tables. When you append S+ to the command, for example \*dt*[S+], additional metadata columns are included, such as description in this case.

\**du**

Lists the users for the database. Includes their name and their roles, such as Superuser, and attributes.

\**dv*[S+] [PATTERN]**

Lists views. Includes schema, type, and owner data. When you append S+ to the command, for example \*dv*[S+], additional metadata columns are included.

\**H**

Turns on HTML output. This is useful to quickly return formatted results. For example, select * from sales; \*H returns results from the sales table, in HTML. To switch back to tabular results, use \*q, or quiet.

\**i**

Runs commands from a file. For example, assuming you have rsql_steps.sql in your working directory, the following runs the commands in the file: \*i rsql_steps.sql.
Lists databases. Includes owner, encoding, and additional information.

The quit, or \q command, logs off database sessions and exits RSQl.

Shows a view's definition.

Shows the run time, for a query, for instance.

The same output as \dp.

Shows help information. The optional parameter specifies the item to explain.

Logs off all database sessions and exits Amazon Redshift RSQl. In addition, you can specify an optional exit code. For example, \EXIT 15 will exit the Amazon Redshift RSQl terminal and return exit code 15.

The following example shows output from a connection and exit from RSQl.

```
% rsql -D testuser
DSN Connected
DBMS Name: Amazon Redshift
Driver Name: Amazon Redshift ODBC Driver
Driver Version: 1.4.34.1000
Rsql Version: 1.0.1
Redshift Version: 1.0.29306
Type "help" for help.

(testcluster) user1@dev=# \exit 15

% echo $?
15
```

Specifies the name of an export file that RSQl uses to store database information returned by a subsequent SQL SELECT statement.

```
\EXPORT

\export report file='E:\\accounts.out'
\rset rformat off
\rset width 1500
\rset heading "General Title"
\rset titledashes on
select * from td_dwh.accounts;
\export reset
```

Console output

Rformat is off.
\LOGON

Connects to a database. You can specify connection parameters using positional syntax or as a connection string.

Command syntax is the following: \logon {[DBNAME|USERNAME|HOST|PORT| [PASSWORD]] | conninfo}

The DBNAME is the name of the database to connect to. The USERNAME is the user name to connect as. The default HOST is localhost. The default PORT is 5439.

When a host name is specified in a \LOGON command, it becomes the default host name for additional \LOGON commands. To change the default host name, specify a new HOST in an additional \LOGON command.

Sample output from the \LOGON command for user1 follows.

(testcluster) user1@redshiftdb=# \logon dev  
DBMS Name: Amazon Redshift  
Driver Name: Amazon Redshift ODBC Driver  
Driver Version: 1.4.27.1000  
Rsql Version: 1.0.1  
You are now connected to database "dev" as user "user1".  
(testcluster) user1@dev=# 

Sample output for user2.

(testcluster) user1@dev=# \logon dev user2 testcluster2.example.com  
Password for user user2:  
DBMS Name: Amazon Redshift  
Driver Name: Amazon Redshift ODBC Driver  
Driver Version: 1.4.27.1000  
Rsql Version: 1.0.1  
You are now connected to database "dev" as user "user2" on host "testcluster2.example.com" at port "5439".  
(testcluster2) user2@dev=# 

\REMARK

An extension of the \echo command. \REMARK prints the specified string to the output stream. \REMARK extends \echo by adding the ability to break the output over separate lines.

The following sample shows output from the command.

(testcluster) user1@dev=# \remark 'hello//world'  
hello  
world  

\RSET

The command \rset sets command parameters and variables. \rset has both an interactive and a batch mode. It doesn't support options as bash options, like -x, or arguments, for instance --<arg>.

It sets variables, such as the following:

• ERRORLEVEL
The following example specifies a heading.

```
\rset heading "Winter Sales Report"
```

For more examples of how to use `\rset`, you can find several in the Amazon Redshift RSQL variables (p. 216) topics.

`\RUN`

Runs the Amazon Redshift RSQL script contained in the specified file. `\RUN` extends the `\i` command by adding an option to skip header lines in a file.

If the file name includes a comma, semicolon, or space, enclose it in single quotation marks. Additionally, if text follows the file name, enclose it in quotation marks. In UNIX, file names are case sensitive. In Windows, file names are case insensitive.

The following sample shows output from the command.

```
(testcluster) user1@dev=日照 \cat test.sql
select count(*) as lineitem_cnt from lineitem;
select count(*) as customer_cnt from customer;
select count(*) as orders_cnt from orders;

(testcluster) user1@dev=日照 \run file=test.sql
lineitem_cnt
--------------
4307207
(1 row)
customer_cnt
--------------
37796166
(1 row)
orders_cnt
------------
0
(1 row)

(testcluster) user1@dev=日照 \run file=test.sql skip=2
2 records skipped in RUN file.
orders_cnt
------------
0
(1 row)
```

`\OS`

An alias for the `\!` command. `\OS` runs the operating system command that is passed as a parameter. Control returns to Amazon Redshift RSQL after the command is run. For example, you can run the following command to print the current system date time and return to the RSQL terminal: `\os date`.
\GOTO

A new command for Amazon Redshift RSQL. \GOTO skips all intervening commands and resumes processing at the specified \LABEL. The \LABEL must be a forward reference. You cannot jump to a \LABEL that lexically precedes the \GOTO.

The following shows sample output.

(testcluster) user1@dev=ʥ \! cat test.sql
select count(*) as cnt from lineitem \gset
select :cnt as cnt;
\if :cnt > 100
\goto LABELB
\endif
\label LABELA
\remark 'this is label LABELA'
\label LABELB
\remark 'this is label LABELB'

(testcluster) user1@dev=ʥ \i test.sql

cnt
---------
4307207
(1 row)
\label LABELA ignored
\label LABELB processed
this is label LABELB

\LABEL

A new command for Amazon Redshift RSQL. \LABEL establishes an entry point for running the program, as the target for a \GOTO command.

The following shows sample output from the command.

(testcluster) user1@dev= реша cat test.sql
select count(*) from lineitem limit 5;
\goto LABELB
\remark "this step was skipped by goto label";
\label LABELA
\remark 'this is label LABELA'
\label LABELB
\remark 'this is label LABELB'

(testcluster) user1@dev= реша \i testgoto.sql

count
4307193
(1 row)
\label LABELA ignored
\label LABELB processed
this is label LABELB
\IF (\ELSEIF, \ELSE, \ENDIF)

\IF and related commands conditionally run portions of the input script. An extension of the PSQL \if (\elif, \else, \endif) command. \IF and \ELSEIF support boolean expressions including AND, OR and NOT conditions.

The following shows sample output from the commands.

```sql
(testcluster) user1@dev=# \! cat test.sql
SELECT query FROM stv_inflight LIMIT 1 \gset
select :query as query;
\if :query > 1000000
 \remark 'Query id is greater than 1000000'
\elseif :query = 1000000
 \remark 'Query id is equal than 1000000'
\else
 \remark 'Query id is less than 1000000'
\endif

(testcluster) user1@dev=# \i test.sql
query
--------
994803
(1 row)
Query id is less than 1000000
```

Use ERRORCODE in your branching logic.

```sql
\if :ERRORCODE = '00000'
 \remark 'The statement was executed without error'
\else
 \remark :LAST_ERROR_MESSAGE
\endif
```

Use \GOTO within an \IF block to control how code is run.

**Amazon Redshift RSQL variables**

Some keywords act as variables in RSQL. You can set each to a specific value, or re-set the value. Most are set with \rset, which has an interactive mode and a batch mode. Commands may be defined in lower or upper case.

**ACTIVITYCOUNT**

Indicates the number of rows affected by the last submitted request. For a data-returning request, this is the number of rows returned to RSQL from the database. The value is 0 or a positive integer. The maximum value is $18,446,744,073,709,551,615$.

The specially treated variable ACTIVITYCOUNT is similar to the variable ROW_COUNT. However, ROW_COUNT doesn't report a count of affected rows to the client application at command completion for SELECT, COPY or UNLOAD. But ACTIVITYCOUNT does.

activitycount_01.sql:

```sql
select viewname, schemaname
from pg_views
where schemaname = 'not_existing_schema';
\if :ACTIVITYCOUNT = 0
```
\remark 'views do not exist'
\endif

Console output:

viewname | schemaname
----------+------------
(0 rows)
views do not exist

**ERRORLEVEL**

Assigns severity levels to errors. Use the severity levels to determine a course of action. If the ERRORLEVEL command has not been used, its value is ON by default.

**errorlevel_01.sql:**

```
\reset errorlevel 42P01 severity 0
select * from tbl;
select 1 as col;
\echo exit
\quit
```

Console output:

Errorlevel is on.
rsql: ERROR: relation "tbl" does not exist
(1 row)
col
1
exit

**HEADING and RTITLE**

Enables users to specify a header that appears at the top of a report. Header specified by the RSET RTITLE command automatically includes the current system date of the client computer.

**rset_heading_rtitle_02.rsql content:**

```
\remark Starting...
\reset rtitle "Marketing Department||Confidential//Third Quarter//Chicago"
\reset width 70
\reset rformat on
select * from rsql_test.tbl_currency order by id limit 2;
\exit
\remark Finishing...
```

Console output:

Starting...
Rtitle is set to: &DATE||Marketing Department||Confidential//Third Quarter//Chicago
(Changes will take effect after RFORMAT is switched ON)
Target width is 70.
Rformat is on.

09/11/20       Marketing       Department Confidential
Third Quarter
Chicago
id  | bankid | name |      start_date
100 |       1 | USD | 2020-09-11 10:51:39.106905
110 |       1 | EUR | 2020-09-11 10:51:39.106905
(2 rows)

Press any key to continue . . .

**MAXERROR**

Designates a maximum error-severity level beyond which RSQL terminates job processing. Return codes are integer values that RSQL returns to the client operating system after completing each job or task. The value of the return code indicates the completion status of the job or task. If a script contains a statement that produces an error-severity level greater than the designated `maxerror` value, RSQL immediately exits. Therefore, to have RSQL exit on an error-severity level of 8, use `RSET MAXERROR 7`.

**maxerror_01.sql** content:

```sql
RSET MAXERROR 0
select 1 as col;
quilt
```

Console output:

```
Maxerror is default.
(1 row)
col
1
```

**RFORMAT**

Enables users to specify whether to apply settings for the formatting commands.

**rset_rformat.rsql** content:

```sql
\remark Starting...
pset border 2
pset format wrapped
pset expanded on
pset title 'Great Title'
select * from rsql_test.tbl_long where id = 500;
rset rformat
select * from rsql_test.tbl_long where id = 500;
rset rformat off
select * from rsql_test.tbl_long where id = 500;
rset rformat on
select * from rsql_test.tbl_long where id = 500;
exit
\remark Finishing...
```

Console output:

```
Starting...
```
In general, the higher the number the more borders and lines the tables will have, but details depend on the particular format.

(1 row)

Rformat is on.
Great Title
+-[ RECORD
  1 ]+----------------------------------------------------------+
| id           | 500 |
| long_string  | In general, the higher the number the more borders and lines the tables will have, but details depend on the particular format. |
+-------------+

Rformat is off.
id | long_string
500 | In general, the higher the number the more borders and lines the tables will have, but details depend on the particular format.
(1 row)

Rformat is on.
Great Title
+-[ RECORD
  1 ]+----------------------------------------------------------+
| id           | 500 |
| long_string  | In general, the higher the number the more borders and lines the tables will have, but details depend on the particular format. |
+-------------+

Press any key to continue . . .

**ROW_COUNT**

Gets the number of records affected by the previous query. It's typically used to check a result, like in the following code fragment:

```sql
SET result = ROW_COUNT;
IF result = 0
...
```

**TITLEDASHES**

This control enables users to specify whether a line of dash characters is to be printed above the column data returned for SQL statements.

Example:

```
\reset titledashes on
```
```sql
select dept_no, emp_no, salary from rsql_test.EMPLOYEE
where dept_no = 100;
set titledashes off
select dept_no, emp_no, salary from rsql_test.EMPLOYEE
where dept_no = 100;
```

**Console output:**

<table>
<thead>
<tr>
<th>dept_no</th>
<th>emp_no</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1000346</td>
<td>1300.00</td>
</tr>
<tr>
<td>100</td>
<td>1000245</td>
<td>5000.00</td>
</tr>
<tr>
<td>100</td>
<td>1000262</td>
<td>2450.00</td>
</tr>
</tbody>
</table>

**WIDTH**

Sets the output format to wrapped and specifies the target width for each line in a report. Without a parameter, it returns the current settings for both the format and target width.

**rset_width_01.rsq1 content:**

```sql
\echo Starting...
\rset width
\rset width 50
\rset width
\quit
\echo Finishing...
```

**Console output:**

Starting...
Target width is 75.
Target width is 50.
Target width is 50.
Press any key to continue . . .

**Example with parameter:**

```sql
\echo Starting...
\rset rformat on
\pset format wrapped
select * from rsql_test.tbl_long where id = 500;
\rset width 50
select * from rsql_test.tbl_long where id = 500;
\quit
\echo Finishing...
```

**Console output:**

Starting...
Rformat is on.
Output format is wrapped.

<table>
<thead>
<tr>
<th>id</th>
<th>long_string</th>
</tr>
</thead>
<tbody>
<tr>
<td>220</td>
<td></td>
</tr>
</tbody>
</table>
Connect to your cluster by using SQL Workbench/J

Amazon Redshift doesn't provide or install any SQL client tools or libraries, so you must install any that you want to use with your clusters. If you already have a business intelligence application or any other application that can connect to your clusters using a standard PostgreSQL JDBC or ODBC driver, then you can skip this section. If you don't already have an application that can connect to your cluster, this section presents one option for doing so using SQL Workbench/J, a free, DBMS-independent, cross-platform SQL query tool.

Install SQL Workbench/J

The Amazon Redshift Getting Started Guide uses SQL Workbench/J. In this section, we explain in detail how to connect to your cluster by using SQL Workbench/J.

To install SQL Workbench/J

1. Review the SQL Workbench/J software license.
2. Go to the SQL Workbench/J website and download the appropriate package for your operating system on your client computer or Amazon EC2 instance.
3. Go to the Installing and starting SQL Workbench/J page. Follow the instructions for installing SQL Workbench/J on your system.

Note

SQL Workbench/J requires the Java Runtime Environment (JRE) be installed on your system. Ensure you are using the correct version of the JRE required by the SQL Workbench/J client. To determine which version of the Java Runtime Environment is running on your system, do one of the following:

- Mac: In the System Preferences, choose the Java icon.
- Windows: In the Control Panel, choose the Java icon.
- Any system: In a command shell, type `java -version`. You can also visit https://www.java.com, choose the Do I have Java? link, and choose the Verify Java button.

For information about installing and configuring the Java Runtime Environment, go to https://www.java.com.

Connect to your cluster over a JDBC connection in SQL Workbench/J

Important

Before you perform the steps in this procedure, make sure that your client computer or Amazon EC2 instance has the recommended Amazon Redshift JDBC driver. For links to download the latest drivers, see Download the Amazon Redshift JDBC driver (p. 170). Also, make sure you have configured firewall settings to allow access to your cluster. For more information, see Step 4: Authorize access to the cluster.
To use a JDBC connection in SQL Workbench/J

1. Open SQL Workbench/J.
2. Choose File, and then choose Connect window.
3. Choose Create a new connection profile.
4. In the New profile box, type a name for the profile. For example, examplecluster_jdbc.
5. Choose Manage Drivers. The Manage Drivers dialog opens. In the Name box, type a name for the driver.

Choose the folder icon next to the Library box, navigate to the location of the driver, choose it, and then choose Open.

If the Please select one driver dialog box displays, select com.amazon.redshift.jdbc4.Driver or com.amazon.redshift.jdbc41.Driver and choose OK. SQL Workbench/J automatically completes the Classname box. Leave the Sample URL box blank, and then choose OK.

6. In the Driver box, select the driver you just added.
7. In URL, copy the JDBC URL from the Amazon Redshift console and paste it here.

For more information about finding the JDBC URL, see Configuring a JDBC driver version 1.x connection (p. 170).

8. In Username, type the name of the admin user.
If you are following the Amazon Redshift Getting Started Guide, type `adminuser`.

9. In **Password**, type the password associated with the admin user account.

10. Select the **Autocommit** box.

11. Choose the **Save profile list** icon, as shown below:

![Save profile list icon](image)

12. Choose **OK**.

---

**Test the SQL Workbench/J connection**

After you configure your JDBC or ODBC connection, you can test the connection by running an example query.

1. You can use the following query to test your connection.

   ```sql
   select * from information_schema.tables;
   ```

   If your connection is successful, a listing of records appears in the **Results** tab.
2. Alternatively, if you loaded the sample tables and data from the Amazon Redshift Getting Started Guide, you can test your connection by typing the following query into the **Statement** window:

```sql
select * from users order by userid limit 100;
```

If your connection is successful, a listing of records appears in the **Results** tab.

---

**Connect to your cluster by using the psql tool**

After you create an Amazon Redshift cluster, you can use psql, a terminal-based front end from PostgreSQL, to query the data in your cluster. You can type the queries interactively or read them from a file. To connect from psql, you must specify the cluster endpoint, database, and port.

**Note**

Amazon Redshift does not provide the psql tool; it is installed with PostgreSQL. For information about using psql, go to [https://www.postgresql.org/docs/8.4/static/app-psql.html](https://www.postgresql.org/docs/8.4/static/app-psql.html). For
information about installing the PostgreSQL client tools, select your operating system from the PostgreSQL binary downloads page at https://www.postgresql.org/download/.
If you have trouble connecting from a Microsoft Windows prompt due to an invalid client_encoding, set the PGCLIENTENCODING environment variable to UTF-8 before running psql.

set PGCLIENTENCODING=UTF8

**Connect by using the psql defaults**

By default, psql does not validate the Amazon Redshift service; it makes an encrypted connection by using Secure Sockets Layer (SSL).

**To connect by using psql defaults**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. Choose one of the following steps depending on which Amazon Redshift console you are using:
   - In the navigation pane, choose Clusters. Then choose the cluster name from the list to open its details. On the Properties tab, in the Database configurations section, record the Database name and Port. View the Connection details section and record the Endpoint which is in the following form:

   ```
   endpoint:port/databasename
   ```

   - At a command prompt, specify the connection information by using either command line parameters or a connection information string. To use parameters:
psql -h <endpoint> -U <userid> -d <databasename> -p <port>

Where:

- `<endpoint>` is the **Endpoint** you recorded in the previous step.
- `<userid>` is a user ID with permissions to connect to the cluster.
- `<databasename>` is the **Database Name** you recorded in the previous step.
- `<port>` is the **Port** you recorded in the previous step.

For example:

```bash
psql -h examplecluster.<XXXXXXXXXXXX>.us-west-2.redshift.amazonaws.com -U adminuser -d dev -p 5439
```

4. At the psql password prompt, enter the password for the `<userid>` user.

You are connected to the cluster, and you can interactively enter commands.

**Connect by using a certificate**

To control whether psql authenticates the service using a certificate, you must use a connection information string to specify connection information, and specify the `sslmode` keyword. By default, `psql` operates with `sslmode=prefer`. To specify that `psql` opens an encrypted connection and uses an Amazon Redshift certificate to verify the service, download an Amazon Redshift certificate to your computer. Specify `verify-full` unless you use a DNS alias. If you use a DNS alias, select `verify-ca`. Specify `sslrootcert` with the location of the certificate. For more information about `sslmode`, see Configuring security options for connections (p. 193).

For more information about connection information string parameters, see https://www.postgresql.org/docs/8.4/static/libpq-connect.html.

**To connect by using a certificate**

1. Save the download of the Amazon trust certificate authority bundle, a .crt file, to your computer. For download information, see Connect using SSL (p. 193). If you do a File\Save as using Internet Explorer, specify the file type as **Text file (*.txt)** and delete the .txt extension. For example, save it as the file `C:\MyDownloads\amazon-trust-ca-bundle.crt`.

2. Choose one of the following steps depending on which Amazon Redshift console you are using:

   - In the navigation pane, choose **Clusters**. Then choose the cluster name from the list to open its details. On the **Properties** tab, in the **Database configurations** section, record the **Database name** and **Port**. View the **Connection details** section and record the **Endpoint** which is in the following form:

   ```plaintext
   endpoint:port/databasename
   ```
3. At a command prompt, specify the connection information using a connection information string:

```bash
psql "host=<endpoint> user=<userid> dbname=<databasename> port=<port> sslmode=verify-ca sslrootcert=<certificate>"
```

Where:
- `<endpoint>` is the **Endpoint** you recorded in the previous step.
- `<userid>` is a user ID with permissions to connect to the cluster.
- `<databasename>` is the **Database Name** you recorded in the previous step.
- `<port>` is the **Port** you recorded in the previous step.
- `<certificate>` is the full path to the certificate file. On Windows systems, the certificate path must be specified using Linux-style `/` separators instead of the Windows `\` separator.

On Linux and macOS X operating systems, the path is

```
~/.postgresql/root.crt
```

On Microsoft Windows, the path is

```
%APPDATA%\postgresql\root.crt
```

For example:
psql "host=examplecluster.<xxxxxxxxxxxxx>.us-west-2.redshift.amazonaws.com user=adminuser dbname=dev port=5439 sslmode=verify-ca sslrootcert=C:/MyDownloads/amazon-trust-ca-bundle.crt"

4. At the psql password prompt, enter the password for the <userid> user.

You are connected to the cluster, and you can interactively enter commands.

**Connect to your cluster programmatically**

This section explains how to connect to your cluster programmatically. If you are using an application like SQL Workbench/J that manages your client connections for you, then you can skip this section.

**Connecting to a cluster by using Java**

When you use Java to programmatically connect to your cluster, you can do so with or without server authentication. If you plan to use server authentication, follow the instructions in Configuring security options for connections (p. 193) to put the Amazon Redshift server certificate into a keystore. You can refer to the keystore by specifying a property when you run your code as follows:

```
-Djavax.net.ssl.trustStore=<path to keystore>
-Djavax.net.ssl.trustStorePassword=<keystore password>
```

**Example: Connect to a cluster by using Java**

The following example connects to a cluster and runs a sample query that returns system tables. It is not necessary to have data in your database to use this example.

If you are using a server certificate to authenticate your cluster, you can restore the line that uses the keystore, which is commented out:

```
props.setProperty("ssl", "true");
```

For more information about the server certificate, see Configuring security options for connections (p. 193).

For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449).

```
package connection;

import java.sql.*;
import java.util.Properties;

public class ConnectToCluster {

  //Redshift driver: "jdbc:redshift://x.y.us-west-2.redshift.amazonaws.com:5439/dev";
  static final String dbURL = "***jdbc cluster connection string ****";
  static final String MasterUsername = "***master user name***";
  static final String MasterUserPassword = "***master user password***";

  public static void main(String[] args) {
    Connection conn = null;
    Statement stmt = null;
    try{
      //Dynamically load driver at runtime.
      //Redshift JDBC 4.1 driver: com.amazon.redshift.jdbc41.Driver
      //Redshift JDBC 4 driver: com.amazon.redshift.jdbc4.Driver
      Class.forName("com.amazon.redshift.jdbc.Driver");

      //Open a connection and define properties.
      System.out.println("Connecting to database...");
      Properties props = new Properties();
      //Uncomment the following line if using a keystore.
      //props.setProperty("ssl", "true");
      props.setProperty("user", MasterUsername);
      props.setProperty("password", MasterUserPassword);
      conn = DriverManager.getConnection(dbURL, props);

      //Try a simple query.
      System.out.println("Listing system tables...");
      stmt = conn.createStatement();
      String sql;
      sql = "select * from information_schema.tables;";
      ResultSet rs = stmt.executeQuery(sql);

      //Get the data from the result set.
      while(rs.next()){
        //Retrieve two columns.
        String catalog = rs.getString("table_catalog");
        String name = rs.getString("table_name");

        //Display values.
        System.out.print("Catalog: " + catalog);
        System.out.println(" Name: " + name);
      }
      rs.close();
      stmt.close();
      conn.close();
    }catch(Exception ex){
      //For convenience, handle all errors here.
      ex.printStackTrace();
    }finally{
      //Finally block to close resources.
      try{

Connecting to a cluster by using .NET

When you use .NET (C#) to programmatically connect to your cluster, you can do so with or without server authentication. If you plan to use server authentication, follow the instructions in Configuring security options for connections (p. 193) to download the Amazon Redshift server certificate, and then put the certificate in the correct form for your .NET code.

Example Connect to a cluster by using .NET

The following example connects to a cluster and runs a sample query that returns system tables. It does not show server authentication. It is not necessary to have data in your database to use this example. This example uses the System.Data.Odbc namespace, a .NET Framework Data Provider for ODBC.

```csharp
using System;
using System.Data;
using System.Data.Odbc;

namespace redshift.amazon.com.docexamples
{
    class ConnectToClusterExample
    {
        public static void Main(string[] args)
        {

            DataSet ds = new DataSet();
            DataTable dt = new DataTable();

            // Server, e.g. "examplecluster.xyz.us-west-2.redshift.amazonaws.com"
            string server = "***provide server name part of connection string***";

            // Port, e.g. "5439"
            string port = "***provide port***";

            // MasterUserName, e.g. "masteruser".
            string masterUsername = "***provide master user name***";

            // MasterUserPassword, e.g. "mypassword".
            string masterUserPassword = "***provide master user password***";

            // DBName, e.g. "dev"
            string dbName = "***provide name of database***";

            string query = "select * from information_schema.tables;";

            try
            {
                // Create the ODBC connection string.
```
Connecting to clusters from client tools and code

```csharp
// Redshift ODBC Driver - 64 bits
/*
string connString = "Driver={Amazon Redshift (x64)};" +
    String.Format("Server={0};Database={1};" +
    "UID={2};PWD={3};Port={4};SSL=true;Sslmode=Require", server, DBName, masterUsername,
    masterUserPassword, port);
*/

// Redshift ODBC Driver - 32 bits
string connString = "Driver={Amazon Redshift (x86)};" +
    String.Format("Server={0};Database={1};" +
    "UID={2};PWD={3};Port={4};SSL=true;Sslmode=Require", server, DBName, masterUsername,
    masterUserPassword, port);

// Make a connection using the psqlODBC provider.
OdbcConnection conn = new OdbcConnection(connString);
conn.Open();

// Try a simple query.
string sql = query;
OdbcDataAdapter da = new OdbcDataAdapter(sql, conn);
da.Fill(ds);
dt = ds.Tables[0];
foreach (DataRow row in dt.Rows)
{
    Console.WriteLine(row["table_catalog"] + ", " + row["table_name"]);
}

conn.Close();
Console.ReadKey();
} catch (Exception ex)
{
    Console.Error.WriteLine(ex.Message);
}
```

Using an authentication profile to connect to Amazon Redshift

If you have many connections to Amazon Redshift, it can be difficult to manage settings for all of them. Often, each JDBC or ODBC connection uses specific configuration options. By using an authentication profile, you can store connection options together. This way, your users can choose a profile to connect with and avoid managing settings for individual options. Profiles can apply to various scenarios and user types.

After you create an authentication profile, users can add the ready-to-use profile to a connection string. By doing this, they can connect to Amazon Redshift with the right settings for each role and use case.

For Amazon Redshift API information, see CreateAuthenticationProfile.

Creating an authentication profile

Using the AWS CLI, you create an authentication profile with the create-authentication-profile command. This assumes that you have an existing Amazon Redshift cluster and an existing database. Your credentials must have permission to connect to the Amazon Redshift database and rights to fetch the authentication profile. You provide the configuration options as a JSON string, or reference a file containing your JSON string.
create-authentication-profile --authentication-profile-name<value: String> --authentication-profile-content<value: String>

The following example creates a profile called ExampleProfileName. Here, you can add keys and values that define your cluster name and other option settings, as a JSON string.

create-authentication-profile --authentication-profile-name "ExampleProfileName" --authentication-profile-content "{"AllowDBUserOverride":1,"Client_ID":"ExampleClientID","App_ID":"ExampleAppID","AutoCreate":false,"enableFetchRingBuffer":true,"databaseMetadataCurrentDbOnly":true}"

This command creates the profile with the specified JSON settings. The following is returned, which indicates that the profile is created.

{"AuthenticationProfileName": "ExampleProfileName", "AuthenticationProfileContent": "{"AllowDBUserOverride":1,"Client_ID":"ExampleClientID","App_ID":"ExampleAppID","AutoCreate":false,"enableFetchRingBuffer":true,"databaseMetadataCurrentDbOnly":true}" }  

Limitations and quotas for creating an authentication profile

Each customer has a quota of ten (10) authentication profiles.

Certain errors can occur with authentication profiles. Examples are if you create a new profile with an existing name, or if you exceed your profile quota. For more information, see CreateAuthenticationProfile.

You can't store certain option keys and values for JDBC, ODBC, and Python connection strings in the authentication profile store:

- AccessKeyID
- access_key_id
- SecretAccessKey
- secret_access_key_id
- PWD
- Password
- password

You can't store the key or value AuthProfile in the profile store, for JDBC or ODBC connection strings. For Python connections, you can't store auth_profile.

Authentication profiles are stored in Amazon DynamoDB and managed by AWS.

Working with authentication profiles

After you create an authentication profile, you can include the profile name as a connection option for JDBC version 2.0 AuthProfile. Using this connection option retrieves the stored settings.


The following is an example JDBC URL string.
jdbc:redshift:iam://examplecluster:us-west-2/dev?
AuthProfile="ExampleProfile"&AccessKeyId="AKIAIOSFODNN7EXAMPLE"&SecretAccessKey="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"

Specify both the AccessKeyID and SecretAccessKey in the JDBC URL, along with the authentication profile name.

You can also separate the configuration options with semicolon delimiters, such as in the following example, which includes options for logging.

jdbc:redshift:iam://my_redshift_end_point:5439/dev?LogLevel=6;LogPath=/tmp;AuthProfile=my_profile;AccessKeyId="AKIAIOSFODNN7EXAMPLE";SecretAccessKey="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"

**Note**
Don't add confidential information to the authentication profile. For example, don't store an AccessKeyID or SecretAccessKey value in an authentication profile. The authentication profile store has rules to prohibit storage of secret keys. You get an error if you try to store a key and value associated with sensitive information.

**Getting authentication profiles**

To list existing authentication profiles, call the following command.

describe-authentication-profiles --authentication-profile-name <value: String>

The following example shows two retrieved profiles. All profiles are returned if you don't specify a profile name.

```
{ "AuthenticationProfiles": [ { "AuthenticationProfileName": "testProfile1", "AuthenticationProfileContent": '{"AllowDBUserOverride":1,"Client_ID":"ExampleClientID","App_ID":"ExampleAppID","AutoCreate":false,"enableFetchRingBuffer":true,"databaseMetadataCurrentDbOnly":true} ' }, { "AuthenticationProfileName": "testProfile2", "AuthenticationProfileContent": '{"AllowDBUserOverride":1,"Client_ID":"ExampleClientID","App_ID":"ExampleAppID","AutoCreate":false,"enableFetchRingBuffer":true,"databaseMetadataCurrentDbOnly":true}' ] }
```

**Troubleshooting connection issues in Amazon Redshift**

If you have issues with connecting to your cluster from a SQL client tool, there are several things that you can check to narrow down the problem. If you are using SSL or server certificates, first remove this complexity while you troubleshoot the connection issue. Then add this back when you have found a solution. For more information, see Configuring security options for connections (p. 193).

**Important**
Amazon Redshift has changed the way that SSL certificates are managed. If you have trouble connecting using SSL, you might need to update your current trust root CA certificates. For more information, see Transitioning to ACM certificates for SSL connections (p. 196).

The following section has some example error messages and possible solutions for connection issues. Because different SQL client tools provide different error messages, this is not a complete list, but should be a good starting point for troubleshooting issues.
Troubleshooting connection issues in Amazon Redshift

Topics
- Connecting from outside of Amazon EC2—firewall timeout issue (p. 234)
- Connection is refused or fails (p. 236)
- Client and driver are incompatible (p. 236)
- Queries appear to hang and sometimes fail to reach the cluster (p. 236)

Connecting from outside of Amazon EC2—firewall timeout issue

Example issue

Your client connection to the database appears to hang or timeout when running long queries, such as a COPY command. In this case, you might observe that the Amazon Redshift console displays that the query has completed, but the client tool itself still appears to be running the query. The results of the query might be missing or incomplete depending on when the connection stopped.

Possible solutions

This issue happens when you connect to Amazon Redshift from a machine other than an Amazon EC2 instance. In this case, idle connections are terminated by an intermediate network component, such as a firewall, after a period of inactivity. This behavior is typical when you log on from a virtual private network (VPN) or your local network.

To avoid these timeouts, we recommend the following changes:

- Increase client system values that deal with TCP/IP timeouts. Make these changes on the computer you are using to connect to your cluster. The timeout period should be adjusted for your client and network. For more information, see Change TCP/IP timeout settings (p. 234).
- Optionally, set keepalive behavior at the DSN level. For more information, see Change DSN timeout settings (p. 235).

Change TCP/IP timeout settings

To change TCP/IP timeout settings, configure the timeout settings according to the operating system that you use to connect to your cluster.

- Linux — If your client is running on Linux, run the following command as the root user to change the timeout settings for the current session:

```
/sbin/sysctl -w net.ipv4.tcp_keepalive_time=200 net.ipv4.tcp_keepalive_intvl=200 net.ipv4.tcp_keepalive_probes=5
```

To persist the settings, create or modify the file `/etc/sysctl.conf` with the following values then reboot your system.

```
net.ipv4.tcp_keepalive_time=200
net.ipv4.tcp_keepalive_intvl=200
net.ipv4.tcp_keepalive_probes=5
```

- Windows — If your client runs on Windows, edit the values for the following registry settings under `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`:
  - KeepAliveTime: 30000
  - KeepAliveInterval: 1000
Troubleshooting connection issues in Amazon Redshift

- **TcpMaxDataRetransmissions**: 10

  These settings use the DWORD data type. If they do not exist under the registry path, you can create the settings and specify these recommended values. For more information about editing the Windows registry, refer to Windows documentation.

  After you set these values, restart your computer for the changes to take effect.

- **Mac** — If your client is running on a Mac, run the following commands to change the timeout settings for the current session:

  ```bash
  sudo sysctl net.inet.tcp.keepintvl=200000
  sudo sysctl net.inet.tcp.keepidle=200000
  sudo sysctl net.inet.tcp.keepinit=200000
  sudo sysctl net.inet.tcp.always_keepalive=1
  ```

  To persist the settings, create or modify the file `/etc/sysctl.conf` with the following values:

  ```bash
  net.inet.tcp.keepidle=200000
  net.inet.tcp.keepintvl=200000
  net.inet.tcp.keepinit=200000
  net.inet.tcp.always_keepalive=1
  ```

  Restart your computer, and then run the following commands to verify that the values are set.

  ```bash
  sysctl net.inet.tcp.keepidle
  sysctl net.inet.tcp.keepintvl
  sysctl net.inet.tcp.keepinit
  sysctl net.inet.tcp.always_keepalive
  ```

**Change DSN timeout settings**

You can set keepalive behavior at the DSN level if you choose. You do this by adding or modifying the following parameters in the `odbc.ini` file:

- **KeepAlivesCount**
  The number of TCP keepalive packets that can be lost before the connection is considered broken.

- **KeepAlivesIdle**
  The number of seconds of inactivity before the driver sends a TCP keepalive packet.

- **KeepAlivesInterval**
  The number of seconds between each TCP keepalive retransmission.

On Windows, you modify these parameters in the registry by adding or changing keys in `HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBC.INI\your_DSN`. On Linux and macOS, you add or modify these parameters in the target DSN entry directly in the `odbc.ini` file. For more information on modifying the `odbc.ini` file on Linux and macOS computers, see Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 184).

If these parameters don't exist, or if they have a value of 0, the system uses the keepalive parameters specified for TCP/IP to determine DSN keepalive behavior. On Windows, you can find the TCP/IP parameters in the registry in `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services`
Connection is refused or fails

Example errors

- "Failed to establish a connection to `<endpoint>`."
- "Could not connect to server: Connection timed out. Is the server running on host `<endpoint>` and accepting TCP/IP connections on port `<port>`?"
- "Connection refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections."

Possible solutions

Generally, when you receive an error message indicating that there is a failure to establish a connection, it is an issue with permission to access the cluster.

To connect to the cluster from a client tool outside of the network that the cluster is in, add an ingress rule. Add the rule to the cluster security group for the CIDR/IP that you are connecting from:

- If you created your Amazon Redshift cluster in a virtual private cloud (VPC) based on Amazon VPC, add your client CIDR/IP address to the VPC security group in Amazon VPC. For more information about configuring VPC security groups for your cluster, see Managing clusters in a VPC (p. 79).
- If you created your Amazon Redshift cluster outside a VPC, add your client CIDR/IP address to the cluster security group in Amazon Redshift. For more information about configuring cluster security groups, see Amazon Redshift cluster security groups (p. 438).

If you attempt to connect to the cluster from a client tool in an Amazon EC2 instance, you also add an ingress rule. In this case, add the rule to the cluster security group for the Amazon EC2 security group that is associated with the Amazon EC2 instance. For more information about configuring cluster security groups, see Amazon Redshift cluster security groups (p. 438).

In some cases, you might have a layer between your client and server, such as a firewall. In these cases, make sure that the firewall accepts inbound connections over the port that you configured for your cluster.

Client and driver are incompatible

Example error

"The specified DSN contains an architecture mismatch between the Driver and Application."

Possible solution

When you attempt to connect and get an error about an architecture mismatch, this means that the client tool and the driver aren't compatible. This occurs because their system architecture doesn't match. For example, this can happen if you have a 32-bit client tool but have installed the 64-bit version of the driver. Sometimes 64-bit client tools can use 32-bit drivers, but you can't use 32-bit applications with 64-bit drivers. Make sure that the driver and client tool are using the same version of the system architecture.

Queries appear to hang and sometimes fail to reach the cluster

Example issue
You experience an issue with queries completing, where the queries appear to be running but hang in the SQL client tool. Sometimes the queries fail to appear in the cluster, such as in system tables or the Amazon Redshift console.

Possible solution

This issue can happen due to packet drop. In this case, there is a difference in the maximum transmission unit (MTU) size in the network path between two Internet Protocol (IP) hosts. The MTU size determines the maximum size, in bytes, of a packet that can be transferred in one Ethernet frame over a network connection. In AWS, some Amazon EC2 instance types support an MTU of 1500 (Ethernet v2 frames) and other instance types support an MTU of 9001 (TCP/IP jumbo frames).

To avoid issues that can occur with differences in MTU size, we recommend doing one of the following:

• If your cluster uses the EC2-VPC platform, configure the Amazon VPC security group with an inbound custom Internet Control Message Protocol (ICMP) rule that returns Destination Unreachable. The rule thus instructs the originating host to use the lowest MTU size along the network path. For details on this approach, see Configuring security groups to allow ICMP “destination unreachable” (p. 237).

• If your cluster uses the EC2-Classic platform, or you can't allow the ICMP inbound rule, disable TCP/IP jumbo frames so that Ethernet v2 frames are used. For details on this approach, see Configuring the MTU of an instance (p. 237).

Configuring security groups to allow ICMP "destination unreachable"

When there is a difference in the MTU size in the network between two hosts, first make sure that your network settings don't block path MTU discovery (PMTUD). PMTUD enables the receiving host to respond to the originating host with the following ICMP message: Destination Unreachable: fragmentation needed and DF set (ICMP Type 3, Code 4). This message instructs the originating host to use the lowest MTU size along the network path to resend the request. Without this negotiation, packet drop can occur because the request is too large for the receiving host to accept. For more information about this ICMP message, go to RFC792 on the Internet Engineering Task Force (IETF) website.

If you don't explicitly configure this ICMP inbound rule for your Amazon VPC security group, PMTUD is blocked. In AWS, security groups are virtual firewalls that specify rules for inbound and outbound traffic to an instance. For information about Amazon Redshift cluster security group, see Amazon Redshift cluster security groups (p. 438). For clusters using the EC2-VPC platform, Amazon Redshift uses VPC security groups to allow or deny traffic to the cluster. By default, the security groups are locked down and deny all inbound traffic. For information about how to set inbound and outbound rules for EC2-Classic or EC2-VPC instances, see Differences between instances in EC2-Classic and a VPC in the Amazon EC2 User Guide for Linux Instances.

For more information about how to add rules to VPC security groups, see Managing VPC security groups for a cluster (p. 81). For more information about specific PMTUD settings required in this rule, see Path MTU discovery in the Amazon EC2 User Guide for Linux Instances.

Configuring the MTU of an instance

In some cases, your cluster might use the EC2-Classic platform or you can't allow the custom ICMP rule for inbound traffic. In these cases, we recommend that you adjust the MTU to 1500 on the network interface (NIC) of the EC2 instances you connect to your Amazon Redshift cluster from. This adjustment disables TCP/IP jumbo frames to ensure that connections consistently use the same packet size. However, this option reduces your maximum network throughput for the instance entirely, not just for connections to Amazon Redshift. For more information, see the following procedures.

To set MTU on a Microsoft Windows operating system

If your client runs in a Microsoft Windows operating system, you can review and set the MTU value for the Ethernet adapter by using the netsh command.
1. Run the following command to determine the current MTU value:

```
netsh interface ipv4 show subinterfaces
```

2. Review the **MTU** value for the **Ethernet** adapter in the output.
3. If the value is not **1500**, run the following command to set it:

```
netsh interface ipv4 set subinterface "Ethernet" mtu=1500 store=persistent
```

After you set this value, restart your computer for the changes to take effect.

**To set MTU on a Linux operating system**

If your client runs in a Linux operating system, you can review and set the MTU value by using the **ip** command.

1. Run the following command to determine the current MTU value:

```
$ ip link show eth0
```

2. Review the value following **mtu** in the output.
3. If the value is not **1500**, run the following command to set it:

```
$ sudo ip link set dev eth0 mtu 1500
```

**To set MTU on a Mac operating system**

- Follow instructions on the MacOS support site about **How to change the MTU for troubleshooting purposes**. For more information, search the support site.

---

**Using the Amazon Redshift Data API**

You can access your Amazon Redshift database using the built-in Amazon Redshift Data API. Using this API, you can access Amazon Redshift data with web services–based applications, including AWS Lambda, Amazon SageMaker notebooks, and AWS Cloud9. For more information on these applications, see AWS Lambda, Amazon SageMaker, and AWS Cloud9.

The Data API doesn't require a persistent connection to the cluster. Instead, it provides a secure HTTP endpoint and integration with AWS SDKs. You can use the endpoint to run SQL statements without managing connections. Calls to the Data API are asynchronous.

The Data API uses either credentials stored in AWS Secrets Manager or temporary database credentials. You don't need to pass passwords in the API calls with either authorization method. For more information about AWS Secrets Manager, see What Is AWS Secrets Manager? in the AWS Secrets Manager User Guide.

For more information about the Data API operations, see the Amazon Redshift Data API Reference.

**Working with the Amazon Redshift Data API**

Before you use the Amazon Redshift Data API, review the following steps:
1. Determine if you, as the caller of the Data API, are authorized. For more information about authorization, see Authorizing access to the Amazon Redshift Data API (p. 239).

2. Determine if you plan to call the Data API with authentication credentials from Secrets Manager or temporary credentials. For more information, see Choosing authentication credentials when calling the Amazon Redshift Data API (p. 242).

3. Set up a secret if you use Secrets Manager for authentication credentials. For more information, see Storing database credentials in AWS Secrets Manager (p. 240).

4. Review the considerations and limitations when calling the Data API. For more information, see Considerations when calling the Amazon Redshift Data API (p. 242).

5. Call the Data API from the AWS Command Line Interface (AWS CLI), from your own code, or using the query editor in the Amazon Redshift console. For examples of calling from the AWS CLI, see Calling the Data API with the AWS CLI (p. 244).

### Authorizing access to the Amazon Redshift Data API

To access the Data API, a user must be authorized. You can authorize a user to access the Data API by adding a managed policy, which is a predefined AWS Identity and Access Management (IAM) policy, to that user. To see the permissions allowed and denied by managed policies, see the IAM console (https://console.aws.amazon.com/iam/).

Amazon Redshift provides the AmazonRedshiftDataFullAccess managed policy. This policy provides full access to Amazon Redshift Data API operations. This policy also allows scoped access to specific Amazon Redshift, AWS Secrets Manager, and IAM API operations needed to authenticate and access an Amazon Redshift cluster. If you use AWS Secrets Manager to authenticate, the policy allows use of the secretsmanager:GetSecretValue action to retrieve the secret tagged with the key RedshiftDataFullAccess. If you use temporary credentials to authenticate, the policy allows use of the redshift:GetClusterCredentials action to the database user name redshift_data_api_user for any database in the cluster. This user name must have already been created in your database.

You can also create your own IAM policy that allows access to specific resources. To create your policy, use the AmazonRedshiftDataFullAccess policy as your starting template. After you create your policy, add it to each user that requires access to the Data API.

To run a query on a cluster that is owned by another account, the owning account must provide an IAM role that the Data API can assume in the calling account. For example, suppose Account B owns a cluster that Account A needs to access. Account B can attach the AWS-managed policy AmazonRedshiftDataFullAccess to Account B’s IAM role. Then Account B trusts Account A using a trust policy such as the following:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::accountID-of-account-A:role/someRoleA"
        ],
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Finally, the Account A IAM role needs to be able to assume the Account B IAM role.
Storing credentials in a secret

The following links provide more information about AWS Identity and Access Management in the IAM User Guide.

- For information about creating an IAM roles, see Creating IAM roles.
- For information about creating an IAM policy, see Creating IAM policies.
- For information about adding an IAM policy to a user, see Adding and removing IAM identity permissions.

Storing database credentials in AWS Secrets Manager

When you call the Data API, you can pass credentials for the cluster by using a secret in AWS Secrets Manager. To pass credentials in this way, you specify the name of the secret or the Amazon Resource Name (ARN) of the secret.

To store credentials with Secrets Manager, you need SecretManagerReadWrite managed policy permission. For more information about the minimum permissions, see Creating and Managing Secrets with AWS Secrets Manager in the AWS Secrets Manager User Guide.

To store your credentials in a secret for an Amazon Redshift cluster

1. Use AWS Secrets Manager to create a secret that contains credentials for your cluster:
   - When you choose Store a new secret, choose Credentials for Redshift cluster.
   - Store your values for User name (database user), Password, and DB cluster (cluster identifier) in your secret.
   - Tag the secret with the key RedshiftDataFullAccess. The AWS-managed policy AmazonRedshiftDataFullAccess only allows the action secretsmanager:GetSecretValue for secrets tagged with the key RedshiftDataFullAccess.

   For instructions, see Creating a Basic Secret in the AWS Secrets Manager User Guide.

2. Use the AWS Secrets Manager console to view the details for the secret you created, or run the aws secretsmanager describe-secret AWS CLI command.

   Note the name and ARN of the secret. You can use these in calls to the Data API.

Creating an Amazon VPC endpoint (AWS PrivateLink) for the Data API

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources, such as Amazon Redshift clusters and applications, into a virtual private cloud (VPC). AWS PrivateLink provides private connectivity between virtual private clouds (VPCs) and AWS services securely on the Amazon network. Using AWS PrivateLink, you can create VPC endpoints, which you can use connect to services across
Creating an Amazon VPC endpoint

You can call the Data API with Amazon VPC endpoints. Using an Amazon VPC endpoint keeps traffic between applications in your Amazon VPC and the Data API in the AWS network, without using public IP addresses. Amazon VPC endpoints can help you meet compliance and regulatory requirements related to limiting public internet connectivity. For example, if you use an Amazon VPC endpoint, you can keep traffic between an application running on an Amazon EC2 instance and the Data API in the VPCs that contain them.

After you create the Amazon VPC endpoint, you can start using it without making any code or configuration changes in your application.

To create an Amazon VPC endpoint for the Data API

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. Choose Endpoints, and then choose Create Endpoint.
3. On the Create Endpoint page, for Service category, choose AWS services. For Service Name, choose redshift-data (com.amazonaws.region.redshift-data).
4. For VPC, choose the VPC to create the endpoint in.
   - Choose the VPC that contains the application that makes Data API calls.
5. For Subnets, choose the subnet for each Availability Zone (AZ) used by the AWS service that is running your application.
   - To create an Amazon VPC endpoint, specify the private IP address range in which the endpoint is accessible. To do this, choose the subnet for each Availability Zone. Doing so restricts the VPC endpoint to the private IP address range specific to each Availability Zone and also creates an Amazon VPC endpoint in each Availability Zone.
6. For Enable DNS name, select Enable for this endpoint.
   - Private DNS resolves the standard Data API DNS hostname (https://redshift-data.region.amazonaws.com) to the private IP addresses associated with the DNS hostname specific to your Amazon VPC endpoint. As a result, you can access the Data API VPC endpoint using the AWS CLI or AWS SDKs without making any code or configuration changes to update the Data API endpoint URL.
7. For Security group, choose a security group to associate with the Amazon VPC endpoint.
   - Choose the security group that allows access to the AWS service that is running your application. For example, if an Amazon EC2 instance is running your application, choose the security group that allows access to the Amazon EC2 instance. The security group enables you to control the traffic to the Amazon VPC endpoint from resources in your VPC.
8. Choose Create endpoint.

After the endpoint is created, choose the link in the AWS Management Console to view the endpoint details.

The endpoint Details tab shows the DNS hostnames that were generated while creating the Amazon VPC endpoint.

You can use the standard endpoint (redshift-data.region.amazonaws.com) or one of the VPC-specific endpoints to call the Data API within the Amazon VPC. The standard Data API endpoint automatically routes to the Amazon VPC endpoint. This routing occurs because the Private DNS hostname was enabled when the Amazon VPC endpoint was created.
When you use an Amazon VPC endpoint in a Data API call, all traffic between your application and the Data API remains in the Amazon VPCs that contain them. You can use an Amazon VPC endpoint for any type of Data API call. For information about calling the Data API, see Considerations when calling the Amazon Redshift Data API (p. 242).

Considerations when calling the Amazon Redshift Data API

Consider the following when calling the Data API:

- The maximum duration of a query is 24 hours.
- The maximum number of active queries (STARTED and SUBMITTED queries) per Amazon Redshift cluster is 200.
- The maximum query result size is 100 MB. If a call returns more than 100 MB of response data, the call is ended.
- The maximum retention time for query results is 24 hours.
- The maximum query statement size is 100 KB.
- The Data API is available to query single-node and multiple-node clusters of the following node types:
  - dc2.large
  - dc2.8xlarge
  - ds2.xlarge
  - ds2.8xlarge
  - ra3.xlplus
  - ra3.4xlarge
  - ra3.16xlarge
- The cluster must be in a virtual private cloud (VPC) based on the Amazon VPC service.
- By default, users with the same IAM role or IAM user as the runner of an ExecuteStatement or BatchExecuteStatement API operation can act on the same statement with CancelStatement, DescribeStatement, GetStatementResult, and ListStatements API operations.
- For a list of AWS Regions where the Data API is available, see Redshift Data API Endpoints in the Amazon Web Services General Reference.

Choosing authentication credentials when calling the Amazon Redshift Data API

When you call the Data API, you use one of the following authentication methods for some API operations. Each method requires a different combination of parameters.

**AWS Secrets Manager**

With this method, provide the secret-arn secret value that is stored in AWS Secrets Manager. The specified secret contains credentials to connect to the database you specify. You also supply a value for cluster-identifier that matches the cluster identifier in the secret.

**Temporary credentials**

With this method, provide your cluster-identifier, database, and db-user values.

With either method, you can also supply a region value that specifies the AWS Region where your cluster is located.
## Mapping JDBC data types when calling the Amazon Redshift Data API

The following table maps Java Database Connectivity (JDBC) data types to the data types you specify in Data API calls.

<table>
<thead>
<tr>
<th>JDBC data type</th>
<th>Data API data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER, TINYINT, SMALLINT, BIGINT</td>
<td>LONG</td>
</tr>
<tr>
<td>FLOAT, REAL, DOUBLE</td>
<td>DOUBLE</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>STRING</td>
</tr>
<tr>
<td>BOOLEAN, BIT</td>
<td>BOOLEAN</td>
</tr>
<tr>
<td>BLOB, BINARY, LONGVARBINARY, VARBINARY</td>
<td>BLOB</td>
</tr>
<tr>
<td>CLOB</td>
<td>STRING</td>
</tr>
<tr>
<td>Other types (including types related to date and time)</td>
<td>STRING</td>
</tr>
</tbody>
</table>

String values are passed to the Amazon Redshift database and implicitly converted into a database data type.

**Note**
Currently, the Data API doesn't support arrays of universal unique identifiers (UUIDs).

## Running SQL statements with parameters when calling the Amazon Redshift Data API

You can control the SQL text submitted to the database engine by calling the Data API operation using parameters for parts of the SQL statement. Named parameters provide a flexible way to pass in parameters without hardcoding them in the SQL text. They help you reuse SQL text and avoid SQL injection problems.

The following example shows the named parameters of a `parameters` field of an `execute` statement AWS CLI command.

```bash
--parameters "[{"name": "id", "value": "1"},{"name": "address", "value": "Seattle"}]
```

Consider the following when using named parameters:

- The named parameters can be in any order and parameters can be used more than one time in the SQL text. The parameters option shown in the previous example, the values 1 and Seattle are inserted into the table columns `id` and `address`. In the SQL text, you specify the named parameters as follows:

  ```sql
  --sql "insert into mytable values (:id, :address)"
  ```

- When the SQL runs, data is implicitly cast to a data type. For more information about data type casting, see Data types in the Amazon Redshift Database Developer Guide.

- You can't set a value to NULL. The Data API interprets it as the literal string `NULL`. The following example replaces `id` with the literal string `null`. Not the SQL NULL value.
You can't set a zero length value. The Data API SQL statement fails. The following example tries to set id with a zero length value and results in a failure of the SQL statement.

```sql
--parameters "[{"name": "id", "value": "null"}]"
```

You can't set a table name in the SQL statement with a parameter. The Data API follows the rule of the JDBC PreparedStatement.

- The output of the describe statement operation returns the query parameters of an SQL statement.
- Only the execute-statement operation supports SQL statements with parameters.

### Calling the Data API

You can call the Data API or the AWS CLI to run SQL statements on your cluster. The primary operation to run an SQL statement is `ExecuteStatement`. The Data API supports the programming languages that are supported by the AWS SDK. For more information on these, see Tools to Build on AWS.

To see code examples of calling the Data API, see Getting Started with Redshift Data API in GitHub. This repository has examples of using AWS Lambda to access Amazon Redshift data from Amazon EC2, AWS Glue Data Catalog, and Amazon SageMaker. Example programming languages include Python, Go, Java, and Javascript.

### Calling the Data API with the AWS CLI

You can call the Data API using the AWS CLI.

The following examples use the AWS CLI to call the Data API. To run the examples, edit the parameter values to match your environment. These examples demonstrate a few of the Data API operations. For more information, see the AWS CLI Command Reference.

Commands in the following examples have been split and formatted for readability.

#### To run an SQL statement

To run an SQL statement, use the `aws redshift-data execute-statement` AWS CLI command.

The following AWS CLI command runs an SQL statement and returns an identifier to fetch the results. This example uses the AWS Secrets Manager authentication method.

```bash
aws redshift-data execute-statement
  --region us-west-2
  --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hRgPWN
  --cluster-identifier mycluster-test
  --sql "select * from stl_query limit 1"
  --database dev
```

The following is an example of the response.

```json
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598323175.823,
  "Database": "dev",
  "Error": null,
  "ResponseMetadata": {
    "HTTPStatusCode": 200,
    "RequestId": "df281a16-7d0e-43f0-83a6-5d3da1b9a441",
    "RetryAttempts": 0
  }}
```
The following AWS CLI command runs an SQL statement and returns an identifier to fetch the results. This example uses the temporary credentials authentication method.

```
aws redshift-data execute-statement
  --region us-west-2
  --db-user myuser
  --cluster-identifier mycluster-test
  --database dev
  --sql "select * from stl_query limit 1"
```

The following is an example of the response.

```
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598306924.632,
  "Database": "dev",
  "DbUser": "myuser",
  "Id": "d9b6c0c9-0747-4bf4-b142-e883122f766"
}
```

To run an SQL statement with parameters

To run an SQL statement, use the `aws redshift-data execute-statement` AWS CLI command.

The following AWS CLI command runs an SQL statement and returns an identifier to fetch the results. This example uses the AWS Secrets Manager authentication method. The SQL text has named parameters `colname` and `distance`. In this case, the column name in the table is `ratecode` and the distance used in the predicate is 5. Values for named parameters for the SQL statement are specified in the parameters option.

```
aws redshift-data execute-statement
  --region us-west-2
  --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hKgPWn
  --cluster-identifier mycluster-test
  --sql "SELECT :colname, COUNT(*) FROM demo_table WHERE trip_distance > :distance"
  --parameters "["name": "colname", "value": "ratecode"], ["name": "distance", "value": 5]"
  --database dev
```

The following is an example of the response.

```
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598323175.823,
  "Database": "dev",
  "Id": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814",
}
```

The following example uses the `EVENT` table from the sample database. For more information, see `EVENT` table in the Amazon Redshift Database Developer Guide.
If you don’t already have the EVENT table in your database, you can create one using the Data API as follows:

```
aws redshift-data execute-statement
   --database dev
   --cluster-id my-test-cluster
   --db-user awsuser
   --sql "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)"
```

The following command inserts one row into the EVENT table.

```
aws redshift-data execute-statement
   --database dev
   --cluster-id my-test-cluster
   --db-user awsuser
   --sql "insert into event
     values(:eventid, :venueid::smallint, :catid, :dateid, :eventname, :starttime)"
   --parameters "[{"name": "eventid", "value": "1"}, {"name": "venueid", "value": "1"},
      {"name": "catid", "value": "1"},
      {"name": "dateid", "value": "1"},
      {"name": "eventname", "value": "event 1"},
      {"name": "starttime", "value": "2022-02-22"}]"
```

The following command inserts a second row into the EVENT table. This example demonstrates the following:

- The parameter named id is used four times in the SQL text.
- Implicit type conversion is applied automatically when inserting parameter starttime.
- The venueid column is type cast to SMALLINT data type.
- Character strings that represent the DATE data type are implicitly converted into the TIMESTAMP data type.
- Comments can be used within SQL text.

```
aws redshift-data execute-statement
   --database dev
   --cluster-id my-test-cluster
   --db-user awsuser
   --sql "insert into event values(:id, :id::smallint, :id, :id, :eventname, :starttime) /
     this is comment, and it won’t apply parameterization for :id, :eventname or :starttime here*/"
   --parameters "[{{"name": "eventname", "value": "event 2"}},
      {{"name": "starttime", "value": "2022-02-22"}}]"
```

The following shows the two inserted rows:

```
     eventid | venueid | catid | dateid | eventname |      starttime
-----------------|---------|-------|--------|-----------|---------------------
     246
```
The following command uses a named parameter in a WHERE clause to retrieve the row where `eventid` is 1.

```bash
aws redshift-data execute-statement
  --database dev
  --cluster-id my-test-cluster
  --db-user awsuser
  --sql "select * from event where eventid=:id"
  --parameters "[{"name": "id", "value": "1"}]
```

Run the following command to get the SQL results of the previous SQL statement:

```bash
aws redshift-data get-statement-result --id 7529ad05-b905-4d71-9ec6-8b333836eb5a
```

Provides the following results:

```json
{
  "Records": [
    {
      "longValue": 1
    },
    {
      "longValue": 1
    },
    {
      "longValue": 1
    },
    {
      "longValue": 1
    },
    {
      "stringValue": "event 1"
    },
    {
      "stringValue": "2022-02-22 00:00:00.0"
    }
  ],
  "ColumnMetadata": [
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": true,
      "label": "eventid",
      "length": 0,
      "name": "eventid",
      "nullable": 0,
      "precision": 10,
      "scale": 0,
      "schemaName": "public",
      "tableName": "event",
      "typeName": "int4"
    }
  ]
}
```


```json
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": true,
  "label": "venueid",
  "length": 0,
  "name": "venueid",
  "nullable": 0,
  "precision": 5,
  "scale": 0,
  "schemaName": "public",
  "tableName": "event",
  "typeName": "int2"
},
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": true,
  "label": "catid",
  "length": 0,
  "name": "catid",
  "nullable": 0,
  "precision": 5,
  "scale": 0,
  "schemaName": "public",
  "tableName": "event",
  "typeName": "int2"
},
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": true,
  "label": "dateid",
  "length": 0,
  "name": "dateid",
  "nullable": 0,
  "precision": 5,
  "scale": 0,
  "schemaName": "public",
  "tableName": "event",
  "typeName": "int2"
},
{
  "isCaseSensitive": true,
  "isCurrency": false,
  "isSigned": false,
  "label": "eventname",
  "length": 0,
  "name": "eventname",
  "nullable": 1,
  "precision": 200,
  "scale": 0,
  "schemaName": "public",
  "tableName": "event",
  "typeName": "varchar"
},
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": false,
  "label": "starttime",
  "length": 0,
  "name": "starttime",
  "nullable": 1,
  "precision": 29,
  "scale": 0,
  "schemaName": "public",
  "tableName": "event",
  "typeName": "varchar"
}
```
To run multiple SQL statements

To run multiple SQL statements with one command, use the `aws redshift-data batch-execute-statement` AWS CLI command.

The following AWS CLI command runs three SQL statements and returns an identifier to fetch the results. This example uses the temporary credentials authentication method.

```bash
aws redshift-data batch-execute-statement
  --region us-west-2
  --db-user myuser
  --cluster-identifier mycluster-test
  --database dev
  --sqls "set timezone to BST" "select * from mytable" "select * from another_table"
```

The following is an example of the response.

```json
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598306924.632,
  "Database": "dev",
  "DbUser": "myuser",
  "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766"
}
```

To list metadata about SQL statements

To list metadata about SQL statements, use the `aws redshift-data list-statements` AWS CLI command. Authorization to run this command is based on the caller's IAM permissions.

The following AWS CLI command lists SQL statements that ran.

```bash
aws redshift-data list-statements
  --region us-west-2
  --status ALL
```

The following is an example of the response.

```json
{
  "Statements": [
    {
      "CreatedAt": 1598306924.632,
      "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766",
      "QueryString": "select * from stl_query limit 1",
      "Status": "FINISHED",
      "UpdatedAt": 1598306926.667
    },
    {
      "CreatedAt": 1598311717.437,
      "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766",
      "QueryString": "select * from another_table",
      "Status": "FINISHED",
      "UpdatedAt": 1598311717.437
    }
  ]
}
```
To describe metadata about an SQL statement

To get descriptions of metadata for an SQL statement, use the `aws redshift-data describe-statement` AWS CLI command. Authorization to run this command is based on the caller's IAM permissions.

The following AWS CLI command describes an SQL statement.

```bash
aws redshift-data describe-statement
  --id d9b6c0c9-0747-4bf4-b142-e8883122f766
  --region us-west-2
```

The following is an example of the response.

```json
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598306924.632,
  "Duration": 1095981511,
  "Id": "d9b6c0c9-0747-4bf4-b142-e8883122f766",
  "QueryString": "select * from stl_query limit 1",
  "RedshiftPid": 20859,
  "RedshiftQueryId": 48879,
  "ResultRows": 1,
  "ResultSize": 4489,
  "Status": "FINISHED",
  "UpdatedAt": 1598306926.667
}
```

The following is an example of a `describe-statement` response after running a `batch-execute-statement` command with multiple SQL statements.

```json
{
  "ClusterIdentifier": "mayo",
  "CreatedAt": 1623979777.126,
  "Duration": 6591877,
  "HasResultSet": true,
  "Id": "db906c76-fa6e-4cdf-8c5f-4de1ff9b7652",
  "RedshiftPid": 31459,
  "UniqueId": "d9b6c0c9-0747-4bf4-b142-e8883122f766",
  "WorkUnitId": "1623979777.126-1-
```
To fetch the results of an SQL statement

To fetch the result from an SQL statement that ran, use the `redshift-data get-statement-result` AWS CLI command. You can provide an ID that you receive in response to `execute-statement` or `batch-execute-statement`. The ID value for an SQL statement run by `batch-execute-statement` can be retrieved in the result of `describe-statement` and is suffixed by a colon and sequence number such as `b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:1`. If you run multiple SQL statements with `batch-execute-statement`, each SQL statement has an ID value as shown in `describe-statement`. Authorization to run this command is based on the caller's IAM permissions.

The following statement returns the result of an SQL statement run by `execute-statement`.

```bash
aws redshift-data get-statement-result
  --id d9b6c0c9-0747-4bf4-b142-e8883122f766
  --region us-west-2
```

The following statement returns the result of the second SQL statement run by `batch-execute-statement`.

```bash
aws redshift-data get-statement-result
  --id b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2
  --region us-west-2
```

The following is an example of the response to a call to `get-statement-result`.

```json
{
        "RedshiftQueryId": 0,
        "ResultRows": 2,
        "ResultSize": 22,
        "Status": "FINISHED",
        "SubStatements": [
            {
                "CreatedAt": 1623979777.274,
                "Duration": 3396637,
                "HasResultSet": true,
                "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:1",
                "QueryString": "select 1;",
                "RedshiftQueryId": -1,
                "ResultRows": 1,
                "ResultSize": 11,
                "Status": "FINISHED",
                "UpdatedAt": 1623979777.903
            },
            {
                "CreatedAt": 1623979777.274,
                "Duration": 3195240,
                "HasResultSet": true,
                "Id": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652:2",
                "QueryString": "select 2;",
                "RedshiftQueryId": -1,
                "ResultRows": 1,
                "ResultSize": 11,
                "Status": "FINISHED",
                "UpdatedAt": 1623979778.076
            }
        ],
        "UpdatedAt": 1623979778.183
    }
```
"ColumnMetadata": [
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": true,
      "label": "userid",
      "length": 0,
      "name": "userid",
      "nullable": 0,
      "precision": 10,
      "scale": 0,
      "schemaName": "",
      "tableName": "stll_query",
      "typeName": "int4"
    },
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": true,
      "label": "query",
      "length": 0,
      "name": "query",
      "nullable": 0,
      "precision": 10,
      "scale": 0,
      "schemaName": "",
      "tableName": "stll_query",
      "typeName": "int4"
    },
    {
      "isCaseSensitive": true,
      "isCurrency": false,
      "isSigned": false,
      "label": "label",
      "length": 0,
      "name": "label",
      "nullable": 0,
      "precision": 320,
      "scale": 0,
      "schemaName": "",
      "tableName": "stll_query",
      "typeName": "bpchar"
    },
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": true,
      "label": "xid",
      "length": 0,
      "name": "xid",
      "nullable": 0,
      "precision": 19,
      "scale": 0,
      "schemaName": "",
      "tableName": "stll_query",
      "typeName": "int8"
    },
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": true,
      "label": "pid",
      "length": 0,
      "name": "pid",
      "nullable": 0,
      "precision": 10,
      "scale": 252
    }
  ]
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>scale</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>schemaName</td>
<td></td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>tableName</td>
<td></td>
<td>&quot;stll_query&quot;</td>
</tr>
<tr>
<td>typeName</td>
<td></td>
<td>&quot;int4&quot;</td>
</tr>
<tr>
<td>isCaseSensitive</td>
<td></td>
<td>true</td>
</tr>
<tr>
<td>isCurrency</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>isSigned</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>label</td>
<td>&quot;database&quot;</td>
<td></td>
</tr>
<tr>
<td>length</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>name</td>
<td>&quot;database&quot;</td>
<td></td>
</tr>
<tr>
<td>nullable</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>precision</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>scale</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>schemaName</td>
<td></td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>tableName</td>
<td></td>
<td>&quot;stll_query&quot;</td>
</tr>
<tr>
<td>typeName</td>
<td></td>
<td>&quot;bpchar&quot;</td>
</tr>
<tr>
<td>isCaseSensitive</td>
<td></td>
<td>true</td>
</tr>
<tr>
<td>isCurrency</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>isSigned</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>label</td>
<td>&quot;querytxt&quot;</td>
<td></td>
</tr>
<tr>
<td>length</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>name</td>
<td>&quot;querytxt&quot;</td>
<td></td>
</tr>
<tr>
<td>nullable</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>precision</td>
<td></td>
<td>4000</td>
</tr>
<tr>
<td>scale</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>schemaName</td>
<td></td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>tableName</td>
<td></td>
<td>&quot;stll_query&quot;</td>
</tr>
<tr>
<td>typeName</td>
<td></td>
<td>&quot;bpchar&quot;</td>
</tr>
<tr>
<td>isCaseSensitive</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>isCurrency</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>isSigned</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>label</td>
<td>&quot;starttime&quot;</td>
<td></td>
</tr>
<tr>
<td>length</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>name</td>
<td>&quot;starttime&quot;</td>
<td></td>
</tr>
<tr>
<td>nullable</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>precision</td>
<td></td>
<td>29</td>
</tr>
<tr>
<td>scale</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>schemaName</td>
<td></td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>tableName</td>
<td></td>
<td>&quot;stll_query&quot;</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>93</td>
</tr>
<tr>
<td>typeName</td>
<td></td>
<td>&quot;timestamp&quot;</td>
</tr>
<tr>
<td>isCaseSensitive</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>isCurrency</td>
<td></td>
<td>false</td>
</tr>
<tr>
<td>isSigned</td>
<td></td>
<td>true</td>
</tr>
<tr>
<td>label</td>
<td>&quot;endtime&quot;</td>
<td></td>
</tr>
<tr>
<td>length</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>name</td>
<td>&quot;endtime&quot;</td>
<td></td>
</tr>
<tr>
<td>nullable</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>precision</td>
<td></td>
<td>29</td>
</tr>
<tr>
<td>scale</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>schemaName</td>
<td></td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>tableName</td>
<td></td>
<td>&quot;stll_query&quot;</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>93</td>
</tr>
<tr>
<td>typeName</td>
<td></td>
<td>&quot;timestamp&quot;</td>
</tr>
</tbody>
</table>
"label": "aborted",
"length": 0,
"name": "aborted",
"nullable": 0,
"precision": 10,
"scale": 0,
"schemaName": 
"tableName": "stll_query",
"typeName": "int4"
},
{
"isCaseSensitive": false,
"isCurrency": false,
"isSigned": true,
"label": "insert_pristine",
"length": 0,
"name": "insert_pristine",
"nullable": 0,
"precision": 10,
"scale": 0,
"schemaName": 
"tableName": "stll_query",
"typeName": "int4"
},
{
"isCaseSensitive": false,
"isCurrency": false,
"isSigned": true,
"label": "concurrency_scaling_status",
"length": 0,
"name": "concurrency_scaling_status",
"nullable": 0,
"precision": 10,
"scale": 0,
"schemaName": 
"tableName": "stll_query",
"typeName": "int4"
},
]
],
"Records": [
[
  {
    "longValue": 1
  },
  {
    "longValue": 3
  },
  {
    "stringValue": "health"
  },
  {
    "longValue": 1023
  },
  {
    "longValue": 15279
  },
  {
    "stringValue": "dev"
  },
  {
    "stringValue": "select system_status from stv_gui_status;"
  },
  {
    "stringValue": "2020-08-21 17:33:51.88712"
  }
]
To describe a table

To get metadata that describes a table, use the `aws redshift-data describe-table` AWS CLI command.

The following AWS CLI command runs an SQL statement and returns metadata that describes a table. This example uses the AWS Secrets Manager authentication method.

```
aws redshift-data describe-table
  --region us-west-2
  --cluster-identifier mycluster-test
  --database dev
  --schema information_schema
  --table sql_features
  --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hKgPWn
```

The following is an example of the response.

```
{
  "ColumnList": [
  {
    "isCaseSensitive": false,
    "isCurrency": false,
    "isSigned": false,
    "length": 2147483647,
    "name": "feature_id",
    "nullable": 1,
    "precision": 2147483647,
    "scale": 0,
    "schemaName": "information_schema",
    "tableName": "sql_features",
    "typeName": "character_data"
  },
  {
    "isCaseSensitive": false,
    "isCurrency": false,
    "isSigned": false,
    "length": 2147483647,
    "name": "feature_name",
    "nullable": 1,
    "precision": 2147483647,
    "scale": 0,
    "schemaName": "information_schema",
    "tableName": "sql_features",
    "typeName": "character_data"
  }
  ]
}
```
The following AWS CLI command runs an SQL statement that describes a table. This example uses the temporary credentials authentication method.

```bash
aws redshift-data describe-table
  --region us-west-2
  --db-user myuser
  --cluster-identifier mycluster-test
  --database dev
  --schema information_schema
  --table sql_features
```

The following is an example of the response.

```json
{
  "ColumnList": [
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": false,
      "length": 2147483647,
      "name": "feature_id",
      "nullable": 1,
      "precision": 2147483647,
      "scale": 0,
      "schemaName": "information_schema",
      "tableName": "sql_features",
      "typeName": "character_data"
    },
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": false,
      "length": 2147483647,
      "name": "feature_name",
      "nullable": 1,
      "precision": 2147483647,
      "scale": 0,
      "schemaName": "information_schema",
      "tableName": "sql_features",
      "typeName": "character_data"
    },
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": false,
      "length": 2147483647,
      "name": "sub_feature_id",
      "nullable": 1,
      "precision": 2147483647,
      "scale": 0,
      "schemaName": "information_schema",
      "tableName": "sql_features",
      "typeName": "character_data"
    },
    {
      "isCaseSensitive": false,
      "isCurrency": false,
      "isSigned": false,
      "length": 2147483647,
      "name": "sub_feature_name",
      "nullable": 1,
      "precision": 2147483647,
      "scale": 0,
      "schemaName": "information_schema",
      "tableName": "sql_features",
      "typeName": "character_data"
    }
  ]
}
```
To list the databases in a cluster

To list the databases in a cluster, use the `aws redshift-data list-databases` AWS CLI command.

The following AWS CLI command runs an SQL statement to list databases. This example uses the AWS Secrets Manager authentication method.

```bash
aws redshift-data list-databases
  --region us-west-2
  --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hRgPWhn
  --cluster-identifier mycluster-test
```
The following is an example of the response.

```json
{
    "Databases": [
        "dev"
    ]
}
```

The following AWS CLI command runs an SQL statement to list databases. This example uses the temporary credentials authentication method.

```bash
aws redshift-data list-databases
    --region us-west-2
    --db-user myuser
    --cluster-identifier mycluster-test
    --database dev
```

The following is an example of the response.

```json
{
    "Databases": [
        "dev"
    ]
}
```

**To list the schemas in a database**

To list the schemas in a database, use the `aws redshift-data list-schemas` AWS CLI command.

The following AWS CLI command runs an SQL statement to list schemas in a database. This example uses the AWS Secrets Manager authentication method.

```bash
aws redshift-data list-schemas
    --region us-west-2
    --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hKgPWn
    --cluster-identifier mycluster-test
    --database dev
```

The following is an example of the response.

```json
{
    "Schemas": [
        "information_schema",
        "pg_catalog",
        "pg_internal",
        "public"
    ]
}
```

The following AWS CLI command runs an SQL statement to list schemas in a database. This example uses the temporary credentials authentication method.

```bash
aws redshift-data list-schemas
```
Calling the Data API

```
--region us-west-2
--db-user mysuser
--cluster-identifier mycluster-test
--database dev
```

The following is an example of the response.

```
{
  "Schemas": [  
    "information_schema",
    "pg_catalog",
    "pg_internal",
    "public"
  ]
}
```

**To list the tables in a database**

To list the tables in a database, use the `aws redshift-data list-tables` AWS CLI command.

The following AWS CLI command runs an SQL statement to list tables in a database. This example uses the AWS Secrets Manager authentication method.

```
aws redshift-data list-tables
  --region us-west-2
  --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hRgPWn
  --cluster-identifier mycluster-test
  --database dev
  --schema information_schema
```

The following is an example of the response.

```
{
  "Tables": [  
    {  
      "name": "sql_features",
      "schema": "information_schema",
      "type": "SYSTEM TABLE"
    },
    {  
      "name": "sql_implementation_info",
      "schema": "information_schema",
      "type": "SYSTEM TABLE"
    }
  ]
}
```

The following AWS CLI command runs an SQL statement to list tables in a database. This example uses the temporary credentials authentication method.

```
aws redshift-data list-tables
  --region us-west-2
  --db-user myuser
  --cluster-identifier mycluster-test
  --database dev
  --schema information_schema
```

259
Troubleshooting issues for Amazon Redshift Data API

Use the following sections, titled with common error messages, to help troubleshoot problems that you have with the Data API.

Topics

- Packet for query is too large (p. 260)
- Database response exceeded size limit (p. 260)

Packet for query is too large

If you see an error indicating that the packet for a query is too large, generally the result set returned for a row is too large. The Data API size limit is 64 KB per row in the result set returned by the database.

To solve this issue, make sure that each row in a result set is 64 KB or less.

Database response exceeded size limit

If you see an error indicating that the database response has exceeded the size limit, generally the size of the result set returned by the database was too large. The Data API limit is 100 MB in the result set returned by the database.

To solve this issue, make sure that calls to the Data API return 100 MB of data or less. If you need to return more than 100 MB, you can run multiple statement calls with the `LIMIT` clause in your query.

Scheduling Amazon Redshift Data API operations with Amazon EventBridge

Amazon EventBridge helps you to respond to state changes in your AWS resources. When your resources change state, they automatically send events into an event stream. Events are sent to the account that contains the Amazon Redshift database. You can create rules that match selected events in the stream and route them to targets to take action. You can also use rules to take action on a predetermined schedule. For more information, see the Amazon EventBridge User Guide.

To schedule Data API operations with EventBridge, the associated IAM role must trust the principal for CloudWatch Events (events.amazonaws.com). This role should have the equivalent of the managed policy `AmazonEventBridgeFullAccess` attached. It should also have `AmazonRedshiftDataFullAccess` policy permissions that are managed by the Data API. You can create an IAM role with these permissions.
Monitoring the Data API

Monitoring is an important part of maintaining the reliability, availability, and performance of the Data API and your other AWS solutions. AWS provides the following monitoring tools to watch the Data API, report when something is wrong, and take automatic actions when appropriate:

- Amazon EventBridge can be used to automate your AWS services and respond automatically to system events, such as application availability issues or resource changes. Events from AWS services are delivered to EventBridge in near-real time. You can write simple rules to indicate which events are of interest to you and which automated actions to take when an event matches a rule. For more information, see the Amazon EventBridge User Guide.

- AWS CloudTrail captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.

Topics
- Monitoring events for the Amazon Redshift Data API in Amazon EventBridge (p. 262)
- Logging Amazon Redshift Data API calls with AWS CloudTrail (p. 262)
Monitoring events for the Amazon Redshift Data API in Amazon EventBridge

You can monitor Data API events in EventBridge, which delivers a stream of real-time data from your own applications, software-as-a-service (SaaS) applications, and AWS services. EventBridge routes that data to targets such as AWS Lambda and Amazon SNS. These events are the same as those that appear in CloudWatch Events, which delivers a near-real time stream of system events that describe changes in AWS resources. Events are sent to the account that contains the Amazon Redshift database. For example, if you assume a role in another account, events are sent to that account. For more information, see Amazon EventBridge events in the Amazon EventBridge User Guide.

Data API events are sent when the ExecuteStatement or BatchExecuteStatement API operation sets the WithEvent option to true. The state field of the event contains one of the following values:

- ABORTED – The query run was stopped by the user.
- FAILED – The query run failed.
- FINISHED – The query has finished running.

Events are delivered on a guaranteed basis. For more information, see Events from AWS services in the Amazon EventBridge User Guide.

Example for Data API finished event

The following example shows an event for the Data API when the ExecuteStatement API operation finished. In this example, a statement named test.testtable finished running.

```
{
    "version": "0",
    "id": "18e7079c-dd4b-dd64-caf9-e2a31640dab0",
    "detail-type": "Redshift Data Statement Status Change",
    "source": "aws.redshift-data",
    "account": "123456789012",
    "time": "2020-10-01T21:14:26Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:redshift:us-east-1:123456789012:cluster:redshift-cluster-1"
    ],
    "detail": {
        "principal": "arn:aws:iam::123456789012:user/myuser",
        "statementName": "test.testtable",
        "statementId": "dd2e1ec9-2ee3-49a0-819f-905fa7d75a4a",
        "redshiftQueryId": -1,
        "state": "FINISHED",
        "rows": 1,
        "expireAt": 1601673265
    }
}
```

Logging Amazon Redshift Data API calls with AWS CloudTrail

Amazon Redshift Data API is integrated with AWS CloudTrail. CloudTrail is a service that provides a record of actions taken by a user, role, or an AWS service in Amazon Redshift Data API. CloudTrail captures all API calls for Amazon Redshift Data API as events. The calls captured include calls from the Amazon Redshift Data API console and code calls to the Amazon Redshift Data API operations.

If you create a CloudTrail trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon Redshift Data API. If you don't configure a trail, you can still
view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine certain things. These include the request that was made to Amazon Redshift Data API, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

**Working with Data API information in CloudTrail**

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Redshift Data API, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History in the AWS CloudTrail User Guide.

For an ongoing record of events in your AWS account, including events for Amazon Redshift Data API, create a trail. A **trail** enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following in the AWS CloudTrail User Guide:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All Amazon Redshift Data API actions are logged by CloudTrail and are documented in the Amazon Redshift Data API Reference. For example, calls to the ExecuteStatement, GetStatementResults and CancelStatement actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or IAM user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see CloudTrail userIdentity Element in the AWS CloudTrail User Guide.

**Understanding log file entries for Data API**

A **trail** is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An **event** represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the ExecuteStatement action.

```json
{
   "eventVersion": "1.05",
   "userIdentity": {
      "type": "IAMUser",
      "principalId": "AKIAIOSFODNN7EXAMPLE:janedoe",
      "arn": "arn:aws:sts::123456789012:user/janedoe",
```
"accountId":"123456789012",
"accessKeyId":"AKIAI44QH8DHBEXAMPLE",
"userName": "janedoe"
},
"eventTime":"2020-08-19T17:55:59Z",
"eventSource":"redshift-data.amazonaws.com",
"eventName":"ExecuteStatement",
"awsRegion":"us-east-1",
"sourceIPAddress":"192.0.2.0",
"userAgent":"aws-cli/1.18.118 Python/3.6.10
Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.41",
"requestParameters":{
  "clusterIdentifier":"example-cluster-identifier",
  "database":"example-database-name",
  "dbUser":"example_db_user_name",
  "sql":"***OMITTED***"
},
"responseElements":{
  "clusterIdentifier":"example-cluster-identifier",
  "createdAt":"Aug 19, 2020 5:55:58 PM",
  "database":"example-database-name",
  "dbUser":"example_db_user_name",
  "id":"5c52b37b-9e07-40c1-98de-12ccd1419be7"
},
"requestID":"00c924d3-652e-4939-8a7a-cd0612eb8ac",
"eventID":"c1fb7076-102f-43e5-9ec9-40820bcc1175",
"readOnly":false,
"eventType":"AwsApiCall",
"recipientAccountID":"123456789012"}
Enhanced VPC routing in Amazon Redshift

When you use Amazon Redshift enhanced VPC routing, Amazon Redshift forces all COPY and UNLOAD traffic between your cluster and your data repositories through your virtual private cloud (VPC) based on the Amazon VPC service. By using enhanced VPC routing, you can use standard VPC features, such as VPC security groups, network access control lists (ACLs), VPC endpoints, VPC endpoint policies, internet gateways, and Domain Name System (DNS) servers, as described in the Amazon VPC User Guide. You use these features to tightly manage the flow of data between your Amazon Redshift cluster and other resources. When you use enhanced VPC routing to route traffic through your VPC, you can also use VPC flow logs to monitor COPY and UNLOAD traffic.

If enhanced VPC routing is not enabled, Amazon Redshift routes traffic through the internet, including traffic to other services within the AWS network.

**Important**
Because enhanced VPC routing affects the way that Amazon Redshift accesses other resources, COPY and UNLOAD commands might fail unless you configure your VPC correctly. You must specifically create a network path between your cluster's VPC and your data resources, as described following.

When you run a COPY or UNLOAD command on a cluster with enhanced VPC routing enabled, your VPC routes the traffic to the specified resource using the *strictest*, or most specific, network path available.

For example, you can configure the following pathways in your VPC:

- **VPC endpoints** – For traffic to an Amazon S3 bucket in the same AWS Region as your cluster, you can create a VPC endpoint to direct traffic directly to the bucket. When you use VPC endpoints, you can attach an endpoint policy to manage access to Amazon S3. For more information about using endpoints with Amazon Redshift, see Working with VPC endpoints (p. 266).

- **NAT gateway** – You can connect to an Amazon S3 bucket in another AWS Region, and you can connect to another service within the AWS network. You can also access a host instance outside the AWS network. To do so, configure a network address translation (NAT) gateway, as described in the Amazon VPC User Guide.

- **Internet gateway** – To connect to AWS services outside your VPC, you can attach an internet gateway to your VPC subnet, as described in the Amazon VPC User Guide. To use an internet gateway, your cluster must have a public IP to allow other services to communicate with your cluster.

For more information, see VPC Endpoints in the Amazon VPC User Guide.

There is no additional charge for using enhanced VPC routing. You might incur additional data transfer charges for certain operations. These include such operations as UNLOAD to Amazon S3 in a different AWS Region, COPY from Amazon EMR, or Secure Shell (SSH) with public IP addresses. For more information about pricing, see Amazon EC2 Pricing.

**Topics**
- Working with VPC endpoints (p. 266)
- Enabling enhanced VPC routing (p. 266)
You can use a VPC endpoint to create a managed connection between your Amazon Redshift cluster in a VPC and Amazon Simple Storage Service (Amazon S3). When you do, COPY and UNLOAD traffic between your cluster and your data on Amazon S3 stays in your Amazon VPC. You can attach an endpoint policy to your endpoint to more closely manage access to your data. For example, you can add a policy to your VPC endpoint that permits unloading data only to a specific Amazon S3 bucket in your account.

**Important**
Currently, Amazon Redshift supports VPC endpoints only for connecting to Amazon S3. When Amazon VPC adds support for other AWS services to use VPC endpoints, Amazon Redshift will support those VPC endpoint connections also. To connect to an Amazon S3 bucket using a VPC endpoint, the Amazon Redshift cluster and the Amazon S3 bucket that it connects to must be in the same AWS Region.

To use VPC endpoints, create a VPC endpoint for the VPC that your cluster is in and then enable enhanced VPC routing for your cluster. You can enable enhanced VPC routing when you create your cluster in a VPC, or you can modify a cluster in a VPC to use enhanced VPC routing.

A VPC endpoint uses route tables to control the routing of traffic between a cluster in the VPC and Amazon S3. All clusters in subnets associated with the specified route tables automatically use that endpoint to access the service.

Your VPC uses the most specific, or most restrictive, route that matches your cluster's traffic to determine how to route the traffic. For example, suppose that you have a route in your route table for all internet traffic (0.0.0.0/0) that points to an internet gateway and an Amazon S3 endpoint. In this case, the endpoint route takes precedence for all traffic destined for Amazon S3. This is because the IP address range for the Amazon S3 service is more specific than 0.0.0.0/0. In this example, all other internet traffic goes to your internet gateway, including traffic that's destined for Amazon S3 buckets in other AWS Regions.

For more information about creating endpoints, see [VPC Endpoints](#) in the Amazon VPC User Guide.

You use endpoint policies to control access from your cluster to the Amazon S3 buckets that hold your data files. By default, the Create Endpoint wizard attaches an endpoint policy doesn't further restrict access from any user or service within the VPC. For more specific control, you can optionally attach a custom endpoint policy. For more information, see [Using Endpoint Policies](#) in the Amazon VPC User Guide.

There is no additional charge for using endpoints. Standard charges for data transfer and resource usage apply. For more information about pricing, see [Amazon EC2 Pricing](#).

### Enabling enhanced VPC routing

You can enable enhanced VPC routing when you create a cluster, or you can modify an existing cluster to enable enhanced VPC routing.

To work with enhanced VPC routing, your cluster must meet the following requirements and constraints:

- Your cluster must be in a VPC.

If you attach an Amazon S3 VPC endpoint, your cluster uses the VPC endpoint only for access to Amazon S3 buckets in the same AWS Region. To access buckets in another AWS Region (not using the
VPC endpoint) or to access other AWS services, make your cluster publicly accessible or use a network address translation (NAT) gateway. For more information, see Creating a cluster in a VPC (p. 81).

- You must enable Domain Name Service (DNS) resolution in your VPC. Alternatively, if you're using your own DNS server, make sure that DNS requests to Amazon S3 are resolved correctly to the IP addresses that are maintained by AWS. For more information, see Using DNS with Your VPC in the Amazon VPC User Guide.

- DNS hostnames must be enabled in your VPC. DNS hostnames are enabled by default.

- Your VPC endpoint policies must allow access to any Amazon S3 buckets used with COPY, UNLOAD, or CREATE LIBRARY calls in Amazon Redshift, including access to any manifest files involved. For COPY from remote hosts, your endpoint policies must allow access to each host machine. For more information, see IAM Permissions for COPY, UNLOAD, and CREATE LIBRARY in the Amazon Redshift Database Developer Guide.

To create a cluster with enhanced VPC routing

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose Create cluster and enter the Cluster details properties.
3. To display the Additional configurations section, choose to switch off Use defaults.
4. To enable Enhanced VPC routing select Enabled to force cluster traffic through the VPC.
5. Choose Create cluster to create the cluster. The cluster might take several minutes to be ready to use.

You can use the following AWS Command Line Interface (AWS CLI) operations for Amazon Redshift to enable enhanced VPC routing:

- create-cluster
- modify-cluster

You can use the following Amazon Redshift API actions to enable enhanced VPC routing:

- CreateCluster
- ModifyCluster

Using Amazon Redshift Spectrum with enhanced VPC routing

Amazon Redshift enhanced VPC routing routes specific traffic through your VPC. All traffic between your cluster and your Amazon S3 buckets is forced to pass through your Amazon VPC. Redshift Spectrum runs on AWS-managed resources that are owned by Amazon Redshift. Because these resources are outside your VPC, Redshift Spectrum doesn't use enhanced VPC routing.

When your cluster is configured to use enhanced VPC routing, traffic between Redshift Spectrum and Amazon S3 is securely routed through the AWS private network, outside of your VPC. In-flight traffic is signed using Amazon Signature Version 4 protocol (SIGv4) and encrypted using HTTPS. This traffic is authorized based on the IAM role that is attached to your Amazon Redshift cluster. To further manage Redshift Spectrum traffic, you can modify your cluster's IAM role and your policy attached to the Amazon S3 bucket. You might also need to configure your VPC to allow your cluster to access AWS Glue or Athena, as detailed following.
Considerations for using enhanced VPC routing for Redshift Spectrum

Following are considerations when using Redshift Spectrum enhanced VPC routing:

- Bucket access policies (p. 268)
- Cluster IAM role (p. 268)
- Logging and auditing Amazon S3 access (p. 269)
- Access to AWS Glue or Amazon Athena (p. 269)

Bucket access policies

You can control access to data in your Amazon S3 buckets by using a bucket policy attached to the bucket and by using an IAM role attached to the cluster.

Redshift Spectrum can't access data stored in Amazon S3 buckets that use a bucket policy that restricts access to only specified VPC endpoints. Instead, use a bucket policy that restricts access to only specific principals, such as a specific AWS account or specific users.

For the IAM role that is granted access to the bucket, use a trust relationship that allows the role to be assumed only by the Amazon Redshift service principal. When attached to your cluster, the role can be used only in the context of Amazon Redshift and can't be shared outside of the cluster. For more information, see Restricting access to IAM roles (p. 406).

To use Redshift Spectrum, no IAM policies blocking the use of presigned URLs can be in place.

The following example bucket policy permits access to the specified bucket only from traffic originated by Redshift Spectrum owned by AWS account 123456789012.

```json
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"BucketPolicyForSpectrum",
      "Effect":"Allow",
      "Principal": {
        "AWS": ["arn:aws:iam::123456789012:root"]
      },
      "Action": ["s3:GetObject","s3:List*"],
      "Resource": ["arn:aws:s3:::examplebucket/*"],
      "Condition": {"StringEquals": {"aws:UserAgent": "AWS Redshift/Spectrum"}}
    }
  ]
}
```

Cluster IAM role

The role attached to your cluster should have a trust relationship that permits it to be assumed only by the Amazon Redshift service, as shown following.

```json
{
  "Version": "2012-10-17",
  "Statement": [ 
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
    } 
  ]
}
```
You can add a policy to the cluster role that prevents COPY and UNLOAD access to a specific bucket. The following policy permits traffic to the specified bucket only from Redshift Spectrum.

```json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:Get*", "s3:List*"],
    "Resource": "arn:aws:s3:::myBucket/*",
    "Condition": {"StringEquals": {"aws:UserAgent": "AWS Redshift/Spectrum"}}
  }]
}
```

For more information, see IAM Policies for Amazon Redshift Spectrum in the Amazon Redshift Database Developer Guide.

Logging and auditing Amazon S3 access

One benefit of using Amazon Redshift enhanced VPC routing is that all COPY and UNLOAD traffic is logged in the VPC flow logs. Traffic originating from Redshift Spectrum to Amazon S3 doesn't pass through your VPC, so it isn't logged in the VPC flow logs. When Redshift Spectrum accesses data in Amazon S3, it performs these operations in the context of the AWS account and respective role privileges. You can log and audit Amazon S3 access using server access logging in AWS CloudTrail and Amazon S3.

**AWS CloudTrail Logs**

To trace all access to objects in Amazon S3, including Redshift Spectrum access, enable CloudTrail logging for Amazon S3 objects.

You can use CloudTrail to view, search, download, archive, analyze, and respond to account activity across your AWS infrastructure. For more information, see Getting Started with CloudTrail.

By default, CloudTrail tracks only bucket-level actions. To track object-level actions (such as GetObject), enable data and management events for each logged bucket.

**Amazon S3 Server Access Logging**

Server access logging provides detailed records for the requests that are made to a bucket. Access log information can be useful in security and access audits. For more information, see How to Enable Server Access Logging in the Amazon Simple Storage Service User Guide.

For more information, see the AWS Security blog post How to Use Bucket Policies and Apply Defense-in-Depth to Help Secure Your Amazon S3 Data.

Access to AWS Glue or Amazon Athena

Redshift Spectrum accesses your data catalog in AWS Glue or Athena. Another option is to use a dedicated Hive metastore for your data catalog.

To enable access to AWS Glue or Athena, configure your VPC with an internet gateway or NAT gateway. Configure your VPC security groups to allow outbound traffic to the public endpoints for AWS Glue and Athena. Alternatively, you can configure an interface VPC endpoint for AWS Glue to access your AWS
Glue Data Catalog. When you use a VPC interface endpoint, communication between your VPC and AWS Glue is conducted within the AWS network. For more information, see Creating an Interface Endpoint.

You can configure the following pathways in your VPC:

- **Internet gateway** – To connect to AWS services outside your VPC, you can attach an internet gateway to your VPC subnet, as described in the Amazon VPC User Guide. To use an internet gateway, your cluster must have a public IP address to allow other services to communicate with your cluster.

- **NAT gateway** – To connect to an Amazon S3 bucket in another AWS Region or to another service within the AWS network, configure a network address translation (NAT) gateway, as described in the Amazon VPC User Guide. Use this configuration also to access a host instance outside the AWS network.

For more information, see Enhanced VPC routing in Amazon Redshift (p. 265).
Amazon Redshift parameter groups

Overview

In Amazon Redshift, you associate a parameter group with each cluster that you create. A parameter group is a group of parameters that apply to all of the databases that you create in the cluster. These parameters configure database settings such as query timeout and date style.

About parameter groups

Each parameter group has several parameters to configure settings for the database. The list of available parameters depends on the parameter group family to which the parameter group belongs. The parameter group family is the version of the Amazon Redshift engine to which the parameters in the parameter group apply. The format of the parameter group family name is redshift-version where version is the engine version. For example, the current version of the engine is redshift-1.0.

Amazon Redshift provides one default parameter group for each parameter group family. The default parameter group has preset values for each of its parameters, and it cannot be modified. The format of the default parameter group name is default.parameter_group_family, where parameter_group_family is the version of the engine to which the parameter group belongs. For example, the default parameter group for the redshift-1.0 version is named default.redshift-1.0.

Note

At this time, redshift-1.0 is the only version of the Amazon Redshift engine. Consequently, default.redshift-1.0 is the only default parameter group.

If you want to use different parameter values than the default parameter group, you must create a custom parameter group and then associate your cluster with it. Initially, the parameter values in a custom parameter group are the same as in the default parameter group. The initial source for all of the parameters is engine-default because the values are preset by Amazon Redshift. After you change a parameter value, the source changes to user to indicate that the value has been modified from its default value.

Note

The Amazon Redshift console does not display the source of each parameter. You must use the Amazon Redshift API, the AWS CLI, or one of the AWS SDKs to view the source.

For parameter groups that you create, you can modify a parameter value at any time, or you can reset all parameter values to their defaults. You can also associate a different parameter group with a cluster. In some cases, you might modify parameter values in a parameter group that is already associated with a cluster or associate a different parameter group with a cluster. In these cases, you might need to restart the cluster for the updated parameter values to take effect. If the cluster fails and is restarted by Amazon Redshift, your changes are applied at that time. Changes aren't applied if your cluster is restarted during maintenance. For more information, see WLM dynamic and static properties (p. 274).

Default parameter values

The following table shows the default parameter values at a glance with links to more in-depth information about each parameter. These are the default values for the redshift-1.0 parameter group family.
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Value</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto_analyze</td>
<td>true</td>
<td>auto_analyze in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>datestyle</td>
<td>ISO, MDY</td>
<td>datestyle in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>enable_case_sensitive_identifier</td>
<td>false</td>
<td>enable_case_sensitive_identifier in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>enable_user_activity_logging</td>
<td>false</td>
<td>Database audit logging (p. 423) in this guide</td>
</tr>
<tr>
<td>extra_float_digits</td>
<td>0</td>
<td>extra_float_digits in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>max_concurrency_scaling_clusters</td>
<td>1</td>
<td>max_concurrency_scaling_clusters in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>query_group</td>
<td>default</td>
<td>query_group in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>require_ssl</td>
<td>false</td>
<td>Configuring security options for connections (p. 193) in this guide</td>
</tr>
<tr>
<td>search_path</td>
<td>$user, public</td>
<td>search_path in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>statement_timeout</td>
<td>0</td>
<td>statement_timeout in the Amazon Redshift Database Developer Guide</td>
</tr>
<tr>
<td>wlm_json_configuration</td>
<td>[&quot;auto_wlm&quot;:true]</td>
<td>Configuring workload management (p. 273) in this guide</td>
</tr>
<tr>
<td>use_fips_ssl</td>
<td>false</td>
<td>Enable FIPS-compliant SSL mode only if your system is required to be FIPS-compliant.</td>
</tr>
</tbody>
</table>

**Note**

The `max_cursor_result_set_size` parameter is deprecated. For more information about cursor result set size, seeAnd ***Cursor constraints*** in the Amazon Redshift Database Developer Guide.

You can temporarily override a parameter by using the `SET` command in the database. The `SET` command overrides the parameter for the duration of your current session only. In addition to the parameters listed in the preceding table, you can also temporarily adjust the slot count by setting `wlm_query_slot_count` in the database. The `wlm_query_slot_count` parameter is not available for configuration in parameter groups. For more information about adjusting the slot count, see `wlm_query_slot_count` in the Amazon Redshift Database Developer Guide. For more information about temporarily overriding the other parameters, see Modifying the server configuration in the Amazon Redshift Database Developer Guide.

### Configuring parameter values using the AWS CLI

To configure Amazon Redshift parameters by using the AWS CLI, you use the `modify-cluster-parameter-group` command for a specific parameter group. You specify the parameter group to modify in `parameter-group-name`. You use the `parameters` parameter (for the `modify-cluster-parameter-group` command) to specify name/value pairs for each parameter that you want to modify in the parameter group.
Note
There are special considerations when configuring the `wlm_json_configuration` parameter by using the AWS CLI. The examples in this section apply to all of the parameters except `wlm_json_configuration`. For more information about configuring `wlm_json_configuration` by using the AWS CLI, see Configuring workload management (p. 273).

After you modify parameter values, you must reboot any clusters that are associated with the modified parameter group. The cluster status displays applying for `ParameterApplyStatus` while the values are being applied, and then `pending-reboot` after the values have been applied. After you reboot, the databases in your cluster begin to use the new parameter values. For more information about rebooting clusters, see Rebooting a cluster (p. 74).

Note
The `wlm_json_configuration` parameter contains some properties that are dynamic and do not require you to reboot associated clusters for the changes to be applied. For more information about dynamic and static properties, see WLM dynamic and static properties (p. 274).

Syntax
The following syntax shows how to use the `modify-cluster-parameter-group` command to configure a parameter. You specify `parameter_group_name` and replace both `parameter_name` and `parameter_value` with an actual parameter to modify and a value for that parameter. If you want to modify more than one parameter at the same time, separate each parameter and value set from the next with a space.

```
aws redshift modify-cluster-parameter-group --parameter-group-name parameter_group_name --parameters ParameterName=parameter_name,ParameterValue=parameter_value
```

Example
The following example shows how to configure the `statement_timeout` and `enable_user_activity_logging` parameters for the `myclusterparametergroup` parameter group.

Note
For readability purposes, the example is displayed on several lines, but in the actual AWS CLI this is one line.

```
aws redshift modify-cluster-parameter-group --parameter-group-name myclusterparametergroup --parameters ParameterName=statement_timeout,ParameterValue=20000 ParameterName=enable_user_activity_logging,ParameterValue=true
```

You can manage parameter groups using the console. For more information, see Managing parameter groups using the console (p. 283).

Configuring workload management

In Amazon Redshift, you use workload management (WLM) to define the number of query queues that are available, and how queries are routed to those queues for processing. WLM is part of parameter group configuration. A cluster uses the WLM configuration that is specified in its associated parameter group.

When you create a parameter group, the default WLM configuration contains one queue that can run up to five queries concurrently. You can add additional queues and configure WLM properties in each
of them if you want more control over query processing. Each queue that you add has the same default WLM configuration until you configure its properties.

When you add additional queues, the last queue in the configuration is the default queue. Unless a query is routed to another queue based on criteria in the WLM configuration, it is processed by the default queue. You can specify mode and concurrency level (query slots) for the default queue, but you can't specify user groups or query groups for the default queue.

As with other parameters, you cannot modify the WLM configuration in the default parameter group. Clusters associated with the default parameter group always use the default WLM configuration. To modify the WLM configuration, create a new parameter group and then associate that parameter group with any clusters that require your custom WLM configuration.

## WLM dynamic and static properties

The WLM configuration properties are either dynamic or static. You can apply dynamic properties to the database without a cluster reboot, but static properties require a cluster reboot for changes to take effect. For more information about static and dynamic properties, see WLM dynamic and static configuration properties.

### Properties for the `wlm_json_configuration` parameter

You can configure WLM by using the Amazon Redshift console, the AWS CLI, the Amazon Redshift API, or one of the AWS SDKs. WLM configuration uses several properties to define queue behavior, such as memory allocation across queues, the number of queries that can run concurrently in a queue, and so on.

**Note**
The following properties appear with their Amazon Redshift console names, with the corresponding JSON property names in the descriptions.

The following table summarizes whether a property is applicable to automatic WLM or manual WLM.

<table>
<thead>
<tr>
<th>WLM property</th>
<th>Automatic WLM</th>
<th>Manual WLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto WLM</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Enable short query acceleration</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Maximum run time for short queries</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Priority</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Queue type</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Queue name</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Concurrency Scaling mode</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Concurrency</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>User groups</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>User group wildcard</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Query groups</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Query group wildcard</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The following list describes the WLM properties that you can configure for each queue.

**Auto WLM**

Auto WLM set to true enables automatic WLM. Automatic WLM sets the values for **Concurrency on main** and **Memory (%)** to Auto. Amazon Redshift manages query concurrency and memory allocation. The default is true.

JSON property: `auto_wlm`

**Enable short query acceleration**

Short query acceleration (SQA) prioritizes selected short-running queries ahead of longer-running queries. SQA executes short-running queries in a dedicated space, so that SQA queries aren't forced to wait in queues behind longer queries. With SQA, short-running queries begin executing more quickly and users see results sooner. When you enable SQA, you can also specify the maximum run time for short queries. To enable SQA, specify true. The default is false.

JSON property: `short_query_queue`

**Maximum run time for short queries**

When you enable SQA, you can specify 0 to let WLM dynamically set the maximum run time for short queries. Alternatively, you can specify a value of 1–20 seconds, in milliseconds. The default value is 0.

JSON property: `max_execution_time`

**Priority**

Priority sets the priority of queries that run in a queue. To set the priority, **WLM mode** must be set to **Auto WLM**; that is, **auto_wlm** must be true. Priority values can be highest, high, normal, low, and lowest. The default is normal.

JSON property: `priority`

**Queue type**

Queue type designates a queue as used either by **Auto WLM** or **Manual WLM**. Set `queue_type` to either auto or manual. If not specified, the default is manual.

JSON property: `queue_type`

**Queue name**

The name of the queue. You can set the name of the queue based on your business needs. Queue names must be unique within an WLM configuration, are up to 64 alphanumeric characters, underscores or spaces, and can't contain quotation marks. For example, if you have a queue for your ETL queries, you might name it ETL queue. This name is used in metrics, system table values, and the Amazon Redshift console to identify the queue. Queries and reports that use the name from these sources need to be able to handle changes of the name. Previously, the queue names were generated by Amazon Redshift. The default names of queues are Queue 1, Queue 2, to the last queue named Default queue.
Important
If you change a queue name, the QueueName dimension value of WLM queue metrics (such as, WLMQueueLength, WLMQueueWaitTime, WLMQueriesCompletedPerSecond, WLMQueryDuration, WLMRunningQueries, and so on) also changes. So, if you change the name of a queue, you might need to change CloudWatch alarms you have set up.

JSON property: name

Concurrency Scaling mode

To enable concurrency scaling on a queue, set Concurrency Scaling mode to auto. When the number of queries routed to a queue exceeds the queue's configured concurrency, eligible queries go to the scaling cluster. When slots become available, queries run on the main cluster. The default is off.

JSON property: concurrency_scaling

Concurrency

The number of queries that can run concurrently in a manual WLM queue. This property only applies to manual WLM. If concurrency scaling is enabled, eligible queries go to a scaling cluster when a queue reaches the concurrency level (query slots). If concurrency scaling isn't enabled, queries wait in the queue until a slot becomes available. The range is between 1 and 50.

JSON property: query_concurrency

User Groups

A comma-separated list of user group names. When members of the user group run queries in the database, their queries are routed to the queue that is associated with their user group.

JSON property: user_group

User Group Wildcard

A Boolean value that indicates whether to enable wildcards for user groups. If this is 0, wildcards are disabled; if this is 1, wildcards are enabled. When wildcards are enabled, you can use "*" or "?" to specify multiple user groups when running queries. For more information, see Wildcards.

JSON property: user_group_wild_card

Query Groups

A comma-separated list of query groups. When members of the query group run queries in the database, their queries are routed to the queue that is associated with their query group.

JSON property: query_group

Query Group Wildcard

A Boolean value that indicates whether to enable wildcards for query groups. If this is 0, wildcards are disabled; if this is 1, wildcards are enabled. When wildcards are enabled, you can use "*" or "?" to specify multiple query groups when running queries. For more information, see Wildcards.

JSON property: query_group_wild_card

Timeout (ms)

WLM timeout (max_execution_time) is deprecated. It is not available when using automatic WLM. Instead, create a query monitoring rule (QMR) using query_execution_time to limit the elapsed execution time for a query. For more information, see WLM query monitoring rules.

The maximum time, in milliseconds, that queries can run before being canceled. In some cases, a read-only query, such as a SELECT statement, might be canceled due to a WLM timeout. In these cases, WLM attempts to route the query to the next matching queue based on the WLM queue
assignment rules. If the query doesn't match any other queue definition, the query is canceled; it isn't assigned to the default queue. For more information, see WLM query queue hopping. WLM timeout doesn't apply to a query that has reached the returning state. To view the state of a query, see the STV_WLM_QUERY_STATE system table.

JSON property: max_execution_time

**Memory (%)**

The percentage of memory to allocate to the queue. If you specify a memory percentage for at least one of the queues, you must specify a percentage for all other queues, up to a total of 100 percent. If your memory allocation is below 100 percent across all of the queues, the unallocated memory is managed by the service. The service can temporarily give this unallocated memory to a queue that requests additional memory for processing.

JSON property: memory_percent_to_use

**Query Monitoring Rules**

You can use WLM query monitoring rules to continuously monitor your WLM queues for queries based on criteria, or predicates, that you specify. For example, you might monitor queries that tend to consume excessive system resources, and then initiate a specified action when a query exceeds your specified performance boundaries.

**Note**

If you choose to create rules programmatically, we strongly recommend using the console to generate the JSON that you include in the parameter group definition.

You associate a query monitoring rule with a specific query queue. You can have up to 25 rules per queue, and the total limit for all queues is 25 rules.

JSON property: rules

JSON properties hierarchy:

```
rules
  rule_name
  predicate
    metric_name
    operator
    value
  action
    value
```

For each rule, you specify the following properties:

- **rule_name** – Rule names must be unique within WLM configuration. Rule names can be up to 32 alphanumeric characters or underscores, and can't contain spaces or quotation marks. You can have up to eight rules per queue, and the total limit for all queues is eight rules.
- **predicate** – You can have up to three predicates per rule. For each predicate, specify the following properties.
  - **metric_name** – For a list of metrics, see Query monitoring metrics in the Amazon Redshift Database Developer Guide.
  - **operator** – Operations are =, <, and >.
  - **value** – The threshold value for the specified metric that triggers an action.
- **action** – Each rule is associated with one action. Valid actions are:
  - log
  - hop (only available with manual WLM)
  - abort
• change_query_priority (only available with automatic WLM)

The following example shows the JSON for a WLM query monitoring rule named rule_1, with two predicates and the action hop.

```
"rules": [
  {
    "rule_name": "rule_1",
    "predicate": [
      {
        "metric_name": "query_execution_time",
        "operator": ">",
        "value": 100000
      },
      {
        "metric_name": "query_blocks_read",
        "operator": ">",
        "value": 1000
      }
    ],
    "action": "hop"
  }
]
```

For more information about each of these properties and strategies for configuring query queues, see Implementing workload management in the Amazon Redshift Database Developer Guide.

### Configuring the wlm_json_configuration parameter using the AWS CLI

To configure WLM, you modify the `wlm_json_configuration` parameter. The maximum size of the `wlm_json_configuration` property value is 8000 characters. The value is formatted in JavaScript Object Notation (JSON). If you configure WLM by using the AWS CLI, Amazon Redshift API, or one of the AWS SDKs, use the rest of this section to learn how to construct the JSON structure for the `wlm_json_configuration` parameter.

**Note**

If you configure WLM by using the Amazon Redshift console, you don't need to understand JSON formatting because the console provides an easy way to add queues and configure their properties. For more information about configuring WLM by using the console, see Modifying a parameter group (p. 284).

**Example**

The following example is the default WLM configuration, which defines one queue with automatic WLM.

```
{
  "auto_wlm": true
}
```

**Example**

The following example is a custom WLM configuration, which defines one manual WLM queue with a concurrency level (query slots) of five.

```
{
  "query_concurrency": 5
}
```
Syntax

The default WLM configuration is very simple, with only queue and one property. You can add more queues and configure multiple properties for each queue in the JSON structure. The following syntax represents the JSON structure that you use to configure multiple queues with multiple properties:

```
[  
  
  {  
    "ParameterName":"wlm_json_configuration", "ParameterValue":  
      "[
        
          
            "q1_first_property_name":"q1_first_property_value",  
            "q1_second_property_name":"q1_second_property_value",  
            ...
          
        
        
          
            "q2_first_property_name":"q2_first_property_value",  
            "q2_second_property_name":"q2_second_property_value",  
            ...
          
        
        
          
          
        ]"  
  
  }
]
```

In the preceding example, the representative properties that begin with q1 are objects in an array for the first queue. Each of these objects is a name/value pair; name and value together set the WLM properties for the first queue. The representative properties that begin with q2 are objects in an array for the second queue. If you require more queues, you add another array for each additional queue and set the properties for each object.

When you modify the WLM configuration, you must include in the entire structure for your queues, even if you only want to change one property within a queue. This is because the entire JSON structure is passed in as a string as the value for the wlm_json_configuration parameter.

Formatting the AWS CLI command

The wlm_json_configuration parameter requires a specific format when you use the AWS CLI. The format that you use depends on your client operating system. Operating systems have different ways to enclose the JSON structure so it's passed correctly from the command line. For details on how to construct the appropriate command in the Linux, Mac OS X, and Windows operating systems, see the sections following. For more information about the differences in enclosing JSON data structures in the AWS CLI in general, see Quoting strings in the AWS Command Line Interface User Guide.

Examples

The following example command configures manual WLM for a parameter group called example-parameter-group. The configuration enables short-query acceleration with a maximum run time for short queries set to 0, which instructs WLM to set the value dynamically. The ApplyType setting is dynamic. This setting means that any changes made to dynamic properties in the parameter are applied immediately unless other static changes have been made to the configuration. The configuration defines three queues with the following:

- The first queue enables users to specify report as a label (as specified in the query_group property) in their queries to help in routing queries to that queue. Wildcard searches are enabled for the report* label, so the label doesn't need to be exact for queries to be routed to the queue. For
example, reports and reporting both match this query group. The queue is allocated 25 percent of the total memory across all queues, and can run up to four queries at the same time. Queries are limited to a maximum time of 20000 milliseconds (ms). mode is set to auto, so when the queue’s query slots are full eligible queries are sent to a scaling cluster.

- The second queue enables users who are members of admin or dba groups in the database to have their queries routed to the queue for processing. Wildcard searches are disabled for user groups, so users must be matched exactly to groups in the database in order for their queries to be routed to the queue. The queue is allocated 40 percent of the total memory across all queues, and it can run up to five queries at the same time. mode is set to off, so all queries sent by members of the admin or dba groups run on the main cluster.

- The last queue in the configuration is the default queue. This queue is allocated 35 percent of the total memory across all queues, and it can process up to five queries at a time. mode is set to auto.

Note
The example is shown on several lines for demonstration purposes. Actual commands should not have line breaks.

```
aws redshift modify-cluster-parameter-group
--parameter-group-name example-parameter-group
--parameters
  [
    {
      "query_concurrency": 4,
      "max_execution_time": 20000,
      "memory_percent_to_use": 25,
      "query_group": ["report"],
      "query_group_wild_card": 1,
      "user_group": [],
      "user_group_wild_card": 0,
      "concurrency_scaling": "auto",
      "queue_type": "manual"
    },
    {
      "query_concurrency": 5,
      "memory_percent_to_use": 40,
      "query_group": [],
      "query_group_wild_card": 0,
      "user_group": [
        "admin",
        "dba"
      ],
      "user_group_wild_card": 0,
      "concurrency_scaling": "off",
      "queue_type": "manual"
    },
    {
      "query_concurrency": 5,
      "query_group": [],
      "query_group_wild_card": 0,
      "user_group": [],
      "user_group_wild_card": 0,
      "concurrency_scaling": "auto",
      "queue_type": "manual"
    },
    {"short_query_queue": true}
  ]
```

The following is an example of configuring WLM query monitoring rules for an automatic WLM configuration. The example creates a parameter group named example-monitoring-rules. The configuration defines the same three queues as the previous example, but the query_concurrency and
memory_percent_to_use are not specified anymore. The configuration also adds the following rules and query priorities:

- The first queue defines a rule named `rule_1`. The rule has two predicates: `query_cpu_time > 10000000` and `query_blocks_read > 1000`. The rule action is `log`. The priority of this queue is Normal.
- The second queue defines a rule named `rule_2`. The rule has two predicates: `query_execution_time > 600000000` and `scan_row_count > 1000000000`. The rule action is `abort`. The priority of this queue is Highest.
- The last queue in the configuration is the default queue. The priority of this queue is Low.

**Note**
The example is shown on several lines for demonstration purposes. Actual commands should not have line breaks.

```bash
aws redshift modify-cluster-parameter-group
--parameter-group-name example-monitoring-rules
--parameters
'[
  {
    "query_group" : [ "report" ],
    "query_group_wild_card" : 1,
    "user_group" : [ ],
    "user_group_wild_card" : 0,
    "concurrency_scaling" : "auto",
    "rules" : [{
      "rule_name": "rule_1",
      "predicate": [{
        "metric_name": "query_cpu_time",
        "operator": ">",
        "value": 1000000 },
      { "metric_name": "query_blocks_read",
        "operator": ">",
        "value": 1000 }],
      "action": "log"
    }],
    "priority": "normal",
    "queue_type": "auto"
  },
  {
    "query_group" : [ ],
    "query_group_wild_card" : 0,
    "user_group" : [ "admin", "dba" ],
    "user_group_wild_card" : 0,
    "concurrency_scaling" : "off",
    "rules" : [{
      "rule_name": "rule_2",
      "predicate": [{
        "metric_name": "query_execution_time",
        "operator": ">",
        "value": 600000000},
      { "metric_name": "scan_row_count",
        "operator": ">",
        "value": 1000000000}],
      "action": "abort"
    }],
    "priority": "high",
    "queue_type": "auto"
  },
  {
    "query_group" : [ ],
    "query_group_wild_card" : 0,
    "user_group" : [ ],
    "user_group_wild_card" : 0,
    "concurrency_scaling" : "auto",
    "rules" : [{
      "rule_name": "default",
      "predicate": [{
        "metric_name": "query_group",
        "operator": "in",
        "value": [ "report" ]},
      { "metric_name": "user_group",
        "operator": "in",
        "value": [ "admin", "dba" ]}],
      "action": "schedule"}
  ]
]
' --cluster-name example-cluster
'
Configuring WLM by using the AWS CLI in the command line with a JSON file

You can modify the `wlm_json_configuration` parameter using the AWS CLI and pass in the value of the parameters argument as a JSON file.

```bash
aws redshift modify-cluster-parameter-group --parameter-group-name myclusterparametergroup --parameters file://modify_pg.json
```

The arguments for `--parameters` are stored in file `modify_pg.json`. The file location is specified in the format for your operating system. For more information, see Loading parameters from a file. The following shows examples of the content of the `modify_pg.json` JSON file.

```
[
  {
    "ParameterName": "wlm_json_configuration",
    "ParameterValue": "["user_group": "example_user_group1", "query_group": "example_query_group1", "query_concurrency": 7], "query_concurrency": 5"
  }
]
```

```
[
  {
    "ParameterName": "wlm_json_configuration",
    "ParameterValue": "["query_group": ["reports"], "query_group_wild_card": 0, "query_concurrency": 4, "max_execution_time": 20000, "memory_percent_to_use": 25], "user_group": ["admin", "dba"], "user_group_wild_card": 1, "query_concurrency": 5, "memory_percent_to_use": 35, "short_query_queue": true, "max_execution_time": 5000 },
    "ApplyType": "dynamic"
  }
]
```

Rules for configuring WLM by using the AWS CLI in the command line on the Linux and macOS X operating systems

Follow these rules to run an AWS CLI command with parameters on one line:

- The entire JSON structure must be enclosed in single quotation marks (') and brackets ({}).
- All parameter names and parameter values must be enclosed in double quotation marks ("").
- Within the ParameterValue value, you must enclose the entire nested structure in double-quotiation marks ("" ) and brackets ({}).
- Within the nested structure, each of the properties and values for each queue must be enclosed in curly braces ({}).
- Within the nested structure, you must use the backslash (\) escape character before each double-quotiation mark (""").
- For name/value pairs, a colon (:) separates each property from its value.
- Each name/value pair is separated from another by a comma (,).
Multiple queues are separated by a comma (,) between the end of one queue's curly brace () and the beginning of the next queue's curly brace ().

Rules for configuring WLM by using the AWS CLI in Windows PowerShell on Microsoft Windows operating systems

Follow these rules to run an AWS CLI command with parameters on one line:

- The entire JSON structure must be enclosed in single quotation marks (') and brackets ([ ]).
- All parameter names and parameter values must be enclosed in double quotation marks (").
- Within the ParameterValue value, you must enclose the entire nested structure in double-quotation marks ("), and brackets ([ ]).
- Within the nested structure, each of the properties and values for each queue must be enclosed in curly braces ({ }).
- Within the nested structure, you must use the backslash (\) escape character before each double-quotation mark ("), and its backslash (\) escape character. This requirement means that you will use three backslashes and a double quotation mark to make sure that the properties are passed in correctly (\\").
- For name/value pairs, a colon (:) separates each property from its value.
- Each name/value pair is separated from another by a comma (,).
- Multiple queues are separated by a comma (,) between the end of one queue's curly brace () and the beginning of the next queue's curly brace ().

Rules for configuring WLM by using the command prompt on Windows operating systems

Follow these rules to run an AWS CLI command with parameters on one line:

- The entire JSON structure must be enclosed in double-quotation marks ("), and brackets ([ ]).
- All parameter names and parameter values must be enclosed in double quotation marks (").
- Within the ParameterValue value, you must enclose the entire nested structure in double-quotation marks ("), and brackets ([ ]).
- Within the nested structure, each of the properties and values for each queue must be enclosed in curly braces ({ }).
- Within the nested structure, you must use the backslash (\) escape character before each double-quotation mark ("), and its backslash (\) escape character. This requirement means that you will use three backslashes and a double quotation mark to make sure that the properties are passed in correctly (\\").
- For name/value pairs, a colon (:) separates each property from its value.
- Each name/value pair is separated from another by a comma (,).
- Multiple queues are separated by a comma (,) between the end of one queue's curly brace () and the beginning of the next queue's curly brace ().

Managing parameter groups using the console

You can view, create, modify, and delete parameter groups on the Amazon Redshift console.

You can view any of your parameter groups to see a summary of the values for parameters and workload management (WLM) configuration. Group parameters appear on the Parameters tab, and Workload queues appear on the Workload Management tab.
Creating a parameter group

If you want to set parameter values that are different from the default parameter group, you can create your own parameter group,

To create a parameter group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Workload management to display the Workload management page.
3. Choose Create to display the Create parameter group window.
4. Enter a value for Parameter group name and Description.
5. Choose Create to create the parameter group.

Modifying a parameter group

You can modify parameters to change the parameter settings and WLM configuration properties.

Note
You can't modify the default parameter group.

To modify a parameter group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Workload management to display the Workload management page.
3. Choose the parameter group that you want to modify to display the details page, with tabs for Parameters and Workload management.
4. Choose the Parameters tab to view the current parameter settings.
5. Choose Edit parameters to enable changing settings for these parameters:

   • auto_analyze
   • datestyle
   • enable_case_sensitive_identifier
   • enable_user_activity_logging
   • extra_float_digits
   • max_concurrency_scaling_clusters
   • max_cursor_result_set_size
   • query_group
   • require_ssl
   • search_path
   • statement_timeout
   • use_fips_ssl

   For more information about these parameters, see Amazon Redshift parameter groups (p. 271).
6. Enter your changes and then choose Save to update the parameter group.
To modify the WLM configuration for a parameter group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Workload management to display the Workload management page.
3. Choose the parameter group that you want to modify to display the details page with tabs for Parameters and Workload management.
4. Choose the Workload management tab to view the current WLM configuration.
5. Choose Edit workload queues to edit the WLM configuration,
6. (Optional) Select Enable short query acceleration to enable short query acceleration (SQA).

   When you enable SQA, Maximum run time for short queries (1 to 20 seconds) is set to Dynamic by default. To set the maximum runtime to a fixed value, choose a value of 1–20.
7. Do one or more of the following to modify the queue configuration:
   - Choose Switch WLM mode to choose between Automatic WLM and Manual WLM.
     With Automatic WLM, the Memory and Concurrency on main values are set to auto.
   - To create a queue, choose Edit workload queues, then choose Add Queue.
   - To modify a queue, change property values in the table. Depending on the type of queue, properties can include the following:
     - Queue name can be changed.
     - Memory (%)
     - Concurrency on main cluster
     - Concurrency scaling mode can be off or auto
     - Timeout (ms)
     - User groups
     - Query groups

   For more information about these properties, see Properties for the wlm_json_configuration parameter (p. 274).

   Important
   If you change a queue name, the QueueName dimension value of WLM queue metrics (such as, WLMQueueLength, WLMQueueWaitTime, WLMQueriesCompletedPerSecond, WLMQueryDuration, WLMRunningQueries, and so on) also changes. So, if you change the name of a queue, you might need to change CloudWatch alarms you have set up.
   - To change the order of queues, choose the Up and Down arrow buttons.
   - To delete a queue, choose Delete in the queue's row in the table.
8. (Optional) Select Defer dynamic changes until reboot to have the changes applied to clusters after their next reboot.

   Note
   Some changes require a cluster reboot regardless of this setting. For more information, see WLM dynamic and static properties (p. 274).
9. Choose Save.
Creating or modifying a query monitoring rule using the console

You can use the Amazon Redshift console to create and modify WLM query monitoring rules. Query monitoring rules are part of the WLM configuration parameter for a parameter group. If you modify a query monitoring rule (QMR), the change happens automatically without the need to modify the cluster. For more information, see WLM query monitoring rules.

When you create a rule, you define the rule name, one or more predicates, and an action.

When you save WLM configuration that includes a rule, you can view the JSON code for the rule definition as part of the JSON for the WLM configuration parameter.

To create a query monitoring rule

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Workload management to display the Workload management page.
3. Choose the parameter group that you want to modify to display the details page with tabs for Parameters and Workload management.
4. Choose the Workload management tab, and choose Edit workload queues to edit the WLM configuration,
5. Add a new rule either by using a predefined template or from scratch.

To use a predefined template, do the following:

1. Choose Add rule from template in the Query monitoring rules group. The list of rule templates is displayed.
2. Choose one or more rule templates. When you choose Save, WLM creates one rule for each template that you choose.
3. Enter or confirm values for the rule, including Rule names, Predicates and Actions.
4. Choose Save.

To add a new rule from scratch, do the following:

1. To add additional predicates, choose Add predicate. You can have up to three predicates for each rule. If all of the predicates are met, WLM triggers the associated action.
2. Choose an Action. Each rule has one action.
3. Choose Save.

Amazon Redshift generates your WLM configuration parameter in JSON format and displays it in the JSON section.

Deleting a parameter group

You can delete a parameter group if you no longer need it and it is not associated with any clusters. You can only delete custom parameter groups.

To delete a parameter group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Configurations**, then choose **Workload management** to display the **Workload management** page.

3. For **Parameter groups**, choose the parameter group that you want to modify.

   **Note**
   You can't delete the default parameter group.

4. Choose **Delete** and confirm that you want to delete the parameter group.

## Associating a parameter group with a cluster

When you launch a cluster, you must associate it with a parameter group. If you want to change the parameter group later, you can modify the cluster and choose a different parameter group.

## Managing parameter groups using the AWS SDK for Java

This example demonstrates the following tasks related to parameter groups:

- Creating a parameter group
- Modifying a parameter group
- Associating a parameter group with a cluster
- Getting information about parameter groups

This example creates a new parameter group, `parametergroup1`, and makes the following updates:

- Changes the parameter `extra_float_digits` to 2 from the default value of 0.
- Replaces the existing workload management configuration (`wlm_json_configuration` parameter) with the following JSON which defines a queue in addition to the default queue.

```json
[
  {
    "user_group": ["example_user_group1"],
    "query_group": ["example_query_group1"],
    "query_concurrency": 7
  },
  {
    "query_concurrency": 5
  }
]
```

The preceding JSON is an array of two objects, one for each queue. The first object defines a queue with specific user group and query group. It also sets the concurrency level to 7.

```json
{
  "user_group": ["example_user_group1"],
}
```
Because this example replaces the WLM configuration, this JSON configuration also defines the default queue with no specific user group or query group. It sets the concurrency to the default value, 5.

```json
{
  "query_concurrency":5
}
```

For more information about Workload Management (WML) configuration, go to Implementing workload management.

For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). You need to update the code and provide a cluster identifier.

**Example**

```java
package com.amazonaws.services.redshift;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import com.amazonaws.services.redshift.model.*;

public class CreateAndModifyClusterParameterGroup {
    public static AmazonRedshift client;
    public static String clusterParameterGroupName = "parametergroup1";
    public static String clusterIdentifier = "***provide a cluster identifier***";
```
public static String parameterGroupFamily = "redshift-1.0";

public static void main(String[] args) throws IOException {
    // Default client using the @link
    com.amazonaws.auth.DefaultAWSCredentialsProviderChain
    client = AmazonRedshiftClientBuilder.defaultClient();
    try {
        createClusterParameterGroup();
        modifyClusterParameterGroup();
        associateParameterGroupWithCluster();
        describeClusterParameterGroups();
    } catch (Exception e) {
        System.err.println("Operation failed: " + e.getMessage());
    }
}

private static void createClusterParameterGroup() {
    CreateClusterParameterGroupRequest request = new
    CreateClusterParameterGroupRequest()
        .withDescription("my cluster parameter group")
        .withParameterGroupName(clusterParameterGroupName)
        .withParameterGroupFamily(parameterGroupFamily);
    client.createClusterParameterGroup(request);
    System.out.println("Created cluster parameter group.");
}

private static void describeClusterParameterGroups() {
    DescribeClusterParameterGroupsResult result =
    client.describeClusterParameterGroups();
    printResultClusterParameterGroups(result);
}

private static void modifyClusterParameterGroup() {
    List<Parameter> parameters = new ArrayList<Parameter>();
    parameters.add(new Parameter()
        .withParameterName("extra_float_digits")
        .withParameterValue("2"));
    // Replace WLM configuration. The new configuration defines a queue (in addition to
    the default).
    parameters.add(new Parameter()
        .withParameterName("wlm_json_configuration")
        .withParameterValue("[["user_group":"example_user_group1"],"query_group":
          ["example_query_group1"],"query_concurrency":7,"query_concurrency":5]]");
    ModifyClusterParameterGroupRequest request = new
    ModifyClusterParameterGroupRequest()
        .withParameterGroupName(clusterParameterGroupName)
        .withParameters(parameters);
    client.modifyClusterParameterGroup(request);
}

private static void associateParameterGroupWithCluster() {
    ModifyClusterRequest request = new ModifyClusterRequest()
        .withClusterIdentifier(clusterIdentifier)
        .withClusterParameterGroupName(clusterParameterGroupName);
    Cluster result = client.modifyCluster(request);
    System.out.format("Parameter Group %s is used for Cluster %s\n",
        clusterParameterGroupName, result.getClusterParameterGroups().get(0).getParameterGroupName());
}
Managing parameter groups using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift operations in the AWS CLI to manage parameter groups.

- `create-cluster-parameter-group`
- `delete-cluster-parameter-group`
- `describe-cluster-parameters`

```java
private static void printResultClusterParameterGroups(DescribeClusterParameterGroupsResult result) {
    if (result == null) {
        System.out.println("Describe cluster parameter groups result is null.");
        return;
    }

    System.out.println("Printing parameter group results:
    for (ClusterParameterGroup group : result.getParameterGroups()) {
        System.out.format("Description: %s\n", group.getDescription());
        System.out.format("Group Family Name: %s\n", group.getParameterGroupFamily());
        System.out.format("Group Name: %s\n", group.getParameterGroupName());
        describeClusterParameters(group.getParameterGroupName());
    }
}

private static void describeClusterParameters(String parameterGroupName) {
    DescribeClusterParametersRequest request = new DescribeClusterParametersRequest()
            .withParameterGroupName(parameterGroupName);
    DescribeClusterParametersResult result = client.describeClusterParameters(request);
    printResultClusterParameters(result, parameterGroupName);
}

private static void printResultClusterParameters(DescribeClusterParametersResult result, String parameterGroupName) {
    if (result == null) {
        System.out.println("Cluster parameters is null.");
        return;
    }

    System.out.format("Printing cluster parameters for "%s"\n", parameterGroupName);
    for (Parameter parameter : result.getParameters()) {
        System.out.println("Name: " + parameter.getParameterName() + ", Value: " +
                parameter.getParameterValue());
        System.out.println("DataType: " + parameter.getDataType() + ",
                MinEngineVersion: " + parameter.getMinimumEngineVersion());
        System.out.println("AllowedValues: " + parameter.getAllowedValues() + ",
                Source: " + parameter.getSource());
        System.out.println("IsModifiable: " + parameter.getIsModifiable() + ",
                Description: " + parameter.getDescription());
    }
}
// snippet-end:[redshift.java.CreateAndModifyClusterParameterGroup.complete]
```
Managing parameter groups using the AWS CLI and Amazon Redshift API

- describe-cluster-parameter-groups
- describe-default-cluster-parameters
- modify-cluster-parameter-group
- reset-cluster-parameter-group

You can use the following Amazon Redshift API operations to manage parameter groups.

- CreateClusterParameterGroup
- DeleteClusterParameterGroup
- DescribeClusterParameters
- DescribeClusterParameterGroups
- DescribeDefaultClusterParameters
- ModifyClusterParameterGroup
- ResetClusterParameterGroup
Amazon Redshift snapshots

Overview

Snapshots are point-in-time backups of a cluster. There are two types of snapshots: *automated* and *manual*. Amazon Redshift stores these snapshots internally in Amazon S3 by using an encrypted Secure Sockets Layer (SSL) connection.

Amazon Redshift automatically takes incremental snapshots that track changes to the cluster since the previous automated snapshot. Automated snapshots retain all of the data required to restore a cluster from a snapshot. You can create a snapshot schedule to control when automated snapshots are taken, or you can take a manual snapshot any time.

When you restore from a snapshot, Amazon Redshift creates a new cluster and makes the new cluster available before all of the data is loaded, so you can begin querying the new cluster immediately. The cluster streams data on demand from the snapshot in response to active queries, then loads the remaining data in the background.

When you launch a cluster, you can set the retention period for automated and manual snapshots. You can change the default retention period for automated and manual snapshots by modifying the cluster. You can change the retention period for a manual snapshot when you create the snapshot or by modifying the snapshot.

You can monitor the progress of snapshots by viewing the snapshot details in the AWS Management Console, or by calling `describe-cluster-snapshots` in the CLI or the `DescribeClusterSnapshots` API action. For an in-progress snapshot, these display information such as the size of the incremental snapshot, the transfer rate, the elapsed time, and the estimated time remaining.

To ensure that your backups are always available to your cluster, Amazon Redshift stores snapshots in an internally managed Amazon S3 bucket that is managed by Amazon Redshift. To manage storage charges, evaluate how many days you need to keep automated snapshots and configure their retention period.
Automated snapshots

When automated snapshots are enabled for a cluster, Amazon Redshift periodically takes snapshots of that cluster. By default, Amazon Redshift takes a snapshot about every eight hours or following every 5 GB per node of data changes, or whichever comes first. Alternatively, you can create a snapshot schedule to control when automated snapshots are taken. Automated snapshots are enabled by default when you create a cluster.

Automated snapshots are deleted at the end of a retention period. The default retention period is one day, but you can modify it by using the Amazon Redshift console or programmatically by using the Amazon Redshift API or CLI.

To disable automated snapshots, set the retention period to zero. If you disable automated snapshots, Amazon Redshift stops taking snapshots and deletes any existing automated snapshots for the cluster. You can't disable automated snapshots for RA3 node types. You can set an RA3 node type automated retention period from 1–35 days.

Only Amazon Redshift can delete an automated snapshot; you cannot delete them manually. Amazon Redshift deletes automated snapshots at the end of a snapshot’s retention period, when you disable automated snapshots for the cluster, or when you delete the cluster. Amazon Redshift retains the latest automated snapshot until you disable automated snapshots or delete the cluster.

If you want to keep an automated snapshot for a longer period, you can create a copy of it as a manual snapshot. The automated snapshot is retained until the end of the retention period, but the corresponding manual snapshot is retained until you manually delete it or until the end of the retention period.

Automated snapshot schedules

To precisely control when snapshots are taken, you can create a snapshot schedule and attach it to one or more clusters. When you modify a snapshot schedule, the schedule is modified for all associated clusters. If a cluster doesn’t have a snapshot schedule attached, the cluster uses the default automated snapshot schedule.

A snapshot schedule is a set of schedule rules. You can define a simple schedule rule based on a specified interval, such as every 8 hours or every 12 hours. You can also add rules to take snapshots on certain days of the week, at specific times, or during specific periods. Rules can also be defined using Unix-like cron expressions.

Snapshot schedule format

On the Amazon Redshift console, you can create a snapshot schedule. Then, you can attach a schedule to a cluster to trigger the creation of a system snapshot. A schedule can be attached to multiple clusters, and you can create multiple cron definitions in a schedule to trigger a snapshot.

You can define a schedule for your snapshots using a cron syntax. The definition of these schedules uses a modified Unix-like cron syntax. You specify time in Coordinated universal time (UTC). You can create schedules with a maximum frequency of one hour and minimum precision of one minute.

Amazon Redshift modified cron expressions have 3 required fields, which are separated by white space.
Amazon Redshift Cluster Management Guide
Snapshot schedule format

Syntax

cron(Minutes Hours Day-of-week)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Values</th>
<th>Wildcards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minutes</td>
<td>0–59</td>
<td></td>
</tr>
<tr>
<td>Hours</td>
<td>0–23</td>
<td>, - */</td>
</tr>
<tr>
<td>Day-of-week</td>
<td>1–7 or SUN-SAT</td>
<td>, - */</td>
</tr>
</tbody>
</table>

Wildcards

- The , (comma) wildcard includes additional values. In the Day-of-week field, MON, WED, FRI would include Monday, Wednesday, and Friday. Total values are limited to 24 per field.
- The - (dash) wildcard specifies ranges. In the Hour field, 1–15 would include hours 1 through 15 of the specified day.
- The * (asterisk) wildcard includes all values in the field. In the Hours field, * would include every hour.
- The / (forward slash) wildcard specifies increments. In the Hours field, you could enter 1/10 to specify every 10th hour, starting from the first hour of the day (for example, the 01:00, 11:00, and 21:00).

Limits

- Snapshot schedules that lead to backup frequencies less than 1 hour or greater than 24 hours are not supported. If you have overlapping schedules that result in scheduling snapshots within a 1 hour window, a validation error results.

When creating a schedule, you can use the following sample cron strings.

<table>
<thead>
<tr>
<th>Minutes</th>
<th>Hours</th>
<th>Day of week</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>14-20/1</td>
<td>TUE</td>
<td>Every hour between 2pm and 8pm on Tuesday.</td>
</tr>
<tr>
<td>0</td>
<td>21</td>
<td>MON-FRI</td>
<td>Every night at 9pm Monday–Friday.</td>
</tr>
<tr>
<td>30</td>
<td>0/6</td>
<td>SAT-SUN</td>
<td>Every 6 hour increment on Saturday and Sunday starting at 30 minutes after midnight (00:30) that day. This results in a snapshot at [00:30, 06:30, 12:30, and 18:30] each day.</td>
</tr>
<tr>
<td>30</td>
<td>12/4</td>
<td>*</td>
<td>Every 4 hour increment starting at 12:30 each day. This resolves to [12:30, 16:30, 20:30].</td>
</tr>
</tbody>
</table>
Manual snapshots

You can take a manual snapshot any time. By default, manual snapshots are retained indefinitely, even after you delete your cluster. You can specify the retention period when you create a manual snapshot, or you can change the retention period by modifying the snapshot. For more information about changing the retention period, see Changing the manual snapshot retention period (p. 302).

If a snapshot is deleted, you can't start any new operations that reference that snapshot. However, if a restore operation is in progress, that restore operation will run to completion.

Amazon Redshift has a quota that limits the total number of manual snapshots that you can create; this quota is per AWS account per AWS Region. The default quota is listed at Quotas and limits in Amazon Redshift (p. 505).

Managing snapshot storage

Because snapshots accrue storage charges, it's important that you delete them when you no longer need them. Amazon Redshift deletes automated and manual snapshots at the end of their respective snapshot retention periods. You can also delete manual snapshots using the AWS Management Console or with the batch-delete-cluster-snapshots CLI command.

You can change the retention period for a manual snapshot by modifying the manual snapshot settings.

You can get information about how much storage your snapshots are consuming using the Amazon Redshift Console or using the describe-storage CLI command.

Excluding tables from snapshots

By default, all user-defined permanent tables are included in snapshots. If a table, such as a staging table, doesn't need to be backed up, you can significantly reduce the time needed to create snapshots and restore from snapshots. You also reduce storage space on Amazon S3 by using a no-backup table. To create a no-backup table, include the BACKUP NO parameter when you create the table. For more information, see CREATE TABLE and CREATE TABLE AS in the Amazon Redshift Database Developer Guide.

Copying snapshots to another AWS Region

You can configure Amazon Redshift to automatically copy snapshots (automated or manual) for a cluster to another AWS Region. When a snapshot is created in the cluster's primary AWS Region, it's copied to
a secondary AWS Region. The two AWS Regions are known respectively as the source AWS Region and destination AWS Region. If you store a copy of your snapshots in another AWS Region, you can restore your cluster from recent data if anything affects the primary AWS Region. You can configure your cluster to copy snapshots to only one destination AWS Region at a time. For a list of Amazon Redshift Regions, see Regions and endpoints in the Amazon Web Services General Reference.

When you enable Amazon Redshift to automatically copy snapshots to another AWS Region, you specify the destination AWS Region to copy the snapshots to. For automated snapshots, you can also specify the retention period to keep them in the destination AWS Region. After an automated snapshot is copied to the destination AWS Region and it reaches the retention time period there, it's deleted from the destination AWS Region. Doing this keeps your snapshot usage low. To keep the automated snapshots for a shorter or longer time in the destination AWS Region, change this retention period.

The retention period that you set for automated snapshots that are copied to the destination AWS Region is separate from the retention period for automated snapshots in the source AWS Region. The default retention period for copied snapshots is seven days. That seven-day period applies only to automated snapshots. In both the source and destination AWS Regions, manual snapshots are deleted at the end of the snapshot retention period or when you manually delete them.

You can disable automatic snapshot copy for a cluster at any time. When you disable this feature, snapshots are no longer copied from the source AWS Region to the destination AWS Region. Any automated snapshots copied to the destination AWS Region are deleted as they reach the retention period limit, unless you create manual snapshot copies of them. These manual snapshots, and any manual snapshots that were copied from the destination AWS Region, are kept in the destination AWS Region until you manually delete them.

To change the destination AWS Region that you copy snapshots to, first disable the automatic copy feature. Then re-enable it, specifying the new destination AWS Region.

After a snapshot is copied to the destination AWS Region, it becomes active and available for restoration purposes.

To copy snapshots for AWS KMS–encrypted clusters to another AWS Region, create a grant for Amazon Redshift to use a customer managed key in the destination AWS Region. Then choose that grant when you enable copying of snapshots in the source AWS Region. For more information about configuring snapshot copy grants, see Copying AWS KMS–encrypted snapshots to another AWS Region (p. 325).

### Restoring a cluster from a snapshot

A snapshot contains data from any databases that are running on your cluster. It also contains information about your cluster, including the number of nodes, node type, and admin user name. If you restore your cluster from a snapshot, Amazon Redshift uses the cluster information to create a new cluster. Then it restores all the databases from the snapshot data.

For the new cluster created from the original snapshot, you can choose the configuration, such as node type and number of nodes. The cluster is restored in the same AWS Region and a random, system-chosen Availability Zone, unless you specify another Availability Zone in your request. When you restore a cluster from a snapshot, you can optionally choose a compatible maintenance track for the new cluster.

**Note**

When you restore a snapshot to a cluster with a different configuration, the snapshot must have been taken on a cluster with cluster version 1.0.10013, or later.

When a restore is in progress, events are typically emitted in the following order:

1. RESTORE_STARTED – REDSHIFT-EVENT-2008 sent when the restore process begins.
2. RESTORE_SUCCEEDED – REDSHIFT-EVENT-3003 sent when the new cluster has been created.
The cluster is available for queries.

3. DATA_TRANSFER_COMPLETED – REDSHIFT-EVENT-3537 sent when data transfer complete.

**Note**
RA3 clusters only emit RESTORE_STARTED and RESTORE_SUCCEEDED events. There is no explicit data transfer to be done after a RESTORE succeeds because RA3 node types store data in Amazon Redshift managed storage. With RA3 nodes, data is continuously transferred between RA3 nodes and Amazon Redshift managed storage as part of normal query processing. RA3 nodes cache hot data locally and keep less frequently queried blocks in Amazon Redshift managed storage automatically.

You can monitor the progress of a restore by either calling the DescribeClusters API operation, or viewing the cluster details in the AWS Management Console. For an in-progress restore, these display information such as the size of the snapshot data, the transfer rate, the elapsed time, and the estimated time remaining. For a description of these metrics, see RestoreStatus.

You can't use a snapshot to revert an active cluster to a previous state.

**Note**
When you restore a snapshot into a new cluster, the default security group and parameter group are used unless you specify different values.

You might want to restore a snapshot to a cluster with a different configuration for these reasons:

- When a cluster is made up of smaller node types and you want to consolidate it into a larger node type with fewer nodes.
- When you have monitored your workload and determined the need to move to a node type with more CPU and storage.
- When you want to measure performance of test workloads with different node types.

Restore has the following constraints:

- The new node configuration must have enough storage for existing data. Even when you add nodes, your new configuration might not have enough storage because of the way that data is redistributed.
- The restore operation checks if the snapshot was created on a cluster version that is compatible with the cluster version of the new cluster. If the new cluster has a version level that is too early, then the restore operation fails and reports more information in an error message.
- The possible configurations (number of nodes and node type) you can restore to is determined by the number of nodes in the original cluster and the target node type of the new cluster. To determine the possible configurations available, you can use the Amazon Redshift console or the describe-node-configuration-options AWS CLI command with action-type restore-cluster. For more information about the restoring using the Amazon Redshift console, see Restoring a cluster from a snapshot (p. 303).

The following steps take a cluster with many nodes and consolidate it into a bigger node type with a smaller number of nodes using the AWS CLI. For this example, we start with a source cluster of 24 ds2.xlarge nodes. In this case, suppose that we already created a snapshot of this cluster and want to restore it into a bigger node type.

1. Run the following command to get the details of our 24-node ds2.xlarge cluster.

   ```bash
   aws redshift describe-clusters --region eu-west-1 --cluster-identifier mycluster-123456789012
   ```

2. Run the following command to get the details of the snapshot.
3. Run the following command to describe the options available for this snapshot.

```
aws redshift describe-node-configuration-options --snapshot-identifier mycluster-snapshot
--region eu-west-1 --action-type restore-cluster
```

This command returns an option list with recommended node types, number of nodes, and disk utilization for each option. For this example, the preceding command lists the following possible node configurations. We choose to restore into a three-node ds2.8xlarge cluster.

```
{
   "NodeConfigurationOptionList": [
      {
         "EstimatedDiskUtilizationPercent": 65.26134808858235,
         "NodeType": "ds2.xlarge",
         "NumberOfNodes": 24
      },
      {
         "EstimatedDiskUtilizationPercent": 32.630674044291176,
         "NodeType": "ds2.xlarge",
         "NumberOfNodes": 48
      },
      {
         "EstimatedDiskUtilizationPercent": 65.26134808858235,
         "NodeType": "ds2.8xlarge",
         "NumberOfNodes": 3
      },
      {
         "EstimatedDiskUtilizationPercent": 48.94601106643677,
         "NodeType": "ds2.8xlarge",
         "NumberOfNodes": 4
      },
      {
         "EstimatedDiskUtilizationPercent": 39.156808853149414,
         "NodeType": "ds2.8xlarge",
         "NumberOfNodes": 5
      },
      {
         "EstimatedDiskUtilizationPercent": 32.630674044291176,
         "NodeType": "ds2.8xlarge",
         "NumberOfNodes": 6
      }
   ]
}
```

4. Run the following command to restore the snapshot into the cluster configuration that we chose.

After this cluster is restored, we have the same content as the source cluster, but the data has been consolidated into three ds2.8xlarge nodes.

```
aws redshift restore-from-cluster-snapshot --region eu-west-1 --snapshot-identifier mycluster-snapshot
--cluster-identifier mycluster-123456789012-x --node-type ds2.8xlarge
--number-of-nodes 3
```

If you have reserved nodes, for example DS2 or DC2 reserved nodes, you can upgrade to RA3 reserved nodes. You can do this when you restore from a snapshot or perform an elastic resize. You can use the console to guide you through this process. For more information about upgrading to RA3 nodes, see Upgrading to RA3 node types.
Restoring a table from a snapshot

You can restore a single table from a snapshot instead of restoring an entire cluster. When you restore a single table from a snapshot, you specify the source snapshot, database, schema, and table name, and the target cluster, schema, and a new table name for the restored table.

The new table name cannot be the name of an existing table. To replace an existing table with a restored table from a snapshot, rename or drop the existing table before you restore the table from the snapshot.

The target table is created using the source table's column definitions, table attributes, and column attributes except for foreign keys. To prevent conflicts due to dependencies, the target table doesn't inherit foreign keys from the source table. Any dependencies, such as views or permissions granted on the source table, are not applied to the target table.

If the owner of the source table exists, then that user is the owner of the restored table, provided that the user has sufficient permissions to become the owner of a relation in the specified database and schema. Otherwise, the restored table is owned by the admin user that was created when the cluster was launched.

The restored table returns to the state it was in at the time the backup was taken. This includes transaction visibility rules defined by the Amazon Redshift adherence to serializable isolation, meaning that data will be immediately visible to in flight transactions started after the backup.

Restoring a table from a snapshot has the following limitations:

- You can restore a table only to the current, active running cluster and from a snapshot that was taken of that cluster.
- You can restore only one table at a time.
- You can’t restore a table from a cluster snapshot that was taken prior to a cluster being resized. An exception is that you can restore a table after an elastic resize if the node type didn’t change.

To restore a table from a snapshot

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to use to restore a table.
3. For Actions, choose Restore table to display the Restore table page.
4. Enter the information about which snapshot, source table, and target table to use, and then choose Restore table.

Example Example: Restoring a table from a snapshot using the AWS CLI

The following example uses the `restore-table-from-cluster-snapshot` AWS CLI command to restore the `my-source-table` table from the `sample-database` schema in the `my-snapshot-id` snapshot. The example restores the snapshot to the `mycluster-example` cluster with a new table name of `my-new-table`.

```
aws redshift restore-table-from-cluster-snapshot --cluster-identifier mycluster-example
--new-table-name my-new-table
--snapshot-identifier my-snapshot-id
--source-database-name sample-database
--source-table-name my-source-table
```
Sharing snapshots

You can share an existing manual snapshot with other AWS customer accounts by authorizing access to the snapshot. You can authorize up to 20 for each snapshot and 100 for each AWS Key Management Service (AWS KMS) key. That is, if you have 10 snapshots that are encrypted with a single KMS key, then you can authorize 10 AWS accounts to restore each snapshot, or other combinations that add up to 100 accounts and do not exceed 20 accounts for each snapshot. A person logged in as a user in one of the authorized accounts can then describe the snapshot or restore it to create a new Amazon Redshift cluster under their account. For example, if you use separate AWS customer accounts for production and test, a user can log on using the production account and share a snapshot with users in the test account. Someone logged on as a test account user can then restore the snapshot to create a new cluster that is owned by the test account for testing or diagnostic work.

A manual snapshot is permanently owned by the AWS customer account under which it was created. Only users in the account owning the snapshot can authorize other accounts to access the snapshot, or to revoke authorizations. Users in the authorized accounts can only describe or restore any snapshot that has been shared with them; they cannot copy or delete snapshots that have been shared with them. An authorization remains in effect until the snapshot owner revokes it. If an authorization is revoked, the previously authorized user loses visibility of the snapshot and cannot launch any new actions referencing the snapshot. If the account is in the process of restoring the snapshot when access is revoked, the restore runs to completion. You cannot delete a snapshot while it has active authorizations; you must first revoke all of the authorizations.

AWS customer accounts are always authorized to access snapshots owned by the account. Attempts to authorize or revoke access to the owner account will receive an error. You cannot restore or describe a snapshot that is owned by an inactive AWS customer account.

After you have authorized access to an AWS customer account, no IAM users in that account can perform any actions on the snapshot unless they have IAM policies that allow them to do so.

- IAM users in the snapshot owner account can authorize and revoke access to a snapshot only if they have an IAM policy that allows them to perform those actions with a resource specification that includes the snapshot. For example, the following policy allows a user in AWS account 012345678912 to authorize other accounts to access a snapshot named my-snapshot20130829:

```json
{
    "Version": "2012-10-17",
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "redshift:AuthorizeSnapshotAccess",  
                "redshift:RevokeSnapshotAccess"
            ],  
            "Resource": [  
                "arn:aws:redshift:us-east-1:012345678912:snapshot:*/my-snapshot20130829"
            ]
        }
    ]
}
```

- IAM users in an AWS account with which a snapshot has been shared cannot perform actions on that snapshot unless they have IAM policies allowing those actions:

  - To list or describe a snapshot, they must have an IAM policy that allows the DescribeClusterSnapshots action. The following code shows an example:

```json
{
    "Version": "2012-10-17",
...
```
To restore a snapshot, users must have an IAM policy that allows the `redshift:RestoreFromClusterSnapshot` action and has a resource element that covers both the cluster they are attempting to create and the snapshot. For example, if a user in account 012345678912 has shared snapshot `my-snapshot20130829` with account 219876543210, in order to create a cluster by restoring the snapshot, a user in account 219876543210 must have a policy such as the following:

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": ["redshift:RestoreFromClusterSnapshot"],
      }
   ]
}
```

• After access to a snapshot has been revoked from an AWS account, no users in that account can access the snapshot. This is so even if they have IAM policies that allow actions on the previously shared snapshot resource.

Managing snapshots using the console

Amazon Redshift takes automatic, incremental snapshots of your data periodically and saves them to Amazon S3. Additionally, you can take manual snapshots of your data whenever you want. In this section, you can find how to manage your snapshots from the Amazon Redshift console. For more information about snapshots, see Amazon Redshift snapshots (p. 292).

All snapshot tasks in the Amazon Redshift console start from the snapshot list. You can filter the list by using a time range, the snapshot type, and the cluster associated with the snapshot. In addition, you can sort the list by date, size, and snapshot type. Depending on the snapshot type that you select, you might have different options available for working with the snapshot.

Topics
• Creating a snapshot schedule (p. 302)
• Creating a manual snapshot (p. 302)
• Changing the manual snapshot retention period (p. 302)
• Deleting manual snapshots (p. 303)
• Copying an automated snapshot (p. 303)
Creating a snapshot schedule

To precisely control when snapshots are taken, you can create a snapshot schedule and attach it to one or more clusters. You can attach a schedule when you create a cluster or by modifying the cluster. For more information, see Automated snapshot schedules (p. 293).

To create a snapshot schedule

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, Snapshots, then choose the Snapshot schedules tab. The snapshot schedules are displayed.
3. Choose Add schedule to display the page to add a schedule.
4. Enter the properties of the schedule definition, then choose Add schedule.
5. On the page that appears, you can attach clusters to your new snapshot schedule, then choose OK.

Creating a manual snapshot

You can create a manual snapshot of a cluster from the snapshots list as follows. Or, you can take a snapshot of a cluster in the cluster configuration pane. For more information, see Creating a snapshot of a cluster (p. 75).

To create a manual snapshot

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, Snapshots, then choose Create snapshot. The snapshot page to create a manual snapshot is displayed.
3. Enter the properties of the snapshot definition, then choose Create snapshot. It might take some time for the snapshot to be available.

Changing the manual snapshot retention period

You can change the retention period for a manual snapshot by modifying the snapshot settings.

To change the manual snapshot retention period

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, Snapshots, then choose the manual snapshot to change.
3. For Actions, choose Manual snapshot settings to display the properties of the manual snapshot.
4. Enter the revised properties of the snapshot definition, then choose Save.
Deleting manual snapshots

You can delete manual snapshots by selecting one or more snapshots in the snapshot list.

**To delete a manual snapshot**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Clusters, Snapshots**, then choose the snapshot to delete.
3. For **Actions**, choose **Delete snapshot** to delete the snapshot.
4. Confirm the deletion of the listed snapshots, then choose **Delete**.

Copying an automated snapshot

Automated snapshots are automatically deleted when their retention period expires, when you disable automated snapshots, or when you delete a cluster. If you want to keep an automated snapshot, you can copy it to a manual snapshot.

**To copy an automated snapshot**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Clusters, Snapshots**, then choose the snapshot to copy.
3. For **Actions**, choose **Copy automated snapshot** to copy the snapshot.
4. Update the properties of the new snapshot, then choose **Copy**.

Restoring a cluster from a snapshot

When you restore a cluster from a snapshot, Amazon Redshift creates a new cluster with all the snapshot data on the new cluster.

**To restore a cluster from a snapshot**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Clusters, Snapshots**, then choose the snapshot to restore.
3. Choose **Restore from snapshot** to view the **Cluster configuration** and **Cluster details** values of the new cluster to be created using the snapshot information.
4. Update the properties of the new cluster, then choose **Restore cluster from snapshot**.

If you have reserved nodes, for example DS2 or DC2 reserved nodes, you can upgrade to RA3 reserved nodes. You can do this when you restore from a snapshot or perform an elastic resize. You can use the console to guide you through this process. For more information about upgrading to RA3 nodes, see Upgrading to RA3 node types.

Sharing a cluster snapshot

You can authorize other users to access a manual snapshot you own, and you can later revoke that access when it is no longer required.
To share a snapshot with another account

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, Snapshots, then choose the manual snapshot to share.
3. For Actions, choose Manual snapshot settings to display the properties of the manual snapshot.
4. Enter the account or accounts to share with in the Manage access section, then choose Save.

Security considerations for sharing encrypted snapshots

When you provide access to an encrypted snapshot, Redshift requires that the AWS KMS customer managed key used to create the snapshot is shared with the account(s) performing the restore. If the key isn't shared, attempting to restore the snapshot results in an access-denied error.

The customer managed key policy can be updated programmatically or on the console, using the AWS Key Management Service.

Allowing access to the AWS KMS key for an encrypted snapshot

To share the AWS KMS customer managed key for an encrypted snapshot, update the key policy by performing the following steps:

1. Update the KMS key policy with the Amazon Resource Name (ARN) of the AWS account that you are sharing to as Principal in the KMS key policy.
2. Allow the kms:Decrypt action.

In the following key-policy example, user 111122223333 is the owner of the KMS key, and user 444455556666 is the account that the key is shared with. This key policy gives the AWS account access to the sample KMS key by including the ARN for the root AWS account identity for user 444455556666 as a Principal for the policy, and by allowing the kms:Decrypt action.

```json
{
   "Id": "key-policy-1",
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Allow use of the key",
         "Effect": "Allow",
         "Principal": {
            "AWS": [
               "arn:aws:iam::111122223333:user/KeyUser",
               "arn:aws:iam::444455556666:root"
            ]
         },
         "Action": [
            "kms:Decrypt"
         ],
         "Resource": "*"
      }
   ]
}
```

After access is granted to the customer managed KMS key, the account that restores the encrypted snapshot must create an AWS Identity and Access Management (IAM) role, or user, if it doesn't already have one. In addition, that AWS account must also attach an IAM policy to that IAM role or user that allows them to restore an encrypted database snapshot, using your KMS key.
Configuring cross-Region snapshot copy for a nonencrypted cluster

You can configure Amazon Redshift to copy snapshots for a cluster to another AWS Region. To configure cross-Region snapshot copy, you need to enable this copy feature for each cluster and configure where to copy snapshots and how long to keep copied automated or manual snapshots in the destination AWS Region. When cross-Region copy is enabled for a cluster, all new manual and automated snapshots are copied to the specified AWS Region. Copied snapshot names are prefixed with `copy:`.

To configure a cross-Region snapshot

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to move snapshots for.
3. For Actions, choose Configure cross-region snapshot.
   The Configure cross-Region dialog box appears.
4. For Copy snapshots, choose Yes.
5. In Destination AWS Region, choose the AWS Region to which to copy snapshots.
6. In Automated snapshot retention period (days), choose the number of days for which you want automated snapshots to be retained in the destination AWS Region before they are deleted.
7. In Manual snapshot retention period, choose the value that represents the number of days for which you want manual snapshots to be retained in the destination AWS Region before they are deleted. If you choose Custom value, the retention period must be between 1 to 3653 days.
8. Choose Save.

Configure cross-Region snapshot copy for an AWS KMS–encrypted cluster

When you launch an Amazon Redshift cluster, you can choose to encrypt it with a root key from the AWS Key Management Service (AWS KMS). AWS KMS keys are specific to an AWS Region. If you want to enable cross-Region snapshot copy for an AWS KMS–encrypted cluster, you must configure a snapshot copy grant for a root key in the destination AWS Region. By doing this, you enable Amazon Redshift to perform encryption operations in the destination AWS Region.

The following procedure describes the process of enabling cross-Region snapshot copy for an AWS KMS-encrypted cluster. For more information about encryption in Amazon Redshift and snapshot copy grants, see Copying AWS KMS–encrypted snapshots to another AWS Region (p. 325).

To configure a cross-Region snapshot for an AWS KMS–encrypted cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to move snapshots for.
3. For **Actions**, choose **Configure cross-region snapshot**.

   The Configure cross-Region dialog box appears.

4. For **Copy snapshots**, choose **Yes**.

5. In **Destination AWS Region**, choose the AWS Region to which to copy snapshots.

6. In **Automated snapshot retention period (days)**, choose the number of days for which you want automated snapshots to be retained in the destination AWS Region before they are deleted.

7. In **Manual snapshot retention period**, choose the value that represents the number of days for which you want manual snapshots to be retained in the destination AWS Region before they are deleted. If you choose **Custom value**, the retention period must be between 1 to 3653 days.

8. Choose **Save**.

### Modifying the retention period for cross-Region snapshot copy

After you configure cross-Region snapshot copy, you might want to change the settings. You can easily change the retention period by selecting a new number of days and saving the changes.

**Warning**

You can’t modify the destination AWS Region after cross-Region snapshot copy is configured. If you want to copy snapshots to a different AWS Region, first disable cross-Region snapshot copy. Then re-enable it with a new destination AWS Region and retention period. Any copied automated snapshots are deleted after you disable cross-Region snapshot copy. Thus, you should determine if there are any that you want to keep and copy them to manual snapshots before disabling cross-Region snapshot copy.

**To modify a cross-Region snapshot**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.

2. On the navigation menu, choose **Clusters**, then choose the cluster that you want to modify snapshots for.

3. For **Actions**, choose **Configure cross-region snapshot** to display the properties of the snapshot.

4. Enter the revised properties of the snapshot definition, then choose **Save**.

### Managing snapshots using the AWS SDK for Java

The following example demonstrates these common operations involving a snapshot:

- Creating a manual cluster snapshot of a cluster.
- Displaying information about all the snapshots of a cluster.
- Deleting manual snapshots of a cluster.

In this example, a snapshot of the cluster is initiated. When the snapshot is successfully created, all manual snapshots for the cluster that were created before the new snapshot are deleted. When creation of the manual snapshot is initiated, the snapshot is not immediately available. Therefore, this example uses a loop to poll for the status of the snapshot by calling the `describeClusterSnapshot` method. It normally takes a few moments for a snapshot to become available after initiation. For more information about snapshots, see Amazon Redshift snapshots (p. 292).
For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). You need to update the code and provide a cluster identifier.

Example

```java
/**
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 * This file is licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License. A copy of
 * the License is located at
 * http://aws.amazon.com/apache2.0/
 * This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
 * CONDITIONS OF ANY KIND, either express or implied. See the License for the
 * specific language governing permissions and limitations under the License.
 */

// snippet-sourcetype:[full-example]
// snippet-sourcerepos:[]
// snippet-sourcelanguage:[]
// snippet-sourcell:[]
// snippet-sourcel:[]
// snippet-sourceauthor:[AWS]
// snippet-source:[]
// snippet-service:[redshift]
// snippet-keyword:[Java]
// snippet-keyword:[Amazon Redshift]
// snippet-keyword:[Code Sample]
// snippet-keyword:[CreateClusterSnapshot]
// snippet-keyword:[DeleteClusterSnapshot]
// snippet-keyword:[DescribeClusterSnapshots]
// snippet-keyword:[full-example]
// snippet-sourcedate:[2019-01-30]
package com.amazonaws.services.redshift;

import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import com.amazonaws.services.redshift.model.*;

public class CreateAndDescribeSnapshot {

    public static AmazonRedshift client;
    public static String clusterIdentifier = "***provide a cluster identifier***";
    public static long sleepTime = 20;

    public static void main(String[] args) throws IOException {
        // Default client using the DefaultAWSCredentialsProviderChain
        client = AmazonRedshiftClientBuilder.defaultClient();

        try {
            // Unique snapshot identifier
            String snapshotId = "my-snapshot-" + (new SimpleDateFormat("yyyy-MM-dd-HH-mm-ss")).format(new Date());

            Date createDate = createManualSnapshot(snapshotId);
            waitForSnapshotAvailable(snapshotId);
            describeSnapshots();
            deleteManualSnapshotsBefore(createDate);
            describeSnapshots();
        } catch (Exception e) {
        }
    }
}
```
private static Date createManualSnapshot(String snapshotId) {
    CreateClusterSnapshotRequest request = new CreateClusterSnapshotRequest()
        .withClusterIdentifier(clusterIdentifier)
        .withSnapshotIdentifier(snapshotId);
    Snapshot snapshot = client.createClusterSnapshot(request);
    System.out.format("Created cluster snapshot: %s
", snapshotId);
    return snapshot.getSnapshotCreateTime();
}

private static void describeSnapshots() {
    DescribeClusterSnapshotsRequest request = new DescribeClusterSnapshotsRequest()
        .withClusterIdentifier(clusterIdentifier);
    DescribeClusterSnapshotsResult result = client.describeClusterSnapshots(request);
    printResultSnapshots(result);
}

private static void deleteManualSnapshotsBefore(Date creationDate) {
    DescribeClusterSnapshotsRequest request = new DescribeClusterSnapshotsRequest()
        .withEndTime(creationDate)
        .withClusterIdentifier(clusterIdentifier)
        .withSnapshotType("manual");
    DescribeClusterSnapshotsResult result = client.describeClusterSnapshots(request);
    for (Snapshot s : result.getSnapshots()) {
        DeleteClusterSnapshotRequest deleteRequest = new DeleteClusterSnapshotRequest()
            .withSnapshotIdentifier(s.getSnapshotIdentifier());
        Snapshot deleteResult = client.deleteClusterSnapshot(deleteRequest);
        System.out.format("Deleted snapshot %s
", deleteResult.getSnapshotIdentifier());
    }
}

private static void printResultSnapshots(DescribeClusterSnapshotsResult result) {
    System.out.println("Snapshot listing:");
    for (Snapshot snapshot : result.getSnapshots()) {
        System.out.format("Identifier: %s
", snapshot.getSnapshotIdentifier());
        System.out.format("Snapshot type: %s
", snapshot.getSnapshotType());
        System.out.format("Snapshot create time: %s
", snapshot.getSnapshotCreateTime());
        System.out.format("Snapshot status: %s\n", snapshot.getStatus());
    }
}

private static Boolean waitForSnapshotAvailable(String snapshotId) throws InterruptedException {
    Boolean snapshotAvailable = false;
    System.out.println("Waiting for snapshot to become available.");
    while (!snapshotAvailable) {
        DescribeClusterSnapshotsResult result = client.describeClusterSnapshots(new
            DescribeClusterSnapshotsRequest()
                .withSnapshotIdentifier(snapshotId));
        String status = (result.getSnapshots()).get(0).getStatus();
        if (status.equalsIgnoreCase("available")) {
            snapshotAvailable = true;
        } else {
            System.out.print(".");
        }
    }
    return snapshotAvailable;
}
Managing snapshots using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift CLI operations to manage snapshots.

- authorize-snapshot-access
- copy-cluster-snapshot
- create-cluster-snapshot
- delete-cluster-snapshot
- describe-cluster-snapshots
- disable-snapshot-copy
- enable-snapshot-copy
- modify-snapshot-copy-retention-period
- restore-from-cluster-snapshot
- revoke-snapshot-access

You can use the following Amazon Redshift API actions to manage snapshots.

- AuthorizeSnapshotAccess
- CopyClusterSnapshot
- CreateClusterSnapshot
- DeleteClusterSnapshot
- DescribeClusterSnapshots
- DisableSnapshotCopy
- EnableSnapshotCopy
- ModifySnapshotCopyRetentionPeriod
- RestoreFromClusterSnapshot
- RevokeSnapshotAccess

For more information about Amazon Redshift snapshots, see Amazon Redshift snapshots (p. 292).
Integrating Amazon Redshift with an AWS Partner

By working with Amazon Redshift, you can integrate with AWS Partners from the **Cluster details** page on the Amazon Redshift console. From the **Cluster details** page, you can speed up your data onboarding into your Amazon Redshift data warehouse with AWS Partner applications. You can also join and analyze data from different sources together with existing data in your cluster. The following AWS Partners can integrate with Amazon Redshift:

- Datacoral
- Etleap
- Fivetran
- Informatica
- SnapLogic
- Stitch
- Upsolver
- Matillion (preview)
- Sisense (preview)

AWS Partners can integrate with Amazon Redshift using the AWS CLI or Amazon Redshift API operations. For more information, see the **AWS CLI Command Reference** or the **Amazon Redshift API Reference**.

Integrating with an AWS Partner using the Amazon Redshift console

Use the following procedure to integrate a cluster with an AWS Partner.

**To integrate an Amazon Redshift cluster with an AWS Partner**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at [https://console.aws.amazon.com/redshift/](https://console.aws.amazon.com/redshift/).
2. On the navigation menu, choose **Clusters**.
3. Choose the cluster that you want to integrate.
4. Choose **Add partner integration**. The **Choose partner** page opens with details about the available AWS Partners.
5. Choose an AWS Partner, then choose **Next**.

More details about the chosen AWS Partner appear, along with details about the cluster that you are integrating. The **Cluster details** section includes information that you provide on the AWS Partner website such as the **Cluster identifier**, **Endpoint**, **Database name**, and **User name** (which is a database user name). This information is sent to the partner that you chose.

6. Choose **Add partner** to open the AWS Partner's website.
7. Configure the integration with your Amazon Redshift cluster on the partner's website. On the partner's website, you can select and configure the data sources that are loaded to your Amazon
Redshift cluster. You can also define additional extract, load, and transform (ELT) transformations to process your business data, join it with other datasets, and build consolidated views for analysis and reporting.

You can view and manage AWS Partner integrations from the cluster details Properties tab. The Integrations section lists the Partner name that you can use to link to the AWS Partner website, the Status of the integration, the Database that receives the data, and the Last successful connection that might have updated the cluster.

The possible status values are as follows:

- Active – The AWS Partner can connect to the cluster and complete configured tasks.
- Inactive – The AWS Partner integration doesn't exist.
- Runtime failure – The AWS Partner can connect to the cluster but can't complete configured tasks.
- Connection failure – The AWS Partner can't connect to the cluster.

After you delete an AWS Partner integration from Amazon Redshift, data continues to flow into your cluster. Complete the delete on the partner's website.
Purchasing Amazon Redshift reserved nodes

Overview

In AWS, the charges that you accrue for using Amazon Redshift are based on compute nodes. Each compute node is billed at an hourly rate. The hourly rate varies depending on factors such as region, node type, and whether the node receives on-demand node pricing or reserved node pricing.

On-demand node pricing is the most expensive, but most flexible option in Amazon Redshift. With on-demand rates, you are charged only for compute nodes that you have in a running cluster. If you shut down or delete a cluster, you are no longer charged for compute nodes that were in that cluster. You are billed only for the compute nodes that you use, and no more. The hourly rate that you are charged for each compute node varies depending on factors such as region and node type.

Reserved node pricing is less expensive than on-demand pricing because compute nodes are billed at discounted hourly rates. However, to receive these discounted rates, you must purchase reserved node offerings. When you purchase an offering, you make a reservation. The reservation sets a discounted rate for each node that you reserve for the duration of the reservation. The discounted rate in an offering varies depending on factors such as the region, node type, duration, and payment option.

You may designate a node as a reserved node by calling the PurchaseReservedNodeOffering API operation or choosing Purchase reserved nodes on the Amazon Redshift console. When you purchase a reserved node, you must specify an AWS Region, node type, term, quantity of nodes, and offering type for the applicable reserved node type. The reserved node may only be used in the designated AWS Region.

This topic discusses what reserved node offerings are and how you can purchase them to reduce the cost of running your Amazon Redshift clusters. This topic discusses rates in general terms as on-demand or discounted so you can understand pricing concepts and how pricing affects billing. For more information about specific rates, go to Amazon Redshift Pricing.

About reserved node offerings

If you intend to keep your Amazon Redshift cluster running continuously for a prolonged period, you should consider purchasing reserved node offerings. These offerings provide significant savings over on-demand pricing, but they require you to reserve compute nodes and commit to paying for those nodes for either a one-year or three-year duration.

Reserved nodes are a billing concept that is used strictly to determine the rate at which you are charged for nodes. Reserving a node does not actually create any nodes for you. You are charged for reserved nodes regardless of usage, which means that you must pay for each node that you reserve for the duration of the reservation, whether or not you have any nodes in a running cluster to which the discounted rate applies.

In the evaluation phase of your project or when you're developing a proof of concept, on-demand pricing gives you the flexibility to pay as you go, to pay only for what you use, and to stop paying at any time by shutting down or deleting clusters. After you have established the needs of your production environment and begin the implementation phase, you should consider reserving compute nodes by purchasing one or more offerings.

An offering can apply to one or more compute nodes. You specify the number of compute nodes to reserve when you purchase the offering. You might choose to purchase one offering for multiple
compute nodes, or you might choose to purchase multiple offerings and specify a certain number of compute nodes in each offering.

For example, any of the following are valid ways to purchase an offering for three compute nodes:

- Purchase one offering and specify three compute nodes.
- Purchase two offerings, and specify one compute node for the first offering and two compute nodes for the second offering.
- Purchase three offerings, and specify one compute node for each of the offerings.

Comparing pricing among reserved node offerings

Amazon Redshift provides several payment options for offerings. The payment option that you choose affects the payment schedule and the discounted rate that you are charged for the reservation. The more that you pay upfront for the reservation, the better the overall savings are.

The following payment options are available for offerings. The offerings are listed in order from least to most savings over on-demand rates.

<table>
<thead>
<tr>
<th>Payment option</th>
<th>Payment schedule</th>
<th>Comparative savings</th>
<th>Duration</th>
<th>Upfront charges</th>
<th>Recurring monthly charges</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Upfront</td>
<td>Monthly installments for the duration of the reservation. No upfront payment.</td>
<td>About a 20 percent discount over on-demand rates.</td>
<td>One-year term</td>
<td>None</td>
<td>Yes</td>
</tr>
<tr>
<td>Partial Upfront</td>
<td>Partial upfront payment, and monthly installments for the duration of the reservation.</td>
<td>Up to 41 percent to 73 percent discount depending on duration.</td>
<td>One-year or three-year term</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>All Upfront</td>
<td>Full upfront payment for the reservation. No monthly charges.</td>
<td>Up to 42 percent to 76 percent discount depending on duration.</td>
<td>One-year or three-year term</td>
<td>Yes</td>
<td>None</td>
</tr>
</tbody>
</table>

Note
You are charged the applicable hourly rate for every hour in the specified duration of the reservation, regardless of whether you use the reserved node or not. The payment option just determines the frequency of payments and the discount to be applied. For more information, see About reserved node offerings (p. 312).

How reserved nodes work

With reserved node offerings, you pay according to the payment terms as described in the preceding section. You pay this way whether you already have a running cluster or you launch a cluster after you have a reservation.
When you purchase an offering, your reservation has a status of **payment-pending** until the reservation is processed. If the reservation fails to be processed, the status displays as **payment-failed** and you can try the process again. Once your reservation is successfully processed, its status changes to **active**. The applicable discounted rate in your reservation is not applied to your bill until the status changes to **active**. After the reservation duration elapses, the status changes to **retired** but you can continue to access information about the reservation for historical purposes. When a reservation is **retired**, your clusters continue to run but you might be billed at the on-demand rate unless you have another reservation that applies discounted pricing to the nodes.

Reserved nodes are specific to the region in which you purchase the offering. If you purchase an offering by using the Amazon Redshift console, select the AWS region in which you want to purchase an offering, and then complete the reservation process. If you purchase an offering programmatically, the region is determined by the Amazon Redshift endpoint that you connect to. For more information about Amazon Redshift regions, go to Regions and Endpoints in the Amazon Web Services General Reference.

To ensure that the discounted rate is applied to all of the nodes when you launch a cluster, make sure that the region, the node type, and the number of nodes that you select match one or more active reservations. Otherwise, you'll be charged at the on-demand rate for nodes that don't match an active reservation.

In a running cluster, if you exceed the number of nodes that you have reserved, you begin to accrue charges for those additional nodes at the on-demand rate. This accrual means that it is possible for you to be charged varying rates for nodes in the same cluster depending on how many nodes you've reserved. You can purchase another offering to cover those additional nodes, and then the discounted rate is applied to those nodes for the remainder of the duration once the reservation status becomes **active**.

If you resize your cluster into a different node type and you haven't reserved nodes of that type, you'll be charged at the on-demand rate. You can purchase another offering with the new node type if you want to receive discounted rates for your resized cluster. However, you also continue to pay for the original reservation until its duration elapses. If you need to alter your reservations before the term expires, please create a support case using the AWS Console.

## Reserved nodes and consolidated billing

The pricing benefits of Reserved Nodes are shared when the purchasing account is part of a set of accounts billed under one consolidated billing payer account. The hourly usage across all sub-accounts is aggregated in the payer account every month. This is typically useful for companies in which there are different functional teams or groups; then, the normal Reserved Nodes logic is applied to calculate the bill. For more information, see Consolidated Billing in the AWS Billing User Guide.

## Reserved node examples

The scenarios in this section demonstrate how nodes accrue charges based on on-demand and discounted rates using the following reservation details:

- **Region**: US West (Oregon)
- **Node Type**: ds2.xlarge
- **Payment Option**: No Upfront
- **Duration**: one year
- **Number of Reserved Nodes**: 16

### Example 1

You have one ds2.xlarge cluster in the US West (Oregon) region with 20 nodes.
In this scenario, 16 of the nodes receive the discounted rate from the reservation, but the additional 4 nodes in the cluster are billed at the on-demand rate.

**Example 2**

You have one ds2.xlarge cluster in the US West (Oregon) region with 12 nodes. In this scenario, all 12 nodes in the cluster receive the discounted rate from the reservation. However, you also pay for the remaining reserved nodes in the reservation even though you don't currently have a running cluster to which they apply.

**Example 3**

You have one ds2.xlarge cluster in the US West (Oregon) region with 12 nodes. You run the cluster for several months with this configuration, and then you need to add nodes to the cluster. You resize the cluster, choosing the same node type and specifying a total of 16 nodes. In this scenario, you are billed the discounted rate for 16 nodes. Your charges remain the same for the full year duration because the number of nodes that you have in the cluster is equal to the number of nodes that you have reserved.

**Example 4**

You have one ds2.xlarge cluster in the US West (Oregon) region with 16 nodes. You run the cluster for several months with this configuration, and then you need to add nodes. You resize the cluster, choosing the same node type and specifying a total of 20 nodes. In this scenario, you are billed the discounted rate for all the nodes prior to the resize. After the resize, you are billed the discounted rate for 16 of the nodes for the rest of the year, and you are billed at the on-demand rate for the additional 4 nodes that you added to the cluster.

**Example 5**

You have two ds2.xlarge clusters in the US West (Oregon) region. One of the clusters has 6 nodes, and the other has 10 nodes. In this scenario, you are billed at the discounted rate for all of the nodes because the total number of nodes in both clusters is equal to the number of nodes that you have reserved.

**Example 6**

You have two ds2.xlarge clusters in the US West (Oregon) region. One of the clusters has 4 nodes, and the other has 6 nodes. In this scenario, you are billed the discounted rate for the 10 nodes that you have in running clusters, and you also pay the discounted rate for the additional 6 nodes that you have reserved even though you don't currently have any running clusters to which they apply.

**Purchasing a reserved node offering with the Amazon Redshift console**

You use the **Reserved Nodes** page in the Amazon Redshift console to purchase reserved node offerings, and to view current and past reservations.
After you purchase an offering, the Reserved Node list displays your reservations and the details of each one, such as the node type, number of nodes, and status of the reservation. For more information about the reservation details, see How reserved nodes work (p. 313).

To purchase a reserved node

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose Reserved nodes to display the list of reserved nodes.
3. Choose Purchase reserved nodes to display the page to choose the properties of the node that you want to purchase.
4. Enter the properties of the node, then choose Purchase reserved nodes.

To upgrade a reserved node, use the AWS CLI.

Upgrading reserved nodes with the AWS CLI

To upgrade a reserved node reservation with the AWS CLI

1. Obtain a list of ReservedNodeOfferingID’s for offerings that meet your requirements for payment type, term, and charges. The following example illustrates this step.

   ```bash
   aws redshift get-reserved-node-exchange-offerings --reserved-node-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
   { "ReservedNodeOfferings": [ 
   { "Duration": 3153600, 
   "ReservedNodeOfferingId": "yyyyyyyy-yyy-yyyy-yyyy-yyyy-yyyyyyyyyyyy", 
   "UsagePrice": 0.0, 
   "NodeType": "dc2.large", 
   "RecurringCharges": [ 
   { "RecurringChargeFrequency": "Hourly", 
   "RecurringChargeAmount": 0.2
   } ]}, 
   "CurrencyCode": "USD", 
   "OfferingType": "No Upfront", 
   "ReservedNodeOfferingType": "Regular", 
   "FixedPrice": 0.0
   }
   ]
   }
   
   aws redshift accept-reserved-node-exchange --reserved-node-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx --target-reserved-node-offering-id yyyyyyyy-yyyy-yyyy-yyyy-yyyyyyyyyyyyy
   { "ExchangedReservedNode": { 
   "UsagePrice": 0.0, 
   "OfferingType": "No Upfront", 
   
2. Call accept-reserved-node-exchange and provide the ID for the DC1 reserved node that you want to exchange along with the ReservedNodeOfferingID you obtained in the previous step.

   The following example illustrates this step.

   ```bash
   aws redshift accept-reserved-node-exchange --reserved-node-id xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx --target-reserved-node-offering-id yyyyyyyy-yyyy-yyyy-yyyy-yyyyyyyyyyyyy
   { "ExchangedReservedNode": { 
   "UsagePrice": 0.0, 
   "OfferingType": "No Upfront", 
   ```
"State": "exchanging",
"FixedPrice": 0.0,
"CurrencyCode": "USD",
"ReservedNodeId": "zzzzzzzz-zzzz-zzzz-zzzz-zzzzzzzzzzzz",
"NodeType": "dc2.large",
"NodeCount": 1,
"RecurringCharges": [
  {
    "RecurringChargeFrequency": "Hourly",
    "RecurringChargeAmount": 0.2
  }
],
"ReservedNodeOfferingType": "Regular",
"StartTime": "2018-06-27T18:02:58Z",
"ReservedNodeOfferingId": "yyyyyyyy-yyyy-yyyy-yyyy-yyyyyyyyyyyyy",
"Duration": 31536000
}

import com.amazonaws.services.redshift.model娲
import java.util.List
import java.util.ArrayList
import java.util.Arrays

You can confirm that the exchange is complete by calling describe-reserved-nodes and checking the value for Node type.

Purchasing a reserved node offering using the AWS SDK for Java

The following example demonstrates how to use the AWS SDK for Java to do the following:

- List existing reserved nodes.
- Search for a new reserved node offering based on specified node criteria.
- Purchase a reserved node.

This example, first selects all the reserved node offerings that match a specified node type and fixed price value. Then, this example goes through each offering found and lets you purchase the offering.

Important

If you run this example and accept the offer to purchase a reserved node offering, you will be charged for the offering.

For step-by-step instructions to run this example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). To get information about a node type and fixed price other than those listed, update the code and provide that node type and fixed price.

Example

```java
/**
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 * This file is licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License. A copy of
 * the License is located at
 * http://aws.amazon.com/apache2.0/
 * This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
 * CONDITIONS OF ANY KIND, either express or implied. See the License for the
```
package com.amazonaws.services.redshift;
import java.io.DataInput;
import java.io.DataInputStream;
import java.io.IOException;
import java.util.ArrayList;
import com.amazonaws.services.redshift.model.*;

public class ListAndPurchaseReservedNodeOffering {
    public static AmazonRedshift client;
    public static String nodeTypeToPurchase = "dc2.large";
    public static Double fixedPriceLimit = 10000.00;
    public static ArrayList<ReservedNodeOffering> matchingNodes = new ArrayList<ReservedNodeOffering>();

    public static void main(String[] args) throws IOException {
        // Default client using the @link
        com.amazonaws.auth.DefaultAWSCredentialsProviderChain
        client = AmazonRedshiftClientBuilder.defaultClient();

        try {
            listReservedNodes();
            findReservedNodeOffer();
            purchaseReservedNodeOffer();
        } catch (Exception e) {
            System.err.println("Operation failed: "+ e.getMessage());
        }
    }

    private static void listReservedNodes() {
        DescribeReservedNodesResult result = client.describeReservedNodes();
        System.out.println("Listing nodes already purchased.");
        for (ReservedNode node : result.getReservedNodes()) {
            printReservedNodeDetails(node);
        }
    }

    private static void findReservedNodeOffer() {
        DescribeReservedNodeOfferingsRequest request = new DescribeReservedNodeOfferingsRequest();
        DescribeReservedNodeOfferingsResult result = client.describeReservedNodeOfferings(request);
        Integer count = 0;
    }

    private static void purchaseReservedNodeOffer() {
    }
}
Purchasing a reserved node offering using Java

```java
System.out.println("Finding nodes to purchase.");
for (ReservedNodeOffering offering : result.getReservedNodeOfferings()) {
    if (offering.getNodeType().equals(nodeTypeToPurchase)) {
        if (offering.getFixedPrice() < fixedPriceLimit) {
            matchingNodes.add(offering);
            printOfferingDetails(offering);
            count += 1;
        }
    }
}
if (count == 0) {
    System.out.println("No reserved node offering matches found.");
} else {
    System.out.println("Found " + count + " matches.");
}

private static void purchaseReservedNodeOffer() throws IOException {
    if (matchingNodes.size() == 0) {
        return;
    } else {
        System.out.println("Purchasing nodes.");
        for (ReservedNodeOffering offering : matchingNodes) {
            printOfferingDetails(offering);
            System.out.println("WARNING: purchasing this offering will incur costs.");
            System.out.println("Purchase this offering [Y or N]?\n");
            DataInputStream in = new DataInputStream(System.in);
            String purchaseOpt = in.readLine();
            if (purchaseOpt.equalsIgnoreCase("y")) {
                try {
                    PurchaseReservedNodeOfferingRequest request = new PurchaseReservedNodeOfferingRequest()
                        .withReservedNodeOfferingId(offering.getReservedNodeOfferingId());
                    ReservedNode reservedNode = client.purchaseReservedNodeOffering(request);
                    printReservedNodeDetails(reservedNode);
                } catch (ReservedNodeAlreadyExistsException ex1) {
                } catch (ReservedNodeOfferingNotFoundException ex2) {
                } catch (ReservedNodeQuotaExceededException ex3) {
                } catch (Exception ex4) {
                }
            }
        }
        System.out.println("Finished.");
    }
}

private static void printOfferingDetails(ReservedNodeOffering offering) {
    System.out.println("Offering Match:");
    System.out.format("Id: %s\n", offering.getReservedNodeOfferingId());
    System.out.format("NodeType: %s\n", offering.getNodeType());
    System.out.format("FixedPrice: %s\n", offering.getFixedPrice());
    System.out.format("OfferingType: %s\n", offering.getOfferingType());
    System.out.format("Duration: %s\n", offering.getDuration());
}
```
Purchasing a reserved node offering using the AWS CLI and Amazon Redshift API

You can use the following AWS CLI operations to purchase reserved node offerings.

- `purchase-reserved-node-offering`
- `describe-reserved-node-offerings`
- `describe-orderable-cluster-options`

You can use the following Amazon Redshift API operations to purchase reserved node offerings.

- `PurchaseReservedNodeOffering`
- `DescribeReservedNodeOfferings`
- `DescribeOrderableClusterOptions`
Security in Amazon Redshift

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. The effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS compliance programs. To learn about the compliance programs that apply to Amazon Redshift, see AWS services in scope by compliance program.

- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your organization's requirements, and applicable laws and regulations.

Access to Amazon Redshift resources is controlled at four levels:

- **Cluster management** – The ability to create, configure, and delete clusters is controlled by the permissions given to the IAM user or account associated with your AWS security credentials. IAM users with the proper permissions can use the AWS Management Console, AWS Command Line Interface (CLI), or Amazon Redshift Application Programming Interface (API) to manage their clusters. This access is managed by using IAM policies. For details, see Identity and access management in Amazon Redshift (p. 334).

- **Cluster connectivity** – Amazon Redshift security groups specify the AWS instances that are authorized to connect to an Amazon Redshift cluster in Classless Inter-Domain Routing (CIDR) format. For information about creating Amazon Redshift, Amazon EC2, and Amazon VPC security groups and associating them with clusters, see Amazon Redshift cluster security groups (p. 438).

- **Database access** – The ability to access database objects, such as tables and views, is controlled by user accounts in the Amazon Redshift database. Users can only access resources in the database that their user accounts have been granted permission to access. You create these Amazon Redshift user accounts and manage permissions by using the CREATE USER, CREATE GROUP, GRANT, and REVOKE SQL statements. For more information, see Managing database security in the Amazon Redshift Database Developer Guide.

- **Temporary database credentials and single sign-on** – In addition to creating and managing database users using SQL commands, such as CREATE USER and ALTER USER, you can configure your SQL client with custom Amazon Redshift JDBC or ODBC drivers. These drivers manage the process of creating database users and temporary passwords as part of the database logon process.

The drivers authenticate database users based on AWS Identity and Access Management (IAM) authentication. If you already manage user identities outside of AWS, you can use a SAML 2.0-compliant identity provider (IdP) to manage access to Amazon Redshift resources. You use an IAM role to configure your IdP and AWS to permit your federated users to generate temporary database credentials and log on to Amazon Redshift databases. For more information, see Using IAM authentication to generate database user credentials (p. 370).

This documentation helps you understand how to apply the shared responsibility model when using Amazon Redshift. The following topics show you how to configure Amazon Redshift to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Amazon Redshift resources.
Data protection in Amazon Redshift

The AWS shared responsibility model applies to data protection in Amazon Redshift. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the AWS Security Blog.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers’ email addresses, into tags or free-form fields such as a Name field. This includes when you work with Amazon Redshift or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

AQUA (Advanced Query Accelerator) is a multitenant service running on servers that are separate from the Amazon Redshift cluster. AQUA supports authentication, encryption, isolation, and compliance to keep your data at rest and data in transit secure. The same permissions to access Amazon Redshift are used for AQUA. Queries processed on AQUA servers are run in isolated processes to ensure data protection.

Data encryption

Data protection refers to protecting data while in-transit (as it travels to and from Amazon Redshift) and at rest (while it is stored on disks in Amazon Redshift data centers). You can protect data in transit by using SSL or by using client-side encryption. You have the following options of protecting data at rest in Amazon Redshift.
- **Use server-side encryption** – You request Amazon Redshift to encrypt your data before saving it on disks in its data centers and decrypt it when you download the objects.

- **Use client-side encryption** – You can encrypt data client-side and upload the encrypted data to Amazon Redshift. In this case, you manage the encryption process, the encryption keys, and related tools.

### Encryption at rest

Server-side encryption is about data encryption at rest—that is, Amazon Redshift optionally encrypts your data as it writes it in its data centers and decrypts it for you when you access it. As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted data.

Amazon Redshift protects data at rest through encryption. Optionally, you can protect all data stored on disks within a cluster and all backups in Amazon S3 with Advanced Encryption Standard AES-256.

To manage the keys used for encrypting and decrypting your Amazon Redshift resources, you use [AWS Key Management Service (AWS KMS)](https://aws.amazon.com/kms/). AWS KMS combines secure, highly available hardware and software to provide a key management system scaled for the cloud. Using AWS KMS, you can create encryption keys and define the policies that control how these keys can be used. AWS KMS supports AWS CloudTrail, so you can audit key usage to verify that keys are being used appropriately. You can use your AWS KMS keys in combination with Amazon Redshift and supported AWS services. For a list of services that support AWS KMS, see [How AWS Services Use AWS KMS](https://docs.aws.amazon.com/kms/latest/developerguide/using-with-aws-services.html) in the AWS Key Management Service Developer Guide.

AQUA (Advanced Query Accelerator) receives and uses the same database encryption keys (DEKs) used by Amazon Redshift to store the data snapshots in Amazon S3. Amazon Redshift uses envelope encryption, which means that you can rotate keys without having to re-encrypt data blocks. If the Amazon Redshift cluster isn't encrypted, then AQUA uses its own managed keys to encrypt the data. For more information, see Amazon Redshift database encryption (p. 323).

In some cases, you might revoke AWS KMS access to an IAM role associated with a cluster. If so, after the in-flight queries are complete, no further access to the data cached in AQUA is possible. This is because AQUA doesn't persist keys beyond a single pushed-down scan or aggregate operation. This lack of persistence also means that you can't access any metadata about those blocks.

Amazon Redshift query editor v2 securely stores information entered into the query editor as follows:

- The Amazon Resource Name (ARN) of the KMS key used to encrypt query editor v2 data.
- Database connection information.
- Names and content of files and folders.

Amazon Redshift query editor v2 encrypts information using block-level encryption with either your KMS key or the service account KMS key. The encryption of your Amazon Redshift data is controlled by your Amazon Redshift cluster properties.

### Topics

- [Amazon Redshift database encryption](#)

### Amazon Redshift database encryption

In Amazon Redshift, you can enable database encryption for your clusters to help protect data at rest. When you enable encryption for a cluster, the data blocks and system metadata are encrypted for the cluster and its snapshots.
You can enable encryption when you launch your cluster, or you can modify an unencrypted cluster to use AWS Key Management Service (AWS KMS) encryption. To do so, you can use either an AWS-managed key or a customer managed key. When you modify your cluster to enable AWS KMS encryption, Amazon Redshift automatically migrates your data to a new encrypted cluster. Snapshots created from the encrypted cluster are also encrypted. You can also migrate an encrypted cluster to an unencrypted cluster by modifying the cluster and changing the Encrypt database option. For more information, see Changing cluster encryption (p. 327).

Though encryption is an optional setting in Amazon Redshift, we recommend that you enable it for clusters that contain sensitive data. Additionally, you might be required to use encryption depending on the guidelines or regulations that govern your data. For example, the Payment Card Industry Data Security Standard (PCI DSS), the Sarbanes-Oxley Act (SOX), the Health Insurance Portability and Accountability Act (HIPAA), and other such regulations provide guidelines for handling specific types of data.

Amazon Redshift uses a hierarchy of encryption keys to encrypt the database. You can use either AWS Key Management Service (AWS KMS) or a hardware security module (HSM) to manage the top-level encryption keys in this hierarchy. The process that Amazon Redshift uses for encryption differs depending on how you manage keys. Amazon Redshift automatically integrates with AWS KMS but not with an HSM. When you use an HSM, you must use client and server certificates to configure a trusted connection between Amazon Redshift and your HSM.

**Topics**
- Database encryption for Amazon Redshift using AWS KMS (p. 324)
- Encryption for Amazon Redshift using hardware security modules (p. 326)
- Encryption key rotation in Amazon Redshift (p. 327)
- Changing cluster encryption (p. 327)
- Configuring database encryption using the console (p. 330)
- Configuring database encryption using the Amazon Redshift API and AWS CLI (p. 330)

**Database encryption for Amazon Redshift using AWS KMS**

When you choose AWS KMS for key management with Amazon Redshift, there is a four-tier hierarchy of encryption keys. These keys, in hierarchical order, are the root key, a cluster encryption key (CEK), a database encryption key (DEK), and data encryption keys.

When you launch your cluster, Amazon Redshift returns a list of the AWS KMS keys that your AWS account has created or has permission to use in AWS KMS. You select a KMS key to use as your root key in the encryption hierarchy.

By default, Amazon Redshift selects your default key as the root key. Your default key is an AWS-managed key that is created for your AWS account to use in Amazon Redshift. AWS KMS creates this key the first time you launch an encrypted cluster in an AWS Region and choose the default key.

If you don't want to use the default key, you must have (or create) a customer managed KMS key separately in AWS KMS before you launch your cluster in Amazon Redshift. Customer managed keys give you more flexibility, including the ability to create, rotate, disable, define access control for, and audit the encryption keys used to help protect your data. For more information about creating KMS keys, see Creating Keys in the AWS Key Management Service Developer Guide.

If you want to use a AWS KMS key from another AWS account, you must have permission to use the key and specify its Amazon Resource Name (ARN) in Amazon Redshift. For more information about access to keys in AWS KMS, see Controlling Access to Your Keys in the AWS Key Management Service Developer Guide.

After you choose a root key, Amazon Redshift requests that AWS KMS generate a data key and encrypt it using the selected root key. This data key is used as the CEK in Amazon Redshift. AWS KMS exports the
encrypted CEK to Amazon Redshift, where it is stored internally on disk in a separate network from the cluster along with the grant to the KMS key and the encryption context for the CEK. Only the encrypted CEK is exported to Amazon Redshift; the KMS key remains in AWS KMS. Amazon Redshift also passes the encrypted CEK over a secure channel to the cluster and loads it into memory. Then, Amazon Redshift calls AWS KMS to decrypt the CEK and loads the decrypted CEK into memory. For more information about grants, encryption context, and other AWS KMS-related concepts, see Concepts in the AWS Key Management Service Developer Guide.

Next, Amazon Redshift randomly generates a key to use as the DEK and loads it into memory in the cluster. The decrypted CEK is used to encrypt the DEK, which is then passed over a secure channel from the cluster to be stored internally by Amazon Redshift on disk in a separate network from the cluster. Like the CEK, both the encrypted and decrypted versions of the DEK are loaded into memory in the cluster. The decrypted version of the DEK is then used to encrypt the individual encryption keys that are randomly generated for each data block in the database.

When the cluster reboots, Amazon Redshift starts with the internally stored, encrypted versions of the CEK and DEK, reloads them into memory, and then calls AWS KMS to decrypt the CEK with the KMS key again so it can be loaded into memory. The decrypted CEK is then used to decrypt the DEK again, and the decrypted DEK is loaded into memory and used to encrypt and decrypt the data block keys as needed.

For more information about creating Amazon Redshift clusters that are encrypted with AWS KMS keys, see Creating a cluster (p. 71) and Managing clusters using the AWS CLI and Amazon Redshift API (p. 76).

Copying AWS KMS–encrypted snapshots to another AWS Region

AWS KMS keys are specific to an AWS Region. If you enable copying of Amazon Redshift snapshots to another AWS Region, and the source cluster and its snapshots are encrypted using a root key from AWS KMS, you need to configure a grant for Amazon Redshift to use a root key in the destination AWS Region. This grant enables Amazon Redshift to encrypt snapshots in the destination AWS Region. For more information about cross-Region snapshot copy, see Copying snapshots to another AWS Region (p. 295).

**Note**

If you enable copying of snapshots from an encrypted cluster and use AWS KMS for your root key, you cannot rename your cluster because the cluster name is part of the encryption context. If you must rename your cluster, you can disable copying of snapshots in the source AWS Region, rename the cluster, and then configure and enable copying of snapshots again.

The process to configure the grant for copying snapshots is as follows.

1. In the destination AWS Region, create a snapshot copy grant by doing the following:
   - If you do not already have an AWS KMS key to use, create one. For more information about creating AWS KMS keys, see Creating Keys in the AWS Key Management Service Developer Guide.
   - Specify a name for the snapshot copy grant. This name must be unique in that AWS Region for your AWS account.
   - Specify the AWS KMS key ID for which you are creating the grant. If you do not specify a key ID, the grant applies to your default key.

2. In the source AWS Region, enable copying of snapshots and specify the name of the snapshot copy grant that you created in the destination AWS Region.

This preceding process is only necessary if you enable copying of snapshots using the AWS CLI, the Amazon Redshift API, or SDKs. If you use the console, Amazon Redshift provides the proper workflow to configure the grant when you enable cross-Region snapshot copy. For more information about configuring cross-Region snapshot copy for AWS KMS–encrypted clusters by using the console, see Configure cross-Region snapshot copy for an AWS KMS–encrypted cluster (p. 305).

Before the snapshot is copied to the destination AWS Region, Amazon Redshift decrypts the snapshot using the root key in the source AWS Region and re-encrypts it temporarily using a randomly generated
RSA key that Amazon Redshift manages internally. Amazon Redshift then copies the snapshot over a secure channel to the destination AWS Region, decrypts the snapshot using the internally managed RSA key, and then re-encrypts the snapshot using the root key in the destination AWS Region.

For more information about configuring snapshot copy grants for AWS KMS-encrypted clusters, see Configuring Amazon Redshift to use AWS KMS encryption keys using the Amazon Redshift API and AWS CLI (p. 330).

Encryption for Amazon Redshift using hardware security modules

If you don't use AWS KMS for key management, you can use a hardware security module (HSM) for key management with Amazon Redshift.

**Important**

HSM encryption is not supported for DC2 and RA3 node types.

HSMs are devices that provide direct control of key generation and management. They provide greater security by separating key management from the application and database layers. Amazon Redshift supports AWS CloudHSM Classic for key management. The encryption process is different when you use HSM to manage your encryption keys instead of AWS KMS.

**Important**

Amazon Redshift supports only AWS CloudHSM Classic. We don't support the newer AWS CloudHSM service.

AWS CloudHSM Classic is closed to new customers. For more information, see CloudHSM Classic Pricing. AWS CloudHSM Classic isn't available in all AWS Regions. For more information about available AWS Regions, see AWS Region Table.

When you configure your cluster to use an HSM, Amazon Redshift sends a request to the HSM to generate and store a key to be used as the CEK. However, unlike AWS KMS, the HSM doesn't export the CEK to Amazon Redshift. Instead, Amazon Redshift randomly generates the DEK in the cluster and passes it to the HSM to be encrypted by the CEK. The HSM returns the encrypted DEK to Amazon Redshift, where it is further encrypted using a randomly-generated, internal root key and stored internally on disk in a separate network from the cluster. Amazon Redshift also loads the decrypted version of the DEK in memory in the cluster so that the DEK can be used to encrypt and decrypt the individual keys for the data blocks.

If the cluster is rebooted, Amazon Redshift decrypts the internally-stored, double-encrypted DEK using the internal root key to return the internally stored DEK to the CEK-encrypted state. The CEK-encrypted DEK is then passed to the HSM to be decrypted and passed back to Amazon Redshift, where it can be loaded in memory again for use with the individual data block keys.

**Configuring a trusted connection between Amazon Redshift and an HSM**

When you opt to use an HSM for management of your cluster key, you need to configure a trusted network link between Amazon Redshift and your HSM. Doing this requires configuration of client and server certificates. The trusted connection is used to pass the encryption keys between the HSM and Amazon Redshift during encryption and decryption operations.

Amazon Redshift creates a public client certificate from a randomly generated private and public key pair. These are encrypted and stored internally. You download and register the public client certificate in your HSM, and assign it to the applicable HSM partition.

You provide Amazon Redshift with the HSM IP address, HSM partition name, HSM partition password, and a public HSM server certificate, which is encrypted by using an internal root key. Amazon Redshift completes the configuration process and verifies that it can connect to the HSM. If it cannot, the cluster is put into the INCOMPATIBLE_HSM state and the cluster is not created. In this case, you must delete the incomplete cluster and try again.
Important
When you modify your cluster to use a different HSM partition, Amazon Redshift verifies that it can connect to the new partition, but it does not verify that a valid encryption key exists. Before you use the new partition, you must replicate your keys to the new partition. If the cluster is restarted and Amazon Redshift cannot find a valid key, the restart fails. For more information, see Replicating Keys Across HSMs.

After initial configuration, if Amazon Redshift fails to connect to the HSM, an event is logged. For more information about these events, see Amazon Redshift Event Notifications.

Encryption key rotation in Amazon Redshift

In Amazon Redshift, you can rotate encryption keys for encrypted clusters. When you start the key rotation process, Amazon Redshift rotates the CEK for the specified cluster and for any automated or manual snapshots of the cluster. Amazon Redshift also rotates the DEK for the specified cluster, but cannot rotate the DEK for the snapshots while they are stored internally in Amazon Simple Storage Service (Amazon S3) and encrypted using the existing DEK.

While the rotation is in progress, the cluster is put into a ROTATING KEYS state until completion, at which time the cluster returns to the AVAILABLE state. Amazon Redshift handles decryption and re-encryption during the key rotation process.

Note
You cannot rotate keys for snapshots without a source cluster. Before you delete a cluster, consider whether its snapshots rely on key rotation.

Because the cluster is momentarily unavailable during the key rotation process, you should rotate keys only as often as your data needs require or when you suspect the keys might have been compromised. As a best practice, you should review the type of data that you store and plan how often to rotate the keys that encrypt that data. The frequency for rotating keys varies depending on your corporate policies for data security, and any industry standards regarding sensitive data and regulatory compliance. Ensure that your plan balances security needs with availability considerations for your cluster.

For more information about rotating keys, see Rotating encryption keys using the Amazon Redshift console (p. 330) and Rotating encryption keys using the Amazon Redshift API and AWS CLI (p. 331).

Changing cluster encryption

You can modify an unencrypted cluster to use AWS Key Management Service (AWS KMS) encryption, using either an AWS-managed key or a customer managed key. When you modify your cluster to enable AWS KMS encryption, Amazon Redshift automatically migrates your data to a new encrypted cluster. You can also migrate an unencrypted cluster to an encrypted cluster by modifying the cluster.

During the migration operation, your cluster is available in read-only mode, and the cluster status appears as resizing.

If your cluster is configured to enable cross-AWS Region snapshot copy, you must disable it before changing encryption. For more information, see Copying snapshots to another AWS Region (p. 295) and Configure cross-Region snapshot copy for an AWS KMS–encrypted cluster (p. 305). You can't enable hardware security module (HSM) encryption by modifying the cluster. Instead, create a new, HSM-encrypted cluster and migrate your data to the new cluster. For more information, see Migrating to an HSM-encrypted cluster (p. 328).

To modify database encryption on a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to modify encryption.
3. Choose **Properties**.
4. In the **Database configurations** section, choose **Edit**, then choose **Edit encryption**.
5. Choose one of the encryption options and choose **Save changes**.

**To change cluster encryption using the CLI**

To modify your unencrypted cluster to use AWS KMS, run the `modify-cluster` CLI command and specify `--encrypted`, as shown following. By default, your default KMS key is used. To specify a customer managed key, include the `--kms-key-id` option.

```
aws redshift modify-cluster --cluster-identifier <value> --encrypted --kms-key-id <value>
```

To remove encryption from your cluster, run the following CLI command.

```
aws redshift modify-cluster --cluster-identifier <value> --no-encrypted
```

**Migrating to an HSM-encrypted cluster**

To migrate an unencrypted cluster to a cluster encrypted using a hardware security module (HSM), you create a new encrypted cluster and move your data to the new cluster. You can’t migrate to an HSM-encrypted cluster by modifying the cluster.

To migrate from an unencrypted cluster to an HSM-encrypted cluster, you first unload your data from the existing, source cluster. Then you reload the data in a new, target cluster with the chosen encryption setting. For more information about launching an encrypted cluster, see Amazon Redshift database encryption (p. 323).

During the migration process, your source cluster is available for read-only queries until the last step. The last step is to rename the target and source clusters, which switches endpoints so all traffic is routed to the new, target cluster. The target cluster is unavailable until you reboot following the rename. Suspend all data loads and other write operations on the source cluster while data is being transferred.

**To prepare for migration**

1. Identify all the dependent systems that interact with Amazon Redshift, for example business intelligence (BI) tools and extract, transform, and load (ETL) systems.
2. Identify validation queries to test the migration.

    For example, you can use the following query to find the number of user-defined tables.

    ```
    select count(*)
    from pg_table_def
    where schemaname != 'pg_catalog';
    ```

    The following query returns a list of all user-defined tables and the number of rows in each table.

    ```
    select "table", tbl_rows
    from svv_table_info;
    ```

3. Choose a good time for your migration. To find a time when cluster usage is lowest, monitor cluster metrics such as CPU utilization and number of database connections. For more information, see **Viewing cluster performance data** (p. 467).
4. Drop unused tables.
To create a list of tables and the number of the times each table has been queried, run the following query:

```sql
select database,
schema,
table_id,
"table",
round(size::float/(1024*1024)::float,2) as size,
sortkey1,
nvl(s.num_qs,0) num_qs
from svv_table_info t
left join (select tbl,
perm_table_name,
count(distinct query) num_qs
from stl_scan s
where s.userid > 1
and s.perm_table_name not in ('internal worktable','s3')
group by tbl,
perm_table_name) s on s.tbl = t.table_id
where t."schema" not in ('pg_internal');
```

5. Launch a new, encrypted cluster.

Use the same port number for the target cluster as for the source cluster. For more information about launching an encrypted cluster, see Amazon Redshift database encryption (p. 323).

6. Set up the unload and load process.

You can use the Amazon Redshift Unload/Copy Utility to help you to migrate data between clusters. The utility exports data from the source cluster to a location on Amazon S3. The data is encrypted with AWS KMS. The utility then automatically imports the data into the target. Optionally, you can use the utility to clean up Amazon S3 after migration is complete.

7. Run a test to verify your process and estimate how long write operations must be suspended.

During the unload and load operations, maintain data consistency by suspending data loads and other write operations. Using one of your largest tables, run through the unload and load process to help you estimate timing.

8. Create database objects, such as schemas, views, and tables. To help you generate the necessary data definition language (DDL) statements, you can use the scripts in AdminViews in the AWS GitHub repository.

To migrate your cluster

1. Stop all ETL processes on the source cluster.

   To confirm that there are no write operations in process, use the Amazon Redshift Management Console to monitor write IOPS. For more information, see Viewing cluster performance data (p. 467).

2. Run the validation queries you identified earlier to collect information about the unencrypted source cluster before migration.

3. (Optional) Create one workload management (WLM) queue to use the maximum available resources in both the source and target cluster. For example, create a queue named data_migrate and configure the queue with memory of 95 percent and concurrency of 4. For more information, see Routing Queries to Queues Based on User Groups and Query Groups in the Amazon Redshift Database Developer Guide.

4. Using the data_migrate queue, run the UnloadCopyUtility.

   Monitor the UNLOAD and COPY process using the Amazon Redshift Console.
5. Run the validation queries again and verify that the results match the results from the source cluster.
6. Rename your source and target clusters to swap the endpoints. To avoid disruption, perform this operation outside of business hours.
7. Verify that you can connect to the target cluster using all of your SQL clients, such as ETL and reporting tools.
8. Shut down the unencrypted source cluster.

Configuring database encryption using the console

You can use the Amazon Redshift console to configure Amazon Redshift to use an HSM and to rotate encryption keys. For information about how to create clusters using AWS KMS encryption keys, see Creating a cluster (p. 71) and Managing clusters using the AWS CLI and Amazon Redshift API (p. 76).

To modify database encryption on a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to move snapshots for.
3. For Actions, choose Modify to display the configuration page.
4. In the Database configuration section, choose a setting for Encryption, then choose Modify cluster.

Rotating encryption keys using the Amazon Redshift console

You can use the following procedure to rotate encryption keys by using the Amazon Redshift console.

To rotate the encryption keys for a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to update encryption keys.
3. For Actions, choose Rotate encryption to display the Rotate encryption keys page.

Configuring database encryption using the Amazon Redshift API and AWS CLI

Use the Amazon Redshift API and AWS Command Line Interface (AWS CLI) to configure encryption key options for Amazon Redshift databases. For more information about database encryption, see Amazon Redshift database encryption (p. 323).

Configuring Amazon Redshift to use AWS KMS encryption keys using the Amazon Redshift API and AWS CLI

You can use the following Amazon Redshift API actions to configure Amazon Redshift to use AWS KMS encryption keys.

- CreateCluster
- CreateSnapshotCopyGrant
- DescribeSnapshotCopyGrants
- DeleteSnapshotCopyGrant
- DisableSnapshotCopy
• **EnableSnapshotCopy**

You can use the following Amazon Redshift CLI operations to configure Amazon Redshift to use AWS KMS encryption keys.

• `create-cluster`
• `create-snapshot-copy-grant`
• `describe-snapshot-copy-grants`
• `delete-snapshot-copy-grant`
• `disable-snapshot-copy`
• `enable-snapshot-copy`

**Configuring Amazon Redshift to use an HSM using the Amazon Redshift API and AWS CLI**

You can use the following Amazon Redshift API actions to manage hardware security modules.

• `CreateHsmClientCertificate`
• `CreateHsmConfiguration`
• `DeleteHsmClientCertificate`
• `DeleteHsmConfiguration`
• `DescribeHsmClientCertificates`
• `DescribeHsmConfigurations`

You can use the following AWS CLI operations to manage hardware security modules.

• `create-hsm-client-certificate`
• `create-hsm-configuration`
• `delete-hsm-client-certificate`
• `delete-hsm-configuration`
• `describe-hsm-client-certificates`
• `describe-hsm-configurations`

**Rotating encryption keys using the Amazon Redshift API and AWS CLI**

You can use the following Amazon Redshift API actions to rotate encryption keys.

• `RotateEncryptionKey`

You can use the following AWS CLI operations to rotate encryption keys.

• `rotate-encryption-key`

**Encryption in transit**

You can configure your environment to protect the confidentiality and integrity data in transit.

Encryption of data in transit between an Amazon Redshift cluster and SQL clients over JDBC/ODBC:

• You can connect to Amazon Redshift clusters from SQL client tools over Java Database Connectivity (JDBC) and Open Database Connectivity (ODBC) connections.
Data encryption

• Amazon Redshift supports Secure Sockets Layer (SSL) connections to encrypt data and server certificates to validate the server certificate that the client connects to. The client connects to the leader node of an Amazon Redshift cluster. For more information, see Configuring security options for connections (p. 193).

• To support SSL connections, Amazon Redshift creates and installs AWS Certificate Manager (ACM) issued certificates on each cluster. For more information, see Transitioning to ACM certificates for SSL connections (p. 196).

• To protect your data in transit within the AWS Cloud, Amazon Redshift uses hardware accelerated SSL to communicate with Amazon S3 or Amazon DynamoDB for COPY, UNLOAD, backup, and restore operations.

Encryption of data in transit between an Amazon Redshift cluster and Amazon S3 or DynamoDB:

• Amazon Redshift uses hardware accelerated SSL to communicate with Amazon S3 or DynamoDB for COPY, UNLOAD, backup, and restore operations.

• Redshift Spectrum supports the Amazon S3 server-side encryption (SSE) using your account's default key managed by the AWS Key Management Service (KMS).

• Encrypt Amazon Redshift loads with Amazon S3 and AWS KMS. For more information, see Encrypt Your Amazon Redshift Loads with Amazon S3 and AWS KMS.

Encryption and signing of data in transit between AWS CLI, SDK, or API clients and Amazon Redshift endpoints:

• Amazon Redshift provides HTTPS endpoints for encrypting data in transit.

• To protect the integrity of API requests to Amazon Redshift, API calls must be signed by the caller. Calls are signed by an X.509 certificate or the customer's AWS secret access key according to the Signature Version 4 Signing Process (Sigv4). For more information, see Signature Version 4 Signing Process in the AWS General Reference.

• Use the AWS CLI or one of the AWS SDKs to make requests to AWS. These tools automatically sign the requests for you with the access key that you specify when you configure the tools.

Encryption of data in transit between Amazon Redshift clusters and AQUA:

• Data is transmitted between AQUA and Amazon Redshift clusters over a TLS-encrypted channel. This channel is signed according to the Signature Version 4 Signing Process (Sigv4). For more information, see Signature Version 4 Signing Process in the AWS General Reference.

Encryption of data in transit between Amazon Redshift clusters and Amazon Redshift query editor v2

• Data is transmitted between query editor v2 and Amazon Redshift clusters over a TLS-encrypted channel.

**Key management**

You can configure your environment to protect data with keys:

• Amazon Redshift automatically integrates with AWS Key Management Service (AWS KMS) for key management. AWS KMS uses envelope encryption. For more information, see Envelope Encryption.

• When encryption keys are managed in AWS KMS, Amazon Redshift uses a four-tier, key-based architecture for encryption. The architecture consists of randomly generated AES-256 data encryption keys, a database key, a cluster key, and a root key. For more information, see How Amazon Redshift Uses AWS KMS.
• You can create your own customer managed key in AWS KMS. For more information, see Creating Keys.
• You can also import your own key material for new AWS KMS keys. For more information, see Importing Key Material in AWS Key Management Service (AWS KMS).
• Amazon Redshift supports management of encryption keys in external hardware security modules (HSMs). The HSM can be on-premises or can be AWS CloudHSM. When you use an HSM, you must use client and server certificates to configure a trusted connection between Amazon Redshift and your HSM. Amazon Redshift supports only AWS CloudHSM Classic for key management. For more information, see Encryption for Amazon Redshift using hardware security modules (p. 326). For information about AWS CloudHSM, see What is AWS CloudHSM?
• You can rotate encryption keys for encrypted clusters. For more information, see Encryption key rotation in Amazon Redshift (p. 327).

**Data tokenization**

*Tokenization* is the process of replacing actual values with opaque values for data security purposes. Security-sensitive applications use tokenization to replace sensitive data such as personally identifiable information (PII) or protected health information (PHI) with tokens to reduce the security risks.

*Detokenization* reverses tokens with actual values for authorized users with appropriate security policies.

For integration with third-party tokenization services, you can use Amazon Redshift user-defined functions (UDFs) that you create using AWS Lambda. For more information, see Lambda user-defined functions in the *Amazon Redshift Database Developer Guide*. For example, see Protegrity.

Amazon Redshift sends tokenization requests to a tokenization server accessed through a REST API or predefined endpoint. Two or more complimentary Lambda functions process the tokenization and detokenization requests. For this processing, you can use Lambda functions provided by a third-party tokenization provider. You can also use Lambda functions that you register as Lambda UDFs in Amazon Redshift.

For example, suppose that a query is submitted that invokes a tokenization or detokenization UDF on a column. The Amazon Redshift cluster spools the applicable rows of arguments and sends those rows in batches to the Lambda function in parallel. The data transfers between the Amazon Redshift compute nodes and Lambda in a separate, isolated network connection that’s not accessible to clients. The Lambda function passes the data to the tokenization server endpoint. The tokenization server tokenizes or detokenizes the data as necessary and returns it. The Lambda functions then transmit the results to the Amazon Redshift cluster for further processing, if necessary, and then return the query results.

**Internetwork traffic privacy**

To route traffic between Amazon Redshift and clients and applications on a corporate network:

• Set up a private connection between your virtual private cloud (VPC) and your corporate network. Set up either an IPsec VPN connection over the internet or a private physical connection using AWS Direct Connect connection. AWS Direct Connect enables you to establish a private virtual interface from your on-premises network directly to your Amazon VPC, providing you with a private, high-bandwidth network connection between your network and your VPC. With multiple virtual interfaces, you can even establish private connectivity to multiple VPCs while maintaining network isolation. For more information, see What is AWS Site-to-Site VPN? and What is AWS Direct Connect?

To route traffic between an Amazon Redshift cluster in a VPC and Amazon S3 buckets in the same AWS Region:

• Set up an Amazon S3 private VPC endpoint to privately access Amazon S3 data from an ETL load or unload. For more information, see Endpoints for Amazon S3.
Identity and access management in Amazon Redshift

Access to Amazon Redshift requires credentials that AWS can use to authenticate your requests. Those credentials must have permissions to access AWS resources, such as an Amazon Redshift cluster. The following sections provide details on how you can use AWS Identity and Access Management (IAM) and Amazon Redshift to help secure your resources by controlling who can access them:

- Authentication (p. 334)
- Access control (p. 335)

Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create a cluster in Amazon Redshift). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Management Console, AWS Discussion Forums, or the AWS Support Center.

In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several SDKs or by using the AWS Command Line Interface (CLI). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don't use AWS tools, you must sign the request yourself. Amazon Redshift supports Signature Version 4, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 signing process in the AWS General Reference.

- **IAM role** – An IAM role is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated users and roles in the IAM User Guide.
Access control

You can have valid credentials to authenticate your requests, but unless you have permissions you cannot create or access Amazon Redshift resources. For example, you must have permissions to create an Amazon Redshift cluster, create a snapshot, add an event subscription, and so on.

The following sections describe how to manage permissions for Amazon Redshift. We recommend that you read the overview first.

- Overview of managing access permissions to your Amazon Redshift resources (p. 335)
- Using identity-based policies (IAM policies) for Amazon Redshift (p. 340)

Overview of managing access permissions to your Amazon Redshift resources

Every AWS resource is owned by an AWS account, and permissions to create or access the resources are governed by permissions policies. An account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles), and some services (such as AWS Lambda) also support attaching permissions policies to resources.

**Note**

An account administrator (or administrator user) is a user with administrator privileges. For more information, see IAM best practices in the IAM User Guide.

When granting permissions, you decide who is getting the permissions, which resources they get permissions for, and the specific actions that you want to allow on those resources.

Amazon Redshift resources and operations

In Amazon Redshift, the primary resource is a cluster. Amazon Redshift supports other resources that can be used with the primary resource such as snapshots, parameter groups, and event subscriptions. These are referred to as subresources.

These resources and subresources have unique Amazon Resource Names (ARNs) associated with them as shown in the following table.

<table>
<thead>
<tr>
<th>Resource type</th>
<th>ARN format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource type</td>
<td>ARN format</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Cluster database user group</td>
<td>arn:aws:redshift:region:account-id:dbgroup:cluster-name/database-group-name</td>
</tr>
<tr>
<td>Cluster parameter group</td>
<td>arn:aws:redshift:region:account-id:parametergroup:parameter-group-name</td>
</tr>
<tr>
<td>Cluster security group</td>
<td>arn:aws:redshift:region:account-id:securitygroup:security-group-name</td>
</tr>
<tr>
<td>CIDR/IP address</td>
<td>arn:aws:redshift:region:account-id:securitygroupingress:security-group-name/cidrip/IP-range</td>
</tr>
<tr>
<td>EC2 security group</td>
<td>arn:aws:redshift:region:account-id:securitygroupingress:security-group-name/ec2securitygroup/owner/EC2-security-group-id</td>
</tr>
<tr>
<td>Event subscription</td>
<td>arn:aws:redshift:region:account-id:eventsubscription:event-subscription-name</td>
</tr>
<tr>
<td>Parameter group</td>
<td>arn:aws:redshift:region:account-id:parametergroup:parameter-group-name</td>
</tr>
<tr>
<td>Redshift-managed VPC endpoint</td>
<td>arn:aws:redshift:region:account-id:managedvpcendpoint:endpoint-name</td>
</tr>
<tr>
<td>Snapshot copy grant</td>
<td>arn:aws:redshift:region:account-id:snapshotcopygrant:snapshot-copy-grant-name</td>
</tr>
<tr>
<td>Subnet group</td>
<td>arn:aws:redshift:region:account-id:subnetgroup:subnet-group-name</td>
</tr>
</tbody>
</table>

Amazon Redshift provides a set of operations to work with the Amazon Redshift resources. For a list of available operations, see Amazon Redshift API permissions reference (p. 366).

**Amazon Redshift query editor v2 resources**

These query editor v2 resources have unique Amazon Resource Names (ARNs) associated with them as shown in the following table.

<table>
<thead>
<tr>
<th>Resource type</th>
<th>ARN format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>arn:aws:sqlworkbench:region:account-id:connection/</td>
</tr>
</tbody>
</table>
Understanding resource ownership

A resource owner is the AWS account that created a resource. That is, the resource owner is the AWS account of the principal entity (the root account, an IAM user, or an IAM role) that authenticates the request that creates the resource. The following examples illustrate how this works:

- If you use the root account credentials of your AWS account to create a DB cluster, your AWS account is the owner of the Amazon Redshift resource.
- If you create an IAM user in your AWS account and grant permissions to create Amazon Redshift resources to that user, the user can create Amazon Redshift resources. However, your AWS account, to which the user belongs, owns the Amazon Redshift resources.
- If you create an IAM role in your AWS account with permissions to create Amazon Redshift resources, anyone who can assume the role can create Amazon Redshift resources. Your AWS account, to which the role belongs, owns the Amazon Redshift resources.

Managing access to resources

A permissions policy describes who has access to what. The following section explains the available options for creating permissions policies.

Note
This section discusses using IAM in the context of Amazon Redshift. It doesn't provide detailed information about the IAM service. For complete IAM documentation, see What is IAM? in the IAM User Guide. For information about IAM policy syntax and descriptions, see AWS IAM policy reference in the IAM User Guide.

Policies attached to an IAM identity are referred to as identity-based policies (IAM policies) and policies attached to a resource are referred to as resource-based policies. Amazon Redshift supports only identity-based policies (IAM policies).

Identity-based policies (IAM policies)

You can attach policies to IAM identities. For example, you can do the following:

- **Attach a permissions policy to a user or a group in your account** – An account administrator can use a permissions policy that is associated with a particular user. Such a policy grants permissions for that user to create an Amazon Redshift resource, such as a cluster.

- **Attach a permissions policy to a role (grant cross-account permissions)** – You can attach an identity-based permissions policy to an IAM role to grant cross-account permissions. For example, the administrator in Account A can create a role to grant cross-account permissions to another AWS account (for example, Account B) or an AWS service as follows:
  1. Account A administrator creates an IAM role and attaches a permissions policy to the role that grants permissions on resources in Account A.
  2. Account A administrator attaches a trust policy to the role identifying Account B as the principal who can assume the role.
  3. Account B administrator can then delegate permissions to assume the role to any users in Account B. Doing this allows users in Account B to create or access resources in Account A. The principal
in the trust policy can also be an AWS service principal if you want to grant an AWS service permissions to assume the role.

For more information about using IAM to delegate permissions, see Access management in the IAM User Guide.

The following is an example policy that allows a user to create, delete, modify, and reboot Amazon Redshift clusters for your AWS account.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":"AllowManageClusters",
      "Effect":"Allow",
      "Action": [
        "redshift:CreateCluster",
        "redshift:DeleteCluster",
        "redshift:ModifyCluster",
        "redshift:RebootCluster"
      ],
      "Resource":"*"
    }
  ]
}
```

For more information about using identity-based policies with Amazon Redshift, see Using identity-based policies (IAM policies) for Amazon Redshift (p. 340). For more information about users, groups, roles, and permissions, see Identities (users, groups, and roles) in the IAM User Guide.

Resource-based policies

Other services, such as Amazon S3, also support resource-based permissions policies. For example, you can attach a policy to an S3 bucket to manage access permissions to that bucket. Amazon Redshift doesn't support resource-based policies.

Specifying policy elements: Actions, effects, resources, and principals

For each Amazon Redshift resource (see Amazon Redshift resources and operations (p. 335)), the service defines a set of API operations (see Actions). To grant permissions for these API operations, Amazon Redshift defines a set of actions that you can specify in a policy. Performing an API operation can require permissions for more than one action.

The following are the basic policy elements:

- **Resource** – In a policy, you use an Amazon Resource Name (ARN) to identify the resource to which the policy applies. For more information, see Amazon Redshift resources and operations (p. 335).
- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, the `redshift:DescribeClusters` permission allows the user permissions to perform the Amazon Redshift DescribeClusters operation.
- **Effect** – You specify the effect when the user requests the specific action—this can be either allow or deny. If you don’t explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user that the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity
that you want to receive permissions (applies to resource-based policies only). Amazon Redshift
doesn't support resource-based policies.

To learn more about IAM policy syntax and descriptions, see AWS IAM policy reference in the IAM User
Guide.

For a table showing all of the Amazon Redshift API actions and the resources that they apply to, see
Amazon Redshift API permissions reference (p. 366).

Specifying conditions in a policy

When you grant permissions, you can use the access policy language to specify the conditions when a
policy should take effect. For example, you might want a policy to be applied only after a specific date.
For more information about specifying conditions in an access policy language, see IAM JSON policy
elements: Condition in the IAM User Guide.

To identify conditions where a permissions policy applies, include a Condition element in your IAM
permissions policy. For example, you can create a policy that permits a user to create a cluster using the
redshift:CreateCluster action, and you can add a Condition element to restrict that user to only
create the cluster in a specific region. For details, see Using IAM policy conditions for fine-grained access
control (p. 339). For a list showing all of condition key values and the Amazon Redshift actions and
resources that they apply to, see Amazon Redshift API permissions reference (p. 366).

Using IAM policy conditions for fine-grained access control

In Amazon Redshift, you can use condition keys to restrict access to resources based on the tags for those
resources. The following are common Amazon Redshift condition keys.

<table>
<thead>
<tr>
<th>Condition key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aws:RequestTag</td>
<td>Requires users to include a tag key (name) and value whenever they create a</td>
</tr>
<tr>
<td></td>
<td>resource. For more information, see aws:RequestTag in the IAM User Guide.</td>
</tr>
<tr>
<td>aws:ResourceTag</td>
<td>Restricts user access to resources based on specific tag keys and values. For</td>
</tr>
<tr>
<td></td>
<td>more information, see aws:ResourceTag in the IAM User Guide.</td>
</tr>
<tr>
<td>aws:TagKeys</td>
<td>Use this key to compare the tag keys in a request with the keys that you</td>
</tr>
<tr>
<td></td>
<td>specify in the policy. For more information, see aws:TagKeys in the IAM User</td>
</tr>
<tr>
<td></td>
<td>Guide.</td>
</tr>
</tbody>
</table>

For information on tags, see Tagging overview (p. 512).

For a list of the API actions that support the redshift:RequestTag and redshift:ResourceTag
condition keys, see Amazon Redshift API permissions reference (p. 366).

The following condition keys can be used with the Amazon Redshift GetClusterCredentials action.

<table>
<thead>
<tr>
<th>Condition key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>redshift:DurationSeconds</td>
<td><strong>Limits</strong> the number of seconds that can be specified for duration.</td>
</tr>
<tr>
<td>redshift:DbName</td>
<td>Restricts database names that can be specified.</td>
</tr>
<tr>
<td>redshift:DbUser</td>
<td>Restricts database user names that can be specified.</td>
</tr>
</tbody>
</table>
Example 1: Restricting access by using the aws:ResourceTag condition key

Use the following IAM policy to let a user modify an Amazon Redshift cluster only for a specific AWS account in the us-west-2 region with a tag named environment with a tag value of test.

```json
{  
  "Version": "2012-10-17",
  "Statement": {  
    "Sid": "AllowModifyTestCluster",
    "Effect": "Allow",
    "Action": "redshift:ModifyCluster",
    "Condition": {  
      "StringEquals": {  
        "aws:ResourceTag/environment": "test"
      }
    }
  }
}
```

Example 2: Restricting access by using the aws:RequestTag condition key

Use the following IAM policy to let a user create an Amazon Redshift cluster only if the command to create the cluster includes a tag named usage and a tag value of production. The condition with aws:TagKeys and the ForAllValues modifier specifies that only the keys costcenter and usage can be specified in the request.

```json
{  
  "Version": "2012-10-17",
  "Statement": {  
    "Sid": "AllowCreateProductionCluster",
    "Effect": "Allow",
    "Action": [  
      "redshift:CreateCluster",
      "redshift:CreateTags"
    ],
    "Resource": "*",
    "Condition": {  
      "StringEquals": {  
        "aws:RequestTag/usage": "production"
      },
      "ForAllValues:StringEquals": {  
        "aws:TagKeys": [  
          "costcenter",
          "usage"
        ]
      }
    }
  }
}
```

Using identity-based policies (IAM policies) for Amazon Redshift

This topic provides examples of identity-based policies in which an account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles).

Important

We recommend that you first review the introductory topics that explain the basic concepts and options available for you to manage access to your Amazon Redshift resources. For
more information, see Overview of managing access permissions to your Amazon Redshift resources (p. 335).

The following shows an example of a permissions policy. The policy allows a user to create, delete, modify, and reboot all clusters, and then denies permission to delete or modify any clusters where the cluster identifier starts with production.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid":"AllowClusterManagement",
            "Action": [
                "redshift:CreateCluster",
                "redshift:DeleteCluster",
                "redshift:ModifyCluster",
                "redshift:RebootCluster"
            ],
            "Resource": ["*"],
            "Effect": "Allow"
        },
        {
            "Sid":"DenyDeleteModifyProtected",
            "Action": [
                "redshift:DeleteCluster",
                "redshift:ModifyCluster"
            ],
            "Effect": "Deny"
        }
    ]
}
```

The policy has two statements:

- The first statement grants permissions for a user to a user to create, delete, modify, and reboot clusters. The statement specifies a wildcard character (*) as the Resource value so that the policy applies to all Amazon Redshift resources owned by the root AWS account.
- The second statement denies permission to delete or modify a cluster. The statement specifies a cluster Amazon Resource Name (ARN) for the Resource value that includes a wildcard character (*). As a result, this statement applies to all Amazon Redshift clusters owned by the root AWS account where the cluster identifier begins with production.

**AWS-managed (predefined) policies for Amazon Redshift**

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. Managed policies grant necessary permissions for common use cases so you can avoid having to investigate what permissions are needed. For more information, see AWS managed policies in the IAM User Guide.

You can also create your own custom IAM policies to allow permissions for Amazon Redshift API operations and resources. You can attach these custom policies to the IAM users or groups that require those permissions.

The following sections describe AWS managed policies, which you can attach to users in your account, and are specific to Amazon Redshift.
**AmazonRedshiftReadOnlyAccess**
Grants read-only access to all Amazon Redshift resources for an AWS account.
You can find the AmazonRedshiftReadOnlyAccess policy on the IAM console.

**AmazonRedshiftFullAccess**
Grants full access to all Amazon Redshift resources for an AWS account.
You can find the AmazonRedshiftFullAccess policy on the IAM console.

**AmazonRedshiftQueryEditor**
Grants full access to the query editor on the Amazon Redshift console.
You can find the AmazonRedshiftQueryEditor policy on the IAM console.

**AmazonRedshiftDataFullAccess**
Grants full access to the Amazon Redshift Data API operations and resources for an AWS account.
You can find the AmazonRedshiftDataFullAccess policy on the IAM console.

**AmazonRedshiftQueryEditorV2FullAccess**
Grants full access to the Amazon Redshift query editor v2 operations and resources. This policy also grants access to other required services.
You can find the AmazonRedshiftQueryEditorV2FullAccess policy on the IAM console.

**AmazonRedshiftQueryEditorV2NoSharing**
Grants the ability to work with Amazon Redshift query editor v2 without sharing resources. This policy also grants access to other required services. The principal using this policy can't tag its resources (such as queries) to share them with other principals in the same AWS account.
You can find the AmazonRedshiftQueryEditorV2NoSharing policy on the IAM console.

**AmazonRedshiftQueryEditorV2ReadSharing**
Grants the ability to work with Amazon Redshift query editor v2 with limited sharing of resources. This policy also grants access to other required services. The principal using this policy can tag its resources (such as queries) to share them with other principals in the same AWS account. The granted principal can read the resources shared with its team but can't update them.
You can find the AmazonRedshiftQueryEditorV2ReadSharing policy on the IAM console.

**AmazonRedshiftQueryEditorV2ReadWriteSharing**
Grants the ability to work with Amazon Redshift query editor v2 with sharing of resources. This policy also grants access to other required services. The principal using this policy can tag its resources (such as queries) to share them with other principals in the same AWS account. The granted principal can read and update the resources shared with its team.
You can find the AmazonRedshiftQueryEditorV2ReadWriteSharing policy on the IAM console.
Amazon Redshift Cluster Management Guide
Using identity-based policies (IAM policies)

**AmazonRedshiftServiceLinkedRolePolicy**

You can't attach `AmazonRedshiftServiceLinkedRolePolicy` to your IAM entities. This policy is attached to a service-linked role that allows Amazon Redshift to access account resources. For more information, see Using service-linked roles for Amazon Redshift.

**AmazonRedshiftAllCommandsFullAccess**

Grants the ability to use the IAM role created from the Amazon Redshift console and set it as default for the cluster to run the COPY from Amazon S3, UNLOAD, CREATE EXTERNAL SCHEMA, CREATE EXTERNAL FUNCTION, and CREATE MODEL commands. The policy also grants permissions to run SELECT statements for related services, such as Amazon S3, CloudWatch Logs, Amazon SageMaker, or AWS Glue.

You can find the `AmazonRedshiftAllCommandsFullAccess` policy on the IAM console.

You can also create your own custom IAM policies to allow permissions for Amazon Redshift API operations and resources. You can attach these custom policies to the IAM users or groups that require those permissions.

**Amazon Redshift updates to AWS-managed policies**

View details about updates to AWS managed policies for Amazon Redshift since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Amazon Redshift Document history page.

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AmazonRedshiftQueryEditorV2FullAccess</code></td>
<td>Update to an existing policy</td>
<td>February 22, 2022</td>
</tr>
<tr>
<td>Permissions to the action <code>sqlworkbench:ListTaggedResources</code>. It is scoped specifically to Amazon Redshift query editor v2 resources. This policy update gives the right to call <code>tag:GetResources</code> only through query editor v2.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AmazonRedshiftQueryEditorV2NoSharing</code></td>
<td>Update to an existing policy</td>
<td>February 22, 2022</td>
</tr>
<tr>
<td>Permissions to the action <code>sqlworkbench:ListTaggedResources</code>. It is scoped specifically to Amazon Redshift query editor v2 resources. This policy update gives the right to call <code>tag:GetResources</code> only through query editor v2.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AmazonRedshiftQueryEditorV2ReadSharing</code></td>
<td>Update to an existing policy</td>
<td>February 22, 2022</td>
</tr>
<tr>
<td>Permissions to the action <code>sqlworkbench:ListTaggedResources</code>. It is scoped specifically to Amazon Redshift query editor v2 resources. This policy update gives the right to call <code>tag:GetResources</code> only through query editor v2.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing (p. 342)</td>
<td>– Update to an existing policy</td>
<td>February 22, 2022</td>
</tr>
<tr>
<td></td>
<td>Permissions to the action sqlworkbench:ListTaggedResources. It is scoped specifically to Amazon Redshift query editor v2 resources. This policy update gives the right to call tag:GetResources only through query editor v2.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing (p. 342)</td>
<td>– Update to an existing policy</td>
<td>February 22, 2022</td>
</tr>
<tr>
<td></td>
<td>Permission for the action sqlworkbench:AssociateQueryWithTab is added to the managed policy. Adding it allows customers to create editor tabs linked to a query that is shared with them.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftServiceLinkedRolePolicy (p. 343)</td>
<td>– Update to an existing policy</td>
<td>November 22, 2021</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift added permissions for new actions to allow management of Amazon Redshift network and VPC resources.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftAllCommandsFullAccess (p. 342)</td>
<td>– New policy</td>
<td>November 18, 2021</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift added a new policy to allow using the IAM role created from the Amazon Redshift console and set it as default for the cluster to run the COPY from Amazon S3, UNLOAD, CREATE EXTERNAL SCHEMA, CREATE EXTERNAL FUNCTION, CREATE MODEL, or CREATE LIBRARY commands.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftServiceLinkedRolePolicy (p. 343)</td>
<td>– Update to an existing policy</td>
<td>November 15, 2021</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift added permissions for new actions to allow management of Amazon Redshift CloudWatch log groups and log streams, including audit-log export.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftFullAccess (p. 342)</td>
<td>– Update to an existing policy</td>
<td>October 07, 2021</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift added new permissions to allow model explainability, DynamoDB, Redshift Spectrum, and Amazon RDS federation.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftFullAccess (p. 342)</td>
<td>– New policy</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift added a new policy to allow full access to Amazon Redshift query editor v2.</td>
<td></td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2NoSharing (p. 342)</td>
<td>– New policy</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift added a new policy to allow using Amazon Redshift query editor v2 without sharing resources.</td>
<td></td>
</tr>
</tbody>
</table>
Permissions required to use Redshift Spectrum

Amazon Redshift Spectrum requires permissions to other AWS services to access resources. For details about permissions in IAM policies for Redshift Spectrum, see IAM policies for Amazon Redshift Spectrum in the Amazon Redshift Database Developer Guide.

Permissions required to use the Amazon Redshift console

For a user to work with the Amazon Redshift console, that user must have a minimum set of permissions that allows the user to describe the Amazon Redshift resources for their AWS account. These permissions must also allow the user to describe other related information, including Amazon EC2 security, Amazon CloudWatch, Amazon SNS, and network information.

If you create an IAM policy that is more restrictive than the minimum required permissions, the console doesn’t function as intended for users with that IAM policy. To ensure that those users can still use the Amazon Redshift console, also attach the AmazonRedshiftReadOnlyAccess managed policy to the user. How to do this is described in AWS-managed (predefined) policies for Amazon Redshift (p. 341).

For information to give a user access to the query editor on the Amazon Redshift console, see Permissions required to use the Amazon Redshift console query editor (p. 346).

You don’t need to allow minimum console permissions for users that are making calls only to the AWS CLI or the Amazon Redshift API.
Permissions required to use the Amazon Redshift console query editor

For a user to work with the Amazon Redshift query editor, that user must have a minimum set of permissions to Amazon Redshift and Amazon Redshift Data API operations. To connect to a database using a secret, you must also have Secrets Manager permissions.

To give a user access to the query editor on the Amazon Redshift console, attach the `AmazonRedshiftQueryEditor` and `AmazonRedshiftReadOnlyAccess` AWS-managed policies. The `AmazonRedshiftQueryEditor` policy allows the user permission to retrieve the results of only their own SQL statements. That is, statements submitted by the same `aws:userid` as shown in this section of the `AmazonRedshiftQueryEditor` AWS-managed policy.

```
{
  "Sid": "DataAPIIAMSessionPermissionsRestriction",
  "Action": [
    "redshift-data:GetStatementResult",
    "redshift-data:CancelStatement",
    "redshift-data:DescribeStatement",
    "redshift-data:ListStatements"
  ],
  "Effect": "Allow",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "redshift-data:statement-owner-iam-userid": "${aws:userid}"
    }
  }
}
```

To allow a user to retrieve the results of SQL statements of others in the same IAM role, create your own policy without the condition to limit access to the current user. Also limit access to change a policy to an administrator.

Permissions required to use the query editor v2

For a user to work with the Amazon Redshift query editor v2, that user must have a minimum set of permissions to Amazon Redshift, the query editor v2 operations, and other AWS services such as AWS Key Management Service, AWS Secrets Manager, and tagging service.

To give a user full access to the query editor v2, attach the `AmazonRedshiftQueryEditorV2FullAccess` AWS-managed policy. The `AmazonRedshiftQueryEditorV2FullAccess` policy allows the user permission to share query editor v2 resources, such as queries, with others in the same team. For details about how access to query editor v2 resources are controlled, see the definition of the specific managed policy for query editor v2 in the IAM console.

Some Amazon Redshift query editor v2 AWS-managed policies use AWS tags within conditions to scope access to resources. Within query editor v2, sharing queries is based on the tag key and value "aws:ResourceTag/sqlworkbench-team": "${aws:PrincipalTag/sqlworkbench-team}" in the IAM policy attached to principal (either the IAM user or IAM role). Principals in the same AWS account with the same tag value (for example, accounting-team), are on the same team in query editor v2. You can only be associated with one team at a time. A user with administrative permissions can set up teams in the IAM console by giving all team members the same value for the sqlworkbench-team tag. If the tag value of the sqlworkbench-team is changed for an IAM user or an IAM role, there might be a delay until the change is reflected in shared resources. If the tag value of a resource (such as a query) is changed, again there might be a delay until the change is reflected. Team members must also have the `tag:GetResources` permission to share.
Example: To add the `accounting-team` tag for an IAM role

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane of the console, choose Roles and then choose the name of the role that you want to edit.
3. Choose the Tags tab and then choose Add tags.
4. Add the tag key `sqlworkbench-team` and the value `accounting-team`.
5. Choose Save changes.

Now when an IAM principal (with this IAM role attached) shares a query with the team, other principals with the same `accounting-team` tag value can view the query.

For more information on how to attach a tag to a principal, including IAM roles and IAM users, see Tagging IAM resources in the IAM User Guide.

You can also set up teams at the session level using an Identity Provider (IdP). This allows multiple users using the same IAM role to have different teams. The IAM role trust policy must allow the `sts:TagSession` operation. For more information, see Permissions required to add session tags in the IAM User Guide. Add the principal tag attribute to the SAML assertion provided by your IdP.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:sqlworkbench-team">
  <AttributeValue>accounting-team</AttributeValue>
</Attribute>
```

Follow the instructions for your Identity provider (IdP) to populate the SAML attribute with the content coming from your directory. For more information about Identity providers (IdPs) and Amazon Redshift, see Using IAM authentication to generate database user credentials (p. 370) and Identity providers and federation in the IAM User Guide.

Permissions required to use the Amazon Redshift scheduler

When you use the Amazon Redshift scheduler, you set up an IAM role with a trust relationship to the Amazon Redshift scheduler (`scheduler.redshift.amazonaws.com`) to allow the scheduler to assume permissions on your behalf. You also attach a policy (permissions) to the role for the Amazon Redshift API operations that you want to schedule.

The following example shows the policy document in JSON format to set up a trust relationship with the Amazon Redshift scheduler and Amazon Redshift.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "scheduler.redshift.amazonaws.com",
          "redshift.amazonaws.com"
        ],
      "Action": "sts:AssumeRole"
    }
  ]
}
```

For more information about trust entities, see Creating a role to delegate permissions to an AWS service in the IAM User Guide.
You also must add permission for the Amazon Redshift operations you want to schedule.

For the scheduler to use the `ResizeCluster` operation, add a permission that is similar to the following to your IAM policy. Depending on your environment, you might want to make the policy more restrictive.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "redshift:ResizeCluster",
         "Resource": "*"
      }
   ]
}
```

For the steps to create a role for the Amazon Redshift scheduler, see Creating a role for an AWS service (console) in the IAM User Guide. Make these choices when you create a role in the IAM console:

- **For Choose the service that will use this role**: Choose Redshift.
- **For Select your use case**: Choose Redshift - Scheduler.
- Create or attach a policy to the role that allows an Amazon Redshift operation to be scheduled. Choose Create policy or modify the role to attach a policy. Enter the JSON policy for the operation that is to be scheduled.
- After you create the role, edit the Trust Relationship of the IAM role to include the service `redshift.amazonaws.com`.

The IAM role you create has trusted entities of `scheduler.redshift.amazonaws.com` and `redshift.amazonaws.com`. It also has an attached policy that allows a supported Amazon Redshift API action, such as, "redshift:ResizeCluster".

### Permissions required to use the Amazon EventBridge scheduler

When you use the Amazon EventBridge scheduler, you set up an IAM role with a trust relationship to the EventBridge scheduler (`events.amazonaws.com`) to allow the scheduler to assume permissions on your behalf. You also attach a policy (permissions) to the role for the Amazon Redshift Data API operations that you want to schedule and a policy for Amazon EventBridge operations.

You use the EventBridge scheduler when you create scheduled queries with the Amazon Redshift query editor on the console.

You can create an IAM role to run scheduled queries on the IAM console. In this IAM role, attach `AmazonEventBridgeFullAccess` and `AmazonRedshiftDataFullAccess`.

The following example shows the policy document in JSON format to set up a trust relationship with the EventBridge scheduler.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "Service": ["events.amazonaws.com", "redshift.amazonaws.com"]
         },
         "Action": "sts:AssumeRole"
      }
   ]
}
```
For more information about trust entities, see Creating a role to delegate permissions to an AWS service in the IAM User Guide.

For the steps to create a role for the EventBridge scheduler, see Creating a role for an AWS service (console) in the IAM User Guide. Make these choices when you create a role in the IAM console:

- For **Choose the service that will use this role**: Choose CloudWatch Events.
- For **Select your use case**: Choose CloudWatch Events.
- Attach the following permission policies: AmazonEventBridgeFullAccess and AmazonRedshiftDataFullAccess.

The IAM role that you create has a trusted entity of events.amazonaws.com. It also has an attached policy that allows supported Amazon Redshift Data API actions, such as, "redshift-data:*".

**Permissions required to use Amazon Redshift machine learning (ML) with Amazon SageMaker**

Following, you can find a description of the permissions required to use Amazon Redshift machine learning (ML) with Amazon SageMaker for different use cases.

For your users to use Amazon Redshift ML with Amazon SageMaker, create an IAM role with a more restrictive policy than the default. You can use the policy following. You can also modify this policy to meet your needs.

The following policy shows the permissions required to run SageMaker Autopilot with model explainability from Amazon Redshift.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sagemaker:CreateTrainingJob",
                "sagemaker:CreateAutoMLJob",
                "sagemaker:CreateCompilationJob",
                "sagemaker:CreateEndpoint",
                "sagemaker:DescribeAutoMLJob",
                "sagemaker:DescribeTrainingJob",
                "sagemaker:DescribeCompilationJob",
                "sagemaker:DescribeEndpoint",
                "sagemaker:DescribeTransformJob",
                "sagemaker:ListCandidatesForAutoMLJob",
                "sagemaker:StopAutoMLJob",
                "sagemaker:StopCompilationJob",
                "sagemaker:StopTrainingJob",
                "sagemaker:DescribeAutoMLJob",
                "sagemaker:InvokeEndpoint",
                "sagemaker:StopProcessingJob",
                "sagemaker:CreateModel",
                "sagemaker:CreateProcessingJob"
            ],
            "Resource": [
                "arn:aws:sagemaker:***:model/redshift***",
                "arn:aws:sagemaker:***:training-job/redshift***",
            ]
        }
    ]
}
```
"arn:aws:sagemaker:*:*:automl-job/*redshift*",
"arn:aws:sagemaker:*:*:compilation-job/*redshift*",
"arn:aws:sagemaker:*:*:processing-job/*redshift*",
"arn:aws:sagemaker:*:*:transform-job/*redshift*",
"arn:aws:sagemaker:*:*:endpoint/*redshift*"
]
},
{
  "Effect": "Allow",
  "Action": [
    "logs:CreateLogGroup",
    "logs:CreateLogStream",
    "logs:DescribeLogStreams",
    "logs:PutLogEvents"
  ],
  "Resource": [
    "arn:aws:logs:*:*:log-group:/aws/sagemaker/Endpoints/*redshift*",
    "arn:aws:logs:*:*:log-group:/aws/sagemaker/ProcessingJobs/*redshift*",
    "arn:aws:logs:*:*:log-group:/aws/sagemaker/TrainingJobs/*redshift*",
    "arn:aws:logs:*:*:log-group:/aws/sagemaker/TransformJobs/*redshift*"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "cloudwatch:PutMetricData"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "cloudwatch:namespace": [
        "SageMaker",
        "/aws/sagemaker/Endpoints",
        "/aws/sagemaker/ProcessingJobs",
        "/aws/sagemaker/TrainingJobs",
        "/aws/sagemaker/TransformJobs"
      ]
    }
  }
}
},
{
  "Effect": "Allow",
  "Action": [
    "ecr:BatchCheckLayerAvailability",
    "ecr:BatchGetImage",
    "ecr:GetAuthorizationToken",
    "ecr:GetDownloadUrlForLayer"
  ],
  "Resource": "*"
},
{
  "Effect": "Allow",
  "Action": [
    "s3:GetObject",
    "s3:GetBucketAcl",
    "s3:GetBucketCors",
    "s3:GetEncryptionConfiguration",
    "s3:GetBucketLocation",
    "s3:ListBucket",
    "s3:ListAllMyBuckets",
    "s3:ListMultipartUploadParts",
    "s3:ListBucketMultipartUploads",
    "s3:PutObject",
    "s3:PutBucketAcl",
    "s3:PutBucketCors",
    "s3:DeleteObject"
  ]
}
The following policy shows the full minimal permissions to allow access to Amazon DynamoDB, Redshift Spectrum and Amazon RDS federation.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sagemaker:CreateTrainingJob",
                "sagemaker:DescribeTrainingJob",
                "sagemaker:ListTrainingJobs",
                "sagemaker:StartTrainingJob",
                "sagemaker:StopTrainingJob",
                "sagemaker:UpdateTrainingJob",
                "sagemaker:TagResource",
                "sagemaker:UntagResource"
            ],
            "Resource": "arn:aws:sagemaker:*:resource",
            "Condition": {
                "StringEquals": {
                    "sagemaker:SourceArn": "arn:aws:sagemaker:*:resource"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "redshift.amazonaws.com",
                        "sagemaker.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```
Using identity-based policies (IAM policies)

```
"sagemaker:CreateAutoMLJob",
"sagemaker:CreateCompilationJob",
"sagemaker:CreateEndpoint",
"sagemaker:DescribeAutoMLJob",
"sagemaker:DescribeCompilationJob",
"sagemaker:DescribeTrainingJob",
"sagemaker:DescribeProcessingJob",
"sagemaker:DescribeTransformJob",
"sagemaker:ListCandidatesForAutoMLJob",
"sagemaker:StopAutoMLJob",
"sagemaker:StopCompilationJob",
"sagemaker:StopTrainingJob",
"sagemaker:DescribeEndpoint",
"sagemaker:InvokeEndpoint",
"sagemaker:StopProcessingJob",
"sagemaker:CreateModel",
"sagemaker:CreateProcessingJob"
],
"Resource": [ 
  "arn:aws:sagemaker:*:*:model/*redshift*",
  "arn:aws:sagemaker:*:*:training-job/*redshift*",
  "arn:aws:sagemaker:*:*:automl-job/*redshift*",
  "arn:aws:sagemaker:*:*:compilation-job/*redshift*",
  "arn:aws:sagemaker:*:*:processing-job/*redshift*",
  "arn:aws:sagemaker:*:*:transform-job/*redshift*",
  "arn:aws:sagemaker:*:*:endpoint/*redshift*"
]
}
{
"Effect": "Allow",
"Action": [ 
  "logs:CreateLogGroup",
  "logs:CreateLogStream",
  "logs:DescribeLogStreams",
  "logs:PutLogEvents"
],
"Resource": [ 
  "arn:aws:logs:*:*:log-group:/aws/sagemaker/Endpoints/*redshift*",
  "arn:aws:logs:*:*:log-group:/aws/sagemaker/ProcessingJobs/*redshift*",
  "arn:aws:logs:*:*:log-group:/aws/sagemaker/TrainingJobs/*redshift*",
  "arn:aws:logs:*:*:log-group:/aws/sagemaker/TransformJobs/*redshift*"
]
}
{
"Effect": "Allow",
"Action": [ 
  "cloudwatch:PutMetricData"
],
"Resource": "*",
"Condition": { 
  "StringEquals": { 
    "cloudwatch:namespace": [ 
      "SageMaker",
      "/aws/sagemaker/Endpoints",
      "/aws/sagemaker/ProcessingJobs",
      "/aws/sagemaker/TrainingJobs",
      "/aws/sagemaker/TransformJobs"
    ]
  }
}
}
{
"Effect": "Allow",
"Action": [ 
  "ecr:BatchCheckLayerAvailability",
  "ecr:BatchGetImage",
```
Using identity-based policies (IAM policies)

```json
{
    "Effect": "Allow",
    "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:GetDownloadUrlForLayer"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "s3:GetObject",
        "s3:GetBucketAcl",
        "s3:GetBucketCors",
        "s3:GetEncryptionConfiguration",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:ListAllMyBuckets",
        "s3:ListMultipartUploadParts",
        "s3:ListBucketMultipartUploads",
        "s3:PutObject",
        "s3:PutBucketAcl",
        "s3:PutBucketCors",
        "s3:DeleteObject",
        "s3:AbortMultipartUpload",
        "s3:CreateBucket"
    ],
    "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*",
        "arn:aws:s3:::*redshift*",
        "arn:aws:s3:::*redshift/*"
    ],
    "Condition": {
        "StringEqualsIgnoreCase": {
            "s3:ExistingObjectTag/Redshift": "true"
        }
    }
},
{
    "Effect": "Allow",
    "Action": [
        "dynamodb:Scan",
        "dynamodb:DescribeTable",
        "dynamodb:GetItem"
    ],
    "Resource": [
        "arn:aws:dynamodb::*::*:table/*redshift*"
    ]
}
```
Using identity-based policies (IAM policies)

```
"arn:aws:dynamodb:*:*:table/redshift*/index/*",
],
}),
{
"Effect": "Allow",
"Action": [
"elasticmapreduce:ListInstances"
],
"Resource": [
"arn:aws:elasticmapreduce:*:*:cluster/redshift*"
]
}),
{
"Effect": "Allow",
"Action": [
"elasticmapreduce:ListInstances"
],
"Resource": "*",
"Condition": {
"StringEqualsIgnoreCase": {
"elasticmapreduce:ResourceTag/Redshift": "true"
}
}
},
{
"Effect": "Allow",
"Action": [
"lambda:InvokeFunction"
],
"Resource": "arn:aws:lambda:*:*:function:redshift*"
},
{
"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": [
"arn:aws:glue:*:*:table/redshift*/**",
"arn:aws:glue:*:*:catalog",
"arn:aws:glue:*:*:database/redshift*"
]
},
{
"Effect": "Allow",
"Action": [
"secretsmanager:GetResourcePolicy",
"secretsmanager:GetSecretValue",
"secretsmanager:DescribeSecret",
```
**Using identity-based policies (IAM policies)**

```json
{
  "Effect": "Allow",
  "Action": [
    "secretsmanager:GetRandomPassword",
    "secretsmanager:ListSecrets"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "secretsmanager:ResourceTag/Redshift": "true"
    }
  }
}
{
  "Effect": "Allow",
  "Action": [
    "iam:PassRole"
  ],
  "Resource": "arn:aws:iam::*:role/*",
  "Condition": {
    "StringEquals": {
      "iam:PassedToService": [
        "redshift.amazonaws.com",
        "glue.amazonaws.com",
        "sagemaker.amazonaws.com",
        "athena.amazonaws.com"
      ]
    }
  }
}
}
```

Optionally, to use an AWS KMS key for encryption, add the following permissions to the policy.

```json
{
  "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>"
  ]
}
```

To allow Amazon Redshift and SageMaker to assume the preceding IAM role to interact with other services, add the following trust policy to the role.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "redshift.amazonaws.com",
          "glue.amazonaws.com",
          "sagemaker.amazonaws.com",
          "athena.amazonaws.com"
        ]
      }
    }
  ]
}
```
"Principal": {
    "Service": [
        "redshift.amazonaws.com",
        "sagemaker.amazonaws.com"
    ],
    "Action": "sts:AssumeRole"
}

In the preceding, the Amazon S3 bucket redshift-downloads/redshift-ml/ is the location where the sample data used for other steps and examples is stored. You can remove this bucket if you don’t need to load data from Amazon S3. Or replace it with other Amazon S3 buckets that you use to load data into Amazon Redshift.

The your-account-id, your-role, and your-s3-bucket values are the account ID, role, and bucket that you specify in your CREATE MODEL command.

Optionally, you can use the AWS KMS keys section of the sample policy if you specify an AWS KMS key for use with Amazon Redshift ML. The your-kms-key value is the key that you use as part of your CREATE MODEL command.

When you specify a private virtual private cloud (VPC) for a hyperparameter tuning job, add the following permissions.

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:CreateNetworkInterfacePermission",
        "ec2:DeleteNetworkInterface",
        "ec2:DeleteNetworkInterfacePermission",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
    ]
}
```

To work with model explanation, make sure that you have the permissions to call SageMaker API operations. We recommend that you use the AmazonSageMakerFullAccess managed policy. If you want to create an IAM role with a more restrictive policy, use the policy following.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sagemaker::CreateEndpoint",
                "sagemaker::CreateEndpointConfig",
                "sagemaker::DeleteEndpoint",
                "sagemaker::DeleteEndpointConfig",
                "sagemaker::DescribeEndpoint",
                "sagemaker::DescribeEndpointConfig",
                "sagemaker::DescribeModel",
                "sagemaker::InvokeEndpoint",
                "sagemaker::ListTags"
            ],
            "Resource": "*
```

356
For more information about the `AmazonSageMakerFullAccess` managed policy, see `AmazonSageMakerFullAccess` in the *Amazon SageMaker Developer Guide*.

For more information about Amazon Redshift ML, see *Using machine learning in Amazon Redshift* or `CREATE MODEL`.

**Permissions required to use the data sharing API operations**

To control access to the data sharing API operations, use IAM action-based policies. For information about how to manage IAM policies, see *Managing IAM policies* in the *IAM User Guide*.

In particular, suppose that a producer cluster administrator needs to use the `AuthorizeDataShare` call to authorize egress for a datashare outside of an AWS account. In this case, you set up an IAM action-based policy to grant this permission. Use the `DeauthorizeDataShare` call to revoke egress.

When using IAM action-based policies, you can also specify an IAM resource in the policy, such as `DataShareARN`. The following shows the format and an example for `DataShareARN`.

```plaintext
```

You can restrict `AuthorizeDataShare` access to a specific datashare by specifying the datashare name in the IAM policy.

```json
{
   "Statement": [
      {
         "Action": [
            "redshift:AuthorizeDataShare",
         ],
         "Resource": [
            "arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-e2e24359e9a8/SalesShare"
         ],
         "Effect": "Deny"
      }
   ]
}
```

You can also restrict the IAM policy to all datashares owned by a specific producer cluster. To do this, replace the `datashare-name` value in the policy with a wildcard or an asterisk. Keep the cluster's namespace-guid value.

```plaintext
arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-e2e24359e9a8/*
```

Following is an IAM policy that prevents an entity from calling `AuthorizeDataShare` on the datashares owned by a specific producer cluster.

```json
{
   "Statement": [
      {
         "Action": [
            "redshift:AuthorizeDataShare",
         ],
      }
   ]
}
```
DataShareARN restricts the access based on both the datashare name and the globally unique ID (GUID) for the owning cluster's namespace. It does this by specifying the name as an asterisk.

**Resource policies for GetClusterCredentials**

To connect to a cluster database using a JDBC or ODBC connection with IAM database credentials, or to programmatically call the `GetClusterCredentials` action, you need a minimum set of permissions. At a minimum, you need permission to call the `redshift:GetClusterCredentials` action with access to a `dbuser` resource.

If you use a JDBC or ODBC connection, instead of `server` and `port` you can specify `cluster_id` and `region`, but to do so your policy must permit the `redshift:DescribeClusters` action with access to the cluster resource.

If you call `GetClusterCredentials` with the optional parameters `Autocreate`, `DbGroups`, and `DbName`, make sure to also allow the actions and permit access to the resources listed in the following table.

<table>
<thead>
<tr>
<th>GetClusterCredentials parameter</th>
<th>Action</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocreate</td>
<td><code>redshift:CreateClusterUser</code></td>
<td></td>
</tr>
<tr>
<td>DbGroups</td>
<td><code>redshift:JoinGroup</code></td>
<td></td>
</tr>
<tr>
<td>DbName</td>
<td>NA</td>
<td><code>dbname</code></td>
</tr>
</tbody>
</table>

For more information about resources, see [Amazon Redshift resources and operations (p. 335)](https://docs.aws.amazon.com/redshift/latest/us-gov/pricing.html).

You can also include the following conditions in your policy:

- `redshift:DurationSeconds`
- `redshift:DbName`
- `redshift:DbUser`

For more information about conditions, see [Specifying conditions in a policy (p. 339)](https://docs.aws.amazon.com/redshift/latest/us-gov/security/using-policies.html).

**Customer managed policy examples**

In this section, you can find example user policies that grant permissions for various Amazon Redshift actions. These policies work when you are using the Amazon Redshift API, AWS SDKs, or the AWS CLI.

**Note**

All examples use the US West (Oregon) Region (`us-west-2`) and contain fictitious account IDs.

**Example 1: Allow user full access to all Amazon Redshift actions and resources**

The following policy allows access to all Amazon Redshift actions on all resources.
Using identity-based policies (IAM policies)

The value `redshift:*` in the `Action` element indicates all of the actions in Amazon Redshift.

**Example 2: Deny a user access to a set of Amazon Redshift actions**

By default, all permissions are denied. However, sometimes you need to explicitly deny access to a specific action or set of actions. The following policy allows access to all the Amazon Redshift actions and explicitly denies access to any Amazon Redshift action where the name starts with `Delete`. This policy applies to all Amazon Redshift resources in `us-west-2`.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":"AllowUSWest2Region",
      "Action": [ "redshift:*" ],
      "Effect": "Allow",
      "Resource": "arn:aws:redshift:us-west-2:*"
    },
    {
      "Sid":"DenyDeleteUSWest2Region",
      "Action": [ "redshift:Delete*" ],
      "Effect": "Deny",
      "Resource": "arn:aws:redshift:us-west-2:*"
    }
  ]
}
```

**Example 3: Allow a user to manage clusters**

The following policy allows a user to create, delete, modify, and reboot all clusters, and then denies permission to delete any clusters where the cluster name starts with `protected`.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":"AllowClusterManagement",
      "Action": [ "redshift:CreateCluster", "redshift:DeleteCluster", "redshift:ModifyCluster", "redshift:RebootCluster" ],
      "Effect": "Allow",
      "Resource": "arn:aws:redshift:us-west-2:*"
    }
  ]
}
```
Using identity-based policies (IAM policies)

Example 4: Allow a user to authorize and revoke snapshot access

The following policy allows a user, for example User A, to do the following:

- Authorize access to any snapshot created from a cluster named `shared`.
- Revoke snapshot access for any snapshot created from the `shared` cluster where the snapshot name starts with `revokable`.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSharedSnapshots",
            "Action": ["redshift:AuthorizeSnapshotAccess"],
            "Effect": "Allow"
        },
        {
            "Sid": "AllowRevokableSnapshot",
            "Action": ["redshift:RevokeSnapshotAccess"],
            "Effect": "Allow"
        }
    ]
}
```

If User A has allowed User B to access a snapshot, User B must have a policy such as the following to allow User B to restore a cluster from the snapshot. The following policy allows User B to describe and restore from snapshots, and to create clusters. The name of these clusters must start with `from-other-account`.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
```
Example 5: Allow a user to copy a cluster snapshot and restore a cluster from a snapshot

The following policy allows a user to copy any snapshot created from the cluster named big-cluster-1, and restore any snapshot where the snapshot name starts with snapshot-for-restore.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid":"AllowCopyClusterSnapshot",
         "Action": [ "redshift:CopyClusterSnapshot" ],
         "Effect": "Allow"
      },
      {
         "Sid":"AllowRestoreFromClusterSnapshot",
         "Action": [ "redshift:RestoreFromClusterSnapshot" ],
         "Effect": "Allow"
      }
   ]
}
```

Example 6: Allow a user access to Amazon Redshift, and common actions and resources for related AWS services

The following example policy allows access to all actions and resources for Amazon Redshift, Amazon Simple Notification Service (Amazon SNS), and Amazon CloudWatch. It also allows specified actions on all related Amazon EC2 resources under the account.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid":"AllowDescribeSnapshots",
         "Action": [ "redshift:DescribeClusterSnapshots" ],
         "Resource": [ "*" ],
         "Effect": "Allow"
      },
      {
         "Sid":"AllowUserRestoreFromSnapshot",
         "Action": [ "redshift:RestoreFromClusterSnapshot" ],
         "Effect": "Allow"
      }
   ]
}
```
Note
Resource-level permissions are not supported for the Amazon EC2 actions that are specified in this example policy.

```
{"Version": "2012-10-17",
"Statement": [
    {
        "Sid":"AllowRedshift",
        "Effect": "Allow",
        "Action": [
            "redshift:*"
        ],
        "Resource": [
            "*"
        ]
    },
    {
        "Sid":"AllowSNS",
        "Effect": "Allow",
        "Action": [
            "sns:*"
        ],
        "Resource": [
            "*"
        ]
    },
    {
        "Sid":"AllowCloudWatch",
        "Effect": "Allow",
        "Action": [
            "cloudwatch:*"
        ],
        "Resource": [
            "*"
        ]
    },
    {
        "Sid":"AllowEC2Actions",
        "Effect": "Allow",
        "Action": [
            "ec2:AllocateAddress",
            "ec2:AssociateAddress",
            "ec2:AttachNetworkInterface",
            "ec2:DescribeAccountAttributes",
            "ec2:DescribeAddresses",
            "ec2:DescribeAvailabilityZones",
            "ec2:DescribeInternetGateways",
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeSubnets",
            "ec2:DescribeVpcs"
        ],
        "Resource": [
            "*"
        ]
    }
]
```

Example 7: Allow a user to tag resources with the Amazon Redshift console

The following example policy allows a user to tag resources with the Amazon Redshift console using the AWS Resource Groups. This policy can be attached to a user role that invokes the new or original
Amazon Redshift console. For more information about tagging, see Tagging resources in Amazon Redshift (p. 512).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Tagging permissions",
            "Effect": "Allow",
            "Action": [
                "redshift:DeleteTags",
                "redshift:CreateTags",
                "redshift:DescribeTags",
                "tag:UntagResources",
                "tag:TagResources"
            ],
            "Resource": "*"
        }
    ]
}
```

**Example policy for using GetClusterCredentials**

The following policy uses these sample parameter values:

- **Region**: us-west-2
- **AWS Account**: 123456789012
- **Cluster name**: examplecluster

The following policy enables the GetCredentials, CreateClusterUser, and JoinGroup actions. The policy uses condition keys to allow the GetClusterCredentials and CreateClusterUser actions only when the AWS user ID matches "AIDIODR4TAW7CSEXAMPLE: ${redshift:DbUser}@yourdomain.com". IAM access is requested for the "testdb" database only. The policy also allows users to join a group named "common_group".

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetClusterCredsStatement",
            "Effect": "Allow",
            "Action": ["redshift:GetClusterCredentials"],
            "Resource": [
            ],
            "Condition": {
                "StringEquals": {
                    "aws:userid": "AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}@yourdomain.com"
                }
            }
        },
        {
            "Sid": "CreateClusterUserStatement",
            "Effect": "Allow",
            "Action": ["redshift:CreateClusterUser"
```

363
Native identity provider (IdP) federation for Amazon Redshift

Managing identities and permissions for Amazon Redshift is made easier with native identity provider federation because it leverages your existing identity provider to simplify authentication and managing permissions. It does this by making it possible to share identity metadata to Redshift from your identity provider. For the first iteration of this feature, the supported identity provider is Microsoft Azure Active Directory (Azure AD).

To configure Amazon Redshift so it can authenticate identities from the third-party identity provider, you register the identity provider with Amazon Redshift. Doing this enables Redshift to authenticate users and roles defined by the identity provider. Thus you can avoid having to perform granular identity management in both your third-party identity provider and in Amazon Redshift, because identity information is shared.

Setting up the identity provider on Amazon Redshift

This section shows the steps to configure the identity provider and Amazon Redshift to establish communication for native identity provider federation. You need an active account with your identity provider. Prior to configuring Amazon Redshift, you register Redshift as an application with your identity provider, granting administrator consent.

Complete the following steps in Amazon Redshift:

1. You run a SQL statement to register the identity provider, including descriptions of the Azure application metadata. To create the identity provider in Amazon Redshift, run the following command after replacing the parameter values `issuer`, `client_id`, `client_secret`, and `audience`. These parameters are specific to Microsoft Azure AD. Replace the identity provider name with a name of your choosing, and replace the namespace with a unique name to contain users and roles from your identity provider directory.

   ```sql
   CREATE IDENTITY PROVIDER oauth_standard TYPE azure
   ```
Native identity provider (IdP) federation for Amazon Redshift

<table>
<thead>
<tr>
<th>NAMESPACE 'aad'</th>
</tr>
</thead>
</table>
| PARAMETERS '{
|   "issuer":"https://sts.windows.net/2sddf4d3f-d475-420d-b5ac-667adad7c702/",
|   "client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
|   "client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
|   "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
|}'

The type azure indicates that the provider specifically facilitates communication with Microsoft Azure AD. This is currently the only supported third-party identity provider.

- **issuer** - The issuer ID to trust when a token is received. The unique identifier for the tenant_id is appended to the issuer.
- **client_id** - The unique, public identifier of the application registered with the identity provider. This can be referred to as the application ID.
- **client_secret** - A secret identifier, or password, known only to the identity provider and the registered application.
- **audience** - The Application ID that is assigned to the application in Azure.

2. Optional: Run SQL commands in Amazon Redshift to pre-create users and roles. This facilitates granting permissions in advance. The role name in Amazon Redshift is like the following: `<Namespace><GroupName on Azure AD>`. For example, when you create a group in Microsoft Azure AD called rsrgroup and a namespace called aad, the role name is aad:rsrgroup. The user name and group memberships are mapped to a user and roles in Amazon Redshift, in the identity provider namespace. This process includes verifying the users' and roles' external_id to ensure they are up to date.

   ```sql
   create role "aad:rsrgroup";
   ```

3. Grant relevant permissions to roles per your requirements. For example:

   ```sql
   GRANT SELECT on all tables in schema public to role "aad:rsrgroup";
   ```

4. You can also grant permissions to a specific user.

   ```sql
   GRANT SELECT on table foo to aad:alice@example.com
   ```

An explanation of namespaces

A namespace maps a user or role to a specific identity provider. For example, IAM users are prefixed by `iam:`. This is done to avoid user name collisions and support multiple identity stores. If a user alice@example.com from the identity source registered with `aad` namespace logs in, the user `aad:alice@example.com` is created in Redshift if it doesn't already exist. Note that a user and role namespace has a different function than an Amazon Redshift cluster namespace, which is a unique identifier associated with a cluster.

How login works with native identity provider (IdP) federation

To complete the preliminary setup between the identity provider and Amazon Redshift, you perform a couple of steps: First, you register Amazon Redshift as a third-party application with your identity provider, requesting the necessary API permissions. Then you create users and groups in the identity provider. Last, you register the identity provider with Amazon Redshift, using SQL statements, which set authentication parameters that are unique to the identity provider. As part of registering the identity provider with Redshift, you assign a namespace to make sure users and roles are grouped correctly.

With the identity provider registered with Amazon Redshift, communication is set up between Redshift and the identity provider. A client can then pass tokens and authenticate to Redshift as an identity
Amazon Redshift uses the IdP group membership information to map to Redshift roles. If the user doesn't previously exist in Redshift, the user is created. Roles are created that map to identity provider groups, if they don't exist. The Amazon Redshift administrator grants permission on the roles, and users can run queries and perform other database tasks.

The following steps outline how native identity provider federation works, when a user logs in:

1. When a user logs in using the native IdP option, from the client, the identity provider token is sent from the client to the driver.
2. The user is authenticated. If the user doesn't already exist in Amazon Redshift, a new user is created. Redshift maps the user's identity provider groups to Redshift roles.
3. Permissions are assigned, based on the user's Redshift roles. These are granted to users and roles by an administrator.
4. The user can query Redshift.

Using desktop client tools to connect to Amazon Redshift

For instructions on how to use native identity provider federation to connect to Amazon Redshift with Power BI, see the blog post Integrate Amazon Redshift native IdP federation with Microsoft Azure Active Directory (AD) and Power BI. It describes a step-by-step implementation of the Amazon Redshift native IdP setup with Azure AD. It also details the steps to set up the client connection for either Power BI Desktop or the Power BI service.

For instructions on how to use native identity provider federation to connect to Amazon Redshift with a SQL client, specifically DBeaver or SQL Workbench/J, see the blog post Integrate Amazon Redshift native IdP federation with Microsoft Azure AD using a SQL client.

Amazon Redshift API permissions reference

When you set up Access control (p. 335), you write permission policies that you can attach to an IAM identity (identity-based policies). For detailed reference information, see the following topics in the Service Authorization Reference:

- Actions, resources, and condition keys for Amazon Redshift.
- Actions, resources, and condition keys for Amazon Redshift Data API.
- Actions, resources, and condition keys for AWS SQL Workbench which list actions for Amazon Redshift query editor v2.

This reference contains information about which API operations can be used in an IAM policy. It also includes the AWS resource for which you can grant the permissions, and condition keys that you can include for fine-grained access control. For more information about conditions, see Using IAM policy conditions for fine-grained access control (p. 339).

You specify the actions in the policy's Action field, the resource value in the policy's Resource field, and conditions in the policy's Condition field. To specify an action for Amazon Redshift, use the redshift: prefix followed by the API operation name (for example, redshift:CreateCluster).

Using service-linked roles for Amazon Redshift

Amazon Redshift uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to Amazon Redshift. Service-linked roles are predefined by Amazon Redshift and include all the permissions that the service requires to call AWS services on behalf of your Amazon Redshift cluster.
A service-linked role makes setting up Amazon Redshift easier because you don't have to add the necessary permissions manually. The role is linked to Amazon Redshift use cases and has predefined permissions. Only Amazon Redshift can assume the role, and only the service-linked role can use the predefined permissions policy. Amazon Redshift creates a service-linked role in your account the first time you create a cluster. You can delete the service-linked role only after you delete all of the Amazon Redshift clusters in your account. This protects your Amazon Redshift resources because you can't inadvertently remove permissions needed for access to the resources.

Amazon Redshift supports using service-linked roles in all of the Regions where the service is available. For more information, see AWS Regions and Endpoints.

For information about other services that support service-linked roles, see AWS services that work with IAM and look for the services that have Yes in the Service-Linked Role column. Choose a Yes with a link to view the service-linked role documentation for that service.

### Service-linked role permissions for Amazon Redshift

Amazon Redshift uses the service-linked role named AWSServiceRoleForRedshift – Allows Amazon Redshift to call AWS services on your behalf. This service-linked role is attached to the following managed policy: AmazonRedshiftServiceLinkedRolePolicy. For updates to this policy, see AWS-managed (predefined) policies for Amazon Redshift.

The AWSServiceRoleForRedshift service-linked role trusts only redshift.amazonaws.com to assume the role.

The AWSServiceRoleForRedshift service-linked role permissions policy allows Amazon Redshift to complete the following on all related resources:

- `ec2:DescribeVpcs`
- `ec2:DescribeSubnets`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeAddress`
- `ec2:AssociateAddress`
- `ec2:DisassociateAddress`
- `ec2:CreateNetworkInterface`
- `ec2:DeleteNetworkInterface`
- `ec2:ModifyNetworkInterfaceAttribute`
- `ec2:CreateVpcEndpoint`
- `ec2:DeleteVpcEndpoints`
- `ec2:DescribeVpcEndpoints`
- `ec2:ModifyVpcEndpoint`
- `ec2:DescribeVpcAttribute`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeInternetGateways`
- `ec2:DescribeSecurityGroupRules`
- `ec2:DescribeAvailabilityZones`
- `ec2:DescribeNetworkAcls`
- `ec2:DescribeRouteTables`

**Permissions for network and VPC resources**
The following permissions allow actions on Amazon EC2 for creation and management of network and virtual-private-cloud resources. These are specifically associated with the `Purpose:RedshiftMigrateToVpc` resource tag. This limits the scope of the permissions to specific Amazon EC2 Classic to Amazon EC2 VPC migration tasks. For more information about resource tags, see Controlling access to AWS resources using tags.

- `ec2:AuthorizeSecurityGroupEgress`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:UpdateSecurityGroupRuleDescriptionsEgress`
- `ec2:ReplaceRouteTableAssociation`
- `ec2:CreateRouteTable`
- `ec2:AttachInternetGateway`
- `ec2:UpdateSecurityGroupRuleDescriptionsIngress`
- `ec2:AssociateRouteTable`
- `ec2:RevokeSecurityGroupIngress`
- `ec2:CreateRoute`
- `ec2:CreateSecurityGroup`
- `ec2:RevokeSecurityGroupEgress`
- `ec2:ModifyVpcAttribute`
- `ec2:CreateSubnet`
- `ec2:CreateInternetGateway`
- `ec2:CreateVpc`

See more information about actions and resources in Amazon EC2 at Actions, resources, and condition keys for Amazon EC2.

**Actions for audit logging**

Actions listed with the `logs` prefix pertain to audit logging and related features. Specifically, creation and management of log groups and log streams.

- `logs:CreateLogGroup`
- `logs:PutRetentionPolicy`
- `logs:CreateLogStream`
- `logs:PutLogEvents`
- `logs:DescribeLogStreams`
- `logs:GetLogEvents`

The following JSON shows actions and resource scope, to Amazon Redshift, for audit logging.

```json
[
  {
    "Sid": "EnableCreationAndManagementOfRedshiftCloudwatchLogGroups",
    "Effect": "Allow",
    "Action": [
      "logs:CreateLogGroup",
      "logs:PutRetentionPolicy"
    ],
    "Resource": [
      "arn:aws:logs:*:*:log-group:/aws/redshift/**"
    ]
  }
]```
To allow an IAM entity to create AWSServiceRoleForRedshift service-linked roles

```
{
  "Effect": "Allow",
  "Action": [
    "iam:CreateServiceLinkedRole"
  ],
  "Resource": "arn:aws:iam::<AWS-account-ID>:role/aws-service-role/redshift.amazonaws.com/AWSServiceRoleForRedshift",
  "Condition": {"StringLike": {"iam:AWSServiceName": "redshift.amazonaws.com"}}
}
```

To allow an IAM entity to delete AWSServiceRoleForRedshift service-linked roles

Add the following policy statement to the permissions for that IAM entity:

```
{
  "Effect": "Allow",
  "Action": [
    "iam:DeleteServiceLinkedRole",
    "iam:GetServiceLinkedRoleDeletionStatus"
  ],
  "Resource": "arn:aws:iam::<AWS-account-ID>:role/aws-service-role/redshift.amazonaws.com/AWSServiceRoleForRedshift",
  "Condition": {"StringLike": {"iam:AWSServiceName": "redshift.amazonaws.com"}}
}
```

Alternatively, you can use an AWS managed policy to provide full access to Amazon Redshift.

Creating a service-linked role for Amazon Redshift

You don't need to manually create an AWSServiceRoleForRedshift service-linked role. Amazon Redshift creates the service-linked role for you. If the AWSServiceRoleForRedshift service-linked role has been deleted from your account, Amazon Redshift creates the role when you launch a new Amazon Redshift cluster.

**Important**
If you used the Amazon Redshift service before September 18, 2017, when it began supporting service-linked roles, then Amazon Redshift created the AWSServiceRoleForRedshift role in your account. To learn more, see A new role appeared in my IAM account.
Editing a service-linked role for Amazon Redshift

Amazon Redshift does not allow you to edit the AWSServiceRoleForRedshift service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using the IAM console, the AWS Command Line Interface (AWS CLI), or IAM API. For more information, see Modifying a role in the IAM User Guide.

Deleting a service-linked role for Amazon Redshift

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained.

Before you can delete a service-linked role for an account, you must shut down and delete any clusters in the account. For more information, see Shutting down and deleting clusters (p. 59).

You can use the IAM console, the AWS CLI, or the IAM API to delete a service-linked role. For more information, see Deleting a service-linked role in the IAM User Guide.

Using IAM authentication to generate database user credentials

You can generate temporary database credentials based on permissions granted through an AWS Identity and Access Management (IAM) permissions policy to manage the access that your users have to your Amazon Redshift database.

Commonly, Amazon Redshift database users log in to the database by providing a database user name and password. However, you don't have to maintain user names and passwords in your Amazon Redshift database. As an alternative, you can configure your system to permit users to create user credentials and log in to the database based on their IAM credentials.

For more information, see Identity Providers and Federation in the IAM User Guide.

Topics

- Overview (p. 370)
- Creating temporary IAM user credentials (p. 371)
- Options for providing IAM credentials (p. 380)

Overview

Amazon Redshift provides the GetClusterCredentials API operation to generate temporary database user credentials. You can configure your SQL client with Amazon Redshift JDBC or ODBC drivers that manage the process of calling the GetClusterCredentials operation. They do so by retrieving the database user credentials, and establishing a connection between your SQL client and your Amazon Redshift database. You can also use your database application to programmatically call the GetClusterCredentials operation, retrieve database user credentials, and connect to the database.

If you already manage user identities outside AWS, you can use an identity provider (IdP) compliant with Security Assertion Markup Language (SAML) 2.0 to manage access to Amazon Redshift resources. You configure your IdP to permit your federated users access to an IAM role. With that IAM role, you can generate temporary database credentials and log in to Amazon Redshift databases.

Your SQL client needs permission to call the GetClusterCredentials operation for you. You manage those permissions by creating an IAM role and attaching an IAM permissions policy that grants or restricts access to the GetClusterCredentials operation and related actions.
The policy also grants or restricts access to specific resources, such as Amazon Redshift clusters, databases, database user names, and user group names.

**Note**

We recommend using the Amazon Redshift JDBC or ODBC drivers to manage the process of calling the `GetClusterCredentials` operation and logging on to the database. For simplicity, we assume that you are using a SQL client with the JDBC or ODBC drivers throughout this topic. For specific details and examples of using the `GetClusterCredentials` operation or the `get-cluster-credentials` CLI command, see `GetClusterCredentials` and `get-cluster-credentials`.

To manage authentication and authorization centrally, Amazon Redshift supports database authentication with IAM, enabling user authentication through enterprise federation. Instead of creating an IAM user, in this approach you use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an IdP.

To provide federated access to a user or client application in your organization to call Amazon Redshift API operations, you can also use the JDBC or ODBC driver with SAML 2.0 support to request authentication from your organization IdP. In this case, your organization's users don't have direct access to Amazon Redshift.

### Creating temporary IAM user credentials

In this section, you can find how to configure your system to generate temporary IAM-based database user credentials and log in to your database using the new credentials.

At a high level, the process flows as follows:

1. **Step 1: Create an IAM role for IAM Single Sign-On (SSO) access** *(p. 371)*
   
   (Optional) You can authenticate users for access to an Amazon Redshift database by integrating IAM authentication and a third-party identity provider (IdP).

2. **Step 2: Configure SAML assertions for your IdP** *(p. 372)*
   
   (Optional) To use IAM authentication using an IdP, you need to define a claim rule in your IdP application that maps users or groups in your organization to the IAM role. Optionally, you can include attribute elements to set `GetClusterCredentials` parameters.

3. **Step 3: Create an IAM role or user with permissions to call GetClusterCredentials** *(p. 373)*
   
   Your SQL client application assumes the IAM role when it calls the `GetClusterCredentials` operation. If you created an IAM role for identity provider access, you can add the necessary permission to that role.

4. **Step 4: Create a database user and database groups** *(p. 374)*
   
   (Optional) By default, `GetClusterCredentials` returns credentials create a new user if the user name doesn't exist. You can also choose to specify user groups that users join at logon. By default, database users join the PUBLIC group.

5. **Step 5: Configure a JDBC or ODBC connection to use IAM credentials** *(p. 375)*
   
   To connect to your Amazon Redshift database, you configure your SQL client to use an Amazon Redshift JDBC or ODBC driver.

**Step 1: Create an IAM role for IAM Single Sign-On (SSO) access**

If you don't use an identity provider for single sign-on access, you can skip this step.
If you already manage user identities outside of AWS, you can authenticate users for access to an Amazon Redshift database by integrating IAM authentication and a third-party SAML-2.0 identity provider (IdP).

For more information, see Identity Providers and Federation in the IAM User Guide.

Before you can use Amazon Redshift IdP authentication, create an AWS SAML identity provider. You create an IdP in the IAM console to inform AWS about the IdP and its configuration. Doing this establishes trust between your AWS account and the IdP. For steps to create a role, see Creating a Role for SAML 2.0 Federation (Console) in the IAM User Guide.

**Step 2: Configure SAML assertions for your IdP**

After you create the IAM role, you define a claim rule in your IdP application to map users or groups in your organization to the IAM role. For more information, see Configuring SAML Assertions for the Authentication Response in the IAM User Guide.

If you choose to use the optional GetClusterCredentials parameters `DbUser`, `AutoCreate`, and `DbGroups`, you have two options. You can set the values for the parameters with your JDBC or ODBC connection, or you can set the values by adding SAML attribute elements to your IdP. For more information about the `DbUser`, `AutoCreate`, and `DbGroups` parameters, see Step 5: Configure a JDBC or ODBC connection to use IAM credentials (p. 375).

**Note**

If you use an IAM policy variable `${redshift:DbUser}`, as described in Resource policies for GetClusterCredentials (p. 358) the value for `DbUser` is replaced with the value retrieved by the API operation's request context. The Amazon Redshift drivers use the value for the `DbUser` variable provided by the connection URL, rather than the value supplied as a SAML attribute.

To help secure this configuration, we recommend that you use a condition in an IAM policy to validate the `DbUser` value by using `RoleSessionName`. You can find examples of how to set a condition using an IAM policy in Example policy for using GetClusterCredentials (p. 363).

To configure your IdP to set the `DbUser`, `AutoCreate`, and `DbGroups` parameters, include the following Attribute elements:

- An Attribute element with the Name attribute set to "https://redshift.amazon.com/SAML/Attributes/DbUser"

Set the AttributeValue element to the name of a user that will connect to the Amazon Redshift database.

The value in the AttributeValue element must be lowercase, begin with a letter, contain only alphanumeric characters, underscore (`_`), plus sign (`+`), dot (`.`), at (`@`), or hyphen (`-`), and be fewer than 128 characters. Typically, the user name is a user ID (for example, bobsmith) or an email address (for example bobsmith@example.com). The value can't include a space (for example, a user's display name such as Bob Smith).

```
<Attribute Name="https://redshift.amazon.com/SAML/Attributes/DbUser">
  <AttributeValue>user-name</AttributeValue>
</Attribute>
```

- An Attribute element with the Name attribute set to "https://redshift.amazon.com/SAML/Attributes/AutoCreate"

Set the AttributeValue element to true to create a new database user if one doesn't exist. Set the AttributeValue to false to specify that the database user must exist in the Amazon Redshift database.

```
<Attribute Name="https://redshift.amazon.com/SAML/Attributes/AutoCreate">
  <AttributeValue>true</AttributeValue>
</Attribute>
```
• An Attribute element with the Name attribute set to set to "https://redshift.amazon.com/SAML/Attributes/DbGroups"

This element contains one or more AttributeValue elements. Set each AttributeValue element to a database group name that the DbUser joins for the duration of the session when connecting to the Amazon Redshift database.

<Attribute Name="https://redshift.amazon.com/SAML/Attributes/DbGroups">
  <AttributeValue>group1</AttributeValue>
  <AttributeValue>group2</AttributeValue>
  <AttributeValue>group3</AttributeValue>
</Attribute>

Step 3: Create an IAM role or user with permissions to call GetClusterCredentials

Your SQL client needs authorization to call the GetClusterCredentials operation on your behalf. To provide that authorization, you create an IAM user or role and attach a policy that grants the necessary permissions.

To create an IAM role with permissions to call GetClusterCredentials

1. Using the IAM service, create an IAM user or role. You can also use an existing user or role. For example, if you created an IAM role for identity provider access, you can attach the necessary IAM policies to that role.

2. Attach a permission policy with permission to call the redshift:GetClusterCredentials operation. Depending on which optional parameters you specify, you can also allow or restrict additional actions and resources in your policy:

   • To permit your SQL client to retrieve cluster ID, AWS Region, and port, include permission to call the redshift:DescribeClusters operation with the Redshift cluster resource.

   • If you use the AutoCreate option, include permission to call redshift:CreateClusterUser with the dbuser resource. The following Amazon Resource Name (ARN) specifies the Amazon Redshift dbuser. Replace region, account-id, and cluster-name with the values for your AWS Region, account, and cluster. For dbuser-name, specify the user name to use to log in to the cluster database.

     arn:aws:redshift:region:account-id:dbuser:cluster-name/dbuser-name

   • (Optional) Add an ARN that specifies the Amazon Redshift dbname resource in the following format. Replace region, account-id, and cluster-name with the values for your AWS Region, account, and cluster. For database-name, specify the name of a database that the user will log in to.


   • If you use the DbGroups option, include permission to call the redshift:JoinGroup operation with the Amazon Redshift dbgroup resource in the following format. Replace region, account-id, and cluster-name with the values for your AWS Region, account, and cluster. For dbgroup-name, specify the name of a user group that the users join at login.

     arn:aws:redshift:region:account-id:dbgroup:cluster-name/dbgroup-name

For more information and examples, see Resource policies for GetClusterCredentials (p. 358).
The following example shows a policy that allows the IAM role to call the `GetClusterCredentials` operation. Specifying the Amazon Redshift `dbuser` resource grants the role access to the database user name `temp_creds_user` on the cluster named `examplecluster`.

```json
{
   "Version": "2012-10-17",
   "Statement": {
      "Effect": "Allow",
      "Action": "redshift:GetClusterCredentials",
   }
}
```

You can use a wildcard (*) to replace all, or a portion of, the cluster name, user name, and database group names. The following example allows any user name beginning with `temp_` with any cluster in the specified account.

**Important**
The statement in the following example specifies a wildcard character (*) as part of the value for the resource so that the policy permits any resource that begins with the specified characters. Using a wildcard character in your IAM policies might be overly permissive. As a best practice, we recommend using the most restrictive policy feasible for your business application.

```json
{
   "Version": "2012-10-17",
   "Statement": {
      "Effect": "Allow",
      "Action": "redshift:GetClusterCredentials",
   }
}
```

The following example shows a policy that allows the IAM role to call the `GetClusterCredentials` operation with the option to automatically create a new user and specify groups the user joins at login. The "Resource": "*" clause grants the role access to any resource, including clusters, database users, or user groups.

```json
"Version": "2012-10-17",
"Statement": {
   "Effect": "Allow",
   "Action": ["redshift:GetClusterCredentials",
               "redshift:CreateClusterUser",
               "redshift:JoinGroup"
   ],
   "Resource": "*"
}
```

For more information, see Amazon Redshift ARN syntax.

### Step 4: Create a database user and database groups

Optionally, you can create a database user that you use to log in to the cluster database. If you create temporary user credentials for an existing user, you can disable the user's password to force the user to log on with the temporary password. Alternatively, you can use the `GetClusterCredentials` Autocreate option to automatically create a new database user.
You can create database user groups with the permissions you want the IAM database user to join at login. When you call the GetClusterCredentials operation, you can specify a list of user group names that the new user joins at login. These group memberships are valid only for sessions created using credentials generated with the given request.

**To create a database user and database groups**

1. Log in to your Amazon Redshift database and create a database user using `CREATE USER` or alter an existing user using `ALTER USER`.
2. Optionally, specify the `PASSWORD DISABLE` option to prevent the user from using a password. When a user's password is disabled, the user can log on only using temporary IAM user credentials. If the password is not disabled, the user can log on either with the password or using temporary IAM user credentials. You can't disable the password for a superuser.

   The following example creates a user with password disabled.

   ```
   create user temp_creds_user password disable;
   ```

   The following example disables the password for an existing user.

   ```
   alter user temp_creds_user password disable;
   ```

3. Create database user groups using `CREATE GROUP`.
4. Use the `GRANT` command to define access privileges for the groups.

**Step 5: Configure a JDBC or ODBC connection to use IAM credentials**

You can configure your SQL client with an Amazon Redshift JDBC or ODBC driver. This driver manages the process of creating database user credentials and establishing a connection between your SQL client and your Amazon Redshift database.

If you use an identity provider for authentication, specify the name of a credential provider plugin. The Amazon Redshift JDBC and ODBC drivers include plugins for the following SAML-based identity providers:

- Active Directory Federation Services (AD FS)
- PingOne
- Okta
- Microsoft Azure AD

For the steps to set up Microsoft Azure AD as an identity provider, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 385).

**To configure a JDBC connection to use IAM credentials**

1. Download the latest Amazon Redshift JDBC driver from the Configuring a JDBC driver version 1.x connection (p. 170) page.
2. Create a JDBC URL with the IAM credentials options in one of the following formats. To use IAM authentication, add `iam:` to the Amazon Redshift JDBC URL following `jdbc:redshift:` as shown in the following example.

   ```
   jdbc:redshift:iam://
   ```
Add cluster-name, region, and account-id. The JDBC driver uses your IAM account information and cluster name to retrieve the cluster ID and AWS Region. To do so, your IAM user or role must have permission to call the redshift:DescribeClusters operation with the specified cluster. If your IAM user or role doesn’t have permission to call the redshift:DescribeClusters operation, include the cluster ID, AWS Region, and port as shown in the following example. The port number is optional. The default port is 5439.

```
jdbc:redshift:iam://examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com:5439/dev
```

3. Add JDBC options to provide IAM credentials. You use different combinations of JDBC options to provide IAM credentials. For details, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

The following URL specifies AccessKeyId and SecretAccessKey for an IAM user.

```
```

The following example specifies a named profile that contains the IAM credentials.

```
```

4. Add JDBC options that the JDBC driver uses to call the GetClusterCredentials API operation. Don’t include these options if you call the GetClusterCredentials API operation programmatically.

The following example includes the JDBC GetClusterCredentials options.

```
```

**To configure an ODBC connection to use IAM credentials**

In the following procedure, you can find steps only to configure IAM authentication. For steps to use standard authentication, using a database user name and password, see Configuring an ODBC connection (p. 179).

1. Install and configure the latest Amazon Redshift OBDC driver for your operating system. For more information, see Configuring an ODBC connection (p. 179) page.

   **Important**
   
   The Amazon Redshift ODBC driver must be version 1.3.6.1000 or later.

2. Follow the steps for your operating system to configure connection settings.

   For more information, see one of the following:

   - Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180)
   - Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 184)

3. On Microsoft Windows operating systems, access the Amazon Redshift ODBC Driver DSN Setup window.

   a. Under **Connection Settings**, enter the following information:
• **Data Source Name**
• **Server** (optional)
• **Port** (optional)
• **Database**

If your IAM user or role has permission to call the `redshift:DescribeClusters` operation, only **Data Source Name** and **Database** are required. Amazon Redshift uses **ClusterId** and **Region** to get the server and port by calling the `DescribeCluster` operation.

If your IAM user or role doesn't have permission to call the `redshift:DescribeClusters` operation, specify **Server** and **Port**. The default port is 5439.

b. Under **Authentication**, choose a value for **Auth Type**.

For each authentication type, enter values as listed following:

**AWS Profile**

Enter the following information:

• **ClusterID**
• **Region**
• **Profile name**

Enter the name of a profile in an AWS config file that contains values for the ODBC connection options. For more information, see Using a Configuration Profile (p. 380).

(Optional) Provide details for options that the ODBC driver uses to call the `GetClusterCredentials` API operation:

• **DbUser**
• **User AutoCreate**
• **DbGroups**

For more information, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

**IAM Credentials**

Enter the following information:

• **ClusterID**
• **Region**
• **AccessKeyId** and **SecretAccessKey**

The access key ID and secret access key for the IAM role or IAM user configured for IAM database authentication.

• **SessionToken**

**SessionToken** is required for an IAM role with temporary credentials. For more information, see Temporary Security Credentials.

Provide details for options that the ODBC driver uses to call the `GetClusterCredentials` API operation:

• **DbUser** (required)
• **User AutoCreate** (optional)
Identity Provider: AD FS

For Windows Integrated Authentication with AD FS, leave **User** and **Password** empty.

Provide IdP details:

- **IdP Host**

  The name of the corporate identity provider host. This name should not include any slashes (/).

- **IdP Port** (optional)

  The port used by identity provider. The default is 443.

- **Preferred Role**

  An Amazon Resource Name (ARN) for the IAM role from the multi-valued AttributeValue elements for the Role attribute in the SAML assertion. To find the appropriate value for the preferred role, work with your IdP administrator. For more information, see Configure SAML assertions for your IdP (p. 372).

(Optional) Provide details for options that the ODBC driver uses to call the GetClusterCredentials API operation:

- **DbUser**
- **User AutoCreate**
- **DbGroups**

For more information, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

Identity Provider: PingFederate

For **User** and **Password**, enter your IdP user name and password.

Provide IdP details:

- **IdP Host**

  The name of the corporate identity provider host. This name should not include any slashes (/).

- **IdP Port** (optional)

  The port used by identity provider. The default is 443.

- **Preferred Role**

  An Amazon Resource Name (ARN) for the IAM role from the multi-valued AttributeValue elements for the Role attribute in the SAML assertion. To find the appropriate value for the preferred role, work with your IdP administrator. For more information, see Configure SAML assertions for your IdP (p. 372).

(Optional) Provide details for options that the ODBC driver uses to call the GetClusterCredentials API operation:

- **DbUser**
Using IAM authentication to generate database user credentials

• User AutoCreate
• DbGroups

For more information, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

Identity Provider: Okta

For User and Password, enter your IdP user name and password.

Provide IdP details:

• IdP Host
  The name of the corporate identity provider host. This name should not include any slashes (/).

• IdP Port
  This value is not used by Okta.

• Preferred Role
  An Amazon Resource Name (ARN) for the IAM role from the AttributeValue elements for the Role attribute in the SAML assertion. To find the appropriate value for the preferred role, work with your IdP administrator. For more information, see Configure SAML assertions for your IdP (p. 372).

• Okta App ID
  An ID for an Okta application. The value for App ID follows "amazon_aws" in the Okta application embed link. Work with your IdP administrator to get this value.

(Optional) Provide details for options that the ODBC driver uses to call the GetClusterCredentials API operation:

• DbUser
• User AutoCreate
• DbGroups

For more information, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

Identity Provider: Azure AD

For User and Password, enter your IdP user name and password.

For Cluster ID and Region, enter the cluster ID and AWS Region of your Amazon Redshift cluster.

For Database, enter the database that you created for your Amazon Redshift cluster.

Provide IdP details:

• IdP Tenant
  The tenant used for Azure AD.

• Azure Client Secret
  The client secret of the Amazon Redshift enterprise app in Azure.
The client ID (application ID) of the Amazon Redshift enterprise app in Azure.

(Optional) Provide details for options that the ODBC driver uses to call the GetClusterCredentials API operation:

- DbUser
- User AutoCreate
- DbGroups

For more information, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

**Options for providing IAM credentials**

To provide IAM credentials for a JDBC or ODBC connection, choose one of the following options.

- **AWS profile**

  As an alternative to providing credentials values in the form of JDBC or ODBC settings, you can put the values in a named profile. For more information, see Using a Configuration Profile (p. 380).

- **IAM credentials**

  Provide values for AccessKeyID, SecretAccessKey, and, optionally, SessionToken in the form of JDBC or ODBC settings. SessionToken is required only for an IAM role with temporary credentials. For more information, see JDBC and ODBC options for providing IAM credentials (p. 381).

- **Identity provider federation**

  When you use identity provider federation to enable users from an identity provider to authenticate to Amazon Redshift, specify the name of a credential provider plugin. For more information, see Using a credentials provider plugin (p. 381).

  The Amazon Redshift JDBC and ODBC drivers include plugins for the following SAML-based identity federation credential providers:

  - Microsoft Active Identity Federation Services (AD FS)
  - PingOne
  - Okta
  - Microsoft Azure Active Directory (Azure AD)

  You can provide the plugin name and related values in the form of JDBC or ODBC settings or by using a profile. For more information, see Configure JDBC driver options (p. 175) and Configure ODBC driver options (p. 188).

  For more information, see Configure a JDBC or ODBC connection to use IAM credentials (p. 375).

**Using a Configuration Profile**

You can supply the IAM credentials options and GetClusterCredentials options as settings in named profiles in your AWS configuration file. To provide the profile name, use the Profile JDBC option. The configuration is stored in a file named `config` or a file named `credentials` in a folder named `.aws` in your home directory.

For a SAML-based credential provider plugin included with an Amazon Redshift JDBC or ODBC driver, you can use the settings described just preceding in Using a credentials provider plugin (p. 381). If `plugin_name` isn’t used, the other options are ignored.

The following example shows the `~/.aws/credentials` file with two profiles.
Using IAM authentication to generate database user credentials

```text
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtznFEMT/K7MDENG/bFxRfiCYEXAMPLEKEY

[user2]
aws_access_key_id=AKIAI44QH8DH5BEXAMPLE
aws_secret_access_key=je7MtGbc1wBF25p9Utk/h3yCo8nvbEXAMPLEKEY
session_token=AQoDYXdzEPT/////////wEXAMPLEtc76bNrC9APBSM2wDOk4x4HlZ8j4FZTwqQLWwKHNCBuFqwAeMicRXmxzfPSFfIeoYRqTflfKD8YUuwthAxF7mSEI/qkPpEPi/kMcGd
QrmGdemeH41C1NtBmp2wUB6phUZampKsbYEDy0PKpyQDYw77Z0wq5V5Xdp75YU9HFv1rd8Tx6g6fE8YQcHNVXakiY9g6d+xo0rKWvT38xVq7ZDqU0iPPkUL641IZbqBAZ+scqKnNz6FFdypN9CYjec8fPGLn9FX9SYVKT4rvx3I1I7Jab1QwjJ21CCROLx8A==
```

To use the credentials for the user2 example, specify Profile=user2 in the JDBC URL.

For more information on using profiles, see Named Profiles in the AWS Command Line Interface User Guide.

For more information on using profiles for JDBC driver, see Amazon Redshift JDBC driver installation and configuration guide.

For more information on using profiles for ODBC driver, see Amazon Redshift ODBC driver installation and configuration guide.

### JDBC and ODBC options for providing IAM credentials

The following table lists the JDBC and ODBC options for providing IAM credentials.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iam</td>
<td>For use only in an ODBC connection string. Set to 1 to use IAM authentication.</td>
</tr>
<tr>
<td>AccessKeyId</td>
<td>The access key ID and secret access key for the IAM role or IAM user configured for IAM database authentication. SessionToken is required only for an IAM role with temporary credentials. SessionToken is not used for an IAM user. For more information, see Temporary Security Credentials.</td>
</tr>
<tr>
<td>SecretAccessKey</td>
<td></td>
</tr>
<tr>
<td>SessionToken</td>
<td></td>
</tr>
<tr>
<td>plugin_name</td>
<td>The fully qualified name of a class that implements a credentials provider. The Amazon Redshift JDBC driver includes SAML-based credential provider plugins. If you provide plugin_name, you can also provide other related options. For more information, see Using a credentials provider plugin (p. 381).</td>
</tr>
<tr>
<td>Profile</td>
<td>The name of a profile in an AWS credentials or config file that contains values for the JDBC connection options. For more information, see Using a Configuration Profile (p. 380).</td>
</tr>
</tbody>
</table>

**Using a credentials provider plugin**

Amazon Redshift uses credentials provider plugins for SSO authentication.

To support SSO authentication, Amazon Redshift provides the Azure AD plugin for Microsoft Azure Active Directory. For information on how to configure this plugin, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 385).
Setting up multi-factor authentication

To support multi-factor authentication (MFA), Amazon Redshift provides browser-based plugins. Use the browser SAML plugin for Okta, PingOne, and the browser Azure AD plugin for Microsoft Azure Active Directory.

With the browser SAML plugin, SAML authentication flows like this:

1. A user tries to log in.
2. The plugin launches a local server to listen to incoming connections on the localhost.
3. The plugin launches a web browser to request a SAML response over HTTPS from the specified SSO login URL federated identity provider endpoint.
4. The web browser follows the link and prompts the user to enter credentials.
5. After the user authenticates and grants consent, the federated identity provider endpoint returns a SAML response over HTTPS to the URI indicated by redirect_uri.
6. The web browser moves the response message with the SAML response to the indicated redirect_uri.
7. The local server accepts the incoming connection and the plugin retrieves the SAML response and passes it to Amazon Redshift.

With the browser Azure AD plugin, SAML authentication flows like this:
1. A user tries to log in.
2. The plugin launches a local server to listen to incoming connections on the localhost.
3. The plugin launches a web browser to request an authorization code from the Azure AD `oauth2/authorize` endpoint.
4. The web browser follows the generated link over HTTPS and prompts the user to enter credentials. The link is generated using configuration properties, such as tenant and client_id.
5. After the user authenticates and grants consent, the Azure AD `oauth2/authorize` endpoint returns and sends a response over HTTPS with the authorization code to the indicated redirect_uri.
6. The web browser moves the response message with the SAML response to the indicated redirect_uri.
7. The local server accepts the incoming connection and the plugin requests and retrieves the authorization code and sends a POST request to the Azure AD `oauth2/token` endpoint.
8. The Azure AD `oauth2/token` endpoint returns a response with an access token to the indicated redirect_uri.
9. The plugin retrieves the SAML response and passes it to Amazon Redshift.

See the following sections:

- Active Directory Federation Services (AD FS)
  For more information, see Setting up JDBC or ODBC Single Sign-on authentication with AD FS (p. 392).
- PingOne (Ping)
  Ping is supported only with the predetermined PingOne IdP Adapter using Forms authentication.
For more information, see Setting up JDBC or ODBC Single Sign-on authentication with Ping Identity (p. 395).

- **Okta**

  Okta is supported only for the Okta-supplied application used with the AWS Management Console.

  For more information, see Setting up JDBC or ODBC Single Sign-on authentication with Okta (p. 399).

- **Microsoft Azure Active Directory**

  For more information, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 385).

**Configuring plugin options**

Configuring plugin options

To use a SAML-based credentials provider plugin, specify the following options using JDBC or ODBC options or in a named profile. If `plugin_name` isn't specified, the other options are ignored.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>plugin_name</strong></td>
<td>For JDBC, the class name that implements a credentials provider. Specify one of the following:</td>
</tr>
<tr>
<td></td>
<td>For Active Directory Federation Services</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.AdfsCredentialsProvider</td>
</tr>
<tr>
<td></td>
<td>For Okta</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.OktaCredentialsProvider</td>
</tr>
<tr>
<td></td>
<td>For PingFederate</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.PingCredentialsProvider</td>
</tr>
<tr>
<td></td>
<td>For Microsoft Azure Active Directory</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.AzureCredentialsProvider</td>
</tr>
<tr>
<td></td>
<td>For SAML MFA</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.BrowserSamlCredentialsProvider</td>
</tr>
<tr>
<td></td>
<td>For Microsoft Azure Active Directory SSO with MFA</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.BrowserAzureCredentialsProvider</td>
</tr>
<tr>
<td></td>
<td>For ODBC, specify one of the following:</td>
</tr>
<tr>
<td></td>
<td>For Active Directory Federation Services: adfs</td>
</tr>
<tr>
<td></td>
<td>For Okta: okta</td>
</tr>
<tr>
<td></td>
<td>For PingFederate: ping</td>
</tr>
</tbody>
</table>
Using IAM authentication to generate database user credentials

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• For Microsoft Azure Active Directory: azure</td>
</tr>
<tr>
<td></td>
<td>• For SAML MFA: browser saml</td>
</tr>
<tr>
<td></td>
<td>• For Microsoft Azure Active Directory SSO with MFA: browser azure ad</td>
</tr>
<tr>
<td>idp_host</td>
<td>The name of the corporate identity provider host. This name should not include any slashes ('/'). For an Okta identity provider, the value for idp_host should end with .okta.com.</td>
</tr>
<tr>
<td>idp_port</td>
<td>The port used by the identity provider. The default is 443. This port is ignored for Okta.</td>
</tr>
<tr>
<td>preferred_role</td>
<td>An Amazon Resource Name (ARN) from the AttributeValue elements for the Role attribute in the SAML assertion. To find the appropriate value for the preferred role, work with your IdP administrator. For more information, see Configure SAML assertions for your IdP (p. 372).</td>
</tr>
<tr>
<td>user</td>
<td>A corporate user name, including the domain when applicable. For example, for Active Directory, the domain name is required in the format domain\username.</td>
</tr>
<tr>
<td>password</td>
<td>The corporate user's password. We recommend not using this option. Instead, use your SQL client to supply the password.</td>
</tr>
<tr>
<td>app_id</td>
<td>An ID for an Okta application. Used only with Okta. The value for app_id follows amazon_aws in the Okta application embed link. To get this value, work with your IdP administrator. The following is an example of an application embed link: <a href="https://example.okta.com/home/amazon_aws/0oa2hy1wrpM8Ugehdt7/272">https://example.okta.com/home/amazon_aws/0oa2hy1wrpM8Ugehdt7/272</a></td>
</tr>
<tr>
<td>idp_tenant</td>
<td>A tenant used for Azure AD. Used only with Azure.</td>
</tr>
<tr>
<td>client_id</td>
<td>A client ID for the Amazon Redshift enterprise application in Azure AD. Used only with Azure.</td>
</tr>
</tbody>
</table>

Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD

You can use Microsoft Azure AD as an identity provider (IdP) to access your Amazon Redshift cluster. Following, you can find a procedure that describes how to set up a trust relationship for this purpose. For more information about configuring AWS as a service provider for the IdP, see Configuring Your SAML 2.0 IdP with Relying Party Trust and Adding Claims in the IAM User Guide.

**Note**
To use Azure AD with JDBC, the Amazon Redshift JDBC driver must be version 1.2.37.1061 or later. To use Azure AD with ODBC, the Amazon Redshift ODBC driver must be version 1.4.10.1000 or later.

Watch the following video to learn how to federate Amazon Redshift access with Microsoft Azure AD single sign-on: Federating Amazon Redshift access with Microsoft Azure AD single sign-on.

**To set up Azure AD and your AWS account to trust each other**

1. Create or use an existing Amazon Redshift cluster for your Azure AD users to connect to. To configure the connection, certain properties of this cluster are needed, such as the cluster identifier. For more information, see Creating a Cluster.
2. Set up an Azure Active Directory, groups, users used for AWS on the Microsoft Azure portal.
3. Add Amazon Redshift as an enterprise application on the Microsoft Azure portal to use for single sign-on to the AWS Console and federated login to Amazon Redshift. Choose Enterprise application.
5. Search for **AWS** in the search field.
6. Choose **Amazon Web Services (AWS)** and choose **Add**. This creates the AWS application.
7. Under **Manage**, choose **Single sign-on**.
8. Choose **SAML**. The Amazon Web Services (AWS) | SAML-based Sign-on page appears.
9. Choose Yes to proceed to the Set up Single Sign-On with SAML page. This page shows the list of pre-configured AWS SSO-related attributes.
10. For **Basic SAML Configuration**, choose the edit icon and choose **Save**.
11. When you are configuring for more than one application, provide an identifier value. For example, enter [https://signin.aws.amazon.com/saml#2](https://signin.aws.amazon.com/saml#2). Note that from the second application onwards, use this format with a # sign to specify a unique SPN value.
12. In the **User Attributes and Claims** section, choose the edit icon.

   By default, the Unique User Identifier (UID), Role, RoleSessionName, and SessionDuration claims are pre-configured.
13. Choose + **Add new claim** to add a claim for database users.

   For **Name**, enter **DbUser**.

   For **Namespace**, enter [https://redshift.amazon.com/SAML/Attributes](https://redshift.amazon.com/SAML/Attributes).

   For **Source**, choose **Attribute**.

   For **Source attribute**, choose user.userprincipalname. Then, choose **Save**.
14. Choose + **Add new claim** to add a claim for AutoCreate.

   For **Name**, enter **AutoCreate**.

   For **Namespace**, enter [https://redshift.amazon.com/SAML/Attributes](https://redshift.amazon.com/SAML/Attributes).

   For **Source**, choose **Attribute**.

   For **Source attribute**, choose “true”. Then, choose **Save**.

   Here, 123456789012 is your AWS account, AzureSSO is an IAM role you created, and AzureADProvider is the IAM provider.

<table>
<thead>
<tr>
<th>Claim name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique user identifier (name ID)</td>
<td>user.userprincipalname</td>
</tr>
<tr>
<td><a href="https://aws.amazon.com/SAML/Attributes/SessionDuration">https://aws.amazon.com/SAML/Attributes/SessionDuration</a></td>
<td>&quot;900&quot;</td>
</tr>
<tr>
<td><a href="https://aws.amazon.com/SAML/Attributes/RoleSessionName">https://aws.amazon.com/SAML/Attributes/RoleSessionName</a></td>
<td>user.userprincipalname</td>
</tr>
<tr>
<td><a href="https://redshift.amazon.com/SAML/Attributes/AutoCreate">https://redshift.amazon.com/SAML/Attributes/AutoCreate</a></td>
<td>&quot;true&quot;</td>
</tr>
<tr>
<td><a href="https://redshift.amazon.com/SAML/Attributes/DbGroups">https://redshift.amazon.com/SAML/Attributes/DbGroups</a></td>
<td>user.assignedroles</td>
</tr>
</tbody>
</table>
Using IAM authentication to generate database user credentials

15. Under App Registration > your-application-name > Authentication, add Mobile And Desktop Application. Specify the URL as http://localhost/redshift/.

16. In the SAML Signing Certificate section, choose Download to download and save the federation metadata XML file for use when you create an IAM SAML identity provider. This file is used to create the AWS SSO federated identity.

17. Create an IAM SAML identity provider on the IAM console. The metadata document that you provide is the federation metadata XML file that you saved when you set up Azure Enterprise Application. For detailed steps, see Creating and Managing an IAM Identity Provider (Console) in the IAM User Guide.

18. Create an IAM role for SAML 2.0 federation on the IAM console. For detailed steps, see Creating a Role for SAML in the IAM User Guide.

19. Create an IAM policy that you can attach to the IAM role that you created for SAML 2.0 federation on the IAM console. For detailed steps, see Creating IAM Policies (Console) in the IAM User Guide.

Modify the following policy (in JSON format) for your environment:

- Substitute the AWS Region of your cluster for us-west-1.
- Substitute your AWS account for 123456789012.
- Substitute your cluster identifier (or * for all clusters) for cluster-identifier.
- Substitute your database (or * for all databases) for dev.
- Substitute the unique identifier of your IAM role for AROAJ2UCCR6DPCEXAMPLE.
- Substitute your tenant or company email domain for example.com.
- Substitute the database group that you plan to assign the user to for my_dbgroup.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentials",
            "Resource": [
            ],
            "Condition": {
                "StringEquals": {
                    "aws:userid": "AROAJ2UCCR6DPCEXAMPLE:${redshift:DbUser}@example.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "redshift:CreateClusterUser",
```
Amazon Redshift Cluster Management Guide
Using IAM authentication to generate database user credentials

```
"Action": "redshift:JoinGroup",
},
{
 "Effect": "Allow",
 "Action": [
  "redshift:DescribeClusters",
  "iam:ListRoles"
  ],
  "Resource": "*"
}
```

This policy grants permissions as follows:

- The first section grants permission to the GetClusterCredentials API operation to get temporary credentials for the specified cluster. In this example, the resource is `cluster-identifier` with database `dev`, in account `123456789012`, and in AWS Region `us-west-1`. The `$(redshift:DbUser)` clause allows only users that match the `DbUser` value specified in Azure AD to connect.

- The condition clause enforces that only certain users get temporary credentials. These are users under the role specified by the role unique ID `AROAJ2UCCR6DPCEXAMPLE` in the IAM account identified by an email address in your company's email domain. For more information about unique IDs, see Unique IDs in the IAM User Guide.

Your setup with your IdP (in this case, Azure AD) determines how the condition clause is written. If your employee's email is `johndoe@example.com`, first set `$(redshift:DbUser)` to the super field that matches the employee's user name `johndoe`. Then, to make this condition work, set the AWS SAML RoleSessionName field to the super field that matches the employee's email `johndoe@example.com`. When you take this approach, consider the following:

- If you set `$(redshift:DbUser)` to be the employee's email, then remove the `@example.com` in the example JSON to match the RoleSessionName.

- If you set the RoleSessionId to be just the employee's user name, then remove the `@example.com` in the example to match the RoleSessionName.

- In the example JSON, the `$(redshift:DbUser)` and RoleSessionName are both set to the employee's email. This example JSON uses the Amazon Redshift database user name with `@example.com` to sign the user in to access the cluster.

- The second section grants permission to create a `dbuser` name in the specified cluster. In this example JSON, it restricts creation to `$(redshift:DbUser)`.  

- The third section grants permission to specify which `dbgroup` a user can join. In this example JSON, a user can join the `my_dbgroup` group in the specified cluster.

- The fourth section grants permission to actions the user can do on all resources. In this example JSON, it allows users to call `redshift:DescribeClusters` to get cluster information such as the cluster endpoint, AWS Region, and port. It also allows users to call `iam:ListRoles` to check which roles a user can assume.

To set up JDBC for authentication to Microsoft Azure AD

- Configure your database client to connect to your cluster through JDBC using your Azure AD single sign-on.

You can use any client that uses a JDBC driver to connect using Azure AD single sign-on or use a language like Java to connect using a script. For installation and configuration information, see Configuring a JDBC driver version 1.x connection (p. 170).
For example, you can use SQLWorkbench/J as the client. When you configure SQLWorkbench/J, the URL of your database uses the following format.

```
jdbc:redshift:iam://cluster-identifier:us-west-1/dev
```

If you use SQLWorkbench/J as the client, take the following steps:

a. Start SQL Workbench/J. On the Select Connection Profile page, add a Profile Group called AzureAuth.

b. For Connection Profile, enter Azure.

c. Choose Manage Drivers, and choose Amazon Redshift. Choose the Open Folder icon next to Library, then choose the appropriate JDBC .jar file.

d. On the Select Connection Profile page, add information to the connection profile as follows:
   - For User, enter your Microsoft Azure user name. This is the user name of the Microsoft Azure account that you are using for single sign-on that has permission to the cluster that you are trying to authenticate using.
   - For Password, enter your Microsoft Azure password.
   - For Drivers, choose Amazon Redshift (com.amazon.redshift.jdbc.Driver).

e. Choose Extended Properties to add additional information to the connection properties, as described following.

For Azure AD SSO configuration, add additional information as follows:

   - For plugin_name, enter `com.amazon.redshift.plugin.AzureCredentialsProvider`. This value specifies to the driver to use Azure AD Single Sign-On as the authentication method.
   - For idp_tenant, enter `your-idp-tenant`. Used only for Microsoft Azure AD. This is the tenant name of your company configured on Azure AD. This value can either be the tenant name or the tenant unique ID with hyphens.
   - For client_secret, enter `your-azure-redshift-application-client-secret`. Used only for Microsoft Azure AD. This is your client secret of the Amazon Redshift application that you created when setting up your Azure Single Sign-On configuration. This is only applicable to the com.amazon.redshift.plugin.AzureCredentialsProvider plugin.
   - For client_id, enter `your-azure-redshift-application-client-id`. Used only for Microsoft Azure AD. This is the client ID (with hyphens) of the Amazon Redshift application that you created when setting up your Azure Single Sign-On configuration.

For Azure AD SSO with MFA configuration, add additional information to the connection properties as follows:

   - For plugin_name, enter `com.amazon.redshift.plugin.BrowserAzureCredentialsProvider`. This value specifies to the driver to use Azure AD SSO with MFA as the authentication method.
   - For idp_tenant, enter `your-idp-tenant`. Used only for Microsoft Azure AD. This is the tenant name of your company configured on Azure AD. This value can either be the tenant name or the tenant unique ID with hyphens.
   - For client_id, enter `your-azure-redshift-application-client-id`. This option is used only for Microsoft Azure AD. This is the client ID (with hyphens) of the Amazon Redshift application that you created when setting up your Azure AD SSO with MFA configuration.
Using IAM authentication to
generate database user credentials

- For `listen_port`, enter `your-listen-port`. This is the port that local server is listening to. The default is 7890.
- For `idp_response_timeout`, enter `the-number-of-seconds`. This is the number of seconds to wait before timing out when the IdP server sends back a response. The minimum number of seconds must be 10. If establishing the connection takes longer than this threshold, then the connection is aborted.

To set up ODBC for authentication to Microsoft Azure AD

- Configure your database client to connect to your cluster through ODBC using your Azure AD single sign-on.

Amazon Redshift provides ODBC drivers for Linux, Windows, and macOS operating systems. Before you install an ODBC driver, determine whether your SQL client tool is 32-bit or 64-bit. Install the ODBC driver that matches the requirements of your SQL client tool.

Also install and configure the latest Amazon Redshift ODBC driver for your operating system as follows:

- For Windows, see Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180).
- For macOS, see Install the Amazon Redshift ODBC driver on macOS X (p. 183).
- For Linux, see Install the Amazon Redshift ODBC driver on Linux (p. 182).

On Windows, in the Amazon Redshift ODBC Driver DSN Setup page, under Connection Settings, enter the following information:

- For Data Source Name, enter `your-DSN`. This specifies the data source name used as the ODBC profile name.
- For Auth type for Azure AD SSO configuration, choose Identity Provider: Azure AD. This is the authentication method that the ODBC driver uses to authenticate using Azure single sign-on.
- For Auth type for Azure AD SSO with MFA configuration, choose Identity Provider: Browser Azure AD. This is the authentication method that the ODBC driver uses to authenticate using Azure single sign-on with MFA.
- For Cluster ID, enter `your-cluster-identifier`.
- For Region, enter `your-cluster-region`.
- For Database, enter `your-database-name`.
- For User, enter `your-azure-username`. This is the user name for the Microsoft Azure account that you are using for single sign-on that has permission to the cluster that you're trying to authenticate using. Use this only for Auth Type is Identity Provider: Azure AD.
- For Password, enter `your-azure-password`. Use this only for Auth Type is Identity Provider: Azure AD.
- For IdP Tenant, enter `your-idp-tenant`. This is the tenant name of your company configured on your IdP (Azure). This value can either be the tenant name or the tenant unique ID with hyphens.
- For Azure Client Secret, enter `your-azure-redshift-application-client-secret`. This is the client secret of the Amazon Redshift application that you created when setting up your Azure single sign-on configuration.
- For Azure Client ID, enter `your-azure-redshift-application-client-id`. This is the client ID (with hyphens) of the Amazon Redshift application that you created when setting up your Azure single sign-on configuration.
- For Listen Port, enter `your-listen-port`. This is the default listen port that local server is listening to. The default is 7890. This applies only to the Browser Azure AD plugin.
Amazon Redshift Cluster Management Guide
Using IAM authentication to generate database user credentials

- For **Response Timeout**, enter *the-number-of-seconds*. This is the number of seconds to wait before timing out when the IdP server sends back a response. The minimum number of seconds must be 10. If establishing the connection takes longer than this threshold, then the connection is aborted. This option applies only to the Browser Azure AD plugin.

On macOS and Linux, edit the odbc.ini file as follows:

**Note**
All entries are case-insensitive.

- For **clusterid**, enter *your-cluster-identifier*. This is the name of the created Amazon Redshift cluster.
- For **region**, enter *your-cluster-region*. This is the AWS Region of the created Amazon Redshift cluster.
- For **database**, enter *your-database-name*. This is the name of the database that you're trying to access on the Amazon Redshift cluster.
- For **locale**, enter *en-us*. This is the language that error messages display in.
- For **iam**, enter *1*. This value specifies to the driver to authenticate using IAM credentials.
- For **plugin_name** for Azure AD SSO configuration, enter *AzureAD*. This specifies to the driver to use Azure Single Sign-On as the authentication method.
- For **plugin_name** for Azure AD SSO with MFA configuration, enter *BrowserAzureAD*. This specifies to the driver to use Azure Single Sign-On with MFA as the authentication method.
- For **uid**, enter *your-azure-username*. This is the user name of the Microsoft Azure account you are using for single sign-on that has permission to the cluster you are trying to authenticate against. Use this only for **plugin_name** is *AzureAD*.
- For **pwd**, enter *your-azure-password*. Use this only for **plugin_name** is *AzureAD*.
- For **idp_tenant**, enter *your-idp-tenant*. This is the tenant name of your company configured on your IdP (Azure). This value can either be the tenant name or the tenant unique ID with hyphens.
- For **client_secret**, enter *your-azure-redshift-application-client-secret*. This is the client secret of the Amazon Redshift application that you created when setting up your Azure single sign-on configuration.
- For **client_id**, enter *your-azure-redshift-application-client-id*. This is the client ID (with hyphens) of the Amazon Redshift application that you created when setting up your Azure single sign-on configuration.
- For **listen_port**, enter *your-listen-port*. This is the port that local server is listening to. The default is 7890. This applies to the Browser Azure AD plugin.
- For **idp_response_timeout**, enter *the-number-of-seconds*. This is the specified period of time in seconds to wait for response from Azure. This option applies to the Browser Azure AD plugin.

On macOS and Linux, also edit the profile settings to add the following exports.

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

**To troubleshoot issues with the Browser Azure AD plugin**

1. To use the Browser Azure AD plugin, you must set the reply URL specified in the request to match the reply URL configured for your application.
Using IAM authentication to generate database user credentials

Navigate to the Set up Single Sign-On with SAML page on the Microsoft Azure portal. Then check the Reply URL is set to http://localhost/redshift/.

2. If you get an IdP tenant error, verify that the IdP Tenant name matches the domain name you initially used to set up the Active Directory in Microsoft Azure.

On Windows, navigate to the Connection Settings section of the Amazon Redshift ODBC DSN Setup page. Then check the tenant name of your company configured on your IdP (Azure) matches the domain name you initially used to set up the Active Directory in Microsoft Azure.

On macOS and Linux, find the odbc.ini file. Then check the tenant name of your company configured on your IdP (Azure) matches the domain name you initially used to set up the Active Directory in Microsoft Azure.

3. If you get an error that the reply URL specified in the request does not match the reply URLs configured for your application, verify that the Redirect URIs is the same as the reply URL.

Navigate to the App registration page of your application on the Microsoft Azure portal. Then check the Redirect URIs matches the reply URL.

4. If you get the unexpected response: unauthorized error, verify that you completed the Mobile and desktop applications configuration.

Navigate to the App registration page of your application on the Microsoft Azure portal. Then navigate to Authentication and check that you configured Mobile and desktop applications to use http://localhost/redshift/ as the redirect URIs.

Setting up JDBC or ODBC Single Sign-on authentication with AD FS

You can use AD FS as an identity provider (IdP) to access your Amazon Redshift cluster. Following, you can find a procedure that describes how to set up a trust relationship for this purpose. For more information about configuring AWS as a service provider for AD FS, see Configuring Your SAML 2.0 IdP with Relying Party Trust and Adding Claims in the IAM User Guide.

To set up AD FS and your AWS account to trust each other

1. Create or use an existing Amazon Redshift cluster for your AD FS users to connect to. To configure the connection, certain properties of this cluster are needed, such as the cluster identifier. For more information, see Creating a Cluster.

2. Set up AD FS to control Amazon Redshift access on the Microsoft Management Console:

   2. On the Select Data Source page, choose Import data about the relying party published online or on a local network.
   3. For Federation metadata address (host name or URL), enter https://signin.aws.amazon.com/saml-metadata.xml. The metadata XML file is a standard SAML metadata document that describes AWS as a relying party.
   4. On the Specify Display Name page, enter a value for Display name.
   5. On the Choose Issuance Authorization Rules page, choose an issuance authorization rule to either permit or deny all users to access this relying party.
   6. On the Ready to Add Trust page, review your settings.
   7. On the Finish page, choose Open the Edit Claim Rules dialog for this relying party trust when the wizard closes.
   8. On the context (right-click) menu, choose Relying Party Trusts.
10. For **Claim rule template**, choose **Transform an Incoming Claim**, and then on the **Edit Rule – NameId** page, do the following:
   - For **Claim rule name**, enter **NameId**.
   - For **Incoming claim name**, choose **Windows Account Name**.
   - For **Outgoing claim name**, choose **Name ID**.
   - For **Outgoing name ID format**, choose **Persistent Identifier**.
   - Choose **Pass through all claim values**.

11. On the **Edit Claim Rules** page, choose **Add Rule**. On the **Select Rule Template** page, for **Claim rule template**, choose **Send LDAP Attributes as Claims**.

12. On the **Configure Rule** page, do the following:
   - For **Claim rule name**, enter **RoleSessionName**.
   - For **Attribute store**, choose **Active Directory**.
   - For **LDAP Attribute**, choose **Email Addresses**.
   - For **Outgoing Claim Type**, choose **https://aws.amazon.com/SAML/Attributes/RoleSessionName**.

13. On the **Edit Claim Rules** page, choose **Add Rule**. On the **Select Rule Template** page, for **Claim rule template**, choose **Send Claims Using a Custom Rule**.

14. On the **Edit Rule – Get AD Groups** page, for **Claim rule name**, enter **Get AD Groups**.

15. For **Custom rule**, enter the following.

```plaintext
C: [Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname", Issuer == "AD AUTHORITY"] => add(store = "Active Directory", types = ("http://temp/variable"), query = ";tokenGroups;{0} ", param = c.Value);
```

16. On the **Edit Claim Rules** page, choose **Add Rule**. On the **Select Rule Template** page, for **Claim rule template**, choose **Send Claims Using a Custom Rule**.

17. On the **Edit Rule – Roles** page, for **Claim rule name**, type **Roles**.

18. For **Custom rule**, enter the following.

```plaintext
```

Note the ARNs of the SAML provider and role to assume. In this example, `arn:aws:iam::123456789012:saml-provider/ADFS` is the ARN of the SAML provider and `arn:aws:iam::123456789012:role/ADFS-` is the ARN of the role.

3. Make sure that you have downloaded the `federationmetadata.xml` file. Check that the document contents do not have invalid characters. This is the metadata file you use when configuring the trust relationship with AWS.

4. Create an IAM SAML identity provider on the IAM console. The metadata document that you provide is the federation metadata XML file that you saved when you set up Azure Enterprise Application. For detailed steps, see Creating and Managing an IAM Identity Provider (Console) in the IAM User Guide.

5. Create an IAM role for SAML 2.0 federation on the IAM console. For detailed steps, see Creating a Role for SAML in the IAM User Guide.
6. Create an IAM policy that you can attach to the IAM role that you created for SAML 2.0 federation on the IAM console. For detailed steps, see Creating IAM Policies (Console) in the IAM User Guide. For an Azure AD example, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 385).

To set up JDBC for authentication to AD FS

- Configure your database client to connect to your cluster through JDBC using AD FS SSO.

You can use any client that uses a JDBC driver to connect using AD FS SSO or use a language like Java to connect using a script. For installation and configuration information, see Configuring a JDBC driver version 1.x connection (p. 170).

For example, you can use SQLWorkbench/J as the client. When you configure SQLWorkbench/J, the URL of your database uses the following format.

```
jdbc:redshift:iam://cluster-identifier:us-west-1/dev
```

If you use SQLWorkbench/J as the client, take the following steps:

a. Start SQL Workbench/J. In the Select Connection Profile page, add a Profile Group, for example ADFS.

b. For Connection Profile, enter your connection profile name, for example ADFS.

c. Choose Manage Drivers, and choose Amazon Redshift. Choose the Open Folder icon next to Library, then choose the appropriate JDBC .jar file.

d. On the Select Connection Profile page, add information to the connection profile as follows:

   - For User, enter your AD FS user name. This is the user name of the account that you are using for single sign-on that has permission to the cluster that you are trying to authenticate using.

   - For Password, enter your AD FS password.

   - For Drivers, choose Amazon Redshift (com.amazon.redshift.jdbc.Driver).


   - Choose Extended Properties. For plugin_name, enter `com.amazon.redshift.plugin.AdfsCredentialsProvider`. This value specifies to the driver to use AD FS SSO as the authentication method.

To set up ODBC for authentication to AD FS

- Configure your database client to connect to your cluster through ODBC using AD FS SSO.

Amazon Redshift provides ODBC drivers for Linux, Windows, and macOS operating systems. Before you install an ODBC driver, determine whether your SQL client tool is 32-bit or 64-bit. Install the ODBC driver that matches the requirements of your SQL client tool.

Also install and configure the latest Amazon Redshift OBDC driver for your operating system as follows:

- For Windows, see Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180).

- For macOS, see Install the Amazon Redshift ODBC driver on macOS X (p. 183).

- For Linux, see Install the Amazon Redshift ODBC driver on Linux (p. 182).
On Windows, in the Amazon Redshift ODBC Driver DSN Setup page, under Connection Settings, enter the following information:

- For Data Source Name, enter your-DSN. This specifies the data source name used as the ODBC profile name.
- For Auth type, choose Identity Provider: SAML. This is the authentication method that the ODBC driver uses to authenticate using AD FS SSO.
- For Cluster ID, enter your-cluster-identifier.
- For Region, enter your-cluster-region.
- For Database, enter your-database-name.
- For User, enter your-adfs-username. This is the user name for the AD FS account that you are using for SSO that has permission to the cluster that you're trying to authenticate using. Use this only for Auth type is Identity Provider: SAML.
- For Password, enter your-adfs-password. Use this only for Auth type is Identity Provider: SAML.

On macOS and Linux, edit the odbc.ini file as follows:

**Note**
All entries are case-insensitive.

- For clusterid, enter your-cluster-identifier. This is the name of the created Amazon Redshift cluster.
- For region, enter your-cluster-region. This is the AWS Region of the created Amazon Redshift cluster.
- For database, enter your-database-name. This is the name of the database that you're trying to access on the Amazon Redshift cluster.
- For locale, enter en-us. This is the language that error messages display in.
- For iam, enter 1. This value specifies to the driver to authenticate using IAM credentials.
- For plugin_name, do one of the following:
  - For AD FS SSO with MFA configuration, enter BrowserSAML. This is the authentication method that the ODBC driver uses to authenticate to AD FS.
  - For AD FS SSO configuration, enter ADFS. This is the authentication method that the ODBC driver uses to authenticate using Azure AD SSO.
  - For uid, enter your-adfs-username. This is the user name of the Microsoft Azure account that you are using for SSO that has permission to the cluster you are trying to authenticate against. Use this only for plugin_name is ADFS.
- For pwd, enter your-adfs-password. Use this only for plugin_name is ADFS.

On macOS and Linux, also edit the profile settings to add the following exports.

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

**Setting up JDBC or ODBC Single Sign-on authentication with Ping Identity**

You can use Ping Identity as an identity provider (IdP) to access your Amazon Redshift cluster. Following, you can find a procedure that describes how to set up a trust relationship for this purpose using the
PingOne portal. For more information about configuring AWS as a service provider for Ping Identity, see Configuring Your SAML 2.0 IdP with Relying Party Trust and Adding Claims in the IAM User Guide.

To set up Ping Identity and your AWS account to trust each other

1. Create or use an existing Amazon Redshift cluster for your Ping Identity users to connect to. To configure the connection, certain properties of this cluster are needed, such as the cluster identifier. For more information, see Creating a Cluster.
2. Add Amazon Redshift as a new SAML application on the PingOne portal. For detailed steps, see the Ping Identity documentation.

   1. Go to My Applications.
   2. Under Add Application, choose New SAML Application.
   3. For Application Name, enter Amazon Redshift.
   4. For Protocol Version, choose SAML v2.0.
   5. For Category, choose your-application-category.
   6. For Assertion Consumer Service (ACS), type your-redshift-local-host-url. This is the local host and port that the SAML assertion redirects to.
   7. For Entity ID, enter urn:amazon:webservices.
   8. For Signing, choose Sign Assertion.
   9. In the SSO Attribute Mapping section, create the claims as shown in the following table.

<table>
<thead>
<tr>
<th>Application attribute</th>
<th>Identity bridge attribute of literal value</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://aws.amazon.com/SAML/Attributes/Role">https://aws.amazon.com/SAML/Attributes/Role</a></td>
<td>arn:aws:iam::123456789012:role/Ping,arn:aws:iam::123456789012:saml-provider/PingProvider</td>
</tr>
<tr>
<td><a href="https://aws.amazon.com/SAML/Attributes/RoleSessionName">https://aws.amazon.com/SAML/Attributes/RoleSessionName</a></td>
<td>email</td>
</tr>
<tr>
<td><a href="https://redshift.amazonaws.com/SAML/Attributes/AutoCreate">https://redshift.amazonaws.com/SAML/Attributes/AutoCreate</a></td>
<td>&quot;true&quot;</td>
</tr>
<tr>
<td><a href="https://redshift.amazonaws.com/SAML/Attributes/DbUser">https://redshift.amazonaws.com/SAML/Attributes/DbUser</a></td>
<td>email</td>
</tr>
<tr>
<td><a href="https://redshift.amazonaws.com/SAML/Attributes/DbGroups">https://redshift.amazonaws.com/SAML/Attributes/DbGroups</a></td>
<td>The groups in the &quot;DbGroups&quot; attributes contain the @directory prefix. To remove this, in Identity bridge, enter memberOf. In Function, choose ExtractByRegularExpression. In Expression, enter (.*)<a href="?:.*">@</a></td>
</tr>
</tbody>
</table>

3. For Group Access, set up the following group access, if needed:
   - https://aws.amazon.com/SAML/Attributes/Role
   - https://aws.amazon.com/SAML/Attributes/RoleSessionName
   - https://redshift.amazonaws.com/SAML/Attributes/AutoCreate
   - https://redshift.amazonaws.com/SAML/Attributes/DbUser
4. Review your setup and make changes, if necessary.
5. Use the Initiate Single Sign-On (SSO) URL as the login URL for the Browser SAML plugin.
6. Create an IAM SAML identity provider on the IAM console. The metadata document that you provide is the federation metadata XML file that you saved when you set up Ping Identity. For detailed steps, see Creating and Managing an IAM Identity Provider (Console) in the IAM User Guide.
7. Create an IAM role for SAML 2.0 federation on the IAM console. For detailed steps, see Creating a Role for SAML in the IAM User Guide.

8. Create an IAM policy that you can attach to the IAM role that you created for SAML 2.0 federation on the IAM console. For detailed steps, see Creating IAM Policies (Console) in the IAM User Guide. For an Azure AD example, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 385).

To set up JDBC for authentication to Ping Identity

- Configure your database client to connect to your cluster through JDBC using Ping Identity SSO.

You can use any client that uses a JDBC driver to connect using Ping Identity SSO or use a language like Java to connect using a script. For installation and configuration information, see Configuring a JDBC driver version 1.x connection (p. 170).

For example, you can use SQLWorkbench/J as the client. When you configure SQLWorkbench/J, the URL of your database uses the following format.

```
jdbc:redshift:iam://cluster-identifier:us-west-1/dev
```

If you use SQLWorkbench/J as the client, take the following steps:

a. Start SQL Workbench/J. In the Select Connection Profile page, add a Profile Group, for example Ping.

b. For Connection Profile, enter your-connection-profile-name, for example Ping.

c. Choose Manage Drivers, and choose Amazon Redshift. Choose the Open Folder icon next to Library, then choose the appropriate JDBC .jar file.

d. On the Select Connection Profile page, add information to the connection profile as follows:

   - For User, enter your PingOne user name. This is the user name of the PingOne account that you are using for single sign-on that has permission to the cluster that you are trying to authenticate using.
   - For Password, enter your PingOne password.
   - For Drivers, choose Amazon Redshift (com.amazon.redshift.jdbc.Driver).

e. Choose Extended Properties and do one of the following:

   - For login_url, enter your-ping-sso-login-url. This value specifies to the URL to use SSO as the authentication to log in.
   - For Ping Identity, for plugin_name, enter com.amazon.redshift.plugin.PingCredentialsProvider. This value specifies to the driver to use Ping Identity SSO as the authentication method.
   - For Ping Identity with SSO, for plugin_name, enter com.amazon.redshift.plugin.BrowserSamlCredentialsProvider. This value specifies to the driver to use Ping Identity PingOne with SSO as the authentication method.

To set up ODBC for authentication to Ping Identity

- Configure your database client to connect to your cluster through ODBC using Ping Identity PingOne SSO.
Using IAM authentication to generate database user credentials

Amazon Redshift provides ODBC drivers for Linux, Windows, and macOS operating systems. Before you install an ODBC driver, determine whether your SQL client tool is 32-bit or 64-bit. Install the ODBC driver that matches the requirements of your SQL client tool.

Also install and configure the latest Amazon Redshift OBDC driver for your operating system as follows:

- For Windows, see Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180).
- For macOS, see Install the Amazon Redshift ODBC driver on macOS X (p. 183).
- For Linux, see Install the Amazon Redshift ODBC driver on Linux (p. 182).

On Windows, in the Amazon Redshift ODBC Driver DSN Setup page, under Connection Settings, enter the following information:

- For Data Source Name, enter your-DSN. This specifies the data source name used as the ODBC profile name.
- For Auth type, do one of the following:
  - For Ping Identity configuration, choose Identity Provider: Ping Federate. This is the authentication method that the ODBC driver uses to authenticate using Ping Identity SSO.
  - For Ping Identity with SSO configuration, choose Identity Provider: Browser SAML. This is the authentication method that the ODBC driver uses to authenticate using Ping Identity with SSO.
- For Cluster ID, enter your-cluster-identifier.
- For Region, enter your-cluster-region.
- For Database, enter your-database-name.
- For User, enter your-ping-username. This is the user name for the PingOne account that you are using for SSO that has permission to the cluster that you're trying to authenticate using. Use this only for Auth type is Identity Provider: PingFederate.
- For Password, enter your-ping-password. Use this only for Auth type is Identity Provider: PingFederate.
- For Listen Port, enter your-listen-port. This is the port that local server is listening to. The default is 7890. This applies only to the Browser SAML plugin.
- For Response Timeout, enter the-number-of-seconds. This is the number of seconds to wait before timing out when the IdP server sends back a response. The minimum number of seconds must be 10. If establishing the connection takes longer than this threshold, then the connection is aborted. This applies only to the Browser SAML plugin.
- For Login URL, enter your-login-url. This applies only to the Browser SAML plugin.

On macOS and Linux, edit the odbc.ini file as follows:

Note
All entries are case-insensitive.

- For clusterid, enter your-cluster-identifier. This is the name of the created Amazon Redshift cluster.
- For region, enter your-cluster-region. This is the AWS Region of the created Amazon Redshift cluster.
- For database, enter your-database-name. This is the name of the database that you're trying to access on the Amazon Redshift cluster.
- For locale, enter en-us. This is the language that error messages display in.
- For iam, enter 1. This value specifies to the driver to authenticate using IAM credentials.
Using IAM authentication to generate database user credentials

- For `plugin_name`, do one of the following:
  - For Ping Identity configuration, enter `BrowserSAML`. This is the authentication method that the ODBC driver uses to authenticate to Ping Identity.
  - For Ping Identity with SSO configuration, enter `Ping`. This is the authentication method that the ODBC driver uses to authenticate using Ping Identity with SSO.
  - For `uid`, enter `your-ping-username`. This is the user name of the Microsoft Azure account you are using for SSO that has permission to the cluster you are trying to authenticate against. Use this only for `plugin_name` is `Ping`.
  - For `pwd`, enter `your-ping-password`. Use this only for `plugin_name` is `Ping`.
  - For `login_url`, enter `your-login-url`. This is the Initiate SSO URL that returns the SAML Response. This applies only to the Browser SAML plugin.
  - For `idp_response_timeout`, enter `the-number-of-seconds`. This is the specified period of time in seconds to wait for response from PingOne Identity. This applies only to the Browser SAML plugin.
  - For `listen_port`, enter `your-listen-port`. This is the port that local server is listening to. The default is 7890. This applies only to the Browser SAML plugin.

On macOS and Linux, also edit the profile settings to add the following exports.

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

### Setting up JDBC or ODBC Single Sign-on authentication with Okta

You can use Okta as an identity provider (IdP) to access your Amazon Redshift cluster. Following, you can find a procedure that describes how to set up a trust relationship for this purpose. For more information about configuring AWS as a service provider for Okta, see Configuring Your SAML 2.0 IdP with Relying Party Trust and Adding Claims in the IAM User Guide.

**To set up Okta and your AWS account to trust each other**

1. Create or use an existing Amazon Redshift cluster for your Okta users to connect to. To configure the connection, certain properties of this cluster are needed, such as the cluster identifier. For more information, see Creating a Cluster.
2. Add Amazon Redshift as a new application on the Okta portal. For detailed steps, see the Okta documentation.
   - Choose Add Application.
   - Under Add Application, choose Create New App.
   - On the Create a New Add Application Integration page, for Platform, choose Web.
   - For Sign on method, choose SAML v2.0.
   - On the General Settings page, for App name, enter `your-redshift-saml-sso-name`. This is the name of your application.
   - On the SAML Settings page, for Single sign on URL, enter `your-redshift-local-host-url`. This is the local host and port that the SAML assertion redirects to, for example `http://localhost:7890/redshift/`.
3. Use the Single sign on URL value as the Recipient URL and Destination URL.
4. For Signing, choose Sign Assertion.
5. For **Audience URI (SP Entity ID)**, enter `urn:amazon:webservices` for the claims, as shown in the following table.

6. In the **Advanced Settings** section, for **SAML Issuer ID**, enter `your-Identity-Provider-Issuer-ID`, which you can find in the **View Setup Instructions** section.

7. In the **Attribute Statements** section, create the claims as shown in the following table.

<table>
<thead>
<tr>
<th>Claim name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>https://aws.amazon.com/SAML/Attributes/RoleSessionName</code></td>
<td><code>user.email</code></td>
</tr>
<tr>
<td><code>https://redshift.amazon.com/SAML/Attributes/AutoCreate</code></td>
<td>&quot;true&quot;</td>
</tr>
<tr>
<td><code>https://redshift.amazon.com/SAML/Attributes/DbUser</code></td>
<td><code>user.email</code></td>
</tr>
</tbody>
</table>

8. In the **App Embed Link** section, find the URL that you can use as the login URL for the Browser SAML plugin.

9. Create an IAM SAML identity provider on the IAM console. The metadata document that you provide is the federation metadata XML file that you saved when you set up Okta. For detailed steps, see Creating and Managing an IAM Identity Provider (Console) in the **IAM User Guide**.

10. Create an IAM role for SAML 2.0 federation on the IAM console. For detailed steps, see Creating a Role for SAML in the **IAM User Guide**.

11. Create an IAM policy that you can attach to the IAM role that you created for SAML 2.0 federation on the IAM console. For detailed steps, see Creating IAM Policies (Console) in the **IAM User Guide**. For an Azure AD example, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 385).

**To set up JDBC for authentication to Okta**

- Configure your database client to connect to your cluster through JDBC using Okta SSO.

You can use any client that uses a JDBC driver to connect using Okta SSO or use a language like Java to connect using a script. For installation and configuration information, see Configuring a JDBC driver version 1.x connection (p. 170).

For example, you can use SQLWorkbench/J as the client. When you configure SQLWorkbench/J, the URL of your database uses the following format.

```
jdbc:redshift:iam://cluster-identifier:us-west-1/dev
```

If you use SQLWorkbench/J as the client, take the following steps:

a. Start SQL Workbench/J. In the **Select Connection Profile** page, add a **Profile Group**, for example **Okta**.

b. For **Connection Profile**, enter `your-connection-profile-name`, for example **Okta**.

c. Choose **Manage Drivers**, and choose **Amazon Redshift**. Choose the **Open Folder** icon next to **Library**, then choose the appropriate JDBC .jar file.

d. On the **Select Connection Profile** page, add information to the connection profile as follows:
Using IAM authentication to generate database user credentials

- For **User**, enter your Okta user name. This is the user name of the Okta account that you are using for single sign-on that has permission to the cluster that you are trying to authenticate using.
- For **Password**, enter your Okta password.
- For **Drivers**, choose **Amazon Redshift** (com.amazon.redshift.jdbc.Driver).

**e.** Choose **Extended Properties** and do one of the following:

- For **login_url**, enter `your-okta-sso-login-url`. This value specifies to the URL to use SSO as the authentication to log in to Okta.
- For Okta SSO, for **plugin_name**, enter `com.amazon.redshift.plugin.OktaCredentialsProvider`. This value specifies to the driver to use Okta SSO as the authentication method.
- For Okta SSO with MFA, for **plugin_name**, enter `com.amazon.redshift.plugin.BrowserSamlCredentialsProvider`. This value specifies to the driver to use Okta SSO with MFA as the authentication method.

**To set up ODBC for authentication to Okta**

- Configure your database client to connect to your cluster through ODBC using Okta SSO.

Amazon Redshift provides ODBC drivers for Linux, Windows, and macOS operating systems. Before you install an ODBC driver, determine whether your SQL client tool is 32-bit or 64-bit. Install the ODBC driver that matches the requirements of your SQL client tool.

Also install and configure the latest Amazon Redshift ODBC driver for your operating system as follows:

- For Windows, see [Install and configure the Amazon Redshift ODBC driver on Microsoft Windows](p. 180).
- For macOS, see [Install the Amazon Redshift ODBC driver on macOS X](p. 183).
- For Linux, see [Install the Amazon Redshift ODBC driver on Linux](p. 182).

On Windows, in the **Amazon Redshift ODBC Driver DSN Setup** page, under **Connection Settings**, enter the following information:

- For **Data Source Name**, enter `your-DSN`. This specifies the data source name used as the ODBC profile name.
- For **Auth type**, do one of the following:
  - For Okta SSO configuration, choose **Identity Provider: Okta**. This is the authentication method that the ODBC driver uses to authenticate using Okta SSO.
  - For Okta SSO with MFA configuration, choose **Identity Provider: Browser SAML**. This is the authentication method that the ODBC driver uses to authenticate using Okta SSO with MFA.
- For **Cluster ID**, enter `your-cluster-identifier`.
- For **Region**, enter `your-cluster-region`.
- For **Database**, enter `your-database-name`.
- For **User**, enter `your-okta-username`. This is the user name for the Okta account that you are using for SSO that has permission to the cluster that you're trying to authenticate using. Use this only for **Auth type** is **Identity Provider: Okta**.
- For **Password**, enter `your-okta-password`. Use this only for **Auth type** is **Identity Provider: Okta**.
On macOS and Linux, edit the `odbc.ini` file as follows:

**Note**

- All entries are case-insensitive.

- For `clusterid`, enter `your-cluster-identifier`. This is the name of the created Amazon Redshift cluster.
- For `region`, enter `your-cluster-region`. This is the AWS Region of the created Amazon Redshift cluster.
- For `database`, enter `your-database-name`. This is the name of the database that you're trying to access on the Amazon Redshift cluster.
- For `locale`, enter `en-us`. This is the language that error messages display in.
- For `iam`, enter `1`. This value specifies to the driver to authenticate using IAM credentials.
- For `plugin_name`, do one of the following:
  - For Okta SSO with MFA configuration, enter `BrowserSAML`. This is the authentication method that the ODBC driver uses to authenticate to Okta SSO with MFA.
  - For Okta SSO configuration, enter `Okta`. This is the authentication method that the ODBC driver uses to authenticate using Okta SSO.
  - For `uid`, enter `your-okta-username`. This is the user name of the Okta account you are using for SSO that has permission to the cluster you are trying to authenticate against. Use this only for `plugin_name` is `Okta`.
  - For `pwd`, enter `your-okta-password`. Use this only for `plugin_name` is `Okta`.
  - For `login_url`, enter `your-login-url`. This is the Initiate SSO URL that returns the SAML Response. This applies only to the Browser SAML plugin.
  - For `idp_response_timeout`, enter `the-number-of-seconds`. This is the specified period of time in seconds to wait for response from PingOne. This applies only to the Browser SAML plugin.
  - For `listen_port`, enter `your-listen-port`. This is the port that local server is listening to. The default is 7890. This applies only to the Browser SAML plugin.

On macOS and Linux, also edit the profile settings to add the following exports.

```
export ODBCINI=/opt/amazon/redshift/Setup/odbc.ini
```

```
export ODBCINSTINI=/opt/amazon/redshift/Setup/odbcinst.ini
```

**JDBC and ODBC Options for Creating Database User Credentials**

To use the Amazon Redshift JDBC or ODBC driver to create database user credentials, provide the database user name as a JDBC or ODBC option. Optionally, you can have the driver create a new database user if one doesn’t exist, and you can specify a list of database user groups the user joins at login.

If you use an identity provider (IdP), work with your IdP administrator to determine the correct values for these options. Your IdP administrator can also configure your IdP to provide these options, in which case you don’t need to provide them as JDBC or ODBC options. For more information, see Configure SAML assertions for your IdP (p. 372).

**Note**

If you use an IAM policy variable `$(redshift:DbUser)`, as described in Resource policies for `GetClusterCredentials` (p. 358) the value for `DbUser` is replaced with the value retrieved by
the API operation’s request context. The Amazon Redshift drivers use the value for the `DbUser` variable provided by the connection URL, rather than the value supplied as a SAML attribute.

To help secure this configuration, we recommend that you use a condition in an IAM policy to validate the `DbUser` value with the `RoleSessionName`. You can find examples of how to set a condition using an IAM policy in Example policy for using GetClusterCredentials (p. 363).

The following table lists the options for creating database user credentials.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DbUser</td>
<td>The name of a database user. If a user named DbUser exists in the database, the temporary user credentials have the same permissions as the existing user. If DbUser doesn't exist in the database and AutoCreate is true, a new user named DbUser is created. Optionally, disable the password for an existing user. For more information, see ALTER_USER.</td>
</tr>
<tr>
<td>AutoCreate</td>
<td>Specify true to create a database user with the name specified for DbUser if one does not exist. The default is false.</td>
</tr>
<tr>
<td>DbGroups</td>
<td>A comma-delimited list of the names of one or more existing database groups the database user joins for the current session. By default, the new user is added only to PUBLIC.</td>
</tr>
</tbody>
</table>

Generating IAM database credentials using the Amazon Redshift CLI or API

To programmatically generate temporary database user credentials, Amazon Redshift provides the `get-cluster-credentials` command for the AWS Command Line Interface (AWS CLI) and the `GetClusterCredentials` API operation. Or you can configure your SQL client with Amazon Redshift JDBC or ODBC drivers that manage the process of calling the `GetClusterCredentials` operation, retrieving the database user credentials, and establishing a connection between your SQL client and your Amazon Redshift database. For more information, see JDBC and ODBC Options for Creating Database User Credentials (p. 402).

**Note**

We recommend using the Amazon Redshift JDBC or ODBC drivers to generate database user credentials.

In this section, you can find steps to programmatically call the `GetClusterCredentials` operation or `get-cluster-credentials` command, retrieve database user credentials, and connect to the database.

To generate and use temporary database credentials

1. Create or modify an IAM user or role with the required permissions. For more information about IAM permissions, see Create an IAM role or user role or user with permissions to call GetClusterCredentials (p. 373).
2. As an IAM user or role you authorized in the previous step, run the `get-cluster-credentials` CLI command or call the `GetClusterCredentials` API operation and provide the following values:
   - **Cluster identifier** – The name of the cluster that contains the database.
   - **Database user name** – The name of an existing or new database user.
     - If the user doesn't exist in the database and AutoCreate is true, a new user is created with PASSWORD disabled.
     - If the user doesn't exist, and AutoCreate is false, the request fails.
     - For this example, the database user name is `temp_creds_user`.
   - **Autocreate** – (Optional) Create a new user if the database user name doesn't exist.
• **Database name** – (Optional) The name of the database that the user is authorized to log on to. If database name isn't specified, the user can log on to any cluster database.

• **Database groups** – (Optional) A list of existing database user groups. Upon successful login, the database user is added to the specified user groups. If no group is specified, the user has only PUBLIC permissions. The user group names must match the dbgroup resources ARNs specified in the IAM policy attached to the IAM user or role.

• **Expiration time** – (Optional) The time, in seconds, until the temporary credentials expire. You can specify a value between 900 seconds (15 minutes) and 3600 seconds (60 minutes). The default is 900 seconds.

3. Amazon Redshift verifies that the IAM user has permission to call the `GetClusterCredentials` operation with the specified resources.

4. Amazon Redshift returns a temporary password and the database user name.

   The following example uses the Amazon Redshift CLI to generate temporary database credentials for an existing user named `temp_creds_user`.

   ```bash
   aws redshift get-cluster-credentials --cluster-identifier examplecluster --db-user temp_creds_user --db-name exampledb --duration-seconds 3600
   ```

   The result is as follows.

   ```json
   {
   "DbUser": "IAM:temp_creds_user",
   "Expiration": "2016-12-08T21:12:53Z",
   "DbPassword": "EXAMPLEjArE3hcnQj8zt4XQj9Xtma8oxYEM80yxpDHWxXVPyJYBDm/gqX2Beaq6F3DgTzgPgs="
   }
   ```

   The following example uses the Amazon Redshift CLI with autocreate to generate temporary database credentials for a new user and add the user to the group `example_group`.

   ```bash
   aws redshift get-cluster-credentials --cluster-identifier examplecluster --db-user temp_creds_user --auto-create --db-name exampledb --db-groups example_group --duration-seconds 3600
   ```

   The result is as follows.

   ```json
   {
   "DbUser": "IAMA:temp_creds_user:example_group",
   "Expiration": "2016-12-08T21:12:53Z",
   "DbPassword": "EXAMPLEjArE3hcnQj8zt4XQj9Xtma8oxYEM80yxpDHWxXVPyJYBDm/gqX2Beaq6F3DgTzgPgs="
   }
   ```

5. Establish a Secure Socket Layer (SSL) authentication connection with the Amazon Redshift cluster and send a login request with the user name and password from the `GetClusterCredentials` response. Include the IAM: or IAMA: prefix with the user name, for example IAM:temp_creds_user or IAMA:temp_creds_user.

   **Important**

   Configure your SQL client to require SSL. Otherwise, if your SQL client automatically tries to connect with SSL, it can fall back to non-SSL if there is any kind of failure. In that case, the first connection attempt might fail because the credentials are expired or invalid, then a second connection attempt fails because the connection is not SSL. If that occurs, the first error message might be missed. For more information about connecting to your cluster using SSL, see [Configuring security options for connections (p. 193)](https://aws.amazon.com/documentation/redshift/configure-security-options/).
6. If the connection doesn't use SSL, the connection attempt fails.
7. The cluster sends an authentication request to the SQL client.
8. The SQL client then sends the temporary password to the cluster.
9. If the password is valid and has not expired, the cluster completes the connection.

Authorizing Amazon Redshift to access other AWS services on your behalf

Some Amazon Redshift features require Amazon Redshift to access other AWS services on your behalf. For example, the COPY and UNLOAD commands can load or unload data into your Amazon Redshift cluster using an Amazon S3 bucket. The CREATE EXTERNAL FUNCTION command can invoke an AWS Lambda function using a scalar Lambda user-defined function (UDF). Amazon Redshift Spectrum can use a data catalog in Amazon Athena or AWS Glue. For your Amazon Redshift clusters to act on your behalf, you supply security credentials to your clusters. The preferred method to supply security credentials is to specify an AWS Identity and Access Management (IAM) role. For COPY and UNLOAD, you can provide AWS access keys.

Following, find out how to create an IAM role with the appropriate permissions to access other AWS services. You also need to associate the role with your cluster and specify the Amazon Resource Name (ARN) of the role when you run the Amazon Redshift command. For more information, see Authorizing COPY, UNLOAD, CREATE EXTERNAL FUNCTION, and CREATE EXTERNAL SCHEMA operations using IAM roles (p. 410).

In addition, a superuser can grant the ASSUMEROLE privilege to specific users and groups to provide access to a role for COPY and UNLOAD operations. For information, see GRANT in the Amazon Redshift Database Developer Guide.

Creating an IAM role to allow your Amazon Redshift cluster to access AWS services

To create an IAM role to permit your Amazon Redshift cluster to communicate with other AWS services on your behalf, take the following steps. The values used in this section are examples, you can choose values based on your needs.

To create an IAM role to allow Amazon Redshift to access AWS services

1. Open the IAM console.
2. In the navigation pane, choose Roles.
3. Choose Create role.
4. Choose AWS service, and then choose Redshift.
5. Under Select your use case, choose Redshift - Customizable and then choose Next: Permissions. The Attach permissions policy page appears.
6. For access to Amazon S3 using COPY, as an example, you can use AmazonS3ReadOnlyAccess and append. For access to Amazon S3 using COPY or UNLOAD, we suggest that you can create managed policies that restrict access to the desired bucket and prefix accordingly. For both read and write operations, we recommend enforcing the least privileges and restricting to only the Amazon S3 buckets and key prefixes that Amazon Redshift requires.

   For access to invoke Lambda functions for the CREATE EXTERNAL FUNCTION command, add AWSLambdaRole.

   For Redshift Spectrum, in addition to Amazon S3 access, add AWSGlueConsoleFullAccess or AmazonAthenaFullAccess.
Choose **Next: Tags**.

7. The **Add tags** page appears. You can optionally add tags. Choose **Next: Review**.

8. For **Role name**, type a name for your role, for example **RedshiftCopyUnload**. Choose **Create role**.

9. The new role is available to all users on clusters that use the role. To restrict access to only specific users on specific clusters, or to clusters in specific regions, edit the trust relationship for the role. For more information, see **Restricting access to IAM roles** (p. 406).

10. Associate the role with your cluster. You can associate an IAM role with a cluster when you create the cluster, or you add the role to an existing cluster. For more information, see **Associating IAM roles with clusters** (p. 410).

**Note**

To restrict access to specific data, use an IAM role that grants the least privileges required.

---

**Restricting access to IAM roles**

By default, IAM roles that are available to an Amazon Redshift cluster are available to all users on that cluster. You can choose to restrict IAM roles to specific Amazon Redshift database users on specific clusters or to specific regions.

To permit only specific database users to use an IAM role, take the following steps.

**To identify specific database users with access to an IAM role**

1. Identify the Amazon Resource Name (ARN) for the database users in your Amazon Redshift cluster. The ARN for a database user is in the format: `arn:aws:redshift:region:account-id:dbuser:cluster-name/user-name`.

2. Open the IAM console at url="https://console.aws.amazon.com/".

3. In the navigation pane, choose **Roles**.

4. Choose the IAM role that you want to restrict to specific Amazon Redshift database users.

5. Choose the **Trust Relationships** tab, and then choose **Edit Trust Relationship**. A new IAM role that allows Amazon Redshift to access other AWS services on your behalf has a trust relationship as follows:

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Principal": {
   "Service": "redshift.amazonaws.com"
   },
   "Action": "sts:AssumeRole"
   }
   ]
   }
   ```

6. Add a condition to the `sts:AssumeRole` action section of the trust relationship that limits the `sts:ExternalId` field to values that you specify. Include an ARN for each database user that you want to grant access to the role. The external ID can be any unique string.

   For example, the following trust relationship specifies that only database users user1 and user2 on cluster `my-cluster` in region `us-west-2` have permission to use this IAM role.

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   ```
Authorizing Amazon Redshift to access AWS services

```json
{
    "Effect": "Allow",
    "Principal": {
        "Service": "redshift.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringEquals": {
            "sts:ExternalId": [
            ]
        }
    }
}
```

7. Choose **Update Trust Policy**.

### Restricting an IAM role to an AWS Region

You can restrict an IAM role to only be accessible in a certain AWS Region. By default, IAM roles for Amazon Redshift are not restricted to any single region.

To restrict use of an IAM role by region, take the following steps.

**To identify permitted regions for an IAM role**

2. In the navigation pane, choose **Roles**.
3. Choose the role that you want to modify with specific regions.
4. Choose the **Trust Relationships** tab and then choose **Edit Trust Relationship**. A new IAM role that allows Amazon Redshift to access other AWS services on your behalf has a trust relationship as follows:

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

5. Modify the **Service** list for the **Principal** with the list of the specific regions that you want to permit use of the role for. Each region in the **Service** list must be in the following format: `redshift.<region>.amazonaws.com`.

For example, the following edited trust relationship permits the use of the IAM role in the `us-east-1` and `us-west-2` regions only.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.us-east-1.amazonaws.com",
                "Service": "redshift.us-west-2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```
Chaining IAM roles in Amazon Redshift

When you attach a role to your cluster, your cluster can assume that role to access Amazon S3, Amazon Athena, AWS Glue, and AWS Lambda on your behalf. If a role attached to your cluster doesn’t have access to the necessary resources, you can chain another role, possibly belonging to another account. Your cluster then temporarily assumes the chained role to access the data. You can also grant cross-account access by chaining roles. Each role in the chain assumes the next role in the chain, until the cluster assumes the role at the end of chain. The maximum number of IAM roles that you can associate is subject to a quota. For more information, see the quota “Cluster IAM roles for Amazon Redshift to access other AWS services” in Amazon Redshift quotas (p. 505).

For example, suppose Company A wants to access data in an Amazon S3 bucket that belongs to Company B. Company A creates an AWS service role for Amazon Redshift named RoleA and attaches it to their cluster. Company B creates a role named RoleB that’s authorized to access the data in the Company B bucket. To access the data in the Company B bucket, Company A runs a COPY command using an iam_role parameter that chains RoleA and RoleB. For the duration of the COPY operation, RoleA temporarily assumes RoleB to access the Amazon S3 bucket.

To chain roles, you establish a trust relationship between the roles. A role that assumes another role (for example, RoleA) must have a permissions policy that allows it to assume the next chained role (for example, RoleB). In turn, the role that passes permissions (RoleB) must have a trust policy that allows it to pass its permissions to the previous chained role (RoleA). For more information, see Using IAM roles in the IAM User Guide.

The first role in the chain must be a role attached to the cluster. The first role, and each subsequent role that assumes the next role in the chain, must have a policy that includes a specific statement. This statement has the Allow effect on the sts:AssumeRole action and the Amazon Resource Name (ARN) of the next role in a Resource element. In our example, RoleA has the following permission policy that allows it to assume RoleB, owned by AWS account 210987654321.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Stmt1487639602000",
         "Effect": "Allow",
         "Action": [
            "sts:AssumeRole"
         ],
         "Resource": "arn:aws:iam::210987654321:role/RoleB"
      }
   ]
}
```

A role that passes to another role must establish a trust relationship with the role that assumes the role or with the AWS account that owns the role. In our example, RoleB has the following trust policy to establish a trust relationship with RoleA.

```json
"Principal": {
   "Service": [
      "redshift.us-east-1.amazonaws.com",
      "redshift.us-west-2.amazonaws.com"
   ],
   "Action": "sts:AssumeRole"
}
```

6. Choose Update Trust Policy
The following trust policy establishes a trust relationship with the owner of RoleA, AWS account 123456789012.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/RoleA"
      }
    }
  ]
}
```

**Note**

To restrict role chaining authorization to specific users, define a condition. For more information, see [Restricting access to IAM roles](p. 406).

When you run an UNLOAD, COPY, CREATE EXTERNAL FUNCTION, or CREATE EXTERNAL SCHEMA command, you chain roles by including a comma-separated list of role ARNs in the `iam_role` parameter. The following shows the syntax for chaining roles in the `iam_role` parameter.

```sql
unload ('select * from venue limit 10')
to 's3://acmedata/redshift/venue_pipe_'
IAM_ROLE 'arn:aws:iam::<aws-account-id-1>:role/<role-name-1>[,arn:aws:iam::<aws-account-id-2>:role/<role-name-2>][,...]';
```

**Note**

The entire role chain is enclosed in single quotes and must not contain spaces.

In the following examples, RoleA is attached to the cluster belonging to AWS account 123456789012. RoleB, which belongs to account 210987654321, has permission to access the bucket named `s3://companyb/redshift/`. The following example chains RoleA and RoleB to UNLOAD data to the `s3://companyb/redshift/` bucket.

```sql
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';
```

The following example uses a COPY command to load the data that was unloaded in the previous example.

```sql
copy venue
from 's3://companyb/redshift/venue_pipe_'
```
In the following example, `CREATE EXTERNAL SCHEMA` uses chained roles to assume the role `RoleB`.

```sql
create external schema spectrumexample from data catalog
database 'exampledb' region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

In the following example, `CREATE EXTERNAL FUNCTION` uses chained roles to assume the role `RoleB`.

```sql
create external function lambda_example(varchar)
returns varchar
volatile
lambda 'exampleLambdaFunction'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

**Additional information**

For more information, see also Authorizing COPY, UNLOAD, CREATE EXTERNAL FUNCTION, and CREATE EXTERNAL SCHEMA operations using IAM roles (p. 410).

**Authorizing COPY, UNLOAD, CREATE EXTERNAL FUNCTION, and CREATE EXTERNAL SCHEMA operations using IAM roles**

You can use the `COPY` command to load (or import) data into Amazon Redshift and the `UNLOAD` command to unload (or export) data from Amazon Redshift. You can use the `CREATE EXTERNAL FUNCTION` command to create user-defined functions that invoke functions from AWS Lambda.

When you use Amazon Redshift Spectrum, you use the `CREATE EXTERNAL SCHEMA` command to specify the location of an Amazon S3 bucket that contains your data. When you run the `COPY`, `UNLOAD`, or `CREATE EXTERNAL SCHEMA` commands, you provide security credentials. These credentials authorize your Amazon Redshift cluster to read or write data to and from your target destination, such as an Amazon S3 bucket.

When you run the `CREATE EXTERNAL FUNCTION`, you provide security credentials using the IAM role parameter. These credentials authorize your Amazon Redshift cluster to invoke Lambda functions from AWS Lambda. The preferred method to supply security credentials is to specify an AWS Identity and Access Management (IAM) role. For `COPY` and `UNLOAD`, you can provide AWS access keys. For information about creating an IAM role, see Authorizing Amazon Redshift to access other AWS services on your behalf (p. 405).

The steps for using an IAM role are as follows:

- Create an IAM role for use with your Amazon Redshift cluster.
- Associate the IAM role with the cluster.
- Include the IAM role's ARN when you call the `COPY`, `UNLOAD`, `CREATE EXTERNAL SCHEMA`, or `CREATE EXTERNAL FUNCTION` command.

In this topic, you learn how to associate an IAM role with an Amazon Redshift cluster.

**Associating IAM roles with clusters**

After you have created an IAM role that authorizes Amazon Redshift to access other AWS services for you, you must associate that role with an Amazon Redshift cluster. You must do this before you can use the role to load or unload data.
Permissions required to associate an IAM role with a cluster

To associate an IAM role with a cluster, an IAM user must have `iam:PassRole` permission for that IAM role. This permission allows an administrator to restrict which IAM roles a user can associate with Amazon Redshift clusters.

The following example shows an IAM policy that can be attached to an IAM user that allows the user to take these actions:

- Get the details for all Amazon Redshift clusters owned by that user's account.
- Associate any of three IAM roles with either of two Amazon Redshift clusters.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "redshift:DescribeClusters",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "redshift:ModifyClusterIamRoles",
                "redshift:CreateCluster"
            ],
            "Resource": [
                "arn:aws:redshift:us-east-1:123456789012:cluster:my-second-redshift-cluster"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/MyRedshiftRole",
                "arn:aws:iam::123456789012:role/SecondRedshiftRole",
                "arn:aws:iam::123456789012:role/ThirdRedshiftRole"
            ]
        }
    ]
}
```

After an IAM user has the appropriate permissions, that user can associate an IAM role with an Amazon Redshift cluster. The IAM role is then ready to use with the COPY or UNLOAD command or other Amazon Redshift commands.

For more information on IAM policies, see Overview of IAM policies in the IAM User Guide.

Managing IAM role association with a cluster

You can associate an IAM role with an Amazon Redshift cluster when you create the cluster. Or you can modify an existing cluster and add or remove one or more IAM role associations.

Be aware of the following:

- The maximum number of IAM roles that you can associate is subject to a quota.
- An IAM role can be associated with multiple Amazon Redshift clusters.
• An IAM role can be associated with an Amazon Redshift cluster only if both the IAM role and the cluster are owned by the same AWS account.

Using the console to manage IAM role associations

You can manage IAM role associations for a cluster with the console by using the following procedure.

To manage IAM role associations

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to update.
3. For Actions, choose Manage IAM roles to display the current list IAM roles associated with the cluster.
4. On the Manage IAM roles page, choose the available IAM roles to add, and then choose Add IAM role.
5. Choose Done to save your changes.

Using the AWS CLI to manage IAM role associations

You can manage IAM role associations for a cluster with the AWS CLI by using the following approaches.

Associating an IAM role with a cluster using the AWS CLI

To associate an IAM role with a cluster when the cluster is created, specify the Amazon Resource Name (ARN) of the IAM role for the --iam-role-arns parameter of the create-cluster command. The maximum number of IAM roles that you can add when calling the create-cluster command is subject to a quota.

Associating and disassociating IAM roles with Amazon Redshift clusters is an asynchronous process. You can get the status of all IAM role cluster associations by calling the describe-clusters command.

The following example associates two IAM roles with the newly created cluster named my-redshift-cluster.

```bash
aws redshift create-cluster
  --cluster-identifier "my-redshift-cluster" 
  --node-type "dc1.large" 
  --number-of-nodes 16 
  --iam-role-arns "arn:aws:iam::123456789012:role/RedshiftCopyUnload" 
   "arn:aws:iam::123456789012:role/SecondRedshiftRole"
```

To associate an IAM role with an existing Amazon Redshift cluster, specify the Amazon Resource Name (ARN) of the IAM role for the --add-iam-roles parameter of the modify-cluster-iam-roles command. The maximum number of IAM roles that you can add when calling the modify-cluster-iam-roles command is subject to a quota.

The following example associates an IAM role with an existing cluster named my-redshift-cluster.

```bash
aws redshift modify-cluster-iam-roles
  --cluster-identifier "my-redshift-cluster" 
  --add-iam-roles "arn:aws:iam::123456789012:role/RedshiftCopyUnload"
```

Disassociating an IAM role from a cluster using the AWS CLI

To disassociate an IAM role from a cluster, specify the ARN of the IAM role for the --remove-iam-roles parameter of the modify-cluster-iam-roles command. modify-cluster-iam-roles The
maximum number of IAM roles that you can remove when calling the modify-cluster-iam-roles command is subject to a quota.

The following example removes the association for an IAM role for the 123456789012 AWS account from a cluster named my-redshift-cluster.

```
aws redshift modify-cluster-iam-roles \
  --cluster-identifier "my-redshift-cluster" \
  --remove-iam-roles "arn:aws:iam::123456789012:role/RedshiftCopyUnload"
```

Listing IAM role associations for a cluster using the AWS CLI

To list all of the IAM roles that are associated with an Amazon Redshift cluster, and the status of the IAM role association, call the describe-clusters command. The ARN for each IAM role associated with the cluster is returned in the IamRoles list as shown in the following example output.

Roles that have been associated with the cluster show a status of in-sync. Roles that are in the process of being associated with the cluster show a status of adding. Roles that are being disassociated from the cluster show a status of removing.

```json
{
  "Clusters": [
    {
      "ClusterIdentifier": "my-redshift-cluster",
      "NodeType": "dc1.large",
      "NumberOfNodes": 16,
      "IamRoles": [
        {
          "IamRoleArn": "arn:aws:iam::123456789012:role/MyRedshiftRole",
          "IamRoleApplyStatus": "in-sync"
        },
        {
          "IamRoleArn": "arn:aws:iam::123456789012:role/SecondRedshiftRole",
          "IamRoleApplyStatus": "in-sync"
        }
      ],
      ...
    },
    {
      "ClusterIdentifier": "my-second-redshift-cluster",
      "NodeType": "dc1.large",
      "NumberOfNodes": 10,
      "IamRoles": [
        {
          "IamRoleArn": "arn:aws:iam::123456789012:role/MyRedshiftRole",
          "IamRoleApplyStatus": "in-sync"
        },
        {
          "IamRoleArn": "arn:aws:iam::123456789012:role/SecondRedshiftRole",
          "IamRoleApplyStatus": "in-sync"
        },
        {
          "IamRoleArn": "arn:aws:iam::123456789012:role/ThirdRedshiftRole",
          "IamRoleApplyStatus": "in-sync"
        }
      ],
      ...
    }
  ]
}
```
For more information on using the AWS CLI, see AWS CLI User Guide.

Creating an IAM role as default for Amazon Redshift

When you create IAM roles through the Redshift console, Amazon Redshift programmatically creates the roles in your AWS account and automatically attaches existing AWS managed policies to them. This approach means that you can stay within the Redshift console and don't have to switch to the IAM console for role creation. For more granular control of permissions for an existing IAM role that was created in the Amazon Redshift console, you can attach a customized managed policy to the IAM role.

Overview of IAM roles created in the console

When you use the Amazon Redshift console to create IAM roles, Amazon Redshift tracks all IAM roles created through the console. Amazon Redshift preselects the most recent default IAM role for creating all new clusters and restoring clusters from snapshots.

You can create an IAM role through the console that has a policy with permissions to run SQL commands. These commands include COPY, UNLOAD, CREATE EXTERNAL FUNCTION, CREATE EXTERNAL TABLE, CREATE EXTERNAL SCHEMA, CREATE MODEL, or CREATE LIBRARY. Optionally, you can get more granular control of user access to your AWS resources by creating and attaching custom policies to the IAM role.

When you created an IAM role and set it as the default for the cluster using console, you don't have to provide the IAM role's Amazon Resource Name (ARN) to perform authentication and authorization.

Using the IAM roles created in the console

The IAM role that you create through the console for your cluster has the AmazonRedshiftAllCommandsFullAccess managed policy automatically attached. This IAM role allows Amazon Redshift to copy, unload, query, and analyze data for AWS resources in your IAM account. The managed policy provides access to COPY, UNLOAD, CREATE EXTERNAL FUNCTION, CREATE EXTERNAL SCHEMA, CREATE MODEL, and CREATE LIBRARY operations. The policy also grants permissions to run SELECT statements for related AWS services, such as Amazon S3, Amazon CloudWatch Logs, Amazon SageMaker, and AWS Glue.

The CREATE EXTERNAL FUNCTION, CREATE EXTERNAL SCHEMA, CREATE MODEL, and CREATE LIBRARY commands have a default keyword. For this keyword for these commands, Amazon Redshift uses the IAM role that is set as the default and associated with the cluster when the command runs. You can run the DEFAULT_IAM_ROLE command to check the current default IAM role that is attached to the cluster.

To control access privileges of the IAM role created and set as default for your Redshift cluster, use the ASSUMEROLE privilege. This access control applies to database users and groups when they run commands such as the ones listed preceding. After you grant the ASSUMEROLE privilege to a user or group for the IAM role, the user or group can assume that role when running these commands. By using the ASSUMEROLE privilege, you can grant access to the appropriate commands as required.

Using the Amazon Redshift console, you can do the following:

- Creating an IAM role as the default (p. 416)
- Removing IAM roles from your cluster (p. 417)
- Associating IAM roles with your cluster (p. 418)
- Setting an IAM role as the default (p. 418)
- Making an IAM role no longer default for your cluster (p. 418)

Permissions of the AmazonRedshiftAllCommandsFullAccess managed policy

The following example shows the permissions in the AmazonRedshiftAllCommandsFullAccess managed policy that allow certain actions for the IAM role that is set as default for your cluster. The IAM role with permission policies attached authorizes what a user or group can and can't do. Given
these permissions, you can run the COPY command from Amazon S3, run UNLOAD, and use the CREATE
MODEL command.

```
{
  "Effect": "Allow",
  "Action": [
    "s3:GetObject",
    "s3:GetBucketAcl",
    "s3:GetBucketCors",
    "s3:GetEncryptionConfiguration",
    "s3:GetBucketLocation",
    "s3:ListBucket",
    "s3:ListAllMyBuckets",
    "s3:ListMultipartUploadParts",
    "s3:ListBucketMultipartUploads",
    "s3:PutObject",
    "s3:PutBucketAcl",
    "s3:PutBucketCors",
    "s3:DeleteObject",
    "s3:AbortMultipartUpload",
    "s3:CreateBucket"
  ],
  "Resource": [
    "arn:aws:s3:::redshift-downloads",
    "arn:aws:s3:::redshift-downloads/**",
    "arn:aws:s3:::*redshift*",
    "arn:aws:s3:::*redshift/**"
  ]
}
```

The following example shows the permissions in the AmazonRedshiftAllCommandsFullAccess
managed policy that allow certain actions for the IAM role that is set as default for the cluster. The
IAM role with permission policies attached authorizes what a user or group can and can't do. Given the
following permissions, you can run the CREATE EXTERNAL FUNCTION command.

```
{
  "Action": [
    "lambda:InvokeFunction"
  ],
  "Resource": "arn:aws:lambda:*:*:function:*redshift*"
}
```

The following example shows the permissions in the AmazonRedshiftAllCommandsFullAccess
managed policy that allow certain actions for the IAM role that is set as default for the cluster. The
IAM role with permission policies attached authorizes what a user or group can and can't do. Given
the following permissions, you can run the CREATE EXTERNAL SCHEMA and CREATE EXTERNAL TABLE
commands needed for Amazon Redshift Spectrum.

```
{
  "Effect": "Allow",
  "Action": [
    "glue:CreateDatabase",
    "glue:DeleteDatabase",
    "glue:GetDatabase",
    "glue:GetDatabases",
    "glue:UpdateDatabase",
    "glue:GetTable",
    "glue:GetTables",
    "glue:GetTables",
```

415
Managing IAM roles created for a cluster using the console

To create, modify, and remove IAM roles created from the Amazon Redshift console, use the Clusters section in the console.

Creating an IAM role as the default

On the console, you can create an IAM role for your cluster that has the AmazonRedshiftAllCommandsFullAccess policy automatically attached. The new IAM role that you create allows Amazon Redshift to copy, load, query, and analyze data from Amazon resources in your IAM account.

There can only be one IAM role set as the default for the cluster. If you create another IAM role as the cluster default when an existing IAM role is currently assigned as the default, the new IAM role replaces the other one as default.
To create a new cluster and an IAM role set as the default for the new cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.
3. Choose Create cluster to create a cluster.
4. Follow the instructions on the console page to enter the properties for Cluster configuration.

Choose one of the following methods to size your cluster:

Note
The following step assumes an AWS Region that supports RA3 node types. For a list of AWS Regions that support RA3 node types, see Overview of RA3 node types in the Amazon Redshift Cluster Management Guide.

- If your AWS Region supports RA3 node types, choose either Production or Free trial to answer the question What are you planning to use this cluster for?
  - If your organization is eligible, you might be able to create a cluster under the Amazon Redshift free trial program. To do this, choose Free trial to create a configuration with the dc2.large node type. For more information about choosing a free trial, see Amazon Redshift free trial.
  - If you don't know how large to size your cluster, choose Help me choose. Doing this starts a sizing calculator that asks you questions about the size and query characteristics of the data that you plan to store in your data warehouse.

  If you know the required size of your cluster (that is, the node type and number of nodes), choose I'll choose. Then choose the Node type and number of Nodes to size your cluster.

5. (Optional) Choose Load sample data to load the sample data set to your Amazon Redshift cluster to start using the query editor to query data.

  If you are behind a firewall, the database port must be an open port that accepts inbound connections.

6. Follow the instructions on the console page to enter properties for Database configurations.

7. Under Cluster permissions, from Manage IAM roles, choose Create IAM role.

8. Specify an Amazon S3 bucket for the IAM role to access by choosing one of the following methods:

  - Choose No additional Amazon S3 bucket to create the IAM role without specifying specific Amazon S3 buckets.
  - Choose Any Amazon S3 bucket to allow users that have access to your Amazon Redshift cluster to also access any Amazon S3 bucket and its contents in your AWS account.
  - Choose Specific Amazon S3 buckets to specify one or more Amazon S3 buckets that the IAM role being created has permission to access. Then choose one or more Amazon S3 buckets from the table.

9. Choose Create IAM role as default. Amazon Redshift automatically creates and sets the IAM role as the default for your cluster.

10. Choose Create cluster to create the cluster. The cluster might take several minutes to be ready to use.

Removing IAM roles from your cluster

You can remove one or more IAM roles from your cluster.
To remove IAM roles from your cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.
3. Choose the cluster that you want to remove the IAM role from.
4. Under Cluster permissions, choose one or more IAM roles that you want to remove from the cluster.
5. From Manage IAM roles, choose Remove IAM roles.

Associating IAM roles with your cluster
You can associate one or more IAM roles with your cluster.

To associate IAM roles with your cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.
3. Choose the cluster that you want to associate IAM roles with.
4. Under Cluster permissions, choose one or more IAM roles that you want to associate with the cluster.
5. From Manage IAM roles, choose Associate IAM roles.
6. Choose one or more IAM roles to associate with your cluster.
7. Choose Associate IAM roles.

Setting an IAM role as the default
You can set an IAM role as the default for your cluster.

To make an IAM role the default for your cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.
3. Choose the cluster that you want to set a default IAM role for.
4. Under Cluster permissions, from Associated IAM roles, choose an IAM role that you want make as default for the cluster.
5. Under Set default, choose Make default.
6. When prompted, choose Set default to confirm making the specified IAM role as the default.

Making an IAM role no longer default for your cluster
You can make an IAM role no longer the default for your cluster.

To clear an IAM role as the default for your cluster
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Clusters**. The clusters for your account in the current AWS Region are listed. A subset of properties of each cluster is displayed in columns in the list.

3. Choose the cluster that you want to associate IAM roles with.

4. Under **Cluster permissions**, from **Associated IAM roles**, choose the default IAM role.

5. Under **Set default**, choose **Clear default**.

6. When prompted, choose **Clear default** to confirm clearing the specified IAM role as the default.

**Managing IAM roles created on the cluster using the AWS CLI**

You can manage IAM roles created on the cluster using the AWS CLI.

**To create an Amazon Redshift cluster with an IAM role set as default**

To create an Amazon Redshift cluster with an IAM role set as the default for the cluster, use the `aws redshift create-cluster` AWS CLI command.

The following AWS CLI command creates an Amazon Redshift cluster and the IAM role named `myrole1`. The AWS CLI command also sets `myrole1` as the default for the cluster.

```bash
aws redshift create-cluster \\
  --node-type dc2.large \\
  --number-of-nodes 2 \\
  --master-username adminuser \\
  --master-user-password TopSecret1 \\
  --cluster-identifier mycluster \\
  --iam-roles 'arn:aws:iam::012345678910:role/myrole1' 'arn:aws:iam::012345678910:role/myrole2' \\
  --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole1'
```

The following snippet is an example of the response.

```json
{
  "Cluster": {
    "ClusterIdentifier": "mycluster",
    "NodeType": "dc2.large",
    "MasterUsername": "adminuser",
    "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
    "IamRoles": [
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "ApplyStatus": "adding"
      },
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
        "ApplyStatus": "adding"
      }
    ]
  },
  ...
}
```

**To add one or more IAM roles to an Amazon Redshift cluster**

To add one or more IAM roles associated to the cluster, use the `aws redshift modify-cluster-iam-roles` AWS CLI command.

The following AWS CLI command adds `myrole3` and `myrole4` to the cluster.
aws redshift modify-cluster-iam-roles \
   --cluster-identifier mycluster \
   --add-iam-roles 'arn:aws:iam::012345678910:role/myrole3' \
   'arn:aws:iam::012345678910:role/myrole4'

The following snippet is an example of the response.

```json
{
   "Cluster": {
      "ClusterIdentifier": "mycluster",
      "NodeType": "dc2.large",
      "MasterUsername": "adminuser",
      "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
      "IamRoles": [
         {
            "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
            "ApplyStatus": "in-sync"
         },
         {
            "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
            "ApplyStatus": "in-sync"
         },
         {
            "IamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
            "ApplyStatus": "adding"
         },
         {
            "IamRoleArn": "arn:aws:iam::012345678910:role/myrole4",
            "ApplyStatus": "adding"
         }
      ],
   ...
}
```

To remove one or more IAM roles from an Amazon Redshift cluster

To remove one or more IAM roles associated to the cluster, use the `aws redshift modify-cluster-iam-roles` AWS CLI command.

The following AWS CLI command removes `myrole3` and `myrole4` from the cluster.

```
aws redshift modify-cluster-iam-roles \
   --cluster-identifier mycluster \
   --remove-iam-roles 'arn:aws:iam::012345678910:role/myrole3' \
   'arn:aws:iam::012345678910:role/myrole4'
```

The following snippet is an example of the response.

```json
{
   "Cluster": {
      "ClusterIdentifier": "mycluster",
      "NodeType": "dc2.large",
      "MasterUsername": "adminuser",
      "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
      "IamRoles": [
         {
            "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
            "ApplyStatus": "in-sync"
         },
         {
            "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
            "ApplyStatus": "in-sync"
         }
      ]
}
```
To set an associated IAM role as the default for the cluster

To set an associated IAM role as the default for the cluster, use the `aws redshift modify-cluster-iam-roles` AWS CLI command.

The following AWS CLI command sets `myrole2` as the default for the cluster.

```
aws redshift modify-cluster-iam-roles \
  --cluster-identifier mycluster \n  --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole2'
```

The following snippet is an example of the response.

```
{
  "Cluster": {
    "ClusterIdentifier": "mycluster",
    "NodeType": "dc2.large",
    "MasterUsername": "adminuser",
    "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
    "IamRoles": [
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "ApplyStatus": "in-sync"
      },
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
        "ApplyStatus": "in-sync"
      }
    ],
    "..., ...
  }
}
```

To set an unassociated IAM role as the default for the cluster

To set an unassociated IAM role as the default for the cluster, use the `aws redshift modify-cluster-iam-roles` AWS CLI command.

The following AWS CLI command adds `myrole2` to the Amazon Redshift cluster and sets it as the default for the cluster.

```
aws redshift modify-cluster-iam-roles \
  --cluster-identifier mycluster \n  --add-iam-roles 'arn:aws:iam::012345678910:role/myrole3' \
```

```
The following snippet is an example of the response.

```
{
  "Cluster": {
    "ClusterIdentifier": "mycluster",
    "NodeType": "dc2.large",
    "MasterUsername": "adminuser",
    "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
    "IamRoles": [
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "ApplyStatus": "in-sync"
      },
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
        "ApplyStatus": "in-sync"
      },
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole3",
        "ApplyStatus": "adding"
      }
    ],
    ...
  }
}
```

To restore a cluster from a snapshot and set an IAM role as the default for it

When you restore your cluster from a snapshot, you can either associate an existing IAM role or create a new one and set it as the default for the cluster.

To restore an Amazon Redshift cluster from a snapshot and set an IAM role as the cluster default, use the `aws redshift restore-from-cluster-snapshot` AWS CLI command.

The following AWS CLI command restores the cluster from a snapshot and sets `myrole2` as the default for the cluster.

```
aws redshift restore-from-cluster-snapshot \
  --cluster-identifier mycluster-clone \
  --snapshot-identifier my-snapshot-id \
  --iam-roles 'arn:aws:iam::012345678910:role/myrole1' 'arn:aws:iam::012345678910:role/myrole2' \
  --default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole1'
```

The following snippet is an example of the response.

```
{
  "Cluster": {
    "ClusterIdentifier": "mycluster-clone",
    "NodeType": "dc2.large",
    "MasterUsername": "adminuser",
    "DefaultIamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
    "IamRoles": [
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole1",
        "ApplyStatus": "adding"
      },
      {
        "IamRoleArn": "arn:aws:iam::012345678910:role/myrole2",
        ...
      }
    ],
    ...
  }
}
```
Logging and monitoring in Amazon Redshift

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon Redshift and your AWS solutions. You can collect monitoring data from all of the parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. AWS provides several tools for monitoring your Amazon Redshift resources and responding to potential incidents:

**Amazon CloudWatch Alarms**

Using Amazon CloudWatch alarms, you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, a notification is sent to an Amazon SNS topic or AWS Auto Scaling policy. CloudWatch alarms do not invoke actions because they are in a particular state. Rather the state must have changed and been maintained for a specified number of periods. For more information, see Managing alarms (p. 486). For a list of metrics, see Monitoring Amazon Redshift using CloudWatch metrics (p. 459).

**AWS CloudTrail Logs**

CloudTrail provides a record of API operations taken by a IAM user, role, or an AWS service in Amazon Redshift. Using the information collected by CloudTrail, you can determine the request that was made to Amazon Redshift, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging Amazon Redshift API calls with AWS CloudTrail (p. 429).

**Database audit logging**

Amazon Redshift logs information about connections and user activities in your database. These logs help you to monitor the database for security and troubleshooting purposes, a process called database auditing. The logs are stored in Amazon S3 buckets. These provide convenient access with data-security features for users who are responsible for monitoring activities in the database.

**Topics**

- Amazon Redshift logs (p. 423)
- Enabling logging (p. 426)
- Managing log files (p. 426)
- Troubleshooting Amazon Redshift audit logging (p. 429)
- Logging Amazon Redshift API calls with AWS CloudTrail (p. 429)
- Amazon Redshift account IDs in AWS CloudTrail logs (p. 434)
- Configuring auditing using the console (p. 436)
- Configuring logging by using the AWS CLI and Amazon Redshift API (p. 436)

**Amazon Redshift logs**

Amazon Redshift logs information in the following log files:
• **Connection log** – Logs authentication attempts, connections, and disconnections.

• **User log** – Logs information about changes to database user definitions.

• **User activity log** – Logs each query before it’s run on the database.

The connection and user logs are useful primarily for security purposes. You can use the connection log to monitor information about users connecting to the database and related connection information. This information might be their IP address, when they made the request, what type of authentication they used, and so on. You can use the user log to monitor changes to the definitions of database users.

The user activity log is useful primarily for troubleshooting purposes. It tracks information about the types of queries that both the users and the system perform in the database.

The connection log and user log both correspond to information that is stored in the system tables in your database. You can use the system tables to obtain the same information, but the log files provide an easier mechanism for retrieval and review. The log files rely on Amazon S3 permissions rather than database permissions to perform queries against the tables. Additionally, by viewing the information in log files rather than querying the system tables, you reduce any impact of interacting with the database.

**Note**

Log files are not as current as the base system log tables, `STL_USERLOG` and `STL_CONNECTION_LOG`. Records that are older than, but not including, the latest record are copied to log files.

### Connection log

Logs authentication attempts, and connections and disconnections. The following table describes the information in the connection log. For more information about these fields, see `STL_CONNECTION_LOG` in the *Amazon Redshift Database Developer Guide*.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>Connection or authentication event.</td>
</tr>
<tr>
<td>recordtime</td>
<td>Time the event occurred.</td>
</tr>
<tr>
<td>remotehost</td>
<td>Name or IP address of remote host.</td>
</tr>
<tr>
<td>remoteport</td>
<td>Port number for remote host.</td>
</tr>
<tr>
<td>pid</td>
<td>Process ID associated with the statement.</td>
</tr>
<tr>
<td>dbname</td>
<td>Database name.</td>
</tr>
<tr>
<td>username</td>
<td>User name.</td>
</tr>
<tr>
<td>authmethod</td>
<td>Authentication method.</td>
</tr>
<tr>
<td>duration</td>
<td>Duration of connection in microseconds.</td>
</tr>
<tr>
<td>sslversion</td>
<td>Secure Sockets Layer (SSL) version.</td>
</tr>
<tr>
<td>sslcipher</td>
<td>SSL cipher.</td>
</tr>
<tr>
<td>mtu</td>
<td>Maximum transmission unit (MTU).</td>
</tr>
<tr>
<td>sslcompression</td>
<td>SSL compression type.</td>
</tr>
<tr>
<td>sslexpansion</td>
<td>SSL expansion type.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>iamauthguid</td>
<td>The AWS Identity and Access Management (IAM) authentication ID for the AWS CloudTrail request.</td>
</tr>
<tr>
<td>application_name</td>
<td>The initial or updated name of the application for a session.</td>
</tr>
<tr>
<td>driver_version</td>
<td>The version of ODBC or JDBC driver that connects to your Amazon Redshift cluster from your third-party SQL client tools.</td>
</tr>
<tr>
<td>os_version</td>
<td>The version of the operating system that is on the client machine that connects to your Amazon Redshift cluster.</td>
</tr>
<tr>
<td>plugin_name</td>
<td>The name of the plugin used to connect to your Amazon Redshift cluster.</td>
</tr>
<tr>
<td>protocol_version</td>
<td>The internal protocol version that the Amazon Redshift driver uses when establishing its connection with the server.</td>
</tr>
<tr>
<td>sessionid</td>
<td>The globally unique identifier for the current session.</td>
</tr>
</tbody>
</table>

**User log**

Records details for the following changes to a database user:

- Create user
- Drop user
- Alter user (rename)
- Alter user (alter properties)

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userid</td>
<td>ID of user affected by the change.</td>
</tr>
<tr>
<td>username</td>
<td>User name of the user affected by the change.</td>
</tr>
<tr>
<td>oldusername</td>
<td>For a rename action, the original user name. For any other action, this field is empty.</td>
</tr>
<tr>
<td>action</td>
<td>Action that occurred. Valid values:</td>
</tr>
<tr>
<td></td>
<td>- Alter</td>
</tr>
<tr>
<td></td>
<td>- Create</td>
</tr>
<tr>
<td></td>
<td>- Drop</td>
</tr>
<tr>
<td></td>
<td>- Rename</td>
</tr>
<tr>
<td>usecreatedb</td>
<td>If true (1), indicates that the user has create database privileges.</td>
</tr>
<tr>
<td>usesuper</td>
<td>If true (1), indicates that the user is a superuser.</td>
</tr>
<tr>
<td>usecatupd</td>
<td>If true (1), indicates that the user can update system catalogs.</td>
</tr>
<tr>
<td>valuntil</td>
<td>Password expiration date.</td>
</tr>
<tr>
<td>pid</td>
<td>Process ID.</td>
</tr>
<tr>
<td>xid</td>
<td>Transaction ID.</td>
</tr>
</tbody>
</table>
**Database audit logging**

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordtime</td>
<td>Time in UTC that the query started.</td>
</tr>
</tbody>
</table>

**User activity log**

Logs each query before it is run on the database.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordtime</td>
<td>Time the event occurred.</td>
</tr>
<tr>
<td>db</td>
<td>Database name.</td>
</tr>
<tr>
<td>user</td>
<td>User name.</td>
</tr>
<tr>
<td>pid</td>
<td>Process ID associated with the statement.</td>
</tr>
<tr>
<td>userid</td>
<td>User ID.</td>
</tr>
<tr>
<td>xid</td>
<td>Transaction ID.</td>
</tr>
<tr>
<td>query</td>
<td>A prefix of LOG: followed by the text of the query, including newlines.</td>
</tr>
</tbody>
</table>

**Enabling logging**

Audit logging is not turned on by default in Amazon Redshift. When you turn on logging on your cluster, Amazon Redshift creates and uploads logs to Amazon S3 that capture data from the time audit logging is enabled to the present time. Each logging update is a continuation of the information that was already logged.

**Note**

Audit logging to Amazon S3 is an optional, manual process. When you enable logging on your cluster, you are enabling logging to Amazon S3 only. Logging to system tables is not optional and happens automatically for the cluster. For more information about logging to system tables, see System Tables Reference in the Amazon Redshift Database Developer Guide.

The connection log, user log, and user activity log are enabled together by using the AWS Management Console, the Amazon Redshift API Reference, or the AWS Command Line Interface (AWS CLI). For the user activity log, you must also enable the enable_user_activity_logging database parameter. If you enable only the audit logging feature, but not the associated parameter, the database audit logs log information for only the connection log and user log, but not for the user activity log. The enable_user_activity_logging parameter is not enabled (false) by default. You can set it to true to enable the user activity log. For more information, see Amazon Redshift parameter groups (p. 271).

**Note**

Currently, you can only use Amazon S3-managed keys (SSE-S3) encryption (AES-256) for audit logging.

**Managing log files**

The number and size of Amazon Redshift log files in Amazon S3 depends heavily on the activity in your cluster. If you have an active cluster that is generating a large number of logs, Amazon Redshift might generate the log files more frequently. You might have a series of log files for the same type of activity, such as having multiple connection logs within the same hour.
Because Amazon Redshift uses Amazon S3 to store logs, you incur charges for the storage that you use in Amazon S3. Before you configure logging, you should have a plan for how long you need to store the log files. As part of this, determine when the log files can either be deleted or archived based on your auditing needs. The plan that you create depends heavily on the type of data that you store, such as data subject to compliance or regulatory requirements. For more information about Amazon S3 pricing, go to Amazon Simple Storage Service (S3) Pricing.

Bucket permissions for Amazon Redshift audit logging

When you turn on logging, Amazon Redshift collects logging information and uploads it to log files stored in Amazon S3. You can use an existing bucket or a new bucket. Amazon Redshift requires the following IAM permissions to the bucket:

- s3:GetBucketAcl The service requires read permissions to the Amazon S3 bucket so it can identify the bucket owner.
- s3:PutObject The service requires put object permissions to upload the logs. Also, the IAM user or IAM role that enables logging must have s3:PutObject permission to the Amazon S3 bucket. Each time logs are uploaded, the service determines whether the current bucket owner matches the bucket owner at the time logging was enabled. If these owners don't match, you receive an error.

If, when you enable audit logging, you select the option to create a new bucket, correct permissions are applied to it. However, if you create your own bucket in Amazon S3, or use an existing bucket, make sure to add a bucket policy that includes the bucket name. Logs are delivered using service-principal credentials. For most AWS Regions, you add the Redshift service-principal name, redshift.amazonaws.com.

The bucket policy uses the following format. ServiceName and BucketName are placeholders for your own values. Also specify the associated actions and resources in the bucket policy.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Put bucket policy needed for audit logging",
            "Effect": "Allow",
            "Principal": {
                "Service": "ServiceName"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::BucketName",
                "arn:aws:s3:::BucketName/*"
            ]
        }
    ]
}
```

The following example is a bucket policy for the US East (N. Virginia) Region and a bucket named AuditLogs.

```json
{
    "Version": "2008-10-17",
    "Statement": [
        {
            "Sid": "Put bucket policy needed for audit logging",
            "Effect": "Allow",
            "Principal": {
```
Regions that aren't enabled by default, also known as "opt-in" regions, require a region-specific service principal name. For these, the service-principal name includes the region, in the format redshift.region.amazonaws.com. For example, redshift.ap-east-1.amazonaws.com for the Asia Pacific (Hong Kong) Region. For a list of the regions that aren't enabled by default, see Managing AWS Regions in the AWS General Reference.

**Note**
The region-specific service-principal name corresponds with the region where the cluster is located.

**Best practices for S3 bucket permissions**

When you give a third party access to your Amazon S3 buckets, make sure to consider security best practices. You can provide optional information in the bucket policy to designate the bucket owner. This way, the account owner can permit the role to be assumed only under specific circumstances, and avoid the confused-deputy problem. For more information, see The confused deputy problem in the IAM User Guide.

We recommend that you configure your bucket policy in a way to specify access granted to a service principal specifically on behalf of the bucket owner's (or their partner's) resources. The following example illustrates how you can configure your bucket to grant Amazon Redshift permission to upload logs into the bucket, by specifying the SourceArn, while blocking any other account from uploading log files. You can use either the SourceArn or SourceAccount to specify access.

```json
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "Put bucket policy needed for audit logging",
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": [
        "s3:PutObject",
        "s3:GetBucketAcl"
      ],
      "Resource": [
        "arn:aws:s3:::AuditLogs",
        "arn:aws:s3:::AuditLogs/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": "arn:aws:redshift:us-east-1:123456789012:cluster:my-cluster"
        }
      }
    }
  ]
}
```
When Redshift uploads log files to Amazon S3, large files can be uploaded in parts. If a multipart upload isn't successful, it's possible for parts of a file to remain in the Amazon S3 bucket. This can result in additional storage costs, so it's important to understand what occurs when a multipart upload fails. For a detailed explanation about multipart upload for audit logs, see Uploading and copying objects using multipart upload and Aborting a multipart upload.

For more information about creating S3 buckets and adding bucket policies, see Creating a Bucket and Editing Bucket Permissions in the Amazon Simple Storage Service User Guide.

**Bucket structure for Amazon Redshift audit logging**

By default, Amazon Redshift organizes the log files in the Amazon S3 bucket by using the following bucket and object structure:

AWSLogs/<AccountID>/<ServiceName>/<Region>/<Year>/<Month>/<Day>/<AccountID_ServiceName_Region_ClusterName_LogType_Timestamp.gz

An example is: AWSLogs/123456789012/redshift/us-east-1/2013/10/29/123456789012_redshift_us-east-1_mycluster_userlog_2013-10-29T18:01.gz

If you provide an Amazon S3 key prefix, put the prefix at the start of the key.

For example, if you specify a prefix of myprefix: myprefix/AWSLogs/123456789012/redshift/us-east-1/2013/10/29/123456789012_redshift_us-east-1_mycluster_userlog_2013-10-29T18:01.gz

The Amazon S3 key prefix can't exceed 512 characters. It can't contain spaces ('), double quotation marks ("), single quotation marks ('), a backslash (\). There is also a number of special characters and control characters that aren't allowed. The hexadecimal codes for these characters are as follows:

- x00 to x20
- x22
- x27
- x5c
- x7f or larger

**Troubleshooting Amazon Redshift audit logging**

Amazon Redshift audit logging can be interrupted for the following reasons:

- Amazon Redshift does not have permission to upload logs to the Amazon S3 bucket. Verify that the bucket is configured with the correct IAM policy. For more information, see Bucket permissions for Amazon Redshift audit logging (p. 427).
- The bucket owner changed. When Amazon Redshift uploads logs, it verifies that the bucket owner is the same as when logging was enabled. If the bucket owner has changed, Amazon Redshift cannot upload logs until you configure another bucket to use for audit logging.
- The bucket cannot be found. If the bucket is deleted in Amazon S3, Amazon Redshift cannot upload logs. You either need to recreate the bucket or configure Amazon Redshift to upload logs to a different bucket.

**Logging Amazon Redshift API calls with AWS CloudTrail**

Amazon Redshift is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Amazon Redshift. CloudTrail captures all API calls for Amazon Redshift
as events. These include calls from the Amazon Redshift console and from code calls to the Amazon Redshift API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon Redshift. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in Event history. Using the information collected by CloudTrail, you can determine certain details. These include the request that was made to Amazon Redshift, the IP address it was made from, who made it, when it was made, and other information.

You can use CloudTrail independently from or in addition to Amazon Redshift database audit logging.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Amazon Redshift information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Redshift, that activity is recorded in a CloudTrail event along with other AWS service events in Event history. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for Amazon Redshift, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all regions. The trail logs events from all regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All Amazon Redshift actions are logged by CloudTrail and are documented in the Amazon Redshift API Reference. For example, calls to the CreateCluster, DeleteCluster, and DescribeCluster actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or IAM user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

Understanding Amazon Redshift log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files are not an ordered stack trace of the public API calls, so they do not appear in any specific order.

The following example shows a CloudTrail log entry for a sample CreateCluster call.

```json
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAJ4D5...4n3Z4",
        "ARN": "arn:aws:iam::123456789012:iam/IAMUser",
        "accountId": "123456789012",
        "details": {
            "awsAccount": "123456789012",
            "type": "IAMUser",
            "arn": "arn:aws:iam::123456789012:iam/IAMUser"
        }
    },
    "eventTime": "2023-10-05T10:34:42.775Z",
    "eventSource": "redshift.amazonaws.com",
    "eventSourceArn": "arn:aws:redshift:us-east-1:123456789012:cluster:my-cluster",
    "eventName": "CreateCluster",
    "serviceRequestParameters": {
        "RedshiftClusterName": "my-cluster"
    },
    "responseElements": {
        "RedshiftClusterArn": "arn:aws:redshift:us-east-1:123456789012:cluster:my-cluster"
    }
}
```
"userIdentity": {  
  "type": "IAMUser",  
  "principalId": "AIDAMVNPBQA3EXAMPLE",  
  "arn": "arn:aws:iam::123456789012:user/Admin",  
  "accountId": "123456789012",  
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",  
  "username": "Admin",  
  "sessionContext": {  
    "attributes": {  
      "mfaAuthenticated": "false",  
      "creationDate": "2017-03-03T16:51:56Z"  
    }  
  },  
  "invokedBy": "signin.amazonaws.com".native
},  
"eventTime": "2017-03-03T16:56:09Z",  
"eventSource": "redshift.amazonaws.com",  
"eventName": "CreateCluster",  
"awsRegion": "us-east-2",  
"sourceIPAddress": "52.95.4.13",  
"userAgent": "signin.amazonaws.com",  
"requestParameters": {  
  "clusterIdentifier": "my-dw-instance",  
  "allowVersionUpgrade": true,  
  " enhancementsVpcRouting": false,  
  "encrypted": false,  
  "clusterVersion": "1.0",  
  "masterUsername": "awsuser",  
  "masterUserPassword": "****",  
  "automatedSnapshotRetentionPeriod": 1,  
  "port": 5439,  
  "dbName": "mydbtest",  
  "clusterType": "single-node",  
  "nodeType": "dc1.large",  
  "publiclyAccessible": true,  
  "vpcSecurityGroupIds": ["sg-95f606fc"]  
},  
"responseElements": {  
  "nodeType": "dc1.large",  
  "preferredMaintenanceWindow": "sat:05:30-sat:06:00",  
  "clusterStatus": "creating",  
  "vpcId": "vpc-84c22a3ed",  
  "enhancementsVpcRouting": false,  
  "masterUsername": "awsuser",  
  "clusterSecurityGroups": [],  
  "pendingModifiedValues": {  
    "masterUserPassword": "****"  
  },  
  "dbName": "mydbtest",  
  "clusterType": "1.0",  
  "encrypted": false,  
  "publiclyAccessible": true,  
  "tags": [],  
  "clusterParameterGroups": [  
    {  
      "parameterGroupName": "default.redshift-1.0",  
      "parameterApplyStatus": "in-sync"  
    }  
  ],  
  "allowVersionUpgrade": true,  
  "automatedSnapshotRetentionPeriod": 1,  
  "numberOfNodes": 1,  
  "vpcSecurityGroupIds": [}
The following example shows a CloudTrail log entry for a sample DeleteCluster call.

```json
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAMVNPBQA3EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Admin",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Admin",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2017-03-03T16:58:23Z"
            }
        },
        "invokedBy": "signin.amazonaws.com"
    },
    "eventTime": "2017-03-03T17:02:34Z",
    "eventSource": "redshift.amazonaws.com",
    "eventName": "DeleteCluster",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "52.95.4.13",
    "userAgent": "signin.amazonaws.com",
    "requestParameters": {
        "clusterIdentifier": "my-dw-instance",
        "skipFinalClusterSnapshot": true
    },
    "responseElements": null,
    "requestID": "324cb76a-0033-11e7-809b-1bbee7710bf",
    "eventID": "59bccc3c-ec63-4ce-b47f-3419a36b3fa5",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Working with data sharing information in CloudTrail

All Amazon Redshift data sharing API operations are logged by CloudTrail. For example, calls to the AuthorizeDataShare, DeauthorizeDataShare, and DescribeDataShares operations generate entries in the CloudTrail log files. For information about the data sharing API operations, see the Amazon Redshift API Reference.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or IAM user credentials.
- Whether the request was made with temporary security credentials for an IAM role or federated user.
• Whether the request was made by another AWS service.

For more information about CloudTrail `userIdentity` element, see [CloudTrail `userIdentity` Element](https://docs.aws.amazon.com/redshift/latest/dg/auditingང.html).

**Understanding log file entries for data sharing**

A *trail* in CloudTrail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An *event* represents a single request from any source. An event includes information about the requested action, the date and time of the action, or request parameters. CloudTrail log files aren't an ordered stack trace of the public API calls; they don't appear in any specific order.

The following example shows a CloudTrail log entry that illustrates the `AuthorizeDataShare` operation.

```json
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AKIAIOSFODNN7EXAMPLE:janedoe",
    "arn": "arn:aws:sts::111122223333:user/janedoe",
    "accountId": "111122223333",
    "accessKeyId": "AKIAI44QH8DH8EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAIOSFODNN7EXAMPLE:janedoe",
        "arn": "arn:aws:sts::111122223333:user/janedoe",
        "accountId": "111122223333",
        "userName": "janedoe"
      },
      "attributes": {
        "creationDate": "2021-08-02T23:40:45Z",
        "mfaAuthenticated": "false"
      }
    },
    "eventTime": "2021-08-02T23:40:58Z",
    "eventSource": "redshift.amazonaws.com",
    "eventName": "AuthorizeDataShare",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "3.227.36.75",
    "userAgent": "aws-cli/1.18.118 Python/3.6.10 Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 botocore/1.17.41",
    "requestParameters": {
      "dataShareArn": "arn:aws:redshift:us-east-1:111122223333:datashare:4c64c6ec-73d5-42be-869b-b7f7c43c7a53/testshare",
      "consumerIdentifier": "5555555555555"
    },
    "responseElements": {
      "dataShareArn": "arn:aws:redshift:us-east-1:111122223333:datashare:4c64c6ec-73d5-42be-869b-b7f7c43c7a53/testshare",
      "producerNamespaceArn": "arn:aws:redshift:us-east-1:111122223333:namespace:4c64c6ec-73d5-42be-869b-b7f7c43c7a53",
      "producerArn": "arn:aws:redshift:us-east-1:111122223333:namespace:4c64c6ec-73d5-42be-869b-b7f7c43c7a53",
      "allowPubliclyAccessibleConsumers": true,
      "dataShareAssociations": [
        {
          "consumerIdentifier": "5555555555555",
          "status": "AUTHORIZED",
          "createdDate": "Aug 2, 2021 11:40:56 PM",
          "statusChangeDate": "Aug 2, 2021 11:40:57 PM"
        }
      ]
    }
  }
}`

433
You can use Amazon S3 bucket notification and direct Amazon S3 to publish object-created events to AWS Lambda. When CloudTrail writes logs to your S3 bucket, Amazon S3 can then invoke your Lambda function by passing the Amazon S3 object-created event as a parameter. Your Lambda function can read this log object and process the access records logged by CloudTrail. For more information, see Using AWS Lambda with AWS CloudTrail.

Amazon Redshift account IDs in AWS CloudTrail logs

When Amazon Redshift calls another AWS service for you, the call is logged with an account ID that belongs to Amazon Redshift. It isn’t logged with your account ID. For example, suppose that Amazon Redshift calls AWS Key Management Service (AWS KMS) operations such as CreateGrant, Decrypt, Encrypt, and RetireGrant to manage encryption on your cluster. In this case, the calls are logged by AWS CloudTrail using an Amazon Redshift account ID.

Amazon Redshift uses the account IDs in the following table when calling other AWS services.

<table>
<thead>
<tr>
<th>Region</th>
<th>Region</th>
<th>Account ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (N. Virginia) Region</td>
<td>us-east-1</td>
<td>368064434614</td>
</tr>
<tr>
<td>US East (Ohio) Region</td>
<td>us-east-2</td>
<td>790247189693</td>
</tr>
<tr>
<td>US West (N. California) Region</td>
<td>us-west-1</td>
<td>703715109447</td>
</tr>
<tr>
<td>US West (Oregon) Region</td>
<td>us-west-2</td>
<td>473191095985</td>
</tr>
<tr>
<td>Africa (Cape Town) Region</td>
<td>af-south-1</td>
<td>420376844563</td>
</tr>
<tr>
<td>Asia Pacific (Hong Kong) Region</td>
<td>ap-east-1</td>
<td>651179539253</td>
</tr>
<tr>
<td>Asia Pacific (Jakarta) Region</td>
<td>ap-southeast-3</td>
<td>623197973179</td>
</tr>
<tr>
<td>Asia Pacific (Mumbai) Region</td>
<td>ap-south-1</td>
<td>408097707231</td>
</tr>
<tr>
<td>Asia Pacific (Osaka) Region</td>
<td>ap-northeast-3</td>
<td>398671365691</td>
</tr>
<tr>
<td>Asia Pacific (Seoul) Region</td>
<td>ap-northeast-2</td>
<td>713597048934</td>
</tr>
<tr>
<td>Asia Pacific (Singapore) Region</td>
<td>ap-southeast-1</td>
<td>960118270566</td>
</tr>
<tr>
<td>Asia Pacific (Sydney) Region</td>
<td>ap-southeast-2</td>
<td>485979073181</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo) Region</td>
<td>ap-northeast-1</td>
<td>615915377779</td>
</tr>
<tr>
<td>Canada (Central) Region</td>
<td>ca-central-1</td>
<td>764870610256</td>
</tr>
<tr>
<td>Europe (Frankfurt) Region</td>
<td>eu-central-1</td>
<td>434091160558</td>
</tr>
<tr>
<td>Europe (Ireland) Region</td>
<td>eu-west-1</td>
<td>246478207311</td>
</tr>
</tbody>
</table>
The following example shows a CloudTrail log entry for the AWS KMS Decrypt operation that was called by Amazon Redshift.

```json
{
   "eventVersion": "1.05",
   "userIdentity": {
      "type": "AssumedRole",
      "principalId": "AROA15QPCM6KL4VHFCYY:i-0f53e22db5e5df8a89",
      "arn": "arn:aws:sts::790247189693:assumed-role/prod-23264-role-wp/i-0f53e22db5e5df8a89",
      "accountId": "790247189693",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "sessionContext": {
         "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2017-03-03T16:24:54Z"
         },
         "sessionIssuer": {
            "type": "Role",
            "principalId": "AROA15QPCM6KL4VHFCYY",
            "arn": "arn:aws:iam::790247189693:role/prod-23264-role-wp",
            "accountId": "790247189693",
            "userName": "prod-23264-role-wp"
         }
      }
   },
   "eventTime": "2017-03-03T17:16:51Z",
   "eventSource": "kms.amazonaws.com",
   "eventName": "Decrypt",
   "awsRegion": "us-east-2",
   "sourceIPAddress": "52.14.143.61",
   "userAgent": "aws-internal/3",
   "requestParameters": {
      "encryptionContext": {
         "aws:redshift:createtime": "20170303T1710Z",
      }
   },
   "responseElements": null,
   "requestID": "30d2fe51-0035-11e7-ab67-17595a8411c8",
   "eventID": "619bad54-1764-4de4-a786-8898b0a7f40c",
   "readOnly": true,
   "resources": [
      {
         "ARN": "arn:aws:kms:us-east-2:123456789012:key/f8f4f94f-e588-4254-b7e8-07b99270be7",
         "accountId": "123456789012"
      }
   ]
}
```
Configuring auditing using the console

You can configure Amazon Redshift to create audit log files and store them in S3.

Enabling audit logging using the console

To enable audit logging for a cluster

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster that you want to update.
3. Choose the Properties tab. Then view the Audit logging in the Database configurations section.
4. Choose Edit, then Edit audit logging.
5. On the Edit audit logging page, choose to Enable audit logging and enter your choices regarding where the logs are stored.
6. Choose Save changes to save your choices.

Configuring logging by using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift CLI operations to configure audit logging:

- describe-logging-status
- disable-logging
- enable-logging

You can use the following Amazon Redshift API operations to configure audit logging:

- DescribeLoggingStatus
- DisableLogging
- EnableLogging

Compliance validation for Amazon Redshift

Third-party auditors assess the security and compliance of Amazon Redshift as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS services in scope by compliance program. For general information, see AWS compliance programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading reports in AWS Artifact.
Your compliance responsibility when using Amazon Redshift is determined by the sensitivity of your data, your organization's compliance objectives, and applicable laws and regulations. If your use of Amazon Redshift is subject to compliance with standards like HIPAA, PCI, or FedRAMP, AWS provides resources to help:

- **Security and compliance quick start guides** that discuss architectural considerations and steps for deploying security- and compliance- focused baseline environments on AWS.
- **Architecting for HIPAA security and compliance whitepaper**, which describes how companies can use AWS to create HIPAA-compliant applications.
- **AWS compliance resources**, workbooks and guides that might apply to your industry and location.
- **AWS Config**, an AWS service, can assess how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- **AWS Security Hub**, an AWS service, provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

The following compliance and security documents cover Amazon Redshift and are available on demand through AWS Artifact. For more information, see AWS Artifact.

- Cloud Computing Compliance Controls Catalogue (C5)
- ISO 27001:2013 Statement of Applicability (SoA)
- ISO 27001:2013 Certification
- ISO 27017:2015 Statement of Applicability (SoA)
- ISO 27017:2015 Certification
- ISO 27018:2015 Statement of Applicability (SoA)
- ISO 27018:2014 Certification
- ISO 9001:2015 Certification
- PCI DSS Attestation of Compliance (AOC) and Responsibility Summary
- Service Organization Controls (SOC) 1 Report
- Service Organization Controls (SOC) 2 Report
- Service Organization Controls (SOC) 2 Report For Confidentiality

## Resilience in Amazon Redshift

The AWS global infrastructure is built around AWS Regions and Availability Zones (AZs). AWS Regions provide multiple, physically separated and isolated Availability Zones that are connected with low latency, high throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single data center infrastructures or multiple data center infrastructures.

Almost all AWS Regions have multiple Availability Zones and data centers. You can deploy your applications across multiple Availability Zones in the same Region for fault tolerance and low latency.

For more information on AWS Regions and Availability Zones, see AWS global infrastructure.

## Infrastructure security in Amazon Redshift

As a managed service, Amazon Redshift is protected by the AWS global network security procedures described in the Amazon Web Services: Overview of security processes whitepaper.
You use AWS published API calls to access Amazon Redshift through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

You can call these API operations from any network location. In addition, Amazon Redshift supports resource-based access policies that can include restrictions based on the source IP address.

Network isolation

A virtual private cloud (VPC) based on the Amazon VPC service is your private, logically isolated network in the AWS Cloud. You can deploy an Amazon Redshift cluster within a VPC by taking the following steps:

- Create a VPC in an AWS Region. For more information, see What is Amazon VPC? in the Amazon VPC User Guide.
- Create two or more private VPC subnets. For more information, see VPCs and subnets in the Amazon VPC User Guide.
- Deploy an Amazon Redshift cluster. For more information, see Amazon Redshift cluster subnet groups (p. 82).

An Amazon Redshift cluster is locked down by default upon provisioning. To allow inbound network traffic from Amazon Redshift clients, associate a VPC security group with an Amazon Redshift cluster. For more information, see Amazon Redshift cluster subnet groups (p. 82).

To allow traffic only to or from specific IP address ranges, update the security groups with your VPC. An example is allowing traffic only from or to your corporate network.

Amazon Redshift supports deploying clusters into dedicated tenancy VPCs. For more information, see Dedicated instances in the Amazon EC2 User Guide.

Amazon Redshift cluster security groups

When you provision an Amazon Redshift cluster, it is locked down by default so nobody has access to it. To grant other users inbound access to an Amazon Redshift cluster, you associate the cluster with a security group. If you are on the EC2-VPC platform, you can either use an existing Amazon VPC security group or define a new one and then associate it with a cluster. For more information on managing a cluster on the EC2-VPC platform, see Managing clusters in a VPC (p. 79).

If you are on the EC2-Classic platform, you define a cluster security group and associate it with a cluster as described in the following sections. We recommend that you launch your cluster in a EC2-VPC platform instead of an EC2-Classic platform. However, you can restore an EC2-Classic snapshot to an EC2-VPC cluster using the Amazon Redshift console. For more information, see Restoring a cluster from a snapshot (p. 303).

Topics

- Overview (p. 439)
- Managing cluster security groups using the console (p. 439)
- Managing cluster security groups using the AWS SDK for Java (p. 440)
- Manage cluster security groups using the AWS CLI and Amazon Redshift API (p. 443)
Overview

A cluster security group consists of a set of rules that control access to your cluster. Individual rules identify either a range of IP addresses or an Amazon EC2 security group that is allowed access to your cluster. When you associate a cluster security group with a cluster, the rules that are defined in the cluster security group control access to the cluster.

You can create cluster security groups independent of provisioning any cluster. You can associate a cluster security group with an Amazon Redshift cluster either at the time you provision the cluster or later. Also, you can associate a cluster security group with multiple clusters.

Amazon Redshift provides a cluster security group called default, which is created automatically when you launch your first cluster. Initially, this cluster security group is empty. You can add inbound access rules to the default cluster security group and then associate it with your Amazon Redshift cluster.

If the default cluster security group is enough for you, you don’t need to create your own. However, you can optionally create your own cluster security groups to better manage inbound access to your cluster. For example, suppose that you are running a service on an Amazon Redshift cluster, and you have a few companies as your customers. If you don’t want to provide the same access to all your customers, you might want to create separate cluster security groups, one for each company. You can add rules in each cluster security group to identify the Amazon EC2 security groups and the IP address ranges specific to a company. You can then associate all these cluster security groups with your cluster.

You can associate a cluster security group with many clusters, and you can associate many cluster security groups with a cluster, subject to AWS service limits. For more information, see Amazon Redshift limits.

You can manage cluster security groups using the Amazon Redshift console, and you can manage cluster security groups programmatically by using the Amazon Redshift API or the AWS SDKs.

Amazon Redshift applies changes to a cluster security group immediately. So if you have associated the cluster security group with a cluster, inbound cluster access rules in the updated cluster security group apply immediately.

Managing cluster security groups using the console

You can create, modify, and delete cluster security groups by using the Amazon Redshift console. You can also manage the default cluster security group in the Amazon Redshift console. All of the tasks start from the cluster security group list. You must choose a cluster security group to manage it.

You can’t delete the default cluster security group, but you can modify it by authorizing or revoking ingress access.

Creating a cluster security group

To create a cluster security group

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Security groups to display the Cluster security groups page.
   Note
   You can only manage cluster security groups when logged in with an EC2-Classic AWS account.
3. Choose Create cluster security group to display the Create cluster security group window.
4. For the new security group, enter values for the following:
   • Name
• Description
• CIDR/IP range to authorize in the form nnn.nnn.nnn.nn/nn
• AWS account ID (without hyphens)
• EC2 security group name
5. Choose Create to create the security group.

Tagging a cluster security group

To tag a cluster security group
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Security groups to display the Cluster security groups page.
   Note
   You can only manage cluster security groups when logged in with an EC2-Classic AWS account.
3. Choose a cluster security group, then choose Manage tags to display the Manage tags page.
4. On the Manage tags page, add new tags, and update or delete existing tags. For each new tag, provide information for Key and Value.
5. Choose Apply to save your tags.

Deleting a cluster security group
If a cluster security group is associated with one or more clusters, you can't delete it.

To delete a cluster security group
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Security groups to display the Cluster security groups page.
   Note
   You can only manage cluster security groups when logged in with an EC2-Classic AWS account.
3. Choose the security group that you want to delete, then choose Delete.

Associating a cluster security group with a cluster
If you are on the EC2-VPC platform, see Managing VPC security groups for a cluster (p. 81) for more information about associating VPC security groups with your cluster. We recommend that you launch your cluster in an EC2-VPC platform. However, you can restore an EC2-Classic snapshot to an EC2-VPC cluster using the Amazon Redshift console. For more information, see Restoring a cluster from a snapshot (p. 303).

Each cluster you provision on the EC2-Classic platform has one or more cluster security groups associated with it. You can associate a cluster security group with a cluster when you create the cluster, or you can associate a cluster security group later by modifying the cluster.

Managing cluster security groups using the AWS SDK for Java
The following example demonstrates common operations on cluster security groups, including:
• Creating a new cluster security group.
• Adding ingress rules to a cluster security group.
• Associating a cluster security group with a cluster by modifying the cluster configuration.

By default, when a new cluster security group is created, it has no ingress rules. This example modifies a new cluster security group by adding two ingress rules. One ingress rule is added by specifying a CIDR/IP range; the other is added by specifying an owner ID and Amazon EC2 security group combination.

For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). You need to update the code and provide a cluster identifier and AWS account number.

Example

```java
package com.amazonaws.services.redshift;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import com.amazonaws.services.redshift.model.*;
public class CreateAndModifyClusterSecurityGroup {
    public static AmazonRedshift client;
    public static String clusterSecurityGroupName = "securitygroup1";
    public static String clusterIdentifier = "***provide a cluster identifier***";
    public static String ownerID = "***provide a 12-digit account number***";
    public static void main(String[] args) throws IOException {
        // Default client using the DefaultAWSCredentialsProviderChain
        client = AmazonRedshiftClientBuilder.defaultClient();
        // Create a new cluster security group
        CreateClusterSecurityGroupResult createResult = client.createClusterSecurityGroup(new CreateClusterSecurityGroupRequest()
            .withClusterIdentifier(clusterIdentifier)
            .withClusterSecurityGroupName(clusterSecurityGroupName)
            .withIpRanges("0.0.0.0/0")
        );
        // Modify the cluster security group to add two ingress rules
        ModifyClusterSecurityGroupRequest modifyRequest = new ModifyClusterSecurityGroupRequest()
            .withClusterIdentifier(clusterIdentifier)
            .withClusterSecurityGroupName(clusterSecurityGroupName)
            .withIpRanges("0.0.0.0/0", "192.168.1.0/24")
            .withEgressIpRanges("0.0.0.0/0", "192.168.1.0/24")
        ;
        client.modifyClusterSecurityGroup(modifyRequest);
    }
}
```

try {
    createClusterSecurityGroup();
    describeClusterSecurityGroups();
    addIngressRules();
    associateSecurityGroupWithCluster();
} catch (Exception e) {
    System.err.println("Operation failed: " + e.getMessage());
}

private static void createClusterSecurityGroup() {
    CreateClusterSecurityGroupRequest request = new CreateClusterSecurityGroupRequest().withDescription("my cluster security group")
        .withClusterSecurityGroupName(clusterSecurityGroupName);
    client.createClusterSecurityGroup(request);
    System.out.format("Created cluster security group: '%s'
", clusterSecurityGroupName);
}

private static void addIngressRules() {
    AuthorizeClusterSecurityGroupIngressRequest request = new 
    AuthorizeClusterSecurityGroupIngressRequest() .withClusterSecurityGroupName(clusterSecurityGroupName) 
        .withCIDRIP("192.168.40.5/32");
    ClusterSecurityGroup result = client.authorizeClusterSecurityGroupIngress(request);

    request = new AuthorizeClusterSecurityGroupIngressRequest() 
        .withClusterSecurityGroupName(clusterSecurityGroupName) 
        .withEC2SecurityGroupName("default") 
        .withEC2SecurityGroupOwnerID(ownerID);
    result = client.authorizeClusterSecurityGroupIngress(request);
    System.out.format("\nAdded ingress rules to security group '%%s'
", clusterSecurityGroupName);
    printResultSecurityGroup(result);
}

private static void associateSecurityGroupWithCluster() {

    // Get existing security groups used by the cluster.
    DescribeClustersRequest request = new DescribeClustersRequest() 
        .withClusterIdentifier(clusterIdentifier);
    DescribeClustersResult result = client.describeClusters(request);

    List<ClusterSecurityGroupMembership> membershipList = 
        result.getClusters().get(0).getClusterSecurityGroups();

    List<String> secGroupNames = new ArrayList<String>();
    for (ClusterSecurityGroupMembership mem : membershipList) {
        secGroupNames.add(mem.getClusterSecurityGroupName());
    }

    // Add new security group to the list.
    secGroupNames.add(clusterSecurityGroupName);

    // Apply the change to the cluster.
    ModifyClusterRequest request2 = new ModifyClusterRequest() 
        .withClusterIdentifier(clusterIdentifier) 
        .withClusterSecurityGroups(secGroupNames);

    Cluster result2 = client.modifyCluster(request2);
    System.out.format("\nAssociated security group '%%s' to cluster '%%s'.", clustera
    erSecurityGroupName, clusterIdentifier);
}
private static void describeClusterSecurityGroups() {
    DescribeClusterSecurityGroupsRequest request = new DescribeClusterSecurityGroupsRequest();
    DescribeClusterSecurityGroupsResult result = client.describeClusterSecurityGroups(request);
    printResultSecurityGroups(result.getClusterSecurityGroups());
}

private static void printResultSecurityGroups(List<ClusterSecurityGroup> groups) {
    if (groups == null) {
        System.out.println("Describe cluster security groups result is null.");
        return;
    }
    System.out.println("Printing security group results:");
    for (ClusterSecurityGroup group : groups) {
        printResultSecurityGroup(group);
    }
}

private static void printResultSecurityGroup(ClusterSecurityGroup group) {
    System.out.format("Name: '%s', Description: '%s'
", group.getClusterSecurityGroupName(), group.getDescription());
    for (EC2SecurityGroup g : group.getEC2SecurityGroups()) {
        System.out.format("EC2group: '%s', '%s', '%s'
", g.getEC2SecurityGroupName(), g.getEC2SecurityGroupOwnerId(), g.getStatus());
    }
    for (IPRange range : group.getIPRanges()) {
        System.out.format("IPRanges: '%s', '%s'
", range.getCIDRIP(), range.getStatus());
    }
}

// snippet-end:[redshift.java.CreateAndModifyClusterSecurityGroup.complete]

Manage cluster security groups using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift CLI operations to manage cluster security groups.

- authorize-cluster-security-group-ingress
- create-cluster-security-group
- delete-cluster-security-group
- describe-cluster-security-groups
- revoke-cluster-security-group-ingress

You can use the following Amazon Redshift API operations to manage cluster security groups.

- AuthorizeClusterSecurityGroupIngress
- CreateClusterSecurityGroup
- DeleteClusterSecurityGroup
- DescribeClusterSecurityGroups
- RevokeClusterSecurityGroupIngress
Connecting to Amazon Redshift using an interface VPC endpoint

You can connect directly to Amazon Redshift API service using an interface VPC endpoint (AWS PrivateLink) in your virtual private cloud (VPC) instead of connecting over the internet. For information about Amazon Redshift API actions, see Actions in the Amazon Redshift API Reference. For more information about AWS PrivateLink, see Interface VPC endpoints (AWS PrivateLink) in the Amazon VPC User Guide. Note that JDBC/ODBC connection to the cluster is not part of Amazon Redshift API service.

When you use an interface VPC endpoint, communication between your VPC and Amazon Redshift is conducted entirely within the AWS network, which can provide greater security. Each VPC endpoint is represented by one or more elastic network interfaces with private IP addresses in your VPC subnets. For more information on elastic network interfaces, see Elastic network interfaces in the Amazon EC2 User Guide for Linux Instances.

An interface VPC endpoint connects your VPC directly to Amazon Redshift. It doesn't use an internet gateway, network address translation (NAT) device, virtual private network (VPN) connection, or AWS Direct Connect connection. The instances in your VPC don't need public IP addresses to communicate with the Amazon Redshift API.

To use Amazon Redshift through your VPC, you have two options. One is to connect from an instance that is inside your VPC. The other is to connect your private network to your VPC by using an AWS VPN option or AWS Direct Connect. For more information about AWS VPN options, see VPN connections in the Amazon VPC User Guide. For information about AWS Direct Connect, see Creating a Connection in the AWS Direct Connect User Guide.

You can create an interface VPC endpoint to connect to Amazon Redshift using the AWS Management Console or AWS Command Line Interface (AWS CLI) commands. For more information, see Creating an Interface Endpoint.

After you create an interface VPC endpoint, you can enable private DNS host names for the endpoint. When you do, the default Amazon Redshift endpoint (https://redshift.Region.amazonaws.com) resolves to your VPC endpoint.

If you don't enable private DNS host names, Amazon VPC provides a DNS endpoint name that you can use in the following format.

```
VPC_endpoint_ID.redshift.Region.vpce.amazonaws.com
```

For more information, see Interface VPC endpoints (AWS PrivateLink) in the Amazon VPC User Guide.

Amazon Redshift supports making calls to all of its API operations inside your VPC.

You can attach VPC endpoint policies to a VPC endpoint to control access for AWS Identity and Access Management (IAM) principals. You can also associate security groups with a VPC endpoint to control inbound and outbound access based on the origin and destination of network traffic. An example is a range of IP addresses. For more information, see Controlling Access to Services with VPC Endpoints in the Amazon VPC User Guide.

Creating a VPC endpoint policy for Amazon Redshift

You can create a policy for VPC endpoints for Amazon Redshift to specify the following:

- The principal that can or can't perform actions
- The actions that can be performed
- The resources on which actions can be performed
For more information, see Controlling access to services with VPC endpoints in the Amazon VPC User Guide.

Following, you can find examples of VPC endpoint policies.

**Topics**
- Example: VPC endpoint policy to deny all access from a specified AWS account (p. 445)
- Example: VPC endpoint policy to allow VPC access only to a specified IAM principal (user) (p. 445)
- Example: VPC endpoint policy to allow read-only Amazon Redshift operations (p. 446)
- Example: VPC endpoint policy denying access to a specified cluster (p. 446)

**Example: VPC endpoint policy to deny all access from a specified AWS account**

The following VPC endpoint policy denies the AWS account 123456789012 all access to resources using this endpoint.

```json
{
   "Statement": [
      {
         "Action": "*",
         "Effect": "Allow",
         "Resource": "*",
         "Principal": "*
      },
      {
         "Action": "*",
         "Effect": "Deny",
         "Resource": "*",
         "Principal": {
            "AWS": [123456789012]
         }
      }
   ]
}
```

**Example: VPC endpoint policy to allow VPC access only to a specified IAM principal (user)**

The following VPC endpoint policy allows full access only to the IAM user redshiftadmin in AWS account 123456789012. All other IAM principals are denied access using the endpoint.

```json
{
   "Statement": [
      {
         "Action": "*",
         "Effect": "Allow",
         "Resource": "*",
         "Principal": {
            "AWS": [
               "arn:aws:iam::123456789012:user/redshiftadmin"
            ]
         }
      }
   ]
}
```
Example: VPC endpoint policy to allow read-only Amazon Redshift operations

The following VPC endpoint policy allows only AWS account 123456789012 to perform the specified Amazon Redshift actions.

The actions specified provide the equivalent of read-only access for Amazon Redshift. All other actions on the VPC are denied for the specified account. Also, all other accounts are denied any access. For a list of Amazon Redshift actions, see Actions, Resources, and Condition Keys for Amazon Redshift in the IAM User Guide.

```json
{
    "Statement": [
        {
            "Action": [
                "redshift:DescribeAccountAttributes",
                "redshift:DescribeClusterParameterGroups",
                "redshift:DescribeClusterParameters",
                "redshift:DescribeClusterSecurityGroups",
                "redshift:DescribeClusterSnapshots",
                "redshift:DescribeClusterSubnetGroups",
                "redshift:DescribeClusterVersions",
                "redshift:DescribeDefaultClusterParameters",
                "redshift:DescribeEventCategories",
                "redshift:DescribeEventSubscriptions",
                "redshift:DescribeHsmClientCertificates",
                "redshift:DescribeHsmConfigurations",
                "redshift:DescribeLoggingStatus",
                "redshift:DescribeOrderableClusterOptions",
                "redshift:DescribeQuery",
                "redshift:DescribeReservedNodeOfferings",
                "redshift:DescribeReservedNodes",
                "redshift:DescribeResize",
                "redshift:DescribeSavedQueries",
                "redshift:DescribeScheduledActions",
                "redshift:DescribeSnapshotCopyGrants",
                "redshift:DescribeSnapshotSchedules",
                "redshift:DescribeStorage",
                "redshift:DescribeTable",
                "redshift:DescribeTableRestoreStatus",
                "redshift:DescribeTags",
                "redshift:FetchResults",
                "redshift:GetReservedNodeExchangeOfferings"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            }
        }
    ]
}
```

Example: VPC endpoint policy denying access to a specified cluster

The following VPC endpoint policy allows full access for all accounts and principals. At the same time, it denies any access for AWS account 123456789012 to actions performed on the Amazon Redshift cluster with cluster ID my-redshift-cluster. Other Amazon Redshift actions that don’t support resource-level permissions for clusters are still allowed. For a list of Amazon Redshift actions and their
corresponding resource type, see Actions, Resources, and Condition Keys for Amazon Redshift in the IAM User Guide.

```
{
    "Statement": [ 
        {
            "Action": ":*",
            "Effect": "Allow",
            "Resource": ":*",
            "Principal": "*"
        },
        {
            "Action": "*:*",
            "Effect": "Deny",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            }
        }
    ]
}
```

Configuration and vulnerability analysis in Amazon Redshift

AWS handles basic security tasks like guest operating system (OS) and database patching, firewall configuration, and disaster recovery (DR). These procedures have been reviewed by certified third parties. For more information, see Compliance validation for Amazon Redshift (p. 436), the Shared responsibility model, and Best Practices for Security, Identity, and Compliance.

Amazon Redshift automatically applies upgrades and patches your data warehouse so you can focus on your application and not on its administration. Patches and upgrades are applied during a configurable maintenance window. For more information, see Maintenance windows (p. 46).

Amazon Redshift query editor v2 is an AWS-managed application. All patches and updates are applied by AWS as needed.
Using the Amazon Redshift management interfaces

Amazon Redshift supports several management interfaces that you can use to create, manage, and delete Amazon Redshift clusters: the AWS SDKs, the AWS Command Line Interface (AWS CLI), and the Amazon Redshift management API.

**The Amazon Redshift API** – You can call this Amazon Redshift management API by submitting a request. Requests are HTTP or HTTPS requests that use the HTTP verbs `GET` or `POST` with a parameter named `Action`. Calling the Amazon Redshift API is the most direct way to access the Amazon Redshift service. However, it requires that your application handle low-level details such as error handling and generating a hash to sign the request.

- For information about building and signing an Amazon Redshift API request, see [Signing an HTTP request](p. 451).
- For information about the Amazon Redshift API actions and data types for Amazon Redshift, see the Amazon Redshift API reference.

**AWS SDKs** – You can use the AWS SDKs to perform Amazon Redshift cluster-related operations. Several of the SDK libraries wrap the underlying Amazon Redshift API. They integrate the API functionality into the specific programming language and handle many of the low-level details, such as calculating signatures, handling request retries, and error handling. Calling the wrapper functions in the SDK libraries can greatly simplify the process of writing an application to manage an Amazon Redshift cluster.

- Amazon Redshift is supported by the AWS SDKs for Java, .NET, PHP, Python, Ruby, and Node.js. The wrapper functions for Amazon Redshift are documented in the reference manual for each SDK. For a list of the AWS SDKs and links to their documentation, see [Tools for Amazon Web Services](p. 453).
- This guide provides examples of working with Amazon Redshift using the Java SDK. For more general AWS SDK code examples, see [Sample code & libraries](p. 453).

**AWS CLI** – The CLI provides a set of command line tools that you can use to manage AWS services from Windows, Mac, and Linux computers. The AWS CLI includes commands based on the Amazon Redshift API actions.

- For information about installing and setting up the Amazon Redshift CLI, see [Setting up the Amazon Redshift CLI](p. 453).
- For reference material on the Amazon Redshift CLI commands, see [Amazon Redshift in the AWS CLI Reference](p. 453).

**Topics**

- [Using the AWS SDK for Java with Amazon Redshift](p. 448)
- [Signing an HTTP request](p. 451)
- [Setting up the Amazon Redshift CLI](p. 453)

Using the AWS SDK for Java with Amazon Redshift

The AWS SDK for Java provides a class named `AmazonRedshiftClientBuilder`, which you can use to interact with Amazon Redshift. For information about downloading the AWS SDK for Java, go to [AWS SDK for Java](p. 453).
Note
The AWS SDK for Java provides thread-safe clients for accessing Amazon Redshift. As a best practice, your applications should create one client and reuse the client between threads.

You can use the AmazonRedshiftClientBuilder and AwsClientBuilder classes to configure an endpoint and create an AmazonRedshift client. You can then use the client object to create an instance of a Cluster object. The Cluster object includes methods that map to underlying Amazon Redshift Query API actions. (These actions are described in the Amazon Redshift API reference). When you call a method, you must create a corresponding request object. The request object includes information that you must pass with the actual request. The Cluster object provides information returned from Amazon Redshift in response to the request.

The following example illustrates using the AmazonRedshiftClientBuilder class to configure an endpoint and then create a 2-node ds2.xlarge cluster.

```java
String endpoint = "https://redshift.us-east-1.amazonaws.com/";
String region = "us-east-1";
AwsClientBuilder.EndpointConfiguration config = new
    AwsClientBuilder.EndpointConfiguration(endpoint, region);
AmazonRedshiftClientBuilder clientBuilder = AmazonRedshiftClientBuilder.standard();
clientBuilder.setEndpointConfiguration(config);
AmazonRedshift client = clientBuilder.build();
CreateClusterRequest request = new CreateClusterRequest()
    .withClusterIdentifier("exampleclusterusingjava")
    .withMasterUsername("masteruser")
    .withMasterUserPassword("12345678Aa")
    .withNodeType("ds2.xlarge")
    .withNumberOfNodes(2);
Cluster createResponse = client.createCluster(request);
System.out.println("Created cluster " + createResponse.getClusterIdentifier());
```

Running Java examples for Amazon Redshift using Eclipse

General process of running Java code examples using Eclipse

1. Create a new AWS Java Project in Eclipse.
   Follow the steps in Setting up the AWS Toolkit for Eclipse in the AWS Toolkit for Eclipse Getting Started Guide.
2. Copy the sample code from the section of this document that you are reading and paste it into your project as a new Java class file.
3. Run the code.

Running Java examples for Amazon Redshift from the command line

General process of running Java code examples from the command line

1. Set up and test your environment as follows:
a. Create a directory to work in and in it create src, bin, and sdk subfolders.
b. Download the AWS SDK for Java and unzip it to the sdk subfolder you created. After you unzip the SDK, you should have four subdirectories in the sdk folder, including a lib and third-party folder.
c. Supply your AWS credentials to the SDK for Java. For more information, go to Providing AWS credentials in the AWS SDK for Java in the AWS SDK for Java Developer Guide.
d. Ensure that you can run the Java program compiler (javac) and the Java application launcher (java) from your working directory. You can test by running the following commands:

```
javac -help
java -help
```

2. Put the code that you want to run in a .java file, and save the file in the src folder. To illustrate the process, we use the code from Managing cluster security groups using the AWS SDK for Java (p. 440) so that the file in the src directory is CreateAndModifyClusterSecurityGroup.java.

3. Compile the code.

```
javac -cp sdk/lib/aws-java-sdk-1.3.18.jar -d bin src
\CreateAndModifyClusterSecurityGroup.java
```

If you are using a different version of the AWS SDK for Java, adjust the classpath (–cp) for your version.

4. Run the code. In the following command, line breaks are added for readability.

```
java -cp "bin;
   sdk/lib/;*
   sdk/third-party/commons-logging-1.1.1/;*
   sdk/third-party/httpcomponents-client-4.1.1/;*
   sdk/third-party/jackson-core-1.8/;*
   CreateAndModifyClusterSecurityGroup"
```

Change the class path separator as needed for your operating system. For example, for Windows, the separator is ";" (as shown), and for Unix, it is ":". Other code examples may require more libraries than are shown in this example, or the version of the AWS SDK you are working with may have different third-party folder names. For these cases, adjust the classpath (–cp) as appropriate.

To run samples in this document, use a version of the AWS SDK that supports Amazon Redshift. To get the latest version of the AWS SDK for Java, go to AWS SDK for Java.

### Setting the endpoint

By default, the AWS SDK for Java uses the endpoint https://redshift.us-east-1.amazonaws.com/. You can set the endpoint explicitly with the client.setEndpoint method as shown in the following Java code snippet.

**Example**

```java
client = new AmazonRedshiftClient(creds);
client.setEndpoint("https://redshift.us-east-1.amazonaws.com/");
```

For a list of supported AWS regions where you can provision a cluster, go to the Regions and endpoints section in the Amazon Web Services Glossary.
Signing an HTTP request

Amazon Redshift requires that every request you send to the management API be authenticated with a signature. This topic explains how to sign your requests.

If you are using one of the AWS Software Development Kits (SDKs) or the AWS Command Line Interface, request signing is handled automatically, and you can skip this section. For more information about using AWS SDKs, see Using the Amazon Redshift management interfaces (p. 448). For more information about using the Amazon Redshift Command Line Interface, go to Amazon Redshift command line reference.

To sign a request, you calculate a digital signature by using a cryptographic hash function. A cryptographic hash is a function that returns a unique hash value that is based on the input. The input to the hash function includes the text of your request and your secret access key. The hash function returns a hash value that you include in the request as your signature. The signature is part of the Authorization header of your request.

Note
For API access, you need an access key ID and secret access key. Use IAM user access keys instead of AWS account root user access keys. For more information about creating access keys, see Managing access keys for IAM users in the IAM User Guide.

After Amazon Redshift receives your request, it recalculates the signature by using the same hash function and input that you used to sign the request. If the resulting signature matches the signature in the request, Amazon Redshift processes the request; otherwise, the request is rejected.

Amazon Redshift supports authentication using AWS signature version 4. The process for calculating a signature is composed of three tasks. These tasks are illustrated in the example that follows.

• Task 1: Create a canonical request

Rearrange your HTTP request into a canonical form. Using a canonical form is necessary because Amazon Redshift uses the same canonical form to calculate the signature it compares with the one you sent.

• Task 2: Create a string to sign

Create a string that you will use as one of the input values to your cryptographic hash function. The string, called the string to sign, is a concatenation of the name of the hash algorithm, the request date, a credential scope string, and the canonicalized request from the previous task. The credential scope string itself is a concatenation of date, region, and service information.

• Task 3: Create a signature

Create a signature for your request by using a cryptographic hash function that accepts two input strings, your string to sign and a derived key. The derived key is calculated by starting with your secret access key and using the credential scope string to create a series of hash-based message authentication codes (HMAC-SHA256).

Example signature calculation

The following example walks you through the details of creating a signature for CreateCluster request. You can use this example as a reference to check your own signature calculation method. Other reference calculations are included in the Signature Version 4 test suite of the Amazon Web Services Glossary.

You can use a GET or POST request to send requests to Amazon Redshift. The difference between the two is that for the GET request your parameters are sent as query string parameters. For the POST request they are included in the body of the request. The example below shows a POST request.
The example assumes the following:

- The time stamp of the request is Fri, 07 Dec 2012 00:00:00 GMT.
- The endpoint is US East (Northern Virginia) Region, us-east-1.

The general request syntax is:

```plaintext
https://redshift.us-east-1.amazonaws.com/
  ?Action=CreateCluster
  &ClusterIdentifier=examplecluster
  &MasterUsername=masteruser
  &MasterUserPassword=12345678Aa
  &NumberOfNode=2
  &NodeType=ds2.xlarge
  &Version=2012-12-01
  &x-amz-algorithm=AWS4-HMAC-SHA256
  &x-amz-credential=AKIAIOSFODNN7EXAMPLE/20121207/us-east-1/redshift/aws4_request
  &x-amz-date=20121207T000000Z
  &x-amz-signedheaders=content-type;host;x-amz-date
```

The canonical form of the request calculated for Task 1: Create a Canonical Request (p. 451) is:

```plaintext
POST
/
content-type:application/x-www-form-urlencoded; charset=utf-8
host:redshift.us-east-1.amazonaws.com
x-amz-date:20121207T000000Z
content-type;host;x-amz-date

55141b5d2aff6042ccd9d2af808f8f95ac78255e25b832d23dbd720226de1625d
```

The last line of the canonical request is the hash of the request body. The third line in the canonical request is empty because there are no query parameters for this API.

The string to sign for Task 2: Create a String to Sign (p. 451) is:

```plaintext
AWS4-HMAC-SHA256
20121207T000000Z
20121207/us-east-1/redshift/aws4_request
06b6bef4f4f060a5558b60c627cc6c5b5a959b99022b5ac2187be80cbac0714
```

The first line of the string to sign is the algorithm, the second line is the time stamp, the third line is the credential scope, and the last line is a hash of the canonical request from Task 1: Create a Canonical Request (p. 451). The service name to use in the credential scope is redshift.

For Task 3: Create a Signature (p. 451), the derived key can be represented as:

```plaintext
derived key = HMAC(HMAC(HMAC(HMAC("AWS4" + YourSecretAccessKey,"20121207"),"us-east-1"),"redshift"),"aws4_request")
```

The derived key is calculated as series of hash functions. Starting from the inner HMAC statement in the formula above, you concatenate the phrase AWS4 with your secret access key and use this as the key to hash the data "us-east-1". The result of this hash becomes the key for the next hash function.

After you calculate the derived key, you use it in a hash function that accepts two input strings, your string to sign and the derived key. For example, if you use the secret access key wJalrXUt

X7MDENG/bPxRfiCYEXAMPLEKEY and the string to sign given earlier, then the calculated signature is as follows:

9a6b557aa9f38dea83d9215d8f0eae54100877f3e0735d38498d7ae489117920

The final step is to construct the Authorization header. For the demonstration access key AKIAIOSFODNN7EXAMPLE, the header (with line breaks added for readability) is:

Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20121207/us-east-1/redshift/aws4_request,
SignedHeaders=content-type;host;x-amz-date,
Signature=9a6b557aa9f38dea83d9215d8f0eae54100877f3e0735d38498d7ae489117920

Setting up the Amazon Redshift CLI

This section explains how to set up and run the AWS CLI command line tools for use in managing Amazon Redshift. The Amazon Redshift command line tools run on the AWS Command Line Interface (AWS CLI), which in turn uses Python (https://www.python.org/). The AWS CLI can be run on any operating system that supports Python.

Installation instructions

To begin using the Amazon Redshift command line tools, you first set up the AWS CLI, and then you add configuration files that define the Amazon Redshift CLI options.

If you have already installed and configured the AWS CLI for another AWS service, you can skip this procedure.

To install the AWS Command Line Interface

1. Go to Getting set up with the AWS command line interface, and then follow the instructions for installing the AWS CLI.

   For CLI access, you need an access key ID and secret access key. Use IAM user access keys instead of AWS account root user access keys. IAM lets you securely control access to AWS services and resources in your AWS account. For more information about creating access keys, see Understanding and getting your security credentials in the AWS General Reference.

2. Create a file containing configuration information such as your access keys, default region, and command output format. Then set the AWS_CONFIG_FILE environment variable to reference that file. For detailed instructions, go to Configuring the AWS command line interface in the AWS Command Line Interface User Guide.

3. Run a test command to confirm that the AWS CLI interface is working. For example, the following command should display help information for the AWS CLI:

   ```
   aws help
   ```

   The following command should display help information for Amazon Redshift:

   ```
   aws redshift help
   ```

   For reference material on the Amazon Redshift CLI commands, go to Amazon Redshift in the AWS CLI Reference.
Getting started with the AWS Command Line Interface

To help you get started using the AWS Command Line Interface (AWS CLI), this section shows how to perform basic administrative tasks for an Amazon Redshift cluster. These tasks are very similar to those in the Amazon Redshift Getting Started Guide, but they are focused on the AWS CLI rather than the Amazon Redshift console.

This section walks you through the process of creating a cluster, creating database tables, uploading data, and testing queries. You use the AWS CLI to provision a cluster and to authorize necessary access permissions. You will then use the SQL Workbench client to connect to the cluster and create sample tables, upload sample data, and run test queries.

Step 1: Before you begin

If you don’t already have an AWS account, you must sign up for one. Then you'll need to set up the Amazon Redshift command line tools. Finally, you'll need to download client tools and drivers in order to connect to your cluster.

Step 1.1: Sign up for an AWS account

For information about signing up for an AWS user account, see the Amazon Redshift Getting Started Guide.

Step 1.2: Download and install the AWS CLI

If you have not installed the AWS CLI, see Setting up the Amazon Redshift CLI (p. 453).

Step 1.3: Download the client tools and drivers

You can use any SQL client tools to connect to an Amazon Redshift cluster with PostgreSQL JDBC or ODBC drivers. If you do not currently have such software installed, you can use SQL Workbench, a free cross-platform tool that you can use to query tables in an Amazon Redshift cluster. The examples in this section will use the SQL Workbench client.

To download SQL Workbench and the PostgreSQL drivers, see the Amazon Redshift Getting Started Guide.

Step 2: Launch a cluster

Now you're ready to launch a cluster by using the AWS CLI.

Important

The cluster that you're about to launch will be live, and not running in a sandbox. You will incur the standard usage fees for the cluster until you terminate it. For pricing information, see the Amazon Redshift pricing page.

If you complete the exercise described here in one sitting and terminate your cluster when you are finished, the total charges will be minimal.

The create-cluster command has a large number of parameters. For this exercise, you will use the parameter values that are described in the following table. Before you create a cluster in a production environment, we recommend that you review all the required and optional parameters so that your cluster configuration matches your requirements. For more information, see create-cluster

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter value for this exercise</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster-identifier</td>
<td>examplecluster</td>
</tr>
<tr>
<td>Parameter name</td>
<td>Parameter value for this exercise</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>master-username</td>
<td>adminuser</td>
</tr>
<tr>
<td>master-user-password</td>
<td>TopSecret1</td>
</tr>
<tr>
<td>node-type</td>
<td>ds2.xlarge or the node size that you want to use. For more information, see Clusters and nodes in Amazon Redshift (p. 34)</td>
</tr>
<tr>
<td>cluster-type</td>
<td>single-node</td>
</tr>
</tbody>
</table>

To create your cluster, enter the following command.

```bash
aws redshift create-cluster --cluster-identifier examplecluster --master-username masteruser --master-user-password TopSecret1 --node-type ds2.xlarge --cluster-type single-node
```

The cluster creation process will take several minutes to complete. To check the status, enter the following command.

```bash
aws redshift describe-clusters --cluster-identifier examplecluster
```

The output will look similar to the following.

```
{
    "Clusters": [
        {
            ...output omitted...
            "ClusterStatus": "creating",
            "ClusterIdentifier": "examplecluster",
            ...output omitted...
        }
    ]
}
```

When the `ClusterStatus` field changes from creating to available, your cluster is ready for use.

In the next step, you will authorize access so that you can connect to the cluster.

**Step 3: Authorize inbound traffic for cluster access**

You must explicitly grant inbound access to your client in order to connect to the cluster. Your client can be an Amazon EC2 instance or an external computer.

When you created a cluster in the previous step, because you did not specify a security group, you associated the default cluster security group with the cluster. The default cluster security group contains no rules to authorize any inbound traffic to the cluster. To access the new cluster, you must add rules for inbound traffic, which are called ingress rules, to the cluster security group.

**Ingress rules for applications running on the internet**

If you are accessing your cluster from the Internet, you will need to authorize a Classless Inter-Domain Routing IP (CIDR/IP) address range. For this example, we will use a CIDR/IP rule of 192.0.2.0/24; you will need to modify this range to reflect your actual IP address and netmask.

To allow network ingress to your cluster, enter the following command.
Ingress rules for EC2 instances

If you are accessing your cluster from an Amazon EC2 instance, you will need to authorize an Amazon EC2 security group. To do so, you specify the security group name, along with the 12-digit account number of the EC2 security group owner.

You can use the Amazon EC2 console to determine the EC2 security group associated with your instance:

To find your AWS account number, go to https://aws.amazon.com/ and sign in to the My Account page. Your AWS account number is shown in the upper right-hand corner of that page.

For this example, we will use myec2securitygroup for the Amazon EC2 security group name, and 123456789012 for the account number. You will need to modify these to suit your needs.

To allow network ingress to your cluster, enter the following command.

```bash
aws redshift authorize-cluster-security-group-ingress --cluster-security-group-name default --ec2-security-group-name myec2securitygroup --ec2-security-group-owner 123456789012
```

Step 4: Connect to your cluster

Now that you have added an ingress rule to the default cluster security group, incoming connections from a specific CIDR/IP or EC2 Security Group to examplecluster are authorized.
You are now ready to connect to the cluster.

For information about connecting to your cluster, go to the Amazon Redshift getting started guide.

**Step 5: Create tables, upload data, and try example queries**

For information about creating tables, uploading data, and issuing queries, go to the Amazon Redshift Getting Started Guide.

**Step 6: Delete your sample cluster**

After you have launched a cluster and it is available for use, you are billed for the time the cluster is running, even if you are not actively using it. When you no longer need the cluster, you can delete it.

When you delete a cluster, you must decide whether to create a final snapshot. Because this is an exercise and your test cluster should not have any important data in it, you can skip the final snapshot.

To delete your cluster, enter the following command.

```
aws redshift delete-cluster --cluster-identifier examplecluster --skip-final-cluster-snapshot
```

Congratulations! You successfully launched, authorized access to, connected to, and terminated a cluster.
Monitoring Amazon Redshift cluster performance

Amazon Redshift provides performance metrics and data so that you can track the health and performance of your clusters and databases. In this section, we discuss the types of data that you can work with in Amazon Redshift, specifically in the Amazon Redshift console.

Topics
- Overview (p. 458)
- Monitoring Amazon Redshift using CloudWatch metrics (p. 459)
- Working with performance data in the Amazon Redshift console (p. 466)

Overview

The performance data that you can use in the Amazon Redshift console falls into two categories:

- **Amazon CloudWatch metrics** – Amazon CloudWatch metrics help you monitor physical aspects of your cluster, such as CPU utilization, latency, and throughput. Metric data is displayed directly in the Amazon Redshift console. You can also view it in the CloudWatch console. Alternatively, you can consume it in any other way you work with metrics, such as with the AWS CLI or one of the AWS SDKs.

- **Query/Load performance data** – Performance data helps you monitor database activity and performance. This data is aggregated in the Amazon Redshift console to help you easily correlate what you see in CloudWatch metrics with specific database query and load events. You can also create your own custom performance queries and run them directly on the database. Query and load performance data is displayed only in the Amazon Redshift console. It is not published as CloudWatch metrics.

Performance data is integrated into the Amazon Redshift console, yielding a richer experience in the following ways:

- Performance data associated with a cluster is displayed contextually when you view a cluster, where you might need it to make decisions about the cluster such as resizing.
- Some performance metrics are displayed in more appropriately scaled units in the Amazon Redshift console as compared to CloudWatch. For example, `WriteThroughput`, is displayed in GB/s (as compared to bytes/s in CloudWatch), which is a more relevant unit for the typical storage space of a node.
- You can easily display performance data for the nodes of a cluster together on the same graph. This way, you can easily monitor the performance of all nodes of a cluster. You can also view performance data for each node.

Amazon Redshift provides performance data (both CloudWatch metrics and query and load data) at no additional charge. Performance data is recorded every minute. You can access historical values of performance data in the Amazon Redshift console. For detailed information about using CloudWatch to access the Amazon Redshift performance data that is exposed as CloudWatch metrics, see [What is CloudWatch?](#) in the *Amazon CloudWatch User Guide*. 

458
Monitoring Amazon Redshift using CloudWatch metrics

Using CloudWatch metrics for Amazon Redshift, you can get information about your cluster's health and performance and see information at the node level. When working with these metrics, keep in mind that each metric has one or more dimensions associated with it. These dimensions tell you what the metric is applicable to, that is the scope of the metric. Amazon Redshift has the following two dimensions:

- Metrics that have a NodeID dimension are metrics that provide performance data for nodes of a cluster. This set of metrics includes leader and compute nodes. Examples of these metrics include CPUUtilization, ReadIOPS, WriteIOPS.
- Metrics that have only a ClusterIdentifier dimension are metrics that provide performance data for clusters. Examples of these metrics include HealthStatus and MaintenanceMode.

Note
In some metric cases, a cluster-specific metric represents an aggregation of node behavior. In these cases, take care in the interpretation of the metric value because the leader node's behavior is aggregated with the compute node.

For general information about CloudWatch metrics and dimensions, see CloudWatch concepts in the Amazon CloudWatch User Guide.

For a further description of CloudWatch metrics for Amazon Redshift, see the following sections.

Topics
- Amazon Redshift metrics (p. 459)
- Dimensions for Amazon Redshift metrics (p. 464)
- Amazon Redshift query and load performance data (p. 465)

Amazon Redshift metrics

The AWS/Redshift namespace includes the following metrics. Unless stated otherwise, metrics are collected at 1-minute intervals.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommitQueueLength</td>
<td>The number of transactions waiting to commit at a given point in time.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>ConcurrencyScalingActiveClusters</td>
<td>The number of concurrency scaling clusters that are actively processing queries at any given time.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>ConcurrencyScalingSeconds</td>
<td>The number of seconds used by concurrency scaling clusters that have active query processing activity.</td>
</tr>
<tr>
<td>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CPUUtilization</td>
<td>The percentage of CPU utilization. For clusters, this metric represents an aggregation of all nodes (leader and compute) CPU utilization values.</td>
</tr>
<tr>
<td></td>
<td>Units: Percent</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td>DatabaseConnections</td>
<td>The number of database connections to a cluster.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>HealthStatus</td>
<td>Indicates the health of the cluster. Every minute the cluster connects to its database and performs a simple query. If it is able to perform this operation successfully, the cluster is considered healthy. Otherwise, the cluster is unhealthy. An unhealthy status can occur when the cluster database is under extremely heavy load or if there is a configuration problem with a database on the cluster.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td>In Amazon CloudWatch, this metric is reported as 1 or 0 whereas in the Amazon Redshift console, this metric is displayed with the words HEALTHY or UNHEALTHY for convenience. When this metric is displayed in the Amazon Redshift console, sampling averages are ignored and only HEALTHY or UNHEALTHY are displayed. In Amazon CloudWatch, values different than 1 and 0 might occur because of sampling issue. Any value below 1 for HealthStatus is reported as 0 (UNHEALTHY).</td>
</tr>
<tr>
<td></td>
<td>Units: Count (1/0) (HEALTHY/UNHEALTHY in the Amazon Redshift console)</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>MaintenanceMode</td>
<td>Indicates whether the cluster is in maintenance mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td>In Amazon CloudWatch, this metric is reported as 1 or 0 whereas in the Amazon Redshift console, this metric is displayed with the words ON or OFF for convenience. When this metric is displayed in the Amazon Redshift console, sampling averages are ignored and only ON or OFF are displayed. In Amazon CloudWatch, values different than 1 and 0 might occur because of sampling issues. Any value greater than 0 for MaintenanceMode is reported as 1 (ON).</td>
</tr>
<tr>
<td></td>
<td>Units: Count (1/0) (ON/OFF in the Amazon Redshift console).</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MaxConfiguredConcurrencyScalingClusters</td>
<td>Maximum number of concurrency scaling clusters configured from the parameter group. For more information, see Amazon Redshift parameter groups (p. 271).</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>NetworkReceiveThroughput</td>
<td>The rate at which the node or cluster receives data.</td>
</tr>
<tr>
<td></td>
<td>Units: Bytes/Second (MB/s in the Amazon Redshift console)</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>NetworkTransmitThroughput</td>
<td>The rate at which the node or cluster writes data.</td>
</tr>
<tr>
<td></td>
<td>Units: Bytes/Second (MB/s in the Amazon Redshift console)</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>PercentageDiskSpaceUsed</td>
<td>The percent of disk space used.</td>
</tr>
<tr>
<td></td>
<td>Units: Percent</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>QueriesCompletedPerSecond</td>
<td>The average number of queries completed per second. Reported in 5-minute intervals.</td>
</tr>
<tr>
<td></td>
<td>Units: Count/Second</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, latency</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, wlmid</td>
</tr>
<tr>
<td>QueryDuration</td>
<td>The average amount of time to complete a query. Reported in 5-minute intervals.</td>
</tr>
<tr>
<td></td>
<td>Units: Microseconds</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID, latency</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, latency</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID, wlmid</td>
</tr>
<tr>
<td>QueryRuntimeBreakdown</td>
<td>The total time queries spent running by query stage. Reported in 5-minute intervals.</td>
</tr>
<tr>
<td></td>
<td>Units: Milliseconds</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID, stage</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, stage</td>
</tr>
<tr>
<td>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>ReadIOPS</strong></td>
<td>The average number of disk read operations per second.</td>
</tr>
<tr>
<td></td>
<td>Units: Count/Second</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td><strong>ReadLatency</strong></td>
<td>The average amount of time taken for disk read I/O operations.</td>
</tr>
<tr>
<td></td>
<td>Units: Seconds</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td><strong>ReadThroughput</strong></td>
<td>The average number of bytes read from disk per second.</td>
</tr>
<tr>
<td></td>
<td>Units: Bytes (GB/s in the Amazon Redshift console)</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td><strong>TotalTableCount</strong></td>
<td>The number of user tables open at a particular point in time. This total doesn't include Amazon Redshift Spectrum tables.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td><strong>WLMQueueLength</strong></td>
<td>The number of queries waiting to enter a workload management (WLM) queue.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, service class</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, QueueName</td>
</tr>
<tr>
<td><strong>WLMQueueWaitTime</strong></td>
<td>The total time queries spent waiting in the workload management (WLM) queue. Reported in 5-minute intervals.</td>
</tr>
<tr>
<td></td>
<td>Units: Milliseconds.</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, QueryPriority</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, wlmid</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, QueueName</td>
</tr>
<tr>
<td>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WLMQueriesCompletedPerSecond</td>
<td>The average number of queries completed per second for a workload management (WLM) queue. Reported in 5-minute intervals.</td>
</tr>
<tr>
<td></td>
<td>Units: Count/Second</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, wlmid</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, QueueName</td>
</tr>
<tr>
<td>WLMQueryDuration</td>
<td>The average length of time to complete a query for a workload management (WLM) queue. Reported in 5-minute intervals.</td>
</tr>
<tr>
<td></td>
<td>Units: Microseconds</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, wlmid</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, QueueName</td>
</tr>
<tr>
<td>WLMRunningQueries</td>
<td>The number of queries running from both the main cluster and concurrency scaling cluster per WLM queue.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, wlmid</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, QueueName</td>
</tr>
<tr>
<td>WriteIOPS</td>
<td>The average number of write operations per second.</td>
</tr>
<tr>
<td></td>
<td>Units: Count/Second</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>WriteLatency</td>
<td>The average amount of time taken for disk write I/O operations.</td>
</tr>
<tr>
<td></td>
<td>Units: Seconds</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>WriteThroughput</td>
<td>The average number of bytes written to disk per second.</td>
</tr>
<tr>
<td></td>
<td>Units: Bytes (GB/s in the Amazon Redshift console)</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
</tbody>
</table>
### Dimensions for Amazon Redshift metrics

Amazon Redshift data can be filtered along any of the dimensions in the table following.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SchemaQuota</td>
<td>The configured quota for a schema.</td>
</tr>
<tr>
<td></td>
<td>Units: Megabytes</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, DatabaseName, SchemaName</td>
</tr>
<tr>
<td></td>
<td>Periodic/Push: Periodic</td>
</tr>
<tr>
<td></td>
<td>Frequency: 5 minutes</td>
</tr>
<tr>
<td></td>
<td>Stop criteria: Schema dropped or quota removed</td>
</tr>
<tr>
<td>NumExceededSchemaQuotas</td>
<td>The number of schemas with exceeded quotas.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td></td>
<td>Periodic/Push: Periodic</td>
</tr>
<tr>
<td></td>
<td>Frequency: 5 minutes</td>
</tr>
<tr>
<td></td>
<td>Stop criteria: N/A</td>
</tr>
<tr>
<td>StorageUsed</td>
<td>The disk or storage space used by a schema.</td>
</tr>
<tr>
<td></td>
<td>Units: Megabytes</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, DatabaseName, SchemaName</td>
</tr>
<tr>
<td></td>
<td>Periodic/Push: Periodic</td>
</tr>
<tr>
<td></td>
<td>Frequency: 5 minutes</td>
</tr>
<tr>
<td></td>
<td>Stop criteria: Schema dropped or quota removed</td>
</tr>
<tr>
<td>PercentageQuotaUsed</td>
<td>The percentage of disk or storage space used relative to the configured schema quota.</td>
</tr>
<tr>
<td></td>
<td>Units: Percent</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, DatabaseName, SchemaName</td>
</tr>
<tr>
<td></td>
<td>Periodic/Push: Periodic</td>
</tr>
<tr>
<td></td>
<td>Frequency: 5 minutes</td>
</tr>
<tr>
<td></td>
<td>Stop criteria: Schema dropped or quota removed</td>
</tr>
</tbody>
</table>

### Dimensions for Amazon Redshift metrics

Amazon Redshift data can be filtered along any of the dimensions in the table following.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>latency</td>
<td>Possible values are as follows:</td>
</tr>
</tbody>
</table>
Amazon Redshift query and load performance data

In addition to the CloudWatch metrics, Amazon Redshift provides query and load performance data. Query and load performance data can be used to help you understand the relation between database performance and cluster metrics. For example, if you notice that a cluster’s CPU spiked, you can find the spike on the cluster CPU graph and see the queries that were running at that time. Conversely, if you are

---

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension</td>
<td>Description</td>
</tr>
<tr>
<td>NodeID</td>
<td>Filters requested data that is specific to the nodes of a cluster. NodeID is either &quot;Leader&quot;, &quot;Shared&quot;, or &quot;Compute-N&quot; where N is 0, 1, ... for the number of nodes in the cluster. &quot;Shared&quot; means that the cluster has only one node, that is the leader node and compute node are combined. Metrics are reported for the leader node and compute nodes only for CPUUtilization, NetworkTransmitThroughput, and ReadIOPS. Other metrics that use the NodeId dimension are reported only for compute nodes.</td>
</tr>
<tr>
<td>ClusterIdentifier</td>
<td>Filters requested data that is specific to the cluster. Metrics that are specific to clusters include HealthStatus, MaintenanceMode, and DatabaseConnections. General metrics for this dimension (for example, ReadIOPS) that are also metrics of nodes represent an aggregate of the node metric data. Take care in interpreting these metrics because they aggregate behavior of leader and compute nodes.</td>
</tr>
<tr>
<td>service class</td>
<td>The identifier for a WLM service class.</td>
</tr>
<tr>
<td>stage</td>
<td>The execution stages for a query. The possible values are as follows:</td>
</tr>
<tr>
<td>wlmid</td>
<td>The identifier for a workload management queue.</td>
</tr>
<tr>
<td>QueryPriority</td>
<td>The priority of the query. Possible values are CRITICAL, HIGHEST, HIGH, NORMAL, LOW, and LOWEST.</td>
</tr>
<tr>
<td>QueueName</td>
<td>The name of the workload management queue.</td>
</tr>
</tbody>
</table>
reviewing a specific query, metric data (like CPU) is displayed in context so that you can understand the query’s impact on cluster metrics.

Query and load performance data are not published as CloudWatch metrics and can only be viewed in the Amazon Redshift console. Query and load performance data are generated from querying with your database’s system tables (for more information, see System tables reference in the Amazon Redshift Developer Guide). You can also generate your own custom database performance queries, but we recommend starting with the query and load performance data presented in the console. For more information about measuring and monitoring your database performance yourself, see Managing performance in the Amazon Redshift Developer Guide.

The following table describes different aspects of query and load data you can access in the Amazon Redshift console.

<table>
<thead>
<tr>
<th>Query/Load data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query summary</td>
<td>A list of queries in a specified time period. The list can be sorted on values such as query ID, query runtime, and status. View this data in the Query monitoring tab of the cluster detail page.</td>
</tr>
</tbody>
</table>
| Query detail    | Provides details on a particular query including:  
• Query properties such as the query ID, type, cluster the query was run on, and runtime.  
• Details such as the status of the query and the number of errors.  
• The SQL statement that was run.  
• An explain plan if available.  
• Cluster performance data during the query execution (for more information, see Viewing query history data (p. 473)). |
| Load summary    | Lists all the loads in a specified time period. The list can be sorted on values such as query ID, query runtime, and status. View this data in the Query monitoring tab of the cluster detail page. |
| Load detail     | Provides details on a particular load operation including:  
• Load properties such as the query ID, type, cluster the query was run on, and runtime.  
• Details such as the status of the load and the number of errors.  
• The SQL statement that was run.  
• A list of loaded files.  
• Cluster performance data during the load operation (for more information, see Viewing query history data (p. 473)). |

Working with performance data in the Amazon Redshift console

In this section, you can find how to view performance data in the Amazon Redshift console, which includes information about cluster and query performance. Additionally, you can create alarms on cluster metrics directly from the Amazon Redshift console.

When you view performance data in the Amazon Redshift console, you view it by cluster. The performance data graphs for a cluster are designed to give you access to data to answer your most
common performance questions. For some performance data (see Monitoring Amazon Redshift using CloudWatch metrics (p. 459)), you can also use CloudWatch to further customize your metrics graphs. For example, you can choose longer times or combine metrics across clusters. For more information about working with the CloudWatch console, see Working with performance metrics in the CloudWatch console (p. 487).

Watch the following video to learn how to monitor, isolate, and optimize your queries using the query monitoring features on the Amazon Redshift console: Query Monitoring with Amazon Redshift.

Topics
• Viewing cluster performance data (p. 467)
• Viewing query history data (p. 473)
• Viewing database performance data (p. 476)
• Viewing workload concurrency and concurrency scaling data (p. 479)
• Viewing queries and loads (p. 481)
• Viewing cluster metrics during load operations (p. 485)
• Analyzing workload performance (p. 485)
• Managing alarms (p. 486)
• Working with performance metrics in the CloudWatch console (p. 487)

Viewing cluster performance data

By using cluster metrics in Amazon Redshift, you can do the following common performance tasks:

• Determine if cluster metrics are abnormal over a specified time range and, if so, identify the queries responsible for the performance hit.
• Check if historical or current queries are impacting cluster performance. If you identify a problematic query, you can view details about it including the cluster performance during the query’s execution. You can use this information in diagnosing why the query was slow and what can be done to improve its performance.

To view performance data
1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the name of a cluster from the list to open its details. The details of the cluster are displayed, which can include Cluster performance, Query monitoring, Databases, Datashares, Schedules, Maintenance, and Properties tabs.
3. Choose the Cluster performance tab for performance information including the following:
   • CPU utilization
   • Percentage disk space used
   • Database connections
   • Health status
   • Query duration
   • Query throughput
   • Concurrency scaling activity

Many more metrics are available. To see the available metrics and choose which are displayed, choose the Preferences icon.
Cluster performance graphs

The following examples show some of the graphs that are displayed in the new Amazon Redshift console.

- **CPU utilization** – Shows the percentage of CPU utilization for all nodes (leader and compute). To find a time when the cluster usage is lowest before scheduling cluster migration or other resource-consuming operations, monitor this chart to see CPU utilization per individual or all of nodes.

- **Maintenance mode** – Shows whether the cluster is in the maintenance mode at a chosen time by using On and Off indicators. You can see the time when the cluster is undergoing maintenance. You can then correlate this time to operations that are done to the cluster to estimate its future downtimes for recurring events.

- **Percentage disk space used** – Shows the percentage of disk space usage per each compute node, and not for the cluster as a whole. You can explore this chart to monitor the disk utilization. Maintenance operations like VACUUM and COPY use intermediate temporary storage space for their sort operations, so a spike in disk usage is expected.
Viewing cluster performance data

- **Read throughput** – Shows the average number of megabytes read from disk per second. You can evaluate this chart to monitor the corresponding physical aspect of the cluster. This throughput doesn't include network traffic between instances in the cluster and its volume.

- **Read latency** – Shows the average amount of time taken for disk read I/O operations per millisecond. You can view the response times for the data to return. When latency is high, it means that the sender spends more time idle (not sending any new packets), which reduces how fast throughput grows.

- **Write throughput** – Shows the average number of megabytes written to disk per second. You can evaluate this metric to monitor the corresponding physical aspect of the cluster. This throughput doesn't include network traffic between instances in the cluster and its volume.
• **Write latency** – Shows the average amount of time in milliseconds taken for disk write I/O operations. You can evaluate the time for the write acknowledgment to return. When latency is high, it means that the sender spends more time idle (not sending any new packets), which reduces how fast throughput grows.

• **Database connections** – Shows the number of database connections to a cluster. You can use this chart to see how many connections are established to the database and find a time when the cluster usage is lowest.

• **Total table count** – Shows the number of user tables open at a particular point in time within a cluster. You can monitor the cluster performance when open table count is high.
• **Health status** – Indicates the health of the cluster as *Healthy* or *Unhealthy*. If the cluster can connect to its database and performs a simple query successfully, the cluster is considered healthy. Otherwise, the cluster is unhealthy. An unhealthy status can occur when the cluster database is under extremely heavy load or if there is a configuration problem with a database on the cluster.

• **Query duration** – Shows the average amount of time to complete a query in microseconds. You can benchmark the data on this chart to measure I/O performance within the cluster and tune its most time-consuming queries if necessary.

• **Query throughput** – Shows the average number of completed queries per second. You can analyze data on this chart to measure database performance and characterize the ability of the system to support a multiuser workload in a balanced way.
• **Query duration per WLM queue** – Shows the average amount of time to complete a query in microseconds. You can benchmark the data on this chart to measure I/O performance per WLM queue and tune its most time-consuming queries if necessary.

• **Query throughput per WLM queue** – Shows the average number of completed queries per second. You can analyze data on this chart to measure database performance per WLM queue.

• **Concurrency scaling activity** – Shows the number of active concurrency scaling clusters. When concurrency scaling is enabled, Amazon Redshift automatically adds additional cluster capacity when you need it to process an increase in concurrent read queries.
Viewing query history data

You can use query history metrics in Amazon Redshift to do the following:

- Isolate and diagnose query performance problems.
- Compare query runtime metrics and cluster performance metrics on the same timeline to see how the two might be related. Doing so helps identify poorly performing queries, look for bottleneck queries, and determine if you need to resize your cluster for your workload.
- Drill down to the details of a specific query by choosing it in the timeline. When Query ID and other properties are displayed in a row below the graph, then you can choose the query to see query details. Details include, for example, the query's SQL statement, execution details, and query plan. For more information, see Viewing query details (p. 482).
- Determine if your load jobs complete successfully and meet your service level agreements (SLAs).

To display query history data

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include Cluster performance, Query monitoring, Databases, DaShares, Schedules, Maintenance, and Properties tabs.
3. Choose the Query monitoring tab for metrics about your queries.
4. In the Query monitoring section, choose the Query history tab.

Using controls on the window, you can toggle between Query list and Cluster metrics.

When you choose Query list, the tab includes the following graphs:

- Query runtime – The query activity on a timeline. Use this graph to see which queries are running in the same timeframe. Choose a query to view more query execution details. The x-axis shows the selected period. You can filter the graphed queries by running, completed, loads, and so on. Each bar represents a query, and the length of the bar represents its runtime from the start of the bar to the end. The queries can include SQL data manipulation statements (such as SELECT, INSERT, DELETE) and loads (such as COPY). By default, the top 100 longest running queries are shown for the selected time period.
- Queries and loads – List of queries and loads that ran on the cluster. The window includes an option to Terminate query if a query is currently running.
When you choose **Cluster metrics**, the tab includes the following graphs:

- **Query runtime** – The query activity on a timeline. Use this graph to see which queries are running in the same timeframe. Choose a query to view more query execution details.
- **CPU utilization** – The CPU utilization of the cluster by leader node and average of compute nodes.
- **Storage capacity used** – The percent of the storage capacity used.
- **Active database connections** – The number of active database connections to the cluster.

Consider the following when working with the query history graphs:

- Choose a bar that represents a specific query on the **Query runtime** chart to see details about that query. You can also, choose a query ID on **Queries and loads** list to see its details.
- You can swipe to select a section of the **Query runtime** chart to zoom in to display a specific time period.
- On the **Query runtime** chart, to have all data considered by your chosen filter, page forward through all pages listed on the **Queries and loads** list.
- You can change which columns and the number of rows displayed on the **Queries and loads** list using the preferences window displayed by the **settings gear icon**.
- The **Queries and loads** list can also be displayed by navigating from the left navigator **Queries** icon, **Queries and loads**. For more information, see Viewing queries and loads (p. 481).

**Query history graphs**

The following examples show graphs that are displayed in the new Amazon Redshift console.

- **Query runtime**
  
  The query activity on a timeline. Use this graph to see which queries are running in the same timeframe. Choose a query to view more query execution details.

- **Queries and loads**
• CPU utilization

![CPU utilization graph](image)

The CPU utilization of the cluster by leader node and average of compute nodes.

• Storage capacity used

![Storage capacity used graph](image)

The percent of the storage capacity used.

• Active database connections
Viewing database performance data

You can use database performance metrics in Amazon Redshift to do the following:

- Analyze the time spent by queries by processing stages. You can look for unusual trends in the amount of time spent in a stage.
- Analyze the number of queries, duration, and throughput of queries by duration ranges (short, medium, long).
- Look for trends in the about of query wait time by query priority (Lowest, Low, Normal, High, Highest, Critical).
- Look for trends in the query duration, throughput, or wait time by WLM queue.

To display database performance data

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details. The details of the cluster are displayed, including Cluster performance, Query monitoring, Databases, Datashares, Schedules, Maintenance, and Properties tabs.
3. Choose the Query monitoring tab for metrics about your queries.
4. In the Query monitoring section, choose Database performance tab.

Using controls on the window, you can toggle between Cluster metrics and WLM queue metrics.

When you choose Cluster metrics, the tab includes the following graphs:

- **Workload execution breakdown** – The time used in query processing stages.
- **Queries by duration range** – The number of short, medium, and long queries.
- **Query throughput** – The average number of queries completed per second.
- **Query duration** – The average amount of time to complete a query.
- **Average queue wait time by priority** – The total time queries spent waiting in the WLM queue by query priority.

When you choose WLM queue metrics, the tab includes the following graphs:

- **Query duration by queue** – The average query duration by WLM queue.
• **Query throughput by queue** – The average number of queries completed per second by WLM queue.

• **Query wait time by queue** – The average duration of queries spent waiting by WLM queue.

### Database performance graphs

The following examples show graphs that are displayed in the new Amazon Redshift console.

- **Workload execution breakdown**

- **Queries by duration range**

- **Query throughput**
• **Query duration**

![Query Duration Graph]

The average amount of time to complete a query.

• **Average queue wait time by priority**

![Average queue wait time by priority Graph]

The total time queries spent waiting in the WLM queue by query priority.

• **Query duration by queue**

![Query Duration by queue Graph]

The average query duration by WLM queue.

• **Query throughput by queue**

![Query throughput by queue Graph]
By using concurrency scaling metrics in Amazon Redshift, you can do the following:

- Analyze whether you can reduce the number of queued queries by enabling concurrency scaling. You can compare by WLM queue or for all WLM queues.
- View concurrency scaling activity in concurrency scaling clusters. This can tell you if concurrency scaling is limited by the `max_concurrency_scaling_clusters`. If so, you can choose to increase the `max_concurrency_scaling_clusters` in the DB parameter.
- View the total usage of concurrency scaling summed across all concurrency scaling clusters.

**To display concurrency scaling data**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **Clusters**, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include **Cluster performance**, **Query monitoring**, **Databases**, **Datashares**, **Schedules**, **Maintenance**, and **Properties** tabs.
3. Choose the **Query monitoring** tab for metrics about your queries.
4. In the **Query monitoring** section, choose **Workload concurrency** tab.

The tab includes the following graphs:

- **Queued vs. Running queries on the cluster** – The number of queries running (from the main cluster and concurrency scaling cluster) compared to the number of queries waiting in all WLM queues in the cluster.

- **Queued vs. Running queries per queue** – The number of queries running (from the main cluster and concurrency scaling cluster) compared to the number or queries waiting in each WLM queue.

- **Concurrency scaling activity** – The number of concurrency scaling clusters that are actively processing queries.

- **Concurrency scaling usage** – The usage of concurrency scaling clusters that have active query processing activity.

### Workload concurrency graphs

The following examples show graphs that are displayed in the new Amazon Redshift console.

- **Queued vs. Running queries on the cluster**

- **Queued vs. Running queries per queue**

- **Concurrency scaling activity**
Viewing queries and loads

The Amazon Redshift console provides information about queries and loads that run in the database. You can use this information to identify and troubleshoot queries that take a long time to process and that create bottlenecks preventing other queries from processing efficiently. You can use the queries information in the Amazon Redshift console to monitor query processing.

To display query performance data

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Queries and loads to display the list of queries for your account.

By default, the list displays queries for all your clusters over the past 24 hours. You can change the scope of the displayed date in the console.

Important
The Queries and loads list displays the longest running queries in the system, up to 100 queries.
Ending a running query

You can also use the Queries page to end a query that is currently in progress.

**Note**

The ability to terminate queries and loads in the Amazon Redshift console requires specific permission. If you want users to be able to terminate queries and loads, make sure to add the `redshift:CancelQuerySession` action to your AWS Identity and Access Management (IAM) policy. This requirement applies whether you select the **Amazon Redshift Read Only** AWS managed policy or create a custom policy in IAM. Users who have the **Amazon Redshift Full Access** policy already have the necessary permission to terminate queries and loads. For more information about actions in IAM policies for Amazon Redshift, see Managing access to resources (p. 337).

**To end a running query**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at [https://console.aws.amazon.com/redshift/](https://console.aws.amazon.com/redshift/).
2. On the navigation menu, choose **Queries and loads** to display the list of queries for your account.
3. Choose the running query that you want to end in the list, and then choose **Terminate query**.

Viewing query details

You can analyze query details on the Amazon Redshift console. With a query identifier, you can view details of a query. Details can include, for example, the query's completion status, duration, SQL statement and whether it's a user query or one that was rewritten by Amazon Redshift. A *user query* is a query that is submitted to Amazon Redshift, either from an SQL client or generated by a business intelligence tool. Amazon Redshift might rewrite the query to optimize it, and this can result in multiple rewritten queries. Although the process is done by Amazon Redshift, you see the rewritten queries on the query details page along with the user query.

**To view a query**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at [https://console.aws.amazon.com/redshift/](https://console.aws.amazon.com/redshift/).
2. On the navigation menu, choose **Queries and loads** to display the list of queries for your account.
3. Choose the **Query** identifier in the list to display **Query details**.

   The **Query details** page includes **Query details** and **Query plan** tabs with metrics about the query.

   Metrics include details about a query such as start time, query ID, status, and duration. Other details include whether a query ran on a main cluster or a concurrency scaling cluster, and if it's a parent or rewritten query.

Analyzing query execution

**To analyze a query**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at [https://console.aws.amazon.com/redshift/](https://console.aws.amazon.com/redshift/).
2. On the navigation menu, choose **Queries and loads** to display the list of queries for your account.
3. Choose the **Query** identifier in the list to display **Query details**.
The **Query details** page includes **Query details** and **Query plan** tabs with metrics about the query.

**Note**
You can also navigate to the **Query details** page from a **Cluster details** page, **Query history** tab when you drill down into a query in a **Query runtime** graph.

The **Query details** page contains the following sections:

- A list of **Rewritten queries**, as shown in the following screenshot.

![Rewritten queries](image)

- A **Query details** section, as shown in the following screenshot.

![Query details](image)

- A **Query details** tab that contains the SQL that was run and **Execution details** about the run.
- A **Query plan** tab that contains the **Query plan** steps and other information about the query plan. This table also contains graphs about the cluster when the query ran.
- **Cluster health status**

  Cluster health status
  Cluster health during the workload.

- **CPU utilization**
Viewing cluster performance as queries run

To display cluster performance as queries run

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.

2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include Cluster performance, Query monitoring, Databases, Datashares, Schedules, Maintenance, and Properties tabs.

3. Choose the Query monitoring tab for more details.

For more information, see Viewing query history data (p. 473).
Viewing cluster metrics during load operations

When you view cluster performance during load operations, you can identify queries that are consuming resources and act to mitigate their effect. You can terminate a load if you don't want it to run to completion.

**Note**

The ability to terminate queries and loads in the Amazon Redshift console requires specific permission. If you want users to be able to terminate queries and loads, make sure to add the `redshift:CancelQuerySession` action to your AWS Identity and Access Management (IAM) policy. This requirement applies whether you select the [Amazon Redshift Read Only](https://aws.amazon.com/redshift/) AWS-managed policy or create a custom policy in IAM. Users who have the [Amazon Redshift Full Access](https://aws.amazon.com/redshift/) policy already have the necessary permission to terminate queries and loads. For more information about actions in IAM policies for Amazon Redshift, see [Managing access to resources](https://aws.amazon.com/redshift/) (p. 337).

**To display cluster performance during load operations**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at [https://console.aws.amazon.com/redshift/](https://console.aws.amazon.com/redshift/).
2. On the navigation menu, choose **Clusters**, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include **Cluster performance**, **Query monitoring**, **Databases**, **Datashares**, **Schedules**, **Maintenance**, and **Properties** tabs.
3. Choose the **Query monitoring** tab for more details.
4. In the **Queries and loads** section, choose **Loads** to view the load operations of a cluster. If a load is running, you can end it by choosing **Terminate query**.

Analyzing workload performance

You can get a detailed view of your workload's performance by looking at the Workload execution breakdown chart in the console. We build the chart with data provided by the QueryRuntimeBreakdown metric. With this chart, you can see how much time your queries spend in the various processing stages, such as waiting and planning.

**Note**

The Workload execution breakdown chart isn't shown for single-node clusters.

The following list of metrics describes the various processing stages:

- `QueryPlanning`: Time spent parsing and optimizing SQL statements.
- `QueryWaiting`: Time spent waiting in the workload management (WLM) queue.
- `QueryExecutingRead`: Time spent running read queries.
- `QueryExecutingInsert`: Time spent running insert queries.
- `QueryExecutingDelete`: Time spent running delete queries.
- `QueryExecutingUpdate`: Time spent running update queries.
- `QueryExecutingCtas`: Time spent running CREATE TABLE AS queries.
- `QueryExecutingUnload`: Time spent running unload queries.
- `QueryExecutingCopy`: Time spent running copy queries.

For example, the following graph in the Amazon Redshift console shows the amount of time that queries have spent in the plan, wait, read, and write stages. You can combine the findings from this graph...
with other metrics for further analysis. In some cases, your graph might show that queries with a short duration (as measured by the `QueryDuration` metric) are spending a long time in the wait stage. In these cases, you can increase the WLM concurrency rate for a particular queue to increase throughput.

Following, is an example of the workload execution breakdown chart. In the chart, the y-axis value is the average duration of each stage at the specified time shown as a stacked bar graph.

The following diagram illustrates how Amazon Redshift aggregates query processing for concurrent sessions.

To display the cluster workload breakdown chart

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Clusters, then choose the cluster name from the list to open its details. The details of the cluster are displayed, which can include Cluster performance, Query monitoring, Databases, Datashares, Schedules, Maintenance, and Properties tabs.
3. Choose the Query monitoring tab for metrics about your queries.
4. In the Query monitoring section, choose Database performance, and choose Cluster metrics.

The following metrics are graphed for the chosen time range as a stacked bar chart:

- Plan time
- Wait time
- Commit time
- Execution time

Managing alarms

Alarms you create in the Amazon Redshift console are CloudWatch alarms. They are useful because they help you make proactive decisions about your cluster and its databases. You can set one or more alarms on any of the metrics listed in Monitoring Amazon Redshift using CloudWatch metrics (p. 459). For example, setting an alarm for high CPUUtilization on a cluster node helps indicate when the node is...
overutilized. Likewise, setting an alarm for low CPUUtilization on a cluster node helps indicate when
the node is underutilized.

From Actions, you can modify or delete alarms. You can also create a chime or slack alert to send an
alert from CloudWatch to Slack or Amazon Chime by specifying a Slack or Amazon Chime webhook URL.

In this section, you can find how to create an alarm using the Amazon Redshift console. You can create
an alarm using the CloudWatch console or any other way you work with metrics, such as with the AWS
CLI or an AWS SDK.

**To create a CloudWatch alarm with the Amazon Redshift console**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://
   console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Alarms, then choose Create alarm.
3. On the Create alarm page, enter the properties to create a CloudWatch alarm.
4. Choose Create alarm.

**Working with performance metrics in the CloudWatch console**

When working with Amazon Redshift metrics in the CloudWatch console, keep a couple of things in
mind:

- Query and load performance data is only available in the Amazon Redshift console.
- Some Metrics in the CloudWatch have different units than those used in the Amazon Redshift console.
  For example, WriteThroughput is displayed in GB/s (as compared to Bytes/s in CloudWatch), which is
  a more relevant unit for the typical storage space of a node.

When working with Amazon Redshift metrics in the CloudWatch console, command line tools, or an
Amazon SDK, keep these concepts in mind:

1. First, specify the metric dimension to work with. A dimension is a name-value pair that helps you
   to uniquely identify a metric. The dimensions for Amazon Redshift are ClusterIdentifier
   and NodeID. In the CloudWatch console, the Redshift Cluster and Redshift Node views
   are provided to easily select cluster and node-specific dimensions. For more information about
dimensions, see Dimensions in the CloudWatch Developer Guide.
2. Then, specify the metric name, such as ReadIOPS.

The following table summarizes the types of Amazon Redshift metric dimensions that are available to
you. Depending on the metric, data is available in either 1-minute or 5-minute intervals at no charge. For
more information, see Amazon Redshift metrics (p. 459).

<table>
<thead>
<tr>
<th>CloudWatch namespace</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS/Redshift</td>
<td>NodeID</td>
<td>Filters requested data that is specific to the nodes of a cluster. NodeID is either &quot;Leader&quot;, &quot;Shared&quot;, or &quot;Compute-N&quot; where N is 0, 1, ... for the number of nodes in the cluster. &quot;Shared&quot; means that the cluster has only one node, that is the leader node and compute node are combined.</td>
</tr>
</tbody>
</table>
Working with performance metrics in the CloudWatch console

<table>
<thead>
<tr>
<th>CloudWatch namespace</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ClusterIdentifier</td>
<td>Filters requested data that is specific to the cluster. Metrics that are specific to clusters include HealthStatus, MaintenanceMode, and DatabaseConnections. General metrics for this dimension (for example, ReadIOPS) that are also metrics of nodes represent an aggregate of the node metric data. Take care in interpreting these metrics because they aggregate behavior of leader and compute nodes.</td>
</tr>
</tbody>
</table>

Working with gateway and volume metrics is similar to working with other service metrics. Many of the common tasks are outlined in the CloudWatch documentation, including the following:

- View available metrics
- Get statistics for a metric
- Creating CloudWatch alarms
Amazon Redshift events

Topics
- Overview (p. 489)
- Viewing events using the console (p. 489)
- Viewing events using the AWS SDK for Java (p. 489)
- Viewing events using the AWS CLI and Amazon Redshift API (p. 491)
- Amazon Redshift event notifications (p. 491)

Overview

Amazon Redshift tracks events and retains information about them for a period of several weeks in your AWS account. For each event, Amazon Redshift reports information such as the date the event occurred, a description, the event source (for example, a cluster, a parameter group, or a snapshot), and the source ID.

Amazon Redshift provides notification in advance for some events. These events have an event category of pending. For example, we send an advance notification if a hardware update is required for one of the nodes in your cluster. You can subscribe to pending events the same as other Amazon Redshift events. For more information, see Subscribing to Amazon Redshift event notifications (p. 492).

You can use the Amazon Redshift Management Console, the Amazon Redshift API, or the AWS SDKs to obtain event information. You can obtain a list of all events, or you can apply filters, such as event duration or start and end date, to obtain events information for a specific period.

You can also obtain events that were generated by a specific source type, such as cluster events or parameter group events. The Source column shows the resource name and resource type that triggers a given action.

You can create Amazon Redshift event notification subscriptions that specify a set of event filters. When an event occurs that matches the filter criteria, Amazon Redshift uses Amazon Simple Notification Service to actively inform you that the event has occurred.

For a list of Amazon Redshift events by source type and category, see the section called "Amazon Redshift event categories and event messages" (p. 493)

Viewing events using the console

To view events

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Events.

Viewing events using the AWS SDK for Java

The following example lists the events for a specified cluster and specified event source type. The example shows how to use pagination.
For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 449). You need to update the code and specify a cluster identifier and event source type.

Example

```java
/**
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 * This file is licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License. A copy of
 * the License is located at
 * http://aws.amazon.com/apache2.0/
 * This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR
 * CONDITIONS OF ANY KIND, either express or implied. See the License for the
 * specific language governing permissions and limitations under the License.
 */

package com.amazonaws.services.redshift;
import java.util.Date;
import java.io.IOException;
import com.amazonaws.services.redshift.model.*;
public class ListEvents {
  public static AmazonRedshift client;
  public static String clusterIdentifier = "***provide cluster identifier***";
  public static String eventSourceType = "cluster"; // e.g. cluster-snapshot
  public static void main(String[] args) throws IOException {
    try {
      listEvents();
    } catch (Exception e) {
      System.err.println("Operation failed: " + e.getMessage());
    }
  }

  private static void listEvents() {
    long oneWeeksAgoMilli = (new Date()).getTime() - (7L*24L*60L*60L*1000L);
    Date oneWeekAgo = new Date();
    oneWeekAgo.setTime(oneWeeksAgoMilli);
    String marker = null;
```

490
You can use the following Amazon Redshift CLI operation to manage events.

- describe-events

Amazon Redshift provides the following API to view events.

- DescribeEvents

Amazon Redshift event notifications

Overview

Amazon Redshift uses the Amazon Simple Notification Service (Amazon SNS) to communicate notifications of Amazon Redshift events. You enable notifications by creating an Amazon Redshift event subscription. In the Amazon Redshift subscription, you specify a set of filters for Amazon Redshift events and an Amazon SNS topic. Whenever an event occurs that matches the filter criteria, Amazon Redshift publishes a notification message to the Amazon SNS topic. Amazon SNS then transmits the message to any Amazon SNS consumers that have an Amazon SNS subscription to the topic. The messages sent to the Amazon SNS consumers can be in any form supported by Amazon SNS for an AWS Region, such as an email, a text message, or a call to an HTTP endpoint. For example, all regions support email notifications, but SMS notifications can only be created in the US East (N. Virginia) Region.
When you create an event notification subscription, you specify one or more event filters. Amazon Redshift sends notifications through the subscription any time an event occurs that matches all of the filter criteria. The filter criteria include source type (such as cluster or snapshot), source ID (such as the name of a cluster or snapshot), event category (such as Monitoring or Security), and event severity (such as INFO or ERROR).

You can easily turn off notification without deleting a subscription by setting the Enabled radio button to No in the AWS Management Console or by setting the Enabled parameter to false using the Amazon Redshift CLI or API.

Billing for Amazon Redshift event notification is through the Amazon Simple Notification Service (Amazon SNS). Amazon SNS fees apply when using event notification; for more information on Amazon SNS billing, go to Amazon Simple Notification Service pricing.

You can also view Amazon Redshift events that have occurred by using the management console. For more information, see Amazon Redshift events (p. 489).

Topics
- Subscribing to Amazon Redshift event notifications (p. 492)

### Subscribing to Amazon Redshift event notifications

You can create an Amazon Redshift event notification subscription so you can be notified when an event occurs for a given cluster, snapshot, security group, or parameter group. The simplest way to create a subscription is with the AWS Management Console. For information on creating an Amazon SNS topic and subscribing to it, see Getting started with Amazon SNS.

You can create an Amazon Redshift event notification subscription so you can be notified when an event occurs for a given cluster, snapshot, security group, or parameter group. The simplest way to create a subscription is with the AWS Management Console. If you choose to create event notification subscriptions using the CLI or API, you must create an Amazon Simple Notification Service topic and subscribe to that topic with the Amazon SNS console or Amazon SNS API. You will also need to retain the Amazon Resource Name (ARN) of the topic because it is used when submitting CLI commands or API actions. For information on creating an Amazon SNS topic and subscribing to it, see Getting started with Amazon SNS.

An Amazon Redshift event subscription can specify these event criteria:

- Source type, the values are cluster, snapshot, parameter-groups, and security-groups.
- Source ID of a resource, such as my-cluster-1 or my-snapshot-20130823. The ID must be for a resource in the same AWS Region as the event subscription.
- Event category, the values are Configuration, Management, Monitoring, and Security.
- Event severity, the values are INFO or ERROR.

The event criteria can be specified independently, except that you must specify a source type before you can specify source IDs in the console. For example, you can specify an event category without having to specify a source type, source ID, or severity. While you can specify source IDs for resources that are not of the type specified in source type, no notifications will be sent for events from those resources. For example, if you specify a source type of cluster and the ID of a security group, none of the events raised by that security group would match the source type filter criteria, so no notifications would be sent for those events.

Amazon Redshift sends a notification for any event that matches all criteria specified in a subscription. Some examples of the sets of events returned:
• Subscription specifies a source type of cluster, a source ID of my-cluster-1, a category of Monitoring, and a severity of ERROR. The subscription will send notifications for only monitoring events with a severity of ERROR from my-cluster-1.

• Subscription specifies a source type of cluster, a category of Configuration, and a severity of INFO. The subscription will send notifications for configuration events with a severity of INFO from any Amazon Redshift cluster in the AWS account.

• Subscription specifies a category of Configuration, and a severity of INFO. The subscription will send notifications for configuration events with a severity of INFO from any Amazon Redshift resource in the AWS account.

• Subscription specifies a severity of ERROR. The subscription will send notifications for all events with a severity of ERROR from any Amazon Redshift resource in the AWS account.

If you delete or rename an object whose name is referenced as a source ID in an existing subscription, the subscription will remain active, but will have no events to forward from that object. If you later create a new object with the same name as is referenced in the subscription source ID, the subscription will start sending notifications for events from the new object.

Amazon Redshift publishes event notifications to an Amazon SNS topic, which is identified by its Amazon Resource Name (ARN). When you create an event subscription using the Amazon Redshift console, you can either specify an existing Amazon SNS topic, or request that the console create the topic when it creates the subscription. All Amazon Redshift event notifications sent to the Amazon SNS topic are in turn transmitted to all Amazon SNS consumers that are subscribed to that topic. Use the Amazon SNS console to make changes to the Amazon SNS topic, such as adding or removing consumer subscriptions to the topic. For more information about creating and subscribing to Amazon SNS topics, go to Getting started with Amazon Simple Notification Service.

The following section lists all categories and events that you can be notified of. It also provides information about subscribing to and working with Amazon Redshift event subscriptions.

Amazon Redshift event categories and event messages

This section shows the event IDs and categories for each Amazon Redshift source type.

The following table shows the event category and a list of events when a cluster is the source type.

<table>
<thead>
<tr>
<th>Amazon Redshift category</th>
<th>Event ID</th>
<th>Event severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1000</td>
<td>INFO</td>
<td>The parameter group [parameter group name] was updated at [time]. If you changed only dynamic parameters, associated clusters are being modified now. If you changed static parameters, all updates, including dynamic parameters, will be applied when you reboot the associated clusters.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1001</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster [cluster name] was modified to use parameter group [parameter group name] at [time].</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1500</td>
<td>ERROR</td>
<td>The Amazon VPC [VPC name] does not exist. Your configuration changes for cluster [cluster</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1501</td>
<td>ERROR</td>
<td>The customer subnets [subnet name] you specified for Amazon VPC [VPC name] do not exist or are invalid. Your configuration changes for cluster [cluster name] were not applied. Please visit the AWS Management Console to correct the issue.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1502</td>
<td>ERROR</td>
<td>Subnets in cluster subnet group [subnet group name] have no available IP addresses. Cluster [cluster name] could not be created.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1503</td>
<td>ERROR</td>
<td>The Amazon VPC [VPC name] has no internet gateway attached to it. Your configuration changes for cluster [cluster name] were not applied. Please visit the AWS Management Console to correct the issue.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1504</td>
<td>ERROR</td>
<td>The HSM for cluster [cluster name] is unreachable.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1505</td>
<td>ERROR</td>
<td>The HSM for cluster [cluster name] cannot be registered. Try a different configuration.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1506</td>
<td>ERROR</td>
<td>Amazon Redshift exceeded your account's elastic network interface limit. Delete up to [maximum number of elastic network interfaces] elastic network interfaces or request a limit increase of the number of network interfaces per AWS Region with EC2.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1509</td>
<td>ERROR</td>
<td>The Amazon Redshift cluster [cluster name] can't be created because your account's VPC endpoint limit has been reached. Delete unused VPC endpoints or request an increase in the limit of VPC endpoints.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For more information, see VPC endpoints in the Amazon VPC User Guide.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1510</td>
<td>ERROR</td>
<td>We have detected that the attempt to load sample data on your Amazon Redshift cluster [cluster name] didn't succeed. To load sample data, first configure your VPC to have access to Amazon S3 buckets, then create a new cluster and load sample data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For more information, see Enabling enhanced VPC routing in the Amazon Redshift Cluster Management Guide.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2000</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster: [cluster name] has been created and is ready for use.</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2001</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster [cluster name] was deleted at [time]. A final snapshot [was / was not] saved.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2002</td>
<td>INFO</td>
<td>Your VPC security group [security group name] was updated at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2003</td>
<td>INFO</td>
<td>Maintenance started on your Amazon Redshift cluster [cluster name] at [time]. The cluster may not be available during maintenance.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2004</td>
<td>INFO</td>
<td>Maintenance completed on your Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2006</td>
<td>INFO</td>
<td>A resize for your Amazon Redshift cluster [cluster name] was started at [time]. Your cluster will be in read-only mode during the resize operation.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2007</td>
<td>INFO</td>
<td>The resize for your Amazon Redshift cluster [cluster name] is in progress. Your cluster is in read-only mode.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2008</td>
<td>INFO</td>
<td>Your restore operation to create a new Amazon Redshift cluster [cluster name] snapshot [snapshot name] was started at [time]. To monitor restore progress, please visit the AWS Management Console.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2013</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster [cluster name] was renamed at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2014</td>
<td>INFO</td>
<td>A table restore request for Amazon Redshift cluster [cluster name] has been received.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2015</td>
<td>INFO</td>
<td>Table restore was cancelled for Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2016</td>
<td>INFO</td>
<td>Replacement of your Amazon Redshift cluster [cluster name] was started at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2017</td>
<td>INFO</td>
<td>Customer initiated maintenance started on your Amazon Redshift cluster [cluster name] at [time]. The cluster may not be available during maintenance.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2018</td>
<td>INFO</td>
<td>Customer initiated maintenance completed on your Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2019</td>
<td>ERROR</td>
<td>Customer initiated maintenance failed on your Amazon Redshift cluster [cluster name] at [time]. Returning the cluster back to its original state.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2020</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster [cluster name]'s track has been modified from [from track] to [to track].</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2021</td>
<td>ERROR</td>
<td>The [operation] of Amazon Redshift cluster [cluster name] did not succeed while acquiring capacity from our capacity pool. We are working to acquire capacity but for now, we have cancelled your request. Delete this cluster and retry later.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2022</td>
<td>ERROR</td>
<td>The [operation] of Amazon Redshift cluster [cluster name] did not succeed while acquiring capacity from our capacity pool. We are working to acquire capacity but for now, we have cancelled your request. Capacity is available in [alternative Availability Zones]. Delete this cluster and retry in an alternative Availability Zone.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2023</td>
<td>ERROR</td>
<td>We have detected hardware failure on your single node Amazon Redshift cluster [cluster name], which may have resulted in failed queries or intermittent availability of the cluster. Replacing the cluster did not succeed while acquiring capacity from our capacity pool. You will need to restore a new cluster from a snapshot. Delete this cluster, select the latest available snapshot, and restore a new cluster from that snapshot. This will automatically provision you on healthy hardware.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2024</td>
<td>ERROR</td>
<td>We have detected hardware failure on your single node Amazon Redshift cluster [cluster name], which may have resulted in failed queries or intermittent availability of the cluster. Replacing cluster did not succeed while acquiring capacity from our capacity pool. Capacity is available in Availability Zone: [alternative Availability Zones]. Delete this cluster, select the latest available snapshot, and restore a new cluster from that snapshot. This will automatically provision you on healthy hardware.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3011</td>
<td>INFO</td>
<td>Elastic resize for Amazon Redshift cluster [cluster name] started at [time]. We will hold the database connections during resize. Some queries and connections may be terminated or timed out during this operation.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3012</td>
<td>INFO</td>
<td>We have received an elastic resize request for the cluster [cluster name] started at [time]. We will provide an event notification when resize begins.</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Pending</td>
<td>REDSHIFT-EVENT-2025</td>
<td>INFO</td>
<td>Your database for cluster &lt;cluster name&gt; will be updated between &lt;start time&gt; and &lt;end time&gt;. Your cluster will not be accessible. Plan accordingly.</td>
</tr>
<tr>
<td>Pending</td>
<td>REDSHIFT-EVENT-2026</td>
<td>INFO</td>
<td>Your cluster &lt;cluster name&gt; will be updated between &lt;start time&gt; and &lt;end time&gt;. Your cluster will not be accessible. Plan accordingly.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-2050</td>
<td>INFO</td>
<td>A hardware issue was detected on Amazon Redshift cluster [cluster name]. A replacement request was initiated at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3000</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster [cluster name] was rebooted at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3001</td>
<td>INFO</td>
<td>A node on your Amazon Redshift cluster: [cluster name] was automatically replaced at [time], and your cluster is operating normally.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3002</td>
<td>INFO</td>
<td>The resize for your Amazon Redshift cluster [cluster name] is complete and your cluster is available for reads and writes. The resize was initiated at [time] and took [hours] hours to complete.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3003</td>
<td>INFO</td>
<td>Amazon Redshift cluster [cluster name] was successfully created from snapshot [snapshot name] and is available for use.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3007</td>
<td>INFO</td>
<td>Your Amazon Redshift snapshot [snapshot name] was copied successfully from [source AWS Region] to [destination AWS Region] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3008</td>
<td>INFO</td>
<td>Table restore started for Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3009</td>
<td>INFO</td>
<td>Table restore completed successfully for Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3010</td>
<td>ERROR</td>
<td>Table restore failed for Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3013</td>
<td>ERROR</td>
<td>The requested elastic resize operation for Amazon Redshift cluster [cluster name] failed at [time] due to [reason].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3014</td>
<td>INFO</td>
<td>Amazon Redshift rebooted cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3500</td>
<td>ERROR</td>
<td>The resize for your Amazon Redshift cluster [cluster name] failed. The resize will be automatically retried in a few minutes.</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3501</td>
<td>ERROR</td>
<td>Your restore operation to create Amazon Redshift cluster [cluster name] from snapshot [snapshot name] failed at [time]. Please retry your operation.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3504</td>
<td>ERROR</td>
<td>The Amazon S3 bucket [bucket name] is not valid for logging for cluster [cluster name].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3505</td>
<td>ERROR</td>
<td>The Amazon S3 bucket [bucket name] does not have the correct IAM policies for cluster [cluster name].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3506</td>
<td>ERROR</td>
<td>The Amazon S3 bucket [bucket name] does not exist. Logging cannot continue for cluster [cluster name].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3507</td>
<td>ERROR</td>
<td>The Amazon Redshift cluster [cluster name] cannot be created using EIP [IP address]. This EIP is already in use.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3508</td>
<td>ERROR</td>
<td>The Amazon Redshift cluster [cluster name] cannot be created using EIP [IP address]. The EIP cannot be found.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3509</td>
<td>ERROR</td>
<td>Cross-region snapshot copy is not enabled for cluster [cluster name].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3510</td>
<td>ERROR</td>
<td>Table restore failed to start for Amazon Redshift cluster [cluster name] at [time]. Reason: [reason].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3511</td>
<td>ERROR</td>
<td>Table restore failed for Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3512</td>
<td>ERROR</td>
<td>Amazon Redshift cluster [cluster name] has failed due to a hardware issue. The cluster is being automatically restored from the latest snapshot [snapshot name] created at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3513</td>
<td>ERROR</td>
<td>Amazon Redshift cluster [cluster name] has failed due to a hardware issue. The cluster is being automatically restored from the latest snapshot [snapshot name] created at [time]. Any database changes made after this time will need to be resubmitted.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3514</td>
<td>ERROR</td>
<td>Amazon Redshift cluster [cluster name] has failed due to a hardware issue. The cluster is being placed in hardware failure status. Please delete the cluster and restore from the latest snapshot [snapshot name] created at [time].</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3515</td>
<td>ERROR</td>
<td>Amazon Redshift cluster [cluster name] has failed due to a hardware issue. The cluster is being placed in hardware failure status. Please delete the cluster and restore from the latest snapshot [snapshot name] created at [time]. Any database changes made after this time will need to be resubmitted.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3516</td>
<td>ERROR</td>
<td>Amazon Redshift cluster [cluster name] has failed due to a hardware issue and there are no backups for the cluster. The cluster is being placed in hardware failure status and can be deleted.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3519</td>
<td>INFO</td>
<td>Cluster [cluster name] began restart at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3520</td>
<td>INFO</td>
<td>Cluster [cluster name] completed restart at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3521</td>
<td>INFO</td>
<td>We detected a connectivity issue on the cluster '[cluster name]'. An automated diagnostics check has been initiated at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3522</td>
<td>INFO</td>
<td>Recovery action on '[cluster name]' cluster failed at [time]. The Amazon Redshift team is working on a solution.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3533</td>
<td>ERROR</td>
<td>Cluster resize on '[cluster name]' was cancelled at [time]. The operation was cancelled because [reason]. [action needed].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3534</td>
<td>INFO</td>
<td>The elastic resize for Amazon Redshift cluster '[cluster name]' completed at [time]. The cluster is now available for read and write operations while we transfer data. Some queries may take longer to finish until data transfer is complete.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3537</td>
<td>INFO</td>
<td>Cluster '[cluster name]' data transfer completed at [time in UTC].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3600</td>
<td>INFO</td>
<td>The requested resize operation for Amazon Redshift cluster '[cluster name]' was cancelled in the past. Rollback was completed at [time].</td>
</tr>
<tr>
<td>Pending</td>
<td>REDSHIFT-EVENT-3601</td>
<td>INFO</td>
<td>A node on your cluster &lt;cluster name&gt; will be replaced between &lt;start time&gt; and &lt;end time&gt;. You can't defer this maintenance. Plan accordingly.</td>
</tr>
<tr>
<td>Pending</td>
<td>REDSHIFT-EVENT-3602</td>
<td>INFO</td>
<td>A node on your cluster &lt;cluster name&gt; is scheduled to be replaced between &lt;start time&gt; and &lt;end time&gt;. Your cluster will not be accessible. Plan accordingly.</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3603</td>
<td>INFO</td>
<td>The restore operation to create cluster [cluster name] from snapshot [snapshot name] failed due to an internal error. The cluster is being placed in incompatible restore status and can be deleted. Try to restore the snapshot into a cluster with a different configuration.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3614</td>
<td>INFO</td>
<td>The scheduled action [scheduled action name] was created at [time in UTC]. The first invocation is scheduled at [time in UTC].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3615</td>
<td>INFO</td>
<td>The scheduled action [scheduled action name] is scheduled at [time in UTC].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3616</td>
<td>INFO</td>
<td>The scheduled action [scheduled action name] at [time in UTC] finished with 'SUCCEEDED' status.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3617</td>
<td>ERROR</td>
<td>The scheduled action [scheduled action name] was skipped at [time in UTC] due to delay.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3618</td>
<td>INFO</td>
<td>The cluster [cluster name] pause operation started at [UTC time]. Pause Started</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3619</td>
<td>INFO</td>
<td>The cluster [cluster name] pause operation failed at [UTC time]. Retry it from the Amazon Redshift console. Pause Failed</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3626</td>
<td>INFO</td>
<td>The scheduled action [scheduled action name] was modified at [time in UTC]. The first invocation is scheduled at [time in UTC].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3627</td>
<td>INFO</td>
<td>The scheduled action [scheduled action name] was deleted at [time in UTC].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3628</td>
<td>ERROR</td>
<td>The scheduled action [scheduled action name] at [time in UTC] finished with 'FAILED' status.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3629</td>
<td>INFO</td>
<td>Amazon Redshift [cluster name] has received your relocation request. When Availability Zone relocation completes, Amazon Redshift sends an event notification.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3630</td>
<td>INFO</td>
<td>Amazon Redshift cluster [cluster name] was successfully relocated from [availability zone] to [availability zone]. You can use the cluster now.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3631</td>
<td>INFO</td>
<td>Amazon Redshift successfully relocated your Amazon Redshift cluster [cluster name] from [availability zone] to [availability zone] for recovery.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3632</td>
<td>INFO</td>
<td>Amazon Redshift has temporarily disabled cluster relocation for your Amazon Redshift cluster [cluster name] due to configuration changes. Try cluster relocation again later.</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-3652</td>
<td>INFO</td>
<td>The AQUA configuration status changed to [aqua configuration status] on Redshift cluster [cluster name]. This change will take effect after the next cluster reboot.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3658</td>
<td>ERROR</td>
<td>EC2-Classic to EC2-VPC migration failed for Redshift cluster [cluster id].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3659</td>
<td>INFO</td>
<td>EC2-Classic to EC2-VPC migration succeeded for Redshift cluster [cluster id].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3660</td>
<td>INFO</td>
<td>The cluster is being placed in hardware failure status. Please delete the EC2-Classic cluster and restore to a EC2-VPC cluster from the latest snapshot [snapshot name] created at [time in UTC].</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4000</td>
<td>INFO</td>
<td>Your admin credentials for your Amazon Redshift cluster: [cluster name] were updated at [time].</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4001</td>
<td>INFO</td>
<td>The security group [security group name] was modified at [time]. The changes will take place for all associated clusters automatically.</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4500</td>
<td>ERROR</td>
<td>The security group [security group name] you provided is invalid. Your configuration changes for cluster [cluster name] were not applied. Please visit the AWS Management Console to correct the issue.</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4501</td>
<td>ERROR</td>
<td>The security group [security group name] specified in Cluster Security Group [cluster security group name] could not be found. The authorization cannot be completed.</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4502</td>
<td>ERROR</td>
<td>The admin credentials for Amazon Redshift cluster [cluster name] failed to update at [time] due to concurrent activity. Allow the current workload to complete or reduce the active workload and then retry the operation.</td>
</tr>
</tbody>
</table>

The following table shows the event category and a list of events when a parameter group is the source type.

**Categories and events for the parameter group source type**

<table>
<thead>
<tr>
<th>Amazon Redshift category</th>
<th>Event ID</th>
<th>Event severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1002</td>
<td>INFO</td>
<td>The parameter [parameter name] was updated from [value] to [value] at [time].</td>
</tr>
<tr>
<td>Amazon Redshift category</td>
<td>Event ID</td>
<td>Event severity</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1003</td>
<td>INFO</td>
<td>Cluster parameter group [group name] was created.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1004</td>
<td>INFO</td>
<td>Cluster parameter group [group name] was deleted.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1005</td>
<td>INFO</td>
<td>Cluster parameter group [name] was updated at [time]. If you changed only dynamic parameters, associated clusters are being modified now. If you changed static parameters, all updates, including dynamic parameters, will be applied when you reboot the associated clusters.</td>
</tr>
</tbody>
</table>

The following tables shows the event category and a list of events when a security group is the source type.

### Categories and events for the security group source type

<table>
<thead>
<tr>
<th>Amazon Redshift category</th>
<th>Event ID</th>
<th>Event severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4002</td>
<td>INFO</td>
<td>Cluster security group [group name] was created.</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4003</td>
<td>INFO</td>
<td>Cluster security group [group name] was deleted.</td>
</tr>
<tr>
<td>Security</td>
<td>REDSHIFT-EVENT-4004</td>
<td>INFO</td>
<td>Cluster security group [group name] was changed at [time]. Changes will be automatically applied to all associated clusters.</td>
</tr>
</tbody>
</table>

The following tables shows the event category and a list of events when a snapshot is the source type.

### Categories and events for the snapshot source type

<table>
<thead>
<tr>
<th>Amazon Redshift category</th>
<th>Event ID</th>
<th>Event severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2009</td>
<td>INFO</td>
<td>A user snapshot [snapshot name] for Amazon Redshift Cluster [cluster name] started at [time]. To monitor snapshot progress, please visit the AWS Management Console.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2010</td>
<td>INFO</td>
<td>The user snapshot [snapshot name] for your Amazon Redshift cluster [cluster name] was cancelled at [time].</td>
</tr>
</tbody>
</table>
Managing event notifications using the Amazon Redshift console

**Topics**
- Creating an event notification subscription (p. 504)

You can create an Amazon Simple Notification Service (Amazon SNS) event notification subscription to send notifications when an event occurs for a given Amazon Redshift cluster, snapshot, security group, or parameter group. These notifications are sent to an SNS topic, which in turn transmits messages to any SNS consumers subscribed to the topic. The SNS messages to the consumers can be in any notification form supported by Amazon SNS for an AWS Region, such as an email, a text message, or a call to an HTTP endpoint. For example, all regions support email notifications, but SMS notifications can only be created in the US East (N. Virginia) Region. For more information, see Amazon Redshift event notifications (p. 491).

This section describes how to manage Amazon Redshift event notification subscriptions from the AWS Management Console.
Creating an event notification subscription

To create an event subscription

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Events.
3. Choose the Event subscription tab, then choose Create event subscriptions.
4. Enter the properties of your event subscription, such as name, source type, category, and severity. You can also enable Amazon SNS topics to get notified of events.
5. Choose Create event subscriptions to create your subscription.

Managing event notifications using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift CLI operations to manage event notifications.

- create-event-subscription
- delete-event-subscription
- describe-event-categories
- describe-event-subscriptions
- describe-events
- modify-event-subscription

You can use the following Amazon Redshift API actions to manage event notifications.

- CreateEventSubscription
- DeleteEventSubscription
- DescribeEventCategories
- DescribeEventSubscriptions
- DescribeEvents
- ModifyEventSubscription

For more information about Amazon Redshift event notifications, see Amazon Redshift event notifications (p. 491).
Quotas and limits in Amazon Redshift

Amazon Redshift quotas

Amazon Redshift has quotas that limit the use of several resources in your AWS account per AWS Region. There is a default value for each quota and some quotas are adjustable. For adjustable quotas, you can request an increase for your AWS account in an AWS Region by submitting an Amazon Redshift Limit Increase Form.

<table>
<thead>
<tr>
<th>Quota name</th>
<th>AWS default value</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS accounts that you can authorize to restore a snapshot per snapshot</td>
<td>20</td>
<td>No</td>
<td>The maximum number of AWS accounts that you can authorize to restore a snapshot, per snapshot.</td>
</tr>
<tr>
<td>AWS accounts that you can authorize to restore a snapshot per AWS KMS key</td>
<td>100</td>
<td>No</td>
<td>The maximum number of AWS accounts that you can authorize to restore a snapshot, per KMS key. That is, if you have 10 snapshots that are encrypted with a single KMS key, then you can authorize 10 AWS accounts to restore each snapshot, or other combinations that add up to 100 accounts and do not exceed 20 accounts for each snapshot.</td>
</tr>
<tr>
<td>Cluster IAM roles for Amazon Redshift to access other AWS services</td>
<td>50(^1)</td>
<td>No</td>
<td>The maximum number of IAM roles that you can associate with a cluster to authorize Amazon Redshift to access other AWS services for the user that owns the cluster and IAM roles.</td>
</tr>
<tr>
<td>Concurrency level (query slots) for all user-defined manual WLM queues</td>
<td>50</td>
<td>No</td>
<td>The maximum query slots for all user-defined queues defined by manual workload management.</td>
</tr>
<tr>
<td>Concurrency scaling clusters</td>
<td>10</td>
<td>Yes</td>
<td>The maximum number of concurrency scaling clusters.</td>
</tr>
</tbody>
</table>

\(^1\)The quota is 10 in the following AWS Regions: ap-northeast-3, af-south-1, eu-south-1, ap-southeast-3, us-gov-east-1, us-gov-west-1, us-iso-east-1, us-isob-east-1.
<table>
<thead>
<tr>
<th>Quota name</th>
<th>AWS default value</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC2 nodes in a cluster</td>
<td>128</td>
<td>Yes</td>
<td>The maximum number of DC2 nodes that you can allocate to a cluster. For more information about node limits for each node type, see Clusters and nodes in Amazon Redshift (p. 34).</td>
</tr>
<tr>
<td>DS2 nodes in a cluster</td>
<td>128</td>
<td>Yes</td>
<td>The maximum number of DS2 nodes that you can allocate to a cluster. For more information about node limits for each node type, see Clusters and nodes in Amazon Redshift (p. 34).</td>
</tr>
<tr>
<td>Event subscriptions</td>
<td>20</td>
<td>Yes</td>
<td>The maximum number of event subscriptions for this account in the current AWS Region.</td>
</tr>
<tr>
<td>Nodes</td>
<td>200</td>
<td>Yes</td>
<td>The maximum number of nodes across all database instances for this account in the current AWS Region.</td>
</tr>
<tr>
<td>Parameter groups</td>
<td>20</td>
<td>No</td>
<td>The maximum number of parameter groups for this account in the current AWS Region.</td>
</tr>
<tr>
<td>RA3 nodes in a cluster</td>
<td>128</td>
<td>Yes</td>
<td>The maximum number of RA3 nodes that you can allocate to a cluster. For more information about node limits for each node type, see Clusters and nodes in Amazon Redshift (p. 34).</td>
</tr>
<tr>
<td>Redshift-managed VPC endpoints</td>
<td>30</td>
<td>Yes</td>
<td>The maximum number of Redshift-managed VPC endpoints that you can connect to a cluster. For more information about Redshift-managed VPC endpoints, see Working with Redshift-managed VPC endpoints in Amazon Redshift (p. 67).</td>
</tr>
<tr>
<td>Grantees to cluster accessed</td>
<td>5</td>
<td>Yes</td>
<td>The maximum number of grantees that a cluster owner can authorize to create a Redshift-managed VPC endpoint for a cluster. For more information about Redshift-managed VPC endpoints, see Working with Redshift-managed VPC endpoints in Amazon Redshift (p. 67).</td>
</tr>
<tr>
<td>Redshift-managed VPC endpoints</td>
<td>5</td>
<td>Yes</td>
<td>The maximum number of Redshift-managed VPC endpoints that you can create per authorization. For more information about Redshift-managed VPC endpoints, see Working with Redshift-managed VPC endpoints in Amazon Redshift (p. 67).</td>
</tr>
<tr>
<td>Reserved nodes</td>
<td>200</td>
<td>Yes</td>
<td>The maximum number of reserved nodes for this account in the current AWS Region.</td>
</tr>
<tr>
<td>Schemas in each database per cluster</td>
<td>9,900</td>
<td>No</td>
<td>The maximum number of schemas that you can create in each database, per cluster. However, pg_temp_* schemas do not count towards this quota.</td>
</tr>
<tr>
<td><strong>Quota name</strong></td>
<td><strong>AWS default value</strong></td>
<td><strong>Adjustable</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------</td>
<td>---------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Security groups</td>
<td>20</td>
<td>Yes</td>
<td>The maximum number of security groups for this account in the current AWS Region.</td>
</tr>
<tr>
<td>Single row size when loading by COPY</td>
<td>4</td>
<td>No</td>
<td>The maximum size (in MB) of a single row when loading by using the COPY command.</td>
</tr>
<tr>
<td>Snapshots</td>
<td>20</td>
<td>Yes</td>
<td>The maximum number of user snapshots for this account in the current AWS Region.</td>
</tr>
<tr>
<td>Subnet groups</td>
<td>20</td>
<td>Yes</td>
<td>The maximum number of subnet groups for this account in the current AWS Region.</td>
</tr>
<tr>
<td>Subnets in a subnet group</td>
<td>20</td>
<td>Yes</td>
<td>The maximum number of subnets for a subnet group.</td>
</tr>
<tr>
<td>Tables for large cluster node type</td>
<td>9,900</td>
<td>No</td>
<td>The maximum number of tables for the large cluster node type. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
<tr>
<td>Tables for xlarge cluster node type</td>
<td>9,900</td>
<td>No</td>
<td>The maximum number of tables for the xlarge cluster node type. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
<tr>
<td>Tables for xlplus cluster node type with a single-node cluster.</td>
<td>9,900</td>
<td>No</td>
<td>The maximum number of tables for the xlplus cluster node type with a single-node cluster. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
<tr>
<td>Tables for xlplus cluster node type with a multiple-node cluster.</td>
<td>20,000</td>
<td>No</td>
<td>The maximum number of tables for the xlplus cluster node type with a multiple-node cluster. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
</tbody>
</table>
### Amazon Redshift quotas

<table>
<thead>
<tr>
<th>Quota name</th>
<th>AWS default value</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tables for 4xlarge cluster node type</td>
<td>100,000</td>
<td>No</td>
<td>The maximum number of tables for the 4xlarge cluster node type. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
<tr>
<td>Tables for 8xlarge cluster node type</td>
<td>100,000</td>
<td>No</td>
<td>The maximum number of tables for the 8xlarge cluster node type. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
<tr>
<td>Tables for 16xlarge cluster node type</td>
<td>100,000</td>
<td>No</td>
<td>The maximum number of tables for the 16xlarge cluster node type. This limit includes permanent tables, temporary tables, datashare tables, and materialized views. External tables are counted as temporary tables. Temporary tables include user-defined temporary tables and temporary tables created by Amazon Redshift during query processing or system maintenance. Views and system tables aren't included in this limit.</td>
</tr>
<tr>
<td>User-defined databases in a cluster</td>
<td>60</td>
<td>No</td>
<td>The maximum number of user-defined databases that you can create per cluster.</td>
</tr>
<tr>
<td>Timeout for idle or inactive sessions</td>
<td>4 hours</td>
<td>No</td>
<td>This setting applies to the cluster. For information about setting the idle-session timeout value for a user, see ALTER USER in the Amazon Redshift Database Developer Guide. The user setting takes precedence over the cluster setting.</td>
</tr>
<tr>
<td>Stored procedures in a database</td>
<td>10,000</td>
<td>No</td>
<td>The maximum number of stored procedures. See Limits and differences for stored procedure support for more limits.</td>
</tr>
</tbody>
</table>

The query editor v2 also has quotas as follows.

<table>
<thead>
<tr>
<th>Quota name</th>
<th>AWS default value</th>
<th>Adjustable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connections</td>
<td>500</td>
<td>Yes</td>
<td>Maximum number of connections that you can create using the query editor v2 in this account in the current Region.</td>
</tr>
<tr>
<td>Quota name</td>
<td>AWS default value</td>
<td>Adjustable</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Saved queries</td>
<td>2,500</td>
<td>Yes</td>
<td>Maximum number of saved queries that you can create using the query editor v2 in this account in the current Region.</td>
</tr>
<tr>
<td>Query versions</td>
<td>20</td>
<td>Yes</td>
<td>Maximum number of versions per query that you can create using the query editor v2 in this account in the current Region.</td>
</tr>
<tr>
<td>Saved charts</td>
<td>500</td>
<td>Yes</td>
<td>Maximum number of saved charts that you can create using the query editor v2 in this account in the current Region.</td>
</tr>
<tr>
<td>Rows fetched per query</td>
<td>100,000</td>
<td>Yes</td>
<td>Maximum number of rows fetched per query by the query editor v2 in this account in the current Region.</td>
</tr>
<tr>
<td>Data fetched size per query</td>
<td>5</td>
<td>Yes</td>
<td>Maximum size, in megabytes, of the data fetched per query by the query editor v2 in this account in the current Region.</td>
</tr>
<tr>
<td>Simultaneous socket connections</td>
<td>10</td>
<td>Yes</td>
<td>Maximum number of simultaneous socket connections to query editor v2 that a single principal can establish in the current Region. Evaluate whether to increase this quota if you receive errors that your socket connections are over the limit.</td>
</tr>
<tr>
<td>Simultaneous socket connections</td>
<td>250</td>
<td>Yes</td>
<td>Maximum number of simultaneous socket connections to query editor v2 that all principals in the account can establish in the current Region. Evaluate whether to increase this quota if you receive errors that your socket connections are over the limit.</td>
</tr>
</tbody>
</table>

Amazon Redshift Spectrum quotas and limits

Amazon Redshift Spectrum has the following quotas and limits:

- The maximum number of databases per AWS account when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://docs.aws.amazon.com/redshift/latest/gsg/redshift-service-limits.html) in the Amazon Web Services General Reference.
- The maximum number of tables per database when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://docs.aws.amazon.com/redshift/latest/gsg/redshift-service-limits.html) in the Amazon Web Services General Reference.
- The maximum number of partitions per table when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://docs.aws.amazon.com/redshift/latest/gsg/redshift-service-limits.html) in the Amazon Web Services General Reference.
- The maximum number of partitions per AWS account when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://docs.aws.amazon.com/redshift/latest/gsg/redshift-service-limits.html) in the Amazon Web Services General Reference.
- The maximum number of columns for external tables when using an AWS Glue Data Catalog, 1,598 when pseudocolumns are enabled, and 1,600 when pseudocolumns aren't enabled.
- The maximum size of a string value in an ION or JSON file when using an AWS Glue Data Catalog is 16 KB.
- You can add a maximum of 100 partitions using a single ALTER TABLE statement.
- All S3 data must be located in the same AWS Region as the Amazon Redshift cluster.
• Timestamps in ION and JSON must use ISO8601 format.
• External compression of ORC files is not supported.
• Text, OpenCSV, and Regex SERDEs do not support octal delimiters larger than \177.
• You must specify a predicate on the partition column to avoid reads from all partitions.

For example, the following predicate filters on the column ship_dtm, but doesn't apply the filter to the partition column ship_yyyymm:

```
WHERE ship_dtm > '2018-04-01'.
```

To skip unneeded partitions you need to add a predicate WHERE ship_yyyymm = '201804'. This predicate limits read operations to the partition \ship_yyyymm=201804\.

These limits don't apply to an Apache Hive metastore.

### Naming constraints

The following table describes naming constraints within Amazon Redshift.

<table>
<thead>
<tr>
<th></th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster identifier</td>
<td>• A cluster identifier must contain only lowercase characters.</td>
</tr>
<tr>
<td></td>
<td>• It must contain 1–63 alphanumeric characters or hyphens.</td>
</tr>
<tr>
<td></td>
<td>• Its first character must be a letter.</td>
</tr>
<tr>
<td></td>
<td>• It cannot end with a hyphen or contain two consecutive hyphens.</td>
</tr>
<tr>
<td></td>
<td>• It must be unique for all clusters within an AWS account.</td>
</tr>
<tr>
<td>Database name</td>
<td>• A database name must contain 1–64 alphanumeric characters.</td>
</tr>
<tr>
<td></td>
<td>• It must contain only lowercase letters.</td>
</tr>
<tr>
<td></td>
<td>• It cannot be a reserved word. For a list of reserved words, see Reserved words in the Amazon Redshift Database Developer Guide.</td>
</tr>
<tr>
<td>Endpoint name of a Redshift-managed VPC endpoint</td>
<td>• An endpoint name must contain 1–30 characters.</td>
</tr>
<tr>
<td></td>
<td>• Valid characters are A-Z, a-z, 0-9, and hyphen(-).</td>
</tr>
<tr>
<td></td>
<td>• The first character must be a letter.</td>
</tr>
<tr>
<td></td>
<td>• The name can't contain two consecutive hyphens or end with a hyphen.</td>
</tr>
<tr>
<td>Admin user name</td>
<td>• An admin user name must contain only lowercase characters.</td>
</tr>
<tr>
<td></td>
<td>• It must contain 1–128 alphanumeric characters.</td>
</tr>
<tr>
<td></td>
<td>• Its first character must be a letter.</td>
</tr>
<tr>
<td></td>
<td>• It cannot be a reserved word. For a list of reserved words, see Reserved words in the Amazon Redshift Database Developer Guide.</td>
</tr>
<tr>
<td>Admin password</td>
<td>• An admin password must contain 8–64 characters.</td>
</tr>
<tr>
<td></td>
<td>• It must contain at least one uppercase letter.</td>
</tr>
<tr>
<td></td>
<td>• It must contain at least one lowercase letter.</td>
</tr>
<tr>
<td></td>
<td>• It must contain one number.</td>
</tr>
</tbody>
</table>
| **Amazon Redshift Cluster Management Guide**  
<table>
<thead>
<tr>
<th><strong>Naming constraints</strong></th>
</tr>
</thead>
</table>
| **Parameter group name**  
- It can use any ASCII characters with ASCII codes 33–126, except ' (single quote), " (double quote), \, /, or @.  
- A parameter group name must contain 1–255 alphanumeric characters or hyphens.  
- It must contain only lowercase characters.  
- Its first character must be a letter.  
- It can't end with a hyphen or contain two consecutive hyphens. |
| **Cluster security group name**  
- A cluster security group name must contain no more than 255 alphanumeric characters or hyphens.  
- It must contain only lowercase characters.  
- It must not be Default.  
- It must be unique for all security groups that are created by your AWS account.  
- A subnet group name must contain no more than 255 alphanumeric characters or hyphens.  
- It must contain only lowercase characters.  
- It must not be Default.  
- It must be unique for all security groups that are created by your AWS account.  
- A cluster snapshot identifier must contain no more than 255 alphanumeric characters or hyphens.  
- It must contain only lowercase characters.  
- It must not be Default.  
- It must be unique for all security groups that are created by your AWS account. |
Tagging resources in Amazon Redshift

Topics
• Tagging overview (p. 512)
• Managing resource tags using the console (p. 513)
• Managing tags using the Amazon Redshift API (p. 513)

Tagging overview

In AWS, tags are user-defined labels that consist of key-value pairs. Amazon Redshift supports tagging to provide metadata about resources at a glance, and to categorize your billing reports based on cost allocation. To use tags for cost allocation, you must first activate those tags in the AWS Billing and Cost Management service. For more information about setting up and using tags for billing purposes, see Use cost allocation tags for custom billing reports and Setting up your monthly cost allocation report.

Tags are not required for resources in Amazon Redshift, but they help provide context. You might want to tag resources with metadata about cost centers, project names, and other pertinent information related to the resource. For example, suppose you want to track which resources belong to a test environment and a production environment. You could create a key named environment and provide the value test or production to identify the resources used in each environment. If you use tagging in other AWS services or have standard categories for your business, we recommend that you create the same key-value pairs for resources in Amazon Redshift for consistency.

Tags are retained for resources after you resize a cluster, and after you restore a snapshot of a cluster within the same region. However, tags are not retained if you copy a snapshot to another region, so you must recreate the tags in the new region. If you delete a resource, any associated tags are deleted.

Each resource has one tag set, which is a collection of one or more tags assigned to the resource. Each resource can have up to 50 tags per tag set. You can add tags when you create a resource and after a resource has been created. You can add tags to the following resource types in Amazon Redshift:

• CIDR/IP
• Cluster
• Cluster security group
• Cluster security group ingress rule
• Amazon EC2 security group
• Hardware security module (HSM) connection
• HSM client certificate
• Parameter group
• Snapshot
• Subnet group

To use tagging from the Amazon Redshift console, your IAM user can attach the AWS-managed policy AmazonRedshiftFullAccess. For an example IAM policy with limited tagging permissions that you can attach to an Amazon Redshift console user, see Example 7: Allow a user to tag resources with the
Tagging requirements

Tags have the following requirements:

- Keys can't be prefixed with aws:
- Keys must be unique per tag set.
- A key must be between 1 and 128 allowed characters.
- A value must be between 0 and 256 allowed characters.
- Values do not need to be unique per tag set.
- Allowed characters for keys and values are Unicode letters, digits, white space, and any of the following symbols: _ . : / = + - @.
- Keys and values are case sensitive.

Managing resource tags using the console

To manage tags on your Amazon Redshift resources

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose Configurations, then choose Manage tags.
3. Enter your choices for the resources and choose which tags to add, modify, or delete. Then choose Manage tags of the resources that you chose.
   
   Resources that you can tag include clusters, parameter groups, subnet groups, HSM client certificates, HSM connections, and snapshots.
4. On the Manage tags navigation page, choose Review and apply tag changes, then choose Apply to save your changes.

Managing tags using the Amazon Redshift API

You can use the following AWS CLI operations to manage tags in Amazon Redshift.

- create-tags
- delete-tags
- describe-tags

You can use the following Amazon Redshift API operations to manage tags:

- CreateTags
- DeleteTags
- DescribeTags
- Tag
- TaggedResource

Also, you can use the following Amazon Redshift API operations to manage and view tags for a specific resource:
• CreateCluster
• CreateClusterParameterGroup
• CreateClusterSecurityGroup
• CreateClusterSnapshot
• CreateClusterSubnetGroup
• CreateHsmClientCertificate
• CreateHsmConfiguration
• DescribeClusters
• DescribeClusterParameterGroups
• DescribeClusterSecurityGroups
• DescribeClusterSnapshots
• DescribeClusterSubnetGroups
• DescribeHsmClientCertificates
• DescribeHsmConfigurations
## Document history

The following table describes the important changes in each release of the Amazon Redshift Cluster Management Guide after June 2018. For notification about updates to this documentation, you can subscribe to an RSS feed.

**API version:** 2012-12-01

**Latest documentation update:** August 27, 2021

For a list of the changes to the Amazon Redshift Database Developer Guide, see Amazon Redshift Database Developer Guide document history.

For more information about new features, including a list of fixes and the associated cluster version numbers for each release, see Cluster version history.

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support for authentication profiles with Amazon Redshift drivers</td>
<td>You can now connect to Amazon Redshift with an authentication profile.</td>
<td>August 2, 2021</td>
</tr>
<tr>
<td>Support for AQUA</td>
<td>AQUA (Advanced Query Accelerator) is an analytics query accelerator for Amazon Redshift that uses custom-designed hardware to speed up queries that scan large datasets.</td>
<td>April 14, 2021</td>
</tr>
<tr>
<td>Support for cross-VPC endpoints for Amazon Redshift powered by AWS PrivateLink</td>
<td>You can now use Redshift-managed VPC endpoints with Amazon Redshift.</td>
<td>April 1, 2021</td>
</tr>
<tr>
<td>Support for Amazon Redshift query editor enhancements</td>
<td>You can now use the query editor with enhanced VPC routing, longer query run times, and more cluster node types.</td>
<td>February 17, 2021</td>
</tr>
<tr>
<td>Support for the console integration with partners</td>
<td>You can integrate with partners using the Amazon Redshift console.</td>
<td>December 9, 2020</td>
</tr>
<tr>
<td>Support for the ability to move clusters between Availability Zones</td>
<td>You can now move RA3 clusters between Availability Zones.</td>
<td>December 9, 2020</td>
</tr>
<tr>
<td>Support for ra3.xlplus node types</td>
<td>You can now create ra3.xlplus node types.</td>
<td>December 9, 2020</td>
</tr>
<tr>
<td>Support for JDBC driver version 2.0</td>
<td>You can now configure the JDBC driver version 2.0.</td>
<td>November 5, 2020</td>
</tr>
<tr>
<td>Support for Lambda UDFs and tokenization</td>
<td>You can now can write Lambda UDFs to enable external tokenization of data.</td>
<td>October 26, 2020</td>
</tr>
<tr>
<td>Support to schedule the run of an SQL statement</td>
<td>You can now schedule a query on the Amazon Redshift console.</td>
<td>October 22, 2020</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td><strong>Support for the Data API for Amazon Redshift</strong></td>
<td>Amazon Redshift can now be accessed using the built-in Data API. Documentation updates include an <em>Amazon Redshift Data API Reference</em>.</td>
<td>September 10, 2020</td>
</tr>
<tr>
<td><strong>Support for Amazon Redshift console query monitoring</strong></td>
<td>Updated the guide to describe new query monitoring graphs.</td>
<td>May 7, 2020</td>
</tr>
<tr>
<td><strong>Support for usage limits</strong></td>
<td>Updated the guide to describe usage limits.</td>
<td>April 23, 2020</td>
</tr>
<tr>
<td><strong>Multi-factor authentication</strong></td>
<td>Updated the guide to describe multi-factor authentication support.</td>
<td>April 20, 2020</td>
</tr>
<tr>
<td><strong>Elastic resize now supports node type changes</strong></td>
<td>Updated elastic resize description.</td>
<td>April 6, 2020</td>
</tr>
<tr>
<td><strong>Support for ra3.4xlarge node types with managed storage</strong></td>
<td>Updated the guide to include ra3.4xlarge node types.</td>
<td>April 2, 2020</td>
</tr>
<tr>
<td><strong>Support for pause and resume</strong></td>
<td>Updated the guide to describe the pause and resume cluster operations.</td>
<td>March 11, 2020</td>
</tr>
<tr>
<td><strong>Support for Microsoft Azure AD as an identity provider</strong></td>
<td>Updated the guide to describe the steps to use Microsoft Azure AD as an identity provider.</td>
<td>February 10, 2020</td>
</tr>
<tr>
<td><strong>Support for the RA3 node type</strong></td>
<td>Updated the guide to describe the new RA3 node type.</td>
<td>December 3, 2019</td>
</tr>
<tr>
<td><strong>Support for the new console</strong></td>
<td>Updated the guide to describe the new Amazon Redshift console.</td>
<td>November 11, 2019</td>
</tr>
<tr>
<td><strong>Security information updates</strong></td>
<td>Updates to the security information documentation.</td>
<td>June 24, 2019</td>
</tr>
<tr>
<td><strong>Snapshot enhancements</strong></td>
<td>Amazon Redshift now supports several enhancements to managing and scheduling snapshots.</td>
<td>April 4, 2019</td>
</tr>
<tr>
<td><strong>Concurrency scaling</strong></td>
<td>You can configure workload management (WLM) to enable concurrency scaling mode. For more information, see Configuring workload management.</td>
<td>March 21, 2019</td>
</tr>
<tr>
<td><strong>Updated JDBC and ODBC drivers</strong></td>
<td>Amazon Redshift now supports new versions of the JDBC and ODBC drivers. For more information, see Configure a JDBC connection and Configure and ODBC connection.</td>
<td>February 4, 2019</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Deferred maintenance</td>
<td>If you need to reschedule your cluster's maintenance window, you have the option to defer maintenance by up to 14 days. If we need to update hardware or make other mandatory updates during your period of deferment, we notify you and make the required changes. Your cluster isn't available during these updates. For more information, see Deferring maintenance.</td>
<td>November 20, 2018</td>
</tr>
<tr>
<td>Advance notification</td>
<td>Amazon Redshift provides notification in advance for some events. These events have an event category of pending. For example, we send an advance notification if a hardware update is required for one of the nodes in your cluster. You can subscribe to pending events the same as other Amazon Redshift events. For more information, see Subscribing to Amazon Redshift event notifications.</td>
<td>November 20, 2018</td>
</tr>
<tr>
<td>Elastic resize</td>
<td>Elastic resize is the fastest method to resize a cluster. Elastic resize adds or removes nodes on an existing cluster, then automatically redistributes the data to the new nodes. Because it doesn't create a new cluster, the elastic resize operation completes quickly, usually in a few minutes. For more information, see Resizing clusters.</td>
<td>November 15, 2018</td>
</tr>
<tr>
<td>New ODBC drivers</td>
<td>Amazon Redshift ODBC drivers have been updated to version 1.4.3.1000. For more information, see Configure an ODBC connection.</td>
<td>November 8, 2018</td>
</tr>
<tr>
<td>Cancel resize operation</td>
<td>You can now cancel a resize operation while it is in progress. For more information, see Resize operation overview.</td>
<td>November 2, 2018</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Modify cluster to change encryption</td>
<td>You can modify an unencrypted cluster to use AWS Key Management Service (AWS KMS) encryption, using either an AWS-managed key or a customer managed key. When you modify your cluster to enable KMS encryption, Amazon Redshift automatically migrates your data to a new encrypted cluster. You can also migrate an unencrypted cluster to an encrypted cluster by modifying the cluster.</td>
<td>October 16, 2018</td>
</tr>
<tr>
<td>Amazon Redshift spectrum supports enhanced VPC routing</td>
<td>You can now use Redshift Spectrum with enhanced VPC routing enabled for your cluster. You might need to perform additional configuration steps. For more information, see Using Amazon Redshift spectrum with enhanced VPC routing.</td>
<td>October 10, 2018</td>
</tr>
<tr>
<td>Query editor</td>
<td>You can now run SQL queries from the Amazon Redshift Management Console.</td>
<td>October 4, 2018</td>
</tr>
<tr>
<td>Workload execution breakdown chart</td>
<td>You can now get a detailed view of your workload's performance by looking at the Workload Execution Breakdown chart in the console. For more information, see Analyzing workload performance.</td>
<td>July 30, 2018</td>
</tr>
<tr>
<td>Maintenance tracks</td>
<td>You can now determine if your cluster will always be updated to the latest version of Amazon Redshift or to a previous version by choosing a maintenance track. For more information, see Choosing cluster maintenance tracks.</td>
<td>July 26, 2018</td>
</tr>
<tr>
<td>Updated JDBC and ODBC drivers</td>
<td>Amazon Redshift now supports new versions of the JDBC and ODBC drivers. For more information, see Configure a JDBC connection and Configure and ODBC connection.</td>
<td>July 13, 2018</td>
</tr>
<tr>
<td>On-demand cluster release versions</td>
<td>You can now upgrade your cluster to the latest release version as soon as it is available. For more information, see Manage cluster versions.</td>
<td>June 29, 2018</td>
</tr>
</tbody>
</table>
The following table describes the important changes to the *Amazon Redshift Cluster Management Guide* before July 2018.

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Release date</th>
</tr>
</thead>
<tbody>
<tr>
<td>New CloudWatch metrics</td>
<td>New CloudWatch metrics added for monitoring query performance. For more information, see [Monitoring Amazon Redshift using CloudWatch metrics](p. 459).</td>
<td>May 17, 2018</td>
</tr>
<tr>
<td>New JDBC and ODBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.2.12.1017. For more information, see [Configuring a JDBC driver version 1.x connection](p. 170).</td>
<td>March 7, 2018</td>
</tr>
<tr>
<td></td>
<td>Amazon Redshift ODBC drivers have been updated to version 1.4.1.1001. For more information, see [Configuring an ODBC connection](p. 179).</td>
<td></td>
</tr>
<tr>
<td>HSM encryption</td>
<td>Amazon Redshift supports only AWS CloudHSM for hardware security module (HSM) key management. For more information, see [Amazon Redshift database encryption](p. 323).</td>
<td>March 6, 2018</td>
</tr>
<tr>
<td>IAM Role Chaining</td>
<td>If an IAM role attached to your cluster doesn't have access to the necessary resources, you can chain another role, possibly belonging to another account. Your cluster then temporarily assumes the chained role to access the data. You can also grant cross-account access by chaining roles. Each role in the chain assumes the next role in the chain, until the cluster assumes the role at the end of chain. You can chain a maximum of 10 roles. For more information, see [Chaining IAM roles in Amazon Redshift](p. 408).</td>
<td>February 23, 2018</td>
</tr>
<tr>
<td>New DC2 node types</td>
<td>The new generation of dense compute (DC) node types offer much better performance at the same price as DC1. To take advantage of performance improvements, you can migrate your DC1 cluster to the newer DC2 node types. For more information, see [Clusters and nodes in Amazon Redshift](p. 34).</td>
<td>October 17, 2017</td>
</tr>
<tr>
<td>New JDBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.2.10.1009. Also, JDBC version 4.2 drivers are now supported. For more information, see [Configuring a JDBC driver version 1.x connection](p. 170).</td>
<td>October 16, 2017</td>
</tr>
<tr>
<td>ACM certificates</td>
<td>Amazon Redshift is replacing the SSL certificates on your clusters with AWS Certificate Manager (ACM) issued certificates. ACM is a trusted public certificate authority (CA) that is trusted by most current systems. You might need to update your current trust root CA certificates to continue to connect to your clusters using SSL. For more information, see [Transitioning to ACM certificates for SSL connections](p. 196).</td>
<td>September 18, 2017</td>
</tr>
<tr>
<td>Service-linked roles</td>
<td>A service-linked role is a unique type of IAM role that is linked directly to Amazon Redshift. Service-linked roles are predefined by Amazon Redshift and include all the permissions that the service requires to call AWS services on behalf of your Amazon Redshift cluster. For more</td>
<td>September 18, 2017</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>IAM database user authentication</td>
<td>You can configure your system to permit users to create user credentials and log on to the database based on their IAM credentials. You can also configure your system to let users sign on using federated single sign-on (SSO) through a SAML 2.0-compliant identity provider. For more information, see Using IAM authentication to generate database user credentials (p. 370).</td>
<td>August 11, 2017</td>
</tr>
<tr>
<td>New JDBC and ODBC drivers</td>
<td>The new JDBC and ODBC drivers support IAM database user authentication. Amazon Redshift JDBC drivers have been updated to version 1.2.7.1003. For more information, see Configuring a JDBC driver version 1.x connection (p. 170). Amazon Redshift ODBC drivers have been updated to version 1.3.6.1000. For more information, see Configuring an ODBC connection (p. 179).</td>
<td>August 11, 2017</td>
</tr>
<tr>
<td>Table-level restore supports enhanced VPC routing</td>
<td>Table-level restore is now supported on clusters that use Enhanced VPC routing (p. 265). For more information, see Restoring a table from a snapshot (p. 299).</td>
<td>July 19, 2017</td>
</tr>
<tr>
<td>Query monitoring rules</td>
<td>Using WLM query monitoring rules, you can define metrics-based performance boundaries for WLM queues and specify what action to take when a query goes beyond those boundaries—log, hop, or abort. You define query monitoring rules as part of your workload management (WLM) configuration. For more information, see Configuring workload management (p. 273).</td>
<td>April 21, 2017</td>
</tr>
<tr>
<td>New JDBC and ODBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.2.1.1001. Also, JDBC version 4.2 drivers are now supported. For more information, see Configuring a JDBC driver version 1.x connection (p. 170). Amazon Redshift ODBC drivers have been updated to version 1.3.1.1000. For more information, see Configuring an ODBC connection (p. 179).</td>
<td>November 18, 2016</td>
</tr>
<tr>
<td>Enhanced VPC routing</td>
<td>When you use Amazon Redshift enhanced VPC routing, Amazon Redshift forces all COPY and UNLOAD traffic between your cluster and your data repositories through your Amazon VPC. For more information, see Enhanced VPC routing in Amazon Redshift (p. 265).</td>
<td>September 15, 2016</td>
</tr>
<tr>
<td>New JDBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.1.17.1017. Also, JDBC version 4.2 drivers are now supported. For more information, see Configuring a JDBC driver version 1.x connection (p. 170).</td>
<td>July 5, 2016</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>New connection log fields</td>
<td>The Connection log (p. 424) audit log has two new fields to track SSL connections. If you routinely load audit logs to an Amazon Redshift table, you will need to add the following new columns to the target table: sslcompression and sslexpansion.</td>
<td>May 5, 2016</td>
</tr>
<tr>
<td>New ODBC drivers</td>
<td>Amazon Redshift ODBC drivers have been updated to version 1.2.7.1007. For more information, see Configuring an ODBC connection (p. 179).</td>
<td>March 30, 2016</td>
</tr>
<tr>
<td>IAM roles for COPY and UNLOAD</td>
<td>You can now specify one or more AWS Identity and Access Management (IAM) roles that your cluster can use for authentication to access other AWS services. IAM roles provide a more secure alternative to provide authentication with COPY, UNLOAD, or CREATE LIBRARY commands. For more information, see Authorizing Amazon Redshift to access other AWS services on your behalf (p. 405) and Authorizing COPY, UNLOAD, CREATE EXTERNAL FUNCTION, and CREATE EXTERNAL SCHEMA operations using IAM roles (p. 410).</td>
<td>March 29, 2016</td>
</tr>
<tr>
<td>Restore from table</td>
<td>You can restore a table from a cluster snapshot to a new table in an active cluster. For more information, see Restoring a table from a snapshot (p. 299).</td>
<td>March 10, 2016</td>
</tr>
<tr>
<td>New JDBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.1.10.1013. For more information, see Configuring a JDBC driver version 1.x connection (p. 170). You can now set the SSLMode property to specify whether the driver verifies host names when validating TLS/SSL certificates. For more information, see Configure JDBC driver options (p. 175).</td>
<td>February 18, 2016</td>
</tr>
<tr>
<td>Using IAM Condition in policies</td>
<td>You can further restrict access to resources by using the Condition element in IAM policies. For more information, see Using IAM policy conditions for fine-grained access control (p. 339).</td>
<td>December 10, 2015</td>
</tr>
<tr>
<td>Modify publicly accessible</td>
<td>You can modify an existing cluster in a VPC to change whether it is publicly accessible. For more information, see Modifying a cluster (p. 73).</td>
<td>November 20, 2015</td>
</tr>
<tr>
<td>New JDBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.1.10.1010. For more information, see Configuring a JDBC driver version 1.x connection (p. 170). Amazon Redshift ODBC drivers have been updated to version 1.2.6.1006. For more information, see Configuring an ODBC connection (p. 179).</td>
<td>November 19, 2015</td>
</tr>
<tr>
<td>Documentation fixes</td>
<td>Published various documentation fixes.</td>
<td>August 28, 2015</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated troubleshooting guidance about configuring network settings to ensure that hosts with different maximum transmission unit (MTU) sizes can determine the packet size for a connection. For more information, see Queries appear to hang and sometimes fail to reach the cluster (p. 236).</td>
<td>August 25, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Revised entire section about parameter groups for better organization and clarity. For more information, see Amazon Redshift parameter groups (p. 271).</td>
<td>August 17, 2015</td>
</tr>
<tr>
<td>New JDBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.1.7. For more information, see Configuring a JDBC driver version 1.x connection (p. 170).</td>
<td>August 14, 2015</td>
</tr>
<tr>
<td>WLM dynamic properties</td>
<td>The WLM configuration parameter now supports applying some properties dynamically. Other properties remain static changes and require that associated clusters be rebooted so that the configuration changes can be applied. For more information, see WLM dynamic and static properties (p. 274) and Amazon Redshift parameter groups (p. 271).</td>
<td>August 3, 2015</td>
</tr>
<tr>
<td>Copy KMS encrypted clusters to another AWS Region</td>
<td>Added content about configuring snapshot copy grants to enable copying of AWS KMS-encrypted clusters to another AWS Region. For more information, see Copying AWS KMS–encrypted snapshots to another AWS Region (p. 325).</td>
<td>July 28, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated the database encryption section to better explain how Amazon Redshift uses AWS KMS or HSMs for managing keys, and how the encryption process works with each of these options. For more information, see Amazon Redshift database encryption (p. 323).</td>
<td>July 28, 2015</td>
</tr>
<tr>
<td>New JDBC drivers</td>
<td>Amazon Redshift JDBC drivers have been updated to version 1.1.7. For more information, see Configuring a JDBC driver version 1.x connection (p. 170).</td>
<td>July 2, 2015</td>
</tr>
<tr>
<td>New node type</td>
<td>Amazon Redshift now offers a new node type, DS2. Updated documentation references to existing node types to use new names introduced in this release. Also revised the section to better explain the node type combinations and clarify default quota limits. For more information, see Clusters and nodes in Amazon Redshift (p. 34).</td>
<td>June 9, 2015</td>
</tr>
<tr>
<td>Reserved node offerings</td>
<td>Added content about new reserved node offerings. Also revised the section to better explain and compare the available offerings, and provided examples to demonstrate how on-demand and reserved node pricing affect billing. For more information, see Overview (p. 312).</td>
<td>June 9, 2015</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>New ODBC drivers</td>
<td>Amazon Redshift ODBC driver have been updated. Added a section for previous versions of these drivers and a link to release notes for the drivers. For more information, see Configuring an ODBC connection (p. 179).</td>
<td>June 5, 2015</td>
</tr>
<tr>
<td>Documentation fixes</td>
<td>Published various documentation fixes.</td>
<td>April 30, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated the download links to new versions of the Amazon Redshift JDBC drivers, and added a section for previous versions of these drivers. Also added a link to release notes for the drivers. For more information, see Configuring a JDBC driver version 1.x connection (p. 170).</td>
<td>April 1, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added downloads for new versions of the Amazon Redshift JDBC drivers. Also updated the format of the Amazon Redshift JDBC URL. For more information, see Configuring a JDBC driver version 1.x connection (p. 170). Added cluster security group ingress rules as a taggable resource. For more information, see Tagging resources in Amazon Redshift (p. 512). Updated the instructions for adding a cluster security group ingress rule, and added instructions for tagging a cluster security group ingress rule. For more information, see Managing cluster security groups using the console (p. 439).</td>
<td>March 16, 2015</td>
</tr>
<tr>
<td>New feature</td>
<td>This release of Amazon Redshift introduces new ODBC and JDBC drivers optimized for use with Amazon Redshift. For more information, see Connecting to an Amazon Redshift cluster using SQL client tools (p. 121).</td>
<td>February 26, 2015</td>
</tr>
<tr>
<td>New feature</td>
<td>This release of Amazon Redshift introduces cluster performance metrics that allow you to view and analyze query execution details. For more information, see Viewing queries and loads (p. 481).</td>
<td>February 26, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added a new example policy that demonstrates granting permission to common AWS service actions and resources on which Amazon Redshift relies. For more information, see Customer managed policy examples (p. 358).</td>
<td>January 16, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated guidance about setting the maximum transmission unit (MTU) to disable TCP/IP jumbo frames. For more information, see Use EC2-VPC when you create your cluster (p. 38) and Queries appear to hang and sometimes fail to reach the cluster (p. 236).</td>
<td>January 16, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Revised the content about the wlm_json_configuration parameter, and provided example syntax to configure this parameter by using the AWS CLI on the Linux, Mac OS X, and Microsoft Windows operating systems. For more information, see Configuring workload management (p. 273).</td>
<td>January 13, 2015</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added missing event notifications and descriptions. For more information, see Amazon Redshift event categories and event messages (p. 493).</td>
<td>January 8, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated guidance about IAM policies for Amazon Redshift actions and resources. Revised the section to improve organization and clarity. For more information, see Security in Amazon Redshift (p. 321).</td>
<td>November 21, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>This release of Amazon Redshift introduces the ability to encrypt clusters using encryption keys from AWS Key Management Service (AWS KMS). AWS KMS combines secure, highly available hardware and software to provide a key management system scaled for the cloud. For more information about AWS KMS and encryption options for Amazon Redshift, see Amazon Redshift database encryption (p. 323) and Managing clusters using the console (p. 70).</td>
<td>November 12, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>This release of Amazon Redshift introduces the ability to tag resources, such as clusters and snapshots. Tags enable you to provide user-defined metadata to categorize your billing reports based on cost allocation, and to help you better identify resources at a glance. For more information, see Tagging resources in Amazon Redshift (p. 512).</td>
<td>November 4, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Increased the maximum node limit to 128 nodes for dw1.8xlarge and dw2.8xlarge node sizes. For more information, see Clusters and nodes in Amazon Redshift (p. 34).</td>
<td>October 30, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added links to the Microsoft Visual C++ 2010 Redistributable Packages that are required for Amazon Redshift to use PostgreSQL ODBC drivers. For more information, see Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 180).</td>
<td>October 30, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added the ability to terminate queries and loads from the Amazon Redshift console. For more information, see Viewing queries and loads (p. 481) and Viewing cluster metrics during load operations (p. 485).</td>
<td>October 28, 2014</td>
</tr>
<tr>
<td>Documentation fixes</td>
<td>Published various documentation fixes.</td>
<td>October 17, 2014</td>
</tr>
<tr>
<td>New content</td>
<td>Added content about shutting down clusters and deleting clusters. For more information, see Shutting down and deleting clusters (p. 59) and Deleting a cluster (p. 73).</td>
<td>August 14, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Clarified the behavior of the Allow Version Upgrade setting for clusters. For more information, see Overview of Amazon Redshift clusters (p. 34).</td>
<td>August 14, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Revised procedures, screenshots, and organization of topic about working with clusters in Amazon Redshift console. For more information, see Managing clusters using the console (p. 70).</td>
<td>July 11, 2014</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>New content</td>
<td>Added a new tutorial about resizing Amazon Redshift clusters, including how to resize a cluster while minimizing the amount of time that the cluster is in read-only mode. For more information, see Resizing clusters in Amazon Redshift (p. 51).</td>
<td>June 27, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added the ability to rename clusters. For more information, see Renaming clusters (p. 58) and Modifying a cluster (p. 73).</td>
<td>June 2, 2014</td>
</tr>
<tr>
<td>Documentation</td>
<td>Updated the .NET code example to use the ODBC data provider when connecting to a cluster programmatically by using .NET. For more information, see Connecting to a cluster by using .NET (p. 230).</td>
<td>May 15, 2014</td>
</tr>
<tr>
<td>Documentation</td>
<td>Added options to select a different parameter group and security group when you restore a cluster from a snapshot. For more information, see Restoring a cluster from a snapshot (p. 303).</td>
<td>May 12, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added new section to describe how to configure a default Amazon CloudWatch alarm to monitor the percentage of disk space used in an Amazon Redshift cluster. This alarm is a new option in the cluster creation process. For more information, see Default disk space alarm (p. 49).</td>
<td>April 28, 2014</td>
</tr>
<tr>
<td>Documentation</td>
<td>Clarified information about Elliptic curve Diffie—Hellman Exchange (ECDHE) support in Amazon Redshift. For more information, see Connect using SSL (p. 193).</td>
<td>April 22, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added statement about Amazon Redshift support for the Elliptic curve Diffie—Hellman (ECDH) key agreement protocol. For more information, see Connect using SSL (p. 193).</td>
<td>April 18, 2014</td>
</tr>
<tr>
<td>Documentation</td>
<td>Revised and reorganized the topics in the Connecting to an Amazon Redshift cluster using SQL client tools (p. 121) section. Added more information about JDBC and ODBC connections, and a new troubleshooting section for connection issues.</td>
<td>April 15, 2014</td>
</tr>
<tr>
<td>Documentation</td>
<td>Added version in IAM policy examples throughout the guide.</td>
<td>April 3, 2014</td>
</tr>
<tr>
<td>Documentation</td>
<td>Added information about how pricing works when you resize a cluster. For more information, see Purchasing Amazon Redshift reserved nodes (p. 312).</td>
<td>April 2, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added a section about a new parameter, max_cursor_result_set_size, which sets the maximum result set size, in megabytes, that can be stored per individual cursor. This parameter value also affects the number of concurrently active cursors for the cluster. For more information, see Amazon Redshift parameter groups (p. 271).</td>
<td>March 28, 2014</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>New feature</td>
<td>Added explanation about the <strong>Cluster Version</strong> field now including both cluster engine version and database revision number. For more information, see <em>Amazon Redshift clusters</em> <em>(p. 34).</em></td>
<td>March 21, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated the resize procedure to show the new resize progress information on the cluster’s <strong>Status</strong> tab. For more information, see <em>Resizing a cluster</em> <em>(p. 74).</em></td>
<td>March 21, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Reorganized and updated <em>What is Amazon Redshift?</em> <em>(p. 1)</em> and revised <em>Amazon Redshift management overview</em> <em>(p. 1).</em> Published various documentation fixes.</td>
<td>February 21, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added new node types and sizes for Amazon Redshift clusters, and rewrote the related cluster overview topic for better organization and clarity based on feedback. For more information, see <em>Amazon Redshift clusters</em> <em>(p. 34).</em></td>
<td>January 23, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added information about using elastic IP (EIP) addresses for publicly-accessible Amazon Redshift clusters in virtual private clouds. For more information about EIP in Amazon Redshift, see <em>Managing clusters in a VPC</em> <em>(p. 79)</em> and <em>Creating a cluster in a VPC</em> <em>(p. 81).</em></td>
<td>December 20, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Added information about the AWS CloudTrail logs for Amazon Redshift. For more information about Amazon Redshift support for CloudTrail, see <em>Logging Amazon Redshift API calls with AWS CloudTrail</em> <em>(p. 429).</em></td>
<td>December 13, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Added information about the new user activity log and the <code>enable_user_activity_logging</code> database parameter for the database audit logging feature in Amazon Redshift. For more information about database audit logging, see <em>Database audit logging</em> <em>(p. 423).</em> For more information about database parameters, see <em>Amazon Redshift parameter groups</em> <em>(p. 271).</em></td>
<td>December 6, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to describe configuring Amazon Redshift to automatically copy automated and manual snapshots to a secondary AWS Region. For more information about configuring cross-Region snapshot copy, see <em>Copying snapshots to another AWS Region</em> <em>(p. 295).</em></td>
<td>November 14, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Added section to describe Amazon Redshift audit logging for connection and user activity, and storing these logs in Amazon S3. For more information about database audit logging, see <em>Database audit logging</em> <em>(p. 423).</em></td>
<td>November 11, 2013</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Release date</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>New feature</td>
<td>Added section to describe Amazon Redshift encryption with new features for managing encryption keys in a hardware security module (HSM) and rotating encryption keys. For more information about encryption, HSM, and key rotation, see Amazon Redshift database encryption (p. 323), Encryption for Amazon Redshift using hardware security modules (p. 326), and Encryption key rotation in Amazon Redshift (p. 327).</td>
<td>November 11, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to describe publishing notifications of Amazon Redshift events by using Amazon SNS. For information about Amazon Redshift event notifications, see Amazon Redshift event notifications (p. 491).</td>
<td>November 11, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to describe IAM resource level permissions. For information about Amazon Redshift IAM permissions, see Security in Amazon Redshift (p. 321).</td>
<td>August 9, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to describe restore progress metrics. For more information, see Restoring a cluster from a snapshot (p. 296).</td>
<td>August 9, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to describe cluster snapshot sharing and create snapshot progress metrics. For more information, see Sharing snapshots (p. 300).</td>
<td>July 17, 2013</td>
</tr>
<tr>
<td>Documentation fixes</td>
<td>Published various documentation fixes.</td>
<td>July 8, 2013</td>
</tr>
<tr>
<td>New console screens</td>
<td>Updated the Amazon Redshift Cluster Management Guide to match changes in the Amazon Redshift console.</td>
<td>April 22, 2013</td>
</tr>
<tr>
<td>New guide</td>
<td>This is the first release of the Amazon Redshift Management Guide.</td>
<td>February 14, 2013</td>
</tr>
</tbody>
</table>