Amazon Redshift: Management Guide
Copyright © 2023 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.
# Table of Contents

What Is Amazon Redshift? ........................................................................................................... 1
Are you a first-time Amazon Redshift user? ............................................................................... 1
Amazon Redshift Serverless feature overview ......................................................................... 1
Amazon Redshift provisioned clusters overview ........................................................................ 3
   Cluster management ............................................................................................................... 3
   Cluster access and security ................................................................................................. 4
   Monitoring clusters .......................................................................................................... 5
   Databases ......................................................................................................................... 5
Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse .... 6
Amazon Redshift Serverless ........................................................................................................ 19
   What is Amazon Redshift Serverless? ............................................................................... 19
   Amazon Redshift Serverless console ............................................................................... 19
   Considerations when using Amazon Redshift Serverless ............................................... 22
Compute capacity for Amazon Redshift Serverless ................................................................ 23
   Understanding Amazon Redshift Serverless capacity ..................................................... 23
Billing for Amazon Redshift Serverless .................................................................................. 24
   Pricing ............................................................................................................................... 24
   Billing for compute capacity ......................................................................................... 24
   Billing for storage ........................................................................................................... 25
   Using the Amazon Redshift Serverless free trial ............................................................. 26
   Billing usage notes ....................................................................................................... 26
Connecting to Amazon Redshift Serverless ............................................................................. 27
   Connecting to Amazon Redshift Serverless .................................................................... 27
   Connecting to Amazon Redshift Serverless through JDBC drivers ................................ 27
   Connecting to Amazon Redshift Serverless with the Data API ....................................... 29
   Connecting with SSL to Amazon Redshift Serverless ..................................................... 29
   Connecting to Amazon Redshift Serverless from an Amazon Redshift managed VPC endpoint ... 29
   Creating a publicly accessible Amazon Redshift Serverless instance and connecting to it ... 30
   Defining database roles to grant to federated users in Amazon Redshift Serverless ........ 31
   Additional resources .................................................................................................... 31
   Defining database roles to grant to federated users in Amazon Redshift Serverless ........ 31
Identity and access management in Amazon Redshift Serverless ............................................. 33
   Granting permissions to Amazon Redshift Serverless .................................................... 34
   Getting started with IAM credentials for Amazon Redshift ............................................. 35
   Managing access to Amazon Redshift Serverless database objects with database-role permissions .......................................................................................................................... 35
Migrating a provisioned cluster to Amazon Redshift Serverless ............................................. 36
   Creating a snapshot of your provisioned cluster .............................................................. 37
   Using a driver endpoint .................................................................................................... 37
   Using the Amazon Redshift Serverless SDK ................................................................... 39
Overview of Amazon Redshift Serverless workgroups and namespaces ................................. 39
   Overview of Amazon Redshift Serverless workgroups and namespaces ......................... 39
Managing Amazon Redshift Serverless using the console ..................................................... 41
   Setting up Amazon Redshift Serverless for the first time ............................................ 41
   Working with workgroups ............................................................................................. 41
   Working with namespaces ............................................................................................. 44
   Managing usage limits, query limits, and other administrative tasks ............................... 46
Monitoring queries and workloads with Amazon Redshift Serverless ................................ 48
   Monitoring queries and workload with Amazon Redshift Serverless ............................ 48
Audit logging for Amazon Redshift Serverless ........................................................................ 52
   Exporting logs ................................................................................................................. 52
   Working with snapshots and recovery points ................................................................ 57
   Working with snapshots and recovery points ................................................................ 57
Data sharing in Amazon Redshift Serverless .......................................................................... 61
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data sharing in Amazon Redshift Serverless</td>
<td>61</td>
</tr>
<tr>
<td>Tagging resources overview</td>
<td>62</td>
</tr>
<tr>
<td>Clusters</td>
<td>64</td>
</tr>
<tr>
<td>Overview of Amazon Redshift clusters</td>
<td>64</td>
</tr>
<tr>
<td>Preview features when using Amazon Redshift clusters</td>
<td>64</td>
</tr>
<tr>
<td>Clusters and nodes</td>
<td>65</td>
</tr>
<tr>
<td>Use EC2-VPC when you create your cluster</td>
<td>69</td>
</tr>
<tr>
<td>EC2-VPC</td>
<td>69</td>
</tr>
<tr>
<td>EC2-Classic</td>
<td>69</td>
</tr>
<tr>
<td>Launch a cluster</td>
<td>69</td>
</tr>
<tr>
<td>Overview of RA3 node types</td>
<td>70</td>
</tr>
<tr>
<td>Networking features supported by RA3 nodes</td>
<td>71</td>
</tr>
<tr>
<td>Working with Amazon Redshift managed storage</td>
<td>71</td>
</tr>
<tr>
<td>Managing RA3 node types</td>
<td>71</td>
</tr>
<tr>
<td>RA3 node type availability in AWS Regions</td>
<td>72</td>
</tr>
<tr>
<td>Upgrading to RA3 node types</td>
<td>72</td>
</tr>
<tr>
<td>Upgrade DS2 reserved nodes to RA3 reserved nodes during elastic resize or snapshot restore</td>
<td>74</td>
</tr>
<tr>
<td>Upgrading from DC1 node types to DC2 node types</td>
<td>75</td>
</tr>
<tr>
<td>Upgrading a DS2 cluster on EC2-Classic to EC2-VPC</td>
<td>76</td>
</tr>
<tr>
<td>Region and Availability Zone considerations</td>
<td>77</td>
</tr>
<tr>
<td>Cluster maintenance</td>
<td>77</td>
</tr>
<tr>
<td>Maintenance windows</td>
<td>77</td>
</tr>
<tr>
<td>Deferring maintenance</td>
<td>78</td>
</tr>
<tr>
<td>Choosing cluster maintenance tracks</td>
<td>79</td>
</tr>
<tr>
<td>Managing cluster versions</td>
<td>80</td>
</tr>
<tr>
<td>Rolling back the cluster version</td>
<td>80</td>
</tr>
<tr>
<td>Determining the cluster maintenance version</td>
<td>80</td>
</tr>
<tr>
<td>Default disk space alarm</td>
<td>81</td>
</tr>
<tr>
<td>Cluster status</td>
<td>81</td>
</tr>
<tr>
<td>Managing clusters overview</td>
<td>83</td>
</tr>
<tr>
<td>Resizing clusters</td>
<td>83</td>
</tr>
<tr>
<td>Pausing and resuming clusters</td>
<td>92</td>
</tr>
<tr>
<td>Renaming clusters</td>
<td>93</td>
</tr>
<tr>
<td>Shutting down and deleting clusters</td>
<td>94</td>
</tr>
<tr>
<td>Managing usage limits</td>
<td>95</td>
</tr>
<tr>
<td>Managing cluster relocation</td>
<td>96</td>
</tr>
<tr>
<td>Limitations</td>
<td>97</td>
</tr>
<tr>
<td>Turning on cluster relocation</td>
<td>97</td>
</tr>
<tr>
<td>Managing relocation using the console</td>
<td>98</td>
</tr>
<tr>
<td>Managing relocation using the Amazon Redshift CLI</td>
<td>99</td>
</tr>
<tr>
<td>Configuring Multi-AZ deployment (preview)</td>
<td>99</td>
</tr>
<tr>
<td>Overview</td>
<td>100</td>
</tr>
<tr>
<td>Managing Multi-AZ deployment</td>
<td>101</td>
</tr>
<tr>
<td>Managing Multi-AZ using the console</td>
<td>101</td>
</tr>
<tr>
<td>Using a custom domain name for client connections</td>
<td>105</td>
</tr>
<tr>
<td>Security for a custom domain name</td>
<td>106</td>
</tr>
<tr>
<td>Setting up a custom domain name</td>
<td>106</td>
</tr>
<tr>
<td>Working with Redshift-managed VPC endpoints</td>
<td>109</td>
</tr>
<tr>
<td>Considerations</td>
<td>110</td>
</tr>
<tr>
<td>Managing using the Redshift console</td>
<td>110</td>
</tr>
<tr>
<td>Managing using the AWS CLI</td>
<td>111</td>
</tr>
<tr>
<td>Managing using Amazon Redshift API operations</td>
<td>112</td>
</tr>
<tr>
<td>Managing clusters using the console</td>
<td>112</td>
</tr>
<tr>
<td>Creating a cluster</td>
<td>112</td>
</tr>
<tr>
<td>Modifying a cluster</td>
<td>114</td>
</tr>
<tr>
<td>Deleting a cluster</td>
<td>115</td>
</tr>
<tr>
<td>Rebooting a cluster</td>
<td>116</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Resizing a cluster</td>
<td>116</td>
</tr>
<tr>
<td>Upgrading the release version of a cluster</td>
<td>116</td>
</tr>
<tr>
<td>Getting information about cluster configuration</td>
<td>117</td>
</tr>
<tr>
<td>Getting an overview of cluster status</td>
<td>117</td>
</tr>
<tr>
<td>Creating a snapshot of a cluster</td>
<td>117</td>
</tr>
<tr>
<td>Creating or editing a disk space alarm</td>
<td>117</td>
</tr>
<tr>
<td>Working with cluster performance data</td>
<td>118</td>
</tr>
<tr>
<td>Managing clusters using the AWS CLI and Amazon Redshift API</td>
<td>118</td>
</tr>
<tr>
<td>Managing clusters using the AWS SDK for Java</td>
<td>119</td>
</tr>
<tr>
<td>Managing clusters in a VPC</td>
<td>121</td>
</tr>
<tr>
<td>Overview</td>
<td>121</td>
</tr>
<tr>
<td>Creating a cluster in a VPC</td>
<td>123</td>
</tr>
<tr>
<td>Managing VPC security groups for a cluster</td>
<td>123</td>
</tr>
<tr>
<td>Configuring security group communication settings for Amazon Redshift clusters</td>
<td>124</td>
</tr>
<tr>
<td>How Amazon Redshift works with VPC sharing for AWS resources</td>
<td>126</td>
</tr>
<tr>
<td>Cluster subnet groups</td>
<td>127</td>
</tr>
<tr>
<td>Cluster version history</td>
<td>131</td>
</tr>
<tr>
<td>Working with Amazon Aurora zero-ETL integrations (preview)</td>
<td>132</td>
</tr>
<tr>
<td>Preview limitations</td>
<td>133</td>
</tr>
<tr>
<td>Amazon Aurora limitations</td>
<td>133</td>
</tr>
<tr>
<td>Amazon Redshift limitations</td>
<td>134</td>
</tr>
<tr>
<td>Getting started with zero-ETL integrations</td>
<td>134</td>
</tr>
<tr>
<td>Create a target Amazon Redshift data warehouse</td>
<td>135</td>
</tr>
<tr>
<td>Turn on case sensitivity</td>
<td>136</td>
</tr>
<tr>
<td>Configure authorization in Amazon Redshift</td>
<td>137</td>
</tr>
<tr>
<td>Next steps</td>
<td>139</td>
</tr>
<tr>
<td>Creating destination databases</td>
<td>139</td>
</tr>
<tr>
<td>Create a destination database in Amazon Redshift</td>
<td>139</td>
</tr>
<tr>
<td>Next steps</td>
<td>140</td>
</tr>
<tr>
<td>Querying and sharing data in Amazon Redshift</td>
<td>140</td>
</tr>
<tr>
<td>Query your data in Amazon Redshift</td>
<td>140</td>
</tr>
<tr>
<td>Share your data in Amazon Redshift</td>
<td>141</td>
</tr>
<tr>
<td>Viewing zero-ETL integrations</td>
<td>142</td>
</tr>
<tr>
<td>Monitoring zero-ETL integrations</td>
<td>142</td>
</tr>
<tr>
<td>Troubleshooting zero-ETL integrations</td>
<td>143</td>
</tr>
<tr>
<td>The Resource policy tab isn't visible for my Amazon Redshift data warehouse</td>
<td>143</td>
</tr>
<tr>
<td>Creation of the integration failed</td>
<td>143</td>
</tr>
<tr>
<td>Tables don't have primary keys</td>
<td>144</td>
</tr>
<tr>
<td>Unsupported data types in tables</td>
<td>144</td>
</tr>
<tr>
<td>Data manipulation language commands failed</td>
<td>144</td>
</tr>
<tr>
<td>Tracked changes between data sources don't match</td>
<td>144</td>
</tr>
<tr>
<td>Authorization failed</td>
<td>144</td>
</tr>
<tr>
<td>Tables or schemas are more than 200K</td>
<td>145</td>
</tr>
<tr>
<td>Amazon Redshift can't load data</td>
<td>145</td>
</tr>
<tr>
<td>Workgroup parameter settings are incorrect</td>
<td>145</td>
</tr>
<tr>
<td>Database isn't created to activate a zero-ETL integration</td>
<td>145</td>
</tr>
<tr>
<td>Querying a database</td>
<td>146</td>
</tr>
<tr>
<td>Querying a database using the Amazon Redshift query editor v2</td>
<td>146</td>
</tr>
<tr>
<td>Configuring your AWS account</td>
<td>147</td>
</tr>
<tr>
<td>Working with query editor v2</td>
<td>151</td>
</tr>
<tr>
<td>Loading data into a database</td>
<td>163</td>
</tr>
<tr>
<td>Authoring and running queries</td>
<td>168</td>
</tr>
<tr>
<td>Authoring and running notebooks</td>
<td>172</td>
</tr>
<tr>
<td>Querying the AWS Glue Data Catalog</td>
<td>174</td>
</tr>
<tr>
<td>Querying a data lake</td>
<td>176</td>
</tr>
<tr>
<td>Working with datashares</td>
<td>178</td>
</tr>
<tr>
<td>Scheduling a query</td>
<td>180</td>
</tr>
</tbody>
</table>
Visualizing results ............................................................. 186
Collaborating and sharing as a team .................................... 191
Querying a database using the query editor .............................. 192
  Considerations .................................................................... 193
Enabling access .................................................................... 193
Connecting with the query editor .......................................... 194
Using the query editor ......................................................... 195
Scheduling a query ............................................................... 196
Connecting to a cluster using SQL client tools .......................... 199
  Configuring connections in Amazon Redshift ....................... 200
  Configuring security options for connections ....................... 303
  Connecting from client tools and code ................................ 308
  Connecting with SQL Workbench/J .................................... 340
Connect to your cluster programmatically .............................. 341
Using an authentication profile to connect to Amazon Redshift ... 344
Troubleshooting connection issues in Amazon Redshift ......... 346
Using the Data API .............................................................. 351
  Working with the Data API ................................................ 351
  Considerations when calling the Data API .......................... 352
  Running SQL statements with an idempotency token .......... 355
  Authorizing access ......................................................... 356
  Calling the Data API ....................................................... 360
  Troubleshooting Data API issues ..................................... 377
Scheduling Data API operations with Amazon EventBridge .... 378
Monitoring the Data API ..................................................... 381
Enhanced VPC routing ....................................................... 383
  Working with VPC endpoints .......................................... 384
  Enhanced VPC routing .................................................... 384
Redshift Spectrum and enhanced VPC routing ....................... 385
  Considerations when using Amazon Redshift Spectrum ........ 386
Parameter groups .............................................................. 389
  Overview ......................................................................... 389
About parameter groups ..................................................... 389
Default parameter values ................................................... 389
Configuring parameter values using the AWS CLI ................. 390
Configuring workload management .................................... 391
  WLM dynamic and static properties ................................ 392
  Properties for the wlm_json_configuration parameter ...... 392
  Configuring the wlm_json_configuration parameter using the AWS CLI .... 396
Managing parameter groups using the console ....................... 402
  Creating a parameter group .......................................... 402
  Modifying a parameter group ....................................... 402
  Creating or modifying a query monitoring rule using the console ......... 404
  Deleting a parameter group .......................................... 405
  Associating a parameter group with a cluster .................... 405
Managing parameter groups using the AWS SDK for Java ....... 405
Managing parameter groups using the AWS CLI and Amazon Redshift API .... 409
Snapshots and backups ...................................................... 410
  Overview of snapshots ................................................ 410
Automated snapshots ....................................................... 411
Automated snapshot schedules .......................................... 411
Snapshot schedule format ................................................ 411
Manual snapshots ........................................................... 413
Managing snapshot storage ................................................. 413
Excluding tables from snapshots ....................................... 414
Copying snapshots to another AWS Region ......................... 414
Restoring a cluster from a snapshot .................................... 414
Restoring a table from a snapshot ................................................................. 417
Sharing snapshots ...................................................................................... 418
Managing snapshots using the console ..................................................... 420
  Creating a snapshot schedule ................................................................. 420
  Creating a manual snapshot ................................................................. 421
  Changing the manual snapshot retention period .................................... 421
  Deleting manual snapshots .................................................................. 421
  Copying an automated snapshot ............................................................ 421
  Restoring a cluster from a snapshot ..................................................... 422
  Restoring a serverless namespace from a snapshot ............................... 422
  Sharing a cluster snapshot ................................................................. 422
  Configuring cross-Region snapshot copy for a nonencrypted cluster .... 424
  Configure cross-Region snapshot copy for an AWS KMS–encrypted cluster 424
  Modifying the retention period for cross-Region snapshot copy ................ 425
Managing snapshots using the AWS SDK for Java .................................... 425
Managing snapshots using the AWS CLI and Amazon Redshift API .......... 428
Working with AWS Backup ...................................................................... 428
  Considerations for using AWS Backup with Amazon Redshift .......... 429
  Managing AWS Backup with Amazon Redshift ................................... 430
Integrating with an AWS Partner .............................................................. 431
  Integrating with an AWS Partner using the Amazon Redshift console 431
Loading data with AWS partners ............................................................. 432
Purchasing reserved nodes ................................................................. 433
  Overview .............................................................................................. 433
    About reserved node offerings .......................................................... 433
    Comparing pricing among reserved node offerings ......................... 434
    How reserved nodes work .............................................................. 435
    Reserved nodes and consolidated billing ....................................... 435
    Reserved node examples .................................................................. 435
  Purchasing a reserved node offering with the console ....................... 437
  Upgrading reserved nodes with the AWS CLI ................................... 437
Purchasing a reserved node offering using Java ...................................... 438
Purchasing a reserved node offering using the AWS CLI and Amazon Redshift API 441
Security .................................................................................................. 442
  Data protection .................................................................................... 443
    Data encryption ................................................................................. 443
    Data tokenization ............................................................................ 454
    Internetwork traffic privacy ............................................................ 455
Identity and access management ......................................................... 455
  Authenticating with identities ............................................................ 456
  Access control ...................................................................................... 458
  Overview of managing access ............................................................ 458
  Using identity-based policies (IAM policies) ...................................... 463
    Native identity provider (IdP) federation for Amazon Redshift .... 494
    Amazon Redshift API permissions reference ................................. 496
  Using service-linked roles .................................................................. 497
  Using IAM authentication to generate database user credentials ....... 500
  Authorizing Amazon Redshift to access AWS services .................... 536
Logging and monitoring ........................................................................ 559
  Database audit logging ...................................................................... 559
  Logging with CloudTrail ................................................................. 567
Compliance validation ............................................................................ 575
Resilience .............................................................................................. 576
Infrastructure security ........................................................................... 576
  Network isolation ................................................................................ 455
  Security groups .................................................................................... 577
  Connecting using an interface VPC endpoint ..................................... 577
Patch 174 .................................................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
  New features for this version ................................................................................. 657
Patch 173 .................................................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
  New features for this version ................................................................................. 658
Patch 172 .................................................................................................................. 659
  New features ........................................................................................................ 659
Patch 171 .................................................................................................................. 660
  New features ........................................................................................................ 660
Patch 170 .................................................................................................................. 660
  New features ........................................................................................................ 660
Patch 169 .................................................................................................................. 660
  New features ........................................................................................................ 660
Patch 168 .................................................................................................................. 661
  New features ........................................................................................................ 661
Document history ...................................................................................................... 662
What is Amazon Redshift?

Welcome to the Amazon Redshift Management Guide. Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. Amazon Redshift Serverless lets you access and analyze data without all of the configurations of a provisioned data warehouse. Resources are automatically provisioned and data warehouse capacity is intelligently scaled to deliver fast performance for even the most demanding and unpredictable workloads. You don’t incur charges when the data warehouse is idle, so you only pay for what you use. You can load data and start querying right away in the Amazon Redshift query editor v2 or in your favorite business intelligence (BI) tool. Enjoy the best price performance and familiar SQL features in an easy-to-use, zero administration environment.

Regardless of the size of the dataset, 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:

- **Service Highlights and Pricing** – This product detail page provides the Amazon Redshift value proposition, service highlights, and pricing.
- **Getting started with Amazon Redshift Serverless** – This topic walks you through the process of setting up a serverless data warehouse, creating resources, and querying sample data.
- **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.

If you prefer to manage your Amazon Redshift resources manually, you can create provisioned clusters for your data querying needs. For more information, see Amazon Redshift clusters.

As an application developer, you can use the Amazon Redshift 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. 586).

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

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

Amazon Redshift Serverless feature overview

Most of the features supported by an Amazon Redshift provisioned data warehouse are also supported by Amazon Redshift Serverless. The following are some of its key capabilities.
## Amazon Redshift Serverless feature overview

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Snapshots</strong></td>
<td>You can restore a snapshot of Amazon Redshift Serverless or a provisioned data warehouse to Amazon Redshift Serverless. For more information, see [Working with snapshots and recovery points](p. 57).</td>
</tr>
<tr>
<td><strong>Recovery points</strong></td>
<td>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 after accidental writes or deletes. When you restore from a recovery point, all the data in your Amazon Redshift Serverless database 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. 57).</td>
</tr>
<tr>
<td><strong>Base RPU capacity</strong></td>
<td>You can set a base capacity in Redshift Processing Units (RPUs). One RPU provides 16 GB of memory. This setting gives you the ability to control the balance between resources in use and cost for your workload. You can increase this value to grow resources available and improve query performance, or lower the value to limit your spending. The default is 128 RPUs. You can also set usage limits, such as RPUs used per day, to control costs. For more information, see [Billing for Amazon Redshift Serverless](p. 24).</td>
</tr>
<tr>
<td><strong>Usage limits of data sharing</strong></td>
<td>You can limit the amount of data transferred from a producer Region to a consumer Region using the console or the API. These data transfer costs differ by AWS Region, and are measured in terabytes. For more information about data sharing, see [Getting started data sharing using the console](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>User-defined functions (UDFs)</strong></td>
<td>You can run user-defined functions (UDFs) in Amazon Redshift Serverless. For more information, see [Creating user-defined functions](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>Stored procedures</strong></td>
<td>You can run stored procedures in Amazon Redshift Serverless. For more information, see [Creating stored procedures](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>Materialized views</strong></td>
<td>You can create materialized views in Amazon Redshift Serverless. For more information, see [Creating materialized views](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>Spatial functions</strong></td>
<td>You can run spatial functions in Amazon Redshift Serverless. For more information, see [Querying spatial data](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>Federated queries</strong></td>
<td>You can run queries to join data with Aurora and Amazon RDS databases from Amazon Redshift Serverless. For more information, see [Querying data with federated queries](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>Data lake queries</strong></td>
<td>You can run queries to join data from your Amazon S3 data lake with Amazon Redshift Serverless. For more information, see [Querying a data lake](in the Amazon Redshift Management Guide).</td>
</tr>
<tr>
<td><strong>HyperLogLog</strong></td>
<td>You can run HyperLogLog functions in Amazon Redshift Serverless. For more information, see [Using HyperLogLog sketches](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td><strong>Querying data across databases</strong></td>
<td>You can query data across databases with Amazon Redshift Serverless. For more information, see [Querying data across databases](in the Amazon Redshift Database Developer Guide).</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data sharing</td>
<td>You can access datashares on provisioned data warehouses with Amazon Redshift Serverless. For more information, see <a href="https://docs.aws.amazon.com/redshift/latest/dg/share-datarows.html">Sharing data across clusters</a> in the <em>Amazon Redshift Database Developer Guide</em>.</td>
</tr>
<tr>
<td>Semistructured data querying</td>
<td>You can ingest and store semistructured data with the SUPER data type with Amazon Redshift Serverless. For more information, see <a href="https://docs.aws.amazon.com/redshift/latest/dg/data-types-super.html">Ingesting and querying semistructured data</a> in the <em>Amazon Redshift Database Developer Guide</em>.</td>
</tr>
<tr>
<td>Tagging resources</td>
<td>You can use the AWS CLI or the Amazon Redshift Serverless API to tag resources with metadata related to the resource. For more information, see <a href="https://docs.aws.amazon.com/redshift/latest/dg/tutorial-tagging-resources.html">Tagging resources</a>.</td>
</tr>
<tr>
<td>Machine learning</td>
<td>You can use Amazon Redshift machine learning with Amazon Redshift Serverless. For more information, see <a href="https://docs.aws.amazon.com/redshift/latest/dg/machine-learning.html">Using machine learning</a> in the <em>Amazon Redshift Database Developer Guide</em>.</td>
</tr>
<tr>
<td>SQL commands and functions</td>
<td>With a few exceptions (such as REBOOT_CLUSTER), you can use Amazon Redshift SQL commands and functions with Amazon Redshift Serverless. For more information, see <a href="https://docs.aws.amazon.com/redshift/latest/dg/sql-reference.html">SQL reference</a> in the <em>Amazon Redshift Database Developer Guide</em>.</td>
</tr>
<tr>
<td>CloudFormation resources</td>
<td>Using CloudFormation templates, you can deploy and update Amazon Redshift Serverless resources. This integration means you can spend less time managing resources and focus on your applications. For more information about CloudFormation resources in Amazon Redshift Serverless, see <a href="https://docs.aws.amazon.com/redshift-serverless/latest/userguide/constructs-index.html">Amazon Redshift Serverless resource type reference</a>.</td>
</tr>
<tr>
<td>CloudTrail resources</td>
<td>Amazon Redshift Serverless is integrated with AWS CloudTrail to provide a record of actions taken in Amazon Redshift Serverless. CloudTrail captures all API calls for Amazon Redshift Serverless as events. For more information, see <a href="https://docs.aws.amazon.com/redshift-serverless/latest/userguide/cloudtrail.html">CloudTrail for Amazon Redshift Serverless</a>.</td>
</tr>
</tbody>
</table>

**Amazon Redshift provisioned clusters 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 runtime 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](https://docs.aws.amazon.com/redshift/latest/dg/creating-cluster.html) (p. 64).

**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. 433).

### 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 and backups](p. 410).

### 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](p. 5) in the [Amazon Redshift Database Developer Guide](p. 410).

#### 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. 442). For more information about managing IAM identities, including guidance and best practices for IAM roles, see [Identity and access management in Amazon Redshift](p. 455).

#### 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. 422).

- 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. 577).

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. 5) 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. 444).

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. 303).

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. 559).

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 inform you that the event has occurred. For more information about events and notifications, see Amazon Redshift events (p. 626).

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. 594).

Databases

Amazon Redshift creates one database when you provision a cluster. This is the database that 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.

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

Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

For Amazon Redshift Serverless, some concepts and features are different than their corresponding feature for an Amazon Redshift provisioned data warehouse. For instance, one contrasting comparison is that Amazon Redshift Serverless doesn't have the concept of a cluster or node. The following table describes features and behavior in Amazon Redshift Serverless and explains how they differ from the equivalent feature in a provisioned data warehouse.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workgroup and Namespace</td>
<td>To isolate workloads and manage different resources in Amazon Redshift Serverless, you can create namespaces and workgroups in order to manage storage and compute resources separately.</td>
<td>A namespace is a collection of database objects and users. A workgroup is a collection of compute resources. For more information, see Amazon Redshift Serverless (p. 19) to understand the design for Amazon Redshift Serverless.</td>
<td>A provisioned cluster is a collection of compute nodes and a leader node, which you manage directly. For more information, see Amazon Redshift clusters (p. 64).</td>
</tr>
<tr>
<td>Node types</td>
<td>When you work with Amazon Redshift Serverless, you don't choose</td>
<td>Amazon Redshift Serverless automatically provisions and manages</td>
<td>You build a cluster with node types that meet your cost and performance specifications. For more information, see Amazon Redshift clusters (p. 64).</td>
</tr>
</tbody>
</table>
### Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td>node types or specify node count like you do with a provisioned Amazon Redshift cluster.</td>
<td>capacity for you. You can optionally specify base data warehouse capacity to select the right price/performance balance for your workloads. You can also specify maximum RPU hours to set cost controls to make sure that costs are predictable. For more information, see [Understanding Amazon Redshift Serverless capacity](p. 23).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Workload management and concurrency scaling</td>
<td>Amazon Redshift can scale for periods of heavy load. Amazon Redshift Serverless also can scale to meet intermittent periods of high load.</td>
<td>Amazon Redshift Serverless automatically manages resources efficiently and scales, based on workloads, within the thresholds of cost controls. For more information, see [Billing for compute capacity](p. 24).</td>
<td>With a provisioned data warehouse, you enable concurrency scaling on your cluster to handle periods of heavy load. For more information, see [Concurrency scaling](p.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Serverless</td>
<td>Provisioned</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Port</td>
<td>The port number that you use to connect.</td>
<td>With Amazon Redshift Serverless, you can change to another port from the port range of 5431–5455 or 8191–8215. For more information, see Connecting to Amazon Redshift Serverless (p. 27).</td>
<td>With a provisioned data warehouse, you can choose any port to connect.</td>
</tr>
<tr>
<td>Resizing</td>
<td>Add or remove compute resources to perform well for the workload.</td>
<td>Resizing is not applicable in Amazon Redshift Serverless. You can however change the base data warehouse RPU capacity, based on your price and performance requirements. For more information, see Understanding Amazon Redshift Serverless capacity (p. 23).</td>
<td>With a provisioned cluster, you perform a cluster resize to add nodes or remove nodes. For more information, see Overview of managing clusters in Amazon Redshift.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Serverless</td>
<td>Provisioned</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Pausing and resuming</td>
<td>You can pause a provisioned cluster when you don't have workloads to run, to save cost.</td>
<td>With Amazon Redshift Serverless, you pay only when queries run, so there is no need to pause or resume. For more information, see [Billing for compute capacity](p. 24).</td>
<td>You pause and resume a cluster manually, based on an assessment of your workload at various times. For more information, see <a href="#">Overview of managing clusters in Amazon Redshift</a>.</td>
</tr>
<tr>
<td>Querying external data with Spectrum queries</td>
<td>You can query data in Amazon S3 buckets, in a variety of formats, such as JSON.</td>
<td>Billing accrues when compute resources process workloads. Also, billing accrues when external Redshift Spectrum data is queried, like any other transaction. For more information, see [Billing for compute capacity](p. 24).</td>
<td>With a provisioned data warehouse, Amazon Redshift Spectrum capacity exists on separate servers that are queried from the Amazon Redshift cluster. For more information, see <a href="#">Querying external data using Amazon Redshift Spectrum</a>.</td>
</tr>
</tbody>
</table>
### Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Compute-resource billing</strong></td>
<td>How billing accrues for Amazon Redshift vs Amazon Redshift Serverless.</td>
<td>With Amazon Redshift Serverless, you pay for the workloads you run, in RPU-hours on a per-second basis, with a 60-second minimum charge. This includes queries that access data in open file formats in Amazon S3. For more information, see [Billing for compute capacity](p. 24).</td>
<td>With a provisioned cluster, billing occurs per second when the cluster isn’t paused.</td>
</tr>
<tr>
<td><strong>Maintenance window</strong></td>
<td>How server maintenance works.</td>
<td>With Amazon Redshift Serverless, there is no maintenance window. Updates are handled seamlessly. For more information, see [What is Amazon Redshift Serverless?](p. 24).</td>
<td>With a provisioned cluster, you specify a maintenance window when patching occurs. (Typically, you choose a recurring time when use is low.)</td>
</tr>
</tbody>
</table>
### Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Encryption</strong></td>
<td>You can enable database encryption.</td>
<td>Amazon Redshift Serverless is always encrypted with AWS KMS, with AWS managed or customer managed keys.</td>
<td>The data in a provisioned data warehouse can be encrypted with AWS KMS (with AWS managed or customer managed keys), or unencrypted. See <a href="https://aws.amazon.com/redshift/database-encryption/">Amazon Redshift database encryption (p. 444)</a>.</td>
</tr>
<tr>
<td><strong>Storage billing</strong></td>
<td>How billing for storage works.</td>
<td>For Amazon Redshift Serverless. The rate is calculated according to GB per month. See <a href="https://aws.amazon.com/redshift/billing/">Billing for compute capacity (p. 24)</a>.</td>
<td>Storage is billed apart from compute resources for a provisioned cluster with RA3 nodes.</td>
</tr>
</tbody>
</table>
### Feature | Description | Serverless | Provisioned
--- | --- | --- | ---
User management | How users are managed. | For both a provisioned data warehouse and for Amazon Redshift Serverless, users are IAM or Redshift users. For more information, see [Identity and access management in Amazon Redshift Serverless](p. 33). | For more information about managing IAM identities, including best practices for IAM roles, see [Identity and access management in Amazon Redshift](p. 455).
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td>JDBC and ODBC tools and</td>
<td>How client connections work.</td>
<td>Both a provisioned data warehouse and Amazon Redshift Serverless are</td>
<td>For Amazon Redshift Serverless, you don’t have to enter credentials in every instance. For more information, see Connecting to Amazon Redshift Serverless (p. 27).</td>
</tr>
<tr>
<td>compatibility</td>
<td></td>
<td>compatible with any JDBC or ODBC compliant tool or client application.</td>
<td>Access to Amazon Redshift requires sign-in credentials from a user associated with an IAM role. The IAM role has specific permissions attached for a provisioned data warehouse. Once authenticated, the user can connect directly to the database, to the Redshift console, and to query editor v2.</td>
</tr>
<tr>
<td>Requirement for credentials</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>on sign in</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For more information about drivers, see Configuring connections in the Amazon Redshift Management Guide. For information about connecting to Amazon Redshift Serverless, see Configuring connections.
Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data API</strong></td>
<td>You can access data from web services and other applications.</td>
<td>Amazon Redshift Serverless supports the Amazon Redshift Data API. With Amazon Redshift Serverless, you use the workgroup-name parameter instead of the cluster-identity parameter. For more information about calling the Data API, see Using the Amazon Redshift Data API (p. 351).</td>
<td></td>
</tr>
<tr>
<td><strong>Snapshots</strong></td>
<td>Provides point-in-time recovery.</td>
<td>Amazon Redshift Serverless supports snapshots and recovery points. For more information about snapshots and recovery points for a namespace, see Working with snapshots and recovery points (p. 57).</td>
<td>Provisioned clusters support snapshots. For more information, see Managing snapshots using the console.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Serverless</td>
<td>Provisioned</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Data Sharing</strong></td>
<td>Provides the ability to share data between databases in the same account or in different accounts.</td>
<td>Amazon Redshift Serverless supports all of the data sharing features that a provisioned data warehouse does. It also supports data sharing between Amazon Redshift Serverless and a provisioned data warehouse, tool, or client application.</td>
<td>Provisioned clusters support cross database, cross account, cross-Region, and AWS Data Exchange data sharing. For more information, see <a href="#">Sharing data across clusters in Amazon Redshift</a>.</td>
</tr>
<tr>
<td><strong>Tracks</strong></td>
<td>Provides a schedule for software updates.</td>
<td>Amazon Redshift Serverless has no concept of a track. Versions and updates are handled by the service. For more information about the design of Amazon Redshift Serverless, see <a href="#">Working with snapshots and recovery points</a>.</td>
<td>Provisioned clusters support switching between current and trailing tracks.</td>
</tr>
</tbody>
</table>
## Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System tables and views</strong></td>
<td>Provides a way to monitor your resources and system metadata.</td>
<td>Amazon Redshift Serverless supports new system tables and views. For more information about system tables, see Monitoring views (p. 49).</td>
<td>A provisioned data warehouse supports the existing set of system tables and views for monitoring and other tasks that require system metadata.</td>
</tr>
<tr>
<td><strong>Parameter groups</strong></td>
<td>This is a group of parameters that apply to all of the databases created in a cluster. These parameters configure database settings such as query timeout and date style.</td>
<td>Amazon Redshift Serverless does not have the concept of a parameter group.</td>
<td>Provisioned data warehouses support parameter groups. For more information about parameter groups for a provisioned cluster, see Amazon Redshift parameter groups (p. 389).</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Serverless</td>
<td>Provisioned</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Query monitoring</td>
<td>Provides a time-based view of queries run.</td>
<td>Query monitoring in Amazon Redshift Serverless requires users to connect to the database to use system tables. Thus, query monitoring and system tables are in sync. Queries of system tables in Amazon Redshift Serverless use the database user mapped to the IAM user for using query monitoring. For more information about monitoring queries, see Monitoring queries and workloads with Amazon Redshift Serverless.</td>
<td>Query monitoring in provisioned clusters does not show all data in system tables.</td>
</tr>
</tbody>
</table>
## Comparing Amazon Redshift Serverless to an Amazon Redshift provisioned data warehouse

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Serverless</th>
<th>Provisioned</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Audit logging</strong></td>
<td>Provides information about connections and user activities in the database.</td>
<td>With Amazon Redshift Serverless, CloudWatch is a destination for audit logs. Amazon S3 based audit log delivery is not supported for Amazon Redshift Serverless. For more information, see <a href="#">Audit logging for Amazon Redshift Serverless</a>.</td>
<td>For a provisioned cluster, Amazon S3-based audit log delivery has been the norm. Now, delivery of audit logs to CloudWatch is extended to cover provisioned data warehouses.</td>
</tr>
<tr>
<td><strong>Event notifications</strong></td>
<td>Amazon EventBridge is a serverless event bus service that you can use to connect your applications with event data from a variety of sources.</td>
<td>Amazon Redshift Serverless uses Amazon EventBridge to manage event notifications to keep you up-to-date regarding changes in your data warehouse. For more information, see <a href="#">Amazon Redshift Serverless event notifications with Amazon EventBridge</a>.</td>
<td>For a provisioned cluster, you manage event notifications using the Amazon Redshift console to create event subscriptions. For more information, see <a href="#">Managing cluster event notifications</a>.</td>
</tr>
</tbody>
</table>
Amazon Redshift Serverless

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

What is Amazon Redshift Serverless?

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

With Amazon Redshift Serverless, 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 data-lake integration to seamlessly query across a data warehouse, a data lake, and operational data sources.
- Deliver consistently high performance and simplified operations for the most demanding and volatile workloads with intelligent and automatic scaling.
- Use workgroups and namespaces to organize compute resources and data with granular cost controls.
- Pay only when the data warehouse is in use.

With Amazon Redshift Serverless, you use a console interface to reach a serverless data warehouse or APIs to build applications. Through the data warehouse, you can access your Amazon Redshift managed storage and your Amazon S3 data lake.

This video shows you how Amazon Redshift Serverless makes it easy to run and scale analytics without having to manage data warehouse infrastructure:

Amazon Redshift Serverless console

To get started with using the Amazon Redshift Serverless console, watch the following video: Getting Started with Amazon Redshift Serverless.

Serverless dashboard

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

- **Namespace overview** – This section shows the amount of snapshots and datashares within your namespace.
- **Workgroups** – This section shows all of the workgroups within Amazon Redshift Serverless.
• **Queries metrics** – This section shows query activity for the last one hour.

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

• **Free trial** – This section shows the free trial credits remaining in your AWS account. This covers all usage of Amazon Redshift Serverless resources and operations, including snapshots, storage, workgroup, and so on, under the same account.

• **Alarms** – This section shows the alarms you configured in Amazon Redshift Serverless.

**Data backup**

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

• **Snapshots** – You can create, delete, and manage snapshots of your Amazon Redshift Serverless data. The default retention period is indefinitely, but you can configure the retention period to be any value between 1 and 3653 days. You can authorize AWS accounts to restore namespaces from a 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. To recover data, you can restore a recovery point to any available namespace. You can create a snapshot from a recovery point if you want to keep a point of recovery for a longer time period. The default retention period is indefinitely, but you can configure the retention period to be any value between 1 and 3653 days.

**Data access**

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 using the console, see [Managing usage limits, query limits, and other administrative tasks](p. 46).

• **AWS KMS key** – The AWS KMS key used to encrypt resources in Amazon Redshift Serverless.

• **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. 33).

• **Redshift-managed VPC endpoints** – You can access your Amazon Redshift Serverless instance from another VPC or subnet. For more information, see [Connecting to Amazon Redshift Serverless from a Redshift managed VPC endpoint](p. 29).

**Limits**

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 Amazon Redshift Serverless instance can use in a time period before an action is initiated. You limit the amount of resource Amazon Redshift Serverless uses to run your workload. Usage is measured in Redshift Processing Unit (RPU) hours. An RPU hour is the number of RPUs used in an hour. 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.
• **Query limits** – You can add a limit to monitor performance and limits. For more information about query monitoring limits, see [WLM query monitoring rules](#).

For more information, see [Understanding Amazon Redshift Serverless capacity (p. 23)](#).

**Datashares**

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. 61)](#).

**Query and database monitoring**

On the **Query and database monitoring** page, you can view graphs of your **Query history** and **Database performance**.

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.

**Database performance**

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.

**Resource monitoring**

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

- **Metric filter** – You can use metric filters to select filters for a specific workgroup, as well as choose the time range and time interval.
- **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. 61).

Considerations when using Amazon Redshift Serverless

For a list of AWS Regions where the Amazon Redshift Serverless is available, see the endpoints listed for Redshift Serverless API in the Amazon Web Services General Reference.

Some resources used by Amazon Redshift Serverless are subject to quotas. For more information, see Quotas for Amazon Redshift Serverless objects (p. 647).

When you DECLARE a cursor, the result-set size specifications for Amazon Redshift Serverless is specified in DECLARE.

Maintenance window – There is no maintenance window with Amazon Redshift Serverless. Software version updates are automatically applied. There's no interruption for existing connection or query execution when Amazon Redshift switches versions. New connections will always connect and work with Amazon Redshift Serverless instantly.

Availability Zone IDs – When you configure your Amazon Redshift Serverless instance, open Additional considerations, and make sure that the subnet IDs provided in Subnet contain at least three 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.

Three subnets – You must have at least three subnets, and they must span across three Availability Zones. For example, you might use three subnets that map to the Availability Zones us-east-1a, us-east-1b, and us-east-1c. An exception to this is the US West (N. California) Region. It requires three subnets, in the same manner as the other regions, but these must span across only two Availability Zones. A condition is that one of the Availability Zones spanned must contain two of the subnets.

Free IP address requirements – You must have free IP addresses available when creating an Amazon Redshift Serverless workgroup. The minimum number of required IP addresses scales higher as the number of Base Redshift Processing Units (RPUs) for your workgroup increases. You must have the minimum number of IP addresses available for each subnet in each workgroup that you want to create. For more information on allocating IP addresses, see IP addressing in the Amazon VPC User Guide.

The number of minimum free IP addresses required when creating a workgroup is are as follows:

<table>
<thead>
<tr>
<th>Redshift Processing Units (RPUs)</th>
<th>Free IP addresses required</th>
<th>Minimum CIDR size</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>13</td>
<td>/27</td>
</tr>
<tr>
<td>64</td>
<td>21</td>
<td>/27</td>
</tr>
<tr>
<td>128</td>
<td>37</td>
<td>/26</td>
</tr>
<tr>
<td>256</td>
<td>69</td>
<td>/25</td>
</tr>
<tr>
<td>512</td>
<td>133</td>
<td>/24</td>
</tr>
</tbody>
</table>

You also need free IP addresses when updating your workgroup to use more RPUs. The number of minimum free IP addresses required when updating a workgroup is as follows:
### Number of free IP addresses required when updating a subnet

<table>
<thead>
<tr>
<th>Redshift Processing Units (RPUs)</th>
<th>Updated Redshift Processing Units (RPUs)</th>
<th>Free IP addresses required</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>64</td>
<td>16</td>
</tr>
<tr>
<td>64</td>
<td>128</td>
<td>28</td>
</tr>
<tr>
<td>128</td>
<td>256</td>
<td>52</td>
</tr>
<tr>
<td>256</td>
<td>512</td>
<td>100</td>
</tr>
</tbody>
</table>

### Storage space after migration

Storage space after migration – When migrating small Amazon Redshift provisioned clusters to Amazon Redshift Serverless, you might see an increase in storage-space allocation after migration. This is a result of optimized storage-space allocation, resulting in preallocated storage space. This space is used over a period of time as data grows in Amazon Redshift Serverless.

### Datasharing between Amazon Redshift Serverless and Amazon Redshift provisioned clusters

Datasharing between Amazon Redshift Serverless and Amazon Redshift provisioned clusters – When datasharing where Amazon Redshift Serverless is the producer and a provisioned cluster is the consumer, the provisioned cluster must have a cluster version later than 1.0.38214. If you use a cluster version earlier than this, an error occurs when you run a query. You can view the cluster version on the Amazon Redshift console on the **Maintenance** tab. You can also run `SELECT version();`.

### Max query execution time

Max query execution time – Elapsed execution time for a query, in seconds. Execution time doesn’t include time spent waiting in a queue. If a query exceeds the set execution time, Amazon RedshiftServerless stops the query. Valid values are 0–86,399.

### Migrating for tables with interleaved sort keys

Migrating for tables with interleaved sort keys – When migrating Amazon Redshift provisioned clusters to Amazon Redshift Serverless, Redshift converts tables with interleaved sort keys and DISTSTYLE KEY to compound sort keys. The DISTSTYLE doesn’t change. For more information on distribution styles, see [Working with data distribution styles](#) in the Amazon Redshift Developer Guide. For more information on sort keys, see [Working with sort keys](#).

## Compute capacity for Amazon Redshift Serverless

### Understanding Amazon Redshift Serverless capacity

#### RPUs

Amazon Redshift Serverless measures data warehouse capacity in Redshift Processing Units (RPUs). RPUs are resources used to handle workloads.

#### Base capacity

This setting specifies the base data warehouse capacity Amazon Redshift uses to serve queries. Base capacity is specified in RPUs. You can set a base capacity in Redshift Processing Units (RPUs). One RPU provides 16 GB of memory. Setting higher base capacity improves query performance, especially for data processing jobs that consume a lot of resources. The default base capacity for Amazon Redshift Serverless is 128 RPUs. You can adjust the **Base capacity** setting from 8 RPUs to 512 RPUs in units of 8 (8, 16, 24...512), using the AWS console, the UpdateWorkgroup API operation, or `update-workgroup` operation in the AWS CLI.

With a minimum capacity of 8 RPU, you now have more flexibility to run simpler to more complex workloads based on performance requirements. The 8, 16, and 24 RPU base RPU capacities are targeted towards workloads that require less than 128TB of data. If your data requirements are greater than 128...
TB, you must use a minimum of 32 RPU. For workloads that have tables with large number columns and higher concurrency, we recommend using 32 or more RPU.

**Considerations and limitations for Amazon Redshift Serverless capacity**

The following are considerations and limitations for Amazon Redshift Serverless capacity.

- Configurations of 8 or 16 RPU support Redshift managed storage capacity of up to 128 TB. If you're using more than 128 TB of managed storage, you can't downgrade to less than 32 RPU.

**Billing for Amazon Redshift Serverless**

**Pricing**

For pricing information, see Amazon Redshift pricing.

**Billing for compute capacity**

**Base capacity and its affect on billing**

When queries run, you're billed according to the capacity used in a given duration, in RPU hours on a per-second basis. When no queries are running, you aren't billed for compute capacity. You are also charged for Redshift Managed Storage (RMS), based on the amount of data stored.

When you create your workgroup, you have the option to set the **Base capacity** for computing. To meet the price/performance requirements of your workload at a workgroup level, adjust the base capacity higher or lower for an existing workgroup. Select the workgroup from **Workgroup configuration** and choose the **Limits** tab to change the base capacity using the console.

As the number of queries increases, Amazon Redshift Serverless scales automatically to provide consistent performance.

**Maximum RPU hours**

To keep costs predictable for Amazon Redshift Serverless, you can set the **Maximum RPU hours** used per day, per week, or per month. You can set this using the console, or with the API. When a limit is reached, you can specify to write a log entry to a system table, or receive an alert, or turn off user queries. Setting the maximum RPU hours helps keep your cost under control. Settings for maximum RPU hours apply to your workgroup for both queries that access data in your data warehouse and queries that access external data, such as in an external table in Amazon S3.

Setting the maximum RPU hours for the workgroup doesn't limit the performance. You can adjust the setting at any time without an interruption to query processing.

Setting the base capacity and maximum RPU hours can help you meet your price/performance requirements while maintaining predictable costs. For more information about the base capacity setting, see Amazon Redshift pricing calculator. For more information about serverless billing, see Amazon Redshift pricing.

Another way to keep the cost for Amazon Redshift Serverless predictable is to use AWS Cost Anomaly Detection to reduce surprises in billing and provide more control.

**Note**

The Amazon Redshift pricing calculator is helpful for estimating pricing. You enter the compute resources you need and it provides a preview of the cost.
Monitoring usage and cost

There are several ways you can estimate usage and billing for Amazon Redshift Serverless. 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 Amazon Redshift Serverless instance, and has additional features to provide insights and set actions.

Visualizing usage by querying a system view

Query the SYS_SERVERLESS_USAGE system table to track usage and get the charges for queries:

```sql
select trunc(start_time) "Day",
       (sum(charged_seconds)/3600::double precision) * <Price for 1 RPU> as cost_incurred
from sys_serverless_usage
group by 1
order by 1
```

This query provides the cost per day incurred for Amazon Redshift Serverless, based on usage.

Usage notes for determining usage and cost

- You pay for the workloads you run in RPU-hours on a per-second basis, with a 60-second minimum charge.
- Records from the sys_serverless_usage system table show cost incurred in 1-minute time intervals. Understanding the following columns is important:

  The charged_seconds column:
  - Provides the compute unit (RPU) seconds that were charged during the time interval. The results include any minimum charges in Amazon Redshift Serverless.
  - Has information about compute-resource usage after transactions complete. Thus, this column value may be 0 if transactions haven't finished.

  The compute_seconds column:
  - Provides real-time compute usage information. This doesn't include any minimum charges in Amazon Redshift Serverless. Thus it can differ to some degree from the charged seconds billed during the interval.
  - Shows usage information during each transaction (even if a transaction hasn't ended), and hence the data provided is real-time.

For more information about monitoring tables and views, see Monitoring queries and workloads with Amazon Redshift Serverless.

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 capacity. Storage used for user snapshots is billed at the standard backup billing rates, depending on your usage tier.
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.

Visualizing billing usage with CloudWatch

The metric SnapshotStorage, which tracks snapshot storage usage, is generated and sent to CloudWatch. For more information about CloudWatch, see What is Amazon CloudWatch?

Using the Amazon Redshift Serverless free trial

Amazon Redshift Serverless offers a free trial. If you participate in the free trial, you can view the free trial credit balance in the Redshift console, and check free trial usage in the SYS_SERVERLESS_USAGE system view. Note that billing details for free trial usage does not appear in the billing console. You can only view usage in the billing console after the free trial ends. For more information about the Amazon Redshift Serverless free trial, see Amazon Redshift Serverless free trial.

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.

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

- **Scaling down** - Amazon Redshift Serverless scales up from its base RPU capacity to handle periods of higher load. It some cases, RPU capacity can remain at a higher setting for a period after query load falls. We recommend that you set maximum RPU hours in the console to guard against unexpected cost.

- **System tables** - When you query a system table, the query time is billed.

- **Redshift Spectrum** - When you have Amazon Redshift Serverless, and you 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 of resource usage, 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.
Amazon Redshift Serverless best practices for keeping billing predictable

The following are best practices and built-in settings that help keep your billing consistent.

- Make sure to end each transaction. When you use `BEGIN` to start a transaction, it's important to `END` it as well.
- Use best-practice error handling to respond gracefully to errors and end each transaction. Minimizing open transactions helps to avoid unnecessary RPU use.
- Use `SESSION TIMEOUT` to help end open transactions and idle sessions. It causes any session kept idle or inactive for more than 3600 seconds (1 hour) to time out. It causes any transaction kept open and inactive for more than 21600 seconds (6 hours) to time out. This timeout setting can be changed explicitly for a specific user, such as when you want to keep a session open for a long-running query. The topic `CREATE USER` shows how to adjust `SESSION TIMEOUT` for a user.
- In most cases, we recommend that you don't extend the `SESSION TIMEOUT` value, unless you have a use case that requires it specifically. If the session remains idle, with an open transaction, it can result in a case where RPUs are used until the session is closed. This will result in unnecessary cost.
- Amazon Redshift Serverless has a maximum time of 86,399 seconds (24 hours) for a running query. The maximum period of inactivity for an open transaction is six hours before Amazon Redshift Serverless ends the session associated with the transaction. For more information, see Quotas for Amazon Redshift Serverless objects (p. 647).

Connecting to Amazon Redshift Serverless

Once you've set up your Amazon Redshift Serverless instance, you can connect to it in a variety of methods, outlined below. If you have multiple teams or projects and want to manage costs separately, you can use separate AWS accounts.

For a list of AWS Regions where the Amazon Redshift Serverless is available, see the endpoints listed for Redshift Serverless API in the Amazon Web Services General Reference.

Amazon Redshift Serverless connects to the serverless environment in your AWS account in the current AWS Region. Amazon Redshift Serverless runs in a VPC within the port ranges port ranges 5431-5455 and 8191-8215. The default is 5439. Currently, you can only change ports with the API operation `UpdateWorkgroup` and the AWS CLI operation `update-workgroup`.

Connecting to Amazon Redshift Serverless

You can connect to a database (named `dev`) in Amazon Redshift Serverless with the following syntax.

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

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

```
default.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 Amazon Redshift Serverless with your preferred SQL client using the Amazon Redshift-provided JDBC driver version 2 driver.
To connect with sign-in credentials for database authentication using JDBC driver version 2.1.x or later, use the following syntax. The port number is optional; if not included, Amazon Redshift Serverless defaults to port number 5439. You can change to another port from the port range of 5431-5455 or 8191-8215. To change the default port for a serverless endpoint, use the AWS CLI and Amazon Redshift API.

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

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

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

To connect with IAM using JDBC driver version 2.1.x or later, use the following syntax. The port number is optional; if not included, Amazon Redshift Serverless defaults to port number 5439. You can change to another port from the port range of 5431-5455 or 8191-8215. To change the default port for a serverless endpoint, use the AWS CLI and Amazon Redshift API.

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

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

```
```

For ODBC, use the following syntax.

```
Driver={Amazon Redshift (x64)}; Server=workgroup-name.account-number.aws-region.redshift-serverless.amazonaws.com; Database=dev
```

If you are using a JDBC driver version prior to 2.1.0.9 and connecting with IAM, you will need to use the following syntax.

```
jdbc:redshift:iam://redshift-serverless-<name>:aws-region/database-name
```

For example, the following connection string specifies the workgroup default and the AWS Region us-east-1.

```
jdbc:redshift:iam://redshift-serverless-default:us-east-1/dev
```

For more information about drivers, see Configuring connections in Amazon Redshift (p. 200).

**Finding your JDBC and ODBC connection string**

To connect to your workgroup with your SQL client tool, you must have the JDBC or ODBC connection string. You can find the connection string in the Amazon Redshift Serverless console, on a workgroup's details page.

**To find the connection string for a workgroup**

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 **Redshift Serverless**.

3. On the navigation menu, choose **Workgroup configuration**, then choose the workgroup name from the list to open its details.

4. 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 workgroup runs. Choose the icon next to the appropriate connection string to copy the connection string.

Connecting to Amazon Redshift Serverless with the Data API

You can also use the Amazon Redshift Data API to connect to Amazon Redshift Serverless. Use the workgroup-name parameter instead of the cluster-identifier parameter in your AWS CLI calls.

For more information about the Data API, see [Using the Amazon Redshift Data API](p. 351). For example code calling the Data API in Python and other examples, see [Getting Started with Redshift Data API](p. 351) and look in the quick-start and use-cases folders in [GitHub](https://github.com). 

Connecting with SSL to Amazon Redshift Serverless

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](p. 351), 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. When connecting to an Amazon Redshift Serverless instance, you don't have to set any parameters to accept SSL connections.

Connecting to Amazon Redshift Serverless from an Amazon Redshift managed VPC endpoint

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 to Amazon Redshift Serverless from a Redshift managed VPC endpoint

Amazon Redshift Serverless is provisioned in a VPC. By creating a Redshift managed VPC endpoint, you privately access your Amazon Redshift Serverless 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 using the console

1. On the console, choose **Workgroup configuration**, and select a workgroup from the list.

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 Amazon Redshift Serverless instance.
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 using the console
1. On the console, choose Workgroup configuration, and select a workgroup from the list.
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 Workgroup configuration, and select a workgroup from the list.
2. In Redshift managed VPC endpoints, select the VPC endpoint to delete.
3. Choose Delete.

Creating a publicly accessible Amazon Redshift Serverless instance and connecting to it

Connect to a publicly accessible Amazon Redshift Serverless instance
You can configure your Amazon Redshift Serverless instance so you can query it from a SQL client in the public internet. You don't always have to connect from the local VPC or from a subnet behind your firewall.

These steps walk you through configuring Amazon Redshift Serverless to accept connections from the internet.

1. On the Redshift console, go to the Amazon Redshift Serverless main menu. Choose Create workgroup and then follow the steps to give it a name. Pick the associated VPC and subnet. Choose Next.
2. Complete the steps to create a namespace. The process includes specifying a database and assigning an IAM role with permissions to perform database tasks.
   If you already created a namespace, that works too.
3. On the Amazon VPC service console, verify that your VPC has an internet gateway attached, with a custom route table. For more information, see Connect to the internet using an internet gateway.
4. After you complete the previous steps, or if you already have a configured namespace and workgroup, choose Workgroup configuration. Choose the workgroup from the list. Then, in the Network and security panel, choose edit.
5. Select Turn on Public Accessible. When you do this, the Amazon Redshift Serverless instance is made public by means of assigning to it a static IPv4 Elastic IP address. This IP address is allocated to your AWS account.
After you configure Amazon Redshift Serverless to accept connections from public clients, follow these steps to connect.

1. On the Amazon Redshift console, select the Serverless dashboard, choose **Workgroup configuration**, and select the workgroup. Under **Data access**, choose **Edit** to view the **Network and security** settings. Note the VPC security group for the workgroup. Go to Amazon VPC and choose **Security groups** from the menu. Choose your security group ID in the list. The security group has configuration settings that include **Inbound rules**. Choose **Edit inbound rules** and create a rule that specifies the source IP address to allow, and the port.

2. On the Amazon VPC service console, verify that your VPC has the internet gateway attached. Confirm that the internet gateway's target is set with source 0.0.0.0/0 or a public IP CIDR. The route table must be associated with the VPC subnet where your cluster resides.

3. On your client, set an inbound firewall rule to accept traffic on the port you chose when you configured the workgroup and namespace.

4. Connect with your client tool, such as Amazon Redshift RSQL. Using your Amazon Redshift Serverless domain as the host, enter the following:

   ```
   rsq -h workgroup-name.account-id.region.amazonaws.com -U admin -d dev -p 5439
   ```

When you turn on the publicly accessible setting, Amazon Redshift Serverless creates an Elastic IP address. It's a static IP address that is associated with your AWS account. Clients outside the VPC can use it to connect. It gives you the ability to change your underlying network configuration without affecting client connections.

### Defining database roles to grant to federated users in Amazon Redshift Serverless

You can define roles in your organization that determine which database roles to grant in Amazon Redshift Serverless. For more information, see [Defining database roles to grant to federated users in Amazon Redshift Serverless](p. 31).

### Additional resources

For more information about secure connections to Amazon Redshift Serverless, including granting permissions, authorizing access to additional services, and creating IAM roles, see [Identity and access management in Amazon Redshift Serverless](p. 33).

### Defining database roles to grant to federated users in Amazon Redshift Serverless

When you're part of an organization, you have a collection of associated roles. For instance, you have roles for your job function, like *programmer* and *manager*. Your roles determine which applications and data you have access to. Most organizations use an identity provider, such as Microsoft Active Directory, to assign roles to users and groups. The use of roles to control resource access has grown, because organizations don't have to do as much management of individual users.

Recently, role-based access control was introduced in Amazon Redshift Serverless. Using database roles, you can secure access to data and objects, like schemas or tables, for example. Or you can use roles to define a set of elevated permissions, such as for a system monitor or database administrator. But after you grant resource permissions to database roles, there is an additional step, which is to connect a user's roles from the organization to the database roles. You can assign each user to their database roles upon initial sign in by running SQL statements, but it's a lot of effort. An easier way is to define the database
Defining database roles to grant to federated users in Amazon Redshift Serverless

You can pass roles to Amazon Redshift Serverless using GetCredentials. When a user signs in for the first time to an Amazon Redshift Serverless database, an associated database user is created and mapped to the matching database roles. This topic details the mechanism for passing roles to Amazon Redshift Serverless.

Passing database roles has a couple primary use cases:

- When a user signs in through a third-party identity provider, typically with federation configured, and passes the roles by means of a session tag.
- When a user signs in through IAM sign-in credentials, and their roles are passed by means of a tag key and value.

For more information about role-based access control, see Role-based access control (RBAC).

Configuring database roles

Before you can pass roles to Amazon Redshift Serverless, you must configure database roles in your database and grant them appropriate permissions on database resources. For instance, in a simple scenario, you can create a database role named sales and grant it access to query tables with sales data. For more information about how to create database roles and grant permissions, see CREATE ROLE and GRANT.

Use cases for defining database roles to grant to federated users

These sections outline a couple use cases where passing database roles to Amazon Redshift Serverless can simplify access to database resources.

Signing in using an identity provider

The first use case assumes that your organization has user identities in an identity and access management service. This service can be cloud based, for example JumpCloud or Okta, or on-premises, such as Microsoft Active Directory. The goal is to automatically map a user's roles from the identity provider to your database roles when they sign in to a client like Query editor V2, for instance, or with a JDBC client. To set this up, you must complete a couple of configuration tasks. These include the following:

1. Configure federated integration with your identity provider (IdP) using a trust relationship. This is a prerequisite. When you set this up, the identity provider is responsible for authenticating the user via a SAML assertion and providing sign-in credentials. For more information, see Integrating third party SAML solution providers with AWS. You can also find more information at Federate access to Amazon Redshift query editor V2 with Active Directory Federation Services (AD FS) or Federate single sign-on access to Amazon Redshift query editor v2 with Okta.

2. The user must have the following policy permissions:
   - GetCredentials – Provides credentials for temporary authorization to log in to Amazon Redshift Serverless.
   - sts:AssumeRoleWithSAML – Provides a mechanism for tying an enterprise identity store or directory to role-based AWS access.
   - sts:TagSession – Permission to the tag-session action, on the identity provider principal.

   In this case, AssumeRoleWithSAML returns a set of security credentials for users who have been authenticated via a SAML authenticated response. This operation provides a mechanism for tying an identity store or directory to role-based AWS access without user-specific credentials. For users with...
permission to AssumeRoleWithSAML, the identity provider is responsible for managing the SAML assertion that is used to pass the role information.

As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift.

3. You configure the tag RedshiftDbRoles with the colon-separated role values, in the format role1:role2. For example, manager:engineer. These can be retrieved from a session-tag implementation configured in your identity provider. The SAML authentication request passes the roles programmatically. For more information about passing session tags, see Passing session tags in AWS STS.

   In a case where you pass a role name that doesn't exist in the database, it's ignored.

   In this use case, when a user signs in using a federated identity, their roles are passed in the authorization request through the session tag key and value. Subsequently, following authorization, GetCredentials passes the roles to the database. Upon a successful connection, the database roles are mapped and the user can perform database tasks corresponding with their role. The essential part of the operation is that the RedshiftDbRoles session tag is assigned the roles in the initial authorization request. For more information about passing session tags, see Passing session tags using AssumeRoleWithSAML.

**Signing in using IAM credentials**

In the second use case, roles can be passed for a user and they can access a database client application through IAM credentials.

1. The user who signs in in this case must be assigned policy permissions for the following actions:
   - **tag:GetResources** – Returns tagged resources associated with specified tags.
   - **tag:GetTagKeys** – Returns tag keys currently in use.

   As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift.

2. Allow permissions are also required to access the database service, such as Amazon Redshift Serverless.

3. For this use case, configure the tag values for your roles in AWS Identity and Access Management. You can choose to edit tags and create a tag key called RedshiftDbRoles with an accompanying tag value string that contains the roles. For example, manager:engineer.

   When a user logs in, their role is added to the authorization request and passed to the database. It is mapped to an existing database role.

**Additional resources**

As mentioned in the use cases, you can configure the trust relationship between your IdP and AWS. For more information, see Configuring your SAML 2.0 IdP with relying party trust and adding claims.

**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 Amazon Redshift Serverless.
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. 455).

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 instance to act for you, supply security credentials to it. 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. 34).

To access other AWS services, create an IAM role with the appropriate permissions. You also need to associate the role with Amazon Redshift Serverless. 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. 536).

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. 34).

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. 545).

When you already have Amazon Redshift Serverless and want to configure IAM roles for it, 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. 545).

Assigning IAM roles to a namespace

Each IAM role is an AWS identity with permissions policies that determine what actions each role can perform in AWS. The role is intended to be assumable by anyone who needs it. Additionally, each namespace is a collection of objects, like tables and schemas, and users. When you use Amazon Redshift Serverless, you can associate multiple IAM roles with your namespace. This makes it easier to structure your permissions appropriately for a collection of database objects, so roles can perform actions on
both internal and external data. For example, so you can run a COPY command in an Amazon Redshift database to retrieve data from Amazon S3 and populate a Redshift table.

You can associate multiple roles to a namespace using the console, as described previously in this section. You can also use the API command CreateNamespace, or the CLI command create-namespace. With the API or CLI command, you can assign IAM roles to the namespace by populating IAMRoles with one or more roles. Specifically, you add ARNs for specific roles to the collection.

Managing namespace associated IAM roles

On the AWS Management Console you can manage permissions policies for roles in AWS Identity and Access Management. You can manage IAM roles for the namespace, using settings available under Namespace configuration. For more information about namespaces and their use in Amazon Redshift Serverless, see Overview of Amazon Redshift Serverless workgroups and namespaces (p. 39).

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, we recommend that you sign in as a user with an attached IAM role that has the policies required. After you get started creating an Amazon Redshift Serverless instance, Amazon Redshift records the IAM role name that you used to sign in. You can use the same credentials to sign in to the Amazon Redshift console and the Amazon Redshift Serverless console.

While creating the Amazon Redshift Serverless instance, you can create a database. 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 workgroup and namespace in the console for the first time, use an IAM role with a permissions policy attached. Make sure that this 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.

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 Amazon Redshift Serverless.
- If you choose Customize settings without specifying the Admin user name and password Amazon Redshift Serverless, your current IAM credentials are used as your default admin user credentials.
- 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.

Managing access to Amazon Redshift Serverless database objects with database-role permissions

This procedure shows how to grant permission to query a table through an Amazon Redshift database role. The role is assigned by means of a tag that's attached to a user in IAM and passed to Amazon
Redshift when they sign in. It's an explanation by example of the concepts in Defining database roles to grant to federated users in Amazon Redshift Serverless. The benefit of completing these steps is that you can associate a user with a database role and avoid setting their permissions for each database object. It simplifies managing the user's ability to query, modify, or add data to tables and to perform other actions.

The procedure assumes you have already set up an Amazon Redshift Serverless database and you have the ability to grant permissions in the database. It also assumes you have permissions to create an IAM user in the AWS console, to create an IAM role, and to assign policy permissions.

1. Create an IAM user, using the IAM console. Later, you will connect to the database with this user.
2. Create a Redshift database role, using query editor v2 or another SQL client. For more information on creating database roles, see CREATE ROLE.

CREATE ROLE urban_planning;

Query the SVV_ROLES system view to check that your role is created. It also returns system roles.

SELECT * from SVV_ROLES;

3. Grant the database role you created permission to select from a table. (The IAM user you created will eventually sign in and select records from the table by means of the database role.) The role name and table name in the following code example are samples. Here, permission is granted to select from a table named cities.

GRANT SELECT on TABLE cities to ROLE urban_planning;

4. Use the AWS Identity and Access Management console to create an IAM role. This role grants permission to use query editor v2. Create a new IAM role and, for the trusted entity type, choose AWS account. Then choose This account. Give the role the following policy permissions:

- AmazonRedshiftReadOnlyAccess
- tag:GetResources
- tag:GetTagKeys
- All actions for sqlworkbench, including sqlworkbench:ListDatabases and sqlworkbench:UpdateConnection.

5. In the IAM console, add a tag with the Key RedshiftDbRoles to the IAM user you created previously. The tag's value should match the database role you created in the first step. It's urban_planning in the sample.

After you complete these steps, assign the IAM role to the user you created in the IAM console. When the user signs in to the database with query editor v2, their database role name in the tag is passed to Amazon Redshift and associated with them. Thus, they can query the appropriate tables by means of the database role. To illustrate, the user in this sample can query the cities table through the urban_planning database role.

Migrating a provisioned cluster to Amazon Redshift Serverless

To migrate from a provisioned cluster to Amazon Redshift Serverless, see the following steps.
Creating a snapshot of your provisioned cluster

To transfer data from your provisioned cluster to Amazon Redshift Serverless, create a snapshot of your provisioned cluster, and then restore the snapshot in Amazon Redshift Serverless. Amazon Redshift automatically converts interleaved keys to compound keys when you restore a provisioned cluster snapshot to a serverless namespace.

**Note**
Before you migrate your data to a serverless workgroup, ensure that your provisioned cluster needs are compatible with the amount of RPU you choose in Amazon Redshift Serverless.

To create a snapshot of your provisioned cluster

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, Snapshots, and then choose Create snapshot.
3. Enter the properties of the snapshot definition, then choose Create snapshot. It might take some time for the snapshot to be available.

To restore a provisioned cluster snapshot to a serverless namespace:

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. Start on the Amazon Redshift provisioned cluster console and navigate to the Clusters, Snapshots page.
3. Choose a snapshot to use.
4. Choose Restore snapshot, Restore to serverless namespace.
5. Choose a namespace to restore your snapshot to.
6. 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. Choose Restore.

For more information about provisioned cluster snapshots, see [Amazon Redshift snapshots](https://docs.aws.amazon.com/redshift/latest/gsg/).  

Using a driver endpoint

To connect to Amazon Redshift Serverless with your preferred SQL client, you can use the Amazon Redshift provided JDBC driver version 2 driver. We recommend connecting using JDBC driver version 2.1.x or later. The port number is optional. If you don’t include it, Amazon Redshift Serverless defaults to port number 5439. You can change to another port from the port range of 5431-5455 or 8191-8215. To change the default port for a serverless endpoint, use the AWS CLI and Amazon Redshift API.

To find the exact endpoint to use for the JDBC, ODBC, or Python driver, see Workgroup configuration in Amazon Redshift Serverless. You can also use the Amazon Redshift Serverless API operation GetWorkgroup or the AWS CLI operation get-workgroups to return information about your workgroup, and then connect.

Connecting using password-based authentication

To connect using password-based authentication, use the following syntax.

```
jdbc:redshift://<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com:5439/?username=enter a username&password=enter a password
```
To connect using the Amazon Redshift Python driver, use the following syntax.

```python
import redshift_connector
with redshift_connector.connect(
    host='<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com',
    database='<database-name>',
    user='enter a user',
    password='enter a password'
# port value of 5439 is specified by default
) as conn:
    pass
```

### Connecting using IAM

If you prefer logging in with IAM, use the following driver endpoint. This driver endpoint lets you connect to a specific database and uses the Amazon Redshift Serverless [GetCredentials API](https://docs.aws.amazon.com/redshift-serverless/latest/dev/getcredentials.html) operation.

```
jdbc:redshift:iam://<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com:5439/<database-name>
```

This driver endpoint doesn't support customizing `dbUser`, `dbGroup` and `auto-create`. By default, the driver automatically creates database users at login and assigns them to groups according to the groups you defined in IAM. Note: Group names you specify in IAM must contain only lowercase letters, numbers, underscore ('_'), plus sign ('+'), period (dot), at symbol (@), or hyphen ('-'). Otherwise, the driver might not connect to `dbGroup`.

Ensure that your AWS identity has the correct IAM policy for the `RedshiftServerlessGetCredentials` action. The following is an example IAM policy that grants the correct permissions to an AWS identity to connect to Amazon Redshift Serverless. For more information about IAM permissions, see [Add IAM Identity Permissions](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_use-service-role-add-arn-policy-examples.html).

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

### Connecting using IAM with `dbUser` and `dbGroups`

If you want to use custom `dbUser` and `dbGroups` connection options, use the following driver endpoint. Like the other Amazon Redshift Serverless driver endpoint, this syntax automatically creates database users at login. This driver endpoint uses the Amazon Redshift Serverless [GetCredentials API](https://docs.aws.amazon.com/redshift-serverless/latest/dev/getcredentials.html) operation. `dbUser` must begin with a letter, must contain only alphanumeric characters, underscore ('_'), plus sign ('+'), dot ('.'), at symbol (@), or hyphen ('-'), and must be less than 128 characters. `dbGroups` must contain only lowercase letters, numbers, underscore ('_'), plus sign ('+'), period (dot), at symbol (@), or hyphen.

```
jdbc:redshift:iam://<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com/<database-name>
```
To connect using the Amazon Redshift Python driver, use the following syntax.

```python
import redshift_connector
with redshift_connector.connect(
    iam=True,
    host='<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com',
    database='<database-name>',
    db_user='enter a user',
    password='enter a password',
    db_groups='<db-groups>'
) as conn:
    pass
```

Connecting using ODBC

To connect using ODBC, use the following syntax.

```
Driver={Amazon Redshift (x64)}; Server=<workgroup-name>.<account-number>.<aws-region>.redshift-serverless.amazonaws.com; Database=dev
```

Using the Amazon Redshift Serverless SDK

If you wrote any management scripts using the Amazon Redshift SDK, you must use the new Amazon Redshift Serverless SDK to manage Amazon Redshift Serverless and associated resources. For more information about available API operations, see the Amazon Redshift Serverless API Reference guide.

Overview of Amazon Redshift Serverless workgroups and namespaces

To isolate workloads and manage different resources in Amazon Redshift Serverless, you can create namespaces and workgroups and manage storage and compute resources separately.

Overview of Amazon Redshift Serverless workgroups and namespaces

Namespace is a collection of database objects and users. The storage-related namespace groups together schemas, tables, users, or AWS Key Management Service keys for encrypting data. Storage properties include the database name and password of the admin user, permissions, and encryption and security. Other resources that are grouped under namespaces include datashares, recovery points, and usage limits. You can configure these storage properties using the Amazon Redshift Serverless console, the AWS Command Line Interface, or the Amazon Redshift Serverless APIs for the specific resource.

Workgroup is a collection of compute resources. The compute-related workgroup groups together compute resources like RPU, VPC subnet groups, and security groups. Properties for the workgroup include network and security settings. Other resources that are grouped under workgroups include access and usage limits. You can configure these compute properties using the Amazon Redshift Serverless console, the AWS Command Line Interface, or the Amazon Redshift Serverless APIs.

You can create one or more namespaces and workgroups. Each namespace can have only one workgroup associated with it. Conversely, each workgroup can be associated with only one namespace.
Getting started with Amazon Redshift Serverless using the console

Setting up Amazon Redshift Serverless involves walking through several configuration steps. When you follow the steps to set up Amazon Redshift Serverless, you create a namespace and workgroup, and associate them with each other. To get started setting Amazon Redshift Serverless configuration using the Amazon Redshift Serverless console, you can choose **Get started with Amazon Redshift Serverless** to set up Amazon Redshift Serverless and begin to interact with it. You can choose an environment with default settings, which makes for quicker setup, or explicitly configure the settings per your organization's requirements. During this process, you specify settings for your workgroup and namespace.

After you set up the environment, **Workgroup properties (p. 42)** and **Namespace properties (p. 44)** help you get familiar with the settings.

Managing Amazon Redshift Serverless using the AWS Command Line Interface and Amazon Redshift Serverless API

To create a namespace, you can use the AWS Command Line Interface operation `create-namespace` or the Amazon Redshift Serverless API operation `CreateNamespace`. Amazon Redshift Serverless creates a default namespace with a default AWS Key Management Service key. You can also specify another key to encrypt your data. Alternatively, you can use the AWS Command Line Interface operation `restore-from-snapshot` to restore an existing namespace from a snapshot. Amazon Redshift Serverless then populates the namespace with data restored from the specified snapshot. You can also use the API operation `RestoreFromSnapshot`.

To create a workgroup, ensure that you have an existing namespace. You can use the AWS Command Line Interface operation `create-workgroup` or the Amazon Redshift Serverless API operation `CreateWorkgroup`. Specify a target namespace that the workgroup is associated to. You can specify any compute resources while creating the workgroup, such as subnets, security groups, or RPUs. Once created, you can access the namespace that the workgroup is associated to.

To update a namespace or workgroup, you can use the AWS Command Line Interface operations `update-namespace` or `update-workgroup` or the Amazon Redshift Serverless API operations `UpdateNamespace` or `UpdateWorkgroup`.

To retrieve the list of instances of namespaces and workgroups, use the AWS Command Line Interface operations `list-namespaces` or `list-workgroups` or the Amazon Redshift Serverless API operations `ListNamespaces` or `ListWorkgroups`.

To retrieve the contents of or metadata for namespaces or workgroups, use the AWS Command Line Interface operations `get-namespaces` or `get-workgroups` or use the Amazon Redshift Serverless API operations `GetNamespace` or `GetWorkgroup`.

To delete a workgroup, use the AWS Command Line Interface operations `delete-workgroup` or the Amazon Redshift Serverless API operation `DeleteWorkgroup`. You won't be able to access the database after deleting a workgroup.

To delete a namespace, first delete the associated workgroup. Use the AWS Command Line Interface operation `delete-namespace` or the Amazon Redshift Serverless API operation `DeleteNamespace`. Amazon Redshift Serverless then deletes all the data in the specified namespace. Prior to the namespace deletion, you can take a final snapshot on the specified namespace to allow you to restore data from the snapshot, if needed. To do so, use the AWS Command Line Interface operation `create-snapshot` or the Amazon Redshift Serverless API operation `CreateSnapshot` before you use the AWS Command Line Interface operation `delete-namespace` or the Amazon Redshift Serverless API operation `DeleteNamespace`.
Managing Amazon Redshift Serverless using the console

To create, edit, and delete your Amazon Redshift Serverless data warehouse, use the Serverless dashboard on the Amazon Redshift console. Access to individual console settings depends on your IAM role and your permissions.

For information on setting up Amazon Redshift Serverless, see Setting up Amazon Redshift Serverless for the first time. For information on creating and configuring workgroups, see Working with workgroups (p. 41). For information on configuring namespaces, see Working with namespaces (p. 44).

Setting up Amazon Redshift Serverless for the first time

The first time you select the Serverless dashboard, it walks you through the steps to set up Amazon Redshift Serverless. Under Get started with the serverless experience, you can set up your Amazon Redshift Serverless data warehouse, using a sample dataset. Amazon Redshift Serverless automatically loads the sample dataset during the creation process. After the data warehouse is created, you can immediately query the data. For more information on how to set up Amazon Redshift Serverless for the first time, see Set up Amazon Redshift Serverless for the first time.

Working with workgroups

To isolate workloads and manage resources in Amazon Redshift Serverless, you can create workgroups and namespaces. The compute-related workgroup groups together compute resources like RPUs and VPC subnet groups. If you haven't created a workgroup and a namespace, and you are looking for instructions that show how to get started with Amazon Redshift Serverless, see Setting up Amazon Redshift Serverless for the first time.

Creating a workgroup with a namespace

These steps assume you have completed the initial configuration for Amazon Redshift Serverless. If you haven't created a workgroup and a namespace, and you are looking for instructions that show how to get started with Amazon Redshift Serverless, see Setting up Amazon Redshift Serverless for the first time.

Complete the following steps to create a workgroup:

1. Choose the Serverless dashboard. Then choose Create workgroup.
2. Enter the workgroup name.
3. Choose a Virtual private cloud (VPC) for Amazon Redshift Serverless. This assigns the workgroup to a specific virtual network in your AWS environment. For more information about VPCs, see Overview of VPCs and subnets.
4. Choose one or more VPC security groups. For more information, see Control traffic to resources using security groups.
5. Under Subnet, specify one or more subnets to associate with your database. These subnets are contained in the VPC you chose previously and must be in three distinct Availability Zones. For more information, see Considerations when using Amazon Redshift Serverless.
6. Select the base RPU capacity that conforms with your requirements.
Choose a namespace

1. Choose either **Create a new namespace**, and enter the namespace name, or **Add to an existing namespace**, and select the namespace from the drop-down list.

2. For **Database name and password**, specify the name of the first database. You can also specify an admin other than your default console admin, by editing the **Admin user credentials**.

3. For **Permissions**, you choose **Associate IAM role** to associate specific IAM roles with your namespace and workgroup. For more information about associating IAM roles with Amazon Redshift, see [Identity and access management in Amazon Redshift](#).

4. You can customize your encryption settings by creating a new key or choosing a key other than the default. For **Audit logging**, choose the logs to export. Each log type specifies different metadata. Choose **Continue** to review your choices.

Review workgroup selections

1. Review your settings under **Review and create**. It shows the settings you chose in the previous steps.

2. Choose **Save**.

After you create the workgroup, it's added to the **Workgroups** list.

Viewing properties for a workgroup

In Amazon Redshift Serverless, a workgroup is a collection of resources available for use. When you choose Amazon Redshift Serverless, in the AWS console, you can choose **Workgroup configuration** from the navigation menu to view a list. You can use the **Search** box to find workgroups that meet your search criteria. Each workgroup entry has a few properties displayed:

- **Workgroup** - The name of the workgroup. You can select it to view and edit the workgroup's properties.
- **Status** - Shows whether the workgroup is available.
- **Namespace** - The namespace associated with the workgroup. Each workgroup is associated with one namespace.
- **Creation date** - The date the workgroup was created.

Workgroup properties

You can list workgroups by choosing **Workgroup configuration** in the left menu. Then you can choose a workgroup from the list. Several panels show properties for the workgroup. You can also perform actions. **General information** displays the following:

- **Workgroup** - The name of the workgroup.
- **Namespace** - The namespace associated with the workgroup. You can choose it to view its properties. A workgroup is associated with a single namespace.
- **Date created** - When the workgroup was created.
- **Status** - Indicates if the workgroup resources are available. If it's available, you can connect with a client to the Amazon Redshift Serverless instance, in order to query data or create database resources, or you can connect with query editor v2.
- **Endpoint** - The URL.
- **JDBC URL** - The URL to establish JDBC client connections. You can use this URL to connect with a JDBC driver for Amazon Redshift. For more information, see [Configuring a connection for JDBC driver version 2.1 for Amazon Redshift](#).
• **ODBC URL** - The URL to establish ODBC client connections. It contains properties, like the database and user ID, and their values.

The **Data access** tab contains several panels:

• **Network and security** - You can see network properties, such as the Virtual private cloud (VPC) identifier, VPC security group list, Enhanced VPC routing, and the Publicly accessible setting. If you choose Edit, you can change these settings. Additionally, you can select Turn on enhanced VPC routing, which routes network traffic between your serverless database and data repositories through a VPC, for enhanced privacy and security. You can also select Turn on Public Accessible, which makes the database publicly accessible from outside the VPC, allowing instances and devices to connect.

• **Redshift managed VPC endpoints** - You can create managed VPC endpoints to access Amazon Redshift Serverless from another VPC.

The **Limits** tab has settings for controlling capacity and use limits for Amazon Redshift Serverless. It contains the following panels:

• **Base capacity in Redshift processing units (RPUs)** - You can set the base capacity of the compute resources used to process your workload. For more information, see Understanding Amazon Redshift Serverless capacity (p. 23).

• **Usage limits** - You can set maximum RPU hours at various frequencies. For instance, you can set a daily RPU maximum. These limits help control your cost and make it more predictable.

• **Query limits** - You can set limits on queries, like the timeout setting. These limits help you optimize cost and performance.

**Deleting a workgroup**

You can delete a workgroup using the console. Before you do this, make sure that you have your data backed up and snapshots in place. Resources deleted as part of the workgroup in many cases can't be retrieved.

Complete the following steps:

1. Choose Amazon Redshift Serverless, choose Workgroup configuration and choose Delete Amazon Redshift Serverless instance.

2. A dialogue opens. When you choose to delete the workgroup, all usage limits are removed, all VPC endpoints are removed, and access to VPC endpoints is removed.

   Type delete and select Delete to confirm.

After you complete the steps, the status of the workgroup is Deleting and a banner indicates that the workgroup is being deleted. While the delete process is in progress, some features under the Serverless dashboard are disabled. But you can configure provisioned clusters on the Provisioned clusters dashboard.

After you delete the workgroup, it doesn't appear with the namespace. You can choose the Create workgroup button to create a new one.

You can delete an existing workgroup and associate a new workgroup with a different configuration to the same namespace. When creating the new workgroup, choose the base capacity that works with the size of the data associated with the namespace.

You can associate a workgroup with a namespace that was created with a customer-managed key (CMK). For more information about AWS KMS, see AWS KMS concepts.
Working with namespaces

In Amazon Redshift Serverless, a namespace defines a logical container for database objects. It can hold tables, workgroups, and other database resources. If you haven't created a workgroup and a namespace, and you are looking for instructions in how to get started with Amazon Redshift Serverless, see Setting up Amazon Redshift Serverless for the first time.

Searching for a namespace

From the Amazon Redshift menu, you can choose from the Namespaces list in order to view or edit the properties for a namespace. Information on the console includes the namespace name, the admin name, and other properties.

A namespace's settings and properties are on several tabs. These include the following:

- **Workgroup** - Shows workgroups associated with the namespace.
- **Data back up** - You can configure and create snapshots, and configure recovery points.
- **Security and encryption** - You can manage IAM role permissions and view or edit your security and encryption settings. These include your encryption key status and your audit logging settings.
- **Datashares** - Shows datashares.

Namespace properties

In Amazon Redshift Serverless, a namespace defines a container for database objects. You can choose Namespace configuration from the navigation list, choose a namespace from the list, and edit its settings.

General information for the namespace includes the following:

- **Namespace** - The name.
- **Namespace ID** - The unique identifier.
- **ARN** - A unique identifier used to specify the resource across AWS. It contains properties like the region and the service.
- **Status** - The status, such as **Available**.
- **Date created** - The date the namespace was created.
- **Storage used** - The storage space used by the namespace and all of its objects.
- **Admin user name** - The admin account. This is typically the account used to create the namespace.
- **Database name** - The name of the database contained by the namespace.
- **Total table count** - The count of tables in all schemas.

Additional settings and properties for the namespace are on several tabs. These include the following:

- **Workgroup** - Shows the workgroup associated with the namespace.
- **Data back up** - On this panel, you can configure and create snapshots, and configure recovery points.
- **Security and encryption** - You can manage IAM role permissions and view or edit your security and encryption settings. These include your encryption key status, and the setting to turn on audit logging. For more information about audit logging for Amazon Redshift Serverless, see [Audit logging for Amazon Redshift Serverless](#).
- **Datashares** - Shows datashares. With data sharing, you can provide access to data without the need to copy it or move it. For more information about data sharing, see [Data sharing in Amazon Redshift Serverless](#).
Editing security and encryption

Amazon Redshift Serverless is secured by means of KMS encryption. You can update encryption settings via the console:

1. Choose **Namespace configuration** from the main menu on the console, choose the namespace to edit, and choose **Edit** on the **Security and encryption** tab. A dialog appears.
2. You can select **Customize encryption settings** and then **Choose an AWS customer managed key** to change the key used to encrypt your resources.
3. For **Audit logging**, choose the logs to export. Each log type specifies different metadata.
4. To complete the configuration update, choose **Save changes**.

Changing the AWS KMS key for a namespace

In Amazon Redshift, encryption protects data at rest. Amazon Redshift Serverless uses AWS KMS key encryption automatically to encrypt both your Amazon Redshift Serverless resources and snapshots. As a best practice, most organizations review the type of data they store and have a plan to rotate encryption keys on a schedule. The frequency for rotating keys can vary, depending on your policies for data security. Amazon Redshift Serverless supports changing the AWS KMS key for the namespace so you can adhere to your organization's security policies.

When you change the AWS KMS key, the data remains unchanged.

Changing an AWS KMS key using the console

In Amazon Redshift, encryption protects data at rest. Amazon Redshift Serverless uses AWS KMS key encryption automatically to encrypt both Amazon Redshift Serverless and snapshots. As a best practice, most organizations review the type of data they store and have a plan to rotate encryption keys on a schedule. The frequency for rotating keys can vary, depending on your policies for data security. Amazon Redshift Serverless supports changing the AWS KMS key for the namespace so you can adhere to your organization's security policies.

When you change the AWS KMS key, the data remains unchanged.

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 **Namespace configuration**. Choose your namespace from the list.
3. From the **Security and encryption** tab, choose **Edit**.
4. Choose **Customize encryption settings** and then choose a key for the namespace. You can optionally create a new key.

Changing AWS KMS encryption keys using the AWS CLI

Use `update-namespace` to change the AWS KMS key for the namespace. The following shows the syntax for the command:

```bash
aws redshift-serverless update-namespace
--namespace-name
[--kms-key-id <id-of-kms-key>]
// other parameters omitted here
```

You must have a namespace created or the CLI command results in an error.
The time it takes to change the key depends on the amount of data in Amazon Redshift Serverless. This typically takes fifteen minutes per 8TB of stored data.

Limitations

You can't change from a customer managed KMS Key to an AWS KMS key. In this case, you have to create a new namespace.

You can't perform other actions while the key is being changed.

Deleting a namespace

If you want to delete a namespace with an associated workgroup, you have to first delete the workgroup.

On the Amazon Redshift Serverless console, complete the following steps:

1. Choose **Namespace configuration** from the left menu and then choose the namespace you want to delete from the list.
2. Choose **Actions** and select **Delete namespace**.
3. A dialogue box opens. You can keep your data by creating a manual snapshot prior to completing the delete operation.

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

Managing usage limits, query limits, and other administrative tasks

You can configure settings in the console to control usage and limit cost.

Managing usage limits, including setting RPU limits

Under the **Limits** tab for a workgroup, you can add one or more usage limits to control the maximum RPU units you use in a given time period, or to set a data sharing usage limit.

1. Choose **Manage usage limits**. Choose **Add limit** on the **Redshift processing units (RPUs) maximum** panel.
2. First, you choose a **Frequency**, which is either **Daily**, **Weekly**, or **Monthly**. This sets the time period for the limit. Choosing **Daily** in this instance gives you more detailed control.
3. Set a usage limit, in number of hours.
4. Set the action. These are the following:
   - **Log to system table** - Adds a record to a system table, which you can query to determine if the limit is exceeded.
   - **Alert** - Uses Amazon SNS to set up notification subscriptions and send notifications if a limit is breached. You can choose an existing Amazon SNS topic or create a new one.
   - **Turn off user queries** - Disables queries to stop use of Amazon Redshift Serverless. It also sends a notification.

   The first two actions are informational, but the last turns off query processing.

5. Optionally, you can set a **Cross-Region data sharing usage limit**, which limits how much data transferred from producer Region to consumer Region consumers can query. To do this, choose **Add limit** and follow the steps.
6. Choose **Save changes** to save the limit.

For more conceptual information about RPUs and billing, see [Billing for Amazon Redshift Serverless](#).

### Managing query limits

Under the **Limits** tab for a workgroup, you can add a limit to monitor performance and limits. For more information about query monitoring limits, see [WLM query monitoring rules](#).

1. Choose **Manage query limits**. Choose **Add new limit** on the **Manage query limits** dialogue.
2. Choose the limit type you want to set and enter a value for its corresponding limit.
3. Choose **Save changes** to save the limit.

When you change your query limit and configuration parameters, your database will restart.

### 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.

### Changing your admin password

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

### Checking Amazon Redshift Serverless summary data using the dashboard

The Amazon Redshift Serverless dashboard contains a collection of panels that show at-a-glance metrics and information about your workgroup and namespace. These panels include the following:

- **Resources summary** - Displays high-level information about Amazon Redshift Serverless, such as 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 Amazon Redshift Serverless, or review queries, or track work items.
You can monitor your Amazon Redshift Serverless 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.

```json
{
  "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": "*"
    }
  ]
}
```

5. Choose Review policy.
6. For Name, enter a name for the policy, such as query-monitoring.
7. Choose Create policy.

After you create the policy, you can grant the appropriate permissions.

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center (successor to AWS Single Sign-On):

  Create a permission set. Follow the instructions in Create a permission set in the AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide.
Users managed in IAM through an identity provider:

Create a role for identity federation. Follow the instructions in Creating a role for a third-party identity provider (federation) in the IAM User Guide.

IAM users:

• Create a role that your user can assume. Follow the instructions in Creating a role for an IAM user in the IAM User Guide.

• (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the IAM User Guide.

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 role 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 role 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.

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

```
grant role sys:monitor to "IAMR:role-name";
```

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

```
grant role sys:operator to "IAMR:role-name";
```

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 Amazon Redshift Serverless, which is much simpler than that needed for provisioned clusters. The SYS system views have been designed to work with Amazon Redshift Serverless. 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 Amazon Redshift Serverless 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 must 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 how to improve performance.

User objects event monitoring

You need to monitor actions and activities on user objects, such as 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 run 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_ALTER_TABLE_RECOMMENDATIONS
- SVV_COLUMNS
- SVV_COLUMN_PRIVILEGES
- SVV_DATABASE_PRIVILEGES
- SVV_DATASHARES
- SVV_DATASHARE_CONSUMERS
- SVV_DATASHARE_OBJECTS
- SVV_DATASHARE_PRIVILEGES
- SVV_DEFAULT_PRIVILEGES
You can query the following SYS system views to monitor Amazon Redshift Serverless.

- **SYS_CONNECTION_LOG**
- **SYS_DATASHARE_CHANGE_LOG**
- **SYS_DATASHARE_USAGE_CONSUMER**
- **SYS_DATASHARE_USAGE_PRODUCER**
- **SYS_QUERY_HISTORY**
- **SYS_QUERY_DETAIL**
- **SYS_EXTERNAL_QUERY_DETAIL**
- **SYS_LOAD_DETAIL**
- **SYS_LOAD_HISTORY**
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.

You can create CloudWatch alarms to track your metrics using the Amazon Redshift console. For more information on creating alarms, see Managing alarms.

To export generated log data to Amazon CloudWatch Logs, the respective logs must be selected for export in your Amazon Redshift Serverless configuration settings, on the console. You can do this by choosing the Namespace configuration settings, under Security and encryption.

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, in which log_type represents the log type.

/aws/redshift/<namespace>/<log_type>

When you create your first workgroup and namespace, default is the namespace name. The log group name varies according to what you call the namespace.

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

/aws/redshift/default/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 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.

- **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. 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, 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

Amazon Redshift Serverless metrics are divided into compute metrics and data and storage metrics, falling under the workgroup and namespace dimension sets, respectively. For more information about workgroups and namespaces, see [Overview of Amazon Redshift Serverless workgroups and namespaces](#).

CloudWatch compute metrics are the following:

**CloudWatch compute metrics**

<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>{Database, LatencyRange, Workgroup}, {LatencyRange, Workgroup}</td>
</tr>
<tr>
<td>QueryDuration</td>
<td>Microseconds</td>
<td>The average amount of time to complete a query.</td>
<td>{Database, LatencyRange, Workgroup}, {LatencyRange, Workgroup}</td>
</tr>
<tr>
<td>QueriesRunning</td>
<td>Number of queries</td>
<td>The number of running queries at a point in time.</td>
<td>{Database, QueryType, Workgroup}, {QueryType, Workgroup}</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>{Database, QueryType, Workgroup}, {QueryType, Workgroup}</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>{Database, Workgroup}, {Workgroup}</td>
</tr>
<tr>
<td>QueryRuntimeBreakdown</td>
<td>Milliseconds</td>
<td>The total time queries ran, by query stage.</td>
<td>{Database, Stage, Workgroup}, {Stage, Workgroup}</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>{Workgroup}</td>
</tr>
<tr>
<td>ComputeSeconds</td>
<td>RPU-seconds</td>
<td>Accumulated compute-unit seconds used in the last 30 minutes.</td>
<td>{Workgroup}</td>
</tr>
<tr>
<td>QueriesSucceeded</td>
<td>Number of queries</td>
<td>The number of queries that</td>
<td>{Database, QueryType, Workgroup},</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Metric name</th>
<th>Units</th>
<th>Description</th>
<th>Dimension sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>succeeded in the last 5 minutes.</td>
<td></td>
<td></td>
<td>{QueryType, Workgroup}</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>{Database, QueryType, Workgroup},</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{QueryType, Workgroup}</td>
</tr>
<tr>
<td>UsageLimitAvailable</td>
<td>RPU-hours or TBs</td>
<td>Depending on the UsageType, UsageLimitAvailable returns the following:</td>
<td>{UsageLimitId, UsageType, Workgroup}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the UsageType is SERVERLESS_COMPUTE, UsageLimitAvailable returns the remaining number of RPU-hours that the workgroup can query in the given limit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the UsageType is CROSS_REGION_DATASHARING, UsageLimitAvailable returns the remaining number of TBs that the customer can scan in the given limit.</td>
<td></td>
</tr>
</tbody>
</table>
## Exporting logs

<table>
<thead>
<tr>
<th>Metric name</th>
<th>Units</th>
<th>Description</th>
<th>Dimension sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>UsageLimitConsumed</td>
<td>RPU-hours or TBs</td>
<td>Depending on the UsageType, UsageLimitConsumed returns the following:</td>
<td>{UsageLimitId, UsageType, Workgroup}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the UsageType is SERVERLESS_COMPUTE, UsageLimitConsumed returns the number of RPU-hours that the workgroup has already queried in the given limit.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the UsageType is CROSS_REGION_DATASHARING, UsageLimitConsumed returns the number of TBs that the customer has already used to scan in the given limit.</td>
<td></td>
</tr>
</tbody>
</table>

CloudWatch data and storage metrics are the following:

### CloudWatch data and storage metrics

<table>
<thead>
<tr>
<th>Metric name</th>
<th>Units</th>
<th>Description</th>
<th>Dimension sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>TotalTableCount</td>
<td>Number of tables</td>
<td>The number of user tables existing at a point in time. This total doesn't include Amazon Redshift Spectrum tables.</td>
<td>{Database, Namespace}</td>
</tr>
<tr>
<td>DataStorage</td>
<td>Megabytes</td>
<td>The number of megabytes used, in disk or storage space, for Redshift data.</td>
<td>{Namespace}</td>
</tr>
</tbody>
</table>

The SnapshotStorage metric is namespace- and workgroup-agnostic. CloudWatch's SnapshotStorage metric is as follows:
CloudWatch SnapshotStorage metric

<table>
<thead>
<tr>
<th>Metric name</th>
<th>Units</th>
<th>Description</th>
<th>Dimension sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>SnapshotStorage</td>
<td>Megabytes</td>
<td>The number of megabytes used, in disk or storage space, for Snapshots.</td>
<td>{}</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>
<tr>
<td>Namespace</td>
<td>The name of the namespace. A custom value.</td>
</tr>
<tr>
<td>Workgroup</td>
<td>The name of the workgroup. A custom value.</td>
</tr>
</tbody>
</table>
Working with snapshots and recovery points

A serverless snapshot includes all objects and data within a namespace. There are three scenarios in which you can restore snapshots:

- Restore a serverless snapshot to a serverless namespace.
- Restore a serverless snapshot to a provisioned cluster.
- Restore a provisioned cluster snapshot to a serverless namespace.

You can restore a snapshot you created on the Amazon Redshift Serverless console to a provisioned Amazon Redshift cluster. When you do so, you choose the node type to use, such as RA3, and the number of nodes, letting you control settings at the cluster or node level. To initiate the restore, you choose a snapshot from the Amazon Redshift Serverless console and then specify the ID of the provisioned cluster to restore to.

To restore a provisioned cluster snapshot to a serverless namespace, start from the Redshift provisioned console, choose the snapshot to restore, then choose Restore from snapshot, Restore to serverless namespace. Amazon Redshift converts tables with interleaved keys into compound sort keys when you restore a provisioned cluster snapshot to a serverless namespace. For more information about sort keys, see Working with sort keys.

Restoring a snapshot to a serverless namespace replaces the database with the database in the snapshot. You can also restore recovery points, which are created approximately every 30 minutes and kept for 24 hours, to your serverless namespace.

Restoring a snapshot to a serverless namespace is completed in two phases. The first phase completes in a few minutes, restores the data to your namespace, and makes it available for queries. The second phase of restoration is where your database is tuned, which can cause minor performance issues. This second phase can last from a few hours to several days, and in some cases, a couple of weeks. The amount of time depends on the size of the data, but performance progressively improves as the database gets tuned. At the end of this phase, your serverless namespace is fully tuned, and you can submit queries without performance issues.

If you want to add additional context, you can tag snapshots and recovery points with key-value pairs that provide metadata and information to snapshots and recovery points. For more information about tagging resources, see Tagging resources overview.

Finally, you can also share snapshots with other AWS accounts, which lets them access data within the snapshot and run queries.

You can view a snapshot's details by viewing the details page in the Amazon Redshift Serverless console, or by calling the Amazon Redshift Serverless API operation ListSnapshots or the list-snapshots.
Snapshots

You can restore a snapshot that you created on the Amazon Redshift Serverless console to an available namespace that's associated with a workgroup. A namespace is available once it is ready for querying and/or modifying. You can restore a snapshot encrypted with an AWS managed KMS key to a serverless namespace.

To see a list of all of your snapshots, on the Amazon Redshift Serverless console, choose Data backup.

To create a snapshot

1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose Create snapshot.
3. Choose a namespace to create a snapshot of.
4. Enter a snapshot identifier.
5. (Optional) Choose a retention period. If you choose Custom value, choose the number of days. The amount you choose must be between 1-3653 days, inclusive. The default is retain indefinitely.
6. Choose Create.

To create a snapshot from namespace configuration

1. On the Amazon Redshift Serverless console, choose Namespace configuration.
2. Choose the namespace to create a snapshot of. You can only create snapshots of namespaces that are associated with a workgroup and whose statuses are Available.
3. Choose the Data backup tab.
4. Choose Create snapshot.
5. Enter a snapshot identifier.
6. (Optional) Choose a retention period. If you choose Custom value, choose the number of days. The amount you choose must be between 1-3653 days, inclusive.
7. Choose Create.

To restore a snapshot to a serverless namespace

1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose the snapshot to restore. You can only restore one snapshot at a time.
3. Choose Actions, Restore to serverless namespace.
4. Choose an available namespace to restore to. You can only restore to namespaces whose statuses are Available.
5. Choose Restore.

To restore a snapshot to a provisioned cluster

1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose a snapshot to restore.
3. Choose Action, Restore to provisioned cluster.
4. Enter a cluster identifier.
5. Choose a Node type. The number of nodes depends on the node type.
6. Follow the instructions on the page on the console page to enter the properties for Cluster configuration. See Creating a cluster for more information.

To update a snapshot's retention period
1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose a snapshot to update.
4. Choose a retention period. If you choose Custom value, choose the number of days.
5. Choose Save changes.

To delete a snapshot
Note
You can't delete a snapshot that's been shared with another account. You must first remove that account's access to the snapshot before deleting the snapshot.

1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose a snapshot to delete.
3. Choose Actions, Delete.
4. Choose Delete.

To create a final snapshot of all data within a namespace before deleting the namespace.

1. On the Amazon Redshift Serverless console, choose Namespace configuration.
2. Choose the namespace to delete.
3. Choose Actions, Delete.
4. Choose Create final snapshot.
5. Enter a name for the snapshot.
6. Enter delete.
7. Choose Delete.

To share a snapshot with another AWS account or remove an account's access to a snapshot

1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose a snapshot to share.
3. Choose Actions, Manage access.
4. To share a snapshot with another account, enter an AWS account ID. To remove access from an account, choose Remove.
5. Choose Save changes.

For more information about snapshots on provisioned clusters, see Amazon Redshift snapshots in the Amazon Redshift Management Guide.

Recovery points

Recovery points in Amazon Redshift Serverless are created approximately 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 recovery point to a serverless namespace.
• Convert a recovery point to a snapshot.

To restore a recovery point to a serverless namespace
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. You can only restore to namespaces whose statuses are Available.
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.

Restoring a table from a snapshot

Rather than restore an entire snapshot to your serverless namespace, you can also restore a specific table from a snapshot. When you do so, you specify the source snapshot, database, schema, table, and the target database, schema, and new table name. This new table cannot have the same name as an existing table. If you want to replace an existing table by restoring a table from a snapshot, you must first rename or drop the table before you restore the table.

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, aren't 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.

You can use the Amazon Redshift Serverless console to restore tables from a snapshot.

Restoring a table from a snapshot has the following limitations:

• You can only restore one table at a time.
• Any dependencies, such as views or permissions granted on the source table, aren't applied to the target table.
• If row-level security is turned on for a table being restored, Amazon Redshift Serverless restores the table with row-level security turned on.

To restore a table from a snapshot using the Amazon Redshift Serverless console
1. On the Amazon Redshift Serverless console, choose Data backup.
2. Choose a snapshot to restore.
3. Choose Actions, Restore table from snapshot.
4. Enter information about the source snapshot and target table, then choose Restore table.

Managing snapshots and recovery points using the AWS Command Line Interface and Amazon Redshift Serverless API

Aside from using the AWS console, you can also use the AWS CLI or the Amazon Redshift Serverless API to interact with snapshots and recovery points.

To create a snapshot, use the AWS Command Line Interface operation create-snapshot. You can also use the Amazon Redshift Serverless API operation CreateSnapshot. Snapshots must be associated with a namespace, so you must include the name of a namespace in your request. By default, Amazon Redshift Serverless keeps snapshots for an indefinite period, but you can specify a retention period in your request.

To restore a snapshot, use restore-from-snapshot or RestoreFromSnapshot. If you're restoring a snapshot from Amazon Redshift Serverless to a provisioned cluster, you must specify the snapshotArn of the snapshot you're restoring. Otherwise, if you're restoring from serverless to serverless, you can specify snapshotArn or snapshotName, but not both.

To retrieve information about a snapshot, such as which namespace it is associated with or how large the data inside a snapshot is, use the AWS CLI operation get-snapshot. Alternatively, you can also use the API operation GetSnapshot. To get information about multiple snapshots at once, use list-snapshots or ListSnapshots.

To delete a snapshot, use the CLI operation delete-snapshot or the API operation DeleteSnapshot.

Recovery points are automatically created by Amazon Redshift Serverless, so you cannot use a AWS CLI or API operation to create recovery points. You can use restore-from-recovery-point or RestoreFromRecoveryPoint to restore the data within a recovery point to a namespace, just like restoring data from a snapshot. You can also convert a recovery point to a snapshot by using convert-recovery-point-to-snapshot or ConvertRecoveryPointToSnapshot.

To see information about a specific recovery point, use get-recovery-point or GetRecoveryPoint. To retrieve information about multiple recovery points, use list-recovery-points or ListRecoveryPoints.

Data sharing in Amazon Redshift Serverless

Use data sharing to share the most up-to-date and consistent information as it's updated in Amazon Redshift Serverless.

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 Amazon Redshift Serverless.

Getting started with data sharing in Amazon Redshift Serverless

You can share data for read purposes across different Amazon Redshift Serverless instances 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.

With data sharing, Amazon Redshift Serverless namespaces and provisioned clusters can share live data with each other, whether they are within an AWS account across AWS accounts, or across AWS Regions. For more information, see Regions where data sharing is available.

To get started sharing data within an AWS account, open the AWS Management Console, and then choose the Amazon Redshift console. Choose Namespace 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.

To start querying data in a datashare, create a database in a namespace that has a workgroup associated with it. From a specified datashare, choose a namespace that has a workgroup associated with it and create a database to query data. Follow the procedures in Creating databases from datashares.

Granting access to view datashares using the console

A superuser can provide access to users who aren’t superusers so that they can view the datashares created by all users.

To grant access to a datashare for a user, use the following command to provide datashare access for a user, where datashare_name is the name of the datashare and user-name is the name of the user for whom you want to provide access.

```
grant share on datashare datashare_name to "IAM:test_user";
```

To grant access to a datashare for a user group, first create a user group with users. For information on how to create user groups, see CREATE GROUP. Then, grant datashare access to a user using the following command, where datashare_name is the name of the datashare and user-group is the name of the user-group to that you want to grant access.

```
grant share on datashare datashare_name to group user_group;
```

For information on how to use the GRANT statement, see GRANT.

Data sharing considerations in Amazon Redshift Serverless

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

- Amazon Redshift only supports provisioned clusters of instance type ra3.16xlarge, ra3.4xlarge, and ra3.xlplus, and serverless endpoint as data sharing producers or consumers.
- Amazon Redshift Serverless is encrypted by default.

For a list of datasharing limitations, including database objects supported, encryption requirements, and sort-key requirements, see Considerations when using data sharing in Amazon Redshift in the Amazon Redshift Database Developer Guide.

Tagging resources overview

In AWS, tags are user-defined labels that consist of key-value pairs. Amazon Redshift Serverless supports tagging to provide metadata about resources at a glance.
Tags are not required for resources, but they help provide context. You might want to tag resources with metadata with 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 for consistency.

If you delete a resource, any associated tags are deleted. You can use both the AWS CLI and Amazon Redshift Serverless console to tag serverless resources. Available API operations are TagResource, UntagResource, and ListTagsForResource.

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 serverless resource types:

- Workgroups
- Namespaces
- Snapshots
- Recovery points

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.

To manage tags of your Amazon Redshift Serverless resources

1. On the Amazon Redshift Serverless console, choose Manage Tags.
2. Enter the resource type to search for and choose Search resources. Choose the resource for which you want to manage tags, then choose Manage tags.
3. Specify the keys and optional values you want to add to the resource. When modifying a tag, you can change the tag's value, but not the key.
4. After you're done adding, removing, or modifying tags, choose Save changes, then choose Apply to save your changes.
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. 64)
- Use EC2-VPC when you create your cluster (p. 69)
- Overview of RA3 node types (p. 70)
- Upgrading to RA3 node types (p. 72)
- Upgrading from DC1 node types to DC2 node types (p. 75)
- Upgrading a DS2 cluster on EC2-Class to EC2-VPC (p. 76)
- Region and Availability Zone considerations (p. 77)
- Cluster maintenance (p. 77)
- Default disk space alarm (p. 81)
- Cluster status (p. 81)
- Overview of managing clusters in Amazon Redshift (p. 83)
- Managing usage limits in Amazon Redshift (p. 95)
- Managing cluster relocation in Amazon Redshift (p. 96)
- Configuring Multi-AZ deployment (preview) (p. 99)
- Using a custom domain name for client connections (p. 105)
- Working with Redshift-managed VPC endpoints in Amazon Redshift (p. 109)
- Managing clusters using the console (p. 112)
- Managing clusters using the AWS CLI and Amazon Redshift API (p. 118)
- Managing clusters using the AWS SDK for Java (p. 119)
- Managing clusters in a VPC (p. 121)
- Cluster version history (p. 131)

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.

Preview features when using Amazon Redshift clusters

You can create an Amazon Redshift cluster in Preview to test new features of Amazon Redshift. You can't use those features in production or move your Preview cluster to a production cluster or a cluster on another track. For preview terms and conditions, see Beta and Previews in AWS Service Terms.
To create a cluster in Preview

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 Provisioned clusters dashboard, and 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. A banner displays on the Clusters list page that introduces preview. Choose the button Create preview cluster to open the create cluster page.

4. Enter properties for your cluster. Choose the Preview track that contains the features you want to test. We recommend entering a name for the cluster that indicates that it is on a preview track. Choose options for your cluster, including options labeled as -preview, for the features you want to test. For general information about creating clusters, see Creating a cluster in the Amazon Redshift Management Guide.

5. Choose Create preview cluster to create a cluster in preview.

6. When your preview cluster is available, use your SQL client to load and query data.

The following features are currently available in preview clusters:

- SUPER data type larger than 1 MB – see Limitations in the Amazon Redshift Database Developer Guide.
- Auto-copy – see Continuous file ingestion from Amazon S3 in the Amazon Redshift Database Developer Guide.
- Multi-AZ – see Managing Multi-AZ using the console.
- Query Data Catalog – see Querying the AWS Glue Data Catalog (p. 174).

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.

**Note**

When you create a cluster on the Amazon Redshift console (https://console.aws.amazon.com/redshift/), you can get a recommendation of your cluster configuration based on the size of your data and query characteristics. To use this sizing calculator, look for Help me choose on the console in AWS Regions that support RA3 node types. For more information, see Creating a cluster (p. 112).

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. 123).

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. 123).

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. 72) 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. 66).

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 type</th>
<th>vCPU</th>
<th>RAM (GiB)</th>
<th>Default slices per node</th>
<th>Managed storage limit per node</th>
<th>Node range with create cluster</th>
<th>Total managed storage capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ra3.xlplus (single-node)</td>
<td>4</td>
<td>32</td>
<td>2</td>
<td>4 TB</td>
<td>1</td>
<td>4 TB³</td>
</tr>
<tr>
<td>ra3.xlplus (multi-node)</td>
<td>4</td>
<td>32</td>
<td>2</td>
<td>32 TB</td>
<td>2–16⁴</td>
<td>1024 TB⁴</td>
</tr>
<tr>
<td>ra3.4xlarge</td>
<td>12</td>
<td>96</td>
<td>4</td>
<td>128 TB</td>
<td>2–32⁵</td>
<td>8192 TB⁵</td>
</tr>
<tr>
<td>ra3.16xlarge</td>
<td>48</td>
<td>384</td>
<td>16</td>
<td>128 TB</td>
<td>2–128</td>
<td>16,384 TB</td>
</tr>
</tbody>
</table>

¹ The storage limit for Amazon Redshift managed storage. This is a hard limit.

² Total managed storage limit is the maximum number of nodes times the managed storage limit per node.

³ To resize a single-node cluster to multi-node, only classic resize is supported.

⁴ You can create a cluster with the ra3.xlplus (multi-node) node type that has up to 16 nodes. For multiple-node clusters, you can resize with elastic resize to a maximum of 32 nodes.

⁵ You can create a cluster with the ra3.4xlarge node type with up to 32 nodes. You can resize it with elastic resize to a maximum of 64 nodes.

### Dense storage node types

<table>
<thead>
<tr>
<th>Node type</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 type</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[^1]</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[^1]</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. 75).

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. 594).

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. 433).

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. 349).

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. 127).

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. 76) 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. 577).

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. 72).

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. 80).

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. 79).

Consider the following when using single-node RA3 node types.

• 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.

Networking features supported by RA3 nodes

When an RA3 cluster is created or updated, it uses a port within the following ranges: 5431-5455 or 8191-8215. In the console, you receive a warning if you create or modify an RA3 cluster and choose a port outside these ranges. RA3 nodes support a collection of networking features not available to DC2 or DS2 nodes. These include the following:

• **Custom domain name** – You can create a custom domain name, also known as a custom URL, for your Amazon Redshift cluster. It's an easy-to-read DNS record that routes SQL-client connections to your cluster endpoint. For more information, see [Using a custom domain name for client connections](p. 105).

• **Cluster relocation** – You can move a cluster to another Availability Zone (AZ) without any loss of data, when there is an interruption of service. This works well for a disaster-recovery scenario. You enable it on the console. For more information, see [Managing cluster relocation in Amazon Redshift](p. 96).

• **Multi-AZ** – For a Multi-AZ deployment, Amazon Redshift deploys equal compute resources in two Availability Zones that can be accessed through a single endpoint. In the event of a failure, the compute resources in the second Availability Zone are available. For more information, see [Configuring Multi-AZ deployment (preview)](p. 99).

• **Single-subnet RA3 clusters** – You can create an RA3 cluster with a single subnet, but it can't use disaster-recovery features. An exception occurs if you enable cluster relocation when the subnet doesn't have multiple availability zones (AZs).

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. 83)](p. 83). 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. 83)](p. 83).
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)
- Africa (Cape Town) Region (af-south-1)
- Asia Pacific (Hong Kong) Region (ap-east-1)
- Asia Pacific (Hyderabad) Region (ap-south-2)
- Asia Pacific (Jakarta) Region (ap-southeast-3)
- Asia Pacific (Melbourne) Region (ap-southeast-4)
- Asia Pacific (Mumbai) Region (ap-south-1)
- Asia Pacific (Osaka) Region (ap-northeast-3)
- 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)
- Canada (Central) Region (ca-central-1)
- China (Beijing) Region (cn-north-1)
- China (Ningxia) Region (cn-northwest-1)
- Europe (Frankfurt) Region (eu-central-1)
- Europe (Zurich) Region (eu-central-2)
- Europe (Ireland) Region (eu-west-1)
- Europe (London) Region (eu-west-2)
- Europe (Milan) Region (eu-south-1)
- Europe (Spain) Region (eu-south-2)
- Europe (Paris) Region (eu-west-3)
- Europe (Stockholm) Region (eu-north-1)
- Israel (Tel Aviv) Region (il-central-1)
- Middle East (Bahrain) Region (me-south-1)
- Middle East (UAE) Region (me-central-1)
- South America (São Paulo) Region (sa-east-1)
- AWS GovCloud (US-East) (us-gov-east-1)
- AWS GovCloud (US-West) (us-gov-west-1)

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. 414).

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. 93) 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. 83).

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^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>2</td>
<td>ra3.xlplus</td>
<td>Create a 2-node ra3.xlplus cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>3</td>
<td>ra3.xlplus</td>
<td>Create a 2-node ra3.xlplus cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>4</td>
<td>ra3.xlplus</td>
<td>Create a 3-node ra3.xlplus cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>5</td>
<td>ra3.xlplus</td>
<td>Create a 4-node ra3.xlplus cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>6</td>
<td>ra3.xlplus</td>
<td>Create a 4-node ra3.xlplus cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>7</td>
<td>ra3.xlplus</td>
<td>Create a 5-node ra3.xlplus cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>8</td>
<td>ra3.4xlarge</td>
<td>Create a 2-node ra3.4xlarge cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>9</td>
<td>ra3.4xlarge</td>
<td>Create a 3-node ra3.4xlarge cluster^1.</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>10</td>
<td>ra3.4xlarge</td>
<td>Create a 3-node ra3.4xlarge cluster^1.</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^1.</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>ds2.8xlarge</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>5–15</td>
<td>ra3.xlplus</td>
<td>Create 3 nodes of ra3.xlplus for every 8 nodes of dc2.large.</td>
</tr>
<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>

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

2Clusters 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.

### 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.
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 and backups (p. 410).
2. Create a VPC, or choose an existing VPC in your account. For more information, see Managing clusters in a VPC (p. 121).
3. Restore your snapshot to a new DC2 cluster in the VPC. For more information, see Restoring a cluster from a snapshot (p. 414).
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. 83).
- 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. 414).

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 a VPC (p. 121).
- 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. 83).
- 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. 437).
- 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.
- If you use restore to upgrade from dc1.8xlarge to dc2.8xlarge, then the snapshot must have been created at cluster version 1.0.10013 or later.
- If you use elastic resize to upgrade from DC1 to DC2, and change the number of nodes, then the cluster must be at cluster version 1.0.10013 or later.
- If a snapshot of dc1.8xlarge cluster to upgrade is from a cluster earlier than version 1.0.10013, then first restore the snapshot from the dc1.8xlarge cluster into a new dc1.8xlarge cluster with the same number of nodes. Then use one of the following methods to upgrade the new dc1.8xlarge:
  - Use a snapshot from the new restored cluster to upgrade to dc2.8xlarge.
  - Use elastic resize to upgrade the new restored cluster to dc2.8xlarge.

### 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 and backups (p. 410).
2. Create a VPC, or choose an existing VPC in your account. For more information, see Managing clusters in a VPC (p. 121).
3. Restore your snapshot to a new DS2 cluster in the VPC. For more information, see Restoring a cluster from a snapshot (p. 414).
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. 77)
- Deferring maintenance (p. 78)
- Choosing cluster maintenance tracks (p. 79)
- Managing cluster versions (p. 80)
- Rolling back the cluster version (p. 80)
- Determining the cluster maintenance version (p. 80)

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 (Hyderabad) Region: 16:30–00:30 UTC
Deferring maintenance

• Asia Pacific (Jakarta) Region: 15:00–23:00 UTC
• Asia Pacific (Melbourne) Region: 12:00–20:00 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
• Europe (Zurich) Region: 20:00–04:00 UTC
• Israel (Tel Aviv) Region: 20:00–04:00 UTC
• Europe (Spain) Region: 21:00–05:00 UTC
• Middle East (Bahrain) Region: 13:00–21:00 UTC
• Middle East (UAE) Region: 18:00–02:00 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. 112).

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.
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 switch a cluster from a preview track to 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. 131).

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/](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/](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 <clusternname>-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. 112) and Creating or editing a disk space alarm (p. 117).

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. 112).</td>
</tr>
<tr>
<td>Status</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>deleting</td>
<td>Amazon Redshift is deleting the cluster. For more information, see [Deleting a cluster](p. 115).</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. 115).</td>
</tr>
<tr>
<td>hardware-failure</td>
<td>The cluster suffered a hardware failure.</td>
</tr>
<tr>
<td></td>
<td>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 and backups](p. 410).</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. 448).</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. 121).</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. 389).</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 and backups](p. 410).</td>
</tr>
<tr>
<td>modifying</td>
<td>Amazon Redshift is applying changes to the cluster. For more information, see [Modifying a cluster](p. 114).</td>
</tr>
<tr>
<td>paused</td>
<td>The cluster is paused. For more information, see [Pausing and resuming clusters](p. 92).</td>
</tr>
<tr>
<td>rebooting</td>
<td>Amazon Redshift is rebooting the cluster. For more information, see [Rebooting a cluster](p. 116).</td>
</tr>
<tr>
<td>renaming</td>
<td>Amazon Redshift is applying a new name to the cluster. For more information, see [Renaming clusters](p. 93).</td>
</tr>
<tr>
<td>resizing</td>
<td>Amazon Redshift is resizing the cluster. For more information, see [Resizing a cluster](p. 116).</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. 449).</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. 116).</td>
</tr>
<tr>
<td>updating-hsm</td>
<td>Amazon Redshift is updating the HSM configuration.</td>
</tr>
</tbody>
</table>

82
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, you can resize your cluster to make the best use of Amazon Redshift's computing and storage options.

A resize operation comes in two types:

- **Elastic resize** – You can add nodes to or remove nodes from your cluster. You can also change the node type, such as from DS2 nodes to RA3 nodes. Elastic resize is a fast operation, typically completing in minutes. For this reason, we recommend it as a first option. When you perform an elastic resize, it redistributes data slices, which are partitions that are allocated memory and disk space in each node. Elastic resize is appropriate when you:
  - Add or reduce nodes in an existing cluster, but you don't change the node type – This is commonly called an in-place resize. When you perform this type of resize, some running queries complete successfully, but others can be dropped as part of the operation. An elastic resize completes within a few minutes.
  - Change the node type for a cluster – When you change the node type, a snapshot is created and data is redistributed from the source cluster to a cluster comprised of the new node type. On completion, running queries are dropped. Like the in-place resize, it completes quickly.
  - Classic resize – You can change the node type, number of nodes, or both, in a similar manner to elastic resize. Classic resize takes more time to complete, but it can be useful in cases where the change in node count or the node type to migrate to doesn't fall within the bounds for elastic resize. This can apply, for instance, when the change in node count is really large.

Topics

- Elastic resize (p. 83)
- Classic resize (p. 85)

Elastic resize

An elastic resize operation, when you add or remove nodes of the same type, has the following stages:

1. Elastic resize takes a cluster snapshot. This snapshot always includes no-backup tables (p. 414) for nodes where it's applicable. (Some node types, like RA3, don't have no-backup tables.) If your cluster doesn't have a recent snapshot, because you disabled automated snapshots, the backup operation can take longer. (To minimize the time before the resize operation begins, we recommend that you enable automated snapshots or create a manual snapshot before starting the resize.) When you start an elastic resize and a snapshot operation is in progress, the resize can fail if the snapshot operation doesn't complete within a few minutes. For more information, see Amazon Redshift snapshots and backups (p. 410).

2. The operation migrates cluster metadata. The cluster is unavailable for a few minutes. The majority of queries are temporarily paused and connections are held open. It is possible, however, for some queries to be dropped. This stage is short.

3. Session connections are reinstated and queries resume.

4. Elastic resize redistributes data to node slices, in the background. The cluster is available for read and write operations, but some queries can take longer to run.
5. After the operation completes, Amazon Redshift sends an event notification.

When you use elastic resize to change the node type, it works similarly to when you add or subtract nodes of the same type. First, a snapshot is created. A new target cluster is provisioned with the latest data from the snapshot, and data is transferred to the new cluster in the background. During this period, data is read only. When the resize nears completion, Amazon Redshift updates the endpoint to point to the new cluster and all connections to the source cluster are dropped.

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

Elastic resize doesn't sort tables or reclaims disk space, so it isn't a substitute for a vacuum operation. For more information, see [Vacuuming tables](#).

Elastic resize has the following constraints:

- **Elastic resize and data sharing clusters** - When you add or subtract nodes on a cluster that's a producer for data sharing, you can't connect to it from consumers while Amazon Redshift migrates cluster metadata. Similarly, if you perform an elastic resize and choose a new node type, data sharing is unavailable while connections are dropped and transferred to the new target cluster. In both types of elastic resize, the producer is unavailable for several minutes.
- **Single-node clusters** - You can't use elastic resize to resize from or to a single-node cluster.
- **Data transfer from a shared snapshot** - 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.
- **Platform restriction** - 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](#).
- **Storage considerations** - Make sure that your new node configuration has enough storage for existing data. You may have to add additional nodes or change configuration.
- **Source vs target cluster size** - The number of nodes and node type that it's possible to resize to with elastic resize is determined by the number of nodes in the source cluster and the node type chosen for 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](#).

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.

- **Ceiling on additional nodes** - 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 five-node cluster, or add more nodes until you reach eight.

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. (Keep in mind that the ability to grow the cluster by 4x depends on the source cluster size.) 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 of the number (from 16 to 8 nodes, for example)</td>
</tr>
<tr>
<td>dc2.large</td>
<td>2x</td>
<td>To one half of the number</td>
</tr>
<tr>
<td>ds2.8xlarge</td>
<td>2x</td>
<td>To one half of the number</td>
</tr>
<tr>
<td>ds2.xlarge</td>
<td>2x</td>
<td>To one half of the number</td>
</tr>
</tbody>
</table>

**Note**

*Choosing legacy node types when you resize an RA3 cluster* – If you attempt to resize from a cluster with RA3 nodes to another node type, such as DC2 or DS2, a validation warning message appears in the console, and the resize operation won't complete. This occurs because resize to legacy node types isn't supported. This is to prevent a customer from resizing to a node type that's deprecated or soon to be deprecated. This applies for both elastic resize and classic resize.

**Classic resize**

Classic resize handles use cases where the change in cluster size or node type isn't supported by elastic resize. When you perform a classic resize, Amazon Redshift creates a target cluster and migrates your data and metadata to it from the source cluster.

**Classic resize to RA3 can provide better availability**

Classic resize has been enhanced when the target node type is RA3. It does this by using a backup and restore operation between the source and target cluster. When the resize begins, the source cluster restarts and is unavailable for a few minutes. After that, the cluster is available for read and write operations while the resize continues in the background.

**Checking your cluster**

To ensure you have the best performance and results when you perform a classic resize to an RA3 cluster, complete this checklist.
1. The size of the data must be below 400TB. To validate the size of your data, create a snapshot and check its size. You can also run the following query to check the size:

```sql
SELECT
    sum(case when lower(diststyle) like ('%key%') then size else 0 end) distkey_blocks,
    sum(size) as total_blocks,
    ((distkey_blocks/(total_blocks*1.00)))*100 as Blocks_need_redist
FROM svv_table_info;
```

The `svv_table_info` table is visible only to superusers.

2. The node type of the target cluster must be RA3, and the target cluster must have at least two nodes. Classic resize to a single-node cluster takes much longer, even if the target node type is RA3.

3. Before you initiate a classic resize, make sure you have a manual snapshot that is no more than 10 hours old. If not, take a snapshot.

4. The snapshot used to perform the classic resize can't be used for a table restore or other purpose.

5. The cluster must be in a VPC.

Sorting and distribution operations that result from classic resize to RA3

During classic resize to RA3, Distribution Key tables migrated as Distribution Even are converted back to their original distribution style, using operations in the background. The duration of this is dependent on the size of the data and how busy your cluster is. Query workloads are given higher priority to run over data migration. For more information, see Distribution styles. Both reads and writes to the database work during this migration process, but there can be degradation in query performance. However, concurrency scaling can boost performance by adding resources for query workloads. You can see the progress of data migration by using results from the `svl_restore_alter_table_progress` view. More information about monitoring follows.

After the cluster is fully resized, the following sort behavior occurs:

- If the resize results in the cluster having more slices, KEY distribution tables become partially unsorted, but EVEN tables remain sorted. Additionally, the information about how much data is sorted may not be up to date, directly following the resize. After key recovery, automatic vacuum sorts the table over time.
- If the resize results in the cluster having fewer slices, both KEY distribution and EVEN distribution tables become partially unsorted. Automatic vacuum sorts the table over time.

For more information about automatic table vacuum, see Vacuuming tables. For more information about slices in compute nodes, see Data warehouse system architecture.

Classic resize steps when the target cluster is RA3

Classic resize consists of the following steps, when the target cluster type is RA3 and you've met the prerequisites detailed in the previous section.

1. Migration initiates from the source cluster to the target cluster. When the new, target cluster is provisioned, Amazon Redshift sends an event notification that the resize has started. It restarts your existing cluster, which closes all connections. If your existing cluster is a datasharing producer cluster, connections with consumer clusters are also closed. The restart takes a few minutes.

   Note that any database relation, such as a table or materialized view, created with BACKUP NO is not retained during the classic resize. For more information, see CREATE MATERIALIZED VIEW.

2. After the restart, the database is available for reads and writes. Additionally, data sharing resumes, which takes an additional few minutes.
3. Data is migrated to the target cluster. When the target node type is RA3, reads and writes are available during data migration.

4. When the resize process nears completion, Amazon Redshift updates the endpoint to the target cluster, and all connections to the source cluster are dropped. The target cluster becomes the producer for data sharing.

5. The resize completes. Amazon Redshift sends an event notification.

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.

**Note**

**Choosing legacy node types when you resize an RA3 cluster** – If you attempt to resize from a cluster with RA3 nodes to another node type, such as DC2 or DS2, a validation warning message appears in the console, and the resize operation won't complete. This occurs because resize to legacy node types isn't supported. This is to prevent a customer from resizing to a node type that's deprecated or soon to be deprecated. This applies for both elastic resize and classic resize.

**Monitoring a classic resize when the target cluster is RA3**

To monitor a classic resize of a provisioned cluster in progress, including KEY distribution, use `svl_restore_alter_table_progress`. It shows the percentage completed for the table being converted. You must be a super user to access the data. The following shows a sample query:

```sql
select * from svl_restore_alter_table_progress;
```

The result is the following:

<table>
<thead>
<tr>
<th>tbl</th>
<th>progress</th>
<th>message</th>
</tr>
</thead>
<tbody>
<tr>
<td>105614</td>
<td>ABORTED</td>
<td>Abort:Table no longer contains the prior dist key column.</td>
</tr>
<tr>
<td>105610</td>
<td>ABORTED</td>
<td>Abort:Table no longer contains the prior dist key column.</td>
</tr>
<tr>
<td>105594</td>
<td>0.00%</td>
<td>Table waiting for alter diststyle conversion.</td>
</tr>
<tr>
<td>105602</td>
<td>ABORTED</td>
<td>Abort:Table no longer contains the prior dist key column.</td>
</tr>
<tr>
<td>105606</td>
<td>ABORTED</td>
<td>Abort:Table no longer contains the prior dist key column.</td>
</tr>
<tr>
<td>105598</td>
<td>100.00%</td>
<td>Restored to distkey successfully.</td>
</tr>
</tbody>
</table>

Drop tables that you don't need when you perform a classic resize. When you do this, existing tables can be distributed more quickly.

**Classic resize steps when the target cluster isn't RA3**

Classic resize consists of the following, when the target node type is anything other than RA3, like DS2, for instance.

1. Migration initiates from the source cluster to the target cluster. When the new, target cluster is provisioned, Amazon Redshift sends an event notification that the resize has started. It restarts your existing cluster, which closes all connections. If your existing cluster is a datasharing producer cluster, connections with consumer clusters are also closed. The restart takes a few minutes.

   Note that any database relation, such as a table or materialized view, created with `BACKUP NO` is not retained during the classic resize. For more information, see [CREATE MATERIALIZED VIEW](#).

2. Following the restart, the database is available as read only. Data sharing resumes, which takes an additional few minutes.

3. Data is migrated to the target cluster. The database remains read only.

4. When the resize process nears completion, Amazon Redshift updates the endpoint to the target cluster, and all connections to the source cluster are dropped. The target cluster becomes the producer for data sharing.
5. The resize completes. Amazon Redshift sends an event notification.

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.

**Note**
It can take days or possibly weeks to resize a cluster with a large amount of data when the target cluster isn't RA3, or it doesn't meet the prerequisites for an RA3 target cluster detailed in the previous section.
Also note that used storage capacity for the cluster can go up after a classic resize. This is normal system behavior when the cluster has additional data slices that result from the classic resize. This use of additional capacity can occur even when the number of nodes in the cluster stays the same.

### Elastic resize vs classic resize

The following table compares behavior between the two resize types.

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Elastic resize</th>
<th>Classic resize</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System data retention</strong></td>
<td>Elastic resize retains system log data.</td>
<td>Classic resize doesn't retain system tables and data.</td>
<td>If you have audit logging enabled in your source cluster, you can continue to access the logs in Amazon S3 or in CloudWatch, following a resize. You can keep or delete these logs as your data policies specify.</td>
</tr>
<tr>
<td><strong>Changing node types</strong></td>
<td>Elastic resize, when the node type doesn't</td>
<td>Classic Resize: A new cluster is created. Queries</td>
<td></td>
</tr>
</tbody>
</table>
### Resizing clusters

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Elastic resize</th>
<th>Classic resize</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>change: In-place resize, and most queries are held.</td>
<td>are dropped during the resize process.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Elastic resize, with a new node type selected: A new cluster is created. Queries are dropped as the resize process completes.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Session and query retention | Elastic resize retains sessions and queries when the node type is the same in the source cluster and target. If you choose a new node type, queries are dropped. | Classic resize doesn't retain sessions and queries. Queries are dropped.         | When queries are dropped, you can expect some performance degradation. It's best to perform a resize operation during a period of light use. |
### Resizing clusters

#### Behavior

<table>
<thead>
<tr>
<th>Canceling a resize operation</th>
<th>Elastic resize</th>
<th>Classic resize</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can't cancel an elastic resize.</td>
<td>You can cancel a classic resize operation before it completes by choosing <strong>Cancel resize</strong> from the cluster details in the Amazon Redshift console.</td>
<td>The amount of time it takes to cancel a resize depends on the stage of the resize operation when you cancel. When you do this, the cluster isn't available until the cancel operation completes. If the resize operation is in the final stage, you can't cancel. For classic resize to an RA3 cluster, you can't cancel.</td>
<td></td>
</tr>
</tbody>
</table>

#### Scheduling a resize

You can schedule resize operations for your cluster to scale up to anticipate high use or to scale down for cost savings. Scheduling works for both elastic resize and classic resize. You can set up a schedule on the Amazon Redshift console. For more information, see [Resizing a cluster (p. 116)](#), under **Managing**
clusters using the console. You can also use 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.

Snapshot, restore, and resize

Elastic resize (p. 83) 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 require 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.
Getting the leader node IP address

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.

```
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 turned off.

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.
- After a cluster is paused, traces and logs might not be available for troubleshooting problems that occurred before the pause.
- No-backup tables on the cluster are not restored on resume. For more information about no-backup tables, see Excluding tables from snapshots (p. 414).

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 may rename a cluster if you want to change the cluster your applications connect to 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 must 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. 114).

## 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. 115).

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](#).
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 allow 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 turned on, 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 turn on relocation for your cluster, 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.

Cluster relocation is disabled by default on all RA3 clusters. Amazon Redshift assigns 5439 as the default port while creating a provisioned cluster. You can change to another port from the port range of 5431-5455 or 8191-8215. To change the default port for a provisioned cluster, use the Amazon Redshift console, AWS CLI, or Amazon Redshift API. To change the default port for a serverless workgroup, use the AWS CLI or the Amazon Redshift Serverless API.
If you turn on 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.

**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.
- You can't perform a relocation across AWS Regions.
- Amazon Redshift relocation defaults to port number 5439. You can also change to another port from the port range of 5431-5455 or 8191-8215.
- Relocation is available in the following Regions:
  - US East (Ohio) Region (us-east-2)
  - US East (N. Virginia) Region (us-east-1)
  - US West (N. California) Region (us-west-1)
  - US West (Oregon) Region (us-west-2)
  - Asia Pacific (Melbourne) Region (ap-southeast-4)
  - 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)
  - Canada (Central) Region (ca-central-1)
  - Europe (Frankfurt) Region (eu-central-1)
  - Europe (Ireland) Region (eu-west-1)
  - Europe (London) Region (eu-west-2)
  - Europe (Paris) Region (eu-west-3)
  - Europe (Stockholm) Region (eu-north-1)
  - Israel (Tel Aviv) Region (il-central-1)
  - South America (São Paulo) Region (sa-east-1)

**Turning on cluster relocation**

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

To turn on 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.

Managing relocation using the console

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

Turning on relocation when creating a new cluster

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

To turn on 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 Enabled. Relocation is turned off by default.
5. 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 Enabled. Relocation is turned off by default.
6. 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 turned on.

```bash
aws redshift create-cluster --cluster-identifier mycluster --number-of-nodes 2 --master-username enter a username --master-user-password enter a password --node-type ra3.4xlarge --port 5439 --availability-zone-relocation
```

If your current cluster is using a different port, you must modify it to use from the port range of 5431-5455 or 8191-8215 before modifying it to turn on relocation. The default is 5439. The following example command modifies the port in case your cluster doesn't use one from the given range.

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

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

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

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

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

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

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

Configuring Multi-AZ deployment (preview)

This is prerelease documentation for the Multi Availability Zones feature for Amazon Redshift, which is in preview release. The documentation and the feature are both subject to change. We recommend that you use this feature only with test clusters, and not in production environments. Public preview will end on September 30, 2023. Preview clusters and preview serverless workgroups and namespaces will be removed automatically two weeks after the end of the preview. For inquiries, email redshift-maz@amazon.com. For preview terms and conditions, see Betas and Previews in AWS Service Terms.

By using multiple Availability Zones (Multi-AZ) in Amazon Redshift, your Amazon Redshift data warehouse can continue operating in failure scenarios where an unexpected event happens in an Availability Zone.
For a Multi-AZ deployment, Amazon Redshift deploys equal compute resources in two Availability Zones that can be accessed through a single endpoint. In the event of an entire Availability Zone failure, the remaining compute resources in the second Availability Zone will be available to continue processing workloads. Amazon Redshift Multi-AZ also warrants that there is no data loss in the event of a failure. Amazon Redshift charges the same hourly compute rates for RA3 when running a Multi-AZ cluster.

Overview

When setting up a Multi-AZ deployment, specify the number of compute nodes to provision in each Availability Zone. The nodes are used in both Availability Zones in an active-active fashion, for both read and write operations. Amazon Redshift will process an individual query using the compute resources residing in only one Availability Zone. However, it will automatically distribute processing of multiple simultaneous queries to both Availability Zones to increase performance of high concurrency workloads.

When migrating to a Multi-AZ deployment from an existing Single-AZ cluster, maintaining performance of a single query might require the same number of nodes used in the current Single-AZ cluster to be provisioned in both Availability Zones. This results in doubling the number of cluster nodes needed when migrating to Multi-AZ, to facilitate that single query performance is maintained.

In the event of a failure in an Availability Zone, Amazon Redshift continues operating by using the resources in the remaining Availability Zone automatically. However, user connections might be lost and must be re-established. In addition, queries that were running in the failed Availability Zone will be stopped. However, you can reconnect to your cluster and reschedule queries immediately. Amazon Redshift will process the rescheduled queries in the remaining Availability Zone. In the event that a query fails or a connection shuts down, Amazon Redshift re-tries the failed query or re-establishes connection immediately. Queries issued at or after a failure occurs might experience runtime delays while the Multi-AZ data warehouse is recovering.

Limitations

The following are limitations when working with Amazon Redshift Multi-AZ:

- You can't modify an existing Single-AZ deployment to convert it into a Multi-AZ deployment, or vice versa (preview).
- You can't restore an existing Multi-AZ deployment into a Single-AZ deployment on a Current or Trailing maintenance track (preview).
- You can't resize a cluster with Multi-AZ deployment (preview).
- You can't perform table-level restore from a snapshot taken off a cluster with Multi-AZ deployment (preview).
- You can't create a serverless Multi-AZ deployment or convert an existing serverless namespace into a Multi-AZ deployment (preview).
- Amazon Redshift doesn't display node-level metrics for compute resources in the secondary Availability Zone within a Multi-AZ deployment in the Amazon Redshift console (preview).
- You can't use Amazon Redshift API or AWS Command Line Interface to manage your Multi-AZ deployment (preview).
- You can't preview automatic optimization feature on a Multi-AZ deployment (preview).
- You will lose any IAM roles added after the Multi-AZ deployment was created whenever the cluster detects a failure and automatically recovers in the secondary AZ. Any IAM roles or permissions added before the creation of the Multi-AZ deployment will still persist (preview).
- You can't access your Multi-AZ cluster through a public endpoint.
- You can't create a single node Multi-AZ deployment for any of the RA3 instance types. Choose 2 or more nodes per Availability Zone while creating a Multi-AZ deployment.
- You can't create an unencrypted Multi-AZ cluster or restore an unencrypted Single-AZ cluster to a Multi-AZ cluster.
Amazon Redshift Management Guide
Managing Multi-AZ deployment

- Amazon Redshift requires a subnet configuration that can support three Availability Zones. In other words, the configured subnet group requires three subnets.
- You can't relocate a Multi-AZ deployment to another Availability Zone. Relocation will be turned off when using Multi-AZ deployment.
- You can't pause or resume a Multi-AZ deployment, as it is designed to provide the highest level of availability.
- Amazon Redshift Multi-AZ deployment is available in the following AWS Regions:
  - US East (Ohio) (us-east-2)
  - US East (N. Virginia) (us-east-1)
  - US West (Oregon) (us-west-2)
  - Asia Pacific (Tokyo) (ap-northeast-1)
  - Europe (Ireland) (eu-west-1)
  - Europe (Stockholm) (eu-north-1)

Topics
- Managing Multi-AZ deployment (p. 101)
- Managing Multi-AZ using the console (p. 101)

Managing Multi-AZ deployment

Amazon Redshift Multi-AZ supports two Availability Zones at a time. Amazon Redshift selects the Availability Zones automatically based on the network connectivity configured. Also, you can specify which Availability Zones to use explicitly. To convert an existing single Availability Zone data warehouse to a Multi-AZ deployment, you can restore from a snapshot to configure it into a Multi-AZ data warehouse.

Using the Amazon Redshift console, you can easily create new Multi-AZ deployments. To create a new Multi-AZ deployment using the Amazon Redshift console, select the Multi-AZ option when creating the data warehouse. Specify the number of compute nodes required in a single Availability Zone, and Amazon Redshift will deploy the number of nodes in each of two Availability Zones. All nodes can perform read and write workload processing during normal operation.

Multi-AZ deployment only supports RA3 node types that use Amazon Redshift Managed Storage (RMS). Amazon Redshift stores data in RMS, which uses Amazon S3 and is accessible in all Availability Zones in an AWS Region, without having to replicate the data at the Amazon Redshift level. RMS isn't supported on DC2 clusters.

Managing Multi-AZ using the console

You can manage the settings for Multi-AZ using the Amazon Redshift console.

Setting up Multi-AZ when creating a new cluster

Use the following procedure to set up Multi-AZ deployment when creating 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 Provisioned clusters dashboard, and 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. A banner displays on the Clusters list page that introduces preview mode. Choose the button Create preview cluster to open the create cluster page.
4. Enter properties for your cluster. Choose the **Preview track** that contains the features you want to test. Your cluster must be created with the preview track named **preview_2022**. We recommend entering a name for the cluster that indicates that it is on a preview track. Choose options for your cluster, including options labeled as -preview, for the features you want to test. For general information about creating clusters, see **Creating a cluster** (p. 112).

5. Choose one of the RA3 node types from the **Node type** drop-down list. The Multi-AZ deployment option becomes available only when you chose an RA3 node type.

6. Under **Multi-AZ deployment**, choose **Yes**.

7. Under **Number of nodes per AZ**, enter the number of nodes that you need for your cluster.

8. Optionally, do one of the following to load sample data or bring your own data:
   - In **Sample data**, choose **Load sample data** to load the sample dataset into your Amazon Redshift cluster. Amazon Redshift loads the sample dataset Tickit into the default dev database and public schema. Amazon Redshift automatically loads the sample dataset into your Amazon Redshift cluster. You can start using the query editor v2 to query data.
   - To bring your own data to your Amazon Redshift cluster, follow the steps in **Bringing your own data to Amazon Redshift**.

9. Scroll down to **Additional configurations**, expand **Network and security**, and make sure that you either accept the default **Cluster subnet group** or choose another one. If you choose another cluster subnet group, make sure that there are 3 Availability Zones in the subnet group you selected.

10. Under **Additional configurations**, expand **Database configurations**.

11. Under **Database encryption**, to use a custom KMS key other than the default AWS Key Management Service key, click **Customize encryption settings. This option is deselected by default**.

12. Under **Choose an KMS key**, you can either choose an AWS Key Management Service key or enter an ARN. Or, you can click **Create an AWS Key Management Service key** in the AWS Key Management Service console. For more information about creating KMS keys, see **Creating Keys** in the **AWS Key Management Service Developer Guide**.

13. Click **Create cluster**. When the cluster creation succeeds, you can view the details in the cluster details page. You can use your SQL client to load and query data.

---

### Setting up Multi-AZ for a cluster restored from a snapshot

Use the following procedure to set up Multi-AZ for a cluster restored from a snapshot.

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, Snapshots**, then choose the snapshot to use.

3. Choose **Restore snapshot, Restore to a provisioned cluster**.

4. Enter properties for your cluster. Choose **Preview**. Then, choose the **Preview track** that contains the features you want to test. We recommend entering a name for the cluster that indicates that it is on a preview track. Choose options for your cluster, including options labeled as -preview, for the features you want to test. For general information about creating clusters, see **Creating a cluster** (p. 112).

5. Choose one of the RA3 node types from the **Node type** drop-down list. The Multi-AZ deployment option becomes available only when you chose an RA3 node type.

6. Make sure that you choose one of the RA3 node types from the **Node type** drop-down list. The Multi-AZ deployment option becomes available only when you chose an RA3 node type.

7. Under **Multi-AZ deployment**, choose **Yes**.

8. Under **Number of nodes per AZ**, enter the number of nodes that you need for your cluster.

9. Optionally, do one of the following to load sample data or bring your own data:
• In **Sample data**, choose **Load sample data** to load the sample dataset into your Amazon Redshift cluster. Amazon Redshift loads the sample dataset Tickit into the default dev database and public schema. Amazon Redshift automatically loads the sample dataset into your Amazon Redshift cluster. You can start using the query editor v2 to query data.

• To bring your own data to your Amazon Redshift cluster, follow the steps in [Bringing your own data to Amazon Redshift](#).

10. Scroll down to **Additional configurations**, expand **Network and security**, and make sure that you either accept the default **Cluster subnet group** or choose another one. If you choose another cluster subnet group, make sure that there are 3 Availability Zones in the subnet group you selected.

11. Under **Additional configurations**, expand **Database configurations**.

12. Under **Database encryption**, to use a custom KMS key other than the default AWS Key Management Service key, click **Customize encryption settings**. This option is deselected by default.

13. Under **Choose an KMS key**, you can either choose an AWS Key Management Service key or enter an ARN. Or, you can click **Create an AWS Key Management Service key** in the AWS Key Management Service console. For more information about creating KMS keys, see [Creating Keys](#) in the **AWS Key Management Service Developer Guide**.

14. Click **Restore cluster from snapshot**. When the cluster restoration succeeds, you can view the details in the cluster details page.

### Testing Multi-AZ fault tolerance (optional)

Use the following procedure to test Multi-AZ fault tolerance of your Amazon Redshift data warehouse using the **Inject failure** option.

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. Do one of the following:

   • On the navigation menu, choose **Clusters**. Under **Clusters**, choose a cluster. The cluster details page appears.

   • From the cluster dashboard, choose a cluster.

3. From **Actions**, choose **Inject failure (Public preview)**.

4. When prompted, click **Confirm**.

Amazon Redshift injects a failure into your Multi-AZ cluster. It will cause compute nodes in one Availability Zone to become unavailable. Amazon Redshift detects this event and starts an automatic recovery. When the cluster recovery successfully completes, Multi-AZ becomes available. Your Multi-AZ cluster will also automatically provision new compute nodes in another Availability Zone as soon as it is available.

During this process, the cluster status on the console shows as modifying for the entire time, as the cluster automatically recovers and reconfigures back to the Multi-AZ deployment setup. The cluster can accept new connections immediately. Existing connections and inflight queries might be dropped. You can retry them immediately.

### Viewing queries and loads for Multi-AZ clusters

The information shown on the Queries and loads page is populated with information from Amazon Redshift system tables (SYS_* views). This information lets you display additional information about your queries and offers rolling 7 days of retention. You can view information on queries that ran in the past 7 days irrespective of the type, size, and status (pause or resume) of your cluster. Query diagnostics
become faster, letting you filter data by database, username, or SQL statement type. To see these additional filters and information on all queries that ran, note the following prerequisites:

- You must connect to a database by choosing Connect to database.
- Your database user must have the sys:operator or sys:monitor roles and permissions to perform query monitoring. For information about system roles, see Amazon Redshift system-defined roles in the Amazon Redshift Database Developer Guide.

You will see these additional filters and query information once you connect to a database.

**To display query performance data from Queries and loads**

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.
3. You might have to connect to a database to see additional filter. If required, click Connect to database and follow the prompts to connect to a database.

   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.

**To display query performance data from Query monitoring**

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. Under Clusters, select a cluster.
3. Choose Query monitoring.
4. Depending on the configuration or version of your cluster, you might have to connect to a database to see additional filters. If required, click Connect to database and follow the prompts to connect to a database.

**Monitoring a query in a Multi-AZ deployment**

A Multi-AZ deployment uses compute resources that are deployed in both Availability Zones and can continue operating in the event that the resources in a given Availability Zone aren't available. All the compute resources will be used at all times. This allows full operation across two Availability Zones in an active-active fashion for both read and write operations.

You can query SYS_ views in pg_catalog schema to monitor query runtime in a Multi-AZ deployment. The SYS_ views display query runtime activities or statistics from primary and secondary clusters. Following are the system tables in SYS_ view list:

- SYS_QUERY_HISTORY
- SYS_QUERY_DETAIL
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_CONNECTION_LOG
- SYS_DATASHARE_CHANGE_LOG
- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_LOAD_HISTORY
Follow these steps to monitor query runtime for each Availability Zone within the Multi-AZ deployment:

1. Navigate to the Amazon Redshift console and connect to the database in your Multi-AZ deployment and run queries through the query editor.
2. Run any sample query on the Multi-AZ Amazon Redshift deployment.
3. For a Multi-AZ deployment, you can identify a query and the Availability Zone where it is run by using the compute_type column in the SYS_QUERY_HISTORY table. primary stands for queries run on the primary cluster in the Multi-AZ deployment, and secondary stands for queries run on the secondary cluster in the Multi-AZ deployment.

The following query uses compute_type column to monitor a query.

```
select (compute_type) as compute_type, left(query_text, 50) query_text from sys_query_history order by start_time desc;
```

<table>
<thead>
<tr>
<th>compute_type</th>
<th>query_text</th>
</tr>
</thead>
<tbody>
<tr>
<td>secondary</td>
<td>select count(*) from t1;</td>
</tr>
</tbody>
</table>

**Ending a query for Multi-AZ clusters**

**To end a query**

You can also use the Queries page to end a query that is currently in progress.

Your database user must have the sys:operator role and permissions to end a running query. For information about system roles, see Amazon Redshift system-defined roles in the Amazon Redshift Database Developer Guide.

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.
3. Choose the running query that you want to end in the list, and then choose Terminate query.

**Using a custom domain name for client connections**

You can create a custom domain name, also known as a custom URL, for your Amazon Redshift cluster. It's an easy-to-read DNS record that routes SQL client connections to your cluster endpoint. You can configure it for an existing cluster at any time. It provides several benefits:

- The custom domain name is simpler and easier to remember than the default URL. It's easier to recall and use.
- You can route traffic to a new cluster, for example in a fail-over case, or rename a cluster and quickly re-route traffic to the new name. This makes it so clients don't have to make a configuration change when they reconnect. Connections can be re-routed centrally, with minimal disruption.
- To avoid exposing private information in a connection URL, like a server name, you can hide it in a friendly custom URL.
When you set up a custom domain name, using a CNAME, there isn't any additional charge from Amazon Redshift. You may be billed from your DNS provider for a domain name, if you create a new one, but this cost is typically small. For more information, see Setting up a custom domain name.

Security for a custom domain name

Amazon Redshift requires a validated Secure Sockets Layer (SSL) certificate for a custom endpoint to keep communication secure and to verify ownership of the domain name. You can use the your AWS Certificate Manager account with an AWS KMS key for secure certificate management. Security validation includes full host-name verification (sslmode=verify-full).

Renewing a certificate

Certificate renewals are managed by Amazon Redshift only when you choose DNS validation, rather than email validation. If you use email validation, you can use the certificate, but you must perform renewal yourself, prior to its expiration. We recommend that you choose DNS validation for your certificate. You can monitor expiration dates of imported certificates in AWS Certificate Manager.

Setting up a custom domain name

Setting up the custom domain name consists of a several tasks. These include registering your domain name and creating a certificate. After you perform these tasks, you configure the custom domain name in the Amazon Redshift console, or with AWS CLI commands. The following sections detail the steps.

Register a domain name and select a certificate

You must have a registered internet domain name to configure a custom domain name in Amazon Redshift. You can register an internet domain using Route 53, or using a third-party domain registrar. These tasks take place outside of the Amazon Redshift console. A registered domain is a prerequisite for completing the remaining procedures to create a custom domain for Amazon Redshift.

Note
Prior to performing the steps to configure the custom domain name, the cluster must be relocation enabled. For more information, see Managing cluster relocation in Amazon Redshift (p. 96).

The custom domain name typically includes the root domain and a subdomain, like mycluster.mycompany.com. To configure it, perform the following steps:

Create a DNS CNAME entry for your custom domain name

1. Register a root domain, for example mycompany.com. Optionally, you can use an existing domain. Your custom name can be limited by restrictions on particular characters, or other naming validation. For more information about registering a domain with Route 53, see Registering a new domain.

2. Add a DNS record that points your custom CNAME to the Amazon Redshift endpoint. You can find the cluster endpoint string in the cluster properties, in the Redshift console. Copy the JDBC URL in the cluster properties under General information. The URL looks like the following: redshift-cluster-sample.abc123456.us-east-1.redshift.amazonaws.com. You must remove the JDBC prefix.

   Note
   DNS records are subject to availability, because each name must be unique and available for use within your organization. Each must also be validated by your DNS provider.

After you register your domain and create the CNAME record, you select a new or existing certificate. You perform this step using AWS Certificate Manager:
Request a certificate from ACM for a domain name

1. Sign in to the AWS Management Console and open the ACM console at https://console.aws.amazon.com/acm/.
2. Choose Request a certificate.
3. Enter your custom domain name in the Domain name field.
   
   Note
   You can specify many prefixes, in addition to the certificate domain, in order to use a single certificate for multiple custom-domain records. To illustrate, you can use additional records like one.mycompany.com and two.mycompany.com, with the same certificate.

4. Choose Review and request.
5. Choose Confirm and request.
6. For a valid request, a registered owner of the internet domain must consent to the request before ACM issues the certificate. Make sure the status appears as Issued in the ACM console, when you’re finished with the steps.

Create the custom domain for your cluster

You can use the Amazon Redshift console to create your custom domain URL. If you haven't configured it, the Custom domain name property appears as a dash (–) under General information. After you create your CNAME record and the certificate, you assign the Amazon Redshift custom domain name to the cluster.

In order to create or modify a custom domain name, the following IAM permissions are required:

- redshift:CreateCustomDomainAssociation – You can restrict permission to a specific cluster by adding its ARN.
- acm:DescribeCertificate

As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift.

You assign the custom domain name by performing the following steps.

1. Choose the cluster in the Amazon Redshift console and then choose Create custom domain name under the Action menu. A dialogue appears.
2. Choose the DNS provider, such as Route 53. Then enter the CNAME record to use. It will appear like mycluster.mycompany.com.
3. Select the ARN from AWS Certificate Manager for the ACM Certificate. Confirm your changes.
4. Verify in the cluster properties that the Custom domain name and Custom domain certificate ARN are populated with your entries. The Custom domain certificate expiry date is also listed.

After the custom domain is configured, using sslmode=verify-full works only for the new, custom endpoint. It doesn't work for the default endpoint. But you can still connect to the default endpoint by using other ssl modes, such as sslmode=verify-ca.

Renaming a cluster that has a custom domain assigned, using the console

In order to rename a cluster that has a custom domain name, the acm:DescribeCertificate IAM permission is required.
1. Go to the Amazon Redshift console and choose the cluster whose name you want to change. Choose **Edit** to edit the cluster properties.

2. Edit the **Cluster identifier**. You can also change other properties for the cluster. Then choose **Save changes**.

3. After the cluster is renamed, you have to update the DNS record to change the CNAME entry for the custom domain to point to the updated Amazon Redshift endpoint.

### Changing the custom domain, the cluster name, or the certificate, using CLI commands

In order to create or modify a custom domain name, the following IAM permissions are required:

- `redshift:ModifyCustomDomainAssociation`
- `acm:DescribeCertificate`

As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see [Identity and access management in Amazon Redshift](https://docs.aws.amazon.com/redshift/latest/gsg/about-iam-permissions.html).

The following shows a sample command to change to a different custom domain name:

```bash
aws redshift modify-custom-domain-association --cluster-id redshift-cluster-sample --custom-domain-name custom domain name
```

Use the following command to change to a different custom domain that's associated with a different certificate. Note the ARN for the new certificate must be different than the existing ARN. Additionally, the new custom domain name must be associated with the new certificate.

```bash
aws redshift modify-custom-domain-association --cluster-id redshift-cluster-sample --custom-domain-name custom domain name --custom-domain-certificate-arn ARN
```

There is a maximum delay of 30 seconds after creating or updating a custom domain before you can connect to the cluster. Part of the delay occurs as the Amazon Redshift cluster updates its properties, and there is some additional delay as DNS is updated. For more information about the API and each property setting, see [ModifyCustomDomainAssociation](https://docs.aws.amazon.com/redshift/latest/gsg/modify-custom-domain-association.html).

### Deleting the custom domain, using a CLI command

To delete the custom domain name, the user must have permissions for the action `redshift:DeleteCustomDomainAssociation`.

The following sample shows how to delete the custom domain name.

```bash
aws redshift delete-custom-domain-association --cluster-id redshift-cluster-sample
```

For more information, see [DeleteCustomDomainAssociation](https://docs.aws.amazon.com/redshift/latest/gsg/delete-custom-domain-association.html).

### Connect to your cluster with a custom domain name, using a SQL client

In order to connect to a cluster with a custom domain name, the following IAM permissions are required:
As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift.

After you complete the steps to create your CNAME and assign it to your cluster in the Amazon Redshift console, you can provide the custom URL in the connection properties of your SQL client. Note that there can be a delay from DNS propagation immediately following the creation of a CNAME record.

1. Open a SQL client. For example, you can use SQL/Workbench J. Open the properties for a connection, and add the custom domain name for the connection string. For example, jdbc:redshift://mycluster.mycompany.com:5439/dev?sslmode=verify-full. In this example, dev specifies the default database.
2. Add the Username and Password for your database user.
3. Test the connection. Your ability to query database resources such as specific tables can vary, based on the permissions granted to the database user or granted to the Amazon Redshift database roles assigned.

Note that you might have to set your cluster to be publicly accessible to connect to it if it's in a VPC. You can change this setting in the network properties.

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. 96).
- Make sure that the cluster to access is available within the valid port ranges 5431-5455 and 8191-8215. The default is 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.
- When you create a Redshift-managed VPC endpoint, the VPC you choose must have a cluster subnet group. To create a cluster subnet group, see [Managing cluster subnet groups using the console](p. 127).

For information about quotas and naming constraints, see [Quotas and limits in Amazon Redshift](p. 643).

For information about pricing, see [AWS PrivateLink pricing](#).

**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/](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 security group rules in **Security group** define the ports, protocols, and sources for inbound traffic that you are authorizing for your endpoint. Depending on the port you selected when creating, modifying, or migrating the cluster, you allow access to the selected port via 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. 112)
- Modifying a cluster (p. 114)
- Deleting a cluster (p. 115)
- Rebooting a cluster (p. 116)
- Resizing a cluster (p. 116)
- Upgrading the release version of a cluster (p. 116)
- Getting information about cluster configuration (p. 117)
- Getting an overview of cluster status (p. 117)
- Creating a snapshot of a cluster (p. 117)
- Creating or editing a disk space alarm (p. 117)
- Working with cluster performance data (p. 118)

Creating a cluster

Before you create a cluster, read Overview of Amazon Redshift clusters (p. 64) and Clusters and nodes in Amazon Redshift (p. 65).
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.

The following step describes an Amazon Redshift console that is running in 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 Management Guide.

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.

Note
If your organization is eligible and your cluster is being created in an AWS Region where Amazon Redshift Serverless is unavailable, you might be able to create a cluster under the Amazon Redshift free trial program. Choose either Production or Free trial to answer the question What are you planning to use this cluster for? When you choose Free trial, you create a configuration with the dc2.large node type. For more information about choosing a free trial, see Amazon Redshift free trial. For a list of AWS Regions where Amazon Redshift Serverless is available, see the endpoints listed for the Redshift Serverless API in the Amazon Web Services General Reference.

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

Note
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. 444).

- **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. 446).

- **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. 448).

**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.

**Automated snapshot retention period**

You can choose the number of days to retain these snapshots within 35 days. If the node type is DC2 or DS2, you can choose zero (0) days to not create automated snapshots.

**Manual snapshot retention period**

You can choose the number of days or Indefinitely to retain these 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**
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 and backups** (p. 410).

For more information about cluster properties, see **Additional configurations** (p. 113).

**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/](https://console.aws.amazon.com/redshift/).
2. On the navigation menu, choose **Clusters**.
3. Choose the cluster to modify.
4. Choose **Edit**. The **Edit cluster** page appears.
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. 94).

**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/](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. 83).

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`
- `DescribeNodeConfigurationOptions`
- `DescribeResize`
- `DescribeOrderableClusterOptions`
- `ModifyCluster`
- `PauseCluster`
- `RebootCluster`
- `ResizeCluster`
- `ResumeCluster`
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. 584). You need to update the code and specify a cluster identifier.

Example

```java
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;
    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 {link
    com.amazonaws.auth.DefaultAWSCredentialsProviderChain} client = AmazonRedshiftClientBuilder.defaultClient();

    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) {
    if (result == null) {
        System.out.println("Describe clusters result is null.");
        return;
    }

    System.out.println("Cluster property:");
    System.out.format("Preferred Maintenance Window: %s\n", result.getClusters().get(0).getPreferredMaintenanceWindow());
}

private static void waitForClusterReady() throws InterruptedException {
    Boolean clusterReady = false;
    System.out.println("Wating for cluster to become available.");
}
while (!clusterReady) {
    DescribeClustersResult result = client.describeClusters(new DescribeClustersRequest()
        .withClusterIdentifier(clusterIdentifier));
    String status = (result.getClusters()).get(0).getClusterStatus();
    if (status.equalsIgnoreCase("available")) {
        clusterReady = true;
    } else {
        System.out.print(".");
        Thread.sleep(sleepTime*1000);
    }
}
// snippet-end:[redshift.java.CreateAndModifyCluster.complete]

Managing clusters in a VPC

Topics
- Overview (p. 121)
- Creating a cluster in a VPC (p. 123)
- Managing VPC security groups for a cluster (p. 123)
- Configuring security group communication settings for Amazon Redshift clusters (p. 124)
- How Amazon Redshift works with VPC sharing for AWS resources (p. 126)
- Amazon Redshift cluster subnet groups (p. 127)

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. 69).

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 subnet groups (p. 127). 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, Amazon Redshift uses an elastic IP address for the external IP address. An elastic IP address is a static IP address. With it, you can 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. Whether you create an elastic IP address depends on your availability zone relocation setting. There are two options:

1. If you have availability zone relocation turned on and you want to enable public access, you don’t specify an elastic IP address. An elastic IP address managed by Amazon Redshift is assigned. It’s associated with your AWS account.
2. If you have availability zone relocation turned off and you want to enable public access, you can opt to create an elastic IP address for the VPC in Amazon EC2, prior to launching your Amazon Redshift cluster. If you don’t create an IP address, Amazon Redshift provides a configured elastic IP address to use for the VPC. This elastic IP address is managed by Amazon Redshift 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. 123).

**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 and backups (p. 410).

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-VPC when you create your cluster (p. 69). To create a VPC, see Create a 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. 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. 127).

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 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. 577).

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.

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 at 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.

### Configuring security group communication settings for Amazon Redshift clusters

This topic helps you configure your security groups to route and receive network traffic appropriately. The following are a couple common use cases:

- You turn on public accessibility for an Amazon Redshift cluster, but it isn't receiving traffic. You must configure an inbound rule to allow traffic to reach it from the internet.
- Your cluster isn't publicly accessible, and you use the default VPC security group to allow inbound traffic for private connectivity. But you have a requirement to use a security group other than the default, and this custom security group doesn't allow inbound traffic. You must configure it to allow communication.

The following sections help you choose the correct response for each use case and show you how to configure network traffic per your requirements. You can optionally use the steps to set up communication from other private security groups.
Note
Network traffic settings in most cases aren’t configured automatically in Amazon Redshift. This is because they can vary at a granular level, depending on whether the source of traffic is the internet or a private security group, and because security requirements vary.

Public accessibility with default or custom security group configuration

If you are creating or you already have a cluster, perform the following configuration steps to make the cluster publicly accessible. This applies both to when you choose the default security group or a custom security group:

1. Configure the Internet gateway and route table for your VPC. To connect to a publicly accessible cluster from the internet, an internet gateway must be attached to the route table. Confirm that the internet gateway's target is set with source 0.0.0.0/0 or a public IP CIDR. The route table must be associated with the VPC where your cluster resides. For more information regarding setting up internet access for a VPC, like what is described here, see Enable internet access in the Amazon VPC documentation. For more information about configuring a route table, see Configure route tables.

2. After you configure the internet gateway and route table, open inbound access by choosing Inbound rules for the security group associated with your cluster. Then choose Edit inbound rules.

3. Choose the Protocol and Port for the inbound rule, per your requirements. For an RA3 cluster, we recommend selecting a port within the ranges 5431-5455 or 8191-8215. When you are finished, save the rule.

4. Edit the Publicly accessible setting to enable it. You can do this from your cluster's Actions menu.

For more information about configuring your security group, see Amazon Redshift cluster security groups (p. 577).

Private accessibility with default or custom security group configuration

Using the default security group

If you choose the default security group when you create your cluster, and the cluster is private, you don't have to configure rules to enable network traffic. It isn't communicating with the internet. The default security group has pre-configured inbound and outbound rules to enable traffic from or to specific sources and destinations. (The rules can be viewed in the console for a cluster by choosing the Properties tab, then selecting the security group under Network and security settings.) These rules enable communication within the VPC. If you delete any of them, you have to replace them to enable connections to the cluster.

Using a custom security group

If you don't use the default security group, the security group you configure must allow communication to all subnets designated to Amazon Redshift. Perform the following steps to configure communication by adding inbound rules:

1. Choose your cluster and choose Properties. Then, for the Network and security settings, choose your VPC security group.

2. Open inbound access by choosing Edit inbound rules.

3. Configure Inbound access to create rules and allow traffic from other security groups or subnets as required. You can choose any protocol or port that fits your specifications. To illustrate, you
can configure an inbound rule with a **Source** property that's set to a subnet's CIDR range, like 192.0.2.0/24. (A CIDR block is a collection of IP addresses that share the same network prefix and number of bits.) If you don't need fine-grained control, you can add a rule with the source set to the CIDR range of the entire VPC.

4. After you configure rules for the security group, don't forget to assign it to the cluster.

Make sure that, if you change configuration for any security group, rules to allow traffic remain in place for all subnets specified for Amazon Redshift. For more information, see [VPC CIDR blocks](#) in the Amazon VPC user guide.

**How Amazon Redshift works with VPC sharing for AWS resources**

VPC sharing makes it so you can create AWS application resources, such as Amazon EC2 instances and other AWS services, in a shared, centrally-managed virtual private cloud (VPC). The account that owns the VPC (the owner) shares one or more subnets with other accounts (participants) that belong to the same AWS organization. This describes how you can create and use an Amazon Redshift cluster or Amazon Redshift Serverless workgroup in a shared VPC.

The benefits of VPC sharing include that you don't have to manage as many VPCs and it can help you simplify your network. The benefit specifically for Amazon Redshift administrators and users is that Redshift resources can operate productively in the shared VPC. For more information about VPC sharing, see [Share your VPC with other accounts](#), which goes into more detail regarding the benefits of VPC sharing and how it works.

**How to use Amazon Redshift data-warehouse resources in a shared VPC**

Firstly, it's important to understand that an Amazon Redshift cluster or an Amazon Redshift Serverless workgroup can't be made visible to participants in a shared subnet. But this doesn't preclude participants from working with the owner's database in a shared VPC. This is detailed more fully in the steps that follow.

Before you create a provisioned Amazon Redshift cluster in a shared VPC, you must create a subnet group that you intend to use for Amazon Redshift. This should include the subnets from the shared VPC that you want to use. When you create your Amazon Redshift cluster, you must choose this subnet and also specify the shared VPC's security group. Similarly, you have to specify the shared subnets and the security group you created in the shared VPC when you create the Amazon Redshift Serverless workgroup and database. After you set up your subnets, perform these steps to set up Redshift resources in the shared environment:

1. The VPC owner creates an Amazon Redshift cluster or Amazon Redshift Serverless workgroup, using a subnet in the shared VPC.
2. The VPC owner makes the cluster or workgroup available in a cross-VPC scenario. The steps are described in [Working with Redshift-managed VPC endpoints in Amazon Redshift](#) for a provisioned cluster or in [Connecting to Amazon Redshift Serverless from an Amazon Redshift managed VPC endpoint](#) for Amazon Redshift Serverless. By enabling cross-VPC availability, they can make the database available to users in the same AWS account or in other accounts.
3. Conversely, by means of VPC sharing, an owner can share a subnet with a participant, and the participant can create an Amazon Redshift cluster or Amazon Redshift Serverless workgroup in the subnet. However, the owner in this case can't view an Amazon Redshift resource created by a participant. The cluster or workgroup must be made accessible by enabling cross-VPC availability in the same manner as described in the previous step.
Usage notes for using Amazon Redshift resources in a shared VPC

Note the following behaviors regarding using Amazon Redshift in a shared subnet:

- As detailed in the previous section, the VPC owner can't share an Amazon Redshift cluster or Amazon Redshift Serverless workgroup with a participant through VPC sharing. But the participant can create a cluster or Amazon Redshift Serverless workgroup in the owner's subnet. In this instance, Amazon Redshift isn't visible through VPC sharing to the owner.
- The VPC owner can't view, update or delete an Amazon Redshift provisioned cluster or Amazon Redshift Serverless workgroup that the participant creates in the shared subnet.
- There aren't permissions available to make it so another AWS account can access Amazon Redshift resources you create in the shared VPC.

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/](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.

To change or remove subnets in some cases requires extra steps. For example, this AWS Knowledge Center article, [How do I move my provisioned Amazon Redshift cluster into a different subnet?](https://aws.amazon.com/community/articles/how-do-i-move-my-provisioned-amazon-redshift-cluster-into-a-different-subnet/), describes a use case that covers moving a cluster.

**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/](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. 584)](https://aws.amazon.com/community/articles/how-do-i-move-my-provisioned-amazon-redshift-cluster-into-a-different-subnet/). 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
 */
128```
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 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 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);
        // Get existing subnet group.
        DescribeClusterSubnetGroupsRequest request2 = new DescribeClusterSubnetGroupsRequest()
                .withClusterSubnetGroupName(clusterSubnetGroupName);
        DescribeClusterSubnetGroupsResult result2 = client.describeClusterSubnetGroups(request2);
        // Modify cluster subnet group.
        ModifyClusterSubnetGroupRequest request3 = new ModifyClusterSubnetGroupRequest()
                .withClusterSubnetGroupName(clusterSubnetGroupName)
                .withSubnetIds(subnetId2);
        client.modifyClusterSubnetGroup(request3);
        System.out.println("Updated cluster subnet group: " + clusterSubnetGroupName);
    }
}
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);
    printlnSubnetGroups(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()) {
        printlnSubnetGroup(group);
    }
}

private static void printResultSubnetGroup(ClusterSubnetGroup group) {
    System.out.format("Name: %s, Description: %s
", group.getClusterSubnetGroupName(),
    group.getDescription());
    for (Subnet subnet : group.getSubnets()) {
        System.out.format(" Subnet: %s, %s, %s\n", subnet.getSubnetIdentifier(),
            subnet.getSubnetAvailabilityZone().getName(),
            subnet.getSubnetStatus());
    }
}

Manage cluster subnet groups using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift CLI operations to manage cluster subnet groups.

- create-cluster-subnet-group
- delete-cluster-subnet-group
- describe-cluster-subnet-groups
- modify-cluster-subnet-group
You can use the following Amazon Redshift API operations to manage cluster subnet groups.

- CreateClusterSubnetGroup
- DeleteClusterSubnetGroup
- DescribeClusterSubnetGroups
- ModifyClusterSubnetGroup

Cluster version history

Amazon Redshift periodically releases new cluster versions that are used to update your cluster.

**Important**
For information about available Amazon Redshift cluster versions, and their features, improvements, and fixes, see [Cluster versions for Amazon Redshift (p. 654)](#).
Working with Amazon Aurora zero-ETL integrations (preview)

Amazon Aurora zero-ETL integrations with Amazon Redshift enable near real-time analytics and machine learning (ML) using Amazon Redshift on petabytes of transactional data from Amazon Aurora. It’s a fully managed solution for making transactional data available in Amazon Redshift after it's written to an Aurora cluster. Extract, transform, and load (ETL) is the process of combining data from multiple sources into a large, central repository called a data warehouse. For more information, see What Is ETL.

With Amazon Aurora zero-ETL integrations with Amazon Redshift, we take care of the ETL for you by automating the creation and management of ETL pipelines between Aurora and Redshift. You can perform real-time transaction processing on data in Aurora while simultaneously using Amazon Redshift for analytics workloads, such as reporting and dashboards.

To create a zero-ETL integration, you specify an Amazon Aurora DB cluster as the data source and an Amazon Redshift data warehouse as the target. The integration replicates data from the source database into the target data warehouse. The data becomes available in Amazon Redshift within seconds, allowing data analysts to begin using Amazon Redshift's analytics and machine learning functionality on the data.

The integration monitors the health of the data pipeline and recovers from issues when possible. You can create integrations from multiple Aurora clusters into a single Amazon Redshift namespace, enabling you to derive holistic insights across multiple applications.

Amazon Aurora zero-ETL integrations enable you to take advantage of Amazon Redshift's comprehensive capabilities, such as built-in ML, materialized views, data sharing, and direct access to multiple data stores and data lakes. Furthermore, you can use Amazon Aurora for transactions and Amazon Redshift for analytics. Compute resources are isolated for the same data, and you're using the most efficient tools to process it.

You pay for Aurora and Amazon Redshift resources (including data transfer) used as part of a zero-ETL integration. Ongoing replication of data changes is provided at no charge. For more information, see Amazon Redshift pricing and Amazon Aurora pricing.

For data engineers, zero-ETL integrations provide access to time-sensitive data that otherwise can get hindered by intermittent errors in complex data pipelines. You can run analytical queries and machine learning models on transactional data to derive near-real-time insights for time-sensitive events and business decisions.

The following is a summary of steps needed to use zero-ETL integrations:

- From the Resources policy tab, authorize the user or role and the Aurora MySQL DB cluster from which data will be replicated into the associated Amazon Redshift data warehouse.
- From the zero-ETL integration list, view the list of zero-ETL integrations created on Aurora and that are available on Amazon Redshift.
- From the zero-ETL integration list, view and share data that is replicated into the Amazon Redshift data warehouse.
• From the zero-ETL integration details page, view the various details of zero-ETL integration, such as integration metrics, table statistics, and Amazon CloudWatch metrics.
• From the QEv2, query the data that is replicated into the Amazon Redshift data warehouse.

See Benefits, Key concepts, Limitations, Unsupported data types, Quotas, and Supported Regions of zero-ETL integrations in the Amazon Aurora User Guide.

For questions, issues, or feedback, contact AWS Support.

Topics
• Preview limitations (p. 133)
• Getting started with zero-ETL integrations (p. 134)
• Creating destination databases in Amazon Redshift (p. 139)
• Querying and sharing data in Amazon Redshift (p. 140)
• Viewing zero-ETL integrations (p. 142)
• Monitoring zero-ETL integrations (p. 142)
• Troubleshooting zero-ETL integrations (p. 143)

Preview limitations

The following limitations apply to the preview release of Aurora zero-ETL integrations with Amazon Redshift.

Amazon Aurora limitations

• Your source DB cluster must be running Aurora MySQL version 3.03.1 (compatible with MySQL 8.0.26) or higher. Aurora PostgreSQL is not currently supported.
• Your source DB cluster must be in the same Region as the target Amazon Redshift data warehouse.
• You can only create and manage integrations using the AWS Management Console. You can't use the AWS CLI, the Amazon RDS API, or any of the SDKs.
• Some data types in Aurora MySQL aren't supported for zero-ETL integrations. If your source DB cluster includes an unsupported data type, the integration state changes to Needs attention. For more information, see Unsupported data types.
• You can't modify an integration after you create it. If you need to change certain settings, such as the integration name, you must delete and recreate the integration.
• You can't rename an Aurora DB cluster that has existing integrations.
• If you pause the source DB cluster, the last few transactions might not be replicated to the target data warehouse until you resume the cluster.
• Zero-ETL integrations rely on MySQL binary logging (binlog) to capture ongoing data changes. To make sure that Aurora can replicate all data changes to Amazon Redshift, we recommend that you do not use binlog-based data filtering.
• Zero-ETL integrations don't currently support table-level filtering.
• You can't delete an Amazon Aurora cluster that has existing zero-ETL integrations. You must delete all associated integrations first.
• Aurora MySQL system tables, temporary tables, and views aren't replicated to Amazon Redshift.
• Inserts into BLACKHOLE tables are replicated to Amazon Redshift, but updates and deletes are not.
• Integrations don't capture changes to foreign key child tables when foreign key actions are performed on the source DB cluster. Examples of such actions are ON DELETE/UPDATE CASCADE and ON UPDATE/DELETE SET NULL/DEFAULT. We recommend replacing these actions with triggers.
• Some ALTER TABLE partition operations, including DROP PARTITION, DISCARD PARTITION, TRUNCATE PARTITION, and EXCHANGE PARTITION, aren't supported and might cause unexpected results.
• Object identifiers (including database name, table name, column names, and others) can only contain alphanumeric characters, numbers, $, and _ (underscore).
• Object identifiers can't be Amazon Redshift reserved words or MySQL keywords.
• Object identifiers can't start with MySQL keywords as part of the identifier. This might cause unexpected results.
• If an ALTER TABLE statement has a default value with VARCHAR or CHAR data types, the default value can't contain MySQL keywords. This might cause unexpected results. In addition, any default value that contains nonspace whitespaces or consecutive whitespaces will be changed to single space.

Amazon Redshift limitations

• Your target Amazon Redshift data warehouse must meet the following prerequisites:
  • Created in Preview on the PREVIEW_2023 track. For more information, see the section called “Create a target Amazon Redshift data warehouse” (p. 135).
  • Running an RA3 node type (ra3.16xlarge, ra3.4xlarge, and ra3.xlplus), or Amazon Redshift Serverless.
  • Encrypted (if using a provisioned cluster).
  • Has case sensitivity enabled. For more information, see the section called “Turn on case sensitivity” (p. 136).
  • To share data from an Amazon Redshift destination database (producer) to an Amazon Redshift consumer data warehouse, your consumer data warehouse must be on the PREVIEW_2023 track. For more information, see Data sharing concepts for Amazon Redshift.
  • You can't perform pause, resume, or classic resize operations on Amazon Redshift provisioned clusters with integrations configured.
  • You can't enable enhanced VPC support on data warehouses with integrations configured.
  • You can't create an integration if the target Amazon Redshift provisioned cluster is paused. In addition, you can't pause a provisioned cluster if it has an existing integration.
  • If you remove a source DB cluster as an authorized integration source for a Amazon Redshift data warehouse, all associated integrations are deleted.
  • You can't delete an Amazon Redshift data warehouse that has existing zero-ETL integrations. You must delete all associated integrations first.
  • You can't delete a destination database when an integration is associated with the database.
  • The destination database is a read-only database. Materialized views are only supported when used in cross-database queries. For more information, see Querying data across databases.
  • You can't create tables, views, or materialized views in the destination database. When tables in the target data warehouse are in the Resync state, you can't query the tables or create materialized views from them.

Getting started with zero-ETL integrations

Prior to configuring your zero-ETL integration on Amazon Redshift, configure your Amazon Aurora DB cluster and set it up with the required parameters and permissions.

• From the Amazon RDS console – Create a source Amazon Aurora DB cluster as described in the Amazon Aurora User Guide.
• From the Amazon RDS console – Configure DB cluster parameters as described in the Amazon Aurora User Guide.
Then, continue to the rest of the initial setup from the Amazon Redshift console and AWS CLI.

- From the Amazon Redshift console – Create a target Amazon Redshift data warehouse (p. 135).
- From the AWS CLI or Amazon Redshift console – Turn on case sensitivity for your data warehouse (p. 136).
- From the Amazon Redshift console – Configure authorization for your Amazon Redshift data warehouse (p. 137).

After you complete the preceding tasks, continue to Creating zero-ETL integrations from the Amazon RDS console, as detailed in the Amazon Aurora User Guide.

Create a target Amazon Redshift data warehouse

Before this step, create a source DB cluster and configure the DB cluster parameters on the Amazon RDS console.

In this step, you create a target Amazon Redshift data warehouse, such as a preview Redshift Serverless workgroup or a preview provisioned cluster.

**Note**

The target data warehouse must be created in **Preview**, which is an Amazon Redshift deployment option that lets you test out new features. You can’t use those features in production or move your Preview data warehouse to a production deployment. If your Amazon Redshift data warehouse is a provisioned cluster, you must encrypt it. For more information, see Amazon Redshift database encryption (p. 444).

Your target data warehouse must have the following characteristics:

- Running Amazon Redshift Serverless or a provisioned cluster of instance type ra3.16xlarge, ra3.4xlarge, or ra3.xlplus.
- Created as a **Preview** cluster or workgroup. If you’re creating an Amazon Redshift provisioned cluster, you must choose the PREVIEW_2023 track. Data warehouses on the current or trailing track can’t be modified to any preview track.
- Has case sensitivity (**enable_case_sensitive_identifier**) turned on. For more information, see Turn on case sensitivity for your data warehouse (p. 136).
- Encrypted, if your target data warehouse is a Amazon Redshift provisioned cluster. For more information, see Amazon Redshift database encryption (p. 444).

To create a target data warehouse in preview, see one of the following topics depending on your deployment type:

- Amazon Redshift provisioned cluster – Preview features when using Amazon Redshift clusters (p. 64). Make sure that you choose the PREVIEW_2023 track in order to use zero-ETL integrations.
- Amazon Redshift Serverless – Follow the procedure below.

**Note**

You can create an Amazon Redshift Serverless workgroup in **Preview** to test new features of Amazon Redshift Serverless. You can’t use those features in production or move your Preview workgroup to a production workgroup. For preview terms and conditions, see Beta and Previews in AWS Service Terms.

To create a workgroup in Preview

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 **Severless dashboard**, and choose **Workgroup configuration**. The workgroups for your account in the current AWS Region are listed. A subset of properties of each workgroup is displayed in columns in the list.

3. A banner displays on the **Workgroups** list page that introduces preview. Choose the button **Create preview workgroup** to open the create workgroup page.

4. Enter properties for your workgroup. We recommend entering a name for the workgroup that indicates that it is in preview. Choose options for your workgroup, including options labeled as **-preview**, for the features you want to test. Continue through the pages to enter options for your workgroup and namespace. For general information about creating workgroups, see **Creating a workgroup with a namespace** in the *Amazon Redshift Management Guide*.

5. Choose **Create preview workgroup** to create a workgroup in preview.

6. When your preview workgroup is available, use your SQL client to load and query data.

When you create a provisioned cluster, Amazon Redshift also creates a default parameter group. Since you can't edit the default parameter group, you can either create a custom parameter group before creating a new cluster and associate it with the cluster, or edit the parameter group that will be associated with the created cluster. You must also turn on case sensitivity for the parameter group either when you are creating the custom parameter group or when editing a current one to use zero-ETL integrations.

You can create a custom parameter group using either the Amazon Redshift console or the AWS CLI as follows:

- Using the Amazon Redshift console – Managing parameter groups using the console (p. 402)
- Using the AWS CLI – Managing parameter groups using the AWS CLI and Amazon Redshift API (p. 409)

### Turn on case sensitivity for your data warehouse

Before you create a target Amazon Redshift data warehouse, turn on case sensitivity for your parameter group. This is required to support the case sensitivity of MySQL. **enable_case_sensitive_identifier** is a configuration value that determines whether name identifiers of databases, tables, and columns are case sensitive. This parameter must be turned on to create zero-ETL integrations in the data warehouse. For more information, see **enable_case_sensitive_identifier**.

For Amazon Redshift Serverless – Turn on case sensitivity for Amazon Redshift Serverless using the AWS CLI (p. 136). Note that you can only turn on case sensitivity for Amazon Redshift Serverless from the AWS CLI.

For Amazon Redshift provisioned clusters – Turn on case sensitivity for Amazon Redshift provisioned clusters using the Amazon Redshift console (p. 137) or Turn on case sensitivity for Amazon Redshift provisioned clusters using the AWS CLI (p. 137).

### Turn on case sensitivity for Amazon Redshift Serverless using the AWS CLI

Run the following AWS CLI command to turn on case sensitivity of MySQL.

```
aws redshift-serverless update-workgroup \   
  --workgroup-name destination-workgroup \   
  --config-parameters \   
  parameterKey=enable_case_sensitive_identifier,parameterValue=true
```

Wait for the workgroup status to be **Active** before proceeding to the next step.
Turn on case sensitivity for Amazon Redshift provisioned clusters using 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 left navigation pane, choose **Provisioned clusters dashboard**.
3. Choose the provisioned cluster that you want to replicate data into. The cluster details page appears.
4. Choose the **Properties** tab and select the parameter group. The workload management page appears.
5. Choose the parameter group.
6. Choose the **Parameters** tab.
7. Select **Turn on case-sensitive identifier**, then choose **Save**.
8. When prompted, choose **Save**.

Turn on case sensitivity for Amazon Redshift provisioned clusters using the AWS CLI

1. Because you can’t edit the default parameter group, from your terminal program, run the following AWS CLI command to create a custom parameter group and associate it with the provisioned cluster.

   ```bash
   aws redshift create-cluster-parameter-group
   --parameter-group-name zero-etl-params
   --parameter-group-family redshift-1.0
   --description "Param group for zero-ETL integrations"
   ```

2. Run the following AWS CLI command to turn on case sensitivity of MySQL.

   ```bash
   aws redshift modify-cluster-parameter-group
   --parameter-group-name zero-etl-params
   --parameters ParameterName=enable_case_sensitive_identifier,ParameterValue=true
   ```

3. Run the following command to associate the parameter group with the cluster.

   ```bash
   aws redshift modify-cluster
   --cluster-identifier destination-cluster
   --cluster-parameter-group-name zero-etl-params
   ```

4. Wait for the provisioned cluster to be available, then run the following command to reboot the cluster.

   ```bash
   aws redshift reboot-cluster
   --cluster-identifier destination-cluster
   ```

Configure authorization for your Amazon Redshift data warehouse

To replicate data from the Aurora MySQL DB cluster into your Amazon Redshift data warehouse, you must initially add the following two entities:

- **Authorized principal** – identifies the user or role that can create zero-ETL integrations into the data warehouse.
• **Authorized integration source** – identifies the Aurora MySQL DB cluster that can update the data warehouse.

### Add authorized principals

To create a zero-ETL integration into your Redshift Serverless workgroup or provisioned cluster, authorize access to the associated namespace or provisioned cluster.

You can skip this step if both of the following conditions are true:

- The AWS account that owns the Redshift Serverless workgroup or provisioned cluster also owns the source Aurora MySQL DB cluster.
- That principal is associated with an identity-based IAM policy with permissions to create zero-ETL integrations into this Redshift Serverless namespace or provisioned cluster.

#### Add authorized principals to an Amazon Redshift Serverless namespace

1. In the Amazon Redshift console, choose **Redshift Serverless** from the left navigation pane.
2. Choose the name of the namespace and go to the **Resource policy** tab. If you don't see this tab, see [The Resource policy tab isn't visible for my Amazon Redshift data warehouse](p. 143).
3. Choose **Add authorized principals**.
4. For each authorized principal you want to add, enter either the ARN of the AWS user or role, or the AWS account ID that you want to grant access to create zero-ETL integrations into the namespace. An account ID is stored as an ARN.
5. Choose **Save changes**.

#### Add authorized principals to an Amazon Redshift provisioned cluster

1. In the Amazon Redshift console, choose **Provisioned clusters dashboard** from the left navigation pane.
2. Choose **Clusters**, then choose the cluster, and go to the **Resource policy** tab. If you don't see this tab, see [The Resource policy tab isn't visible for my Amazon Redshift data warehouse](p. 143).
3. Choose **Add authorized principals**.
4. For each authorized principal you want to add, enter either the ARN of the AWS user or role, or the AWS account ID that you want to grant access to create zero-ETL integrations into the cluster. An account ID is stored as an ARN.
5. Choose **Save changes**.

### Add authorized integration sources

To allow your source DB cluster to update your Amazon Redshift data warehouse, you must add it as an authorized integration source to the namespace.

#### Add an authorized integration source to an Amazon Redshift Serverless namespace

1. In the Amazon Redshift console, go to **Serverless dashboard**.
2. Choose the name of the namespace.
3. Go to the **Resource policy** tab.
4. Choose **Add authorized integration source**.
5. Specify the ARN of the Aurora MySQL DB cluster that's the data source for the zero-ETL integration. For example, `arn:aws:rds:us-east-1:123456789012:cluster:zero-etl-cluster`.
**Note**
Removing an authorized integration source stops data from replicating into the namespace. This action deactivates all zero-ETL integrations from that source Aurora DB cluster into this namespace.

**Add an authorized integration source to an Amazon Redshift provisioned cluster**

1. In the Amazon Redshift console, go to **Provisioned clusters dashboard**.
2. Choose the name of the provisioned cluster.
3. Go to the **Resource policy** tab.
4. Choose **Add authorized integration source**.
5. Specify the ARN of the source Aurora DB cluster that's the data source for the zero-ETL integration. For example, `arn:aws:rds:us-east-1:123456789012:cluster:zero-etl-cluster`.

**Note**
Removing an authorized integration source stops data from replicating into the provisioned cluster. This action deactivates all zero-ETL integrations from that source Aurora DB cluster into this Amazon Redshift provisioned cluster.

**Next steps**

Now that you have configured authorization for your target Amazon Redshift data warehouse, you can create a zero-ETL integration and start replicating data. For instructions, see [Creating Amazon Aurora zero-ETL integrations with Amazon Redshift](https://docs.aws.amazon.com/redshift/latest/guides/aurora-zero-ETL-integration.html).

**Creating destination databases in Amazon Redshift**

To replicate data from Amazon Aurora into Amazon Redshift, you must create a destination database in Amazon Redshift.

Connect to your Redshift Serverless workgroup or provisioned cluster and create a database with a reference to your integration identifier. This identifier is the value returned for `integration_id` when you query the `SVV_INTEGRATION` view.

**Important**
Prior to adding data to the source cluster, your zero-ETL integration must be created and in the Active state on the Amazon RDS console as described in [Creating a zero-ETL integration](https://docs.aws.amazon.com/aurora/latest/userguide/rds-zero-ETL-integration-configuration.html) in the Amazon Aurora User Guide.

**Create a destination database in Amazon Redshift**

Before you can start replicating data from Amazon Aurora into Amazon Redshift, create a destination database in Amazon Redshift. You can either create the database using the Amazon Redshift console or the Query editor v2.

**Create a destination database using the Amazon Redshift console**

1. From the left navigation pane, choose **Zero-ETL integrations**.
2. From the integration list, choose an integration. The integration details page appears.
3. If you're using a provisioned cluster, you must connect to the database. Choose **Connect to database**. You can connect using a recent connection, or by creating a new connection.
4. To create a database from the integration, choose **Create database from integration**.
5. Enter a **Database name**. The Integration ID and data warehouse name are pre-populated. Then, choose **Create database**.

### Create a destination database using the Query editor v2

1. Navigate to the Amazon Redshift console and choose **Query editor v2**.
2. Choose your Amazon Redshift Serverless workgroup or Amazon Redshift provisioned cluster in the left panel and connect to it.
3. To get the integration ID, navigate to the integration list on the Amazon Redshift console.
   
   Alternatively, run the following command to get the `integration_id` value:
   
   ```sql
   SELECT integration_id from SVV_INTEGRATION;
   ```

4. Then, run the following command to create the database. By specifying the integration ID, you create a connection between the database and your source Aurora DB cluster.

   ```sql
   CREATE DATABASE my_database FROM INTEGRATION 'integration_id';
   ```

Only the zero-ETL integration that you created in Amazon Aurora can update data in this database. You can run only DDL commands against tables in the source cluster and read-only DML commands against tables on both the source and destination.

The following is a list of destination database states that you can view from the integration list:

- **No database** – There is no destination database for the integration.
- **Creating** – Amazon Redshift is creating the destination database for the integration.
- **Active** – Data is being replicated from Amazon Aurora into Amazon Redshift.
- **Error** – There is an error with the integration.
- **Recovering** – The integration is recovering after the data warehouse restarted.
- **Resyncing** – Amazon Redshift is resynchronizing the tables in the integration.

Next, you can add data to your Aurora DB cluster. For more information, see [Adding data to a source Aurora DB cluster and querying it in Amazon Redshift](#).

### Next steps

After creating a destination database, you can [add data to your Aurora MySQL cluster](#). You can then [query that data in Amazon Redshift](#).

### Querying and sharing data in Amazon Redshift

#### Query your data in Amazon Redshift

After you add data to the source Aurora DB cluster, it's replicated near real-time into the Amazon Redshift data warehouse and is ready to be queried. For information about integration metrics and table statistics, see [Monitoring zero-ETL integrations](#).
Note
As a database is the same as a schema in MySQL, MySQL database level maps to Amazon Redshift schema level. Note this mapping difference when you query data replicated from Amazon Aurora.

To query the replicated data
1. Navigate to the Amazon Redshift console and choose Query editor v2.
2. Connect to your Amazon Redshift Serverless workgroup or Amazon Redshift provisioned cluster and choose your database from the drop-down menu.
3. Use a SELECT statement to select all data from the schema and table that you created in the source Aurora DB cluster.

You can also query the data using the Amazon Redshift CLI.

Note
For case-sensitivity, use double quotes (" ") for schema, table, and column names. For more information, see enable_case_sensitive_identifier.

Share your data in Amazon Redshift

After you add data to the source Aurora DB cluster, it’s immediately replicated into Amazon Redshift and ready to be shared by creating datashares.

Before you can share data, you must create a destination database first.

To share data from a Amazon Redshift destination database (producer) to an Amazon Redshift consumer data warehouse, your consumer data warehouse must be on the PREVIEW_2023 track. For more information, see Data sharing concepts for Amazon Redshift.

Share data for Amazon Redshift Serverless using the Amazon Redshift console
1. In the Amazon Redshift console, on the left navigation pane, choose Amazon Redshift Serverless > Serverless dashboard.
2. From the left navigation pane, choose Zero-ETL integrations.
3. Choose Share data. The create datashare page appears.
4. Follow the steps in Creating datashares.

Share data for Amazon Redshift provisioned clusters using the Amazon Redshift console
1. In the Amazon Redshift console, on the left navigation pane, choose Provisioned clusters dashboard.
2. From the left navigation pane, choose Zero-ETL integrations.
3. From the integration list, choose an integration. The integration details page appears.
4. Click Connect to database. The Connection to database page appears.
5. You can either create a new connection or use a recent connection. Make sure that the connection is made to the destination database.
6. If you create a new connection, then enter a Database name for the database. Then, click Connect. The integration details page appears.
7. Choose Share data. The create datashare page appears.
8. Follow the steps in Creating datashares.
Viewing zero-ETL integrations

You can view the details of a zero-ETL integration to see its configuration information and status either from the Amazon Redshift console and system tables.

To view the details of a zero-ETL integration

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. From the left navigation pane, choose either the Serverless or Provisioned clusters dashboard. Then, choose Zero-ETL integrations.
3. Select the zero-ETL integration that you want to view.

An integration can have the following states:

- **Creating** – Amazon Aurora is creating the integration.
- **Active** – The zero-ETL integration is sending transactional data to the Amazon Redshift data warehouse.
- **Inactive** – The zero-ETL integration encountered unrecoverable errors. For help with troubleshooting this state, see Troubleshooting zero-ETL integrations (p. 143).
- **Syncing** – The zero-ETL integration is syncing transactional data to the Amazon Redshift data warehouse.
- **Needs attention** – The integration encountered an event or error that requires manual intervention to resolve it. To fix the issue, follow the steps in the error message.
- **Deleting** – The integration is being deleted.
- **Deleted** – The integration has been deleted.

Monitoring zero-ETL integrations

You can monitor your zero-ETL integrations by querying specific system views in Amazon Redshift. For more information, see SVV_INTEGRATION that provides information about configuration details of zero-ETL integrations, SYS_INTEGRATION_ACTIVITY that provides information about completed zero-ETL integrations, and SVV_INTEGRATION_TABLE_STATE that provides information about integration state.

You can also use the metrics in Amazon Redshift and Amazon CloudWatch to learn about the health and performance of your zero-ETL integrations. You can adjust the metrics to display data for shorter or longer duration, or choose to view metrics in CloudWatch. Navigate to the zero-ETL integration on the Amazon Redshift console and view the metrics.

Amazon Redshift provides two types of metrics, as follows:

- From the Integration metrics tab, these are the available charts:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lag</td>
<td>The lag from the time data is committed to Amazon Aurora to the time when the data is available for queries in Amazon Redshift.</td>
</tr>
<tr>
<td>Units: Second</td>
<td>Dimensions: IntegrationLag</td>
</tr>
</tbody>
</table>
Troubleshooting zero-ETL integrations

You can check the state of a zero-ETL integration by querying the `SVV_INTEGRATION` in Amazon Redshift. If the state column has a value of `ErrorState`, it means something's wrong with the integration. For more information, see Monitoring zero-ETL integrations (p. 142).

Use the following information to troubleshoot common issues with zero-ETL integrations.

**Topics**

- The Resource policy tab isn't visible for my Amazon Redshift data warehouse (p. 143)
- Creation of the integration failed (p. 143)
- Tables don't have primary keys (p. 144)
- Unsupported data types in tables (p. 144)
- Data manipulation language commands failed (p. 144)
- Tracked changes between data sources don't match (p. 144)
- Authorization failed (p. 144)
- Tables or schemas are more than 200K (p. 145)
- Amazon Redshift can't load data (p. 145)
- Workgroup parameter settings are incorrect (p. 145)
- Database isn't created to activate a zero-ETL integration (p. 145)

### The Resource policy tab isn't visible for my Amazon Redshift data warehouse

If you can't see the Resource policy tab for your Amazon Redshift data warehouse, you probably didn't create it in Preview. To resolve this issue, delete the data warehouse and recreate it in Preview. For instructions, see Create a target Amazon Redshift data warehouse (p. 135).

### Creation of the integration failed

If the creation of the zero-ETL integration failed, the status of the integration is Inactive. Make sure that the following are correct for your source Aurora DB cluster:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tables replicated</td>
<td>The number of tables that have been replicated from Amazon Aurora to Amazon Redshift.</td>
</tr>
<tr>
<td>Units: Count</td>
<td></td>
</tr>
<tr>
<td>Dimensions: IntegrationNumTablesReplicated</td>
<td></td>
</tr>
<tr>
<td>Tables failed</td>
<td>The number of tables that failed replication.</td>
</tr>
<tr>
<td>Units: Count</td>
<td></td>
</tr>
<tr>
<td>Dimensions: IntegrationNumTablesFailedReplication</td>
<td></td>
</tr>
</tbody>
</table>
In addition, make sure the following are correct for your Amazon Redshift data warehouse:

- Case sensitivity is turned on. See Turn on case sensitivity for your data warehouse (p. 136).
- You added the correct authorized principal and integration source for your namespace. See Configure authorization for your Amazon Redshift data warehouse (p. 137).

### Tables don't have primary keys

In the destination database, one or more of the tables don't have a primary key and can't be synchronized.

To resolve this issue, go to the Table statistics tab or use SVV_INTEGRATION_TABLE_STATE to view the failed tables. Then, drop these tables on Amazon RDS and create tables with a primary key. For more information, see Amazon Redshift best practices for designing tables.

### Unsupported data types in tables

In the destination database that you created in Amazon Redshift and in which data is replicated from the Aurora DB cluster, one or more of the tables have unsupported data types and can't be synchronized.

To resolve this issue, go to the Table statistics tab or use SVV_INTEGRATION_TABLE_STATE to view the failed tables. Then, modify these tables on Amazon RDS. For more information on unsupported data types, see Unsupported data types.

### Data manipulation language commands failed

Amazon Redshift could not run DML commands on the Redshift tables. Redshift automatically resynchronizes the tables to resolve this error.

### Tracked changes between data sources don't match

This error occurs when changes between Amazon Aurora and Amazon Redshift don't match, leading to the integration entering a Failed state.

To resolve this, delete the zero-ETL integration and create it again in Amazon RDS. For more information, see Creating zero-ETL integrations and Deleting zero-ETL integrations.

### Authorization failed

Authorization failed because the source Aurora DB cluster was removed as an authorized integration source for the Amazon Redshift data warehouse.

To resolve this issue, delete the zero-ETL integration and create it again on Amazon RDS. For more information, see Creating zero-ETL integrations and Deleting zero-ETL integrations.
Tables or schemas are more than 200K

Amazon Aurora can't send data to Amazon Redshift. The number of tables and schemas exceeds the 200K limit.

To resolve this issue, delete the zero-ETL integration and create it again on Amazon RDS. For more information about table limits for Amazon Redshift, see Quotas and limits in Amazon Redshift (p. 643).

Amazon Redshift can't load data

Amazon Redshift can't load data to the zero-ETL integration.

To resolve this issue, delete the zero-ETL integration on Amazon RDS and create it again. For more information, see Creating zero-ETL integrations and Deleting zero-ETL integrations.

Workgroup parameter settings are incorrect

Your workgroup doesn't have case sensitivity turned on.

To resolve this issue, go to the Properties tab, choose the parameter group, and turn on the case-sensitive identifier from the Properties tab. If you don't have an existing parameter group, create one with the case-sensitive identifier turned on. Then, create a new zero-ETL integration on Amazon RDS. For more information, see Creating zero-ETL integrations.

Database isn't created to activate a zero-ETL integration

There isn't a database created for the zero-ETL integration to activate it.

To resolve this issue, create a database for the integration. For more information, see Create a destination database in Amazon Redshift (p. 139).
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.

**Note**

When you write stored procedures, we recommend a best practice for securing sensitive values: Don't hard code any sensitive information in stored procedure logic. For example, don't assign a user password in a CREATE USER statement in the body of a stored procedure. This poses a security risk, because hard-coded values can be recorded as schema metadata in catalog tables. Instead, pass sensitive values, such as passwords, as arguments to the stored procedure, by means of parameters.

For more information about stored procedures, see [CREATE PROCEDURE](#) and [Creating stored procedures in Amazon Redshift](#). For more information about catalog tables, see [System catalog tables](#).

**Topics**

- [Querying a database using the Amazon Redshift query editor v2](#) (p. 146)
- [Querying a database using the query editor](#) (p. 192)
- [Connecting to an Amazon Redshift cluster using SQL client tools](#) (p. 199)
- [Using the Amazon Redshift Data API](#) (p. 351)

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. For a list of AWS Regions where the query editor v2 is available, see the endpoints listed for [Redshift query editor v2](#) in the [Amazon Web Services General Reference](#).

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.

For a demo of using query editor v2 running multiple queries with either an isolated or shared connection, watch the following video. Concurrent Query Execution using 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. 147)
- Working with query editor v2 (p. 151)
- Loading data into a database (p. 163)
- Authoring and running queries (p. 168)
- Authoring and running notebooks (p. 172)
- Querying the AWS Glue Data Catalog (p. 174)
- Querying a data lake (p. 176)
- Working with datashares (p. 178)
- Scheduling a query with query editor v2 (p. 180)
- Visualizing query results (p. 186)
- Collaborating and sharing as a team (p. 191)

**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 or workgroup 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. 148). The administrator can also optionally choose an S3 bucket and path that is used for some features, such as loading data from a file. For more information, see Loading data from a local file setup and workflow (p. 165).

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 Encryption at rest (p. 444), Encryption in transit (p. 453), and Configuration and vulnerability analysis in Amazon Redshift (p. 581).

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. To learn more about how query editor v2 runs on AWS CloudTrail, see Logging with CloudTrail (p. 567). For more information about CloudTrail, see the AWS CloudTrail User Guide.
The query editor v2 has adjustable quotas for some of its resources. For more information, see Quotas for Amazon Redshift objects (p. 643).

**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 with a user. We recommend attaching policies to an IAM role and assigning the role to a 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 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 encryption customer managed key:

You can create a symmetric encryption 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 encryption 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.

When you create an AWS account, you begin with one 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 don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the AWS Account Management Reference Guide.

```json
{
   "Version": "2012-10-17",
   "Id": "key-consolepolicy",
}
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 role to grant permission. (We recommend attaching policies to an IAM role and assigning the role to a user.) 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 (a user with an IAM role assigned) 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 roles used 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. 503).

For a principal (typically a user with an IAM role assigned) to turn on the ability in the Account settings page for others in the account to Export result set, they need the sqlworkbench:UpdateAccountExportSettings permission attached the role. This permission is included in the AmazonRedshiftQueryEditorV2FullAccess AWS managed policy.

As new features are added to query editor v2, the AWS managed policies are updated as needed. If you create your own policy based on the permissions allowed and denied in the provided managed policies, edit your policies to keep them up to date with changes to the managed policies. For more information about managed policies in Amazon Redshift, see AWS managed policies for Amazon Redshift (p. 464).

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center (successor to AWS Single Sign-On):
  
  Create a permission set. Follow the instructions in Create a permission set in the AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide.

- Users managed in IAM through an identity provider:
  
  Create a role for identity federation. Follow the instructions in Creating a role for a third-party identity provider (federation) in the IAM User Guide.

- IAM users:
  
  Create a role that your user can assume. Follow the instructions in Creating a role for an IAM user in the IAM User Guide.

  (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the IAM User Guide.

### Setting up principal tags to connect to query editor v2 as a federated user

To access the query editor v2 as a federated user, either set up your IAM role or user with principal tags. Or, set up your identity provider (IdP) to pass in RedshiftDbUser and (optionally) RedshiftDbGroups. For more information about using IAM to manage tags, see Passing session tags in AWS Security Token Service in the IAM User Guide. To set up access using AWS Identity and Access Management, an administrator can add tags using the IAM console (https://console.aws.amazon.com/iam/).

#### To add principal tags 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** in the navigation pane.
3. Choose the role that needs access to the query editor v2 using a federated user.
4. Choose the **Tags** tab.
5. Choose the **Manage tags**.
6. Choose **Add tag** and enter the **Key** as `RedshiftDbUser` and enter a **Value** of the federated user name.
7. Optionally choose **Add tag** and enter the **Key** as `RedshiftDbGroups` and enter a **Value** of the group name to associate to the user.
8. Choose **Save changes** to view the list of tags associated with your chosen IAM role. Propagating changes might take several seconds.
9. To use the federated user, refresh your query editor v2 page after the changes have propagated.

**Setup your identity provider (IdP) to pass principal tags**

The procedure to set up tags using an identity provider (IdP) varies by IdP. See your IdP documentation for instructions on how to pass user and group information to SAML attributes. When configured correctly, the following attributes appear in your SAML response that is used by the AWS Security Token Service to populate in the principal tags for RedshiftDbUser and RedshiftDbGroups.

```xml
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:RedshiftDbUser">
  <AttributeValue>db-user-name</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:RedshiftDbGroups">
  <AttributeValue>db-groups</AttributeValue>
</Attribute>
```

The optional `db_groups` must be a colon-separated list such as `group1:group2:group3`.

Additionally, you can set the `TransitiveTagKeys` attribute to persist the tags during role chaining.

```xml
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>RedshiftDbUser</AttributeValue>
  <AttributeValue>RedshiftDbGroups</AttributeValue>
</Attribute>
```

For more information about setting up query editor v2, see [Permissions required to use the query editor v2](p. 475).

**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 databases, you can view its schemas. For each schema, you can view its tables, views, UDFs, and stored procedures.

**Topics**

- [Opening query editor v2](p. 152)
- [Connecting to an Amazon Redshift database](p. 154)
- [Browsing an Amazon Redshift database](p. 155)
- [Creating database objects](p. 157)
- [Viewing query and tab history](p. 161)
- [Considerations when working with query editor v2](p. 162)
- [Changing account settings](p. 162)
**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 browser tab.

The query editor page has a navigator menu where you choose a view as follows:

![Editor](image)

You manage and query your data organized as tables and contained in a database. The database can contain stored data or contain a reference to data stored elsewhere, such as Amazon S3. You connect to a database contained in either a cluster or a serverless workgroup.

When working in the Editor view, you have the following controls:

- The Cluster or Workgroup field displays the name you are currently connected to. The Database field displays the databases within the cluster or workgroup. 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 or workgroups, 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 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. You can also Export your results.

- A Notebook area where you can add sections to enter and run SQL or add Markdown.

After you run a query, a Result tab appears with the results. Here is where you can Export your results.

**Queries**

A query contains the SQL commands to manage and query your data in a database. When you use query editor v2 to load sample data, it also creates and saves sample queries for you.
When you choose a saved query, you can open, rename, and delete it using the context (right-click) menu. You can view attributes such as the **Query ARN** of a saved query by choosing **Query details**. You can also view its version history, edit tags attached to the query, and share it with your team.

### Notebooks

A SQL notebook contains SQL and Markdown cells. Use notebooks to organize, annotate, and share multiple SQL commands in a single document.

When you choose a saved notebook, you can open, rename, duplicate, and delete it using the context (right-click) menu. You can view attributes such as the **Notebook ARN** of a saved notebook by choosing **Notebook details**. You can also view its version history, edit tags attached to the notebook, export it, and share it with your team. For more information, see [Authoring and running notebooks](p. 172).

### Charts

A chart is a visual representation of your data. The query editor v2 provides the tools to create many types of charts and save them.

When you choose a saved chart, you can open, rename, and delete it using the context (right-click) menu. You can view attributes such as the **Chart ARN** of a saved chart by choosing **Chart details**. You can also edit tags attached to the chart and export it. For more information, see [Visualizing query results](p. 186).

### History

The query history is a list of queries you ran using Amazon Redshift query editor v2. These queries either ran as individual queries or as part of a SQL notebook. For more information, see [Viewing query and tab history](p. 161).

### Scheduled queries

A scheduled query is a query that is set up to start at specific times.

All query editor v2 views have the following icons:

- A **Visual mode** icon to toggle between light mode and dark mode.
- A **Settings** icon to show a menu of the different settings screens.
- An **Editor preferences** icon to edit your preferences when you use query editor v2.
- A **Connections** icon to view the connections used by your editor tabs.
A connection is used to retrieve data from a database. A connection is created for a specific database. With an isolated connection, the results of a SQL command that changes the database, such as creating a temporary table, in one editor tab, are not visible in another editor tab. When you open an editor tab in query editor v2, the default is an isolated connection. When you create a shared connection, that is, turn off the Isolated session switch, then the results in other shared connections to the same database are visible to each other. However, editor tabs using a shared connection to a database don't run in parallel. Queries using the same connection must wait until the connection is available. A connection to one database can't be shared with another database, and thus SQL results are not visible across different database connections.

The number of connections any user in the account can have active is controlled by a query editor v2 administrator.

• An Account settings icon used by an administrator to change certain settings of all users in the account. For more information, see Changing account settings (p. 162).

### Connecting to an Amazon Redshift database

To connect to a database, choose the cluster or workgroup name in the tree-view panel. If prompted, enter the connection parameters.

When you connect to a cluster or workgroup and its databases, you usually provide a Database name. You also provide parameters required for one of the following authentication methods:

**Federated user**

With this method, the principal tags of your IAM role or user must provide the connection details. You configure these tags in AWS Identity and Access Management or your identity provider (IdP). The query editor v2 relies on the following tags.

- RedshiftDbUser – This tag defines the database user that is used by query editor v2. This tag is required.
- RedshiftDbGroups – This tag defines the database groups that are joined when connecting to query editor v2. This tag is optional and its value must be a colon-separated list such as group1:group2:group3. Empty values are ignored, that is, group1:::group2 is interpreted as group1:group2.

These tags are forwarded to the redshift:GetClusterCredentials API to get credentials for your cluster. For more information, see Setting up principal tags to connect to query editor v2 as a federated user (p. 150).

**Temporary credentials**

This option is only available when connecting to a cluster. With this method, query editor v2, provide a User name for the database. The query editor v2 generates a temporary password to connect to the database. A user using this method to connect must be allowed IAM permission to redshift:GetClusterCredentials. To prevent users from using this method, modify their IAM user or role to deny this permission. Your query editor v2 administrator chooses whether this method is shown to all users in the account by configuring it in Account settings. For more information, see Changing account settings (p. 162).

**Temporary credentials using your IAM identity**

This option is only available when connecting to a cluster. With this method, query editor v2 maps a user name to your IAM identity and generates a temporary password to connect to the database. A user using this method to connect must be allowed IAM permission to redshift:GetClusterCredentialsWithIAM. To prevent users from using this method, modify
their IAM user or role to deny this permission. Your query editor v2 administrator chooses whether this method is shown to all users in the account by configuring it in Account settings. For more information, see Changing account settings (p. 162).

**Database user name and password**

With this method, also provide a **User name** and **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.

**AWS Secrets Manager**

This option is only available when connecting to a cluster. With this method, instead of a database name, you provide a **Secret** stored in Secrets Manager that contains your database and sign-in credentials. Use the Secrets Manager console (https://console.aws.amazon.com/secretsmanager/) to Store a new secret and choose the secret type **Credentials for Amazon Redshift cluster** to enter your credentials for a cluster. You must also add a tag key that starts with the string “Redshift” for the secret to be listed on the query editor v2 console.

When you select a cluster or workgroup with query editor v2, depending on the context, you can create, edit, and delete connections using the context (right-click) menu. You can view attributes such as the **Connection ARN** of the connection by choosing **Connection details**. You can also edit tags attached to the connection.

**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.

The hierarchical tree-view panel displays database objects. To refresh the tree-view panel to display database objects that might have been created after the tree-view was last displayed, choose the icon. Open the context (right-click) menu for an object to see what actions you can perform.
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.

Creating database objects

You can create database objects, including databases, schemas, tables, and user-defined functions (UDFs). You must be connected to a cluster or workgroup and a database to create database objects.

Creating databases

You can use query editor v2 to create databases in your cluster or workgroup.

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. (Optional) You can create the database from a datashare or the AWS Glue Data Catalog.
   - (Optional) Select Create using a datashare, and choose a Select a datashare. The list includes producer datashares that can be used to create a consumer datashare in the current cluster or workgroup.
   - (Optional) Select Create using AWS Glue Data Catalog, and choose a Choose an AWS Glue database.

5. Choose Create database.

The new database displays in the tree-view panel.

When you query a database created from a datashare, connect to a Amazon Redshift database in the cluster or workgroup (for example, the default database dev), and use three-part notation (database.schema.table) that references the database name you created when you selected Create using a datashare. The datasharing database is listed in the query editor v2 editor tab, but it is not enabled for direct connection.
When you query a database created from a AWS Glue Data Catalog, connect to your Amazon Redshift database in the cluster or workgroup (for example, the default database dev), and use three-part notation (database.schema.table) that references the database name you created when you selected Create using AWS Glue Data Catalog, the schema you named in Data catalog schema, and the table in the AWS Glue Data Catalog. Similar to:

```
SELECT * FROM glue-database.glue-schema.glue-table
```

**Note**
Confirm that you are connected to the default database using the connection method Temporary credentials using your IAM identity, and that your IAM credentials have been granted usage privilege to the AWS Glue database.

```
GRANT USAGE ON DATABASE glue-database to "IAM:MyIAMUser"
```

The AWS Glue database is listed in the query editor v2 editor tab, but it is not enabled for direct connection.

For more information about querying an AWS Glue Data Catalog, see Working with Lake Formation-managed datashares as a consumer and Working with Lake Formation-managed datashares as a producer in the Amazon Redshift Database Developer Guide.

**Example creating a database as a datashare consumer**

The following example describes a specific scenario that was used to create a database from a datashare using query editor v2. Review this scenario to learn how you can create a database from a datashare in your environment. This scenario uses two clusters, cluster-base (the producer cluster) and cluster-view (the consumer cluster).

1. Use the Amazon Redshift console to create a datashare for the table category2 in cluster cluster-base. The producer datashare is named datashare_base.

   For information about creating datashares, see Sharing data across clusters in Amazon Redshift in the Amazon Redshift Database Developer Guide.

2. Use the Amazon Redshift console to accept the datashare datashare_base as a consumer for the table category2 in cluster cluster-view.

3. View the tree-view panel in query editor v2 which shows the hierarchy of cluster-base as:

   - Cluster: cluster-base
     - Database: dev
       - Schema: public
         - Tables: category2

4. Choose Create, and then choose Database.

5. Enter see_datashare_base for Database name.

6. Select Create using a datashare, and choose a Select a datashare. Choose datashare_base to use as the source of the database you are creating.

   The tree-view panel in query editor v2 shows the hierarchy of cluster-view as:

   - Cluster: cluster-view
     - Database: see_datashare_base
• Schema: public
• Tables: category2

7. When you query the data, connect to the default database of the cluster `cluster-view` (typically named dev), but reference the datashare database `see_datashare_base` in your SQL.

   **Note**
   In the query editor v2 editor view, the selected cluster is `cluster-view`. The selected database is dev. The database `see_datashare_base` is listed but is not enabled for direct connection. You choose the dev database and reference `see_datashare_base` in the SQL you run.

   ```sql
   SELECT * FROM "see_datashare_base"."public"."category2";
   ```

   The query retrieves data from the datashare `datashare_base` in the cluster `cluster_base`.

**Example creating a database from an AWS Glue Data Catalog**

The following example describes a specific scenario that was used to create a database from an AWS Glue Data Catalog using query editor v2. Review this scenario to learn how you can create a database from an AWS Glue Data Catalog in your environment. This scenario uses one cluster, `cluster-view` to contain the database you create.

1. Choose **Create**, and then choose **Database**.

2. Enter `data_catalog_database` for **Database name**.

3. Select **Create using a AWS Glue Data Catalog**, and choose **Choose an AWS Glue database**. Choose `glue_db` to use as the source of the database you are creating.

   Choose **Data catalog schema** and enter `myschema` as the schema name to use in three-part notation.

   The tree-view panel in query editor v2 shows the hierarchy of `cluster-view` as:

   • Cluster: `cluster-view`
     • Database: `data_catalog_database`
       • Schema: `myschema`
         • Tables: `category3`

4. When you query the data, connect to the default database of the cluster `cluster-view` (typically named dev), but reference the database `data_catalog_database` in your SQL.

   **Note**
   In the query editor v2 editor view, the selected cluster is `cluster-view`. The selected database is dev. The database `data_catalog_database` is listed but is not enabled for direct connection. You choose the dev database and reference `data_catalog_database` in the SQL you run.

   ```sql
   SELECT * FROM "data_catalog_database"."myschema"."category3";
   ```

   The query retrieves data that is cataloged by AWS Glue Data Catalog.
Creating schemas

You can use query editor v2 to create schemas in your cluster or workgroup.

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. Choose either Local or External as the Schema type.
   For more information about local schemas, see CREATE SCHEMA in the Amazon Redshift Database Developer Guide. For more information about external schemas, see CREATE EXTERNAL SCHEMA in the Amazon Redshift Database Developer Guide.
4. If you choose External, then you have the following choices of an external schema.  
   - Glue Data Catalog – to create an external schema in Amazon Redshift that references tables in AWS Glue. Besides choosing the AWS Glue database, choose the IAM role associated with the cluster and the IAM role associated with the Data Catalog.
   - PostgreSQL – to create an external schema in Amazon Redshift that references an Amazon RDS for PostgreSQL or Amazon Aurora PostgreSQL-Compatible Edition database. Also provide the connection information to the database. For more information about federated queries, see Querying data with federated queries in the Amazon Redshift Database Developer Guide.
   - MySQL – to create an external schema in Amazon Redshift that references an Amazon RDS for MySQL or and Amazon Aurora MySQL-Compatible Edition database. Also provide the connection information to the database. For more information about federated queries, see Querying data with federated queries in the Amazon Redshift Database Developer Guide.
5. Choose Create schema.
   The new schema appears in the tree-view panel.

Creating tables

You can use query editor v2 to create tables in your cluster or workgroup.

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 choose 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 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.

Creating functions

You can use query editor v2 to create functions in your cluster or workgroup.

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.

Viewing query and tab history

You can view your query history with query editor v2. Only queries that you ran using query editor v2 appear in the query history. Both queries that ran from using an Editor tab or Notebook tab are
shown. You can filter the list displayed by a time period, such as This week, where a week is defined as Monday–Sunday. The list of queries fetches 25 rows of queries that match your filter at a time. Choose **Load more** to see the next set. Choose a query and from the **Actions** menu. The actions available depend on whether the chosen query has been saved. You can choose the following operations:

- **View query details** – Displays a query details page with more information about the query that ran.
- **Open query in a new tab** – Opens a new editor tab and primes it with the chosen query. If still connected, the cluster or workgroup and database are automatically selected. To run the query, first confirm that the correct cluster or workgroup and database are chosen.
- **Open source tab** – If it is still open, navigates to the editor or notebook tab that contained the query when it ran. The contents of the editor or notebook might have changed after the query ran.
- **Open saved query** – Navigates to the editor or notebook tab and opens the query.

You can also view the history of queries run in an **Editor** tab or the history of queries run in a **Notebook** tab. To see a history of queries in a tab, choose **Tab history**. Within the tab history, you can do the following operations:

- **Copy query** – Copies the query version SQL content to the clipboard.
- **Open query in a new tab** – Opens a new editor tab and primes it with the chosen query. To run the query, you must choose the cluster or workgroup and database.
- **View query details** – Displays a query details page with more information about the query that ran.

### Considerations when working with query editor v2

Consider the following when working with query editor v2.

- The maximum query result size is the smaller of 5 MB or 100,000 rows.
- You can run a query up to 300,000 characters long.
- You can save a query up to 30,000 characters long.
- By default, query editor v2 auto-commits each individual SQL command that runs. When a **BEGIN** statement is provided, statements within **BEGIN-COMMIT** or **BEGIN-ROLLBACK** block are run as a single transaction. For more information about transactions, see **BEGIN** in the *Amazon Redshift Database Developer Guide*.
- The maximum number of warnings that query editor v2 displays while running a SQL statement is 10. For example, when a stored procedure is run, no more than 10 RAISE statements are displayed.

### Changing account settings

A user with the right IAM permissions can view and change **Account settings** for other users in the same AWS account. This administrator can view or set the following:

- The maximum concurrent database connections per user in the account. This includes connections for **Isolated sessions**. When you change this value, it can take 10 minutes for the change to take effect.
- Select whether to **Authenticate with IAM credentials**:
  - When you select this option, then account users are shown **Temporary credentials using your IAM identity** on the connection window to authenticate with their IAM credentials.
  - When you deselect this option, then account users are shown **Temporary credentials** on the connection window to authenticate with database user name and password. The default is for this option to be deselected.
- Allow users in the account to export an entire result set from a SQL command to a file.
- Load and display sample databases with some associated saved queries.
Loading data into a database

You can use query editor v2 to load data into a database in an Amazon Redshift cluster or workgroup.

Loading sample data

The query editor v2 comes with sample data and notebooks 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 notebooks in your Notebooks folder.

The following sample datasets are available.

tickit

Most of the examples in the Amazon Redshift documentation use sample data called tickit. 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 tpch is updated with sample data. For more information about the tpch 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 Amazon S3 data into an existing or new table.

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 many of 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.
2. Confirm that you are connected to the target database in the tree-view panel of query editor v2 before continuing. You can create a connection using the context menu (right-click) to the cluster or workgroup where the data will be loaded.

Choose Load data.

3. For Data source, choose Load from S3 bucket.
4. **In S3 URIs**, choose *Browse S3* to look for the Amazon S3 bucket that contains the data to load.

5. If the specified Amazon S3 bucket isn't in the same AWS Region as the target table, then choose the **S3 file location** for the AWS Region where the data is located.

6. Choose *This file is a manifest file* if the Amazon S3 file is actually a manifest containing multiple Amazon S3 bucket URIs.

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.

   If you choose CSV or DELIMITER, you can also choose the **Delimiter character** and whether to **Ignore header rows** if the specified number of rows are actually column names and not data to load.

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. Choose **Next**.

11. Choose **Load existing table**.

12. Confirm or choose the location of the **Target table** including **Cluster or workgroup**, **Database**, **Schema**, and **Table** name where the data is loaded.

13. Choose an **IAM role** that has the required permissions to load data from Amazon S3.

14. (Optional) Choose column names to enter them in **Column mapping** to map columns in the order of the input data file.

15. 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.

When you load data into a new table, query editor v2 first creates the table in the database, then loads the data as separate actions in the same workflow.

### To load data into a new 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 many of 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 you are connected to the target database in the tree-view panel of query editor v2 before continuing. You can create a connection using the context menu (right-click) to the cluster or workgroup where the data will be loaded.

   Choose **Load data**.

2. For **Data source**, choose **Load from S3 bucket**.

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 as the target table, 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 URLs.

6. 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.

If you choose CSV or DELIMITER, you can also choose the **Delimiter character** and whether to **Ignore header rows** if the specified number of rows are actually column names and not data to load.

7. Choose a compression method to compress your file. The default is no compression.

8. (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.

9. Choose **Next**.

10. Choose **Load new table**.

    The table columns are inferred from the input data. You can modify the definition of the table schema by adding columns and table details. To revert to the query editor v2 inferred table schema, choose **Restore to defaults**.

11. Confirm or choose the location of the **Target table** including **Cluster or workgroup**, **Database**, and **Schema** where the data is loaded. Enter a **Table** name to be created.

12. Choose an **IAM role** that has the required permissions to load data from Amazon S3.

13. Choose **Create table** to create the table using the definition shown.

    A review summary of the table definition is displayed. The table is created in the database. To later delete the table, run a DROP TABLE SQL command. For more information, see **DROP TABLE** in the Amazon Redshift Database Developer Guide.

14. 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.

---

**Loading data from a local file setup and workflow**

You can load data from a local file into an existing or new table.

**Administrator setup to load data from a local file**

Your query editor v2 administrator must specify the common Amazon S3 bucket in the **Account settings** window. The account users must be configured with the proper permissions.

- Required IAM permissions – the users of load from local file must have s3:ListBucket, s3:GetBucketLocation, s3:putObject, s3:getObject, and s3:deleteObject permissions. The **optional-prefix** can be specified to limit query editor v2 related use of this bucket to objects with this prefix. You might use this option when using this same Amazon S3 bucket for uses other than query editor v2. For more information about buckets and prefixes, see Managing user access to specific folders in the Amazon Simple Storage Service User Guide. To make sure that cross user data access is not allowed, we recommend that the query editor v2 administrator use an Amazon S3 bucket policy to restrict object access based on aws:userid. The following example allows Amazon S3 permissions to
a $<\text{staging-bucket-name}>$ with read/write access only to Amazon S3 objects with the $\text{aws:userid}$ as a prefix.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::$<\text{staging-bucket-name}>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::$<\text{staging-bucket-name}>[/<\text{optional-prefix}>]/${\text{aws:userid}}/*"
            ]
        }
    ]
}
```

- **Data separation** – we recommend that users not have access to each other’s data (even briefly). Load from a local file uses the staging Amazon S3 bucket set up by the query editor v2 administrator. Configure the bucket policy for the staging bucket to provide data separation between users. The following example shows a bucket policy that separates data between users of the $<\text{staging-bucket-name}>$.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "userIdPolicy",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "NotResource": [
                "arn:aws:s3:::$<\text{staging-bucket-name}>[/<\text{optional-prefix}>]/${\text{aws:userid}}/*"
            ]
        }
    ]
}
```

**Loading data from a local file**

**To load local file data into an existing table**

Your query editor v2 administrator must specify the common Amazon S3 bucket in the **Account settings** window. Query editor v2 automatically uploads the local file to a common Amazon S3 bucket used by your account, and then uses the COPY command to load data. The COPY command generated and run by the query editor v2 load local file window supports many of 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 S3 in the Amazon Redshift Database Developer Guide.

1. Confirm that the table is already created in the database where you want to load data.
2. Confirm that you are connected to the target database in the tree-view panel of query editor v2. You can create a connection using the context menu (right-click) to the cluster or workgroup where the data will be loaded.
3. Choose Load data.
4. For Data source, choose Load from local file.
5. Choose Browse to find the file that contains the data to Load file. By default, files with extension .csv, .avro, .parquet, and .orc are shown, but you can choose other file types. The maximum file size is 5 MB.
6. 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.

   If you choose CSV or DELIMITER, you can also choose the Delimiter character and whether to Ignore header rows if the specified number of rows are actually column names and not data to load.
7. (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.
8. Choose Next.
9. Choose Load existing table.
10. Confirm or choose the location of the Target table including Cluster or workgroup, Database, Schema, and Table name where the data is loaded.
11. (Optional) You can choose column names to enter in Column mapping to map columns in the order of the input data file.
12. Choose Load data to start the data load.

   When the load completes, a message is displayed whether the load was successful or not. 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.

   The COPY command template that was used to load data appears in your Query history. This COPY command template shows some of the parameters used, but it can't be run directly in an editor tab. For more information about query history, see Viewing query and tab history (p. 161).

When you load data into a new table, query editor v2 first creates the table in the database, then loads the data as separate actions in the same workflow.

To load local file data into a new table

Your query editor v2 administrator must specify the common Amazon S3 bucket in the Account settings window. The local file is automatically uploaded to a common Amazon S3 bucket used by your account, and then the COPY command is used by query editor v2 to load data. The COPY command generated and run by the query editor v2 load local file window supports many of 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 S3 in the Amazon Redshift Database Developer Guide.
1. Confirm that you are connected to the target database in the tree-view panel of query editor v2. You can create a connection using the context menu (right-click) to the cluster or workgroup where the data will be loaded.

2. Choose Load data.

3. For Data source, choose Load from local file.

4. Choose Browse to find the file that contains the data to Load file. By default, files with extension .csv, .avro, .parquet, and .orc are shown, but you can choose other file types. The maximum file size is 5 MB.

5. 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.

If you choose CSV or DELIMITER, you can also choose the Delimiter character and whether to Ignore header rows if the specified number of rows are actually column names and not data to load.

6. (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.

7. Choose Next.

8. Choose Load new table.

9. Confirm or choose the location of the Target table including Cluster or workgroup, Database, and Schema where the data is loaded. Enter a Table name to be created.

10. Choose Create table to create the table using the definition shown.

A review summary of the table definition is displayed. The table is created in the database. To later delete the table, run a DROP TABLE SQL command. For more information, see DROP TABLE in the Amazon Redshift Database Developer Guide.

11. Choose Load data to start the data load.

When the load completes, a message displays indicating whether the load was successful or not. 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.

The COPY command template that was used to load data appears in your Query history. This COPY command template shows some of the parameters used, but it can't be run directly in an editor tab. For more information about query history, see Viewing query and tab history (p. 161).

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. Turn on Explain graph for the results to also display a graphical representation of the explain plan.

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 or copy the results of your query by using the context (right-click) menu in the results area as follows:

- Choose Export result set and either JSON or CSV to download the entire set of row results to a file. The number of rows in the result set might be limited by the Limit option or the SQL limit clause in the query. The maximum size of the downloaded result set is 5 MB.
- If no rows are selected, then choose Export current page and either JSON or CSV to download the rows from the current page to a file.
- If rows are selected, then choose Export selected rows and either JSON or CSV to download the rows that are selected to a file.
- If rows are selected, then choose Copy rows to copy the selected rows to the clipboard.
- If rows are selected, then choose Copy rows with headers to copy the selected rows with column headers to the clipboard.

You can also use the shortcut Ctrl+C on Windows or Cmd+C on macOS to copy data from the current results page to the clipboard. If no rows are selected, then the cell with focus is copied to the clipboard. If rows are selected, then the selected rows are copied to the clipboard.

To add a new query tab, choose the + icon, then Editor, which appears in the row with the query tabs. The query tab is either using an Isolated session or not. With an isolated session, the results of a SQL command, such as creating a temporary table in one editor tab, are not visible in another editor tab. When you open an editor tab in query editor v2, the default is an isolated session.

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 or Workgroup, and Database value for the SQL you plan to run.

Initially, you can choose your Cluster or Workgroup in the tree view. Choose your Database in the tree view also.

You can change the Cluster or Workgroup, and Database within each editor tab with the drop-down control located near the Isolated session header of each editor tab.

For each editor tab, you choose whether to run the SQL in an Isolated session. An isolated session has its own connection to a database. Use it to run SQL that is isolated from other query editor sessions. For more information about connections, see Opening query editor v2 (p. 152).

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.
   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 serially and multiple results tabs open for each query.
- You can author queries with session variables and temporary tables.
- You can author queries with replaceable parameters designated by \$\{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. 171).
- 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.

The SQL editor supports the following features:

- The beginning and ending brackets used in SQL have matching colors. Vertical lines are shown in the editor to help you match brackets.
- You can collapse and expand sections of your SQL.
- You can search and replace text in your SQL.
- You can use shortcut keys for several common editing tasks.
- SQL errors are highlighted in the editor for convenient location of problem areas.

For a demo of editing features, watch the following video: New and Enhanced Editing Experience in Amazon Redshift query editor v2.

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. 163).

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. 171)
Example: Top event by total sales (p. 171)
Example: Sales greater than a specific parameter (p. 171)
Example: Create a temporary table (p. 172)
Example: Selecting from a temporary table (p. 172)

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>Stepheinie Meyer</td>
<td>72</td>
<td>41509</td>
</tr>
<tr>
<td>Indigo Girls</td>
<td>57</td>
<td>45399</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></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</th>
<th>qtysold</th>
</tr>
</thead>
<tbody>
<tr>
<td>20005</td>
<td>8</td>
</tr>
<tr>
<td>21279</td>
<td>8</td>
</tr>
<tr>
<td>130232</td>
<td>8</td>
</tr>
<tr>
<td>42737</td>
<td>8</td>
</tr>
<tr>
<td>74681</td>
<td>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>

Authoring and running notebooks

You can use notebooks to organize, annotate, and share multiple SQL queries in a single document. You can add multiple SQL query and Markdown cells to a notebook. 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 notebooks with team members.

To use notebooks, you must add permission for notebooks to your IAM principal (an IAM user or IAM role). As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift. You can add the permission to one of the query editor v2 managed policies. For more information, see Accessing the query editor v2 (p. 149).

You can Run all the cells of a notebook sequentially. The SQL query cell of a notebook has most of the same capabilities as a query editor tab. For more information, see Authoring and running queries (p. 168). The following are differences between a query editor tab and a SQL cell in a notebook.

- There isn't a control to run Explain on a SQL statement in a notebook.
- You can create only one chart per SQL cell in a notebook.
You can export and import notebooks to files created with query editor v2. The file extension is .ipynb and the file size can be a maximum of 5 MB. The SQL and Markdown cells are stored in the file. A cluster or workgroup and database is not stored in the exported notebook. When you open an imported notebook you choose the cluster or workgroup and the database where to run it. After running SQL cells, you can then choose in the results tab whether to display the current page of results as a chart. The result set of a query is not stored in the notebook.

**To create a notebook**

1. From the navigator menu, choose **Editor**.

2. Choose +, and then choose **Notebook**.

   By default, a SQL query cell appears in the notebook.

3. In the SQL query cell, do one of the following:
   - Enter a query.
   - Paste a query that you copied.

4. (Optionally) Choose the + icon, then choose **Markdown** to add a Markdown cell where you can provide descriptive or explanatory text using standard Markdown syntax.

5. (Optionally) Choose the + icon, then choose **SQL** to insert a SQL cell.

You can rename notebooks using the (pencil) icon.

From the (more) menu, you can also perform the following operations on a notebook:

- **Share with my team** – To share the notebook with your team as defined by tags. For more information, see [Sharing a query (p. 191)]

- **Export** – To export the notebook to a local file with the .ipynb extension.

- **Save version** – To create a version of the notebook. To see versions of a notebook, navigate to your saved notebooks and open **Version history**.

- **Duplicate** – To create a copy of the notebook and open it in a new notebook tab.

- **Shortcuts** – To display the shortcuts available when authoring a notebook.

**To open a saved notebook**

1. From the navigator menu, choose **Notebooks**. Your saved notebooks and notebook folders are displayed.
2. Choose the notebook that you want to open and double-click it.

You can show **My notebooks**, notebooks that are **Shared by me**, and notebooks that are **Shared to my team** within the notebooks tab.

To import a notebook from a local file to **My notebooks**, choose ![Import](import.png) then navigate to the .ipynb file that contains your notebook. The notebook is imported into the currently open notebook folder. You can then open the notebook in the notebook editor.

From the context menu (right-click) of a notebook you can perform the following operations:

- **Open notebook** – To open the notebook in the editor.
- **Save version** – To save a version of the notebook.
- **Version history** – To display the versions of a notebook. From the **Version history** window you can delete and revert versions. You can also create a notebook from the currently selected version.
- **Edit tags** – To create and edit tags on a notebook.
- **Share with my team** – To share a notebook with your team.
  
  To share a notebook 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. For an example, see [Permissions required to use the query editor v2](p. 475).
- **Export** – To export a notebook to a local file.
- **Rename** – To rename a notebook.
- **Duplicate** – To make a copy of a notebook.
- **Delete** – To delete a notebook.

For a demo of notebooks, watch the following video: [Amazon Redshift SQL Notebooks in query editor v2](https://example.com).

**Querying the AWS Glue Data Catalog**

You can use query editor v2 to query data cataloged in your AWS Glue Data Catalog. By default, the AWS Glue Data Catalog is listed as a query editor v2 database named `awsdatacatalog`. Querying the AWS Glue Data Catalog is not available in all Amazon Redshift AWS Regions. Use the **SHOW** command to determine if this capability is available.

**Note**

Querying the AWS Glue Data Catalog is only supported in Amazon Redshift RA3 node type clusters and Amazon Redshift Serverless.

You can configure your data warehouse and view the AWS Glue database objects cataloged using the following SQL commands:

- **SHOW** – to display whether `awsdatacatalog` is mounted for the currently connected data warehouse. For example, to show the `data_catalog_auto_mount` parameter value, run:

  ```sql
  SHOW data_catalog_auto_mount;
  ```

  For more information, see [SHOW](https://example.com) in the [Amazon Redshift Database Developer Guide](https://example.com).
- **ALTER SYSTEM** – to change the system-level configuration of `data_catalog_auto_mount`. For example, to change the `data_catalog_auto_mount` parameter value to `on`, run:
ALTER SYSTEM SET data_catalog_auto_mount = on;

The change takes effect when a provisioned cluster is rebooted or a serverless workgroup is automatically paused and resumed. For more information, see ALTER SYSTEM in the Amazon Redshift Database Developer Guide.

- SHOW SCHEMAS – shows a list of schemas. The schemas in the database named awsdatacatalog represent the AWS Glue databases cataloged in the AWS Glue Data Catalog. For example, to show these schemas, run:

  ```sql
  SHOW SCHEMAS FROM DATABASE awsdatacatalog;
  ```

  For more information, see SHOW SCHEMAS in the Amazon Redshift Database Developer Guide.

- SHOW TABLES – shows a list of tables in a schema. For example, to show the tables in the AWS Glue Data Catalog database named awsdatacatalog that are in schema myglue run:

  ```sql
  SHOW TABLES FROM SCHEMA awsdatacatalog.myschema;
  ```

  For more information, see SHOW TABLES in the Amazon Redshift Database Developer Guide.

- SHOW COLUMNS – shows a list of columns in a table. For example, to show the columns in the AWS Glue Data Catalog database named awsdatacatalog that are in schema myglue and table mytable run:

  ```sql
  SHOW COLUMNS FROM TABLE awsdatacatalog.myglue.mytable;
  ```

  For more information, see SHOW COLUMNS in the Amazon Redshift Database Developer Guide.

To grant your IAM user or role permission to query the AWS Glue Data Catalog, follow these steps

1. In the tree-view pane, connect to your initial database in your provisioned cluster or serverless workgroup using the Database user name and password authentication method. For example, connect to the dev database using the admin user and password you used when you created the cluster or workgroup.

2. In an editor tab, run the following SQL statement to grant an IAM user access to the AWS Glue Data Catalog.

  ```sql
  GRANT USAGE ON DATABASE awsdatacatalog to "IAM:myIAMUser"
  ```

  Where IAM:myIAMUser is an IAM user that you want to grant usage privilege to the AWS Glue Data Catalog. Alternatively, you can grant usage privilege to IAMR:myIAMRole for an IAM role.

3. In the tree-view pane, edit or delete the connection to the cluster or workgroup you previously created. Connect to either your cluster or workgroup in one of the following ways:

   - To access the awsdatacatalog database from a cluster, you must use the authentication method Temporary credentials using your IAM identity. For more information about this authentication method, see Connecting to an Amazon Redshift database (p. 154). Your query editor v2 administrator might need to configure the Account settings for the account to display this authentication method on the connection window.

   - To access the awsdatacatalog database from a workgroup, you must use the authentication method Federated user. For more information about this authentication method, see Connecting to an Amazon Redshift database (p. 154).
4. With the granted privilege, you can use your IAM identity to run SQL against your AWS Glue Data Catalog.

After connecting, you can use query editor v2 to query data cataloged in AWS Glue Data Catalog. On the query editor v2 tree-view pane, choose the cluster or workgroup and `awsdatacatalog` database. In the editor or notebook pane, confirm the correct cluster or workgroup is chosen. The database chosen should be the initial Amazon Redshift database such as dev. For information about authoring queries, see Authoring and running queries (p. 168) and Authoring and running notebooks (p. 172). The database named `awsdatacatalog` is reserved to reference the external Data Catalog database in your account. Queries against the `awsdatacatalog` database can only be read-only. Use three-part notation to reference the table in your SELECT statement. Where the first part is the database name, the second part is the AWS Glue database name, and the third part is the AWS Glue table name.

```
SELECT * FROM awsdatacatalog.<aws-glue-db-name>.<aws-glue-table-name>;
```

You can perform various scenarios that read the AWS Glue Data Catalog data and populate Amazon Redshift tables.

The following example SQL joins two tables that are defined in AWS Glue.

```
SELECT pn.emp_id, alias, role, project_name
FROM "awsdatacatalog"."empl_db"."project_name_table" pn,
"awsdatacatalog"."empl_db"."project_alias_table" pa
WHERE pn.emp_id = pa.emp_id;
```

The following example SQL creates an Amazon Redshift table and populates it with data from a join of two AWS Glue tables.

```
CREATE TABLE dev.public.cranberry AS
SELECT pn.emp_id, alias, role, project_name
FROM "awsdatacatalog"."empl_db"."project_name_table" pn,
"awsdatacatalog"."empl_db"."project_alias_table" pa
WHERE pn.emp_id = pa.emp_id;
```

### 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](video).

#### Prerequisites

Before you work with your data lake in query editor v2, confirm the following was set up in your Amazon Redshift environment:

- Crawl your Amazon S3 data using AWS Glue and enable your Data Catalog for AWS Lake Formation.
- Create an IAM role for Amazon Redshift using the AWS Glue enabled Data Catalog for AWS Lake Formation. For details on this procedure, see [To create an IAM role for Amazon Redshift using an AWS Glue Data Catalog enabled for AWS Lake Formation](role). For more information about using Redshift Spectrum and Lake Formation, see [Using Redshift Spectrum with AWS Lake Formation](spectrum).
- Grant SELECT permissions on the table to query in the Lake Formation database. For details on this procedure, see [To grant SELECT permissions on the table to query in the Lake Formation database](permissions).
You can verify in the Lake Formation console (https://console.aws.amazon.com/lakeformation/), Permissions section, Data lake permissions page, that the IAM role, AWS Glue database, and tables have the proper permissions.

- Confirm your connected user has permission to create schemas in the Amazon Redshift database and access data in your data lake. When you connect to a database in query editor v2, you choose an authentication method that includes credentials, which can be a database user or IAM user. The connected user must have the proper permissions and database privileges, such as a superuser. The Amazon Redshift admin user who created the cluster or workgroup has superuser privileges and can create schemas and manage the Redshift database. For more information about connecting to a database with query editor v2, see Connecting to an Amazon Redshift database (p. 154).

Creating an external schema

To query data in an Amazon S3 data lake, first create an external schema. The external schema references the external database in the AWS Glue Data Catalog.

1. In the Editor view of query editor v2, choose Create, and then choose Schema.
2. Enter a Schema name.
3. For Schema type, choose External.
4. Within Data Catalog details, the Region defaults to the AWS Region where your Redshift database is located.
5. Choose the AWS Glue database that the external schema will map to and that contains references to the AWS Glue tables.
6. Choose an IAM role for Amazon Redshift that has the required permissions to query data on Amazon S3.
7. Optionally, choose an IAM role that has permission to the Data Catalog.
8. Choose Create schema.

The schema appears under your database in the tree-view panel.

When creating the schema, if you receive a permission denied error for your database, check if the connected user has the database privilege to create a schema.

Querying data in your Amazon S3 data lake

You use the schema that you created in the previous procedure.

1. In the 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 in the context menu (right-click), choose Select table to generate a query.
4. Run the query in the Editor.

The following example SQL was generated by query editor v2 to query all the rows in AWS Glue table named flightscsv. The columns and rows shown in the output are truncated for simplicity.

```
SELECT * FROM "dev"."mydatalake_schema"."flightscsv";
```
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:

   ```sql
   create datashare mysource;
   ```

3. Set permissions on the datashare. For example:

   ```sql
   grant alter, share on datashare mysource to admin;
   ```

4. Set permissions on the database objects that you want to share. For example:

   ```sql
   alter datashare mysource add schema public;
   alter datashare mysource add table public.event;
   ```

5. Set permissions on the consumer cluster namespace to access the datashare. For example:

   ```sql
   grant usage on datashare mysource 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:

   ```sql
   show datashares;
   ```
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:

   ```sql
   create database share_db from datashare mysoure of account '123456789012' namespace 'p1234567-8765-4321-p10987654321';
   ```

2. Set permissions so that users can access the database and the schema. For example:

   ```sql
   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:

   ```sql
   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, schema, and name of the object.
Scheduling a query

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;
   ```

Scheduling a query with query editor v2

You can create a schedule to run a SQL statement with Amazon Redshift query editor v2. You 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, the query is started by Amazon EventBridge and uses the Amazon Redshift Data API.

**To create a schedule to run a SQL statement**

1. On the **Editor** view, choose **Schedule** to create a schedule to run a SQL statement.
2. When you define the schedule, you provide the following information.
   - The IAM role that assumes the required permissions to run the query. This IAM role is also attached to your cluster or workgroup.
   - The authentication values for either AWS Secrets Manager or temporary credentials to authorize access your cluster or workgroup. These authentication methods are supported by the Data API. For more information, see **Authenticating a scheduled query** (p. 182).
   - The cluster or workgroup where your database resides.
   - The name of the database that contains the data to be queried.
   - The name of the scheduled query and its description. The query editor v2 prefixes the scheduled query name you provide with "QS2-". The query editor v1 prefixes its scheduled query names with "QS-".
   - The SQL statement to be run on the schedule.
   - The schedule frequency and repeat options or a cron formatted value that defines the schedule. For more information, see **Cron Expressions** in the *Amazon CloudWatch Events User Guide*.
   - Optionally, you can enable standard Amazon SNS notifications to monitor the scheduled query. You might need to confirm the email address you provide to the Amazon SNS notification. Check your email for a link to confirm the email address for the Amazon SNS notification. For more information, see **Email notifications** in the *Amazon Simple Notification Service Developer Guide*. If your query is being run but you don’t see messages published in your SNS topic, see **My rule runs, but I don’t see any messages published into my Amazon SNS topic** in the *Amazon EventBridge User Guide*.
3. Choose **Schedule query** to save and activate the schedule and add the schedule to the list of queries in the **Scheduled queries** view.

The **Scheduled queries** view lists all the scheduled queries for your clusters and workgroups. With this view, you can display schedule query details, activate or deactivate the schedule, edit the
Scheduling a query

You can schedule a query to run at a specific time and delete the scheduled query. When you view query details, you can also view the history of running the query with the schedule.

**Note**

A schedule query run is only available in the **Schedule history** list for 24 hours. Queries that run on a schedule don't appear in the **Query history** view of query editor v2.

Setting up permissions to schedule a query

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 with the IAM permissions to use Amazon EventBridge and Amazon Redshift Data API. To receive emails from scheduled queries, the Amazon SNS notification you optionally specify must be configured also.

The following describes the tasks to use AWS managed policies to provide permission, but depending on your environment, you might want to scope down the permissions allowed.

For the IAM user logged into query editor v2, edit the IAM user using the IAM console (https://console.aws.amazon.com/iam/).

- In addition to permissions to run Amazon Redshift and query editor v2 operations, attach the **AmazonEventBridgeFullAccess** and **AmazonRedshiftDataFullAccess** AWS managed policies to an IAM user.
- Alternatively, assign the permissions to a role and assign the role to the user.

Attach a policy that allows the **sts:AssumeRole** permission to the resource ARN of the IAM role you specify when you define the scheduled query. For more information about assuming roles, see **Granting a user permissions to switch roles** in the **IAM User Guide**.

The following example shows a permission policy that assumes the IAM role myRedshiftRole in account 123456789012. The IAM role myRedshiftRole is also the IAM role that is attached to the cluster or workgroup where the scheduled query runs.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeIAMRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": [
        "arn:aws:iam::123456789012:role/myRedshiftRole"
      ]
    }
  ]
}
```

Update the trust policy of the IAM role used to schedule the query to allow the IAM user to assume it.

```json
{
  "Sid": "AssumeRole",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::123456789012:user/myIAMusername"
  },
  "Action": "sts:AssumeRole"
}
```
For the IAM role that you specify to allow the scheduled query to run, edit the IAM role using the IAM console (https://console.aws.amazon.com/iam/).

- Attach the AmazonRedshiftDataFullAccess and AmazonEventBridgeFullAccess AWS managed policies to the IAM role. The AmazonRedshiftDataFullAccess managed policy only allows redshift-serverless:GetCredentials permission for Redshift Serverless workgroups that are tagged with the key RedshiftDataFullAccess.

### Authenticating a scheduled query

When you schedule a query, you use one of the following authentication methods when the SQL runs. Each method requires a different combination of input on the query editor v2. These authentication methods are supported by the Data API which is used to run your SQL statements.

The database user or role that is used to run the query must have the necessary database privileges. For example, to grant IAMR:MyRedshiftQEv2Scheduler privileges to table mytable, run the following SQL command.

```
GRANT all ON TABLE mytable TO "IAMR:MyRedshiftQEv2Scheduler";
```

To view the list of database users in your cluster or workgroup, query the system view PG_USER_INFO.

**Note**

To schedule a query for an Redshift Serverless workgroup, you must choose **Temporary credentials** when you create the schedule. Currently, you can't use AWS Secrets Manager authentication when scheduling a query for Redshift Serverless. In addition, any Redshift Serverless workgroup for which you schedule queries much be tagged with the key RedshiftDataFullAccess. For more information, see [Authorizing access to the Amazon Redshift Data API](p. 356).

As an alternative to tagging the workgroup, you can add an inline policy to the IAM role (that is specified with the schedule) that allows redshift-serverless:GetCredentials. For example:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "UseTemporaryCredentialsForAllServerlessWorkgrgrroups",
            "Effect": "Allow",
            "Action": "redshift-serverless:GetCredentials",
            "Resource": [
                "arn:aws:redshift-serverless:*::*:workgroup/"
            ]
        }
    ]
}
```

### 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 name** and **Database user** values when connecting to a database in a cluster. You only need to provide your **Database name** when connecting to a database in a workgroup.

When connecting to a cluster, 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 attached to your cluster to allow `redshift:GetClusterCredentials`. The following example policy allows database users `awsuser` and `myuser`.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "UseTemporaryCredentialsForAllDbUsers",
         "Effect": "Allow",
         "Action": "redshift:GetClusterCredentials",
         "Resource": [
            "arn:aws:redshift:*:*:dbuser:*/awsuser",
            "arn:aws:redshift:*:*:dbuser:*/myuser"
         ]
      }
   ]
}
```

Setting up permissions to view schedule query history

To allow users to view schedule query history, edit the IAM role (that is specified with the schedule) **Trust relationships** to add permissions.

The following is an example of a trust policy in an IAM role that allows the IAM user `myIAMusername` to view schedule query history. Instead of allowing an IAM user `sts:AssumeRole` permission you can choose to allow an IAM role this permission.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "Service": [
               "redshift.amazonaws.com",
               "redshift-serverless.amazonaws.com"
            ]
         },
         "Action": "sts:AssumeRole"
      },
      {
         "Effect": "Allow",
         "Principal": {
            "Service": "events.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
      },
      {
         "Sid": "AssumeRole",
         "Effect": "Allow",
         "Principal": {
            "Service": [
               "redshift.amazonaws.com",
               "redshift-serverless.amazonaws.com"
            ]
         },
         "Action": "sts:AssumeRole"
      }
   ]
}
```
Monitoring the scheduled query

For the Amazon SNS topic that you specify to send email notifications, create the Amazon SNS topic using the query editor v2 by navigating to the SNS notifications section, Turn on monitoring, and create the topic with Create SNS topic. The query editor v2 creates the Amazon SNS topic and adds a service principal to the access policy for Amazon EventBridge. The following is an example Access policy that is created in the Amazon SNS topic. In the example, the AWS Region us-west-2, AWS account 123456789012, and Amazon SNS topic select-version-pdx-testunload are used.

```json
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "Allow_Publish_Events",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sns:Publish",
    }
  ]
}
```

When the scheduled query runs, Amazon SNS sends AWS notification emails. The following example shows an email sent to myemail@example.com for scheduled query QS2-may25a that ran on AWS Region eu-north-1 in AWS account 123456789012 using Amazon SNS notification topic may25a-SNS.

```json
{"version":"0","id":"8e4323ec-5258-7138-181b-91290e30ff9b","detail-type":"Scheduled Event","source":"aws.events","account":"123456789012","time":"2023-05-25T15:22:00Z","region":"eu-north-1","resources":["arn:aws:events:eu-north-1:123456789012:rule/QS2-may25a"],"detail":{}}
```

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.eu-north-1.amazonaws.com/unsubscribe.html?
SubscriptionArn=arn:aws:sns:eu-north-1:123456789012:may25a-SNS:0c1a3d05-39c2-4507-bc3d-47250513d7b0&Endpoint=myemail@example.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

Troubleshooting set up of scheduling a query

Consider the following if you have issues scheduling a query.

Queries not running

Check if the IAM role used in the schedule has permission to get the temporary cluster credentials. The permission for provisioned clusters is redshift:GetClusterCredentialsWithIAM. The permission for Redshift Serverless workgroups is redshift-serverless:GetCredentials.
Scheduled history not displaying

The IAM user or IAM role used to log in to the AWS console was not added into the trust policy of the IAM role used to schedule the query.

Query history status is Failed

View the SYS_QUERY_HISTORY system view for details about why the query failed. A common issue is that the database user or role that was used to run the query might not have the required privilege to run the SQL. For more information, see Authenticating a scheduled query (p. 182).

The following SQL queries the SYS_QUERY_HISTORY view to return failed queries.

```
SELECT user_id, query_id, transaction_id, session_id, database_name, query_type, status, error_message, query_text
FROM sys_query_history
WHERE status = 'failed';
```

To find out details for a specific failing scheduled query, see Finding details about scheduled queries with AWS CloudShell (p. 185).

Finding details about scheduled queries with AWS CloudShell

You can use AWS CloudShell to find out details about a schedule query. You must have the proper permissions to run the AWS CLI commands shown in the following procedure.

To view the results of a scheduled query

1. On the AWS console, open the AWS CloudShell command prompt. For more information about AWS CloudShell, see What is AWS CloudShell in the AWS CloudShell User Guide.
2. Assume the IAM role of the scheduled query. To assume the role, find the IAM role associated with the scheduled query in query editor v2 and use it in the AWS CLI command in AWS CloudShell. For example, for the role scheduler enter an AWS STS command to assume the role used by the scheduled query.

   ```
   aws sts assume-role --role-arn "arn:aws:iam::123456789012:role/scheduler" --role-session-name "scheduler-test"
   ```

   The credentials returned are similar to the following.

   ```
   "Credentials": {
   "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
   "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
   "SessionToken": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY...",
   "Expiration": "2023-08-18T18:19:44+00:00"
   },
   "AssumedRoleUser": {
   "AssumedRoleId": "AROA35B2NH6WBTP7ONL4E:scheduler-test",
   "Arn": "arn:aws:sts::123456789012:assumed-role/scheduler/scheduler-test"
   }
   ```

3. Create environmental variables in the AWS CLI using the credentials displayed from assuming the IAM role. You must use these tokens before their expiration time. For example, you enter the following in AWS CloudShell.

   ```
   export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
   ```
export AWS_SECRET_ACCESS_KEY=wJalrXUtntFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=je7Mt6bClwBF/2Zp9Ut/k/h3yCo8nvbEXAMPLEKEY...

4. To view the error of a failed query, run the AWS CLI command to describe a statement. The id of the SQL statement is from the ID shown in the Schedule history section of a scheduled query in the query editor v2.

```bash
cmd
aws redshift-data describe-statement --id 130d2620-05d2-439c-b7cf-815d9767f513
```

In this example, the scheduled SQL `select * from users limit 100` results in a SQL error that the `users` table does not exist.

```json
{
  "CreatedAt": "2023-08-18T17:39:15.563000+00:00",
  "Duration": -1,
  "Error": "ERROR: relation "users" does not exist",
  "HasResultSet": false,
  "Id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE111111",
  "QueryString": "select * from users limit 100
—RequestID=a1b2c3d4-5678-90ab-cdef-EXAMPLE22222; TraceID=1-633c5642-4039308d03f3a0ba53dbdf6f",
  "RedshiftPid": 1073766651,
  "RedshiftQueryId": 0,
  "ResultRows": -1,
  "ResultSize": -1,
  "Status": "FAILED",
  "UpdatedAt": "2023-08-18T17:39:16.116000+00:00",
  "WorkgroupName": "default"
}
```

Demo of scheduling a query

For a demo of scheduling a query, watch the following video. **Video demo of scheduling a query.**

**Visualizing query results**

After you run a query and the results display, you can turn on Chart to display a graphic visualization of the current page of 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](https://docs.aws.amazon.com/redshift/latest/dg/sample-database.html) 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**.

![Chart Image]

**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**.
17. Under **Titles**, choose **Y2**.
18. For **Name**, enter **Sales**.
20. Under **Axis Line**, choose **Show** and for **Position**, choose **Right**.

### Revenue and Sales

![Graph showing Revenue and Sales]
Collaborating and sharing as a team

You can share queries with your team.

A team is defined for a set of users who collaborate and share query editor v2 resources. An administrator can create a team by adding a tag to an IAM role. For more information, see Permissions required to use the query editor v2 (p. 475).

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. For an example, see Permissions required to use the query editor v2 (p. 475).

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 a 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:
   • Revert to selected – Revert to the selected version and continue your work with this version.
   • Save selected as – Create a new query in the editor.

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.

Note
You can't query data in Amazon Redshift Serverless using this original query editor. Use Amazon Redshift query editor v2 instead.

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. 474).

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. 194).

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, we recommend that you attach the AmazonRedshiftQueryEditor and AmazonRedshiftReadOnlyAccess AWS managed policies for IAM permissions to the IAM role that you use to access your cluster. Then, you can assign the role to a user. You can use the IAM console (https://console.aws.amazon.com/iam/) to attach IAM policies. For more information, see Using identity-based policies (IAM policies) for Amazon Redshift.

If you have already created a user to access Amazon Redshift, you can attach the AmazonRedshiftQueryEditor and AmazonRedshiftReadOnlyAccess AWS managed policies to that user by means of an assigned role. If you haven't created a user yet, create one and attach the policy to the IAM role and assign the role to the user.

The AWS managed policy AmazonRedshiftQueryEditor allows the action redshift:GetClusterCredentials, which by default gives 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 that denies permission to redshift:GetClusterCredentials. Any user assigned a role with this permission attached must sign in the query editor with temporary credentials. This denial policy illustrates the example.

```json
{
   "Version": "2012-10-17",
   "Statement": {
      "Effect": "Deny",
      "Action": "redshift:GetClusterCredentials",
      "Resource": "*"
   }
}
```
For more information about creating a role with the required permissions, see Create an IAM role with permissions to call GetClusterCredentials (p. 503).

Any user granted access to Amazon Redshift query editor by means of the AWS managed policy AmazonRedshiftQueryEditor can list all secrets. However, this policy allows creation and retrieval of only secrets tagged with the key RedshiftQueryOwner and the value ${aws:userid}. If you create the key from the Amazon Redshift query editor, 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.

The required permissions to use the Amazon Redshift query editor are AmazonRedshiftQueryEditor and AmazonRedshiftReadOnlyAccess.

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center (successor to AWS Single Sign-On):
  
  Create a permission set. Follow the instructions in Create a permission set in the AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide.

- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Creating a role for a third-party identity provider (federation) in the IAM User Guide.

- IAM users:
  
  - Create a role that your user can assume. Follow the instructions in Creating a role for an IAM user in the IAM User Guide.
  
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the IAM User Guide.

**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](https://docs.aws.amazon.com/secretsmanager/latest/userguide/getting-started.html) 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](https://docs.aws.amazon.com/redshift/latest/gsg/creating-cluster.html).

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/](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.

```sql
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.

```sql
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.

```sql
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>
</tbody>
</table>
Scheduling a query

**Important**

The Amazon Redshift query editor v2 now supports scheduling a query. We recommend using query editor v2. For more information, see [Scheduling a query with query editor v2](p. 180).

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**

1. Open the console and query editor as described in [Using the query editor](p. 195). You can only use this query editor with provisioned clusters.
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](p. 196).
- 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. 198).
- 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** 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 to an IAM role and assign the role to the user.

• 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.

```
{
  "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.

```
{
  "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. 477)](#permissions).

• Attach the AmazonRedshiftDataFullAccess AWS managed policy to the IAM role.

• To allow users to view schedule history, edit the IAM role to add the sts:AssumeRole permission.

The following is an example of a trust policy in an IAM role.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": ["events.amazonaws.com"
      ],
      "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": "6a7e8f92-b491-4cf7-a9f1-bf3703467718",
    "detail-type": "Redshift Data Statement Status Change",
    "source": "aws.redshift-data",
}
```
Connecting to a 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.

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, find information about setting up and connecting from SQL client tools, such as SQL Workbench/J, a third-party tool, and Amazon Redshift RSQL. 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 programmatically. 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. 200)
- Configuring security options for connections (p. 303)
- Connecting from client tools and code (p. 308)
Configuring connections in Amazon Redshift

In the following section, learn 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 must have 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. 201), Configuring the Amazon Redshift Python connector (p. 234), and Configuring an ODBC connection (p. 291).

For more information about managing IAM identities, including best practices for IAM roles, see Identity and access management in Amazon Redshift (p. 455).

Finding your cluster connection string

To connect to your cluster with your SQL client tool, you must have 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.

To connect to a cluster endpoint, use the following syntax.
The following is an example of an endpoint URL.

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

If you have set up a custom domain name for your cluster, you can also use that to connect to your cluster. For more information about creating a custom domain name, see Setting up a custom domain name.

### 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. 201)
- Installing the Amazon Redshift JDBC driver, version 2.1 (p. 202)
- Getting the JDBC URL (p. 203)
- Building the connection URL (p. 204)
- Configuring TCP keepalives for your JDBC connection (p. 205)
- Configuring your JDBC connection with Apache Maven (p. 205)
- Configuring authentication and SSL (p. 207)
- Configuring logging (p. 210)
- Converting data types (p. 212)
- Using prepared statement support (p. 212)
- Differences between the 2.1 and 1.x versions of the JDBC driver (p. 213)
- Creating initialization (.ini) files for JDBC driver version 2.1 (p. 213)
- Options for JDBC driver version 2.1 configuration (p. 214)
- Previous versions of JDBC driver version 2.1 (p. 234)

### 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.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.

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. 202)
• Registering the driver class (p. 203)

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.
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;
    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>
</tbody>
</table>
Configuring connections in Amazon Redshift

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<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. 200).

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. 346).

### 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]
```

The following is the format of a connection URL that specifies some optional settings.

```
jdbc:redshift://[Host]:[Port]/[database];[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 database named dev and authenticate the connection using a database user name and password. In this case, you use the following connection URL.

```
jdbc:redshift://redshift.company.us-west-1.redshift.amazonaws.com:9000/dev;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. 214).

**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. 204).

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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.
   ```
Configuring connections in Amazon Redshift

```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, version 2.1 (p. 201).

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>
  <version>driver-version</version>
</dependency>
```

The class name for this driver is `com.amazon.redshift.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>
```
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.

```java
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](https://www.oracle.com/java/technologies/jdk-diagnose-tls-ssl-https.html) 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 Redshift server that you are connecting to.

You must always provide your 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. 204).

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 Redshift user name and password to authenticate the connection.

**To configure authentication using your Redshift user name and password only**

1. Set the `UID` property to your Redshift user name for accessing the Amazon Redshift server.
2. Set the `PWD` property to the password corresponding to your Redshift 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 Redshift user name for accessing the Amazon Redshift server.
2. Set the **PWD** property to the password corresponding to your Redshift 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 Redshift user name for accessing the Amazon Redshift server.
2. Set the **PWD** property to the password corresponding to your Redshift 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:
   - To specify a server certificate, set the **SSLRootCert** property to the full path of the certificate.
   - To specify a TrustStore, do the following:
     * Use the keytool program to add the server certificate to the TrustStore that you want to use.
     * Specify the TrustStore and password to use when starting the Java application using the driver. For example:

       ```
       -Djavax.net.ssl.trustStore=[TrustStoreName]
       -Djavax.net.ssl.trustStorePassword=[TrustStorePassword]
       ```

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. 455).

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><code>jdbc:redshift:iam:// [host]:[port]/[db]</code></td>
<td>A regular connection string. The driver infers the ClusterID and Region from the host.</td>
</tr>
<tr>
<td><code>jdbc:redshift:iam:// [cluster-id]:[region]/[db]</code></td>
<td>The driver retrieves host information, given the ClusterID and Region.</td>
</tr>
</tbody>
</table>
### Connection string

<table>
<thead>
<tr>
<th>Connection string</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jdbc:redshift:iam:// [host]/[db]</td>
<td>The driver defaults to port 5439, and infers ClusterID and Region from the host. Depending on the port you selected when creating, modifying or migrating the cluster, allow access to the selected port.</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. 214).

### 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. 210).

• To send logged information to the LogStream or LogWriter specified in the DriverManager, see [Using LogStream or LogWriter](p. 211).

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. 204).

#### 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>
</tbody>
</table>
### LogLevel value

<table>
<thead>
<tr>
<th>LogLevel</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Log events that might result in an error if action is not taken. This level of logging and the levels of logging above this level also log the user’s queries.</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.
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>
<tr>
<td>VARCHAR</td>
<td>SQL_VARCHAR</td>
<td>String</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. 223). 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. 224).

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. 204).

Topics

- `AccessKeyId` (p. 216)
- `AllowDBUserOverride` (p. 216)
- `App_ID` (p. 217)
- `App_Name` (p. 217)
- `ApplicationName` (p. 217)
- `AuthProfile` (p. 217)
- `AutoCreate` (p. 218)
- `Client_ID` (p. 218)
- `Client_Secret` (p. 218)
- `ClusterID` (p. 218)
- `connectTimeout` (p. 218)
- `databaseMetadataCurrentDbOnly` (p. 219)
- `DbUser` (p. 219)
- `DbGroups` (p. 219)
- `DBNAME` (p. 219)
- `defaultRowFetchSize` (p. 220)
- `DisableIsValidQuery` (p. 220)
- `enableFetchReadAndProcessBuffers` (p. 220)
- `enableFetchRingBuffer` (p. 220)
- `enableMultiSqlSupport` (p. 221)
- `fetchRingBufferSize` (p. 221)
- `ForceLowercase` (p. 221)
- `groupFederation` (p. 222)
- `HOST` (p. 222)
- `IAMDisableCache` (p. 222)
- `IAMDuration` (p. 222)
- `IdP_Host` (p. 223)
- `IdP_Port` (p. 223)
- `IdP_Tenant` (p. 223)
- `IdP_Response_Timeout` (p. 223)
- `IniFile` (p. 223)
- `IniSection` (p. 224)
- `isServerless` (p. 224)
- `Login_URL` (p. 224)
- `loginTimeout` (p. 224)
- `loginToRp` (p. 225)
- `LogLevel` (p. 225)
- `LogPath` (p. 226)
- `OverrideSchemaPatternType` (p. 226)
- `Partner_SPID` (p. 226)
- `Password` (p. 226)
AccessKeyID

- **Default Value** – None
- **Data Type** – String

You can specify this parameter to enter the IAM access key for the 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. If passed in the JDBC URL, `AccessKeyID` must be URL encoded.

This parameter is optional.

AllowDBUserRoleOverride

- **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. The driver attempts to detect this parameter from the given host. If you’re using a Network Load Balancer (NLB) and connecting via IAM, the driver will fail to detect it, so you can set it using this connection option.
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.

If a transaction detects a Statement containing multiple SQL commands separated by semicolons, the fetch ring buffer for that transaction is set to false. enableFetchRingBuffer's value doesn't change.

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 single sign-on 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. Depending on the port you selected when creating, modifying or migrating the cluster, allow access to the selected port.

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\rjdbic.ini"
```
For information about the .ini file, see *Creating initialization (.ini) files for JDBC driver version 2.1* (p. 213).

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. 213).

The following example specifies the [Prod] section of the .ini file:

```ini
IniSection="Prod"
```

This parameter is optional.

**isServerless**

- **Default Value** – false
- **Data Type** – Boolean

This option specifies whether the Amazon Redshift endpoint host is a serverless instance. The driver attempts to detect this parameter from the given host. If you're using a Network Load Balancer (NLB), the driver will fail to detect it, so you can set it here.

This parameter is optional.

**true**

- The Amazon Redshift endpoint host is a serverless instance.

**false**

- The Amazon Redshift endpoint host is a provisioned cluster.

**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.

**OverrideSchemaPatternType**

• **Default Value** – null
• **Data Type** – Integer

This option specifies whether to override the type of query used in getTables calls.

0

No Schema Universal Query

1

Local Schema Query

2

External Schema Query

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:

```plaintext
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. If passed in the JDBC URL, SecretAccessKey must be URL encoded.

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. If passed in the JDBC URL, the temporary IAM session token must be URL encoded.

This parameter is optional.

**serverlessAcctId**

• **Default Value** – null
• **Data Type** – String

The Amazon Redshift Serverless account ID. The driver attempts to detect this parameter from the given host. If you're using a Network Load Balancer (NLB), the driver will fail to detect it, so you can set it here.

This parameter is optional.

**serverlessWorkGroup**

• **Default Value** – null
• **Data Type** – String

The Amazon Redshift Serverless workgroup name. The driver attempts to detect this parameter from the given host. If you're using a Network Load Balancer (NLB), the driver will fail to detect it, so you can set it here.

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.

**SSLError**

• **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 database 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 the previous JDBC 4.2–compatible JDBC driver version 2.1 drivers:

- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.17/redshift-jdbc42-2.1.0.17.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.16/redshift-jdbc42-2.1.0.16.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.15/redshift-jdbc42-2.1.0.15.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.11/redshift-jdbc42-2.1.0.11.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.10/redshift-jdbc42-2.1.0.10.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.9/redshift-jdbc42-2.1.0.9.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.8/redshift-jdbc42-2.1.0.8.zip
- https://s3.amazonaws.com/redshift-downloads/drivers/jdbc/2.1.0.7/redshift-jdbc42-2.1.0.7.zip

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. 455).
- 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

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.

```bash
>>> pip install redshift_connector
```

You can install the connector within a virtual environment. To do this, run the following command.

```bash
>>> pip install redshift_connector
```

Optionally, you can install pandas and NumPy with the connector.

```bash
>>> pip install "redshift_connector[full]"
```

For more information on pip, see the [pip site](https://pip.pypa.io/en/stable/).

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
```
$ . venv/bin/activate
$ python -m pip install -r requirements.txt
$ python -m pip install -e .
$ python -m pip install redshift_connector

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 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.
**group_federation**

- **Default value** – False
- **Data type** – Boolean

This option specifies whether to use Amazon Redshift IDP groups.

This parameter is optional.

true

Use Amazon Redshift Identity Provider (IDP) groups.

false

Use STS API and GetClusterCredentials for user federation and specify `db_groups` for the connection.

**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.

login_url
- **Default value** – None
- **Data type** – String

The single sign-on Url for the IdP.
This parameter is optional.

max_prepared_statements
- **Default value** – 1000
- **Data type** – Integer

The maximum number of prepared statements that can be open concurrently.
This parameter is required.

numeric_to_float
- **Default value** – False
- **Data type** – Boolean

This option specifies if the connector converts numeric data type values from decimal.Decimal to float.
By default, the connector receives numeric data type values as decimal.Decimal and does not convert them.
We don’t recommend enabling numeric_to_float for use cases that require precision, as results may be rounded.

For more information on decimal.Decimal and the tradeoffs between it and float, see decimal — Decimal fixed point and floating point arithmetic on the Python website.

This parameter is optional.

**partner_sp_id**
- **Default value** – None
- **Data type** – String

The Partner SP ID used for authentication with Ping.

This parameter is optional.

**password**
- **Default value** – None
- **Data type** – String

The password to use for authentication.

This parameter is optional.

**port**
- **Default value** – 5439
- **Data type** – Integer

The port number of the Amazon Redshift cluster.

This parameter is required.

**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 user or IAM role for which you are generating a policy. It’s recommended that you attach a policy to a role and then assign the role to your user, for access.

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.

serverless_acct_id

- **Default value** – None
- **Data type** – String

The Amazon Redshift Serverless account ID. This parameter is optional.

serverless_work_group

- **Default value** – None
- **Data type** – String

The Amazon Redshift Serverless workgroup name. 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.

**timeout**

• **Default value** – None
• **Data type** – Integer

The number of seconds before the connection to the server times out.

This parameter is optional.

**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
# Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(
    host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
    port=5439,
    database='dev',
    user='awsuser',
    password='my_password'
)
# Create a Cursor object
>>> cursor = conn.cursor()
# Query and receive result set
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
# Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(
    host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
    port=5439,
```
```
Configuring connections in Amazon Redshift

```python
# Create a Cursor object
>>> cursor = conn.cursor()

# Query and receive result set
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. 511). For more information about managing IAM identities, including best practices for IAM roles, see Identity and access management in Amazon Redshift (p. 455).

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='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 sign-in credentials are provided by the user.
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 JUMP CLOUD 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='',
    login_url='https://sso.jumpcloud.com/saml2/plustwo_melody'
)
```

EXAMPLES OF USING THE AMAZON REDSHIFT PYTHON CONNECTOR

Following are examples of how to use the Amazon Redshift Python connector. To run them, you must first install the Python connector. For more information on installing the Amazon Redshift Python connector, see Installing the Amazon Redshift Python connector (p. 235). For more information on configuration options you can use with the Python connector, see Configuration options for the Amazon Redshift Python connector (p. 236).

Topics
- Connecting to and querying an Amazon Redshift cluster using AWS credentials (p. 248)
- Enabling autocommit (p. 248)
Connecting to and querying an Amazon Redshift cluster using AWS credentials

The following example guides you through connecting to an Amazon Redshift cluster using your AWS credentials, then querying a table and retrieving the query results.

```python
# Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(
    host='examplecluster.abc123xyz789.us-west-1.redshift.amazonaws.com',
    database='dev',
    port=5439,
    user='awsuser',
    password='my_password'
)

# Create a Cursor object
>>> cursor = conn.cursor()

# Query a table using the Cursor
>>> cursor.execute("select * from book")

# Retrieve the query result set
>>> 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 connection's autocommit property after performing a rollback command to make sure that a transaction is not in progress.

```python
# Connect to the cluster
>>> import redshift_connector
>>> conn = redshift_connector.connect(...)

# Run a rollback command
>>> conn.rollback()

# Turn on autocommit
>>> conn.autocommit = True
>>> conn.run("VACUUM")

# Turn off autocommit
>>> conn.autocommit = False
```

Configuring cursor paramstyle

The paramstyle for a cursor can be modified via cursor.paramstyle. The default paramstyle used is format. Valid values for paramstyle are qmark, numeric, named, format, and pyformat.

The following are examples of using various paramstyles to pass parameters to a sample SQL statement.

```python
# qmark
```
Configuring connections in Amazon Redshift

```python
redshift_connector.paramstyle = 'qmark'
sql = 'insert into foo(bar, jar) VALUES(?, ?)'
cursor.execute(sql, (1, "hello world"))

# numeric
redshift_connector.paramstyle = 'numeric'
sql = 'insert into foo(bar, jar) VALUES(:1, :2)'
cursor.execute(sql, (1, "hello world"))

# named
redshift_connector.paramstyle = 'named'
sql = 'insert into foo(bar, jar) VALUES(:p1, :p2)'
cursor.execute(sql, {'p1':1, 'p2':"hello world"})

# format
redshift_connector.paramstyle = 'format'
sql = 'insert into foo(bar, jar) VALUES(%s, %s)'
cursor.execute(sql, (1, "hello world"))

# pyformat
redshift_connector.paramstyle = 'pyformat'
sql = 'insert into foo(bar, jar) VALUES(%(bar)s, %(jar)s)'
cursor.execute(sql, {'bar': 1, 'jar': "hello world"})
```

Using COPY to copy data from an Amazon S3 bucket and UNLOAD to write data to it

The following example shows how to copy data from an Amazon S3 bucket into a table and then unload from that table back into the bucket.

A text file named `category_csv.txt` containing the following data is uploaded to an Amazon 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
# Connect to the cluster and create a Cursor
>>> import redshift_connector
>>> with redshift_connector.connect(...) as conn:
>>>     with conn.cursor() as cursor:

# Create an empty table
>>>     cursor.execute("create table category (catid int, cargroup varchar, catname varchar, catdesc varchar)"

# Use COPY to copy the contents of the S3 bucket into the empty table
>>>     cursor.execute("copy category from 's3://testing/category_csv.txt' iam_role 'arn:aws:iam::123:role/RedshiftCopyUnload' csv;"

# Retrieve the contents of the table
>>>     cursor.execute("select * from category")
>>>     print(cursor.fetchall())

# Use UNLOAD to copy the contents of the table into the S3 bucket
>>>     cursor.execute("unload ('select * from category') to 's3://testing/unloaded_category_csv.txt' iam_role 'arn:aws:iam::123:role/RedshiftCopyUnload' csv;"

# Retrieve the contents of the bucket
>>>     print(cursor.fetchall())
```
If you don’t have autocommit set to true, commit with `conn.commit()` after running the `execute()`
statements.

The data is unloaded into the file `unloaded_category_csv.text0000_part00` in the S3 bucket, with the following content:

```
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.

insert_data_bulk(filename, table_name, parameter_indices, column_names, delimiter, batch_size)

Runs a bulk INSERT statement.

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.

fetchone

Fetches the next row of a query result set.

fetchmany([num])

Fetches the next set of rows of a query result.
fetchall
    Fetches all remaining rows of a query result.
close
    Closes the cursor now.
__iter__
    A cursor object can be iterated to retrieve the rows from a query.
fetch_dataframe([num])
    Returns a dataframe of the last query results.
write_dataframe(df, table)
    Writes the same structure dataframe into an Amazon Redshift database.
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.
get_tables([catalog, schema_pattern, ...])
    Returns the unique public tables which are user-defined within the system.
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.

```python
redshift_connector.plugin.BrowserAzureCredentialsProvider()
```
BrowserSamlCredentialsProvider plugin

Following is the syntax for the BrowserSamlCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```python
redshift_connector.plugin.BrowserSamlCredentialsProvider()
```

OktaCredentialsProvider plugin

Following is the syntax for the OktaCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```python
redshift_connector.plugin.OktaCredentialsProvider()
```

PingCredentialsProvider plugin

Following is the syntax for the PingCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```python
redshift_connector.plugin.PingCredentialsProvider()
```

SamlCredentialsProvider plugin

Following is the syntax for the SamlCredentialsProvider plugin API operation for the Amazon Redshift Python connector.

```python
redshift_connector.plugin.SamlCredentialsProvider()
```

Amazon Redshift integration for Apache Spark

Apache Spark is a distributed processing framework and programming model that helps you do machine learning, stream processing, or graph analytics. Similar to Apache Hadoop, Spark is an open-source, distributed processing system commonly used for big data workloads. Spark has an optimized directed acyclic graph (DAG) execution engine and actively caches data in-memory. This can boost performance, especially for certain algorithms and interactive queries.

This integration provides you with a Spark connector you can use to build Apache Spark applications that read from and write to data in Amazon Redshift and Amazon Redshift Serverless. These applications don't compromise on application performance or transactional consistency of the data. This integration is automatically included in Amazon EMR and AWS Glue, so you can immediately run Apache Spark jobs that access and load data into Amazon Redshift as part of your data ingestion and transformation pipelines.

Currently, you can use the versions 3.3.0 and 3.3.1 of Spark with this integration.

This integration provides the following:

- AWS Identity and Access Management (IAM) authentication. For more information, see Identity and access management in Amazon Redshift.
- Predicate and query pushdown to improve performance.
- Amazon Redshift data types.
• Connectivity to Amazon Redshift and Amazon Redshift Serverless.

Considerations and limitations when using the Spark connector

• The parameter `unload_s3_format` supports the Parquet format when running read operations from Redshift, but write operations are currently unsupported.
• The `tempdir` URI points to an Amazon S3 location. This temp directory is not cleaned up automatically and could add additional cost. We recommend using Amazon S3 lifecycle policies in the Amazon Simple Storage Service User Guide to define the retention rules for the Amazon S3 bucket.
• By default, copies between Amazon S3 and Redshift will not work if the S3 bucket and Redshift cluster are in different AWS Regions. To use separate AWS Regions, set the `tempdir_region` parameter to the Region of the S3 bucket used for the `tempdir`.
• We recommend using Amazon S3 server-side encryption to encrypt the Amazon S3 buckets used.
• We recommend blocking public access to Amazon S3 buckets.
• We recommend that the Amazon Redshift cluster should not be publicly accessible.
• We recommend turning on Amazon Redshift audit logging.
• We recommend turning on Amazon Redshift at-rest encryption.
• We recommend turning on SSL for the JDBC connection from Spark on Amazon EMR to Amazon Redshift.
• We recommend passing an IAM role using the parameter `aws_iam_role` for the Amazon Redshift authentication parameter.

Authentication with the Spark connector

The following diagram describes the authentication between Amazon S3, Amazon Redshift, the Spark driver, and Spark executors.

Authentication between Redshift and Spark

You can use the Amazon Redshift provided JDBC driver version 2 driver to connect to Amazon Redshift with the Spark connector by specifying sign-in credentials. To use IAM, configure your JDBC url to use IAM authentication. To connect to a Redshift cluster from Amazon EMR or AWS Glue, make sure that your IAM role has the necessary permissions to retrieve temporary IAM credentials. The following list describes all of the permissions that your IAM role needs to retrieve credentials and run Amazon S3 operations.

• `Redshift:GetClusterCredentials` (for provisioned Redshift clusters)
• `Redshift:DescribeClusters` (for provisioned Redshift clusters)
• `Redshift:GetWorkgroup` (for Amazon Redshift Serverless workgroups)
• `Redshift:GetCredentials` (for Amazon Redshift Serverless workgroups)
• `s3:ListBucket`
• `s3:GetBucket`
• `s3:GetObject`
• `s3:PutObject`
• `s3:GetBucketLifecycleConfiguration`

For more information about GetClusterCredentials, see Resource policies for GetClusterCredentials.
You also must make sure that Amazon Redshift can assume the IAM role during COPY and UNLOAD operations.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

If you're using the latest JDBC driver, the driver will automatically manage the transition from an Amazon Redshift self-signed certificate to an ACM certificate. However, you must specify the SSL options to the JDBC url.

The following is an example of how to specify the JDBC driver URL and `aws_iam_role` to connect to Amazon Redshift.

```
df.write \
  .format("io.github.spark_redshift_community.spark.redshift ") \ 
  .option("url", "jdbc:redshift:iam://<the-rest-of-the-connection-string>") \ 
  .option("dbtable", "<your-table-name>") \ 
  .option("tempdir", "s3a://<your-bucket>/<your-directory-path>") \ 
  .option("aws_iam_role", "<your-aws-role-arn>") \ 
  .mode("error") \ 
  .save()
```

Authentication between Amazon S3 and Spark

If you're using an IAM role to authenticate between Spark and Amazon S3, use one of the following methods:

- The AWS SDK for Java will automatically attempt to find AWS credentials by using the default credential provider chain implemented by the DefaultAWSCredentialsProviderChain class. For more information, see Using the Default Credential Provider Chain.

- You can specify AWS keys via Hadoop configuration properties. For example, if your `tempdir` configuration points to a s3n:// filesystem, set the `fs.s3n.awsAccessKeyId` and `fs.s3n.awsSecretAccessKey` properties in a Hadoop XML configuration file or call `sc.hadoopConfiguration.set()` to change Spark's global Hadoop configuration.

For example, if you are using the s3n filesystem, add:

```
sc.hadoopConfiguration.set("fs.s3n.awsAccessKeyId", "YOUR_KEY_ID")
sc.hadoopConfiguration.set("fs.s3n.awsSecretAccessKey", "YOUR_SECRET_ACCESS_KEY")
```

For the s3a filesystem, add:

```
sc.hadoopConfiguration.set("fs.s3a.access.key", "YOUR_KEY_ID")
sc.hadoopConfiguration.set("fs.s3a.secret.key", "YOUR_SECRET_ACCESS_KEY")
```

If you're using Python, use the following operations:

```
sc._jsc.hadoopConfiguration().set("fs.s3n.awsAccessKeyId", "YOUR_KEY_ID")
```
Encode authentication keys in the tempdir URL. For example, the URI s3n://ACCESSKEY:SECRETKEY@bucket/path/to/temp/dir encodes the key pair (ACCESSKEY, SECRETKEY).

**Authentication between Redshift and Amazon S3**

If you’re using the COPY and UNLOAD commands in your query, you also must grant Amazon S3 access to Amazon Redshift to run queries on your behalf. To do so, first authorize Amazon Redshift to access other AWS services, then authorize the COPY and UNLOAD operations using IAM roles.

As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift.

**Note**

Acknowledgement: This documentation contains sample code and language developed by the Apache Software Foundation licensed under the Apache 2.0 license.

**Performance improvements with pushdown**

The Spark connector automatically applies predicate and query pushdown to optimize for performance. This support means that if you’re using a supported function in your query, the Spark connector will turn the function into a SQL query and run the query in Amazon Redshift. This optimization results in less data being retrieved, so Apache Spark can process less data and have better performance. By default, pushdown is automatically activated. To deactivate it, set autopushdown to false.

```scala
import sqlContext.implicits._

sample = sqlContext.read
    .format("io.github.spark_redshift_community.spark.redshift")
    .option("url", jdbcURL )
    .option("tempdir", tempS3Dir)
    .option("dbtable", "event")
    .option("autopushdown", "false")
    .load()
```

The following functions are supported with pushdown. If you’re using a function that’s not in this list, the Spark connector will perform the function in Spark instead of Amazon Redshift, resulting in unoptimized performance. For a complete list of functions in Spark, see Built-in Functions.

- Aggregation functions
  - avg
  - count
  - max
  - min
  - sum
- Boolean operators
  - in
  - isnull
  - isnotnull
  - contains
  - endswith
  - startswith
• Logical operators
  • and
  • or
  • not (or !)
• Mathematical functions
  • +
  • -
  • *
  • /
  • -(unary)
  • abs
  • acos
  • asin
  • atan
  • ceil
  • cos
  • exp
  • floor
  • greatest
  • least
  • log10
  • pi
  • pow
  • round
  • sin
  • sqrt
  • tan
• Miscellaneous functions
  • cast
  • coalesce
  • decimal
  • if
  • in
• Relational operators
  • !=
  • ==
  • >
  • >=
  • <
  • <=
• String functions
  • ascii
  • lpad
  • rpad
  • translate
  • upper
• lower
• length
• trim
• ltrim
• rtrim
• like
• substring
• concat
• Time and date functions
  • add_months
  • date
  • date_add
  • date_sub
  • date_trunc
  • timestamp
  • trunc
• Mathematical operations
  • CheckOverflow
  • PromotePrecision
• Relational operations
  • Aliases (for example, AS)
  • CaseWhen
  • Distinct
  • InSet
  • Joins
  • Limits
  • Unions, union all
  • ScalarSubquery
  • Sorts (ascending and descending)
  • UnscaledValue

Other configuration options

Change the maximum size of string columns

Redshift creates string columns as text columns when creating tables, which are stored as VARCHAR(256). If you want columns that support larger sizes, you can use maxlength to specify the maximum length of string columns. The following is an example of how to specify maxlength.

```scala
columnLengthMap.foreach { case (colName, length) =>
  val metadata = new MetadataBuilder().putLong("maxlength", length).build()
  df = df.withColumn(colName, df(colName).as(colName, metadata))
}
```

Setting a column type

To set a column type, use the redshift_type field.

```scala
columnTypeMap.foreach { case (colName, colType) =>
```
val metadata = new MetadataBuilder().putString("redshift_type", colType).build()
df = df.withColumn(colName, df(colName).as(colName, metadata))
}

### Setting a compression encoding on a column

To use a specific compression encoding on a column, use the encoding field. For a full list of support compression encodings, see [Compression encodings](#).

### Setting a description for a column

To set a description, use the `description` field.

### Authentication between Redshift and Amazon S3

By default, the result is unloaded to Amazon S3 in the parquet format. To unload the result as pipe-delimited text file, specify the following option:

```scala
.option("unload_s3_format", "TEXT")
```

### Connector parameters

The parameter map or `OPTIONS` in Spark SQL supports the following settings.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbtable</td>
<td>Yes, unless query is specified</td>
<td>N/A</td>
<td>The table to create or read from in Redshift. This parameter is required when saving data back to Redshift.</td>
</tr>
<tr>
<td>query</td>
<td>Yes, unless dbtable is specified</td>
<td>N/A</td>
<td>The query to read from in Redshift.</td>
</tr>
<tr>
<td>user</td>
<td>No</td>
<td>N/A</td>
<td>The Redshift username. Must be used with the password parameter. Valid only if the user and password are not parameters in the URL. Using both will cause an error.</td>
</tr>
<tr>
<td>password</td>
<td>No</td>
<td>N/A</td>
<td>The Redshift password. Must be used with the user parameter. Valid only if the user and password are not parameters in the URL. Using both will cause an error.</td>
</tr>
<tr>
<td>url</td>
<td>No</td>
<td>N/A</td>
<td>A JDBC URL. The format is jdbc:subprotocol://</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>host:port/database?</td>
<td>Required</td>
<td>N/A</td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>user=username&amp;password= password.</td>
<td>Required</td>
<td>N/A</td>
<td>Subprotocol can be postgresql or Redshift, depending on which JDBC driver you have loaded. Note that one Redshift compatible driver must be in the classpath and match this URL. Host and port should point to the Redshift master node, so you must configure security groups and/or VPC to allow access from your driver application. Database is the Redshift database name. User and password are credentials to access the database, which must be embedded in this URL for JDBC, and the database user must have the necessary permissions to access the table.</td>
</tr>
<tr>
<td>aws_iam_role</td>
<td>Only if using IAM roles to authorize Redshift COPY/UNLOAD operations</td>
<td>N/A</td>
<td>Fully specified ARN of the IAM Role attached to the Redshift cluster.</td>
</tr>
</tbody>
</table>
### Parameter Configuration in Amazon Redshift

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forward_spark_s3_credentials</td>
<td>No</td>
<td>False</td>
<td>Indicates whether this library should automatically discover the credentials that Spark uses to connect to Amazon S3, and whether to forward those credentials to Redshift over the JDBC driver. These credentials are sent as part of the JDBC query. Therefore we recommend you enable SSL encryption with JDBC connection when using this option.</td>
</tr>
<tr>
<td>temporary_aws_access_key_id</td>
<td>No</td>
<td>N/A</td>
<td>AWS access key. Must have write permissions to the S3 bucket.</td>
</tr>
<tr>
<td>temporary_aws_secret_access_key</td>
<td>No</td>
<td>N/A</td>
<td>AWS secret access key corresponding to the access key.</td>
</tr>
<tr>
<td>temporary_aws_session_token</td>
<td>No</td>
<td>N/A</td>
<td>AWS session token corresponding to provided access key.</td>
</tr>
<tr>
<td>tempdir</td>
<td>No</td>
<td>N/A</td>
<td>A writeable location in Amazon S3. Used for unloading data when reading and Avro data to be loaded into Redshift when writing. If you’re using a Redshift data source for Spark as part of a regular ETL pipeline, it can be useful to set a [lifecycle policy] on a bucket and use that as a temp location for this data.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>jdbcdriver</td>
<td>No</td>
<td>Determined by the JDBC URL's subprotocol</td>
<td>The class name of the JDBC driver to use. This class must be on the classpath. In most cases, it should not be necessary to specify this option, as the appropriate driver classname should automatically be determined by the JDBC URL's subprotocol.</td>
</tr>
<tr>
<td>diststyle</td>
<td>No</td>
<td>Even</td>
<td>The Redshift Distribution Style to use when creating a table. Valid options are EVEN, KEY or ALL. When using KEY, you must also set a distribution key with the distkey option.</td>
</tr>
<tr>
<td>distkey</td>
<td>No, unless using DISTSTYLE_KEY</td>
<td>N/A</td>
<td>The name of a column in the table to use as the distribution key when creating a table.</td>
</tr>
<tr>
<td>sortkeyspec</td>
<td>No</td>
<td>N/A</td>
<td>A full Redshift Sort Key definition.</td>
</tr>
<tr>
<td>include_column_list</td>
<td>No</td>
<td>False</td>
<td>Indicates whether this library should automatically extract the columns from the schema and add them to the COPY command according to the Column mapping options.</td>
</tr>
<tr>
<td>description</td>
<td>No</td>
<td>N/A</td>
<td>A description for the table. The description is set with the SQL COMMENT command, and appears in most query tools. See the description metadata to set descriptions on individual columns.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>preactions</td>
<td>No</td>
<td>N/A</td>
<td>A a semicolon-delimited list of SQL commands to be run before loading COPY command. It might be useful to run DELETE commands or similar before loading new data. If the command contains %s, the table name will be formatted in before runtime (in case you're using a staging table). If this command fails, it is treated as an exception. If you're using a staging table, the changes will be reverted and restore the backup table if preactions fail.</td>
</tr>
<tr>
<td>extracopyoptions</td>
<td>No</td>
<td>N/A</td>
<td>A list of extra options to append to the Redshift COPY command when loading data (such as TRUNCATECOLUMNS or MAXERROR n). See Optional parameter for a full list of available parameters. Note that since these options are appended to the end of the COPY command, you can only use options that make sense at the end of the command. That should cover most possible use cases.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------</td>
<td>---------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sse_kms_key</td>
<td>No</td>
<td>N/A</td>
<td>The AWS KMS key ID to use for server-side encryption in S3 during the Redshift UNLOAD operation rather than the AWS default encryption. The Redshift IAM role must have access to the KMS key for writing with it, and the Spark IAM role must have access to the key for read operations. Reading the encrypted data requires no changes (AWS handles this) as long as Spark's IAM role has the proper access.</td>
</tr>
<tr>
<td>tempformat (experimental)</td>
<td>No</td>
<td>AVRO</td>
<td>The format in which to save temporary files in Amazon S3 when writing to Redshift. Valid values are AVRO, CSV, and CSV GZIP. Redshift is faster when loading CSV than when loading Avro files, so using that tempformat might yield better performance when writing to Redshift.</td>
</tr>
<tr>
<td>csvnullstring (experimental)</td>
<td>No</td>
<td>Null</td>
<td>The string value to write for nulls when using the CSV tempformat. This should be a value that does not appear in your actual data.</td>
</tr>
<tr>
<td>autopushdown</td>
<td>No</td>
<td>True</td>
<td>Indicates whether to apply predicate and query pushdown by capturing and analyzing the Spark logical plans for SQL operations. The operations are translated into a SQL query and then run in Redshift to improve performance.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------</td>
<td>---------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>autopushdown.s3_result_cache</td>
<td>No</td>
<td>False</td>
<td>Cache the query SQL to unload data Amazon S3 path mapping in memory, so that the same query doesn't need to run again in the same Spark session. Only supported when autopushdown is turned on. We do not recommend using this parameter when mixing read and write operations because cached results might contain stale information.</td>
</tr>
<tr>
<td>unload_s3_format</td>
<td>No</td>
<td>Parquet</td>
<td>The format with which to unload query results. Valid options are Parquet and Text, which specifies to unload query results in the pipe-delimited text format.</td>
</tr>
<tr>
<td>extraunloadoptions</td>
<td>No</td>
<td>N/A</td>
<td>Extra options to append to the Redshift UNLOAD command. Not all options are guaranteed to work as some options might conflict with other options set within the connector.</td>
</tr>
<tr>
<td>copydelay</td>
<td>No</td>
<td>30000</td>
<td>The delay (in ms) between retries for Redshift COPY operations.</td>
</tr>
<tr>
<td>copyretrycount</td>
<td>No</td>
<td>2</td>
<td>The number of times to retry Redshift COPY operations.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
<td>---------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| tempdir_region | No       | N/A     | The AWS region where tempdir is located. Setting this parameter allows Redshift and tempdir to be in different AWS Regions by automatically supplying the Region during the COPY and UNLOAD operations. It can also improve S3 access performance in some multi-Region scenarios. This setting is recommended in the following situations:  
1) When the connector is running outside of AWS, as automatic Region discovery will fail and negatively affect connector performance.  
2) When tempdir is in a different Region than the Redshift cluster, as using this setting alleviates the need to supply the Region manually using the extracopyoptions and extraunloadoptions parameters.  
3) When the connector is running in a different Region than tempdir, as it improves the connector's access performance of tempdir. |

Note  
Acknowledgement: This documentation contains sample code and language developed by the Apache Software Foundation licensed under the Apache 2.0 license.
The following data types in Amazon Redshift are supported with the Spark connector. For a complete list of supported data types in Amazon Redshift, see Data types. If a data type is not in the table below, it's not supported in the Spark connector.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Aliases</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLINT</td>
<td>INT2</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INT, INT4</td>
</tr>
<tr>
<td>BIGINT</td>
<td>INT8</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>NUMERIC</td>
</tr>
<tr>
<td>REAL</td>
<td>FLOAT4</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>FLOAT8, FLOAT</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BOOL</td>
</tr>
<tr>
<td>CHAR</td>
<td>CHARACTER, NCHAR, BPCHAR</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>CHARACTER VARYING, NVARCHAR, TEXT</td>
</tr>
<tr>
<td>DATE</td>
<td></td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>Timestamp without time zone</td>
</tr>
<tr>
<td>TIMESTAMPTZ</td>
<td>Timestamp with time zone</td>
</tr>
<tr>
<td>TIME</td>
<td>Time without time zone</td>
</tr>
<tr>
<td>TIMETZ</td>
<td>Time with time zone</td>
</tr>
<tr>
<td>VARBYTE</td>
<td>VARBINARY, BINARY VARYING</td>
</tr>
</tbody>
</table>

Configuring a connection for ODBC driver version 2.x for Amazon Redshift

You can use an ODBC connection to connect to your Amazon Redshift cluster from many third-party SQL client tools and applications. If your client tool supports JDBC, you can 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, you can follow the steps in this section to set up an ODBC connection on your client computer or Amazon EC2 instance.

Amazon Redshift provides 64-bit ODBC drivers for Linux and Windows operating systems; the 32-bit ODBC drivers are discontinued. Currently, macOS X is not supported. Further updates to the 32-bit ODBC drivers will not be released, except for urgent security patches. To download and install ODBC drivers for macOS X and 32-bit operating systems, see Configuring an ODBC connection.

For the latest information about ODBC driver changes, see the change log.

Topics
- Getting the ODBC URL (p. 268)
Getting the ODBC URL

Amazon Redshift displays the ODBC URL for your cluster in the Amazon Redshift console. This URL contains the information required to set up the connection between your client computer and the database.

An ODBC URL has the following format:

```
Driver={driver}; Server=endpoint_host; Database=database_name; UID=user_name; PWD=password; Port=port_number
```

The preceding format's fields have the following values:

**ODBC URL field values**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Driver</strong></td>
<td>The name of the 64-bit ODBC driver to use: <a href="amazon-redshift-odbc-driver">Amazon Redshift ODBC Driver (x64)</a></td>
</tr>
<tr>
<td><strong>Server</strong></td>
<td>The endpoint host of the Amazon Redshift cluster.</td>
</tr>
<tr>
<td><strong>Database</strong></td>
<td>The database that you created for your cluster.</td>
</tr>
<tr>
<td><strong>UID</strong></td>
<td>The user name of a database user account that has permission to connect to the database. Although this value is a database-level permission and not a cluster-level permission, you can use the Redshift admin user account that you set up when you launched the cluster.</td>
</tr>
<tr>
<td><strong>PWD</strong></td>
<td>The password for the database user account to connect to the database.</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, ensure that this port is open for you to use.</td>
</tr>
</tbody>
</table>

The following is an example ODBC URL:

```
Driver={Amazon Redshift ODBC Driver (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 on where to find the ODBC URL, see [Finding your cluster connection string](#).
Installing and configuring the Amazon Redshift ODBC driver on Microsoft Windows

System Requirements

You must install the Amazon Redshift ODBC driver on client computers accessing an Amazon Redshift data warehouse. For each computer where you install the driver, there are the following minimum requirements:

- Administrator rights on the machine.
- The machine meets the following system requirements:
  - One of the following operating systems:
    - Windows 10 or 8.1.
  - 100 MB of available disk space.

Installing the Amazon Redshift ODBC driver

Use the following procedure to download and install the Amazon Redshift ODBC driver for Windows operating systems. Only use a different driver if you're running a third-party application that is certified for use with Amazon Redshift, and that application requires that specific driver.

To download and install the ODBC driver:

1. Download the following driver: [64-bit ODBC driver version 2.0.0.8](https://s3.amazonaws.com/redshift-dev/redshift-odbc/x86/AmazonRedshiftODBCDriver_x64.msi)

   The name for this driver is Amazon Redshift ODBC Driver (x64).

   **Note**
   
   32-bit ODBC drivers are discontinued. Further updates will not be released, except for urgent security patches. To download and install ODBC drivers for 32-bit operating systems, see [Install and configure the Amazon Redshift ODBC driver on Microsoft Windows](https://docs.aws.amazon.com/redshift/latest/gsg-install-redshift-odbc-driver.html).

2. Review the Amazon Redshift ODBC driver version 2.x license.

3. Double-click the .msi file, then follow the steps in the wizard to install the driver.

Creating a system DSN entry for an ODBC connection

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 can 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 database user account, and might not be able to detect user DSNs that are created under another database 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](https://docs.aws.amazon.com/redshift/latest/gsg-authenticate-with-iam.html).

To create a system DSN entry for an ODBC connection:

1. In the **Start** menu, type "ODBC Data Sources." Choose **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 following driver folder: Amazon Redshift ODBC Driver (x64).

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 database user account only.


5. Choose the Amazon Redshift ODBC driver (x64), and then choose Finish. The Amazon Redshift ODBC Driver DSN Setup window opens.

6. Under the Connection Settings section, enter the following information:

   **Data source name**
   
Enter a name for the data source. 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 host 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.

   **Port**
   
Enter the port number that the database uses. Depending on the port you selected when creating, modifying or migrating the cluster, allow access to the selected port.

   **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 the Authentication section, specify the configuration options to configure standard or IAM authentication.

8. Choose SSL Options and specify a value for the following:

   **Authentication mode**
   
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.

9. In the Proxy tab, specify any proxy connection setting.

10. In the Cursor tab, specify options on how to return query results to your SQL client tool or application.

11. In Advanced Options, specify values for the logging option and other options.

12. Choose Test. If the client computer can connect to the Amazon Redshift database, the following message appears: Connection successful. If the client computer fails to connect to the database, you can troubleshoot possible issues by generating a log file and contacting AWS support. For information on generating logs, see (LINK).

13. Choose OK.
Installing and configuring the Amazon Redshift ODBC driver on Linux

System Requirements

You must install the Amazon Redshift ODBC driver on client computers accessing an Amazon Redshift data warehouse. For each computer where you install the driver, there are the following minimum requirements:

- Root access on the machine.
- One of the following distributions:
  - Red Hat® Enterprise Linux® (RHEL) 7 or 8
  - CentOS 7 or 8.
- 150 MB of available disk space.
- unixODBC 2.2.14 or later.
- glibc 2.26 or later.

Installing the Amazon Redshift ODBC driver

To download and install the Amazon Redshift ODBC driver version 2.x for Linux:

1. Download the following driver: **64-bit RPM driver version 2.0.0.8**

   **Note**
   
   32-bit ODBC drivers are discontinued. Further updates will not be released, except for urgent security patches.

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:

   ```bash
   yum --nogpgcheck localinstall RPMFileName
   ``

   Replace RPMFileName with the RPM package file name. For example, the following command demonstrates installing the 64-bit driver:

   ```bash
   yum --nogpgcheck localinstall AmazonRedshiftODBC-64-bit-2.x.xx.xxxxx.x86_64.rpm
   ```

Using an ODBC driver manager to configure the ODBC driver on Linux

On Linux, 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.

Configuring the ODBC driver using unixODBC driver manager

The following files are required to configure the Amazon Redshift ODBC driver:

- `amazon.redshiftodbc.ini`
- `odbc.ini`
- `odbcinst.ini`
If you installed to the default location, the `amazon.redshiftodbc.ini` configuration file is located in `/opt/amazon/redshiftodbcx64`.

Additionally, under `/opt/amazon/redshiftodbcx64`, you can find 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 installed 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 will lose any changes that you might have made to files in the installation directory.

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 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 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.

If you are using unixODBC, do the following:

- Set `ODBCINI` to the full path and file name of the `odbc.ini` file.
- Set `ODBCSYSINI` to the full path of the directory that contains the `odbcinst.ini` file.
- Set `AMAZONREDSHIFTODBCINI` to the full path and file name of the `amazon.redshiftodbc.ini` file.

The following is an example of setting the values above:

```bash
export ODBCINI=/usr/local/odbc/odbc.ini
export ODBCSYSINI=/usr/local/odbc
export AMAZONREDSHIFTODBCINI=/etc/amazon.redshiftodbc.ini
```

**Configuring a connection using a data source name (DSN) on Linux**

When connecting to your data store using a data source name (DSN), configure the `odbc.ini` file to define data source names (DSNs). Set the properties in the `odbc.ini` file to create a DSN that specifies the connection information for your data store.

On Linux operating systems, use the following format:

```ini
[ODBC Data Sources]
```

272
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.

[ODBC Data Sources]
Amazon_Redshift_x64=Amazon Redshift ODBC Driver (x64)

[Amazon_Redshift_x64]
Driver=/opt/amazon/redshiftodbcx64/librsodbc64.so
Host=examplecluster.abc123xyz789.us-west-2.redshift.amazonaws.com
Port=5932
Database=dev
locale=en-US

Configuring a connection without a DSN on Linux

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.

On Linux operating systems, use the following format:

[ODBC Drivers]
driver_name=Installed
...

[driver_name]
Description=driver_description
Driver=path/driver_file
...

The following example shows the configuration for odbcinst.ini with the 64-bit ODBC driver on Linux operating systems.

[ODBC Drivers]
Amazon Redshift ODBC Driver (x64)=Installed

[Amazon Redshift ODBC Driver (x64)]
Description=Amazon Redshift ODBC Driver (64-bit)
Driver=/opt/amazon/redshiftodbcx64/librsodbc64.so

Configuring authentication

To protect data from unauthorized access, Amazon Redshift data stores require all connections to be authenticated using user credentials.
The following table illustrates the required and optional connection options for each authentication method that can be used to connect to the Amazon Redshift ODBC driver version 2.x:

### ODBC authentication method required and optional connection options

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Required</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>• Host</td>
<td>• HostID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• EndpointURL</td>
</tr>
<tr>
<td></td>
<td>• Password</td>
<td>• StsEndpointURL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• InstanceProfile</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IAM Profile</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointURL</td>
</tr>
<tr>
<td></td>
<td>• Profile</td>
<td>• StsEndpointURL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• InstanceProfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IAM Credentials</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointURL</td>
</tr>
<tr>
<td></td>
<td>• AccessKeyID</td>
<td>• StsEndpointURL</td>
</tr>
<tr>
<td></td>
<td>• SecretAccessKey</td>
<td>• InstanceProfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AD FS</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• StsEndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• Preferred_Role</td>
</tr>
<tr>
<td></td>
<td>• Password</td>
<td>• loginToRp</td>
</tr>
<tr>
<td></td>
<td>• IdP_Host</td>
<td>• SSL_Insecure</td>
</tr>
<tr>
<td></td>
<td>• IdP_Port</td>
<td></td>
</tr>
</tbody>
</table>

274
<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Required</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Note</strong> ClusterID and Region must be set in Host if they are not set separately.</td>
</tr>
<tr>
<td>Azure AD</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• StsEndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• Preferred_Role</td>
</tr>
<tr>
<td></td>
<td>• Password</td>
<td>• dbgroups_filter</td>
</tr>
<tr>
<td></td>
<td>• IdP_Tenant</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Client_ID</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Client_Secret</td>
<td></td>
</tr>
<tr>
<td>JWT</td>
<td>• Host</td>
<td>• provider_name</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• web_identity_token</td>
<td></td>
</tr>
<tr>
<td>Okta</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• StsEndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• Preferred_Role</td>
</tr>
<tr>
<td></td>
<td>• Password</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• IdP_Host</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• App_Name</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• App_ID</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note</strong> ClusterID and Region must be set in Host if they are not set separately.</td>
</tr>
</tbody>
</table>
## Configuring connections in Amazon Redshift

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Required</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ping Federate</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• StsEndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• Preferred_Role</td>
</tr>
<tr>
<td></td>
<td>• Password</td>
<td>• SSL_Insecure</td>
</tr>
<tr>
<td></td>
<td>• IdP_Host</td>
<td>• partner_spid</td>
</tr>
<tr>
<td></td>
<td>• IdP_Port</td>
<td></td>
</tr>
</tbody>
</table>

**Note**

ClusterID and Region must be set in Host if they are not set separately.

<table>
<thead>
<tr>
<th>Browser Azure AD</th>
<th>• Host</th>
<th>• ClusterID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• StsEndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• IdP_Tenant</td>
<td>• Preferred_Role</td>
</tr>
<tr>
<td></td>
<td>• Client_ID</td>
<td>• dbgroups_filter</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• IdP_Response_Timeout</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• listen_port</td>
</tr>
</tbody>
</table>

**Note**

ClusterID and Region must be set in Host if they are not set separately.

<table>
<thead>
<tr>
<th>Browser SAML</th>
<th>• Host</th>
<th>• ClusterID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• AutoCreate</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• StsEndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• login_url</td>
<td>• Preferred_Role</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td>• dbgroups_filter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IdP_Response_Timeout</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• listen_port</td>
</tr>
</tbody>
</table>

**Note**

ClusterID and Region must be set in Host if they are not set separately.
### Authentication Method

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Required</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auth Profile</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• AccessKeyId</td>
<td>• IdP_Response_Timeout</td>
</tr>
<tr>
<td></td>
<td>• SecretAccessKey</td>
<td>• listen_port</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• scope</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• provider_name</td>
</tr>
<tr>
<td>Browser Azure AD OAUTH2</td>
<td>• Host</td>
<td>• ClusterID</td>
</tr>
<tr>
<td></td>
<td>• Port )</td>
<td>• Region</td>
</tr>
<tr>
<td></td>
<td>• Database</td>
<td>• EndpointUrl</td>
</tr>
<tr>
<td></td>
<td>• IAM</td>
<td>• IdP_Response_Timeout</td>
</tr>
<tr>
<td></td>
<td>• plugin_name</td>
<td>• listen_port</td>
</tr>
<tr>
<td></td>
<td>• IdP_Tenant</td>
<td>• scope</td>
</tr>
<tr>
<td></td>
<td>• Client_ID</td>
<td>• provider_name</td>
</tr>
<tr>
<td></td>
<td>• UID</td>
<td></td>
</tr>
</tbody>
</table>

**Note**

ClusterID and Region must be set in Host if they are not set separately.

### Using an external credentials service

In addition to built-in support for AD FS, Azure AD, and Okta, the Windows version of the Amazon Redshift ODBC driver also provides support for other credentials services. The driver can authenticate connections using any SAML-based credential provider plugin of your choice.

To configure an external credentials service on Windows:

1. Create an IAM profile that specifies the credential provider plugin and other authentication parameters as needed. The profile must be ASCII-encoded, and must contain the following key-value pair, where PluginPath is the full path to the plugin application:

   ```plaintext
   plugin_name = PluginPath
   ```

   For example:

   ```plaintext
   plugin_name = C:\Users\kjson\myapp\CredServiceApp.exe
   ```

   For information on how to create a profile, see [Using a Configuration Profile](#) in the Amazon Redshift Cluster Management Guide.

2. Configure the driver to use this profile. The driver detects and uses the authentication settings specified in the profile.

### Converting data types

The Amazon Redshift ODBC driver version 2.x supports many common data formats, converting between Amazon Redshift and SQL data types.

The following table lists the supported data type mappings.
<table>
<thead>
<tr>
<th>Amazon Redshift type</th>
<th>SQL type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>CHAR</td>
<td>SQL_CHAR</td>
</tr>
<tr>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>SQL_NUMERIC</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>GEOGRAPHY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>GEOMETRY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>REAL</td>
<td>SQL_REAL</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SQL_SMALLINT</td>
</tr>
<tr>
<td>SUPER</td>
<td>SQL_LONGVARCHAR</td>
</tr>
<tr>
<td>TEXT</td>
<td>SQL_LONGVARCHAR</td>
</tr>
<tr>
<td>TIME</td>
<td>SQL_TYPE_TIME</td>
</tr>
<tr>
<td>TIMETZ</td>
<td>SQL_TYPE_TIME</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>TIMESTAMPTZ</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>VARBYTE</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>SQL_VARCHAR</td>
</tr>
</tbody>
</table>

Configuring ODBC driver options

You can use driver configuration options to control the behavior of the Amazon Redshift ODBC driver. Driver options are not case sensitive.

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. 293).

In Linux, 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. 297). 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`.

Following are descriptions for the options that you can specify for the Amazon Redshift ODBC version 2.x driver:
AccessKeyId

- **Default Value** – None
- **Data Type** – String

The IAM access key for the user or role. If you set this parameter, you must also specify `SecretAccessKey`. This parameter is optional.

app_id

- **Default Value** – None
- **Data Type** – String

The Okta-provided unique ID associated with your Amazon Redshift application. This parameter is optional.

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.

AuthProfile

- **Default Value** – None
- **Data Type** – String

The authentication profile used to manage the connection settings. If you set this parameter, you must also set `AccessKeyId` and `SecretAccessKey`. This parameter is optional.

AuthType

- **Default Value** – Standard
- **Data Type** – String

This option specifies the authentication mode that the driver uses when you configure a DSN using the Amazon Redshift ODBC Driver DSN Setup dialog box:

- Standard: Standard authentication using your Amazon Redshift user name and password.
- AWS Profile: IAM authentication using a profile.
- AWS IAM Credentials: IAM authentication using IAM credentials.
- Identity Provider: JWT: IAM authentication using a JSON Web Token (JWT).
- Identity Provider: PingFederate: IAM authentication using PingFederate.
This option is available only when you configure a DSN using the Amazon Redshift ODBC Driver DSN Setup dialog box in the Windows driver. When you configure a connection using a connection string or a non-Windows machine, the driver automatically determines whether to use Standard, AWS Profile, or AWS IAM Credentials authentication based on your specified credentials. To use an identity provider, you must set the `plugin_name` property.

This parameter is required.

**AutoCreate**

- **Default Value** – 0
- **Data Type** – Boolean

A boolean specifying whether the driver creates a new user when the specified user does not exist.

- 1 | TRUE: If the user specified by the `UID` does not exist, the driver creates a new user.
- 0 | FALSE: The driver does not create a new user. If the specified user does not exist, the authentication fails.

This parameter is optional.

**CaFile**

- **Default Value** – None
- **Data Type** – String

The file path to the CA certificate file used for some forms of IAM authentication.

This parameter is only available on Linux.

This parameter is optional.

**client_id**

- **Default Value** – None
- **Data Type** – String

The client ID associated with your Amazon Redshift application in Azure AD.

This parameter is required if authenticating through the Azure AD service.

**client_secret**

- **Default Value** – None
- **Data Type** – String

The secret key associated with your Amazon Redshift application in Azure AD.

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 you want to connect to. It is used in IAM authentication. The Cluster ID is not specified in the Server parameter.

This parameter is optional.

Database

- Default Value – None
- Data Type – String

The name of the Amazon Redshift database that you want to access.

This parameter is required.

DatabaseMetadataCurrentDbOnly

- Default Value – 1
- Data Type – Boolean

A boolean specifying whether the driver returns metadata from multiple databases and clusters.

- 1 | TRUE: The driver only returns metadata from the current database.
- 0 | FALSE. The driver returns metadata across multiple Amazon Redshift databases and clusters.

This parameter is optional.

dbgroups_filter

- Default Value – None
- Data Type – String

The regular expression you can specify to filter out DbGroups that are received from the SAML response to Amazon Redshift when using Azure, Browser Azure, and Browser SAML authentication types.

This parameter is optional.

Driver

- Default Value – Amazon Redshift ODBC Driver (x64)
- Data Type – String

The name of the driver. The only supported value is Amazon Redshift ODBC Driver (x64).

This parameter is required if you do not set DSN.

DSN

- Default Value – None
- Data Type – String

The name of the driver data source name. The application specifies the DSN in the SQLDriverConnect API.

This parameter is required if you do not set Driver.
EndpointUrl

- **Default Value** – None
- **Data Type** – String

The overriding endpoint used to communicate with the Amazon Redshift Coral Service for IAM authentication.

This parameter is optional.

ForceLowercase

- **Default Value** – 0
- **Data Type** – Boolean

A boolean specifying whether the driver lowercases all DbGroups sent from the identity provider to Amazon Redshift when using single sign-on authentication.

- **1 | TRUE**: The driver lowercases all DbGroups that are sent from the identity provider.
- **0 | FALSE**: The driver does not alter DbGroups.

This parameter is optional.

https_proxy_host

- **Default Value** – None
- **Data Type** – String

The host name or IP address of the proxy server through which you want to pass IAM authentication processes.

This parameter is optional.

https_proxy_password

- **Default Value** – None
- **Data Type** – String

The password that you use to access the proxy server. It’s used for IAM authentication.

This parameter is optional.

https_proxy_port

- **Default Value** – None
- **Data Type** – Integer

The number of the port that the proxy server uses to listen for client connections. It's used for IAM authentication.

This parameter is optional.

https_proxy_username

- **Default Value** – None
- **Data Type** – String

  The user name that you use to access the proxy server. It's used for IAM authentication.

  This parameter is optional.

**IAM**

- **Default Value** – 0
- **Data Type** – Boolean

  A boolean specifying whether the driver uses an IAM authentication method to authenticate the connection.

  - 1 | TRUE: The driver uses one of the IAM authentication methods (using an access key and secret key pair, or a profile, or a credentials service).
  - 0 | FALSE: The driver uses standard authentication (using your database user name and password).

  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 parameter is optional.

**idp_port**

- **Default Value** – None
- **Data Type** – Integer

  The port for an IdP (identity provider) you are using to authenticate into Amazon Redshift. Depending on the port you selected when creating, modifying or migrating the cluster, allow access to the selected port.

  This parameter is optional.

**idp_response_timeout**

- **Default Value** – 120
- **Data Type** – Integer

  The number of seconds that the driver waits for the SAML response from the identity provider when using SAML or Azure AD services through a browser plugin.

  This parameter is optional.

**idp_tenant**

- **Default Value** – None
Amazon Redshift Management Guide
Configuring connections in Amazon Redshift

- **Data Type** – String

The Azure AD tenant ID associated with your Amazon Redshift application.
This parameter is required if authenticating through the Azure AD service.

idp_use_https_proxy

- **Default Value** – 0
- **Data Type** – Boolean

A boolean specifying whether the driver passes the authentication processes for identity providers (IdP) through a proxy server.

- 1 | TRUE: The driver passes IdP authentication processes through a proxy server.
- 0 | FALSE: The driver does not pass IdP authentication processes through a proxy server.

This parameter is optional.

InstanceProfile

- **Default Value** – 0
- **Data Type** – Boolean

A boolean specifying whether the driver uses the Amazon EC2 instance profile, when configured to use a profile for authentication.

- 1 | TRUE: The driver uses the Amazon EC2 instance profile.
- 0 | FALSE: The driver uses the chained roles profile specified by the Profile Name option (Profile) instead.

This parameter is optional.

KeepAlive

- **Default Value** – 1
- **Data Type** – Boolean

A boolean specifying whether the driver uses TCP keepalives to prevent connections from timing out.

- 1 | TRUE: The driver uses TCP keepalives to prevent connections from timing out.
- 0 | FALSE: The driver does not use TCP keepalives.

This parameter is optional.

KeepAliveCount

- **Default Value** – 0
- **Data Type** – Integer

The number of TCP keepalive packets that can be lost before the connection is considered broken. When this parameter is set to 0, the driver uses the system default for this setting.
This parameter is optional.

**KeepAliveInterval**

- **Default Value** – 0
- **Data Type** – Integer

The number of seconds between each TCP keepalive retransmission. When this parameter is set to 0, the driver uses the system default for this setting.

This parameter is optional.

**KeepAliveTime**

- **Default Value** – 0
- **Data Type** – Integer

The number of seconds of inactivity before the driver sends a TCP keepalive packet. When this parameter is set to 0, the driver uses the system default for this setting.

This parameter is optional.

**listen_port**

- **Default Value** – 7890
- **Data Type** – Integer

The port that the driver uses to receive the SAML response from the identity provider when using SAML or Azure AD services through a browser plugin.

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 generic Browser SAML plugin.

This parameter is required if authenticating with the SAML or Azure AD services through a browser plugin.

**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 string is optional.

**LogLevel**

- **Default Value** – 0
- **Data Type** – Integer
Use this property to enable or disable logging in the driver and to specify the amount of detail included in log files. We recommend you only enable logging long enough to capture an issue, as logging decreases performance and can consume a large quantity of disk space.

Set the property to one of the following values:

- 0: OFF. Disable all logging.
- 1: ERROR. Logs error events that might allow the driver to continue running but produce an error.
- 2: API_CALL. Logs ODBC API function calls with function argument values.
- 3: INFO. Logs general information that describes the progress of the driver.
- 4: MSG_PROTOCOL. Logs detailed information of the driver's message protocol.
- 5: DEBUG. Logs all driver activity
- 6: DEBUG_APPEND. Keep appending logs for all driver activities.

When logging is enabled, the driver produces the following log files at the location you specify in the LogPath property:

- A redshift_odbc.log.1 file that logs driver activity that takes place during handshake of a connection.
- A redshift_odbc.log file for all driver activities after a connection is made to the database.

This parameter is optional.

**LogPath**

- **Default Value** – The OS-specific TEMP directory
- **Data Type** – String

The full path to the folder where the driver saves log files when LogLevel is higher than 0.

This parameter is optional.

**Min_TLS**

- **Default Value** – The minimum value.
- **Data Type** – String

The minimum version of TLS/SSL that the driver allows the data store to use for encrypting connections. For example, if TLS 1.3 is specified, TLS 1.2 cannot be used to encrypt connections.

Min_TLS accepts the following values:

- 1.2: The connection must use at least TLS 1.2. This is the minimum TLS version required for Amazon Redshift connections.
- 1.3: The connection must use at least TLS 1.3. We recommend TLS 1.3.

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 | PWS**

- **Default Value** – None
- **Data Type** – String

The password corresponding to the database user name that you provided in the User field (UID | User | LogonID).

This parameter is optional.

**plugin_name**

- **Default Value** – None
- **Data Type** – String

The credentials provider plugin name that you want to use for authentication.

The following values are supported:

- **ADFS**: Use Active Directory Federation Services for authentication.
- **AzureAD**: Use Microsoft Azure Active Directory (AD) Service for authentication.
- **BrowserAzureAD**: Use a browser plugin for the Microsoft Azure Active Directory (AD) Service for authentication.
- **BrowserSAML**: Use a browser plugin for SAML services such as Okta or Ping for authentication.
- **JWT**: Use a JSON Web Token (JWT) for authentication.
- **Ping**: Use the PingFederate service for authentication.
- **Okta**: Use the Okta service for authentication.

This parameter is optional.

**Port | PortNumber**

- **Default Value** – 5439
- **Data Type** – Integer

The number of the TCP port that the Amazon Redshift server uses to listen for client connections.

This parameter is optional.

**preferred_role**

- **Default Value** – None
- **Data Type** – String

The role you want to assume during the connection to Amazon Redshift. It’s used for IAM authentication.

This parameter is optional.
Profile

- **Default Value** – None
- **Data Type** – String

The name of the user AWS profile used to authenticate into Amazon Redshift.

- If the Use Instance Profile parameter (the `InstanceProfile` property) is set to 1 | TRUE, that setting takes precedence and the driver uses the Amazon EC2 instance profile instead.
- The default location for the credentials file that contains profiles is `~/.aws/Credentials`. The `AWS_SHARED_CREDENTIALS_FILE` environment variable can be used to point to a different credentials file.

This parameter is optional.

**provider_name**

- **Default Value** – None
- **Data Type** – String

The authentication provider created by the user using the CREATE IDENTITY PROVIDER query. It's used in native Amazon Redshift authentication.

This parameter is optional.

**ProxyHost**

- **Default Value** – None
- **Data Type** – String

The host name or IP address of the proxy server that you want to connect through.

This parameter is optional.

**ProxyPort**

- **Default Value** – None
- **Data Type** – Integer

The number of the port that the proxy server uses to listen for client connections.

This parameter is optional.

**ProxyPwd**

- **Default Value** – None
- **Data Type** – String

The password that you use to access the proxy server.

This parameter is optional.

**ProxyUid**

- **Default Value** – None
• **Data Type** – String

The user name that you use to access the proxy server.

This parameter is optional.

**ReadOnly**

• **Default Value** – 0
• **Data Type** – Boolean

A boolean specifying whether the driver is in read-only mode.

• 1 | TRUE: The connection is in read-only mode, and cannot write to the data store.
• 0 | FALSE: The connection is not in read-only mode, and can write to the data store.

This parameter is optional.

**region**

• **Default Value** – None
• **Data Type** – String

The AWS region that your cluster is in.

This parameter is optional.

**SecretAccessKey**

• **Default Value** – None
• **Data Type** – String

The IAM secret key for the user or role. If you set this parameter, you must also set **AccessKeyID**.

This parameter is optional.

**SessionToken**

• **Default Value** – None
• **Data Type** – String

The temporary IAM session token associated with the IAM role that you are using to authenticate.

This parameter is optional.

**Server | HostName | Host**

• **Default Value** – None
• **Data Type** – String

The endpoint server to connect to.

This parameter is required.
ssl_insecure

- **Default Value** – 0
- **Data Type** – Boolean

A boolean specifying whether the driver checks the authenticity of the IdP server certificate.

- 1 | TRUE: The driver does not check the authenticity of the IdP server certificate.
- 0 | FALSE: The driver checks the authenticity of the IdP server certificate

This parameter is optional.

SSLMode

- **Default Value** – verify-ca
- **Data Type** – String

The SSL certificate verification mode to use when connecting to Amazon Redshift. The following values are possible:

- verify-full: Connect only using SSL, a trusted certificate authority, and a server name that matches the certificate.
- verify-ca: Connect only using SSL and a trusted certificate authority.
- require: Connect only using SSL.
- prefer: Connect using SSL if available. Otherwise, connect without using SSL.
- allow: By default, connect without using SSL. If the server requires SSL connections, then use SSL.
- disable: Connect without using SSL.

This parameter is optional.

StsConnectionTimeout

- **Default Value** – 0
- **Data Type** – Integer

The maximum wait time for IAM connections, in seconds. If set to 0 or not specified, the driver waits 60 seconds for each AWS STS call.

This parameter is optional.

StsEndpointUrl

- **Default Value** – None
- **Data Type** – String

This option specifies the overriding endpoint used to communicate with the AWS Security Token Service (AWS STS).

This parameter is optional.

UID | User | LogonID

- **Default Value** – None
• Data Type – String

The user name that you use to access the Amazon Redshift server.

This parameter is required if you use database authentication.

web_identity_token

• Default Value – None
• Data Type – String

The OAUTH token that is provided by the identity provider. It's used in the JWT plugin.

This parameter is required if you set the plugin_name parameter to BasicJwtCredentialsProvider.

Previous ODBC driver versions

Download a previous version of the Amazon Redshift ODBC driver version 2.x only if your tool requires a specific version of the driver.

Use previous ODBC driver versions for Microsoft Windows

The following are the previous versions of the Amazon Redshift ODBC driver version 2.x for Microsoft Windows:

• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.6/AmazonRedshiftODBC64-2.0.0.6.msi
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.5/AmazonRedshiftODBC64-2.0.0.5.msi
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.3/AmazonRedshiftODBC64-2.0.0.3.msi
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.1/AmazonRedshiftODBC64-2.0.0.1.msi

Use previous ODBC driver versions for Linux

The following are the previous versions of the Amazon Redshift ODBC driver version 2.x for Linux:

• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.7/AmazonRedshiftODBC64-64bit-2.0.0.7.x86_64.rpm
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.6/AmazonRedshiftODBC64-64bit-2.0.0.6.x86_64.rpm
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.5/AmazonRedshiftODBC64-64bit-2.0.0.5.x86_64.rpm
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.3/AmazonRedshiftODBC64-64bit-2.0.0.3.x86_64.rpm
• https://s3.amazonaws.com/redshift-downloads/drivers/odbc/2.0.0.1/AmazonRedshiftODBC64-64bit-2.0.0.1.x86_64.rpm

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 the 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. 292)
- Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 293)
- Install the Amazon Redshift ODBC driver on Linux (p. 295)
- Install the Amazon Redshift ODBC driver on macOS X (p. 297)
- Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 297)
- Configure ODBC driver options (p. 301)
- Previous ODBC driver versions (p. 302)

**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_number
```

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>
<tr>
<td>PWD</td>
<td>The password for the user account to connect to the database.</td>
</tr>
<tr>
<td>Port</td>
<td>The port number that you specified when you launched the database. If you have a firewall, ensure that this port is open for you to use.</td>
</tr>
</tbody>
</table>
The fields in the preceding tables can contain the following special characters:

```
[]{}(),;?*=!@
```

If you use these special characters you must enclose the value in curly braces. For example, the password value `Your;password123` in a connection string is represented as `PWD={Your;password123};`.

Since `Field=value` pairs are separated by semicolon, the combination of `}` and `;` with any number of spaces in between is considered the end of a `Field=value` pair. We recommend you avoid the sequence `{ } ;` in your field values. For example, if you set your password value as `PWD={This is a password} ;d};`, your password would be `This is a password` and the URL would error out.

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. 200)](https://docs.aws.amazon.com/redshift/latest/dg/concepts/configuration.html).

**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 the [Amazon Redshift ODBC connector installation and configuration guide](https://docs.aws.amazon.com/redshift/latest/dg/concepts/odbc-install.html).

**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 1.4.65**
     The name for this driver is Amazon Redshift (x64).
   - **32-bit ODBC driver version 1.4.52**
     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. 506).

For information about how to create a system DSN entry, see the 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.


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 exampleclusterdsn 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. 200).

   **Port**

   Enter the port number that the database uses. Use the port that the cluster was configured to use when it was launched or modified.
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. 346).

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 the 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:
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 1.4.65
   - 64-bit Debian driver version 1.4.65
   - 32-bit RPM driver version 1.4.52
   - 32-bit Debian driver version 1.4.52

   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.
     ```bash
     yum --nogpgcheck localinstall RPMFileName
     
     Replace `RPMFileName` with the RPM package file name. For example, the following command demonstrates installing the 64-bit driver.
     ```
     ```bash
     yum --nogpgcheck localinstall AmazonRedshiftODBC-64-bit-1.x.xx.xxxxx-x.x86_64.rpm
     ```
   - On SLES, run the following command.
     ```bash
     zypper install RPMFileName
     
     Replace `RPMFileName` with the RPM package file name. For example, the following command demonstrates installing the 64-bit driver.
     ```
     ```bash
     zypper install AmazonRedshiftODBC-1.x.x.xxxxx-x.x86_64.rpm
     ```
   - On Debian, run the following command.
     ```bash
     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.
     ```
     ```bash
     sudo apt install ./AmazonRedshiftODBC-1.x.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. 297).

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 minimum system requirements, see the Amazon Redshift ODBC connector installation and configuration guide.

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. If your macOS X system uses Intel architecture, download the macOS X Intel driver version 1.4.65. If your system uses ARM architecture, download the macOS X ARM driver version 1.4.65. In both cases, the name for this driver is Amazon Redshift ODBC driver.

   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. 297).

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. 295) for Linux operating systems and System
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 the 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 the 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.

[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.

[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.

[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.

[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 connections in Amazon Redshift

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 the Amazon Redshift ODBC connector installation and configuration guide.

Use the following format on Linux operating systems.

```
[ODBC Drivers]
driver_name=Installed
...
[driver_name]
Description=driver_description
Driver=path/driver_file
...
```

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.dylib
...
```

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
```
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 the 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 the 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. 293). For an example of setting driver options in a connection string, see Connect to your cluster programmatically (p. 341).

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. 297). 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 the Amazon Redshift ODBC connector installation and configuration guide.
Previous ODBC driver versions

Download a previous version of the Amazon Redshift ODBC driver only if your tool requires a specific version of the driver.

Use previous ODBC driver versions for Windows

The following are the 64-bit drivers:

- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.62.1000/AmazonRedshiftODBC64-1.4.62.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC64-1.4.59.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC64-1.4.56.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.53.1000/AmazonRedshiftODBC64-1.4.53.1000.msi
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC64-1.4.52.1000.msi

32-bit drivers are discontinued and previous versions are not supported.

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/1.4.62.1000/AmazonRedshiftODBC-64-bit-1.4.62.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC-64-bit-1.4.59.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC-64-bit-1.4.59.1000-1.x86_64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-64-bit-1.4.56.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-64-bit-1.4.56.1000-1.x86_64.deb
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-64-bit-1.4.56.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC-64-bit-1.4.52.1000-1.x86_64.rpm
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC-64-bit-1.4.52.1000-1.x86_64.deb

32-bit drivers are discontinued and previous versions are not supported.

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/1.4.62.1000/AmazonRedshiftODBC-1.4.62.1000.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.59.1000/AmazonRedshiftODBC-1.4.59.1000.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.56.1000/AmazonRedshiftODBC-1.4.56.1000.dmg
- https://s3.amazonaws.com/redshift-downloads/drivers/odbc/1.4.52.1000/AmazonRedshiftODBC-1.4.52.1000.dmg
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 418dea9b6d5d5de7a8f1ac42e164cddf.
  • The sha256 checksum number is 36dba8e4b8041cd14b9d60158893963301bcb92e1c456847784de2acb5bd550.

Don't use the previous certificate bundle that was located at https://s3.amazonaws.com/redshift-downloads/redshift-ca-bundle.crt.

• In the China AWS Region, download the bundle from https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/amazon-trust-ca-bundle.crt.
  • The expected MD5 checksum number is 418dea9b6d5d5de7a8f1ac42e164cddf.
  • The sha256 checksum number is 36dba8e4b8041cd14b9d60158893963301bcb92e1c456847784de2acb5bd550.

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 https://s3.cn-north-1.amazonaws.com.cn/redshift-downloads-cn/redshift-ssl-ca-cert.pem

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 Transitioning to ACM certificates for SSL connections (p. 306).

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. 389).
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 on page 291.

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 on page 303.

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>
</tbody>
</table>
Configuring security options for connections

<table>
<thead>
<tr>
<th><code>sslmode</code> (client)</th>
<th><code>require_SSL</code> (server)</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<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. 303).
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. 291).

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

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. 305).

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
     java -jar redshift-keytool.jar -s
     ```
   - To import the Amazon Redshift Certificate Authority bundle into your private TrustStore, run the following command:
     ```java
     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. 307)
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. 291) or Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 201).

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. 303). For more information, see Importing the Amazon Redshift certificate authority bundle into a TrustStore (p. 307).

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. 303). For more information, see Importing the Amazon Redshift certificate authority bundle into a TrustStore (p. 307).
- 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. 303). For more information, see Importing the Amazon Redshift certificate authority bundle into a TrustStore (p. 307).

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
     java -jar redshift-keytool.jar -s
     ```

   - To import the Amazon Redshift Certificate Authority bundle into your private TrustStore, run the following command:

     
     ```java
     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 Amazon Redshift RSQL 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. 303).
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 from client tools and code

Amazon Redshift provides Amazon Redshift query editor v2 to connect to your clusters and workgroups. For more information, see Querying a database using the Amazon Redshift query editor v2 (p. 146).

This section provides some options for third-party tools to connect. Additionally, it describes how to connect to your cluster programmatically.

Topics

- Connecting with Amazon Redshift RSQL (p. 308)
- Connect to a cluster with Amazon Redshift RSQL (p. 312)
- Amazon Redshift RSQL meta commands (p. 316)
- Amazon Redshift RSQL variables (p. 327)
- Amazon Redshift RSQL error codes (p. 332)
- Amazon Redshift RSQL environment variables (p. 340)

Connecting with 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 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, transform, 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>.

Topics
• Getting started with Amazon Redshift RSQL (p. 309)
• Amazon Redshift RSQL change log (p. 310)

Getting started with Amazon Redshift RSQL

Install Amazon Redshift RSQL on a computer with a Linux, macOS, or Microsoft Windows operating system.

Download RSQL
• Linux 64-bit RPM: RSQL Version 1.0.8
• Mac OS 64-bit DMG: RSQL Version 1.0.8
• Windows 64-bit MSI: RSQL Version 1.0.8

See the change log and downloads for previous versions at Amazon Redshift RSQL change log (p. 310).

Install RSQL for Linux

Follow the steps below to install RSQL for Linux.

1. Install the driver manager with the following command:

   ```bash
   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. 295).

3. Copy the ini file to your home directory:

   ```bash
   cp /opt/amazon/redshiftodbc/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/redshiftodbc/Setup
   export AMAZONREDSHIFTODBCINI=/opt/amazon/redshiftodbc/lib/64/amazon.redshiftodbc.ini
   ```

   For more information about configuring ODBC environment variables, see Configuring environment variables (p. 301).
5. You can now install RSQL by running the following command.

```
sudo rpm -i AmazonRedshiftRsql-<version>-1.x86_64.rpm
```

### Install RSQL for Mac

Follow the steps below to install RQL for Mac OSX.

1. Install the driver manager with the following command:

```
brew install unixodbc openssl@1.1 --build-from-source
```

2. Install the ODBC driver: [Install the Amazon Redshift ODBC driver on macOS X (p. 297)](#).

3. Copy the ini file to your home directory:

```
cp /opt/amazon/redshift/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/redshift/Setup
export AMAZONREDSHIFTODBCINI=/opt/amazon/redshift/lib/amazon.redshiftodbc.ini
```

For more information about configuring ODBC environment variables, see [Configuring environment variables (p. 301)](#).

5. Set `DYLD_LIBRARY_PATH` to location of your `libodbc.dylib` if its not in `/usr/local/lib`

```
export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:/usr/local/lib
```

6. Double-click the dmg file to mount the disk image.

7. Double-click the pkg file to run the installer.

8. Follow the steps in the installer to complete the installation. Agree to the terms of the license agreement.

### Install RSQL for Windows

Follow the instructions at [Install and configure the Amazon Redshift ODBC driver on Microsoft Windows (p. 293)](#) 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](#).

Double-click the RSQL download file to run the installer, then follow the prompts to complete the installation.

### Amazon Redshift RSQL change log

1.0.8 (2023-06-19)

**Bug Fixes**

- Fixed an issue where output is truncated with SHOW commands.
- Added support to `\de` for describing external Kinesis streams and Kafka topics.
1.0.7 (2023-03-22)

Bug Fixes

• Fixed an issue where RSQL could not describe materialized views.
• Fixed permission-denied error on stl_connection_log when using Amazon Redshift Serverless.
• Fixed issue where RSQL may process \GOTO labels incorrectly.
• Fixed issue where SSL messages are printed in quiet mode.
• Fixed issue with random characters displayed when describing stored procedures.
• Fixed issue with printing duplicate ERROR/INFO messages.

New

• RSQL now gets SSL information directly from the ODBC driver.

1.0.6 (2023-02-21)

Bug Fixes

• Fixed an issue where \d throws an error - invalid input syntax for integer: "xid" - on Redshift patch 1.0.46086 (P173).

New

• Renamed installation files to reflect supported architecture.

1.0.5 (2022-06-27)

Bug Fixes

• Send SQL error messages to standard error (stderr).
• Fixed issue with exit codes when using ON_ERROR_STOP. Scripts now end after encountering an error and return the correct exit codes.
• Maxerror is now case insensitive.

New

• Added support for ODBC 2.x driver.

1.0.4 (2022-03-19)

• Add support for RSPASSWORD environment variable. Set a password to connect to Amazon Redshift. For example, export RSPASSWORD=TestPassw0rd.

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.
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 1.0.7
- RSQL Version 1.0.6
- RSQL Version 1.0.5
- RSQL Version 1.0.4
- RSQL Version 1.0.3
- RSQL Version 1.0.1

**Mac OS 64-bit DMG**

- RSQL Version 1.0.7
- RSQL Version 1.0.6
- RSQL Version 1.0.5
- RSQL Version 1.0.4
- RSQL Version 1.0.3
- RSQL Version 1.0.1

**Windows 64-bit MSI**

- RSQL Version 1.0.7
- RSQL Version 1.0.6
- RSQL Version 1.0.5
- RSQL Version 1.0.4
- RSQL Version 1.0.3
- 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.

```bash
rsq1 -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.
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. 301). This topic includes instructions for ODBC-driver installation and descriptions for DSN properties. For example, the following section, **Install and configure the Amazon Redshift ODBC driver on Microsoft Windows**, shows how to connect with a DSN using Windows.

**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
boolsaschar=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
```
Using Single sign-on DSN

You can configure a DSN for single sign-on authentication. The following shows an example of a DSN-connection configuration that uses Okta single sign on.

```
[testokta]
Driver=<path to driver>
SSLMode=verify-ca
Min_TLS=1.2
boolsaschar=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
Type "help" for help.
(testcluster) user1@dev=#
```

The following example shows an example of a DSN-connection configuration that uses Azure single sign on.

```
[testazure]
Driver=<path to driver>
SSLMode=verify-ca
Min_TLS=1.2
boolsaschar=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 = wJalrXUtnFEMI/K7MDENG/bPxRfipUMYQyqYkIij5/s7Q==
```

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.
(testcluster) testuser@dev=>
```

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
```
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 preceeded with a backslash.

```
\d[S+]
```

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>
<tr>
<td>(7 rows)</td>
</tr>
</tbody>
</table>

```
\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 "public.sample"
Amazon Redshift Management Guide
Connecting from client tools and code

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Collation</th>
<th>Nullable</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoding</td>
<td>DistKey</td>
<td>SortKey</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| +-+-----------------------------+----------------+----------+---------------+
| col1 | smallint | 1 | NO | none |
| col2 | character(100) | case_sensitive | YES | none |
| col3 | character varying(100) | case_sensitive | YES | text32k |
| col4 | timestamp without time zone | YES | | |
| runlength | f | 0 | |
| col5 | super | YES | zstd |
| col6 | bigint | YES | az64 |

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.

\dC[+] [PATTERN]

Lists casts. Includes the source type, target type, and whether the cast is implicit.

The following shows a subset of results from \dC+

List of casts

<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>
</tbody>
</table>
| +-+-----------------------------+-------------+----------+
| "char" assignment | character | bpchar | in |
| "char" assignment | character varying | text | in |
| "char" | integer | int4 | no |
| "char" | text | text | yes |
| "path" | point | point | no |
| "path" assignment | polygon | polygon | in |
| abstime assignment | date | date | in |
### Connecting from client tools and code

<table>
<thead>
<tr>
<th>abstime</th>
<th>integer</th>
<th>(binary coercible)</th>
<th>no</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstime</td>
<td>time without time zone</td>
<td>time</td>
<td>in</td>
</tr>
<tr>
<td>abstime</td>
<td>timestamp with time zone</td>
<td>timestamp</td>
<td>yes</td>
</tr>
<tr>
<td>abstime</td>
<td>timestamp without time zone</td>
<td>timestamp</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>bit</td>
<td>bit</td>
<td>no</td>
</tr>
<tr>
<td>bigint</td>
<td>boolean</td>
<td>bool</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>character</td>
<td>bpchar</td>
<td>in</td>
</tr>
<tr>
<td>bigint</td>
<td>character varying</td>
<td>text</td>
<td>in</td>
</tr>
<tr>
<td>bigint</td>
<td>double precision</td>
<td>float8</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>integer</td>
<td>int4</td>
<td>in</td>
</tr>
<tr>
<td>bigint</td>
<td>numeric</td>
<td>numeric</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>oid</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>real</td>
<td>float4</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>regclass</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>regoper</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>regoperator</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>regproc</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>regprocedure</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>regtype</td>
<td>oid</td>
<td>yes</td>
</tr>
<tr>
<td>bigint</td>
<td>smallint</td>
<td>int2</td>
<td>in</td>
</tr>
<tr>
<td>bigint</td>
<td>super</td>
<td>int8_partiql</td>
<td>in</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"
```
Connecting from client tools and code

### A Hive Metastore table.

```sql
# \de emr.lineitem
```

### Postgres Federated Table "pgrsql.alltypes"

```sql
# \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>
<tr>
<td>col22</td>
<td>macaddr8</td>
<td>varchar(65535)</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>col23</td>
<td>money</td>
<td>varchar(65535)</td>
<td>21</td>
<td>0</td>
</tr>
<tr>
<td>col24</td>
<td>numeric</td>
<td>numeric(38,20)</td>
<td>22</td>
<td>0</td>
</tr>
<tr>
<td>col25</td>
<td>path</td>
<td>varchar(65535)</td>
<td>23</td>
<td>0</td>
</tr>
<tr>
<td>col26</td>
<td>pg_lsn</td>
<td>varchar(65535)</td>
<td>24</td>
<td>0</td>
</tr>
<tr>
<td>col28</td>
<td>point</td>
<td>varchar(65535)</td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td>col29</td>
<td>polygon</td>
<td>varchar(65535)</td>
<td>26</td>
<td>0</td>
</tr>
<tr>
<td>col30</td>
<td>real</td>
<td>real</td>
<td>27</td>
<td>0</td>
</tr>
<tr>
<td>col31</td>
<td>smallint</td>
<td>smallint</td>
<td>28</td>
<td>0</td>
</tr>
<tr>
<td>col32</td>
<td>smallint</td>
<td>smallint</td>
<td>29</td>
<td>0</td>
</tr>
<tr>
<td>col33</td>
<td>integer</td>
<td>integer</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>col34</td>
<td>text</td>
<td>varchar(65535)</td>
<td>31</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.

`\dL[S+] [PATTERN]`

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, `\dLS+`, 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:

<table>
<thead>
<tr>
<th>List of languages</th>
<th>name</th>
<th>trusted</th>
<th>internal language</th>
<th>call handler</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>name</td>
<td>trusted</td>
<td>internal language</td>
<td>call handler</td>
<td>description</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>f</td>
<td>t</td>
<td></td>
<td>fmgr_c_validator(oid)</td>
</tr>
<tr>
<td></td>
<td>exfunc</td>
<td>f</td>
<td>t</td>
<td></td>
<td>Dynamically-loaded C functions</td>
</tr>
<tr>
<td></td>
<td>internal</td>
<td>f</td>
<td>t</td>
<td></td>
<td>fmgr_internal_validator(oid)</td>
</tr>
<tr>
<td></td>
<td>mlfunc</td>
<td>f</td>
<td>f</td>
<td>mlfunc_call_handler()</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>plpgsql</td>
<td>f</td>
<td>f</td>
<td>plpgsql_call_handler()</td>
<td>rdsdb=U/rdsdb</td>
</tr>
<tr>
<td></td>
<td>plpythonu</td>
<td>f</td>
<td>f</td>
<td>plpython_call_handler()</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>sql</td>
<td>f</td>
<td>t</td>
<td></td>
<td>=U/rdsdb</td>
</tr>
<tr>
<td></td>
<td>sql</td>
<td>t</td>
<td>t</td>
<td></td>
<td>SQL-</td>
</tr>
</tbody>
</table>

Alternative format output:
\dm[S+] [PATTERN]
Lists materialized views. For example, \dmS+ 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 \dnS+, 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 \dtS+, 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 \dvS+, 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.

\l[+] [PATTERN]
Lists databases. Includes owner, encoding, and additional information.

\q
The quit, or \q command, logs off database sessions and exits RSQl.

\sv[S+] VIEWNAME
Shows a view's definition.

\timing
Shows the run time, for a query, for instance.

\z [PATTERN]
The same output as \dp.
\?

Shows help information. The optional parameter specifies the item to explain.

\EXIT

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.

```bash
% 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
```

\EXPORT

Specifies the name of an export file that RSQL uses to store database information returned by a subsequent SQL SELECT statement.

```bash
export_01.sql
```

```bash
\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.
Target width is 1500.
Heading is set to: General Title
Titledashes is on.
(exported 40 rows)
```

\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| - USENAME| - HOST| - PORT| - [PASSWROD]]] | conninfo\}

The DBNAME is the name of the database to connect to. The USENAME 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.

```sql
(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`.

```sql
(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.

```sql
(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`
- `HEADING` and `RTITLE`
- `RFORMAT`
- `MAXERROR`
- `TITLEDASHES`
- `WIDTH`

The following example specifies a heading.

```sql
\rset heading "Winter Sales Report"
```
For more examples of how to use `\iset`, you can find several in the Amazon Redshift RSQL variables (p. 327) 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=# \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=# \run file=test.sql
lineitem_cnt
---------------
  4307207
(1 row)
customer_cnt
---------------
  37796166
(1 row)
orders_cnt
------------
  0
(1 row)

(testcluster) user1@dev=# \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`.

```
(testcluster) user1@dev=# \os date
Tue Sep 7 20:47:54 UTC 2021
```

\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.
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=# \! 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=# \i test.sql
<table>
<thead>
<tr>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>4307207</td>
</tr>
</tbody>
</table>
(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.

```
(testcluster) user1@dev=# \! cat test.sql
SELECT query FROM stv_inflight LIMIT 1 \gset
select :query as query;
```
Amazon Redshift Management Guide
Connecting from client tools and code

```sql
\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**

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:

```
\set 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...
\set rtitle "Marketing Department||Confidential//Third Quarter//Chicago"
\set width 70
\set 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`.

```
\rset maxerror 0
select 1 as col;
\quit
```

Console output:

```
Maxerror is default.
(1 row)
col
1
```

**RFORMAT**

Enables users to specify whether to apply settings for the formatting commands.

```
\remark Starting...
\psset border 2
\psset format wrapped
\psset expanded on
\psset 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...
Border style is 2. (Changes will take effect after RFORMAT is switched ON)
Output format is wrapped. (Changes will take effect after RFORMAT is switched ON)
Expanded display is on. (Changes will take effect after RFORMAT is switched ON)
Title is "Great Title". (Changes will take effect after RFORMAT is switched ON)
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
```
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:

```
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:

```
\set titledashes on
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:

```
department     employee      salary
--------------------------
```
<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:

```
\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:

```
\echo Starting...
\rset rformat on
\pset format wrapped
\rset 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.
id | long_string
500 | In general, the higher the number the more borders and lines the ta.
| .bles will have, but details depend on the particular format.
(1 row)
```

```
Target width is 50.
id | long_string
500 | In general, the higher the number the more.
| . borders and lines the tables will have, b.
| .ut details depend on the particular format.
```

(1 row)

Press any key to continue . . .
## Amazon Redshift RSQL error codes

Success messages, warnings, and exceptions:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Class</th>
<th>Condition Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>Class 00 — Successful Completion</td>
<td>successful_completion</td>
</tr>
<tr>
<td>01000</td>
<td>Class 01 — Warning</td>
<td>warning</td>
</tr>
<tr>
<td>0100C</td>
<td>Class 01 — Warning</td>
<td>dynamic_result_sets_returned</td>
</tr>
<tr>
<td>01008</td>
<td>Class 01 — Warning</td>
<td>implicit_zero_bit_padding</td>
</tr>
<tr>
<td>01003</td>
<td>Class 01 — Warning</td>
<td>null_value_eliminated_in_set_function</td>
</tr>
<tr>
<td>01007</td>
<td>Class 01 — Warning</td>
<td>privilege_not_granted</td>
</tr>
<tr>
<td>01006</td>
<td>Class 01 — Warning</td>
<td>privilege_not_revoked</td>
</tr>
<tr>
<td>01004</td>
<td>Class 01 — Warning</td>
<td>string_data_right_truncation</td>
</tr>
<tr>
<td>01P01</td>
<td>Class 01 — Warning</td>
<td>deprecated_feature</td>
</tr>
<tr>
<td>02000</td>
<td>Class 02 — No Data</td>
<td>no_data</td>
</tr>
<tr>
<td>02001</td>
<td>Class 02 — No Data</td>
<td>no_additional_dynamic_result_sets_returned</td>
</tr>
<tr>
<td>03000</td>
<td>Class 03 — SQL Statement Not Yet Complete</td>
<td>sql_statement_not_yet_complete</td>
</tr>
<tr>
<td>08000</td>
<td>Class 08 — Connection Exception</td>
<td>connection_exception</td>
</tr>
<tr>
<td>08003</td>
<td>Class 08 — Connection Exception</td>
<td>connection_does_not_exist</td>
</tr>
<tr>
<td>08006</td>
<td>Class 08 — Connection Exception</td>
<td>connection_failure</td>
</tr>
<tr>
<td>08001</td>
<td>Class 08 — Connection Exception</td>
<td>sqlclient_unable_to_establish_sqlconnection</td>
</tr>
<tr>
<td>08004</td>
<td>Class 08 — Connection Exception</td>
<td>sqlserver_rejected_establishment_of_sqlconnection</td>
</tr>
<tr>
<td>08007</td>
<td>Class 08 — Connection Exception</td>
<td>transaction_resolution_unknown</td>
</tr>
<tr>
<td>08P01</td>
<td>Class 08 — Connection Exception</td>
<td>protocol_violation</td>
</tr>
<tr>
<td>09000</td>
<td>Class 09 — Triggered Action Exception</td>
<td>triggered_action_exception</td>
</tr>
<tr>
<td>0A000</td>
<td>Class 0A — Feature Not Supported</td>
<td>feature_not_supported</td>
</tr>
<tr>
<td>0A000</td>
<td>Class 0A — Feature Not Supported</td>
<td>feature_not_supported</td>
</tr>
</tbody>
</table>
### Error Codes and Conditions

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Class</th>
<th>Condition Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0B000</td>
<td>Class 0B — Invalid Transaction Initiation</td>
<td>invalid_transaction_initiation</td>
</tr>
<tr>
<td>0F000</td>
<td>Class 0F — Locator Exception</td>
<td>locator_exception</td>
</tr>
<tr>
<td>0F001</td>
<td>Class 0F — Locator Exception</td>
<td>invalid_locator_specification</td>
</tr>
<tr>
<td>0L000</td>
<td>Class 0L — Invalid Grantor</td>
<td>invalid_grantor</td>
</tr>
<tr>
<td>0LP01</td>
<td>Class 0L — Invalid Grantor</td>
<td>invalid_grant_operation</td>
</tr>
<tr>
<td>0P000</td>
<td>Class 0P — Invalid Role Specification</td>
<td>invalid_role_specification</td>
</tr>
<tr>
<td>0Z000</td>
<td>Class 0Z — Diagnostics Exception</td>
<td>diagnostics_exception</td>
</tr>
<tr>
<td>0Z002</td>
<td>Class 0Z — Diagnostics Exception</td>
<td>stacked_diagnostics_accessed_without_active_handler</td>
</tr>
<tr>
<td>20000</td>
<td>Class 20 — Case Not Found</td>
<td>case_not_found</td>
</tr>
<tr>
<td>21000</td>
<td>Class 21 — Cardinality Violation</td>
<td>cardinality_violation</td>
</tr>
</tbody>
</table>

### Data Exceptions:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Class</th>
<th>Condition Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22000</td>
<td>Class 22 — Data Exception</td>
<td>data_exception</td>
</tr>
<tr>
<td>2202E</td>
<td>Class 22 — Data Exception</td>
<td>array_subscript_error</td>
</tr>
<tr>
<td>22021</td>
<td>Class 22 — Data Exception</td>
<td>character_not_in_repertoire</td>
</tr>
<tr>
<td>22008</td>
<td>Class 22 — Data Exception</td>
<td>datetime_field_overflow</td>
</tr>
<tr>
<td>22012</td>
<td>Class 22 — Data Exception</td>
<td>division_by_zero</td>
</tr>
<tr>
<td>22005</td>
<td>Class 01 — Warning</td>
<td>error_in_assignment</td>
</tr>
<tr>
<td>2200B</td>
<td>Class 01 — Warning</td>
<td>escape_character_conflict</td>
</tr>
<tr>
<td>22022</td>
<td>Class 01 — Warning</td>
<td>indicator_overflow</td>
</tr>
<tr>
<td>22015</td>
<td>Class 01 — Warning</td>
<td>interval_field_overflow</td>
</tr>
<tr>
<td>2201E</td>
<td>Class 01 — Warning</td>
<td>invalid_argument_for_logarithm</td>
</tr>
<tr>
<td>2201F</td>
<td>Class 01 — Warning</td>
<td>invalid_argument_for_power_function</td>
</tr>
<tr>
<td>2201G</td>
<td>Class 01 — Warning</td>
<td>invalid_argument_for_width_bucket_function</td>
</tr>
<tr>
<td>22018</td>
<td>Class 01 — Warning</td>
<td>invalid_character_value_for_cast</td>
</tr>
<tr>
<td>22007</td>
<td>Class 01 — Warning</td>
<td>invalid_datetime_format</td>
</tr>
<tr>
<td>22019</td>
<td>Class 01 — Warning</td>
<td>invalid_escape_character</td>
</tr>
<tr>
<td>2200D</td>
<td>Class 01 — Warning</td>
<td>invalid_escape_octet</td>
</tr>
</tbody>
</table>
### Error Codes and Condition Names

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Class</th>
<th>Condition Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22025</td>
<td>Class 01 — Warning</td>
<td>invalid_escape_sequence</td>
</tr>
<tr>
<td>22P06</td>
<td>Class 01 — Warning</td>
<td>nonstandard_use_of_escape_character</td>
</tr>
<tr>
<td>22010</td>
<td>Class 01 — Warning</td>
<td>invalid_indicator_parameter_value</td>
</tr>
<tr>
<td>22023</td>
<td>Class 01 — Warning</td>
<td>invalid_parameter_value</td>
</tr>
<tr>
<td>2201B</td>
<td>Class 01 — Warning</td>
<td>invalid_regular_expression</td>
</tr>
<tr>
<td>22009</td>
<td>Class 01 — Warning</td>
<td>invalid_time_zone_displacement_value</td>
</tr>
<tr>
<td>2200C</td>
<td>Class 01 — Warning</td>
<td>invalid_use_of_escape_character</td>
</tr>
<tr>
<td>2200G</td>
<td>Class 01 — Warning</td>
<td>most_specific_type_mismatch</td>
</tr>
<tr>
<td>22004</td>
<td>Class 01 — Warning</td>
<td>null_value_not_allowed</td>
</tr>
<tr>
<td>22002</td>
<td>Class 01 — Warning</td>
<td>null_value_no_indicator_parameter</td>
</tr>
<tr>
<td>22003</td>
<td>Class 01 — Warning</td>
<td>numeric_value_out_of_range</td>
</tr>
<tr>
<td>22026</td>
<td>Class 01 — Warning</td>
<td>string_data_length_mismatch</td>
</tr>
<tr>
<td>22011</td>
<td>Class 01 — Warning</td>
<td>string_data_right_truncation</td>
</tr>
<tr>
<td>22027</td>
<td>Class 01 — Warning</td>
<td>substring_error</td>
</tr>
<tr>
<td>22024</td>
<td>Class 01 — Warning</td>
<td>unterminated_c_string</td>
</tr>
<tr>
<td>2200F</td>
<td>Class 01 — Warning</td>
<td>zero_length_character_string</td>
</tr>
<tr>
<td>22P01</td>
<td>Class 01 — Warning</td>
<td>floating_point_exception</td>
</tr>
<tr>
<td>22P02</td>
<td>Class 01 — Warning</td>
<td>invalid_text_representation</td>
</tr>
<tr>
<td>22P03</td>
<td>Class 01 — Warning</td>
<td>invalid_binary_representation</td>
</tr>
<tr>
<td>22P04</td>
<td>Class 01 — Warning</td>
<td>bad_copy_file_format</td>
</tr>
<tr>
<td>22P05</td>
<td>Class 01 — Warning</td>
<td>untranslatable_character</td>
</tr>
</tbody>
</table>

### Integrity Constraint Violations:

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Class</th>
<th>Condition Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>23000</td>
<td>Class 23 — Integrity Constraint Violation</td>
<td>integrity_constraint_violation</td>
</tr>
<tr>
<td>23001</td>
<td>Class 23 — Integrity Constraint Violation</td>
<td>restrict_violation</td>
</tr>
<tr>
<td>23502</td>
<td>Class 23 — Integrity Constraint Violation</td>
<td>not_null_violation</td>
</tr>
<tr>
<td>23503</td>
<td>Class 23 — Integrity Constraint Violation</td>
<td>foreign_key_violation</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Class</td>
<td>Condition Name</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>----------------</td>
</tr>
<tr>
<td>23505</td>
<td>Class 23 — Integrity Constraint Violation</td>
<td>uniqueViolation</td>
</tr>
<tr>
<td>23514</td>
<td>Class 23 — Integrity Constraint Violation</td>
<td>checkViolation</td>
</tr>
<tr>
<td>24000</td>
<td>Class 24 — Invalid Cursor State</td>
<td>invalidCursorState</td>
</tr>
<tr>
<td>01004</td>
<td>Class 01 — Warning</td>
<td>string_data_right_truncation</td>
</tr>
<tr>
<td>25000</td>
<td>Class 25 — Invalid Transaction State</td>
<td>invalidTransactionState</td>
</tr>
<tr>
<td>25001</td>
<td>Class 25 — Invalid Transaction State</td>
<td>activeSqlTransaction</td>
</tr>
<tr>
<td>25002</td>
<td>Class 25 — Invalid Transaction State</td>
<td>invalidTransactionState</td>
</tr>
<tr>
<td>25003</td>
<td>Class 25 — Invalid Transaction State</td>
<td>inappropriate_access_mode_for_branch_transaction</td>
</tr>
<tr>
<td>25004</td>
<td>Class 25 — Invalid Transaction State</td>
<td>inappropriate_isolation_level_for_branch_transaction</td>
</tr>
<tr>
<td>25005</td>
<td>Class 25 — Invalid Transaction State</td>
<td>no_active_sql_transaction_for_branch_transaction</td>
</tr>
<tr>
<td>25006</td>
<td>Class 25 — Invalid Transaction State</td>
<td>read_only_sql_transaction</td>
</tr>
<tr>
<td>25007</td>
<td>Class 25 — Invalid Transaction State</td>
<td>no_active_sql_transaction_for_branch_transaction</td>
</tr>
<tr>
<td>25008</td>
<td>Class 25 — Invalid Transaction State</td>
<td>held_cursor_reQUIRES_same_isolation_level</td>
</tr>
<tr>
<td>25009</td>
<td>Class 25 — Invalid Transaction State</td>
<td>no_active_sql_transaction_for_branch_transaction</td>
</tr>
<tr>
<td>25010</td>
<td>Class 25 — Invalid Transaction State</td>
<td>read_only_sql_transaction</td>
</tr>
<tr>
<td>25011</td>
<td>Class 25 — Invalid Transaction State</td>
<td>no_active_sql_transaction_for_branch_transaction</td>
</tr>
<tr>
<td>25012</td>
<td>Class 25 — Invalid Transaction State</td>
<td>in_failed_sql_transaction</td>
</tr>
<tr>
<td>26000</td>
<td>Class 26 — Invalid SQL Statement Name</td>
<td>invalid_sql_statement_name</td>
</tr>
<tr>
<td>28000</td>
<td>Class 28 — Invalid Authorization Specification</td>
<td>invalid_authorization_specification</td>
</tr>
<tr>
<td>28000</td>
<td>Class 2B — Dependent Privilege Descriptors Still Exist</td>
<td>dependent_privilege_descriptors_still_exist</td>
</tr>
<tr>
<td>28001</td>
<td>Class 2B — Dependent Privilege Descriptors Still Exist</td>
<td>dependent_objects_still_exist</td>
</tr>
<tr>
<td>2D000</td>
<td>Class 2D — Invalid Transaction Termination</td>
<td>invalid_transaction_termination</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Class</td>
<td>Condition Name</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>2F000</td>
<td>Class 2F — SQL Routine Exception</td>
<td>sql_routine_exception</td>
</tr>
<tr>
<td>2F005</td>
<td>Class 2F — SQL Routine Exception</td>
<td>function_executed_no_return_statement</td>
</tr>
<tr>
<td>2F002</td>
<td>Class 2F — SQL Routine Exception</td>
<td>modifying_sql_data_not_permitted</td>
</tr>
<tr>
<td>2F003</td>
<td>Class 2F — SQL Routine Exception</td>
<td>prohibited_sql_statement_attempted</td>
</tr>
<tr>
<td>2F004</td>
<td>Class 2F — SQL Routine Exception</td>
<td>reading_sql_data_not_permitted</td>
</tr>
<tr>
<td>34000</td>
<td>Class 34 — Invalid Cursor Name</td>
<td>invalid_cursor_name</td>
</tr>
<tr>
<td>38000</td>
<td>Class 38 — External Routine Exception</td>
<td>external_routine_exception</td>
</tr>
<tr>
<td>38001</td>
<td>Class 38 — External Routine Exception</td>
<td>containing_sql_not_permitted</td>
</tr>
<tr>
<td>38002</td>
<td>Class 38 — External Routine Exception</td>
<td>modifying_sql_data_not_permitted</td>
</tr>
<tr>
<td>38003</td>
<td>Class 38 — External Routine Exception</td>
<td>prohibited_sql_statement_attempted</td>
</tr>
<tr>
<td>38004</td>
<td>Class 38 — External Routine Exception</td>
<td>reading_sql_data_not_permitted</td>
</tr>
<tr>
<td>39000</td>
<td>Class 39 — External Routine Invocation Exception</td>
<td>external_routine_invocation_exception</td>
</tr>
<tr>
<td>39001</td>
<td>Class 39 — External Routine Invocation Exception</td>
<td>invalid_sqlstate_returned</td>
</tr>
<tr>
<td>39004</td>
<td>Class 39 — External Routine Invocation Exception</td>
<td>null_value_not_allowed</td>
</tr>
<tr>
<td>39P01</td>
<td>Class 39 — External Routine Invocation Exception</td>
<td>trigger_protocol_violated</td>
</tr>
<tr>
<td>39P02</td>
<td>Class 39 — External Routine Invocation Exception</td>
<td>srf_protocol_violated</td>
</tr>
<tr>
<td>3D000</td>
<td>Class 3D — Invalid Catalog Name</td>
<td>invalid_catalog_name</td>
</tr>
<tr>
<td>3F000</td>
<td>Class 3F — Invalid Schema Name</td>
<td>invalid_schema_name</td>
</tr>
<tr>
<td>42000</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>syntax_error_or_access_rule_violation</td>
</tr>
<tr>
<td>42601</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>syntax_error</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Class</td>
<td>Condition Name</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>42501</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>insufficient_privilege</td>
</tr>
<tr>
<td>42846</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>cannot_coerce</td>
</tr>
<tr>
<td>42803</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>grouping_error</td>
</tr>
<tr>
<td>42830</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_foreign_key</td>
</tr>
<tr>
<td>42602</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_name</td>
</tr>
<tr>
<td>42622</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>name_too_long</td>
</tr>
<tr>
<td>42939</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>reserved_name</td>
</tr>
<tr>
<td>42804</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>datatype_mismatch</td>
</tr>
<tr>
<td>42P18</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>indeterminate_datatype</td>
</tr>
<tr>
<td>42809</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>wrong_object_type</td>
</tr>
<tr>
<td>42703</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>undefined_column</td>
</tr>
<tr>
<td>42883</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>undefined_function</td>
</tr>
<tr>
<td>42P01</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>undefined_table</td>
</tr>
<tr>
<td>42P02</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>undefined_parameter</td>
</tr>
<tr>
<td>42704</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>undefined_object</td>
</tr>
<tr>
<td>42701</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_column</td>
</tr>
<tr>
<td>42P03</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_cursor</td>
</tr>
<tr>
<td>42P04</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_database</td>
</tr>
<tr>
<td>42723</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_function</td>
</tr>
<tr>
<td>42P05</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_prepared_statement</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Class</td>
<td>Condition Name</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>42P06</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_schema</td>
</tr>
<tr>
<td>42P07</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_table</td>
</tr>
<tr>
<td>42712</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_alias</td>
</tr>
<tr>
<td>42710</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>duplicate_object</td>
</tr>
<tr>
<td>42702</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>ambiguous_column</td>
</tr>
<tr>
<td>42725</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>ambiguous_function</td>
</tr>
<tr>
<td>42P08</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>ambiguous_parameter</td>
</tr>
<tr>
<td>42P09</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>ambiguous_alias</td>
</tr>
<tr>
<td>42P10</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_column_reference</td>
</tr>
<tr>
<td>42611</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_column_definition</td>
</tr>
<tr>
<td>42P11</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_cursor_definition</td>
</tr>
<tr>
<td>42P12</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_database_definition</td>
</tr>
<tr>
<td>42P13</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_function_definition</td>
</tr>
<tr>
<td>42P14</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_prepared_statement_definition</td>
</tr>
<tr>
<td>42P15</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_schema_definition</td>
</tr>
<tr>
<td>42P16</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_table_definition</td>
</tr>
<tr>
<td>42P17</td>
<td>Class 42 — Syntax Error or Access Rule Violation</td>
<td>invalid_object_definition</td>
</tr>
<tr>
<td>44000</td>
<td>Class 44 — WITH CHECK OPTION Violation</td>
<td>with_check_option_violation</td>
</tr>
<tr>
<td>53000</td>
<td>Class 53 — Insufficient Resources</td>
<td>insufficient_resources</td>
</tr>
<tr>
<td>53100</td>
<td>Class 53 — Insufficient Resources</td>
<td>disk_full</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Class</td>
<td>Condition Name</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>----------------</td>
</tr>
<tr>
<td>53200</td>
<td>Class 53 — Insufficient Resources</td>
<td>out_of_memory</td>
</tr>
<tr>
<td>53300</td>
<td>Class 53 — Insufficient Resources</td>
<td>too_many_connections</td>
</tr>
<tr>
<td>54000</td>
<td>Class 54 — Program Limit Exceeded</td>
<td>program_limit_exceeded</td>
</tr>
<tr>
<td>54001</td>
<td>Class 54 — Program Limit Exceeded</td>
<td>statement_too_complex</td>
</tr>
<tr>
<td>54011</td>
<td>Class 54 — Program Limit Exceeded</td>
<td>too_many_columns</td>
</tr>
<tr>
<td>54023</td>
<td>Class 54 — Program Limit Exceeded</td>
<td>too_many_arguments</td>
</tr>
<tr>
<td>55000</td>
<td>Class 55 — Object Not In Prerequisite State</td>
<td>object_not_in_prerequisite_state</td>
</tr>
<tr>
<td>55006</td>
<td>Class 55 — Object Not In Prerequisite State</td>
<td>object_in_use</td>
</tr>
<tr>
<td>55P02</td>
<td>Class 55 — Object Not In Prerequisite State</td>
<td>cant_change_runtime_param</td>
</tr>
<tr>
<td>55P03</td>
<td>Class 55 — Object Not In Prerequisite State</td>
<td>lock_not_available</td>
</tr>
<tr>
<td>57000</td>
<td>Class 57 — Operator Intervention</td>
<td>operator_intervention</td>
</tr>
<tr>
<td>57014</td>
<td>Class 57 — Operator Intervention</td>
<td>query_canceled</td>
</tr>
<tr>
<td>57P01</td>
<td>Class 57 — Operator Intervention</td>
<td>admin_shutdown</td>
</tr>
<tr>
<td>57P02</td>
<td>Class 57 — Operator Intervention</td>
<td>crash_shutdown</td>
</tr>
<tr>
<td>57P03</td>
<td>Class 57 — Operator Intervention</td>
<td>cannot_connect_now</td>
</tr>
<tr>
<td>58000</td>
<td>Class 58 — System Error (errors external to PostgreSQL)</td>
<td>system_error</td>
</tr>
<tr>
<td>58030</td>
<td>Class 58 — System Error (errors external to PostgreSQL)</td>
<td>io_error</td>
</tr>
<tr>
<td>58P01</td>
<td>Class 58 — System Error (errors external to PostgreSQL)</td>
<td>undefined_file</td>
</tr>
<tr>
<td>58P02</td>
<td>Class 58 — System Error (errors external to PostgreSQL)</td>
<td>duplicate_file</td>
</tr>
<tr>
<td>F0000</td>
<td>Class F0 — Configuration File Error</td>
<td>duplicate_file</td>
</tr>
<tr>
<td>Error Code</td>
<td>Error Class</td>
<td>Condition Name</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>F0001</td>
<td>Class F0 — Configuration File Error</td>
<td>lock_file_exists</td>
</tr>
<tr>
<td>P0000</td>
<td>Class P0 — PL/pgSQL Error</td>
<td>plpgsql_error</td>
</tr>
<tr>
<td>P0001</td>
<td>Class P0 — PL/pgSQL Error</td>
<td>raise_exception</td>
</tr>
<tr>
<td>P0002</td>
<td>Class P0 — PL/pgSQL Error</td>
<td>no_data_found</td>
</tr>
<tr>
<td>P0003</td>
<td>Class P0 — PL/pgSQL Error</td>
<td>too_many_rows</td>
</tr>
<tr>
<td>XX000</td>
<td>Class XX — Internal Error</td>
<td>internal_error</td>
</tr>
<tr>
<td>XX001</td>
<td>Class XX — Internal Error</td>
<td>data_corrupted</td>
</tr>
<tr>
<td>XX002</td>
<td>Class XX — Internal Error</td>
<td>index_corrupted</td>
</tr>
</tbody>
</table>

**Amazon Redshift RSQL environment variables**

Amazon Redshift RSQL can use environment variables to select default parameter values.

**RSPASSWORD**

**Important**

We don't recommend using this environment variable for security reasons, as some operating systems allow non-administrative users to see process environment variables.

Sets the password for Amazon Redshift RSQL to use when connecting to Amazon Redshift. This environment variable requires Amazon Redshift RSQL 1.0.4 and above.

RSQL prioritizes RSPASSWORD if one is set. If RSPASSWORD is not set and you're connecting using a DSN, RSQL takes the password from the DSN file's parameters. Finally, if RSPASSWORD is not set and you're not using a DSN, RSQL provides a password prompt after attempting to connect.

The following is an example of setting an RSPASSWORD:

```bash
export RSPASSWORD=TestPassword
```

**Connecting with SQL Workbench/J**

You can connect to a database using SQL Workbench/J, a free, DBMS-independent, cross-platform SQL query tool.

Amazon Redshift doesn't provide or install any third-party SQL client tools or libraries, so you must install any that you want to use with your database. To install SQL Workbench/J, follow the instructions in the SQL Workbench/J documentation ([SQL Workbench/J](#)). In general, to use SQL Workbench/J, you do the following:

- Review the [SQL Workbench/J software license](#).
- Go to the [SQL Workbench/J](#) website and download the appropriate package for your operating system on your client computer or Amazon EC2 instance.
- Go to the [installing and starting SQL Workbench/J](#) page. Follow the instructions for installing SQL Workbench/J on your system.

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.
• Connect to your database over a JDBC connection in SQL Workbench/J.

Connect to your cluster programmatically

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, version 2.1 (p. 201). Also, make sure you have configured firewall settings to allow access to your database. For more information, see Step 4: Authorize access to the cluster.

• Create a new connection profile in SQL Workbench/J that uses the Amazon Redshift driver.

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. 303) 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. 303).

For step-by-step instructions to run the following example, see Running Java examples for Amazon Redshift using Eclipse (p. 584).

```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:[ConnectToClusterExample demonstrates how to connect to an Amazon Redshift cluster and run a sample query.]
// snippet-service:[redshift]
```
package connection;

import java.sql.*;
import java.util.Properties;

public class ConnectToCluster {
    //Redshift JDBC 4.1 driver: com.amazon.redshift.jdbc41.Driver
    //Redshift JDBC 4 driver: com.amazon.redshift.jdbc4.Driver
    static final String dbURL = "jdbc:redshift://x.y.us-west-2.redshift.amazonaws.com:5439/dev";
    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.
            Class.forName("com.amazon.redshift.jdbc.Driver");

            //Open a connection and define properties.
            System.out.println("Connecting to database...");
            Properties props = new Properties();
            props.setProperty("user", MasterUsername);
            props.setProperty("password", MasterUserPassword);
            conn = DriverManager.getConnection(dbURL, props);

            //Try a simple query.
            System.out.println("Listing system tables...");
            String sql = "select * from information_schema.tables;";
            ResultSet rs = stmt.executeQuery(sql);

            //Get the data from the result set.
            while(rs.next()){
                String catalog = rs.getString("table_catalog");
                String name = rs.getString("table_name");

                //Display values.
                System.out.print("Catalog: " + catalog);
                System.out.print("Name: " + name);
            }
            rs.close();
            stmt.close();
            conn.close();
        }catch(Exception ex){
            //For convenience, handle all errors here.
            ex.printStackTrace();
        }
    }
}
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. 303) 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.docsamples
{
    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.
```
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="wJalrXUtneFEMI/K7MDENG/bPxrCfXYPJEXAMPLEKEY"
```

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="wJalrXUtneFEMI/K7MDENG/bPxrCfXYPJEXAMPLEKEY"
```

**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. 303).

**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. 306).

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.
Topics

• Connecting from outside of Amazon EC2—firewall timeout issue (p. 347)
• Connection is refused or fails (p. 349)
• Client and driver are incompatible (p. 349)
• Queries appear to hang and sometimes fail to reach the cluster (p. 349)

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. 347).
• Optionally, set keepalive behavior at the DSN level. For more information, see Change DSN timeout settings (p. 348).

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
• **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_keealive=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_keealive=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_keealive
```

### 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. 297).

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\SOFTWARE\Microsoft\Ole Autodrive\Proxy\Properties\Connection Properties\TIMEOUTS\KeepAliveInterval`.
Troubleshooting connection issues in Amazon Redshift

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. 121).
- 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. 577).

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. 577).

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. 350).

- 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. 350).

**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](https://tools.ietf.org/html/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. 577). 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](https://docs.aws.amazon.com/redshift/latest/dg/concepts/ec2-classic-vpc-comparison.html) 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. 123). For more information about specific PMTUD settings required in this rule, see [Path MTU discovery](https://docs.aws.amazon.com/redshift/latest/dg/concepts/path-mtu-discovery.html) 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 your database. 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:
Considerations when calling the Amazon Redshift Data API

Consider the following when calling the Data API:

- The Amazon Redshift Data API can access databases in Amazon Redshift provisioned clusters and Redshift Serverless workgroups. For a list of AWS Regions where the Redshift Data API is available, see the endpoints listed for Redshift Data API in the Amazon Web Services General Reference.
- 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 permissions as the runner of an ExecuteStatement or BatchExecuteStatement API operation can act on the same statement with CancelStatement, DescribeStatement, GetStatementResult, and ListStatements API operations. To act on the same SQL statement from another user, the user must be able to assume the IAM role of the user who ran the SQL statement. For more information about how to assume a role, see Authorizing access to the Amazon Redshift Data API (p. 356).
- The SQL statements in the Sqls parameter of BatchExecuteStatement API operation are run as a single transaction. They run serially in the order of the array. Subsequent SQL statements don't start until the previous statement in the array completes. If any SQL statement fails, then because they are run as one transaction, all work is rolled back.
- The maximum retention time for a client token used in ExecuteStatement or BatchExecuteStatement API operation is 8 hours.
Choosing database 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 of a secret stored in AWS Secrets Manager which has username and password. The specified secret contains credentials to connect to the database you specify. When you are connecting to a cluster, you also supply the database name, if you provide a cluster identifier (dbClusterIdentifier), it must match the cluster identifier stored in the secret. When you are connecting to a serverless workgroup, you also supply the database name. For more information, see Storing database credentials in AWS Secrets Manager (p. 357).

Temporary credentials

With this method, choose one of the following options:

- When connecting to a serverless workgroup, specify the workgroup name and database name. The database user name is derived from the IAM identity. For example, arn:iam::123456789012:user:foo has the database user name IAM:foo. Also, permission to call the redshift-serverless:GetCredentials operation is required.
- When connecting to a cluster as an IAM identity, specify the cluster identifier and the database name. The database user name is derived from the IAM identity. For example, arn:iam::123456789012:user:foo has the database user name IAM:foo. Also, permission to call the redshift:GetClusterCredentialsWithIAM operation is required.
- When connecting to a cluster as a database user, specify the cluster identifier, the database name, and the database user name. Also, permission to call the redshift:GetClusterCredentials operation is required. For information about how to join database groups when connecting with this method, see Joining database groups when connecting to a cluster (p. 360).

With these methods, you can also supply a region value that specifies the AWS Region where your data 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, 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>VARBINARY</td>
<td>STRING</td>
</tr>
<tr>
<td>CLOB</td>
<td>STRING</td>
</tr>
</tbody>
</table>
Considerations when calling the Data API

### JDBC data type | Data API data type
---|---
Other types (including types related to date and time) | STRING

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.

```
--parameters "[{"name": \"id\", \"value\": \"1\"},{\"name\": \"address\", \"value\": \"Seattle\"}]"
```

Consider the following when using named parameters:

- Named parameters can only be used to replace values in SQL statements.
  - You can replace the values in an INSERT statement, such as `INSERT INTO mytable VALUES(:val1)`.

  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 a 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 "insert into mytable values (:id, :address)"
  ```

  - You can replace the values in a conditions clause, such as `WHERE attr >= :val1`, `WHERE attr BETWEEN :val1 AND :val2`, and `HAVING COUNT(attr) > :val`.
  - You can't replace column names in an SQL statement, such as `SELECT column-name`, `ORDER BY column-name`, or `GROUP BY column-name`.

  For example, the following SELECT statement fails with invalid syntax.

  ```
  --sql "SELECT :colname, FROM event" --parameters "[{"name": \"colname\", \"value\": \"eventname\"}]"
  ```

  If you describe (describe-statement operation) the statement with the syntax error, the `QueryString` returned does not substitute the column name for the parameter ("QueryString": "SELECT :colname, FROM event"), and an error is reported (ERROR: syntax error at or near \"FROM\n\nPosition: 12)."

- You can't replace column names in an aggregate function, such as `COUNT(column-name)`, `AVG(column-name)`, or `SUM(column-name)`.

- You can't replace column names in a JOIN clause.

---

354
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.

```sql
--parameters "[{"name": "id", "value": "null"}]
```

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": ""}]
```

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 a SQL statement.

Only the execute-statement operation supports SQL statements with parameters.

## Running SQL statements with an idempotency token when calling the Amazon Redshift Data API

When you make a mutating API request, the request typically returns a result before the operation's asynchronous workflows have completed. Operations might also time out or encounter other server issues before they complete, even though the request has already returned a result. This could make it difficult to determine whether the request succeeded or not, and could lead to multiple retries to ensure that the operation completes successfully. However, if the original request and the subsequent retries are successful, the operation is completed multiple times. This means that you might update more resources than you intended.

Idempotency ensures that an API request completes no more than one time. With an idempotent request, if the original request completes successfully, any subsequent retries complete successfully without performing any further actions. The Data API ExecuteStatement and BatchExecuteStatement operations have an optional ClientToken idempotent parameter. The ClientToken expires after 8 hours.

**Important**

If you call ExecuteStatement and BatchExecuteStatement operations from an AWS SDK, it automatically generates a client token to use on retry. In this case, we don't recommend using the client-token parameter with ExecuteStatement and BatchExecuteStatement operations. View the CloudTrail log to see the ClientToken. For a CloudTrail log example, see Amazon Redshift Data API examples (p. 570).

The following execute-statement AWS CLI command illustrates the optional client-token parameter for idempotency.

```
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 * from stl_query limit 1"
  --database dev
  --client-token b855dced-259b-444c-bc7b-d3e8e33f94g1
```

The following table shows some common responses that you might get for idempotent API requests, and provides retry recommendations.
Amazon Redshift Management Guide
Authorizing access

<table>
<thead>
<tr>
<th>Response</th>
<th>Recommendation</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 (OK)</td>
<td>Do not retry</td>
<td>The original request completed successfully. Any subsequent retries return successfully.</td>
</tr>
<tr>
<td>400-series response codes</td>
<td>Do not retry</td>
<td>There is a problem with the request, from among the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• It includes a parameter or parameter combination that is not valid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• It uses an action or resource for which you do not have permissions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• It uses a resource that is in the process of changing states.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the request involves a resource that is in the process of changing states, retrying the request could possibly succeed.</td>
</tr>
<tr>
<td>500-series response codes</td>
<td>Retry</td>
<td>The error is caused by an AWS server-side issue and is generally transient. Repeat the request with an appropriate backoff strategy.</td>
</tr>
</tbody>
</table>

For information about Amazon Redshift response codes, see Common Errors in the Amazon Redshift API Reference.

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. As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see Identity and access management in Amazon Redshift. 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 or Redshift Serverless workgroup.

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.

Consider the following requirements of the IAM policy associated with the user:

- If you use AWS Secrets Manager to authenticate, confirm 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 to a cluster, confirm the policy allows the 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.
- If you use temporary credentials to authenticate to a serverless workgroup, confirm the policy allows the use of the redshift-serverless:GetCredentials action to retrieve the workgroup tagged
with the key RedshiftDataFullAccess. The database user is mapped 1:1 to the source AWS Identity and Access Management (IAM) identity. For example, the user sample_user is mapped to database user IAM:sample_user, and IAM role sample_role is mapped to IAMR:sample_role. For more information about IAM identities, see IAM Identities (users, user groups, and roles) in the IAM User Guide.

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:

```
{
  "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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::accountID-of-account-B:role/someRoleB"
  }
}
```

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 or serverless workgroup 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 the AWS Secrets Manager console 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](https://docs.aws.amazon.com/secretsmanager/latest/userguide/secretsmanager-create-secret.html) 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.

To store your credentials in a secret for a serverless workgroup

1. Use AWS Secrets Manager AWS CLI commands to store a secret that contains credentials for your serverless workgroup:
   - Create your secret in a file, for example a JSON file named `mycreds.json`. Provide the values for **User name** (database user) and **Password** in the file.
     ```json
     {
       "username": "myusername",
       "password": "mypassword"
     }
     ```
   - Store your values in your secret and tag the secret with the key **RedshiftDataFullAccess**.

     ```bash
     aws secretsmanager create-secret --name MyRedshiftSecret --tags Key="RedshiftDataFullAccess",Value="serverless" --secret-string file://mycreds.json
     ```

   The following shows the output.

     ```json
     {
       "Name": "MyRedshiftSecret",
       "VersionId": "a1605925-e8ea-4739-9ae9-e509eEXAMPLE"
     }
     ```

   For more information, see [Creating a Basic Secret with AWS CLI](https://docs.aws.amazon.com/secretsmanager/latest/userguide/secretsmanager-create-secret-cli.html) 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 to connect to services across different accounts and VPCs based on Amazon VPC. For more information about AWS PrivateLink, see VPC Endpoint Services (AWS PrivateLink) in the Amazon Virtual Private Cloud User Guide.

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. 352).

**Joining database groups when connecting to a cluster**

Database groups are collections of database users. Database privileges can be granted to groups. An administrator can configure an IAM role such that these database groups are taken into account when your SQL runs with the Data API. For more information about database groups, see [Groups](Groups) in the **Amazon Redshift Database Developer Guide**.

You can configure a Data API caller's IAM role so that the database user specified in the call joins database groups when the Data API connects to a cluster. This capability is only supported when connecting to provisioned clusters. It's not supported when connecting to Redshift Serverless workgroups. The IAM role of the caller of the Data API must also allow the redshift:JoinGroup action.

Configure this by adding tags to IAM roles. The administrator of the caller's IAM role adds tags with the key RedshiftDbGroups and a key value of a list of database groups. The value is a list of colon (:) separated names of database groups up to a total length of 256 characters. The database groups must be previously defined in the connected database. If any specified group is not found in the database, it's ignored. For example, for database groups accounting and retail, the key-value is accounting:retail. The tag key-value pair 

|"Key":"RedshiftDbGroups","Value":"accounting:retail"| is used by the Data API to determine which database groups are associated with the provided database user in the call to the Data API.

**To add database groups as a tag to an IAM role**

1. Sign in to the AWS Management Console and open the IAM console at [https://console.aws.amazon.com/iam/](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, then choose **Manage tags**.
4. Choose **Add tag**, then add the key RedshiftDbGroups and a value which is a list of database-groups-colon-separated.
5. Choose **Save changes**.

Now when an IAM principal (with this IAM role attached) calls the Data API, the specified database user joins the database groups specified in the IAM role.

For more information on how to attach a tag to a principal, including IAM roles and IAM users, see [Tagging IAM resources](Tagging IAM resources) in the **IAM User Guide**.

**Calling the Data API**

You can call the Data API or the AWS CLI to run SQL statements on your cluster or serverless workgroup. The primary operations to run SQL statements are **ExecuteStatement** and **BatchExecuteStatement**.
in the Amazon Redshift Data API Reference. 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.

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. In many of the examples a cluster-identifier is provided to run against a cluster. When you run against a serverless workgroup, you provide a workgroup-name instead. 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 a SQL statement

To run a SQL statement, use the aws redshift-data execute-statement AWS CLI command.

The following AWS CLI command runs a SQL statement against a cluster 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-hKgPWhn
  --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",
  "Id": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814",
}
```

The following AWS CLI command runs a SQL statement against a cluster and returns an identifier to fetch the results. This example uses the temporary credentials authentication method.

```bash
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.

```json
{

}
```
The following AWS CLI command runs a SQL statement against a serverless workgroup and returns an identifier to fetch the results. This example uses the temporary credentials authentication method.

```bash
aws redshift-data execute-statement
  --database dev
  --workgroup-name myworkgroup
  --sql "select 1;"
```

The following is an example of the response.

```json
{
  "CreatedAt": "2022-02-11T06:25:28.748000+00:00",
  "Database": "dev",
  "DbUser": "IAMR:RoleName",
  "Id": "89dd91f5-2d43-43d5-8461-f33aa093c41e",
  "WorkgroupName": "myworkgroup"
}
```

The following AWS CLI command runs a SQL statement against a cluster and returns an identifier to fetch the results. This example uses the AWS Secrets Manager authentication method and an idempotency token.

```bash
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 * from stl_query limit 1"
  --database dev
  --client-token b855dced-259b-444c-bc7b-d3e8e33f94g1
```

The following is an example of the response.

```json
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598323175.823,
  "Database": "dev",
  "Id": "c016234e-5c6c-4bc5-bb16-2c5b8ff61814",
}
```

To run a SQL statement with parameters

To run a SQL statement, use the `aws redshift-data execute-statement` AWS CLI command.

The following AWS CLI command runs a SQL statement against a cluster and returns an identifier to fetch the results. This example uses the AWS Secrets Manager authentication method. The SQL text has named parameter `distance`. In this case, the distance used in the predicate is 5. In a SELECT statement,
named parameters for column names can only be used in the predicate. Values for named parameters for the SQL statement are specified in the parameters option.

```bash
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 ratecode FROM demo_table WHERE trip_distance > :distance"
  --parameters "["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:

```bash
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.

```bash
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.

```bash
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\"},
  {\"name\": \"id\", \"value\": \"2\"}]"
```

The following shows the two inserted rows:

<table>
<thead>
<tr>
<th>eventid</th>
<th>venueid</th>
<th>catid</th>
<th>dateid</th>
<th>eventname</th>
<th>starttime</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2022-02-22 00:00:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2022-02-22 00:00:00</td>
</tr>
</tbody>
</table>

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"
},
{ "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"}
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 against a cluster and returns an identifier to fetch the results. This example uses the temporary credentials authentication method.

```
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.

```
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598306924.632,
  "Database": "dev",
  "DbUser": "myuser",
}
```

"TotalNumRows": 1

---

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 against a cluster and returns an identifier to fetch the results. This example uses the temporary credentials authentication method.

```
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.

```
{
  "ClusterIdentifier": "mycluster-test",
  "CreatedAt": 1598306924.632,
  "Database": "dev",
  "DbUser": "myuser",
}
```

"TotalNumRows": 1
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": "e0ebd578-58b3-46cc-8e52-8163fd7e01aa",
      "QueryString": "select * from stl_query limit 1",
      "Status": "FAILED",
      "UpdatedAt": 1598311719.008
   },
   { 
      "CreatedAt": 1598313683.65,
      "Id": "c361d4f7-8c53-4543-8c45-6b2b1166330c",
      "QueryString": "select * from stl_query limit 1",
      "Status": "ABORTED",
      "UpdatedAt": 1598313685.495
   },
   { 
      "CreatedAt": 1598306653.333,
      "Id": "a512b7bd-98c7-45d5-985b-a715f3cfde7f",
      "QueryString": "select 1",
      "Status": "FINISHED",
      "UpdatedAt": 1598306653.992
   }
   ]
}
```

To describe metadata about a SQL statement

To get descriptions of metadata for a 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 a SQL statement.
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": "b2906c76-fa6e-4cdf-8c5f-4de1ff9b7652",
  "RedshiftPid": 31459,
  "RedshiftQueryId": 0,
  "ResultRows": 2,
  "ResultSize": 22,
  "Status": "FINISHED",
  "UpdatedAt": 1623979778.183,
  "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
    }
  ]
}
```
To fetch the results of a SQL statement

To fetch the result from a 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 a 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-4de1ff9b7652:2`. 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 a SQL statement run by `execute-statement`.

```
aws redshift-data get-statement-result
  --id d9b6c0c9-8747-4bf4-b142-e883122f766
  --region us-west-2
```

The following statement returns the result of the second SQL statement run by `batch-execute-statement`.

```
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`.

```

{
  "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,
    }
  ]
}
```
Amazon Redshift Management Guide
Calling the Data API

},
{

},
{

},
{

},
{

"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": 0,
"schemaName": "",
"tableName": "stll_query",
"typeName": "int4"
"isCaseSensitive": true,
"isCurrency": false,
"isSigned": false,
"label": "database",
"length": 0,
"name": "database",
"nullable": 0,
"precision": 32,
"scale": 0,
"schemaName": "",
"tableName": "stll_query",
"typeName": "bpchar"
"isCaseSensitive": true,
"isCurrency": false,
"isSigned": false,
"label": "querytxt",
"length": 0,
"name": "querytxt",
"nullable": 0,
"precision": 4000,
"scale": 0,
"schemaName": "",
"tableName": "stll_query",
"typeName": "bpchar"

370


Amazon Redshift Management Guide

Calling the Data API

},
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": false,
  "label": "starttime",
  "length": 0,
  "name": "starttime",
  "nullable": 0,
  "precision": 29,
  "scale": 6,
  "schemaName": "",
  "tableName": "stll_query",
  "typeName": "timestamp"
},
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": false,
  "label": "endtime",
  "length": 0,
  "name": "endtime",
  "nullable": 0,
  "precision": 29,
  "scale": 6,
  "schemaName": "",
  "tableName": "stll_query",
  "type": 93,
  "typeName": "timestamp"
},
{
  "isCaseSensitive": false,
  "isCurrency": false,
  "isSigned": true,
  "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"
}
"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" }, { "stringValue": "2020-08-21 17:33:52.974306" }, { "longValue": 0 }, { "longValue": 0 }, { "longValue": 6 } ] ],
"TotalNumRows": 1
]

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 a SQL statement against a cluster 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
```
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"
    }
  ]
}
```

The following AWS CLI command runs a SQL statement against a cluster 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
```

<table>
<thead>
<tr>
<th>feature_name</th>
<th>type</th>
<th>length</th>
<th>nullable</th>
<th>precision</th>
<th>scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>feature_name</td>
<td>text</td>
<td>2147483647</td>
<td>1</td>
<td>2147483647</td>
<td>0</td>
</tr>
<tr>
<td>sub_feature_id</td>
<td>text</td>
<td>2147483647</td>
<td>1</td>
<td>2147483647</td>
<td>0</td>
</tr>
<tr>
<td>sub_feature_name</td>
<td>text</td>
<td>2147483647</td>
<td>1</td>
<td>2147483647</td>
<td>0</td>
</tr>
<tr>
<td>is_supported</td>
<td>boolean</td>
<td>2147483647</td>
<td>1</td>
<td>2147483647</td>
<td>0</td>
</tr>
<tr>
<td>is_verified_by</td>
<td>text</td>
<td>2147483647</td>
<td>1</td>
<td>2147483647</td>
<td>0</td>
</tr>
</tbody>
</table>
"schemaName": "information_schema",
"tableName": "sql_features",
"typeName": "character_data"
}
{
"isCaseSensitive": false,
"isCurrency": false,
"isSigned": false,
"length": 2147483647,
"name": "comments",
"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 a SQL statement against a cluster 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-hKgPWn
--cluster-identifier mycluster-test
--database dev
```

The following is an example of the response.

```
{
"Databases": [
 "dev"
 ]
}
```

The following AWS CLI command runs a SQL statement against a cluster 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.

```
{
"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 a SQL statement against a cluster to list schemas in a database. This example uses the AWS Secrets Manager authentication method.

```
aws redshift-data list-schemas
  --region us-west-2
  --secret arn:aws:secretsmanager:us-west-2:123456789012:secret:myuser-secret-hKgPw
  --cluster-identifier mycluster-test
  --database dev
```

The following is an example of the response.

```
{
  "Schemas": [
    "information_schema",
    "pg_catalog",
    "pg_internal",
    "public"
  ]
}
```

The following AWS CLI command runs a SQL statement against a cluster to list schemas in a database. This example uses the temporary credentials authentication method.

```
aws redshift-data list-schemas
  --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 a SQL statement against a cluster 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-hKgPw
```
The following is an example of the response.

```json
{
  "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 a SQL statement against a cluster to list tables in a database. This example uses the temporary credentials authentication method.

```bash
aws redshift-data list-tables
  --region us-west-2
  --db-user myuser
  --cluster-identifier mycluster-test
  --database dev
  --schema information_schema
```

The following is an example of the response.

```json
{
  "Tables": [
    {
      "name": "sql_features",
      "schema": "information_schema",
      "type": "SYSTEM TABLE"
    },
    {
      "name": "sql_implementation_info",
      "schema": "information_schema",
      "type": "SYSTEM TABLE"
    }
  ]
}
```

## 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. 378)**
- **Database response exceeded size limit (p. 378)**
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

You can create rules that match selected events 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 on the IAM console. When creating a role on the IAM console, choose the AWS service trusted entity for CloudWatch Events. Specify the IAM role in the RoleArn JSON value in the EventBridge target. For more information about creating an IAM role, see Creating a Role for an AWS Service (Console) in the IAM User Guide.

The following examples show variations of creating EventBridge rules with a single or multiple SQL statements and with an Amazon Redshift cluster or an Amazon Redshift Serverless workgroup as the data warehouse.

Calling with a single SQL statement and cluster

The following example uses the AWS CLI to create an EventBridge rule that is used to run a SQL statement against an Amazon Redshift cluster.

```bash
aws events put-rule
--name test-redshift-cluster-data
--schedule-expression "rate(1 minute)"
```

Then an EventBridge target is created to run on the schedule specified in the rule.

```bash
aws events put-targets
--cli-input-json file://data.json
```

The input data.json file is as follows. The Sql JSON key indicates there is a single SQL statement. The Arn JSON value contains a cluster identifier. The RoleArn JSON value contains the IAM role used to run the SQL as described previously.

```json
{
  "Rule": "test-redshift-cluster-data",
  "Sql": "SELECT * FROM my_table;",
  "RoleArn": "arn:aws:iam::123456789012:role/my_role"
}
```
Calling with a single SQL statement and workgroup

The following example uses the AWS CLI to create an EventBridge rule that is used to run a SQL statement against an Amazon Redshift Serverless workgroup.

```bash
aws events put-rule
  --name test-redshift-serverless-workgroup-data
  --schedule-expression "rate(1 minute)"
```

Then an EventBridge target is created to run on the schedule specified in the rule.

```bash
aws events put-targets
  --cli-input-json file://data.json
```

The input data.json file is as follows. The Sql JSON key indicates there is a single SQL statement. The Arn JSON value contains a workgroup name. The RoleArn JSON value contains the IAM role used to run the SQL as described previously.

```json
{
  "Rule": "test-redshift-serverless-workgroup-data",
  "EventBusName": "default",
  "Targets": [
    {
      "Id": "2",
      "Arn": "arn:aws:redshift-serverless:us-east-1:123456789012:workgroup/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
      "RedshiftDataParameters": {
        "Database": "dev",
        "Sql": "select 1;",
        "StatementName": "test-scheduler-statement",
        "WithEvent": true
      }
    }
  ]
}
```

Calling with multiple SQL statements and cluster

The following example uses the AWS CLI to create an EventBridge rule that is used to run multiple SQL statements against an Amazon Redshift cluster.

```bash
aws events put-rule
```

```bash
--name test-redshift-cluster-data
--schedule-expression "rate(1 minute)"
```
--name  test-redshift-cluster-data
--schedule-expression "rate(1 minute)"

Then an EventBridge target is created to run on the schedule specified in the rule.

aws events put-targets
--cli-input-json file://data.json

The input data.json file is as follows. The Sqls JSON key indicates there are multiple SQL statements. The Arn JSON value contains a cluster identifier. The RoleArn JSON value contains the IAM role used to run the SQL as described previously.

```json
{
  "Rule": "test-redshift-cluster-data",
  "EventBusName": "default",
  "Targets": [
    {
      "Id": "2",
      "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
      "RedshiftDataParameters": {
        "Database": "dev",
        "Sqls": ["select 1;", "select 2;", "select 3;"],
        "StatementName": "test-scheduler-statement",
        "WithEvent": true
      }
    }
  ]
}
```

Calling with multiple SQL statements and workgroup

The following example uses the AWS CLI to create an EventBridge rule that is used to run multiple SQL statements against an Amazon Redshift Serverless workgroup.

aws events put-rule
--name  test-redshift-serverless-workgroup-data
--schedule-expression "rate(1 minute)"

Then an EventBridge target is created to run on the schedule specified in the rule.

aws events put-targets
--cli-input-json file://data.json

The input data.json file is as follows. The Sqls JSON key indicates there are multiple SQL statements. The Arn JSON value contains a workgroup name. The RoleArn JSON value contains the IAM role used to run the SQL as described previously.

```json
{
  "Rule": "test-redshift-serverless-workgroup-data",
  "EventBusName": "default",
  "Targets": [
    {
      "Id": "2",
      "Arn": "arn:aws:redshift-serverless:us-east-1:123456789012:workgroup/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
      "RoleArn": "arn:aws:iam::123456789012:role/Administrator",
      "RedshiftDataParameters": {
        "Database": "dev",
```
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. To learn more about how Amazon Redshift is integrated with AWS CloudTrail, see Logging with CloudTrail. For more information about CloudTrail, see the AWS CloudTrail User Guide.

Topics
- Monitoring events for the Amazon Redshift Data API in Amazon EventBridge (p. 381)

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
  }
}
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.

Amazon Redshift clusters and Amazon Redshift Serverless workgroups support enhanced VPC routing. You can’t use enhanced VPC routing with Redshift Spectrum. For more information, see Redshift Spectrum and enhanced VPC routing (p. 385).

If enhanced VPC routing is not turned on, 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 turned on, 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. 384). If you use Lake Formation, you can find more information about establishing a private connection between your VPC and AWS Lake Formation at AWS Lake Formation and interface VPC endpoints (AWS PrivateLink).

• **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.
Working with VPC endpoints

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 database 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 data warehouse is in and then turn on enhanced VPC routing. You can turn on enhanced VPC routing when you create your cluster or workgroup, or you can modify a cluster or workgroup in a VPC to use enhanced VPC routing.

A VPC endpoint uses route tables to control the routing of traffic between a cluster or workgroup in the VPC and Amazon S3. All clusters and workgroups 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 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 [Create a VPC endpoint](https://docs.aws.amazon.com/vpc/latest/userguide/create-vpc-endpoint.html) in the Amazon VPC User Guide.

You use endpoint policies to control access from your cluster or workgroup to the Amazon S3 buckets that hold your data files. For more specific control, you can optionally attach a custom endpoint policy. For more information, see [Control access to services using endpoint policies](https://docs.aws.amazon.com/privatelink/latest/userguide/private-link-endpoint-control-access.html) in the AWS PrivateLink 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.

Enhanced VPC routing

You can turn on enhanced VPC routing when you create or modify a cluster, and when you create or modify a Amazon Redshift Serverless workgroup.

To work with enhanced VPC routing for a cluster, 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. 123).

• 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 Provisioned clusters dashboard, then choose Create cluster and enter the Cluster details properties.
3. To display the Additional configurations section, choose to switch off Use defaults.
4. Navigate to the Network and security section.
5. To turn on Enhanced VPC routing, choose Turn on to force cluster traffic through the VPC.
6. Choose Create cluster to create the cluster. The cluster might take several minutes to be ready to use.

To create an Amazon Redshift Serverless workgroup 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 Serverless dashboard, then choose Create workgroup and enter the properties for your workgroup.
3. Navigate to the Network and security section.
4. Select Turn on enhanced VPC routing to route network traffic through the VPC.
5. Choose Next and finish entering your workgroup properties until you Create the workgroup.

Redshift Spectrum and enhanced VPC routing

Amazon Redshift Spectrum doesn't support enhanced VPC routing with provisioned clusters. 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.

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...
Considerations when using Amazon Redshift Spectrum

Following are considerations when using Redshift Spectrum:

- Bucket access policies (p. 386)
- Cluster IAM role (p. 387)
- Logging and auditing Amazon S3 access (p. 387)
- Access to AWS Glue or Amazon Athena (p. 388)

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 on provisioned clusters 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. 538). A service control policy (SCP) can also be used to further restrict the role, see Prevent IAM users and roles from making specified changes, with an exception for a specified admin role in the AWS Organizations User Guide.

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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BucketPolicyForSpectrum",
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::123456789012:role/redshift"]
            },
            "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.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "Service": "redshift.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
      }
   ]
}
```

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.

```
{
   "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 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.

Ensure that the S3 IP ranges are added to your allow list. To learn more about the required S3 IP ranges, see [Network isolation](#).

**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](https://docs.aws.amazon.com/AmazonS3/latest/userguide/how-to-configure-server-access-logging.html) 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](https://aws.amazon.com/blogs/security/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](https://docs.aws.amazon.com/AmazonVPC/latest/ug/create-vpc-endpoint-example.html).

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](https://docs.aws.amazon.com/redshift/latest/userguide/ephemeral-network-routing.html) (p. 383).
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. 392).

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.
### Parameter name | Value | More information
--- | --- | ---
auto_analyze | true | [auto_analyze](https://docs.aws.amazon.com/redshift/latest/dg/auto_analyze.html) in the Amazon Redshift Database Developer Guide
auto_mv | true | [Automated materialized views](https://docs.aws.amazon.com/redshift/latest/dg/automated-materialized-views.html) in the Amazon Redshift Database Developer Guide

datestyle | ISO, MDY | [datestyle](https://docs.aws.amazon.com/redshift/latest/dg/datestyle.html) in the Amazon Redshift Database Developer Guide

datestyle in the Amazon Redshift Database Developer Guide

enable_case_sensitive_identifier | false | [enable_case_sensitive_identifier](https://docs.aws.amazon.com/redshift/latest/dg/enable_case_sensitive_identifier.html) in the Amazon Redshift Database Developer Guide

enable_user_activity_logging | false | Database audit logging (p. 559) in this guide

extra_float_digits | 0 | [extra_float_digits](https://docs.aws.amazon.com/redshift/latest/dg/extra_float_digits.html) in the Amazon Redshift Database Developer Guide

max_concurrency_scaling_clusters | 1 | [max_concurrency_scaling_clusters](https://docs.aws.amazon.com/redshift/latest/dg/max_concurrency_scaling_clusters.html) in the Amazon Redshift Database Developer Guide

query_group | default | [query_group](https://docs.aws.amazon.com/redshift/latest/dg/query_group.html) in the Amazon Redshift Database Developer Guide

require_ssl | false | Configuring security options for connections (p. 303) in this guide

search_path | $user, public | [search_path](https://docs.aws.amazon.com/redshift/latest/dg/search_path.html) in the Amazon Redshift Database Developer Guide

statement_timeout | 0 | [statement_timeout](https://docs.aws.amazon.com/redshift/latest/dg/statement_timeout.html) in the Amazon Redshift Database Developer Guide

wlm_json_configuration | ["auto_wlm":true] | [Configuring workload management (p. 391)](https://docs.aws.amazon.com/redshift/latest/dg/configuring-workload-management.html) in this guide

use_fips_ssl | false | Enable FIPS-compliant SSL mode only if your system is required to be FIPS-compliant.

---

**Note**
The max_cursor_result_set_size parameter is deprecated. For more information about cursor result set size, see [Cursor constraints](https://docs.aws.amazon.com/redshift/latest/dg/cursor-constraints.html) 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](https://docs.aws.amazon.com/redshift/latest/dg/wlm_query_slot_count.html) in the Amazon Redshift Database Developer Guide. For more information about temporarily overriding the other parameters, see [Modifying the server configuration](https://docs.aws.amazon.com/redshift/latest/dg/modifying-server-configuration.html) 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. 391).

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. 116).

**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. 392).

**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. 402).

**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>
</tbody>
</table>
Properties for the `wlm_json_configuration` parameter

<table>
<thead>
<tr>
<th>WLM property</th>
<th>Automatic WLM</th>
<th>Manual WLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query group wildcard</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>User roles</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>User role wildcard</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Timeout</td>
<td>No</td>
<td>Deprecated</td>
</tr>
<tr>
<td>Memory</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Query Monitoring Rules</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The following list describes the WLM properties that you can configure.

**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`. This setting is applied for each parameter group rather than queue.

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

User Roles

A comma-separated list of user roles. When members with that user role run queries in the database, their queries are routed to the queue that is associated with their user role. For more information about user roles, see Role-based access control (RBAC).
JSON property: `user_role`

**User RolesWildcard**

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: `user_role_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.

```json
"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. 402).
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,
  "user_role": [],
  "user_role_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,
  "user_role": []
}
}'
```
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.
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.

```
aws redshift modify-cluster-parameter-group --parameter-group-name myclusterparaametergroup --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": "[\"user_group\":\"admin\",\"dba\", \"query_group\":\"auto\", \"priority\": \"high\", \"queue_type\": \"auto\", \"short_query_queue\": true]"
  }
]
```

---

400
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 one set of 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 (" ").
- 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 one set of 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:
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`
   - `auto_mv`
   - `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. 389).

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/](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)**
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/](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.

```
[
  {
    "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.

```
[
  {
    "user_group": ["example_user_group1"],
    "query_group": ["example_query_group1"],
    "query_concurrency": 7
  }
]
```

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.

```
{
  "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. 584). 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-sourcedescription:[CreateAndModifyClusterParameterGroup demonstrates how to create and modify an Amazon Redshift parameter group.]
// snippet-service:[redshift]
// snippet-keyword:[Java]
// snippet-keyword:[Amazon Redshift]
// snippet-keyword:[Code Sample]
// snippet-keyword:[CreateClusterParameterGroup]
// snippet-keyword:[DescribeClusterParameterGroups]
// snippet-keyword:[ModifyClusterParameterGroup]
// snippet-sourcetype:[full-example]
// snippet-sourcedate:[2019-02-01]
// snippet-sourceauthor:[AWS]
// snippet-start:[redshift.java.CreateAndModifyClusterParameterGroup.complete]

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();
        println("Created cluster parameter group.");
    }

    private static void modifyClusterParameterGroup() {
        List<Parameter> parameters = new ArrayList<Parameter>();
        parameters.add(new Parameter())
    }

}
// 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());
}

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:\n");
    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);
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`
- `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 and backups

Topics

- Overview of snapshots (p. 410)
- Automated snapshots (p. 411)
- Automated snapshot schedules (p. 411)
- Snapshot schedule format (p. 411)
- Manual snapshots (p. 413)
- Managing snapshot storage (p. 413)
- Excluding tables from snapshots (p. 414)
- Copying snapshots to another AWS Region (p. 414)
- Restoring a cluster from a snapshot (p. 414)
- Restoring a table from a snapshot (p. 417)
- Sharing snapshots (p. 418)
- Managing snapshots using the console (p. 420)
- Managing snapshots using the AWS SDK for Java (p. 425)
- Managing snapshots using the AWS CLI and Amazon Redshift API (p. 428)
- Working with AWS Backup (p. 428)

Overview of 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 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.
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. If your data is larger than 5 GB * the number of nodes, the shortest amount of time in between automated snapshot creation is 15 minutes. Alternatively, you can create a snapshot schedule to control when automated snapshots are taken. If you're using custom schedules, the minimum amount of time between automated snapshots is one hour. 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.

**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 don't support the following frequencies:
  - Snapshots scheduled more frequently than 1 per hour.
  - Snapshots scheduled less frequently than 1 per day (24 hours).

  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</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. 421).

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. 643).

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. 447).

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. 422).

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.

```bash
aws redshift describe-cluster-snapshots --region eu-west-1 --snapshot-identifier mycluster-snapshot
```

3. Run the following command to describe the options available for this snapshot.

```bash
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 database, 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, aren't applied to the target table.

If the owner of the source table exists, then that database 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.
- Any dependencies, such as views or permissions granted on the source table, aren't applied to the target table.
- If row-level security is turned on for a table being restored, Amazon Redshift restores the table with row-level security turned on.
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`.

You can use the AWS CLI command `describe-table-restore-status` to review the status of your restore operation. 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 users in that account can perform any actions on the snapshot unless they assume a role with policies that allow them to do so.
• Users in the snapshot owner account can authorize and revoke access to a snapshot only if they assume a role with 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 or role 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"
        ]
      }
    ]
}
```

• Users in an AWS account with which a snapshot has been shared cannot perform actions on that snapshot unless they have permissions allowing those actions. You can do this by assigning the policy to a role and assuming the role.

• 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",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "redshift:DescribeClusterSnapshots"
        ],
        "Resource": [
          "*"
        ]
      }
    ]
}
```

• To restore a snapshot, a user must assume a role with an IAM policy that allows the 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 assume a role with a policy such as the following:

```json
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "redshift:RestoreFromClusterSnapshot"
        ],
        "Resource": [
          "arn:aws:redshift:us-east-1:219876543210:cluster:from-another-account"
        ]
      }
    ]
}
```
After access to a snapshot has been revoked from an AWS account, no users in that account can access the snapshot. This is the case even if those accounts 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 and backups (p. 410).

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. 420)
- Creating a manual snapshot (p. 421)
- Changing the manual snapshot retention period (p. 421)
- Deleting manual snapshots (p. 421)
- Copying an automated snapshot (p. 421)
- Restoring a cluster from a snapshot (p. 422)
- Restoring a serverless namespace from a snapshot (p. 422)
- Sharing a cluster snapshot (p. 422)
- Configuring cross-Region snapshot copy for a nonencrypted cluster (p. 424)
- Configure cross-Region snapshot copy for an AWS KMS–encrypted cluster (p. 424)
- Modifying the retention period for cross-Region snapshot copy (p. 425)

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. 411).

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. 117).

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/](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](#).

**Restoring a serverless namespace from a snapshot**

Restoring a serverless namespace from a snapshot replaces all of the namespace's databases with databases in the snapshot. For more information about serverless snapshots, see [Working with snapshots and recovery points](#). Amazon Redshift automatically converts tables with interleaved keys into compound keys when you restore a provisioned cluster snapshot to a serverless namespace. For more information about sort keys, see [Working with sort keys](#).

To restore a snapshot from your provisioned cluster to your serverless namespace.

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, Snapshots**, then choose the snapshot to use.
3. Choose **Restore from snapshot, Restore to serverless namespace**.
4. Choose the namespace you want to restore to.
5. Confirm you want to restore from your snapshot. Choose **restore**. This action replaces all the databases in serverless namespace with the data from your provisioned cluster.

**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/](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 or accounts performing the restore. If the key isn't shared, attempting to restore the snapshot results in an access-denied error. When you authorize snapshot access and share the key, the identity authorizing access must have `kms:DescribeKey` permissions on the key used to encrypt the snapshot. This permission is described in more detail in [AWS KMS permissions](https://docs.aws.amazon.com/IAM/latest/UserGuide/s3 contrôle-access-key-key.html). For more information, see [DescribeKey](https://docs.aws.amazon.com/redshift/latest/api/API_DescribeKey.html) in the Amazon Redshift API reference documentation.

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.

For more information about giving access to an AWS KMS key, see [Allowing users in other accounts to use a KMS key](https://docs.aws.amazon.com/IAM/latest/UserGuide/s3 contrôle-access-key-key.html), in the AWS Key Management Service developer guide.
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. 447).

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/](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 and backups](p. 410).

For step-by-step instructions to run the following example, see [Running Java examples for Amazon Redshift using Eclipse](p. 584). You need to update the code and provide a cluster identifier.

**Example**

```java
/**

```
// snippet-sourcedescription:
/* CreateAndDescribeSnapshot demonstrates how to create an
Amazon Redshift cluster snapshot and describe existing snapshots. */
// snippet-service:[redshift]
// snippet-keyword:[Java]
// snippet-keyword:[Amazon Redshift]
// snippet-keyword:[Code Sample]
// snippet-keyword:[CreateClusterSnapshot]
// snippet-keyword:[DeleteClusterSnapshot]
// snippet-keyword:[DescribeClusterSnapshots]
// snippet-source:[full-example]
// snippet-sourcedate:[2019-01-30]
// snippet-sourceauthor:[AWS]
// snippet-start:[redshift.java.CreateAndDescribeSnapshot.complete]

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 {link com.amazonaws.auth.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) {
            System.err.println("Operation failed: " + e.getMessage());
        }
    }

    private static Date createManualSnapshot(String snapshotId) {
        CreateClusterSnapshotRequest request = new CreateClusterSnapshotRequest()
            .withClusterIdentifier(clusterIdentifier)
            .withSnapshotIdentifier(snapshotId);
        return new Date(createRedshiftClient().createClusterSnapshot(request).getSnapshotTime() - 8 * 3600000); // Convert to GMT
    }

    private static void waitForSnapshotAvailable(String snapshotId) {
        // Wait for snapshot to become available
    }

    private static void describeSnapshots() {
        DescribeClusterSnapshotsRequest request = new DescribeClusterSnapshotsRequest();
        DescribeClusterSnapshotsResult result = createRedshiftClient().describeClusterSnapshots(request);
        // Print out results
    }

    private static void deleteManualSnapshotsBefore(Date date) {
        DeleteClusterSnapshotsRequest request = new DeleteClusterSnapshotsRequest()
            .withSnapshotsToDelete(new DeleteClusterSnapshotsRequest.SnapshotIdentifier[]{snapshotId});
        createRedshiftClient().deleteClusterSnapshots(request);
    }

    private static AmazonRedshiftClient createRedshiftClient() {
        return client;
    }
}
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()
        .withClusterIdentifier(clusterIdentifier)
        .withEndTime(creationDate)
        .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\n", deleteResult.getSnapshotIdentifier());
    }
}

private static void printResultSnapshots(DescribeClusterSnapshotsResult result) {
    System.out.println("Snapshot listing:");
    for (Snapshot snapshot : result.getSnapshots()) {
        System.out.format("Identifier: %s\n", snapshot.getSnapshotIdentifier());
        System.out.format("Snapshot type: %s\n", snapshot.getSnapshotType());
        System.out.format("Snapshot create time: %s\n", snapshot.getSnapshotCreateTime());
        System.out.format("Snapshot status: %s\n\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(".");
            Thread.sleep(sleepTime*1000);
        }
    }
    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 and backups (p. 410).

Working with AWS Backup

AWS Backup is a fully managed service that helps you centralize and automate data protection across AWS services, in the cloud, and on premises.

Using AWS Backup for Amazon Redshift, you can configure data protection policies and monitor activity for different Amazon Redshift resources in one place. You can also create and store snapshots on Amazon Redshift provisioned clusters. This lets you automate and consolidate backup tasks that you had to do separately before, without any manual processes.

A backup, or *recovery point*, represents the content of a resource, such as an Amazon Redshift cluster, at a specified time. A backup generally refers to the different backups in AWS services, such as Amazon
Considerations for using AWS Backup with Amazon Redshift

The following sections describe considerations and limitations for using AWS Backup with Amazon Redshift.

Considerations for using AWS Backup with Amazon Redshift

Following are considerations for using AWS Backup with Amazon Redshift:

- AWS Backup for Amazon Redshift is available where both AWS Backup and Amazon Redshift are available in the same AWS Regions. For information on where AWS Backup is available, see Feature availability by AWS Regions.
- To get started using AWS Backup, verify that you have met all the prerequisites. For more information, see Prerequisites.
- Affirmatively opt in to AWS Backup service. Opt-in choices apply to the specific account and AWS Region. You might have to opt in to multiple Regions using the same account. For more information, see Getting started 1: Service Opt-in.
- From the Amazon Redshift console, you can create manual and automated snapshots. AWS Backup only supports manual snapshots at this time.
- Once you use AWS Backup to manage snapshot settings, you can't continue to manage manual snapshot settings using Amazon Redshift. Instead, you can continue to manage the settings using an AWS Backup plan. For more information, see Managing backups using backup plans.
- To save storage costs when you have versioning-enabled Amazon S3 buckets to be backed up, we recommend that you set a lifecycle expiration rule. For information on specifying a lifecycle rule, see Example 6: Specifying a lifecycle rule for a versioning-enabled bucket. If you don't set a lifecycle expiration period, your Amazon Redshift storage costs might increase, because AWS Backup retains all versions of your Amazon Redshift data.

Limitations

Following are limitations for using AWS Backup in Amazon Redshift:

- You can't use AWS Backup to manage Amazon Redshift automated snapshots. To manage automated snapshots, use tags. For information about tagging resources, see Tagging resources in Amazon Redshift.
- AWS Backup doesn't support Amazon Redshift Serverless.
Managing AWS Backup with Amazon Redshift

To protect resources on your Amazon Redshift provisioned clusters, you can use the AWS Backup console, or programmatically use the AWS Backup API or AWS Command Line Interface (AWS CLI). When you need to recover a resource, you can use either the AWS Backup console or the AWS CLI to find and recover the resource you need. For more information, see AWS Command Line Interface.

When using AWS Backup for Amazon Redshift, you can perform the following actions:

- Create periodic backups that automatically initiate Amazon Redshift snapshots. Periodic backups are useful to meet your long-term data retention needs. For more information, see Amazon Redshift backups.
- Automate backup scheduling and retention by centrally configuring backup plans.
- Restore a cluster to the saved backup you choose. You set how often to back up your resources. For more information, see Restore an Amazon Redshift cluster.
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. Before completing integration with Informatica, you must add the partner's IP addresses to the allowlist of inbound traffic. The following AWS Partners can integrate with Amazon Redshift:

- Datacoral
- Etleap
- Fivetran
- SnapLogic
- Stitch
- Upsolver
- Matillion (preview)
- Sisense (preview)
- Thoughtspot

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/
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.

### Loading data with AWS partners

Aside from integrating a partner with an Amazon Redshift cluster, you can also move data from more than 30 sources into your Amazon Redshift cluster using our partner’s data loading tools. Before you do so, you must add the partner’s IP addresses (found below) to the allowlist of inbound rules. For more information about adding rules to an Amazon EC2 security group, see Authorizing Inbound Traffic for Your Instances in the Amazon EC2 User Guide for Linux Instances. Note that while the Informatica Data Loader tool is free, data ingress charges might apply depending on the data sources and targets you choose.

You can load data from the following partners:

- Informatica – IP addresses

#### To integrate an Amazon Redshift cluster with Informatica

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 AWS partner integration, then choose the partner you want to integrate your cluster with.
3. Choose Complete <partner-name> integration. You will be redirected to the partner’s integration site.
4. Enter the necessary details on the partner’s site and complete the integration.
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.

**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. 433).

#### Comparing reserved node offerings

<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 or three-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>

Specific options and durations are subject to availability.

**Note**

If you previously purchased **Heavy Utilization** offerings for Amazon Redshift, the comparable offering is the **Partial Upfront** offering.
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
Reserved node examples

- 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. 435).

**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.

You can't convert all node types to reserved nodes, and it's also possible that an existing reserved node isn't available for renewal. This might be because the node type is discontinued. Contact customer support to renew a discontinued node type.

**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": 31536000,
      "ReservedNodeOfferingId": "yyyyyyyy-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
    }
  ]
}
```


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
   }
}
```

You can confirm that the exchange is complete by calling `describe-reserved-nodes` and checking the value for `NodeType`.

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. 584). 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
 * specific language governing permissions and limitations under the License.
 */

// snippet-sourcedescription:[ListAndPurchaseReservedNodeOffering demonstrates how to list
and purchase Amazon Redshift reserved node offerings.]
// snippet-service:[redshift]
// snippet-keyword:[Java]
// snippet-keyword:[Amazon Redshift]
// snippet-keyword:[Code Sample]
// snippet-keyword:[DescribeReservedNodeOfferings]
// snippet-keyword:[PurchaseReservedNodeOffering]
// snippet-keyword:[ReservedNode]
// snippet-sourcetype:[full-example]
// snippet-sourcedate:[2019-01-31]
// snippet-sourceauthor:[AWS]
// snippet-start:[redshift.java.ListAndPurchaseReservedNodeOffering.complete]

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} object
        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();

        // Add code here to list reserved nodes
    }

    private static void findReservedNodeOffer() {
        // Add code here to find reserved node offer
    }

    private static void purchaseReservedNodeOffer() {
        // Add code here to purchase reserved node offer
    }

// snippet-end:[redshift.java.ListAndPurchaseReservedNodeOffering.complete]
```
Amazon Redshift Management Guide
Purchasing a reserved node offering using Java

```
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;
    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]?");
            DataInput 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) {
                }
            }
        }
    }
}
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`

You can't convert all node types to reserved nodes, and it's also possible that an existing reserved node isn't available for renewal. This might be because the node type is discontinued.
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 user or account associated with your AWS security credentials. 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.

  **Important**
  Amazon Redshift has a collection of best practices for managing permissions, identities and secure access. We recommend that you get familiar with these as you get started with Amazon Redshift. For more information, see Identity and access management in Amazon Redshift (p. 455).

- **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. 577).

- **Database access** – The ability to access database objects, such as tables and views, is controlled by database 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. 500).
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.

Topics
- Data protection in Amazon Redshift (p. 443)
- Identity and access management in Amazon Redshift (p. 455)
- Logging and monitoring in Amazon Redshift (p. 559)
- Compliance validation for Amazon Redshift (p. 575)
- Resilience in Amazon Redshift (p. 576)
- Infrastructure security in Amazon Redshift (p. 576)
- Configuration and vulnerability analysis in Amazon Redshift (p. 581)

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 users with AWS IAM Identity Center (successor to AWS Single Sign-On) or 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 require TLS 1.2 and recommend TLS 1.3.
- 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 sensitive 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 text 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 text 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.

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)**. 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/aws-cloudtrail/latest/using/aws-kms.html) in the **AWS Key Management Service Developer Guide**.

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](https://docs.aws.amazon.com/redshift/latest/guide/encrypted-cluster.html).

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
Data encryption 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.

Encryption process improvements for better performance and availability

Encryption with RA3 nodes

Updates to the encryption process for RA3 nodes have made the experience much better. Both read and write queries can run during the process with less performance impact from the encryption. Also, encryption finishes much more quickly. The updated process steps include a restore operation and migration of cluster metadata to a target cluster. The improved experience applies to encryption types like AWS KMS, for example. When you have petabyte-scale data volumes, the operation has been reduced from weeks to days.

Prior to encrypting your cluster, if you plan to continue to run database workloads, you can improve performance and speed up the process by adding nodes with elastic resize. You can’t use elastic resize when encryption is in process, so do it before you encrypt. Note that adding nodes typically results in higher cost.

Encryption with other node types

When you encrypt a cluster with DC2 or DS2 nodes, you don’t have the ability to run write queries, like with RA3 nodes. Only read queries can be run.

Usage notes for encryption with RA3 nodes

The following insights and resources help you prepare for encryption and monitor the process.

- **Running queries after starting encryption** – After encryption is started, reads and writes are available within about fifteen minutes. How long it takes the full encryption process to complete depends on the amount of data on the cluster and the workload levels.

- **Monitoring encryption** – To monitor the process as it’s running and determine how much data remains to be encrypted, run the following query:

  ```sql
  select stv_slices.node, count(*) from stv_blocklist, stv_slices
  where stv_slices.slice = stv_blocklist.slice and tbl > 100000 and tombstone = 0 and flags & 67108864 <> 0 group by stv_slices.node order by 1;
  ```

  Note that the monitoring query applies only to RA3-node clusters.

- **How long does encryption take?** – The time to encrypt your data depends on several factors: These include the number of workloads running, the compute resources being used, the number of nodes, and the type of nodes. We recommend that you initially perform encryption in a test environment. As a rule of thumb, if you’re working with data volumes in petabytes, it likely can take 1-3 days for encryption to complete.

- **Rolling back encryption** – If you need to roll back the encryption operation, the best way to do this is to restore from the most recent backup taken prior to when encryption was initiated. You will have to re-apply any new updates (updates/deletes/inserts) following the last-backup.
• **Performing a table restore** – Note that you can't restore a table from an unencrypted cluster to an encrypted cluster.

• **Encrypting a single-node cluster** – Encrypting a single-node cluster has performance limitations. It takes longer than encryption for a multi-node cluster.

• **Creating a backup after encryption** – When you encrypt the data in your cluster, a backup isn't created until the cluster is fully encrypted. The amount of time this takes can vary. The time taken for backup can be hours to days, depending on the cluster size. After encryption completes, there can be a delay before you can create a backup.

Note that because a backup-and-restore operation occurs during the encryption process, any tables or materialized views created with `BACKUP NO` aren't retained. For more information, see [CREATE TABLE](#) or [CREATE MATERIALIZED VIEW](#).

**Topics**

- Database encryption for Amazon Redshift using AWS KMS (p. 446)
- Encryption for Amazon Redshift using hardware security modules (p. 448)
- Encryption key rotation in Amazon Redshift (p. 449)
- Changing cluster encryption (p. 449)
- Configuring database encryption using the console (p. 452)
- Configuring database encryption using the Amazon Redshift API and AWS CLI (p. 452)

**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. 112) and Managing clusters using the AWS CLI and Amazon Redshift API (p. 118).

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. 414).

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. 424).

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. 452).

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](https://aws.amazon.com/cloudhsm/pricing). AWS CloudHSM Classic isn't available in all AWS Regions. For more information about available AWS Regions, see [AWS Region Table](https).

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
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. 452) and Rotating encryption keys using the Amazon Redshift API and AWS CLI (p. 453).

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. 414) and Configure cross-Region snapshot copy for an AWS KMS–encrypted cluster (p. 424). 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. 450).

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.

```bash
aws redshift modify-cluster --cluster-identifier <value> --encrypted --kms-key-id <value>
```

To remove encryption from your cluster, run the following CLI command.

```bash
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. 444).

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.

   ```sql
   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.

   ```sql
   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. 603).
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. 444).

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. 603).

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. 112) and Managing clusters using the AWS CLI and Amazon Redshift API (p. 118).

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. 444).

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
Data encryption

- 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.
- 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. 303).
- 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. 306).
- 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 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. 448). 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. 449).

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
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.
- Enable “Enhanced VPC routing” for an Amazon Redshift cluster, specifying a target Amazon S3 VPC endpoint. Traffic generated by Amazon Redshift COPY, UNLOAD, or CREATE LIBRARY commands are then routed through the private endpoint. For more information, see Enhanced VPC routing (p. 384).

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:

- Authenticaing with identities (p. 456)
- Access control (p. 458)
Important
This topic contains a collection of best practices for managing permissions, identities and secure access. We recommend that you get familiar with best practices for using IAM with Amazon Redshift. These include using IAM roles for applying permissions. Having a good understanding of these sections will help you maintain a more secure Amazon Redshift data warehouse.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (successor to AWS Single Sign-On) (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the AWS Sign-In User Guide.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signing AWS API requests in the IAM User Guide.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide and Using multi-factor authentication (MFA) in AWS in the IAM User Guide.

AWS account root user

When you create an AWS account, you begin with one 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 don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the AWS Account Management Reference Guide.

IAM users and groups

An IAM user is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the IAM User Guide.

An IAM group is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named IAMAdmins and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but
roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the IAM User Guide.

IAM roles

An IAM role is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the IAM User Guide.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the IAM User Guide. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide.

- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the IAM User Guide.

- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.

- **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see Actions, resources, and condition keys for Amazon Redshift in the Service Authorization Reference.

- **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the IAM User Guide.

- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the IAM User Guide.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) 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. 458)
- Using identity-based policies (IAM policies) for Amazon Redshift (p. 463)

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>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>
</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. 496).

### 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>
<tr>
<td>Query</td>
<td>arn:aws:sqlworkbench:region:account-id:query/</td>
</tr>
<tr>
<td>Chart</td>
<td>arn:aws:sqlworkbench:region:account-id:chart/</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:

- **CIDR/IP address**

- **EC2 security group**

- **Event subscription**
  - `arn:aws:redshift:region:account-id:eventsubscription:event-subscription-name`

- **Hardware security module (HSM) client certificate**

- **HSM configuration**

- **Parameter group**
  - `arn:aws:redshift:region:account-id:parametergroup:parameter-group-name`

- **Redshift-managed VPC endpoint**
  - `arn:aws:redshift:region:account-id:managedvpcendpoint:endpoint-name`

- **Snapshot**

- **Snapshot copy grant**

- **Subnet group**
  - `arn:aws:redshift:region:account-id:subnetgroup:subnet-group-name`
• 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 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.

• 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. In most cases this method isn't recommended. We recommend creating an IAM role and attaching permissions to the role, then assigning the role to a user.

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 assign permissions by attaching polices to an IAM role and then assigning that role to a user or group. The following is an example policy that containing permissions 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. 463). 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. 458)), 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. 458).
- **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. 496).

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. 461). 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. 496).

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 resource. For more information, see aws:RequestTag in the IAM User Guide.</td>
</tr>
</tbody>
</table>
### Overview of managing access

<table>
<thead>
<tr>
<th>Condition key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aws:ResourceTag</td>
<td>Restricts user access to resources based on specific tag keys and values. For more information, see <a href="https://docs.aws.amazon.com/IAM/latest/UserGuide/index.html">aws:ResourceTag</a> 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 specify in the policy. For more information, see <a href="https://docs.aws.amazon.com/IAM/latest/UserGuide/index.html">aws:TagKeys</a> in the IAM User Guide.</td>
</tr>
</tbody>
</table>

For information on tags, see [Tagging overview (p. 651)](https://docs.aws.amazon.com/IAM/latest/UserGuide/index.html).

For a list of the API actions that support the `redshift:ResourceTag` and `redshift:RequestTag` condition keys, see [Amazon Redshift API permissions reference (p. 496)](https://docs.aws.amazon.com/redshift/latest/api/iam-permissions-reference.html).

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>Limits 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:CreateClusterInstance"
      ],
      "Condition": {
         "StringEquals": {
            "aws:TagKeys": [ "costcenter", "usage" ]
         },
         "ForAllValues": true
      }
   }
}
```
Using identity-based policies (IAM policies) for Amazon Redshift

Managing access permissions to resources for Amazon Redshift Serverless

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. 458).

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 in AWS Region us-west-2 and AWS account 123456789012.

```
{
  "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"
      ],
```
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 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 roles 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 and AmazonRedshiftReadOnlyAccess in the AWS Managed Policy Reference Guide.

#### AmazonRedshiftFullAccess

Grants full access to all Amazon Redshift resources for an AWS account. Additionally, this policy grants full access to all Amazon Redshift Serverless resources.

You can find the AmazonRedshiftFullAccess policy on the IAM console and AmazonRedshiftFullAccess in the AWS Managed Policy Reference Guide.

#### AmazonRedshiftQueryEditor

Grants full access to the query editor on the Amazon Redshift console.

You can find the AmazonRedshiftQueryEditor policy on the IAM console and AmazonRedshiftQueryEditor in the AWS Managed Policy Reference Guide.

#### 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 and AmazonRedshiftDataFullAccess in the AWS Managed Policy Reference Guide.
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 and AmazonRedshiftQueryEditorV2FullAccess in the AWS Managed Policy Reference Guide.

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 and AmazonRedshiftQueryEditorV2NoSharing in the AWS Managed Policy Reference Guide.

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 and AmazonRedshiftQueryEditorV2ReadSharing in the AWS Managed Policy Reference Guide.

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 and AmazonRedshiftQueryEditorV2ReadWriteSharing in the AWS Managed Policy Reference Guide.

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.

You can find the AmazonRedshiftServiceLinkedRolePolicy policy on the IAM console and AmazonRedshiftServiceLinkedRolePolicy in the AWS Managed Policy Reference Guide.

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 and AmazonRedshiftAllCommandsFullAccess in the AWS Managed Policy Reference 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 roles 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><strong>AmazonRedshiftServiceLinkedRolePolicy</strong></td>
<td>- Update to an existing policy</td>
<td>May 08, 2023</td>
</tr>
<tr>
<td></td>
<td>Permissions for actions on Amazon EC2 to create and manage security groups and routing rules are removed from the managed policy. These permissions pertained to creating subnets and VPCs. Removed permissions are the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:AuthorizeSecurityGroupEgress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:AuthorizeSecurityGroupIngress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:UpdateSecurityGroupRuleDescriptionsEgress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:ReplaceRouteTableAssociation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:CreateRouteTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:AttachInternetGateway</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:UpdateSecurityGroupRuleDescriptionsIngress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:AssociateRouteTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:RevokeSecurityGroupIngress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:CreateRoute</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:CreateSecurityGroup</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:RevokeSecurityGroupEgress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:ModifyVpcAttribute</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:CreateSubnet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:CreateInternetGateway</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ec2:CreateVpc</td>
<td></td>
</tr>
<tr>
<td></td>
<td>These were associated with the <em>Purpose:RedshiftMigrateToVpc</em> resource tag. The tag limited the scope of permissions to tasks for Amazon EC2 Classic to Amazon EC2 VPC migration. For more information about resource tags, see <a href="#">Controlling access to AWS resources using tags</a>.</td>
<td></td>
</tr>
<tr>
<td><strong>AmazonRedshiftDataFullAccess</strong></td>
<td>- Update to an existing policy</td>
<td>April 7, 2023</td>
</tr>
<tr>
<td></td>
<td>Permission for the action <code>redshift:GetClusterCredentialsWithIAM</code> is added to the managed policy. Adding it grants permission</td>
<td></td>
</tr>
</tbody>
</table>

466
<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AmazonRedshiftServiceLinkedRolePolicy</strong></td>
<td>Update to an existing policy</td>
<td>April 06, 2023</td>
</tr>
<tr>
<td><strong>AmazonRedshiftQueryEditorV2NoSharing</strong></td>
<td>Update to an existing policy</td>
<td>March 21, 2023</td>
</tr>
<tr>
<td><strong>AmazonRedshiftQueryEditorV2ReadSharing</strong></td>
<td>Update to an existing policy</td>
<td>March 21, 2023</td>
</tr>
<tr>
<td><strong>AmazonRedshiftQueryEditorV2ReadWriteSharing</strong></td>
<td>Update to an existing policy</td>
<td>March 21, 2023</td>
</tr>
<tr>
<td><strong>AmazonRedshiftQueryEditorV2NoSharing</strong></td>
<td>Update to an existing policy</td>
<td>February 2, 2023</td>
</tr>
<tr>
<td><strong>AmazonRedshiftQueryEditorV2ReadSharing</strong></td>
<td>Update to an existing policy</td>
<td>February 2, 2023</td>
</tr>
</tbody>
</table>

To get enhanced temporary credentials to access an Amazon Redshift database by the specified AWS account.

- Permissions for the actions on Amazon EC2 for creation and management of security group rules are added to the managed policy. These security groups and rules are specifically associated with the Amazon Redshift `aws:RequestTag/Redshift` resource tag. This limits the scope of the permissions to specific Amazon Redshift resources.

- Permission for the action `sqlworkbench:GetSchemaInference` is added to the managed policy. Adding it grants permission to get the columns and data types inferred from a file.

- Permission for the action `sqlworkbench:AssociateNotebookWithTab` is added to the managed policy. Adding it grants permission to create and update tabs linked to a user's own notebook or to a notebook that is shared with them.
<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing</td>
<td>Permission for the action sqlworkbench:AssociateNotebookWithTab is added to the managed policy. Adding it grants permission to create and update tabs linked to a user's own notebook or to a notebook that is shared with them.</td>
<td>February 2, 2023</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2NoSharing</td>
<td>To grant permission to use notebooks, Amazon Redshift added permission for the following actions:</td>
<td>October 17, 2022</td>
</tr>
</tbody>
</table>
|                                          | - sqlworkbench:ListNotebooks  
|                                          | - sqlworkbench:CreateNotebook  
|                                          | - sqlworkbench:DuplicateNotebook  
|                                          | - sqlworkbench:CreateNotebookFromVersion  
|                                          | - sqlworkbench:ImportNotebook  
|                                          | - sqlworkbench:GetNotebook  
|                                          | - sqlworkbench:UpdateNotebook  
|                                          | - sqlworkbench:DeleteNotebook  
|                                          | - sqlworkbench:CreateNotebookCell  
|                                          | - sqlworkbench:DeleteNotebookCell  
|                                          | - sqlworkbench:UpdateNotebookCellContent  
|                                          | - sqlworkbench:UpdateNotebookCellLayout  
|                                          | - sqlworkbench:BatchGetNotebookCell  
|                                          | - sqlworkbench:ListNotebookVersions  
|                                          | - sqlworkbench:CreateNotebookVersion  
|                                          | - sqlworkbench:GetNotebookVersion  
|                                          | - sqlworkbench:DeleteNotebookVersion  
|                                          | - sqlworkbench:RestoreNotebookVersion  
|                                          | - sqlworkbench:ExportNotebook  
<p>|</p>
<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadSharing</td>
<td>Update to an existing policy to grant permission to use notebooks, Amazon Redshift added permission for the following actions:</td>
<td>October 17, 2022</td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:ListNotebooks</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:CreateNotebook</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:DuplicateNotebook</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:CreateNotebookFromVersion</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:ImportNotebook</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:GetNotebook</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:UpdateNotebook</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:DeleteNotebook</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:CreateNotebookCell</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:DeleteNotebookCell</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:UpdateNotebookCellContent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:UpdateNotebookCellLayout</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:BatchGetNotebookCell</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:ListNotebookVersions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:CreateNotebookVersion</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:GetNotebookVersion</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:DeleteNotebookVersion</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:RestoreNotebookVersion</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• sqlworkbench:ExportNotebook</td>
<td></td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
</tbody>
</table>
| AmazonRedshiftQueryEditorV2ReadWriteSharing | To grant permission to use notebooks, Amazon Redshift added permission for the following actions:  
  - sqlworkbench:ListNotebooks  
  - sqlworkbench:CreateNotebook  
  - sqlworkbench:DuplicateNotebook  
  - sqlworkbench:CreateNotebookFromVersion  
  - sqlworkbench:ImportNotebook  
  - sqlworkbench:GetNotebook  
  - sqlworkbench:UpdateNotebook  
  - sqlworkbench:DeleteNotebook  
  - sqlworkbench:CreateNotebookCell  
  - sqlworkbench:DeleteNotebookCell  
  - sqlworkbench:UpdateNotebookCellContent  
  - sqlworkbench:UpdateNotebookCellLayout  
  - sqlworkbench:BatchGetNotebookCell  
  - sqlworkbench:ListNotebookVersions  
  - sqlworkbench:CreateNotebookVersion  
  - sqlworkbench:GetNotebookVersion  
  - sqlworkbench:DeleteNotebookVersion  
  - sqlworkbench:RestoreNotebookVersion  
  - sqlworkbench:ExportNotebook | October 17, 2022 |
<p>| AmazonRedshiftServiceLinkedRolePolicy | Amazon Redshift added the namespace AWS/Redshift to allow publishing metrics to CloudWatch.                                                                                                           | September 7, 2022 |
| AmazonRedshiftQueryEditorV2NoSharing | Amazon Redshift added permission to the actions sqlworkbench:ListQueryExecutionHistory and sqlworkbench:GetQueryExecutionHistory. This grants permission to see query history. | August 30, 2022  |
| AmazonRedshiftQueryEditorV2ReadSharing | Amazon Redshift added permission to the actions sqlworkbench:ListQueryExecutionHistory and sqlworkbench:GetQueryExecutionHistory. This grants permission to see query history. | August 30, 2022  |</p>
<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing</td>
<td>Amazon Redshift added permission to the actions sqlworkbench:ListQueryExecutionHistory and sqlworkbench:GetQueryExecutionHistory. This grants permission to see query history.</td>
<td>August 30, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftFullAccess (p. 464)</td>
<td>Permissions for Amazon Redshift Serverless are added to the existing AmazonRedshiftFullAccess managed policy.</td>
<td>July 22, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftDataFullAccess (p. 464)</td>
<td>Amazon Redshift updated redshift-serverless:GetCredentials default scoping condition of tag aws:ResourceTag/RedshiftDataFullAccess permission from StringEquals to StringLike to grant access to resources tagged with tag key RedshiftDataFullAccess and any tag value.</td>
<td>July 11, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftDataFullAccess (p. 464)</td>
<td>Amazon Redshift added new permissions to allow redshift-serverless:GetCredentials for temporary credentials to Amazon Redshift Serverless.</td>
<td>July 8, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2NoSharing</td>
<td>Amazon Redshift added permission to the action sqlworkbench:GetAccountSettings. This grants permission to get account settings.</td>
<td>June 15, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadSharing</td>
<td>Amazon Redshift added permission to the action sqlworkbench:GetAccountSettings. This grants permission to get account settings.</td>
<td>June 15, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing</td>
<td>Amazon Redshift added permission to the action sqlworkbench:GetAccountSettings. This grants permission to get account settings.</td>
<td>June 15, 2022</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>AmazonRedshiftServiceLinkedRolePolicy</td>
<td>To enable public access to new Amazon Redshift Serverless endpoints, Amazon Redshift allocates and associates Elastic IP addresses to the VPC endpoint's Elastic network interface in the customer account. It does this via permissions provided through the service linked role. To enable this use case, actions to allocate and release an Elastic IP address are added to the Amazon Redshift Serverless service linked role.</td>
<td>May 26, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2FullAccess</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>February 22, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2NoSharing</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>February 22, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing</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>February 22, 2022</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadSharing</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>February 22, 2022</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing</td>
<td>Amazon Redshift added a new policy to allow read and update sharing within Amazon Redshift query editor v2.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadSharing</td>
<td>Amazon Redshift added a new policy to allow read sharing within Amazon Redshift query editor v2.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2NoSharing</td>
<td>Amazon Redshift added a new policy to allow using Amazon Redshift query editor v2 without sharing resources.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2FullAccess</td>
<td>Amazon Redshift added a new policy to allow full access to Amazon Redshift query editor v2.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadWriteSharing</td>
<td>Amazon Redshift added a new policy to allow read and update sharing within Amazon Redshift query editor v2.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2NoSharing</td>
<td>Amazon Redshift added a new policy to allow using Amazon Redshift query editor v2 without sharing resources.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadSharing</td>
<td>Amazon Redshift added a new policy to allow read sharing within Amazon Redshift query editor v2.</td>
<td>September 24, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftServiceLinkedRolePolicy</td>
<td>Amazon Redshift added permissions for new actions to allow management of Amazon Redshift network and VPC resources.</td>
<td>November 15, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftAllCommandsFullAccess</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>November 18, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftServiceLinkedRolePolicy</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>November 22, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftFullAccess (p. 464)</td>
<td>Amazon Redshift added new permissions to allow model explainability, DynamoDB, Redshift Spectrum, and Amazon RDS federation.</td>
<td>October 07, 2021</td>
</tr>
<tr>
<td>AmazonRedshiftQueryEditorV2ReadSharing</td>
<td>Amazon Redshift added an action sqlworkbench:AssociateQueryWithTab to the managed policy. Adding it allows customers to create editor tabs linked to a query that is shared with them.</td>
<td>February 22, 2022</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 policies for Amazon Redshift (p. 464).

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. 474).

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": "DataAPIIAMSessionsPermissionsRestriction",
"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 (the 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 team. 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.

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. 500) and Identity providers and federation in the IAM User Guide.

The sqlworkbench:CreateNotebookVersion grants permission to get the current content of notebook cells and create a notebook version on your account. Meaning, at the time of version creation, the current content of the notebook is the same as the version's content. Later on, the content of the cells in the version stay the same as the current notebook is updated. The sqlworkbench:GetNotebookVersion grants permission to get a version of the notebook. A user who doesn't have sqlworkbench:BatchGetNotebookCell permission but has sqlworkbench:CreateNotebookVersion and sqlworkbench:GetNotebookVersion permissions on a notebook has access to notebook cells in the version. This user without the sqlworkbench:BatchGetNotebookCell permission is still able to retrieve the content of a notebook's cells by first creating a version and then getting this created version.

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.

```json
{
   "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)](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html) 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",
            
            "events.amazonaws.com",
            
            "redshift.amazonaws.com"
            ],
         "Action": "sts:AssumeRole"
      }
   ]
}
```
Using identity-based policies (IAM policies)

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)

Following, you can find a description of the permissions required to use Amazon Redshift machine learning (ML) 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:DescribeAutoMLJob",
                "sagemaker:DescribeTrainingJob",
                "sagemaker:DescribeCompilationJob",
                "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",
    "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"
  ]
}
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:CreateAutoMLJob",
                "sagemaker:CreateCompilationJob",
                "sagemaker:CreateModel",
                "sagemaker:CreateEndpoint",
                "sagemaker:CreatePredictor",
                "sagemaker:StartNotebookInstanceLifecycleConfiguration",
                "sagemaker:StartImageBuilder",
                "sagemaker:StartMLFeatureGroup",
                "sagemaker:StartNotebookInstanceLifecycleConfiguration",
                "sagemaker:StartImageBuilder",
                "sagemaker:StartMLFeatureGroup"
            ],
            "Resource": "arn:aws:sagemaker:*",
            "Condition": {
                "StringEquals": {
                    "sagemaker:ExecutionArn": "*"
                }
            }
        }
    ]
}
```
Using identity-based policies (IAM policies)

```
"sagemaker:CreateEndpoint",
"sagemaker:DescribeAutoMLJob",
"sagemaker:DescribeTrainingJob",
"sagemaker:DescribeCompilationJob",
"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",
  "ecr:GetAuthorizationToken",
  "ecr:GetDownloadUrlForLayer"
]```
Using identity-based policies (IAM policies)

- **Policy 1:**
  - "Effect": "Allow",

- **Policy 2:**
  - "Effect": "Allow",
  - "Action": ["dynamodb:Scan", "dynamodb:DescribeTable", "dynamodb:GetItem"],
  - "Resource": ["arn:aws:dynamodb::*:*:table/*redshift*", "arn:aws:dynamodb::*:*:table/*redshift*/index/**"]

- **Policy 3:**
  - "Effect": "Allow",
  - "Resource": "*",
  - "Condition": {
    "StringEqualsIgnoreCase": {
      "s3:ExistingObjectTag/Redshift": "true"
    }
  }

- **Policy 4:**
  - "Effect": "Allow",
  - "Resource": "$REDSHIFT_BUCKET ARN$"
Amazon Redshift Management Guide
Using identity-based policies (IAM policies)

},
{
   "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",
      "secretsmanager:ListSecretVersionIds"
   ]
},

483
Using identity-based policies (IAM policies)

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"
      ]
    }
  ]
}
```
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.

```json
{
    "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.

```json
{
    "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": "*"
        }
    ]
}
```
For more information about the AmazonSageMakerFullAccess managed policy, see AmazonSageMakerFullAccess in the Amazon SageMaker Developer Guide.

If you want to create Forecast models, we recommend that you use the AmazonForecastFullAccess managed policy. If you want to use a more restrictive policy, add the following policy to your IAM role.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:CreateAutoPredictor",
                "forecast:CreateDataset",
                "forecast:CreateDatasetGroup",
                "forecast:CreateDatasetImportJob",
                "forecast:CreateForecast",
                "forecast:CreateForecastExportJob",
                "forecast:DeleteResourceTree",
                "forecast:DescribeAutoPredictor",
                "forecast:DescribeDataset",
                "forecast:DescribeDatasetGroup",
                "forecast:DescribeDatasetImportJob",
                "forecast:DescribeForecast",
                "forecast:DescribeForecastExportJob",
                "forecast:StopResource",
                "forecast:TagResource",
                "forecast:UpdateDatasetGroup"
            ],
            "Resource": "*"
        }
    ]
}
```

For more information about Amazon Redshift ML, see Using machine learning in Amazon Redshift or CREATE MODEL.

Permissions for streaming ingestion

Streaming ingestion works with two services. These are Kinesis Data Streams and Amazon MSK.

Permissions required to use streaming ingestion with Kinesis Data Streams

A procedure with a managed-policy example is available at Getting started with streaming ingestion from Amazon Kinesis Data Streams.

Permissions required to use streaming ingestion with Amazon MSK

A procedure with a managed-policy example is available at Getting started with streaming ingestion from Amazon Managed Streaming for Apache Kafka.

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.

```
```

You can restrict AuthorizeDataShare access to a specific datashare by specifying the datashare name in the IAM policy.

```
{
  "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.

```
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.

```
{
  "Statement": [
    {
      "Action": [
        "redshift:AuthorizeDataShare",
      ],
      "Resource": [
        "arn:aws:redshift:us-east-1:555555555555:datashare:86b5169f-01dc-4a6f-9fbb-e2e24359e9a8/*"
      ],
      "Effect": "Deny"
    }
  ]
}
```

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>GetClusterCredential parameter</th>
<th>Action</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocreate</td>
<td>redshift:CreateClusterUser</td>
<td></td>
</tr>
<tr>
<td>DbGroups</td>
<td>redshift:JoinGroup</td>
<td></td>
</tr>
<tr>
<td>DbName</td>
<td>NA</td>
<td>dbname</td>
</tr>
</tbody>
</table>

For more information about resources, see Amazon Redshift resources and operations (p. 458).

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. 461).

**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.

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

```
[  
  "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`.

```
[  
  "Version": "2012-10-17",
  "Statement": [
    
    "Sid":"AllowClusterManagement",
    "Action": [
    "redshift:CreateCluster",
    "redshift:DeleteCluster",
    "redshift:ModifyCluster",
    "redshift:RebootCluster"
    ],
    "Resource": [  
    "*"
    ],
    "Effect": "Allow"
    ],
    "Sid":"DenyDeleteProtected",
    "Action": [
    "redshift:DeleteCluster"
    ],
    "Resource": [  
    ],
    "Effect": "Deny"
    ]
]
```
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.

```
{
  "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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDescribeSnapshots",
      "Action": ["redshift:DescribeClusterSnapshots"],
      "Resource": ["*"],
      "Effect": "Allow"
    },
    {
      "Sid": "AllowUserRestoreFromSnapshot",
      "Action": ["redshift:RestoreFromClusterSnapshot"],
      "Resource": [
```
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"
         ],
         "Resource": [
         ],
         "Effect": "Allow"
      },
      {
         "Sid": "AllowRestoreFromClusterSnapshot",
         "Action": [
            "redshift:RestoreFromClusterSnapshot"
         ],
         "Resource": [
         ],
         "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.

**Note**
Resource-level permissions are not supported for the Amazon EC2 actions that are specified in this example policy.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "AllowRedshift",
         "Effect": "Allow",
         "Action": [
            "redshift:*"
         ],
         "Resource": [
         ]
      },
      {
         "Sid": "AllowCopyClusterSnapshot",
         "Action": [
            "redshift:CopyClusterSnapshot"
         ],
         "Resource": [
         ],
         "Effect": "Allow"
      },
      {
         "Sid": "AllowRestoreFromClusterSnapshot",
         "Action": [
            "redshift:RestoreFromClusterSnapshot"
         ],
         "Resource": [
         ],
         "Effect": "Allow"
      }
   ]
}
```
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. 651).

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Tagging permissions",
            "Effect": "Allow",
            "Action": [
                "redshift:DeleteTags",
                "redshift:CreateTags",
                "redshift:DescribeTags",
                "tag:UntagResources"
            ],
            "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:${redshift:DbUser}@yourdomain.com"
                }
            }
        },
        {
            "Sid": "CreateClusterUserStatement",
            "Effect": "Allow",
            "Action": [
                "redshift:CreateClusterUser"
            ],
            "Resource": [
            ],
            "Condition": {  
                "StringEquals": {
                    "aws:userid":"AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}@yourdomain.com"
                }
            }
        },
        {
            "Sid": "RedshiftJoinGroupStatement",
            "Effect": "Allow",
            "Action": [
                "redshift:JoinGroup"
            ],
            "Resource": [
            ],
            "Condition": {  
                "StringEquals": {
                    "aws:userid":"AIDIODR4TAW7CSEXAMPLE:${redshift:DbUser}@yourdomain.com"
                }
            }
        }
    ]
}
```
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
NAMESPACE 'aad'
PARAMETERS '{
  "issuer" : "https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
  "client_id" : "<client_id>",
  "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.

Instead of using a shared client secret, you can set parameters to specify a certificate, a private key, and a private key password when you create the identity provider.

```sql
CREATE IDENTITY PROVIDER example_idp TYPE azure
NAMESPACE 'example_aad'
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id": "<client_id>",
"audience": ["https://analysis.windows.net/powerbi/connector/AmazonRedshift"],
"client_x5t": "<certificate thumbprint>",
"client_pk_base64": "<private key in base64 encoding>",
"client_pk_password": "test_password"}';
```

The private key password, `client_pk_password`, is optional.

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 `rsgroup` and a namespace called `aad`, the role name is `aad:rsgroup`. The user name and group memberships are mapped to a user and roles in Amazon Redshift, in the identity provider namespace.

The mapping for roles and users includes verifying their `external_id` value, to ensure it's up to date. The external ID maps to the identifier of the group or user in the identity provider. For example, a role's external ID maps to the corresponding Azure AD group ID. Similarly, each user's external ID maps to their ID in the identity provider.

```sql
create role "aad:rsgroup";
```

3. Grant relevant permissions to roles per your requirements. For example:

```sql
GRANT SELECT on all tables in schema public to role "aad:rsgroup";
```

4. You can also grant permissions to a specific user.

```sql
GRANT SELECT on table foo to aad:alice@example.com
```

Note that a federated external user's role membership is available only in that user's session. This has implications for creating database objects. When a federated external user creates any view or stored procedure, for instance, the same user can't delegate permission of those objects to other users and roles.

### An explanation of namespaces

A namespace maps a user or role to a specific identity provider. For example, the prefix for users created in AWS IAM is `iam:`. This prefix prevents user name collisions and makes support for multiple identity stores possible. 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 provider entity. 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 details the steps to set up the client connection for either Power BI Desktop or the Power BI service. The steps include application registration, configuring permissions, and configuring credentials.

To learn how to integrate Amazon Redshift native IdP federation with Azure AD, using Power BI Desktop and JDBC Client-SQL Workbench/J, watch the following video:

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. 458), 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 that use the redshift: prefix.
- Actions, resources, and condition keys for Amazon Redshift Serverless that use the redshift-serverless: prefix.
- Actions, resources, and condition keys for Amazon Redshift Data API that use the redshift-data: prefix.
- Actions, resources, and condition keys for AWS SQL Workbench (Amazon Redshift query editor v2) that use the sqlworkbench: prefix.
The query editor v2 includes permission-only actions that don't directly correspond to an API operation. These actions are indicated in the Service Authorization Reference with [permission only].

The Service Authorization 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. 461).

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
Permissions for network resources

The following permissions allow action on Amazon EC2 for creation and management of security group rules. These security groups and rules are specifically associated with the Amazon Redshift resource tag. This limits the scope of the permissions to specific Amazon Redshift resources.

- ec2:CreateSecurityGroup
- ec2:AuthorizeSecurityGroupEgress
- ec2:AuthorizeSecurityGroupIngress
- ec2:RevokeSecurityGroupEgress
- ec2:RevokeSecurityGroupIngress
- ec2:ModifySecurityGroupRules
- ec2:DeleteSecurityGroup

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/"
],

"Sid": "EnableCreationAndManagementOfRedshiftCloudwatchLogStreams",
"Effect": "Allow",
"Action": [
    "logs:CreateLogStream",
    "logs:PutLogEvents",
    "logs:DescribeLogStreams",
    "logs:GetLogEvents"
],
"Resource": [
    "arn:aws:logs:*:*:log-group:/aws/redshift/*:log-stream:"
]
}

For more information about service-linked roles and their purpose in AWS, see Using service-linked roles.
For more information about specific actions and other IAM resources for Amazon Redshift, see Actions, resources, and condition keys for Amazon 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. 94).

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. 500)
- Creating temporary IAM credentials (p. 501)
- Options for providing IAM credentials (p. 511)

### 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. As a best practice, we
recommend attaching permissions policies to an IAM role and then assigning it to users and groups as
needed. For more information, see Identity and access management in Amazon Redshift.

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
parallel 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
a 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 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 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 access (p. 502)**
   (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. 502)**
   (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 with permissions to call GetClusterCredentials (p. 503)**
   Your SQL client application assumes the user 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. 505)**

(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. 506)**

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 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. 506)](#).

**Note**

If you use an IAM policy variable `${redshift:DbUser}`, as described in [Resource policies for GetClusterCredentials (p. 487)](#) 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. 493)](#).

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).

```xml
<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.

```xml
<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.

```xml
<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 with permissions to call GetClusterCredentials

Your SQL client needs authorization to call the GetClusterCredentials operation on your behalf. To provide that authorization, you create a 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 a 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.

```text
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 \textit{region}, \textit{account-id}, and \textit{cluster-name} with the values for your AWS Region, account, and cluster. For \textit{database-name}, specify the name of a database that the user will log into.

\begin{verbatim}
\end{verbatim}

• If you use the DbGroups option, include permission to call the \textit{redshift:JoinGroup} operation with the Amazon Redshift dbgroup resource in the following format. Replace \textit{region}, \textit{account-id}, and \textit{cluster-name} with the values for your AWS Region, account, and cluster. For \textit{dbgroup-name}, specify the name of a user group that the user joins at login.

\begin{verbatim}
arn:aws:redshift:region:account-id:dbgroup:cluster-name/dbgroup-name
\end{verbatim}

For more information and examples, see Resource policies for \textit{GetClusterCredentials} (p. 487).

The following example shows a policy that allows the IAM role to call the \textit{GetClusterCredentials} operation. Specifying the Amazon Redshift \textit{dbuser} resource grants the role access to the database user name \textit{temp_creds_user} on the cluster named examplecluster.

\begin{verbatim}
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "redshift:GetClusterCredentials",
    }
}
\end{verbatim}

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 \textit{temp} with any cluster in the specified account.

\textbf{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.

\begin{verbatim}
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "redshift:GetClusterCredentials",
    }
}
\end{verbatim}

The following example shows a policy that allows the IAM role to call the \textit{GetClusterCredentials} operation with the option to automatically create a new user and specify groups the user joins at login. The \textit{Resource}: "**" clause grants the role access to any resource, including clusters, database users, or user groups.

\begin{verbatim}
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "redshift:GetClusterCredentials",
    }
}
\end{verbatim}
Using IAM authentication to generate database user credentials

```
"Statement": {
   "Effect": "Allow",
   "Action": [
      "redshift:GetClusterCredentials",
      "redshift:CreateClusterUser",
      "redshift:JoinGroup"
   ],
   "Resource": "*
}
```

For more information, see [Amazon Redshift ARN syntax](https://docs.aws.amazon.com/redshift/latest/gsg/arn-syntax.html).

**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 credentials. If the password is not disabled, the user can log on either with the password or using temporary credentials. You can't disable the password for a superuser.

Users need programmatic access if they want to interact with AWS outside of the AWS Management Console. The way to grant programmatic access depends on the type of user that's accessing AWS.

To grant users programmatic access, choose one of the following options.

<table>
<thead>
<tr>
<th>Which user needs programmatic access?</th>
<th>To</th>
<th>By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workforce identity (Users managed in IAM Identity Center)</td>
<td>Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td>Following the instructions for the interface that you want to use.</td>
</tr>
</tbody>
</table>

- For the AWS CLI, see [Configuring the AWS CLI to use AWS IAM Identity Center (successor to AWS Single Sign-On) in the AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/reference/index.html).
- For AWS SDKs, tools, and AWS APIs, see [IAM Identity Center authentication in the AWS SDKs and Tools Reference Guide](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_programmatic_signin.html).
Which user needs programmatic access? | To | By
--- | --- | ---
IAM | Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs. | Following the instructions in Using temporary credentials with AWS resources in the IAM User Guide.

IAM | (Not recommended) Use long-term credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs. | Following the instructions for the interface that you want to use.
- For the AWS CLI, see Authenticating using IAM user credentials in the AWS Command Line Interface User Guide.
- For AWS SDKs and tools, see Authenticate using long-term credentials in the AWS SDKs and Tools Reference Guide.
- For AWS APIs, see Managing access keys for IAM users in the IAM User Guide.

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. 516).
To configure a JDBC connection to use IAM credentials

1. Download the latest Amazon Redshift JDBC driver from the Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 201) 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 user or role must have permission to call the redshift:DescribeClusters operation with the specified cluster. If your 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.

```
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. 534).

The following URL specifies AccessKeyId and SecretAccessKey for a 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. 291).

1. Install and configure the latest Amazon Redshift OBDC driver for your operating system. For more information, see Configuring an ODBC connection (p. 291) 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. 293)
- Use an ODBC driver manager to configure the driver on Linux and macOS X operating systems (p. 297)

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 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 user or role doesn't have permission to call the redshift:DescribeClusters operation, specify **Server** and **Port**.

   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. 512).

      (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. 534).

      **IAM Credentials**

      Enter the following information:

      - **ClusterID**
      - **Region**
      - **AccessKeyId** and **SecretAccessKey**

      The access key ID and secret access key for the IAM role or 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)

• **DbGroups** (optional)

  For more information, see [JDBC and ODBC Options for Creating Database User Credentials](#).

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](#).

(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](#).

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. 502).

(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. 534).

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. 502).
- 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. 534).

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.

- **Azure Client ID**
  
  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. 534).

### 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. 512).

- **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. 512).

- **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. 513).

  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 [Options for JDBC driver version 2.1 configuration](p. 214) and [Configure ODBC driver options](p. 301).

  For more information, see [Configure a JDBC or ODBC connection to use IAM credentials](p. 506).
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. 513). If plugin_name isn’t used, the other options are ignored.

The following example shows the ~/.aws/credentials file with two profiles.

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUttnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

[user2]
aws_access_key_id=AKIAI44QHBDHEXAMPLE
aws_secret_access_key=j7MtcGbClwBF/2ZP9Utk/h3yoCo8nvbEXAMPLEKEY
session_token=AQoDYXdzEPT/////////wEXAMPLEtc764bNzC95ABPBM2wDKk44HIZ8j4FZtwdQWLwsKWHGbuFqwaeM1cRXmxfSPfieoYRqTFfKD8YUwthAx7mSEI/gkPkpPi/kMc6dQrMDeehM4IC1ntBmUpp2wUE8phUZampKsbuxEDYyOKPkyQDYwT7wZ0wq5sVXdv75YU9HFv1rdt8t6xq6f6E8YQcHNVXAk1Y9q6d+x0tzKwsT38xVqr7ZDz0u0iPPkUL641IZbqBAz+scqKmzBFdypNC9Yjc8fPOLn9FX95YvKT4rvx3ISITJabIQwj2lCC/OLxBA==
```

To use the credentials for the user2 example, specify Profile=user2 in the JDBC URL.

For more information on using profiles, see Configuration and credential file settings 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 user configured for IAM database authentication. SessionToken is required only for an IAM role with temporary credentials. SessionToken isn't used for a 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. 513).</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>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. 512).</td>
</tr>
</tbody>
</table>

Using a credentials provider plugin

Amazon Redshift uses credentials provider plugins for single sign-on authentication.

To support single sign-on 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. 516).

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 single sign-on 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 \texttt{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 \texttt{client_id}.
5. After the user authenticates and grants consent, the Azure AD \texttt{oauth2/authorize} endpoint returns and sends a response over HTTPS with the authorization code to the indicated \texttt{redirect_uri}.
6. The web browser moves the response message with the SAML response to the indicated \texttt{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 \texttt{oauth2/token} endpoint.
8. The Azure AD \texttt{oauth2/token} endpoint returns a response with an access token to the indicated \texttt{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. 523).

- 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. 527).

- 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. 530).

- Microsoft Azure Active Directory

For more information, see Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD (p. 516).

### 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>plugin_name</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 single sign-on with MFA</td>
</tr>
<tr>
<td></td>
<td>com.amazon.redshift.plugin.BrowserAzureCredentialsProvider</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>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>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. 502).</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/0oa2hylwrpM8UGehd1t7/272">https://example.okta.com/home/amazon_aws/0oa2hylwrpM8UGehd1t7/272</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 1.4.10.000 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.


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 single sign-on 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. 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.

   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.

   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>
</tbody>
</table>
Amazon Redshift Management Guide
Using IAM authentication to generate database user credentials

<table>
<thead>
<tr>
<th>Claim name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<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>
<tr>
<td><a href="https://redshift.amazon.com/SAML/Attributes/DbUser">https://redshift.amazon.com/SAML/Attributes/DbUser</a></td>
<td>user.userprincipalname</td>
</tr>
</tbody>
</table>

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 single sign-on 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)](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_saml-auth-saml-identityprovider.html) 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](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_saml-auth-saml-role.html) 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)](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_saml-auth-saml-policy.html) 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`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "redshift:GetClusterCredentials",
      "Resource": [
      ],
      "Condition": {
        "StringEquals": {
          "aws:userid": "AROAJ2UCCR6DPCEXAMPLE:${redshift:DbUser}@example.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "redshift:GetDbUserCredentials",
```
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 connection for JDBC driver version 2.1 for Amazon Redshift (p. 201).

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 single sign-on 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 single sign-on with MFA configuration, add additional information to the connection properties as follows:
Using IAM authentication to generate database user credentials

- For **plugin_name**, enter `com.amazon.redshift.plugin.BrowserAzureCredentialsProvider`. This value specifies to the driver to use Azure AD single sign-on 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 single sign-on with MFA configuration.

- 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. 293).
- For macOS, see Install the Amazon Redshift ODBC driver on macOS X (p. 297).
- For Linux, see Install the Amazon Redshift ODBC driver on Linux (p. 295).

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 single sign-on 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 single sign-on 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.

• 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 single sign-on configuration, enter *AzureAD*. This specifies to the driver to use Azure Single Sign-On as the authentication method.

• For **plugin_name** for Azure AD single sign-on 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.

```bash
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.

   Navigate to the [Set up Single Sign-On with SAML](https://portal.azure.com/) 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](https://docs.aws.amazon.com/iam/latest/userguide/saml_aws_rpt.html) 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](https://docs.aws.amazon.com/redshift/latest/userguide/cluster-creation.html).

2. Set up AD FS to control Amazon Redshift access on the Microsoft Management Console:

   1. Choose **ADFS 2.0**, and then choose **Add Relying Party Trust**. On the **Add Relying Party Trust Wizard** page, choose **Start**.

   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**.

9. For your relying party, open the context (right-click) menu and choose **Edit Claim Rules**. On the **Edit Claim Rules** page, choose **Add Rule**.

10. For **Claim rule template**, choose **Transform an Incoming Claim**, and then on the **Edit Rule – Nameld** page, do the following:

    - For **Claim rule name**, enter **Nameld**.
    - 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.

    ```
c:[Type == 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.

    ```
```
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. 516).

To set up JDBC for authentication to AD FS

- Configure your database client to connect to your cluster through JDBC using AD FS single sign-on.

You can use any client that uses a JDBC driver to connect using AD FS single sign-on or use a language like Java to connect using a script. For installation and configuration information, see Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 201).

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).

   e. Choose Extended Properties. For plugin_name, enter `com.amazon.redshift.plugin.AdfsCredentialsProvider`. This value specifies to the driver to use AD FS single sign-on 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 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. 293).
- For macOS, see Install the Amazon Redshift ODBC driver on macOS X (p. 297).
- For Linux, see Install the Amazon Redshift ODBC driver on Linux (p. 295).

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 single sign-on.
- 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 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: 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 single sign-on with MFA configuration, enter BrowserSAML. This is the authentication method that the ODBC driver uses to authenticate to AD FS.
  - For AD FS single sign-on configuration, enter ADFS. This is the authentication method that the ODBC driver uses to authenticate using Azure AD single sign-on.
  - For uid, enter your-adfs-username. This is the user name of the Microsoft Azure account that 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 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.

```bash
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.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/DbUser">https://redshift.amazon.com/SAML/Attributes/DbUser</a></td>
<td>email</td>
</tr>
<tr>
<td><a href="https://redshift.amazon.com/SAML/Attributes/DbGroups">https://redshift.amazon.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</td>
</tr>
</tbody>
</table>
Using IAM authentication to generate database user credentials

<table>
<thead>
<tr>
<th>Application attribute</th>
<th>Identity bridge attribute of literal value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ExtractByRegularExpression. In Expression, enter (.<em>[@][@][@:.]</em>).</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/Role)
   - [https://aws.amazon.com/SAML/Attributes/RoleSessionName](https://aws.amazon.com/SAML/Attributes/RoleSessionName)
   - [https://redshift.amazon.com/SAML/Attributes/AutoCreate](https://redshift.amazon.com/SAML/Attributes/AutoCreate)
   - [https://redshift.amazon.com/SAML/Attributes/DbUser](https://redshift.amazon.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)](https://docs.aws.amazon.com/IAM/latest/userguide/creating-iam-identity-provider.html) 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](https://docs.aws.amazon.com/IAM/latest/userguide/creating-iam-role-for-saml.html) 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)](https://docs.aws.amazon.com/IAM/latest/userguide/creating-iam-policy.html) in the IAM User Guide. For an Azure AD example, see [Setting up JDBC or ODBC single sign-on authentication with Microsoft Azure AD](https://docs.aws.amazon.com/redshift/latest/userguide/iam-authentication-jdbc-odbc.html).

### To set up JDBC for authentication to Ping Identity

- Configure your database client to connect to your cluster through JDBC using Ping Identity single sign-on.

You can use any client that uses a JDBC driver to connect using Ping Identity single sign-on or use a language like Java to connect using a script. For installation and configuration information, see Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 201).

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).

528
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 single sign-on 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 single sign-on as the authentication method.
- For Ping Identity with single sign-on, for **plugin_name**, enter `com.amazon.redshift.plugin.BrowserSamlCredentialsProvider`. This value specifies to the driver to use Ping Identity PingOne with single sign-on 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 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. 293).
- For macOS, see [Install the Amazon Redshift ODBC driver on macOS X](p. 297).
- For Linux, see [Install the Amazon Redshift ODBC driver on Linux](p. 295).

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 single sign-on.
  - For Ping Identity with single sign-on configuration, choose **Identity Provider: Browser SAML**. This is the authentication method that the ODBC driver uses to authenticate using Ping Identity with single sign-on.
- 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 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: 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...
Using IAM authentication to generate database user credentials.

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.
- 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 single sign-on configuration, enter `Ping`. This is the authentication method that the ODBC driver uses to authenticate using Ping Identity with single sign-on.
  - For `uid`, enter `your-ping-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 `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 single sign-on 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.

```bash
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><a href="https://aws.amazon.com/SAML/Attributes/RoleSessionName">https://aws.amazon.com/SAML/Attributes/RoleSessionName</a></td>
<td>user.email</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/DbUser">https://redshift.amazon.com/SAML/Attributes/DbUser</a></td>
<td>user.email</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. 516).
To set up JDBC for authentication to Okta

- Configure your database client to connect to your cluster through JDBC using Okta single sign-on.

You can use any client that uses a JDBC driver to connect using Okta single sign-on or use a language like Java to connect using a script. For installation and configuration information, see Configuring a connection for JDBC driver version 2.1 for Amazon Redshift (p. 201).

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:

   - 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 single sign-on as the authentication to log in to Okta.
   - For Okta single sign-on, for plugin_name, enter `com.amazon.redshift.plugin.OktaCredentialsProvider`. This value specifies to the driver to use Okta single sign-on as the authentication method.
   - For Okta single sign-on with MFA, for plugin_name, enter `com.amazon.redshift.plugin.BrowserSamlCredentialsProvider`. This value specifies to the driver to use Okta single sign-on 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 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. 293).
- For macOS, see Install the Amazon Redshift ODBC driver on macOS X (p. 297).
• For Linux, see Install the Amazon Redshift ODBC driver on Linux (p. 295).

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 single sign-on configuration, choose Identity Provider: Okta. This is the authentication method that the ODBC driver uses to authenticate using Okta single sign-on.
  • For Okta single sign-on with MFA configuration, choose Identity Provider: Browser SAML. This is the authentication method that the ODBC driver uses to authenticate using Okta 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-okta-username. This is the user name for the Okta 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: 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 single sign-on with MFA configuration, enter BrowserSAML. This is the authentication method that the ODBC driver uses to authenticate to Okta single sign-on with MFA.
  • For Okta single sign-on configuration, enter Okta. This is the authentication method that the ODBC driver uses to authenticate using Okta single sign-on.

• For uid, enter your-okta-username. This is the user name of the Okta 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 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 single sign-on 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. 502).

**Note**

If you use an IAM policy variable `${redshift:DbUser}`, as described in [Resource policies for GetClusterCredentials](p. 487) 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. 493).

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](p. 520).</td>
</tr>
<tr>
<td>AutoCreate</td>
<td>Specify <code>true</code> 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. 534).
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 a user or role with the required permissions. For more information about IAM permissions, see Create an IAM role with permissions to call GetClusterCredentials (p. 503).
2. As a 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 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 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.

```
{
"DbUser": "IAM:temp_creds_user",
"Expiration": "2016-12-08T21:12:53Z",
"DbPassword": "EXAMPLEjArE3hcnQj8zt4XQj9Xtma8oxYEM8OyxpDHwXVPyJYBDm/gqX2Eeaq6P3DgTzgPg=="
}
```

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.

```
aws redshift get-cluster-credentials --cluster-identifier examplecluster --db-name exampledb --duration-seconds 3600
```

The result is as follows.

```
{
"DbUser": "IAM:temp_creds_user",
"Expiration": "2016-12-08T21:12:53Z",
"DbPassword": "EXAMPLEjArE3hcnQj8zt4XQj9Xtma8oxYEM8OyxpDHwXVPyJYBDm/gqX2Eeaq6P3DgTzgPg=="
}
```
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": "EXAMPLEjArE3hcnQj8zt4XQj9 XtmaBoxYEM8OyxpDHwXVPyJYBM/gqX2Eeaq6P3DgTzgPg=="
}
```

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. 303).

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 temporary credentials.

Users need programmatic access if they want to interact with AWS outside of the AWS Management Console. The way to grant programmatic access depends on the type of user that's accessing AWS.

To grant users programmatic access, choose one of the following options.

<table>
<thead>
<tr>
<th>Which user needs programmatic access?</th>
<th>To</th>
<th>By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workforce identity</td>
<td>Use temporary credentials to sign programmatic requests to</td>
<td>Following the instructions for the interface that you want to use.</td>
</tr>
</tbody>
</table>

536
### Which user needs programmatic access?

<table>
<thead>
<tr>
<th>Users managed in IAM Identity Center</th>
<th>To the AWS CLI, AWS SDKs, or AWS APIs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>• For the AWS CLI, see <a href="https://docs.aws.amazon.com/cli/latest/reference/index.html">Configuring the AWS CLI to use AWS IAM Identity Center (successor to AWS Single Sign-On)</a> in the AWS Command Line Interface User Guide.</td>
<td></td>
</tr>
<tr>
<td>• For AWS SDKs, tools, and AWS APIs, see [IAM Identity Center authentication](<a href="https://docs.aws.amazon.com/">https://docs.aws.amazon.com/</a> vignettes/IAM_center.html) in the AWS SDKs and Tools Reference Guide.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IAM</th>
<th>Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Following the instructions in <a href="https://docs.aws.amazon.com/IAM/latest/userguide/using-sdk.html">Using temporary credentials with AWS resources</a> in the IAM User Guide.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IAM</th>
<th>(Not recommended) Use long-term credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Following the instructions for the interface that you want to use.</td>
<td></td>
</tr>
<tr>
<td>• For the AWS CLI, see <a href="https://docs.aws.amazon.com/cli/latest/reference/index.html">Authenticating using IAM user credentials</a> in the AWS Command Line Interface User Guide.</td>
<td></td>
</tr>
<tr>
<td>• For AWS SDKs and tools, see [Authenticate using long-term credentials](<a href="https://docs.aws.amazon.com/">https://docs.aws.amazon.com/</a> vignettes/IAM_center.html) in the AWS SDKs and Tools Reference Guide.</td>
<td></td>
</tr>
<tr>
<td>• For AWS APIs, see [Managing access keys for IAM users](<a href="https://docs.aws.amazon.com/">https://docs.aws.amazon.com/</a> vignettes/IAM_center.html) in the IAM User Guide.</td>
<td></td>
</tr>
</tbody>
</table>

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. 542).

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 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.
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. 538).
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. 544).

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:
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
   ```

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
   ```
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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": ["redshift.us-east-1.amazonaws.com", "redshift.us-west-2.amazonaws.com"]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

6. Choose **Update Trust Policy**

**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 [Quotas for Amazon Redshift objects](p. 643).

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.

```
{
  "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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {
        "AWS": "arn:aws:iam::role/RoleA"
      }
    }
  ]
}
```

The following trust policy establishes a trust relationship with the owner of RoleA, AWS account 123456789012.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      }
    }
  ]
}
```

**Note**

To restrict role chaining authorization to specific users, define a condition. For more information, see [Restricting access to IAM roles](p. 538).

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.
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.

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.

copy venue
from 's3://companyb/redshift/venue_pipe_'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';

In the following example, CREATE EXTERNAL SCHEMA uses chained roles to assume the role RoleB.

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.

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. 542).

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 temporary credentials. For information about creating an IAM role, see Authorizing Amazon Redshift to access other AWS services on your behalf (p. 536).

Users need programmatic access if they want to interact with AWS outside of the AWS Management Console. The way to grant programmatic access depends on the type of user that's accessing AWS.

To grant users programmatic access, choose one of the following options.

<table>
<thead>
<tr>
<th>Which user needs programmatic access?</th>
<th>To</th>
<th>By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workforce identity (Users managed in IAM Identity Center)</td>
<td>Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td>Following the instructions for the interface that you want to use.</td>
</tr>
<tr>
<td>- For the AWS CLI, see Configuring the AWS CLI to use AWS IAM Identity Center (successor to AWS Single Sign-On) in the AWS Command Line Interface User Guide.</td>
<td>- For the AWS SDKs, tools, and AWS APIs, see IAM Identity Center authentication in the AWS SDKs and Tools Reference Guide.</td>
<td></td>
</tr>
<tr>
<td>IAM</td>
<td>Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td>Following the instructions in Using temporary credentials with AWS resources in the IAM User Guide.</td>
</tr>
<tr>
<td>IAM</td>
<td>(Not recommended) Use long-term credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td>Following the instructions for the interface that you want to use.</td>
</tr>
<tr>
<td>- For the AWS CLI, see Authenticating using IAM user credentials in the AWS Command Line Interface User Guide.</td>
<td>- For AWS SDKs and tools, see Authenticate using long-term credentials in the AWS SDKs and Tools Reference Guide.</td>
<td></td>
</tr>
<tr>
<td>- For AWS APIs, see Managing access keys for IAM users in the IAM User Guide.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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, a 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. As a best practice, we recommend attaching permissions policies to an IAM role and then assigning it to users and groups as needed. For more information, see [*Identity and access management in Amazon Redshift*](#).

The following example shows an IAM policy that can be attached to a 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 a 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.

```
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.

```
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. 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.

```
{
  "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"
        }
      ],
    },
    ...
  ],
}  
```
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. 550)
- Removing IAM roles from your cluster (p. 550)
- Associating IAM roles with your cluster (p. 551)
- Setting an IAM role as the default (p. 551)
- Making an IAM role no longer default for your cluster (p. 551)

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.

```json
{
    "Effect": "Allow",
    "Action": ["s3:GetObject",
                "s3:GetBucketAcl",
                "s3:GetBucketCors",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:ListBucketMultipartUploads",
                "s3:GetObject",
                "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.

```json
{
    "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: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*"
    ]
}
```

The following example shows the permissions in the AmazonRedshiftAllCommandsFullAccess managed policy that allow certain actions for the IAM role 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 command using federated queries.

```
{
    "Effect": "Allow",
    "Action": [
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret",
        "secretsmanager:ListSecretVersionIds"
    ],
    "Resource": [
        "arn:aws:secretsmanager:*:*:secret:*Redshift*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "secretsmanager:GetRandomPassword",
        "secretsmanager:ListSecrets"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "secretsmanager:ResourceTag/Redshift": "true"
        }
    }
}
```
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. For more information about this step, see Creating a cluster (p. 112).
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 it 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.

```bash
aws redshift modify-cluster-iam-roles \
```

552
Authorizing Amazon Redshift to access AWS services

```bash
--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.

```
{
    "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.

```bash
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.

```
{
    "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 \\
--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 \\
--add-iam-roles 'arn:aws:iam::012345678910:role/myrole3' \\
--default-iam-role-arn 'arn:aws:iam::012345678910:role/myrole3'
```

The following snippet is an example of the response.

```json
{
  "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 authorize Amazon Redshift to access AWS services, you can set up IAM roles for the cluster. The following JSON snippet shows a cluster configuration:

```json
{
    "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:

```json
{
    "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",
                "ApplyStatus": "adding"
            }
        ]
    }
}
```
Using a federated identity to manage Amazon Redshift access to local resources and Amazon Redshift Spectrum external tables

Using identity federation in AWS with credentials provided from GetDatabaseCredentials can simplify authorization and access to local data and to external data. Currently, to give users access to external data that resides in Amazon S3, you create an IAM role with permissions defined in a permissions policy. Then, users with the role attached can access the external data. This works, but if you want to provide granular rules, such as making specific columns unavailable for a particular user, you may have to do additional configuration on the external schema. In this topic, we show how to provide access to resources with AWS identity federation, instead of using a specific IAM role. Identity federation, with credentials provided from GetDatabaseCredentials, can provide access to AWS Glue and Redshift Spectrum resources via granular IAM rules that are easier to specify and change. This makes it easier to apply access that conforms to your business rules.

The benefits of using federated credentials are the following:

- You don't have to manage cluster-attached IAM roles for Redshift Spectrum.
- Cluster administrators can create an external schema that's accessible by consumers with different IAM contexts. This is useful, for example, to perform column filtering on a table, where different consumers query the same external schema and get varying fields in returned records.
- You can query Amazon Redshift using a user with IAM permissions, rather than only with a role.

Preparing an identity to log in with federated identity

Before logging in with federated identity, you must perform several preliminary steps. These instructions assume you have an existing Redshift Spectrum external schema that references a data file stored in an Amazon S3 bucket, and the bucket is in the same account as your Amazon Redshift cluster or Amazon Redshift Serverless data warehouse.

1. Create an IAM identity. This can be a user or an IAM role. Use any name supported by IAM.
2. Attach permissions policies to the identity. Specify either of the following:
   - redshift:GetClusterCredentialsWithIAM (for an Amazon Redshift provisioned cluster)
   - redshift-serverless:GetCredentials (for Amazon Redshift Serverless)

   You can add permissions with the policy editor, using the IAM console.

   The IAM identity also needs permissions to access external data. Grant access to Amazon S3 by adding the following AWS managed policies directly:
   - AmazonS3ReadOnlyAccess
   - AWSGlueConsoleFullAccess

   The last managed policy is required if you're using AWS Glue to prepare your external data. For more information about the steps for granting access to Amazon Redshift Spectrum, see Create an IAM role for Amazon Redshift, which is part of the getting-started guide for Amazon Redshift and Redshift Spectrum. It shows the steps for adding IAM policies to access Redshift Spectrum.
3. Set up your SQL client to connect to Amazon Redshift. Use the Amazon Redshift JDBC driver, and add your user’s credentials to the tool’s credential properties. A client like SQL Workbench/J works well for this. Set the following client-connection extended properties:

   556
• **AccessKeyId** – Your access key identifier.
• **SecretAccessKey** – Your secret access key. (Note the security risk of transmitting the secret key if you don't use encryption.)
• **SessionToken** – A set of temporary credentials for an IAM role.
• **groupFederation** – Set to `true` if you're configuring federated identity for a provisioned cluster. Don't set this parameter if you are using Amazon Redshift Serverless.
• **LogLevel** – Integer log-level value. This is optional.

4. Set the URL to the JDBC endpoint found in the Amazon Redshift or Amazon Redshift Serverless console. Replace your URL schema with `jdbc:redshift:iam:` and use this formatting:

   • Format for an Amazon Redshift provisioned cluster: `jdbc:redshift:iam://<cluster_id>.<unique_suffix>.<region>.redshift.amazonaws.com:<port>/<database_name>

   Example: `jdbc:redshift:iam://test1.12345abcdefg.us-east-1.redshift.amazonaws.com:5439/dev`


   Example: `jdbc:redshift:iam://default.123456789012.us-east-1.redshift-serverless.amazonaws.com:5439/dev`

After you connect to the database for the first time, using an IAM identity, Amazon Redshift automatically creates an Amazon Redshift identity with the same name, prefixed with IAM: for a user or IAMR: for an IAM role. The remaining steps in this topic show examples for a user.

If a Redshift user isn't automatically created, you can create one by running a `CREATE USER` statement, using an admin account, specifying the user name in the format `IAM:<user name>`.

5. As your Amazon Redshift cluster administrator, grant the Redshift user the required permissions to access the external schema.

   ```sql
   GRANT ALL ON SCHEMA my_schema to "IAM:my_user";
   ```

   To grant the ability to your Redshift user to create tables in the external schema, they must be a schema owner. For example:

   ```sql
   ALTER SCHEMA my_schema owner to "IAM:my_user";
   ```

6. To verify the configuration, run a query as the user, using the SQL client, after permissions are granted. This query sample retrieves data from an external table.

   ```sql
   SELECT * FROM my_schema.my_table;
   ```

### Getting started with identity and authorization propagation to Redshift Spectrum

To pass a federated identity to query external tables, you set `SESSION` as the value for the `IAM_ROLE` query parameter of `CREATE EXTERNAL SCHEMA`. The following steps show how to set up and leverage `SESSION` to authorize queries on the external schema.

1. Create local tables and external tables. External tables catalogued with AWS Glue work for this.
2. Connect to Amazon Redshift with your IAM identity. As mentioned in the previous section, when the identity connects to Amazon Redshift, a Redshift database user is created. The user is created if they didn't previously exist. If the user is new, the administrator must grant them permissions to perform tasks in Amazon Redshift, like querying and creating tables.

3. Connect to Redshift with your admin account. Run the command to create an external schema, using the SESSION value.

```
cREATE EXTERNAL SCHEMA spectrum_schema FROM DATA CATALOG
  DATABASE 'my_external_database'
  REGION 'us-east-1'
  IAM_ROLE 'SESSION'
  CATALOG_ID '123456789012';
```

Note that catalog_id is set in this case. This is a new setting added with the feature, because SESSION replaces a specific role.

In this example, values in the query mimic how real values appear.

```
cREATE EXTERNAL SCHEMA spectrum_schema FROM DATA CATALOG
  DATABASE 'spectrum_db'
  REGION 'us-east-1'
  IAM_ROLE 'SESSION'
  CATALOG_ID '123456789012';
```

The catalog_id value in this case is your AWS account ID.

4. Run queries to access your external data, using the IAM identity you connected with in step 2. For example:

```
SELECT * FROM spectrum_schema.table1;
```

In this case, table1 can be, for example, JSON-formatted data in a file, in an Amazon S3 bucket.

5. If you already have an external schema that uses a cluster-attached IAM role, pointing to your external database or schema, you can either replace the existing schema and use a federated identity as detailed in these steps, or create a new one.

SESSION indicates that federated identity credentials are used to query the external schema. When you use the SESSION query parameter, make sure you set the catalog_id. It's required because it points to the data catalog used for the schema. Previously, catalog_id was retrieved from the value assigned to iam_role. When you set up identity and authorization propagation this way, for instance, to Redshift Spectrum, by using federated credentials to query an external schema, authorization by means of an IAM role isn't required.

**Usage notes**

A common connection error is the following: `IAM error retrieving temp credentials: Unable to unmarshall exception response with the unmarshallers provided`. This error is a result of having a legacy JDBC driver. The minimum driver version required for federated identity is 2.1.0.9. You can get the JDBC driver from [Download the Amazon Redshift JDBC driver, version 2.1](#).

**Additional resources**

These links provide additional information for managing access to external data.

- You can still access Redshift Spectrum data using an IAM role. For more information, see [Authorizing Amazon Redshift to access other AWS services on your behalf](p. 536).
When you manage access to external tables with AWS Lake Formation, you can query them using Redshift Spectrum with federated IAM identities. You no longer have to manage cluster-attached IAM roles for Redshift Spectrum to query data registered with AWS Lake Formation. For more information, see Using AWS Lake Formation with Amazon Redshift Spectrum.

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. 623). For a list of metrics, see Monitoring Amazon Redshift using CloudWatch metrics (p. 595).

**AWS CloudTrail Logs**

CloudTrail provides a record of API operations taken by a user, an IAM 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 with CloudTrail (p. 567).

**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 can be stored in:

- **Amazon S3 buckets** - This provides access with data-security features for users who are responsible for monitoring activities in the database.
- **Amazon CloudWatch** - You can view audit-logging data using the features built into CloudWatch, such as visualization features and setting actions.

**Note**

SYS_CONNECTION_LOG collects connection log data for Amazon Redshift Serverless. Note that when you collect audit logging data for Amazon Redshift Serverless, it can't be sent to log files, only to CloudWatch.

**Topics**

- Amazon Redshift logs (p. 560)
- Enabling logging (p. 562)
- Sending audit logs to Amazon CloudWatch (p. 563)
- Managing log files in Amazon S3 (p. 563)
- Troubleshooting Amazon Redshift audit logging in Amazon S3 (p. 566)
- Logging Amazon Redshift API calls with AWS CloudTrail (p. 566)
Amazon Redshift Management Guide
Database audit logging

- Configuring auditing using the console (p. 566)
- Configuring logging by using the AWS CLI and Amazon Redshift API (p. 567)

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 a simpler 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 system log tables which are **STL_USERLOG** and **STL_CONNECTION_LOG**. Records that are older than, but not including, the latest record are copied to log files.

**Note**
For Amazon Redshift Serverless, **SYS_CONNECTION_LOG** collects connection-log data. When you collect audit logging data for Amazon Redshift Serverless, it can't be sent to log files, only to CloudWatch.

**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. For more information about the collected connection log data for Amazon Redshift Serverless, see **SYS_CONNECTION_LOG**.

<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>
</tbody>
</table>
Database audit logging

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>iamauthguid</td>
<td>The AWS Identity and Access Management (IAM) authentication ID for the AWS CloudTrail request. This is the identifier for the GetClusterCredentials API call to create the credentials being used for a given connection.</td>
</tr>
<tr>
<td>application_name</td>
<td>The initial or updated name of the application for a session.</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>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>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>
<tr>
<td>compression</td>
<td>The compression algorithm in use for the connection.</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>
</tbody>
</table>
## Column name | Description
--- | ---
usecreatedb | If true (1), indicates that the user has create database permissions.
usesuper | If true (1), indicates that the user is a superuser.
usecatupd | If true (1), indicates that the user can update system catalogs.
valuntil | Password expiration date.
pid | Process ID.
xid | Transaction ID.
recordtime | Time in UTC that the query started.

Query the **SYS_USERLOG** system view to find additional information about changes to users. This view includes log data from Amazon Redshift Serverless.

### User activity log
Logs each query before it is run on the database.

| Column name | Description |
--- | --- |
recordtime | Time the event occurred. |
db | Database name. |
user | User name. |
pid | Process ID associated with the statement. |
userid | User ID. |
xid | Transaction ID. |
query | A prefix of LOG: followed by the text of the query, including newlines. |

### Enabling logging
Audit logging is not turned on by default in Amazon Redshift. When you turn on logging on your cluster, Amazon Redshift exports logs to Amazon CloudWatch, or 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 previous logs.

Audit logging to CloudWatch or to Amazon S3 is an optional process. Logging to system tables is not optional and happens automatically. 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. 389).

**Sending audit logs to Amazon CloudWatch**

When you enable logging to CloudWatch, Amazon Redshift exports cluster connection, user, and user-activity log data to an Amazon CloudWatch Logs log group. The log data doesn't change, in terms of schema. CloudWatch is built for monitoring applications, and you can use it to perform real-time analysis or set it to take actions. You can also use Amazon CloudWatch Logs to store your log records in durable storage.

Using CloudWatch to view logs is a recommended alternative to storing log files in Amazon S3. It doesn't require much configuration, and it may suit your monitoring requirements, especially if you use it already to monitor other services and applications.

**Log groups and log events in Amazon CloudWatch**

After selecting which Amazon Redshift logs to export, you can monitor log 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/cluster/<cluster_name>/<log_type>
```

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

```
/aws/redshift/cluster/cluster1/connectionlog
```

Log events are exported to a log group using the log stream. 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 information about searching and filtering log data, see Creating metrics from log events using filters.

In CloudWatch, you can search your log data with a query syntax that provides for granularity and flexibility. For more information, see CloudWatch Logs Insights query syntax.

**Migrating to Amazon CloudWatch audit logging**

In any case where you are sending logs to Amazon S3 and you change the configuration, for example to send logs to CloudWatch, logs that remain in Amazon S3 are unaffected. You can still query the log data in the Amazon S3 buckets where it resides.

**Managing log files in Amazon S3**

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.

When Amazon Redshift uses Amazon S3 to store logs, you incur charges for the storage that you use in Amazon S3. Before you configure logging to Amazon S3, 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.

Limitations when you enable logging to Amazon S3

Audit logging has the following constraints:

- You can use only Amazon S3-managed keys (SSE-S3) encryption (AES-256).
- The Amazon S3 buckets must have the S3 Object Lock feature turned off.

Bucket permissions for Amazon Redshift audit logging

When you turn on logging to Amazon S3, 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 user or IAM role that turns on 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
{
```

564
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 to the Region where the cluster is located.

**Best practices for log files**

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: `AWLogs/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:
Troubleshooting Amazon Redshift audit logging in Amazon S3

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. 564).

• 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 must 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. For more information about Amazon Redshift integration with AWS CloudTrail, see Logging with CloudTrail.

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.

Configuring auditing using the console

Configure Amazon Redshift to export audit log data. Logs can be exported to CloudWatch, or as files to Amazon S3 buckets.

Enabling audit logging using the console

Console steps

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. On the Database configurations panel, choose Edit, then Edit audit logging.
4. On the Edit audit logging page, choose Turn on and select S3 bucket or CloudWatch. We recommend using CloudWatch because administration is easy and it has helpful features for data visualization.
5. Choose which logs to export.
6. To save your choices, choose Save changes.
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

Logging with CloudTrail

Logging calls with AWS CloudTrail

Amazon Redshift, data sharing, Amazon Redshift Serverless, Amazon Redshift Data API, and query editor v2 are all 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. CloudTrail captures all API calls for Amazon Redshift as events. The calls captured include calls from the Redshift console and code calls to the Redshift operations.

If you create a CloudTrail trail, you can have continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for 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 things. These include the request that was made to Redshift, the IP address from which the request was made, who made the request, when it was made, and additional details.

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.

Working with information in CloudTrail

CloudTrail is turned on in your AWS account when you create the account. When activity occurs, 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 Redshift, create a trail. CloudTrail uses trails 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, Amazon Redshift Serverless, Data API, data sharing, and query editor v2 actions are logged by CloudTrail. For example, calls to the **AuthorizeDataShare**, **CreateNamespace**, **ExecuteStatement**, and **CreateConnection** 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 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 files entries**

A *trail* is a configuration that allows for 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.

**Amazon Redshift Datashare example**

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": "AKIAI44QH8DHBEXAMPLE",
    "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.metall.x86_64 botocore/1.17.41",
  "requestParameters": {
```
Amazon Redshift Serverless example

Amazon Redshift Serverless is integrated with AWS CloudTrail to provide a record of actions taken in Amazon Redshift Serverless. CloudTrail captures all API calls for Amazon Redshift Serverless as events. For more information about Amazon Redshift Serverless features, see Amazon Redshift Serverless feature overview.

The following example shows a CloudTrail log entry that demonstrates the CreateNamespace action.

```json
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AAKEOFPINEXAMPLE:admin",
        "arn": "arn:aws:sts::111111111111:assumed-role/admin/admin",
        "accountId": "111111111111",
        "accessKeyId": "AAKEOFPINEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AAKEOFPINEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/admin",
                "accountId": "111111111111",
                "userName": "admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-03-21T20:51:58Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-03-21T23:15:40Z",
    "eventSource": "aws.redshift",
    "eventName": "CreateNamespace",
    "responseElements": {
        "dataShareArn": "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",
        "producerNamespaceArn": "arn:aws:redshift:us-east-1:111122223333:namespace:4c64c6ec-73d5-42be-869b-b7f7c43c7a53",
        "allowPubliclyAccessibleConsumers": true,
        "dataShareAssociations": [
            {
                "consumerIdentifier": "555555555555",
                "status": "AUTHORIZED",
                "createdDate": "Aug 2, 2021 11:40:56 PM",
                "statusChangeDate": "Aug 2, 2021 11:40:57 PM"
            }
        ]
    },
    "requestID": "87ee1c99-9e41-42be-a5c4-00495f928422",
    "eventID": "03a3d818-37c8-46a6-aad5-0151803bdb09",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```
"eventSource": "redshift-serverless.amazonaws.com",
"eventName": "CreateNamespace",
"awsRegion": "us-east-1",
"sourceIPAddress": "56.23.155.33",
"userAgent": "aws-cli/2.4.14 Python/3.8.8 Linux/5.4.181-109.354.amzn2int.x86_64.exe/x86_64.amzn2 prompt/off command/redshift-serverless.create-namespace",
"requestParameters": {
  "adminUserPassword": "HIDDEN_DUE_TO_SECURITY_REASONS",
  "adminUsername": "HIDDEN_DUE_TO_SECURITY_REASONS",
  "dbName": "dev",
  "namespaceName": "testnamespace"
},
"responseElements": {
  "namespace": {
    "adminUsername": "HIDDEN_DUE_TO_SECURITY_REASONS",
    "creationDate": "Mar 21, 2022 11:15:40 PM",
    "defaultIamRoleArn": "",
    "iamRoles": [],
    "logExports": [],
    "namespaceId": "8b726a0c-16ca-4799-acca-1d27cb403599",
    "namespaceName": "testnamespace",
    "status": "AVAILABLE"
  }
},
"requestID": "ed4bb777-8127-4daa-bac003999163",
"eventID": "1dbee944-f889-4beb-b22b-7ad0f3126464",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111111111111",
"eventCategory": "Management",
}

Amazon Redshift Data API examples

The following example shows a CloudTrail log entry that demonstrates the ExecuteStatement action.

[
  
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE:janedoe",
    "arn": "arn:aws:sts::123456789012:user/janedoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAI44QH8DHEXAMPLE",
    "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.205.84.352.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",
    "database": "example-database-name",
    "dbUser": "example_db_user_name",
    "sql": "***OMITTED***"
  }
]
The following example shows a CloudTrail log entry that demonstrates the `ExecuteStatement` action showing the `clientToken` used for idempotency.

```
{
  "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.metal.x86_64 botocore/1.17.41",
  "requestParameters": {
    "clusterIdentifier": "example-cluster-identifier",
    "database": "example-database-name",
    "dbUser": "example_db_user_name",
    "sql": "***OMITTED***",
    "clientToken": "32db2e10-69ac-4534-b3fc-a191052616ce"
  },
  "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-cd0612eeb8ac",
  "eventID": "c1fb7076-102f-43e5-9ec9-40820bccc175",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

Amazon Redshift query editor v2 example

The following example shows a CloudTrail log entry that demonstrates the `CreateConnection` action.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AAKEOFPINEXAMPLE:session",
    "arn": "arn:aws:sts::123456789012:assumed-role/your-assumed-role/your-identity",
    "accountId": "123456789012",
    "accessKeyId": "123456789012",
    "userName": "your-assumed-user"
  },
  "eventTime": "2020-08-19T17:55:59Z",
  "eventSource": "redshift-data.amazonaws.com",
  "eventName": "CreateConnection",
  "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.metal.x86_64 botocore/1.17.41",
  "requestParameters": {
    "clusterIdentifier": "example-cluster-identifier",
    "database": "example-database-name",
    "dbUser": "example_db_user_name",
    "sql": "***OMITTED***",
    "clientToken": "32db2e10-69ac-4534-b3fc-a191052616ce"
  },
  "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-cd0612eeb8ac",
  "eventID": "c1fb7076-102f-43e5-9ec9-40820bccc175",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```
"arn": "arn:aws:sts::123456789012:assumed-role/MyRole/session",
"accountId": "123456789012",
"accessKeyId": "AKIAI44Q8DBEXAMPLE",
"sessionContext": {
  "sessionIssuer": {
    "type": "Role",
    "principalId": "AAKEOFPINEXAMPLE",
    "arn": "arn:aws:iam::123456789012:role/MyRole",
    "accountID": "123456789012",
    "userName": "MyRole"
  },
  "webIdFederationData": {},
  "attributes": {
    "creationDate": "2022-09-21T17:19:02Z",
    "mfaAuthenticated": "false"
  }
},
"eventTime": "2022-09-21T22:02:05Z",
"eventSource": "sqlworkbench.amazonaws.com",
"eventName": "CreateConnection",
"awsRegion": "ca-central-1",
"sourceIPAddress": "192.2.0.2",
"userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:102.0) Gecko/20100101 Firefox/102.0",
"requestParameters": {
  "password": "***",
  "databaseName": "***",
  "isServerless": false,
  "name": "***",
  "host": "redshift-cluster-2.c8robpbxvbf9.ca-central-1.redshift.amazonaws.com",
  "authenticationType": "***",
  "clusterId": "redshift-cluster-2",
  "username": "***",
  "tags": {
    "sqlworkbench-resource-owner": "AAKEOFPINEXAMPLE:session"
  }
},
"responseElements": {
  "result": true,
  "code": "",
  "data": {
    "id": "arn:aws:sqlworkbench:ca-central-1:123456789012:connection/ce56b1be-
    dd65-4fbb-8b17-12345123456",
    "name": "***",
    "authenticationType": "***",
    "databaseName": "***",
    "secretArn": "arn:aws:secretsmanager:ca-
    central-1:123456789012:secret:sqlworkbench!7da333b4-9a07-4917-b1dc-12345123456-qTCoFm",
    "clusterId": "redshift-cluster-2",
    "dbUser": "***",
    "userSettings": "***",
    "recordDate": "2022-09-21 22:22:05",
    "updatedDate": "2022-09-21 22:22:05",
    "accountId": "123456789012",
    "tags": {
      "sqlworkbench-resource-owner": "AAKEOFPINEXAMPLE:session"
    },
    "isServerless": false
  }
},
"requestID": "9b82f4f83-9c03-4cdd-bb49-a7009e7da714",
"eventID": "a7cdd442-e92f-46a2-bc82-2325588d41c3",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
Amazon Redshift Management Guide
Logging with 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 (Hyderabad) Region</td>
<td>ap-south-2</td>
<td>297058826802</td>
</tr>
<tr>
<td>Asia Pacific (Jakarta) Region</td>
<td>ap-southeast-3</td>
<td>623197973179</td>
</tr>
<tr>
<td>Asia Pacific (Melbourne) Region</td>
<td>ap-southeast-4</td>
<td>945512339897</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>
<tr>
<td>Europe (London) Region</td>
<td>eu-west-2</td>
<td>885798887673</td>
</tr>
<tr>
<td>Europe (Milan) Region</td>
<td>eu-south-1</td>
<td>041313461515</td>
</tr>
<tr>
<td>Europe (Paris) Region</td>
<td>eu-west-3</td>
<td>694668203235</td>
</tr>
<tr>
<td>Europe (Spain) Region</td>
<td>eu-south-2</td>
<td>028811157404</td>
</tr>
<tr>
<td>Europe (Stockholm) Region</td>
<td>eu-north-1</td>
<td>553461782468</td>
</tr>
</tbody>
</table>
### CloudTrail Log Entry Example

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": "AROAI5QPCMKLTL4VHFCYY:i-0f53e22dbe5df8a89",
        "arn": "arn:aws:sts::790247189693:assumed-role/prod-23264-role-wp/i-0f53e22dbe5df8a89",
        "accountId": "790247189693",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2017-03-03T16:24:54Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAI5QPCMKLTL4VHFCYY",
                "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": "50d2fe51-0035-11e7-ab67-17595a8411c8",
        "eventID": "619bad54-1764-4de4-a786-8898b0a77f40c",
        "readOnly": true,
        "resources": [
            {
                "ARN": "arn:aws:kms:us-east-2:123456789012:key/f8f4f94f-e588-4254-b7e8-078b99270be7",
                "accountId": "123456789012",
                "type": "AWS::KMS::Key"
            }
        ]
    }
}
```
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. Security Hub uses security controls to evaluate resource configurations and security standards to help you comply with various compliance frameworks. For more information about using Security Hub to evaluate Amazon Redshift resources, see [Amazon Redshift controls](#) in the [AWS Security Hub User Guide](#).

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 (CS)
- 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.

To move a cluster to another Availability Zone without any loss of data or changes to your applications, you can set up relocation for your cluster. With relocation, you can continue operations when there is an interruption of service on your cluster with minimal impact. When cluster relocation is turned on, Amazon Redshift might choose to relocate clusters in some situations. For more information on relocation in Amazon Redshift, see Managing cluster relocation in Amazon Redshift (p. 96).

In failure scenarios where an unexpected event happens in an Availability Zone, you can set up a multiple Availability Zones (Multi-AZ) deployment to ensure that your Amazon Redshift data warehouse can continue operating. Amazon Redshift deploys equal compute resources in two Availability Zones that can be accessed through a single endpoint. In the event of an entire Availability Zone failure, the remaining compute resources in the second Availability Zone will be available to continue processing workloads. For more information on Multi-AZ deployments, see Configuring Multi-AZ deployment (preview) (p. 99).

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 AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

You use AWS published API calls to access Amazon Redshift through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). 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.

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. 127).

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. 127).

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.

While configuring network access control lists associated with the subnet(s) your Amazon Redshift cluster is tagged with, ensure that the respective AWS Region's S3 CIDR ranges are added to the allowlist for both ingress and egress rules. Doing so lets you execute S3-based operations such as Redshift Spectrum, COPY, and UNLOAD without any disruptions.

The following example command parses the JSON response for all IPv4 addresses used in Amazon S3 in the us-east-1 Region.

```
curl https://ip-ranges.amazonaws.com/ip-ranges.json | jq -r '.prefixes[] | select(.region=="us-east-1") | select(.service=="S3") | .ip_prefix'
```

54.231.0.0/17
52.92.16.0/20
52.216.0.0/15

For instructions on how to get S3 IP ranges for a particular region, see AWS IP address ranges.

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. 121).

If you are on the EC2-Classic platform, you define a cluster security group and associate it with a cluster. The EC2-Classic platform is being retired. 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. 422).

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. 579)
- [Example: VPC endpoint policy to allow VPC access only to a specified IAM role](p. 579)
- [Example: VPC endpoint policy to allow VPC access only to a specified IAM principal (user)](p. 579)
• Example: VPC endpoint policy to allow read-only Amazon Redshift operations (p. 580)
• Example: VPC endpoint policy denying access to a specified cluster (p. 581)

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 role

The following VPC endpoint policy allows full access only to the IAM role redshiftrole 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:role/redshiftrole]
         }
      }
   ]
}
```

This is only a sample. In most use cases we recommend attaching permissions for specific actions to narrow the scope of permissions.

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.
Connecting using an interface VPC endpoint

This is only a sample. In most use cases we recommend attaching permissions to a role before assigning to a user. Additionally, we recommend using specific actions to narrow the scope of permissions.

**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](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_actions_resources.html) in the IAM User Guide.

```json
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:user/redshiftadmin"
        ]
      }
    }
  ]
}
```
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.

```json
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": [
          "123456789012"
        ]
      }
    },
    {
      "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. 575), 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. 77).
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. 586).
- 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](#).
- This guide provides examples of working with Amazon Redshift using the Java SDK. For more general AWS SDK code examples, see [Sample code & libraries](#).

**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. 589).
- For reference material on the Amazon Redshift CLI commands, see [Amazon Redshift](#) in the [AWS CLI Reference](#).

**Topics**

- [Using the AWS SDK for Java with Amazon Redshift](p. 583)
- [Signing an HTTP request](p. 586)
- [Setting up the Amazon Redshift CLI](p. 589)

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](#).
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 clusters using the AWS SDK for Java (p. 119) so that the file in the src directory is CreateAndModifyClusters.java.

3. Compile the code.

```
javac -cp sdk/lib/aws-java-sdk-1.3.18.jar -d bin src\CreateAndModifyClusters.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/*"
CreateAndModifyClusters
```

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(credentials);
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. 583). 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 that you can get from temporary credentials. 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

Users need programmatic access if they want to interact with AWS outside of the AWS Management Console. The way to grant programmatic access depends on the type of user that's accessing AWS.

To grant users programmatic access, choose one of the following options.

<table>
<thead>
<tr>
<th>Which user needs programmatic access?</th>
<th>To</th>
<th>By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workforce identity</td>
<td>Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td>Following the instructions for the interface that you want to use.</td>
</tr>
<tr>
<td>(Users managed in IAM Identity Center)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For the AWS CLI, see Configuring the AWS CLI to use AWS IAM Identity Center (successor to AWS Single Sign-On) in the AWS Command Line Interface User Guide.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For AWS SDKs, tools, and AWS APIs, see IAM Identity Center authentication in the AWS SDKs and Tools Reference Guide.</td>
</tr>
<tr>
<td>IAM</td>
<td>Use temporary credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td>Following the instructions in Using temporary credentials with AWS resources in the IAM User Guide.</td>
</tr>
<tr>
<td>(Not recommended) Use long-term credentials to sign programmatic requests to the AWS CLI, AWS SDKs, or AWS APIs.</td>
<td></td>
<td>• For the AWS CLI, see Authenticating using IAM user credentials in the AWS Command Line Interface User Guide.</td>
</tr>
</tbody>
</table>
Which user needs programmatic access?

<table>
<thead>
<tr>
<th>To</th>
<th>By</th>
</tr>
</thead>
</table>
|    | • For AWS SDKs and tools, see Authenticate using long-term credentials in the AWS SDKs and Tools Reference Guide.  
  • For AWS APIs, 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:
The canonical form of the request calculated for **Task 1: Create a Canonical Request (p. 587)** 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
defined through SHA256
```

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. 587)** is:

```
AWS4-HMAC-SHA256
20121207T000000Z
20121207/us-east-1/redshift/aws4_request
06b6bef4f4f60a5558b60c627cc6c5b5b5a959b9902b5ac2187be80cbac0714
```

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. 587)**. The service name to use in the credential scope is `redshift`.

For **Task 3: Create a Signature (p. 587)**, the derived key can be represented as:

```
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 `wJalrXUtncFEMI/K7MDENG/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 AKIAI0SFODN7EXAMPLE, the header (with line breaks added for readability) is:

```plaintext
588
```
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 a secret access key. Use temporary credentials instead of long-term access keys when possible. Temporary credentials include an access key ID, a secret access key, and a security token that indicates when the credentials expire. For more information, see Using temporary credentials with AWS resources in the IAM User Guide.

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. 589).

**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>master-username</td>
<td>Enter a user name.</td>
</tr>
<tr>
<td>master-user-password</td>
<td><strong>Note</strong> Enter a password other than the prompt shown.</td>
</tr>
</tbody>
</table>
To create your cluster, enter the following command.

```bash
aws redshift create-cluster --cluster-identifier examplecluster --master-username enter a username --master-user-password enter a password --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.

```json
{
   "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. 594)
- Monitoring Amazon Redshift using CloudWatch metrics (p. 595)
- Working with performance data in the Amazon Redshift console (p. 603)

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.
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](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch-overview.html) in the [Amazon CloudWatch User Guide](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/). For a further description of CloudWatch metrics for Amazon Redshift, see the following sections.

**Topics**
- [Amazon Redshift metrics](#)
- [Dimensions for Amazon Redshift metrics](#)
- [Amazon Redshift query and load performance data](#)

### Amazon Redshift metrics

The AWS/Redshift namespace includes the following metrics. Unless stated otherwise, metrics are collected at 1-minute intervals.

**Title**

<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. Units: Percent</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier, NodeID</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>DatabaseConnections</td>
<td>The number of database connections to a cluster. 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. Note 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). Units: Count (1/0) (HEALTHY/UNHEALTHY in the Amazon Redshift console) Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</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/0FF in the Amazon Redshift console).</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</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. 389).</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>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</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>ReadIOPS</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></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>ReadLatency</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></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>ReadThroughput</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></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>RedshiftManagedStorageTotalCapacity</td>
<td>Total managed storage capacity.</td>
</tr>
<tr>
<td></td>
<td>Units: Megabytes</td>
</tr>
<tr>
<td></td>
<td>Dimensions: ClusterIdentifier</td>
</tr>
<tr>
<td>TotalTableCount</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>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WLMQueueLength</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>WLMQueueWaitTime</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>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>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------</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>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>
<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, Database, Schema</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, Database, Schema</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>
Amazon Redshift Management Guide

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>
<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>NodeID</td>
<td>Filters requested data that is specific to the nodes of a cluster. NodeID is either “Leader”, “Shared”, or “Compute-N” 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></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>
</tbody>
</table>
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 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.

---

### Amazon Redshift query and load performance data

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueryExecutingDelete</td>
<td>Time spent executing delete queries.</td>
</tr>
<tr>
<td>QueryExecutingUpdate</td>
<td>Time spent executing update queries.</td>
</tr>
<tr>
<td>QueryExecutingCtas</td>
<td>Time spent executing create table as queries.</td>
</tr>
<tr>
<td>QueryExecutingUnload</td>
<td>Time spent executing unload queries.</td>
</tr>
<tr>
<td>QueryExecutingCopy</td>
<td>Time spent executing copy queries.</td>
</tr>
<tr>
<td>QueryCommit</td>
<td>Time spent committing.</td>
</tr>
</tbody>
</table>

| wlmid | The identifier for a workload management queue.                        |
| QueryPriority | The priority of the query. Possible values are CRITICAL, HIGHEST, HIGH, NORMAL, LOW, and LOWEST. |
| QueueName | The name of the workload management queue. |

---

The following table describes different aspects of query and load data you can access in the Amazon Redshift console.

### Query/Load data

<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>

<table>
<thead>
<tr>
<th>Query detail</th>
<th>Provides details on a particular query including:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Query properties such as the query ID, type, cluster the query was run on, and runtime.</td>
</tr>
<tr>
<td></td>
<td>• Details such as the status of the query and the number of errors.</td>
</tr>
<tr>
<td></td>
<td>• The SQL statement that was run.</td>
</tr>
<tr>
<td></td>
<td>• An explain plan if available.</td>
</tr>
<tr>
<td></td>
<td>• Cluster performance data during the query execution (for more information, see Viewing query history data (p. 609)).</td>
</tr>
<tr>
<td>Query/Load data</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Load summary</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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. 609)). |

### 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. 595)), 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. 624).

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. 603)
- Viewing query history data (p. 609)
- Viewing database performance data (p. 612)
- Viewing workload concurrency and concurrency scaling data (p. 616)
- Viewing queries and loads (p. 618)
- Viewing cluster metrics during load operations (p. 622)
- Analyzing workload performance (p. 622)
- Managing alarms (p. 623)
- Working with performance metrics in the CloudWatch console (p. 624)

### 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/](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.

![CPU utilization graph](image)

• **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.

• **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.
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. 619).
- 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, Datashares, 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. 618).

### Query history graphs

The following examples show graphs that are displayed in the new Amazon Redshift console.

**Note**

The Amazon Redshift console graphs only contain data for the latest 100,000 queries.

- **Query runtime**
Chapters

• Queries and loads

Queries and loads(100)

Filter queries

<table>
<thead>
<tr>
<th>Start time</th>
<th>Query</th>
<th>Status</th>
<th>Duration</th>
<th>SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 13th, 2020 01:00:55 PM</td>
<td>69248</td>
<td>Completed</td>
<td>11 min</td>
<td>with /query_templates/query67.tpla-0047-06872b3fecc8.stream_10cu_</td>
</tr>
<tr>
<td>Apr 13th, 2020 01:29:23 PM</td>
<td>70089</td>
<td>Completed</td>
<td>10 min</td>
<td>with /query_templates/query67.tpla-0047-06872b3fecc8.stream_10cu_</td>
</tr>
<tr>
<td>Apr 13th, 2020 11:18:35 AM</td>
<td>65543</td>
<td>Completed</td>
<td>9 min</td>
<td>with /query_templates/query67.tpla-0047-06872b3fecc8.stream_05cu_</td>
</tr>
<tr>
<td>Apr 13th, 2020 12:40:30 PM</td>
<td>68729</td>
<td>Completed</td>
<td>9 min</td>
<td>with /query_templates/query67.tpla-0047-06872b3fecc8.stream_10cu_</td>
</tr>
</tbody>
</table>

• CPU utilization
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.

Viewing database performance data
• 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
• Average queue wait time by priority

• Query duration by queue

• Query throughput by queue
Viewing workload concurrency and concurrency scaling data

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/](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 on the cluster graph]

- **Queued vs. Running queries per queue**

![Queued vs. Running queries per queue graph]

- **Concurrency scaling activity**

![Concurrency scaling activity graph]
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. 460).

**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. You might need to change settings on this page to find your query.
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. You might need to change settings on this page to find your query.
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**

- **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. 609).
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 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. 460).

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/.
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 or serverless instance. You can set one or more alarms on any of the metrics listed in Monitoring Amazon Redshift using CloudWatch metrics (p. 595).

For example, setting an alarm for high CPUUtilization on a cluster node helps indicate when the node is overutilized. An alarm for high DataStorage would keep track of the storage space that your serverless namespace is using for your data.
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/.
   If you're using Amazon Redshift Serverless, Choose Go to Serverless on the upper right of the dashboard.
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. 595).

<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
- Cluster events overview (p. 626)
- Working with Amazon Simple Notification Service (p. 626)
- Subscribing to Amazon Redshift cluster event notifications (p. 627)
- Viewing cluster events using the console (p. 628)
- Viewing cluster events using the AWS CLI and Amazon Redshift API (p. 628)
- Managing cluster event notifications (p. 629)
- Amazon Redshift event notifications (p. 630)
- Amazon Redshift Serverless event notifications with Amazon EventBridge (p. 640)

Cluster events overview

Amazon Redshift tracks cluster 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 cluster event notifications (p. 627).

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. 630)

Working with Amazon Simple Notification Service

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.
Subscribing to Amazon Redshift cluster 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. 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, Security, and Pending
- 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 sections list all categories and events that you can be notified of. It also provides information about subscribing to and working with Amazon Redshift event subscriptions.

**Viewing cluster 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 cluster events using the AWS CLI and Amazon Redshift API**

You can use the following Amazon Redshift CLI operation to view events.

- describe-events

Amazon Redshift provides the following API to view events.

- DescribeEvents
Managing cluster event notifications

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. 630).

Managing cluster event notifications using the Amazon Redshift 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 cluster event notifications using the AWS CLI and Amazon Redshift API

You can use the following Amazon Redshift CLI operations to manage cluster 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
Amazon Redshift event notifications

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 name] were not applied. Please visit the AWS Management Console to correct the issue.</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>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-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. 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. For more information, see see Enabling enhanced VPC routing in the Amazon Redshift Management Guide.</td>
</tr>
<tr>
<td>Configuration</td>
<td>REDSHIFT-EVENT-1511</td>
<td>ERROR</td>
<td>Amazon Redshift cluster [cluster name] cannot be created because you exceeded your account's limit of Elastic IP addresses. Delete unused Elastic IP addresses or request a limit increase with Amazon EC2.</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>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>VPC security groups for cluster [cluster name] updated at [time in UTC].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2003</td>
<td>INFO</td>
<td>Maintenance started on cluster [cluster name] at [time in UTC].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2004</td>
<td>INFO</td>
<td>Maintenance on cluster [cluster name] completed at [time in UTC].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2006</td>
<td>INFO</td>
<td>Cluster [cluster name] resize started at [time in UTC]. Cluster is in read-only mode.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2007</td>
<td>INFO</td>
<td>A resize request for cluster [cluster name] has been acknowledged.</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-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>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>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-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-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-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>Pending</td>
<td>REDSHIFT-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-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-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-3000</td>
<td>INFO</td>
<td>Your Amazon Redshift cluster [cluster name] was rebooted 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-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>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>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-3507 ERROR</td>
<td></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 ERROR</td>
<td></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 ERROR</td>
<td></td>
<td>Cross-region snapshot copy is not enabled for cluster [cluster name].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3510 ERROR</td>
<td></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 ERROR</td>
<td></td>
<td>Table restore failed for Amazon Redshift cluster [cluster name] at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3512 ERROR</td>
<td></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 ERROR</td>
<td></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 ERROR</td>
<td></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>Monitoring</td>
<td>REDSHIFT-EVENT-3515 ERROR</td>
<td></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 ERROR</td>
<td></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 INFO</td>
<td></td>
<td>Cluster [cluster name] began restart at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3520 INFO</td>
<td></td>
<td>Cluster [cluster name] completed restart 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-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>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>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-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>Amazon Redshift cluster [cluster name] was successfully paused at [UTC time].</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>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>
</tbody>
</table>
### Amazon Redshift Management Guide

Amazon Redshift event categories and event messages

<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-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>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**
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>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2011</td>
<td>INFO</td>
<td>The user snapshot [snapshot name] for Amazon Redshift cluster [cluster name] was deleted at [time].</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-EVENT-2012</td>
<td>INFO</td>
<td>The final snapshot [snapshot name] for Amazon Redshift cluster [cluster name] was started at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3004</td>
<td>INFO</td>
<td>The user snapshot [snapshot name] for your Amazon Redshift cluster [cluster name] completed successfully at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3005</td>
<td>INFO</td>
<td>The final snapshot [name] for Amazon Redshift cluster [name] completed successfully at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3006</td>
<td>INFO</td>
<td>The final snapshot [snapshot name] for Amazon Redshift cluster [cluster name] was cancelled at [time].</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-EVENT-3502</td>
<td>ERROR</td>
<td>The final snapshot [snapshot name] for Amazon Redshift cluster [cluster name] failed at [time]. The team is investigating</td>
</tr>
</tbody>
</table>
Amazon Redshift Serverless event notifications with Amazon EventBridge

Amazon Redshift Serverless uses Amazon EventBridge to manage event notifications to keep you up-to-date regarding changes in your data warehouse. Amazon EventBridge is a serverless event bus service that you can use to connect your applications with data from a variety of sources. In this case, the event source is Amazon Redshift. Events, which are monitored changes in an environment, are sent to EventBridge from your Amazon Redshift data warehouse automatically. Events are delivered in near-real time.

Capabilities of EventBridge include providing an environment for you to write event rules, which can specify actions to take for specific events. You can also set up targets, which are resources that EventBridge can send an event to. A target can include an API destination, an Amazon CloudWatch log group, and others. For more information about rules, see Amazon EventBridge rules. For more information about targets, see Amazon EventBridge targets.

Events can be classified into severities and categories. The following filters are available:

- **Resource filtering** – Receive messages based on the resource the events are associated with. Resources include a workgroup, a snapshot, and so on.
- **Time window filtering** – Scope events in a specific time period.
- **Category filtering** – Receive event notifications for all events in specified categories.

The following table includes Amazon Redshift Serverless events, with additional metadata:

<table>
<thead>
<tr>
<th>Amazon Redshift Category</th>
<th>External Event ID</th>
<th>Event Severity</th>
<th>Message Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RateChange</td>
<td>REDSHIFT-SERVERLESS-EVENT-1001</td>
<td>INFO</td>
<td>Workgroup base RPU change completed successfully at &lt;time in UTC&gt;.</td>
</tr>
<tr>
<td>RateChange</td>
<td>REDSHIFT-SERVERLESS-EVENT-1002</td>
<td>ERROR</td>
<td>Workgroup base RPU change failed to complete at &lt;time in UTC&gt;.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-SERVERLESS-EVENT-1003</td>
<td>INFO</td>
<td>The software was updated on your</td>
</tr>
<tr>
<td>Amazon Redshift Category</td>
<td>External Event ID</td>
<td>Event Severity</td>
<td>Message Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-SERVERLESS-EVENT-1500</td>
<td>ERROR</td>
<td>Workgroup &lt;workgroup name&gt; cannot be created or updated because you exceeded your account’s limit of Elastic IP addresses. Delete unused Elastic IP addresses or request a limit increase with Amazon EC2.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-SERVERLESS-EVENT-1501</td>
<td>ERROR</td>
<td>Subnet &lt;subnet id&gt; has no available IP addresses. This will prevent the following query types from running successfully on workgroup &lt;workgroup name&gt;: EMR, federated queries, COPY/UNLOAD from Amazon EC2. To correct the issue, free up IPs in your subnet by deleting ENIs.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-SERVERLESS-EVENT-1502</td>
<td>ERROR</td>
<td>Subnet &lt;subnet id&gt; has no available IP addresses. This will prevent the Amazon EMR, Redshift federated queries, Redshift COPY/UNLOAD, Redshift ML query types from running successfully in workgroup &lt;workgroup name&gt;. To correct the issue, free up IPs in your subnet by deleting unused elastic network interfaces (ENIs).</td>
</tr>
<tr>
<td>Amazon Redshift Category</td>
<td>External Event ID</td>
<td>Event Severity</td>
<td>Message Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-SERVERLESS-EVENT-1008</td>
<td>INFO</td>
<td>Your Amazon Redshift workgroup <code>&lt;workgroup name&gt;</code> has been created and is ready for use.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-SERVERLESS-EVENT-1009</td>
<td>INFO</td>
<td>Your Amazon Redshift workgroup <code>&lt;workgroup name&gt;</code> was deleted at <code>&lt;time in UTC&gt;</code>.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>REDSHIFT-SERVERLESS-EVENT-1000</td>
<td>INFO</td>
<td>Snapshot <code>&lt;snapshot name&gt;</code> completed successfully at <code>&lt;time in UTC&gt;</code>.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-SERVERLESS-EVENT-1004</td>
<td>INFO</td>
<td>Restore from snapshot on namespace <code>&lt;namespace name&gt;</code> completed successfully at <code>&lt;time in UTC&gt;</code>.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-SERVERLESS-EVENT-1005</td>
<td>ERROR</td>
<td>Restore from snapshot on namespace <code>&lt;namespace name&gt;</code> failed at <code>&lt;time in UTC&gt;</code>.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-SERVERLESS-EVENT-1006</td>
<td>INFO</td>
<td>Restore from recovery point on namespace <code>&lt;namespace name&gt;</code> completed successfully at <code>&lt;time in UTC&gt;</code>.</td>
</tr>
<tr>
<td>Management</td>
<td>REDSHIFT-SERVERLESS-EVENT-1007</td>
<td>INFO</td>
<td>Restore from recovery point on namespace <code>&lt;namespace name&gt;</code> failed at <code>&lt;time in UTC&gt;</code>.</td>
</tr>
</tbody>
</table>
Quotas and limits in Amazon Redshift

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.

Quotas for Amazon Redshift objects

Amazon Redshift has quotas that limit the use of several object types. 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>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>50¹</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></td>
<td></td>
<td></td>
<td>¹The quota is 10 in the following AWS Regions: us-isoeast-1, us-isowest-1, us-isob-east-1.</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>
<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>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. 65).</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. 65).</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. 65).</td>
</tr>
<tr>
<td>Redshift-managed VPC endpoints connected to a cluster</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. 109).</td>
</tr>
<tr>
<td>Grantees to cluster accessed through a Redshift-managed VPC endpoint</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. 109).</td>
</tr>
<tr>
<td>Redshift-managed VPC endpoints per authorization</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. 109).</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>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>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. 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>
## Quotas for Amazon Redshift objects

<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>200,000</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>200,000</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>200,000</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>Number of databases</td>
<td>60</td>
<td>No</td>
<td>The maximum allowed count of databases in an Amazon Redshift cluster. This excludes databases created from datashares.</td>
</tr>
<tr>
<td>Timeout for idle or inactive sessions</td>
<td>1 hour</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>Timeout for idle transactions</td>
<td>6 hours</td>
<td>No</td>
<td>The maximum period of inactivity for an open transaction before Amazon Redshift ends the session associated with the transaction. This setting takes precedence over any user-defined idle timeout setting. It applies to the cluster.</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>
<tr>
<td>Maximum number of connections</td>
<td>2,000</td>
<td>No</td>
<td>The maximum number of connections to an RA3 cluster. (This applies specifically to the ra3.xlplus, ra3.4xlarge and ra3.16xlarge node types.) The maximum connections allowed varies by node type.</td>
</tr>
</tbody>
</table>
Quotas for Amazon Redshift Serverless objects

Amazon Redshift has quotas that limit the use of several object types in your Amazon Redshift Serverless instance. 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 in an Amazon Redshift Serverless namespace. This excludes databases created from datashares.</td>
</tr>
<tr>
<td>Number of schemas</td>
<td>9,900</td>
<td>No</td>
<td>The maximum allowed count of schemas in an Amazon Redshift Serverless instance.</td>
</tr>
<tr>
<td>Number of tables</td>
<td>200,000</td>
<td>No</td>
<td>The maximum allowed count of tables in an Amazon Redshift Serverless instance.</td>
</tr>
<tr>
<td>Timeout for idle or inactive sessions</td>
<td>4 hours</td>
<td>No</td>
<td>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.</td>
</tr>
<tr>
<td>Timeout for a running query</td>
<td>86,399 seconds (24 hours)</td>
<td>No</td>
<td>The maximum time for a running query before Amazon Redshift ends it.</td>
</tr>
<tr>
<td>Timeout for idle transactions</td>
<td>6 hours</td>
<td>No</td>
<td>The maximum period of inactivity for an open transaction before Amazon Redshift Serverless ends the session associated with the transaction. This setting takes precedence over any user-defined idle timeout setting.</td>
</tr>
<tr>
<td>Number of maximum connections</td>
<td>2000</td>
<td>No</td>
<td>The maximum number of connections allowed to connect to a workgroup.</td>
</tr>
<tr>
<td>Number of workgroups</td>
<td>25</td>
<td>Yes</td>
<td>The number of workgroups supported.</td>
</tr>
<tr>
<td>Number of namespaces</td>
<td>25</td>
<td>Yes</td>
<td>The number of namespaces supported.</td>
</tr>
</tbody>
</table>

For more information about how Amazon Redshift Serverless billing is affected by timeout configuration, see Billing for Amazon Redshift Serverless (p. 24).

Quotas for query editor v2 objects

Amazon Redshift has quotas that limit the use of several object types in your Amazon Redshift query editor v2. 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>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>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>No</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>No</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 per principal</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 per account</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>
<tr>
<td>Maximum concurrent connections</td>
<td>3</td>
<td>No</td>
<td>Maximum database connections per user (includes isolated sessions). This value can be set from 1–10 by the query editor v2 administrator in Account settings. If you reach the limit set by your administrator, consider using shared sessions instead of isolated sessions when running your SQL. For more information about connections, see Opening query editor v2 (p. 152). For more information about setting the limit, see Changing account settings (p. 162).</td>
</tr>
</tbody>
</table>

Quotas and limits for Amazon Redshift Spectrum objects

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://aws.amazon.com/documentation/redshift/devguide/#aws-glue-service-quotas) in the [Amazon Web Services General Reference](https://aws.amazon.com/documentation/redshift/).

• The maximum number of tables per database when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://aws.amazon.com/documentation/redshift/devguide/#aws-glue-service-quotas) in the [Amazon Web Services General Reference](https://aws.amazon.com/documentation/redshift/).

• The maximum number of partitions per table when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://aws.amazon.com/documentation/redshift/devguide/#aws-glue-service-quotas) in the [Amazon Web Services General Reference](https://aws.amazon.com/documentation/redshift/).

• The maximum number of partitions per AWS account when using an AWS Glue Data Catalog. For this value, see [AWS Glue service quotas](https://aws.amazon.com/documentation/redshift/devguide/#aws-glue-service-quotas) in the [Amazon Web Services General Reference](https://aws.amazon.com/documentation/redshift/).

• The maximum number of columns for external tables when using an AWS Glue Data Catalog, 1,597 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. The string can be truncated if you reach this limit.

• 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](https://en.wikipedia.org/wiki/ISO_8601) format.

• External compression of ORC files is not supported.

• Text, OpenCSV, and Regex SERDEs do not support octal delimiters larger than \177.\n
• 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>Cluster identifier</th>
<th>A cluster identifier must contain only lowercase characters.</th>
</tr>
</thead>
<tbody>
<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>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Database name</th>
<th>A database name must contain 1–64 alphanumeric characters.</th>
</tr>
</thead>
<tbody>
<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 <a href="https://aws.amazon.com/documentation/redshift/devguide/#reserved-words">Reserved words</a> in the <a href="https://aws.amazon.com/documentation/redshift/">Amazon Redshift Database Developer Guide</a>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Endpoint name of a Redshift-managed VPC endpoint</th>
<th>An endpoint name must contain 1–30 characters.</th>
</tr>
</thead>
<tbody>
<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>Naming constraint</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Admin user name   | - An admin user name must contain only lowercase characters.  
                        - It must contain 1–128 alphanumeric characters.  
                        - Its first character must be a letter.  
                        - It cannot be a reserved word. For a list of reserved words, see [Reserved words](#) in the [Amazon Redshift Database Developer Guide](#). |
| Admin password    | - An admin password must contain 8–64 characters.  
                        - It must contain at least one uppercase letter.  
                        - It must contain at least one lowercase letter.  
                        - It must contain one number.  
                        - It can use any ASCII characters with ASCII codes 33–126, except ' (single quote), " (double quote), \, /, or @. |
| Parameter group name | - 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. |
| Subnet group name  | - 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 subnet groups that are created by your AWS account. |
| Cluster snapshot identifier | - 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 snapshot identifiers that are created by your AWS account. |

- The name can't contain two consecutive hyphens or end with a hyphen.
Tagging resources in Amazon Redshift

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 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/](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
Cluster versions for Amazon Redshift

Amazon Redshift regularly releases cluster versions. Your Amazon Redshift clusters are patched during your system maintenance window. The timing of the patch depends on your AWS Region and maintenance window settings. You can view or change your maintenance window settings from the Amazon Redshift console. For more information about maintenance, see Cluster maintenance (p. 77).

You can view the cluster version of your cluster on the Amazon Redshift console on the Maintenance tab of the cluster details. Or you can see the cluster version in the output of the SQL command:

```
SELECT version();
```

Topics

- Amazon Redshift patch 177 (p. 654)
- Amazon Redshift patch 176 (p. 655)
- Amazon Redshift patch 175 (p. 656)
- Amazon Redshift patch 174 (p. 657)
- Amazon Redshift patch 173 (p. 658)
- Amazon Redshift patch 172 (p. 659)
- Amazon Redshift patch 171 (p. 660)
- Amazon Redshift patch 170 (p. 660)
- Amazon Redshift patch 169 (p. 660)
- Amazon Redshift patch 168 (p. 661)

Amazon Redshift patch 177

Cluster versions in this patch:

- 1.0.54899 - Current track version – Released on August 15th, 2023
- 1.0.54239 - Current track version – Released on August 3rd, 2023
- 1.0.54321 - Amazon Redshift Serverless version – Released on August 3rd, 2023

New features and improvements in this patch

- Adds the SYS_MV_STATE view, which contains a row for every state transition of a materialized view. SYS_MV_STATE can be used for MV refresh monitoring for Amazon Redshift Serverless and Amazon Redshift provisioned instances.
- Adds the SYS_USERLOG view, which records details for the changes to a database user for Create user, Drop user, Alter user (rename), Alter user (alter properties).
- Adds the SYS_COPY_REPLACEMENTS view, which displays a log that records when invalid UTF-8 characters were replaced by the COPY command with the ACCEPTINVCHARS option.
- Adds the SYS_SPATIAL_SIMPLIFY view, which contains information about simplified spatial geometry objects using the COPY command.
- Adds the SYS_VACUUM_HISTORY view, which you can use to see the details and results of VACUUM operations.
• Adds the SYS_SCHEMA_QUOTA_VIOLATIONS view to record the occurrence, timestamp, XID, and other useful information when a schema quota is exceeded.
• Adds the SYS_RESTORE_STATE view, which you can use to monitor the redistribution progress of each table in the cluster during asynchronous classic resize.
• Adds the SYSEXTERNAL_QUERY_ERROR view that returns information about Redshift Spectrum scan errors.
• Adds the tag parameter to the CREATE MODEL command, so you can now track training costs with autopilot training jobs.
• Adds custom domain names (CNAME) for Amazon Redshift clusters.
• Adds preview support for Apache Iceberg, enabling customers to run analytics queries on Apache Iceberg tables within Amazon Redshift.
• Adds support for using user roles with parameter groups in workload management (WLM).
• Adds support for automatic mounting of AWS Glue Data Catalog, making it easier for customers to run queries in their data lakes.
• Adds functionality such that using grouping functions without a GROUP BY clause or using grouping operations in a WHERE clause results in an error.
• Adds functionality to stored procedures to allow for catching divide by zero errors and handling inside a exception-handling block.
• Fixes a bug that prevented queries from using concurrency scaling to write data to tables when the source table is a data sharing table.
• Fixes the case-sensitive identifier documented at enable_case_sensitive_identifier to now work with MERGE statements.
• Fixes the bug that a query on the function pg_get_late_binding_view_cols() might get ignored occasionally. You can now always cancel such queries.
• Improves performance for data sharing queries running on consumers when running vacuum jobs on the producer.
• Improves performance for data sharing and concurrency scaling queries, especially with concurrent data changes on the producer or when offloading to a concurrency scaling instance attached to the consumer.

Amazon Redshift patch 176

Cluster versions in this patch:
• 1.0.54776 - Current track version – released on August 15th, 2023
• 1.0.54052 - Current track version – Released on July 26th, 2023
• 1.0.53642 - Amazon Redshift Serverless version – Released on July 20th, 2023
• 1.0.53301 - Current track version – Released on July 20th, 2023
• 1.0.52943 - Amazon Redshift Serverless version – Released on July 7, 2023
• 1.0.52931 - Current track version – Released on July 7, 2023
• 1.0.52194 - Amazon Redshift Serverless version – Released on June 21, 2023
• 1.0.51986 - Current track version – Released on June 16, 2023
• 1.0.51594 - Current track version – Released on June 9, 2023

New features and improvements in this patch
• Improved error handling when writing GROUP BY () for an empty grouping set. This was ignored previously and now returns a parser error.
• Performance enhancements for incrementally refreshing materialized views with SUPER columns.
• ALTER TABLE <target_tbl> APPEND FROM <streaming_mv> – (ATA) SQL command now supports moving all records from a streaming materialized view (MV) as a source, in addition to tables as a source, to a target table. The support for ATA on streaming MVs allows users to rapidly purge all records in a streaming MV by moving them to another table to manage data growth.
• TRUNCATE <streaming_mv> – SQL command now supports truncating all records in a streaming materialized view (MV), in addition to tables. TRUNCATE deletes all records in the streaming MV, while leaving the streaming MV structure intact. Running TRUNCATE on streaming MVs allows customers to rapidly purge all records in a streaming MV to manage data growth.
• Added functionality for the QUALIFY clause to the SELECT command.
• Redshift machine-learning support for time series forecasting by integrating with Amazon Forecast.
• AWS Glue Data Catalog auto mounting is supported to simplify querying a data lake without extra steps to create external schema references.
• Altering an RLS policy is now supported. Refer to the documentation for more details at ALTER RLS POLICY.
• Lambda UDFs now support the STABLE function-volatility parameter in the CREATE FUNCTION statement. When the STABLE parameter is used in the CREATE FUNCTION statement and the Lambda UDF is called multiple times, with the same arguments, the expected number of Lambda UDF function invocations is decreased. The STABLE function volatility category is explained in more detail in the CREATE FUNCTION parameters.
• Multiple Lambda UDF performance improvements. Specifically, improved record batching support when querying a table protected by a row-level security (RLS) policy.
• Reduction in the overall encryption time for Amazon Redshift RA3 clusters and improvement in the availability of the data warehouse during encryption. For more information, see Amazon Redshift database encryption.
• A new system view SYS_MV_REFRESH_HISTORY has been added to Redshift. The SYS_MV_REFRESH_HISTORY view contains a row for the refresh activity of materialized views. Using SYS_MV_REFRESH_HISTORY, you can check the refresh history of materialized views. SYS_MV_REFRESH_HISTORY is visible to all users. Superusers can see all rows; regular users can see only their own data.

A new column SPILLED_BLOCK_LOCAL_DISK has been added to system view SYS_QUERY_DETAIL. The new column SPILLED_BLOCK_LOCAL_DISK helps customers to determine blocks spilled to local disk. You can use SYS_QUERYDETAIL to view details for queries at a step level. SYS_QUERY_DETAIL is visible to all users. Superusers can see all rows; regular users can see only metadata to which they have access.
• A new system view, SYS_QUERY_TEXT, has been added to Amazon Redshift Serverless and Amazon Redshift provisioned. The SYS_QUERY_TEXT view is similar to SVL_STATEMENTTEXT for provisioned clusters. Use the sequence column in the SYS_QUERY_TEXT view to get complete SQL statement text.
New features

- 1.0.50300 - Current track version – Released on May 8, 2023
- 1.0.49710 - Amazon Redshift Serverless version – Released on April 28, 2023
- 1.0.49676 - Current track version – Released on April 28, 2023

New features and improvements in this patch

- Minor bug fixes.
- Amazon Redshift streaming ingestion now supports cross-region streaming ingestion where your source Amazon Kinesis Data Streams (KDS) or Amazon Managed Streaming for Apache Kafka (MSK) topic can be located in an AWS region that is different from the AWS region where your Amazon Redshift data warehouse is located. The documentation at Getting started with streaming ingestion from Amazon Kinesis Data Streams has been revised and explains how the REGION keyword is used.
- Egypt daylight saving adjustment.
- Improved overall times for encryption of RA3 clusters.

Amazon Redshift patch 174

1.0.51296 – Released on June 2, 2023
Release to trailing track. No release notes.

1.0.50468 – Released on May 12, 2023
Maintenance release. No release notes.

1.0.49780, 1.0.49868, and 1.0.49997 – Released on April 28, 2023
Release notes for this version:

- Improved batching support for Lambda UDF.
- Incremental batching for Lambda UDF.
- New MERGE SQL command to apply source data changes to Amazon Redshift tables.
- New dynamic data masking capability to simplify the process of protecting sensitive data in an Amazon Redshift data warehouse.
- New centralized access control for data sharing with Lake Formation that allows managing permission grants, viewing access controls, and auditing permissions on the tables and views in the Amazon Redshift datashares using Lake Formation APIs and the AWS Console.
- Egypt daylight saving adjustment.

1.0.49087 – Released on April 12, 2023
Maintenance release. No release notes.

1.0.48805 – Released on April 5, 2023
Release notes for this version:
• Amazon Redshift introduced additional performance enhancements to string-heavy queries using BYTEDICT, a new compression encoding in Amazon Redshift that speeds up string-based data processing between 5x to 63x compared to alternative compression encodings such as LZO or ZSTD. For more information on this feature, see Byte-dictionary encoding in the Amazon Redshift Database Developer Guide.

1.0.48004 – Released on March 17, 2023

Maintenance release. No release notes.

1.0.47470 – Released on March 11, 2023

Release notes for this version:
• Improves query performance on pg_catalog.svv_table_info. Also adds new column create_time. When creating a table, this column stores the date/time stamp in UTC.
• Adds support for specifying session level timeout on federated query.

Amazon Redshift patch 173

1.0.49788 – Released on April 28, 2023

Release notes for this version:
• Egypt daylight saving adjustment.

1.0.49074 – Released on April 12, 2023

Release notes for this version:
• Timezone configuration updated to IANA library release 2022g.

1.0.48766 – Released on April 5, 2023

Maintenance release. No release notes.

1.0.48714 – Released on April 5, 2023

Maintenance release. No release notes.

1.0.48022 – Released on March 17, 2023

Maintenance release. No release notes.

1.0.47357 – Released on March 7, 2023

Maintenance release. No release notes.
1.0.46987 – Released on February 24, 2023
Maintenance release. No release notes.

1.0.46806 – Released on February 18, 2023
Maintenance release. No release notes.

1.0.46607 – Released on February 13, 2023
Release notes for this version:

• We now automatically convert tables with manually set interleaved sort keys to compound sort keys if their distribution style has been set to DISTSTYLE KEY, to improve the performance of these tables. This is done at the time of restoring a snapshot into Amazon Redshift Serverless.

1.0.45698 – Released on January 20, 2023
Release notes for this version:

• Adds a file extension parameter to the UNLOAD command, so file extensions are automatically added to filenames.
• Supports protecting RLS-protected objects by default when adding them to a datashare or if they’re already part of a datashare. Administrators can now turn off RLS for datashares to allow consumers access to the protected object.
• Adds new system tables for monitoring: SVV_ML_MODEL_INFO, SVV_MV_DEPENDENCY, and SYS_LOAD_DETAIL. Also adds the columns data_skewness and time_skewness to the system table SYS_QUERY_DETAIL.

Amazon Redshift patch 172

Cluster versions in this patch:

• 1.0.46534 – Released on February 18, 2023
• 1.0.46523 – Released on February 13, 2023
• 1.0.46206 – Released on February 1, 2023
• 1.0.45603 – Released on January 20, 2023
• 1.0.44924 – Released on December 19, 2022
• 1.0.44903 – Released on December 18, 2022
• 1.0.44540 – Released on December 13, 2022
• 1.0.44126 – Released on November 23, 2022
• 1.0.43980 – Released on November 17, 2022

New features and improvements in this patch

• Tables created by CTAS are AUTO by default.
• Adds support for row-level security (RLS) on materialized views.
• Increases the S3 timeout to improve cross-Region data sharing.
• Adds new spatial function `ST_GeomFromGeohash`.
• Improves automatic selection of distribution key from composite primary keys to improve out-of-the-box performance.
• Adds automatic primary key to distribution key for tables with composite primary keys, improving out-of-the-box performance.
• Improves concurrency scaling to allow more queries to scale even as data changes.
• Improves data sharing query performance.
• Adds Machine Learning probability metrics for classification models.
• Adds new system tables for monitoring: `SVV_USER_INFO`, `SVV_MV_INFO`, `SYS_CONNECTION_LOG`, `SYS_DATASHARE_USAGE_PRODUCER`, `SYS_DATASHARE_USAGE_CONSUMER`, and `SYS_DATASHARE_CHANGE_LOG`.
• Adds support for querying VARBYTE columns in external tables for Parquet and ORC file types.

Amazon Redshift patch 171

Cluster versions in this patch:

• 1.0.43931 – Released on November 16, 2022
• 1.0.43551 – Released on November 5, 2022
• 1.0.43331 – Released on September 29, 2022
• 1.0.43029 – Released on September 26, 2022

New features and improvements in this patch

• CONNECT BY support: Adds support for the CONNECT BY SQL construct, letting you recursively query the hierarchical data in your data warehouse based on parent-child relationship within that data set.

Amazon Redshift patch 170

Cluster versions in this patch:

• 1.0.43922 – Released on November 21, 2022
• 1.0.43573 – Released on November 7, 2022
• 1.0.41881 – Released on September 20, 2022
• 1.0.41465 – Released on September 7, 2022
• 1.0.40325 – Released on July 27, 2022

New features and improvements in this patch

• `ST_GeomFromGeoJSON`: Constructs an Amazon Redshift spatial geometry object from VARCHAR in GeoJSON representation.

Amazon Redshift patch 169

Cluster versions in this patch:
New features and improvements in this patch

- Adds role as a parameter for the Alter Default Privileges command to support Role-based Access Control.
- Adds ACCEPTINVCHARS parameter to support replacing invalid UTF-8 characters when copying from Parquet and ORC files.
- Adds OBJECT(k,v) function to construct SUPER objects from key and value pairs.

Amazon Redshift patch 168

Cluster versions in this patch:

- 1.0.38698 – Released on May 25, 2022
- 1.0.38551 – Released on May 20, 2022
- 1.0.38463 – Released on May 18, 2022
- 1.0.38361 – Released on May 13, 2022
- 1.0.38199 – Released on May 9, 2022
- 1.0.38112 – Released on May 6, 2022
- 1.0.37684 – Released on April 20, 2022

New features and improvements in this patch

- Adds support for the Linear Learner model type in Amazon Redshift ML.
- Adds SNAPSHOT option for SQL transaction isolation level.
- Adds farmhashFingerprint64 as new hashing algorithm for VARBYTE and VARCHAR data.
- Supports the AVG function in incremental refresh of materialized views.
- Supports correlated sub-queries on external tables in Redshift Spectrum.
- To improve the out-of-the-box query performance, Amazon Redshift automatically chooses a single column primary key for specific tables as a distribution key.
The following table describes the important changes in each release of the *Amazon Redshift Management Guide* after June 2018. For notification about updates to this documentation, you can subscribe to an RSS feed.

**API version: 2012-12-01**

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>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Amazon Redshift patch 177 released. (p. 654)</strong></td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see <a href="#">Amazon Redshift patch 177</a>.</td>
<td>August 3, 2023</td>
</tr>
<tr>
<td><strong>Amazon Redshift patch 176 released. (p. 655)</strong></td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see <a href="#">Amazon Redshift patch 176</a>.</td>
<td>June 8, 2023</td>
</tr>
<tr>
<td><strong>Amazon Redshift patch 175 released. (p. 656)</strong></td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see <a href="#">Amazon Redshift patch 175</a>.</td>
<td>April 28, 2023</td>
</tr>
<tr>
<td>Update <a href="#">Amazon Redshift managed policy</a></td>
<td>Updates to <em>AmazonRedshiftServiceLinkedRolePolicy</em> managed policy to remove permissions for ec2 network-related actions. These were specifically associated with the *Purpose:*RedshiftMigrateToVpc resource tag.</td>
<td>April 27, 2023</td>
</tr>
<tr>
<td>Update <a href="#">Amazon Redshift Data API managed policy</a></td>
<td>Updates to <em>AmazonRedshiftDataFullAccess</em> managed policy with permission <code>redshift:GetClusterCredentialsWithIAM</code>.</td>
<td>April 7, 2023</td>
</tr>
</tbody>
</table>
| **Update query editor v2 managed policies** | Updates to March 21, 2023
AmazonRedshiftQueryEditorV2NoSharing, AmazonRedshiftQueryEditorV2ReadSharing, and AmazonRedshiftQueryEditorV2ReadWriteSharing managed policies with permission sqlworkbench:GetSchemaInference. |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Amazon Redshift patch 174 released. (p. 657)</strong></td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see Amazon Redshift patch 174.</td>
</tr>
</tbody>
</table>
| **Update query editor v2 managed policies** | Updates to February 2, 2023
AmazonRedshiftQueryEditorV2NoSharing, AmazonRedshiftQueryEditorV2ReadSharing, and AmazonRedshiftQueryEditorV2ReadWriteSharing managed policies with permission sqlworkbench:AssociateNotebookWithTab. |
<p>| <strong>Amazon Redshift patch 173 released. (p. 658)</strong> | A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see Amazon Redshift patch 173. |
| <strong>Amazon Redshift patch 172 released. (p. 659)</strong> | A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see Amazon Redshift patch 172. |
| <strong>Amazon Redshift patch 171 released. (p. 660)</strong> | A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see Amazon Redshift patch 171. |</p>
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Amazon Redshift patch 170 released.</strong> (p. 660)</td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see <a href="#">Amazon Redshift patch 170</a>.</td>
<td>July 20, 2022</td>
</tr>
<tr>
<td><strong>Amazon Redshift patch 169 released.</strong> (p. 660)</td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see <a href="#">Amazon Redshift patch 169</a>.</td>
<td>June 8, 2022</td>
</tr>
<tr>
<td><strong>Amazon Redshift patch 168 released.</strong> (p. 661)</td>
<td>A new Amazon Redshift patch is being deployed. It takes several weeks for a new version to become available in all Amazon Redshift supported AWS Regions. For more information about this version, see <a href="#">Amazon Redshift patch 168</a>.</td>
<td>April 19, 2022</td>
</tr>
<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 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 write Lambda UDFs to enable external tokenization of data.</td>
<td>October 26, 2020</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------</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>Support for the Data API for Amazon Redshift</td>
<td>Amazon Redshift can now be accessed using the built-in Data API. Documentation updates include an Amazon Redshift Data API Reference.</td>
<td>September 10, 2020</td>
</tr>
<tr>
<td>Support for Amazon Redshift console query monitoring</td>
<td>Updated the guide to describe new query monitoring graphs.</td>
<td>May 7, 2020</td>
</tr>
<tr>
<td>Support for usage limits</td>
<td>Updated the guide to describe usage limits.</td>
<td>April 23, 2020</td>
</tr>
<tr>
<td>Multi-factor authentication</td>
<td>Updated the guide to describe multi-factor authentication support.</td>
<td>April 20, 2020</td>
</tr>
<tr>
<td>Elastic resize now supports node type changes</td>
<td>Updated elastic resize description.</td>
<td>April 6, 2020</td>
</tr>
<tr>
<td>Support for ra3.4xlarge node types with managed storage</td>
<td>Updated the guide to include ra3.4xlarge node types.</td>
<td>April 2, 2020</td>
</tr>
<tr>
<td>Support for pause and resume</td>
<td>Updated the guide to describe the pause and resume cluster operations.</td>
<td>March 11, 2020</td>
</tr>
<tr>
<td>Support for Microsoft Azure AD as an identity provider</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>Support for the RA3 node type</td>
<td>Updated the guide to describe the new RA3 node type.</td>
<td>December 3, 2019</td>
</tr>
<tr>
<td>Support for the new console</td>
<td>Updated the guide to describe the new Amazon Redshift console.</td>
<td>November 11, 2019</td>
</tr>
<tr>
<td>Security information updates</td>
<td>Updates to the security information documentation.</td>
<td>June 24, 2019</td>
</tr>
<tr>
<td>Snapshot enhancements</td>
<td>Amazon Redshift now supports several enhancements to managing and scheduling snapshots.</td>
<td>April 4, 2019</td>
</tr>
<tr>
<td>Concurrency scaling</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>Feature</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</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 <a href="#">Configure a JDBC connection</a> and <a href="#">Configure and ODBC connection</a>.</td>
<td>February 4, 2019</td>
</tr>
<tr>
<td><strong>Deferred maintenance</strong></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 <a href="#">Deferring maintenance</a>.</td>
<td>November 20, 2018</td>
</tr>
<tr>
<td><strong>Advance notification</strong></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 <a href="#">Subscribing to Amazon Redshift event notifications</a>.</td>
<td>November 20, 2018</td>
</tr>
<tr>
<td><strong>Elastic resize</strong></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 <a href="#">Resizing clusters</a>.</td>
<td>November 15, 2018</td>
</tr>
<tr>
<td><strong>New ODBC drivers</strong></td>
<td>Amazon Redshift ODBC drivers have been updated to version 1.4.3.1000. For more information, see <a href="#">Configure an ODBC connection</a>.</td>
<td>November 8, 2018</td>
</tr>
<tr>
<td><strong>Cancel resize operation</strong></td>
<td>You can now cancel a resize operation while it is in progress. For more information, see <a href="#">Resize operation overview</a>.</td>
<td>November 2, 2018</td>
</tr>
</tbody>
</table>
Modify cluster to change 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 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.

October 16, 2018

Amazon Redshift spectrum supports enhanced VPC routing

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.

October 10, 2018

Query editor

You can now run SQL queries from the Amazon Redshift Management Console.

October 4, 2018

Workload execution breakdown chart

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.

July 30, 2018

Maintenance tracks

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.

July 26, 2018

The following table describes the important changes to the Amazon Redshift 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. 595)</td>
<td>May 17, 2018</td>
</tr>
<tr>
<td>HSM encryption</td>
<td>Amazon Redshift supports only AWS CloudHSM for hardware security module (HSM) key management.</td>
<td>March 6, 2018</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 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. 540).</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. 65).</td>
<td>October 17, 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. 306).</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 information, see [Using service-linked roles for Amazon Redshift](p. 497).</td>
<td>September 18, 2017</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 through a SAML 2.0-compliant identity provider. For more information, see [Using IAM authentication to generate database user credentials](p. 500).</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. 383). For more information, see [Restoring a table from a snapshot](p. 417).</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. 391).</td>
<td>April 21, 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>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. 383).</td>
<td>September 15, 2016</td>
</tr>
<tr>
<td>New connection log fields</td>
<td>The Connection log (p. 560) 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. 291).</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. 536) and Authorizing COPY, UNLOAD, CREATE EXTERNAL FUNCTION, and CREATE EXTERNAL SCHEMA operations using IAM roles (p. 542).</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. 417).</td>
<td>March 10, 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. 461).</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. 114).</td>
<td>November 20, 2015</td>
</tr>
<tr>
<td>Documentation fixes</td>
<td>Published various documentation fixes.</td>
<td>August 28, 2015</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. 349).</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. 389).</td>
<td>August 17, 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>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. 392) and Amazon Redshift parameter groups (p. 389).</td>
<td>August 3, 2015</td>
</tr>
<tr>
<td>Copy KMS encrypted clusters</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. 447).</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. 444).</td>
<td>July 28, 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. 65).</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. 433).</td>
<td>June 9, 2015</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. 291).</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>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. 199).</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. 618).</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. 488).</td>
<td>January 16, 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 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. 69) and Queries appear to hang and sometimes fail to reach the cluster (p. 349).</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. 391).</td>
<td>January 13, 2015</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. 630).</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. 442).</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. 444) and Managing clusters using the console (p. 112).</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. 651).</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. 65).</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. 293).</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. 618) and Viewing cluster metrics during load operations (p. 622).</td>
<td>October 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>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. 94) and [Deleting a cluster](p. 115).</td>
<td>August 14, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Clarified the behavior of the <strong>Allow Version Upgrade</strong> setting for clusters. For more information, see [Overview of Amazon Redshift clusters](p. 64).</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. 112).</td>
<td>July 11, 2014</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. 83).</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. 93) and [Modifying a cluster](p. 114).</td>
<td>June 2, 2014</td>
</tr>
<tr>
<td>Documentation update</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. 343).</td>
<td>May 15, 2014</td>
</tr>
<tr>
<td>New feature</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. 422).</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. 81).</td>
<td>April 28, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Clarified information about Elliptic curve Diffie—Hellman Exchange (ECDHE) support in Amazon Redshift. For more information, see [Connect using SSL](p. 303).</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. 303).</td>
<td>April 18, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Revised and reorganized the topics in the [Connecting to an Amazon Redshift cluster using SQL client tools](p. 199) 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>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 version in IAM policy examples throughout the guide.</td>
<td>April 3, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Added information about how pricing works when you resize a cluster. For more information, see [Purchasing Amazon Redshift reserved nodes](p. 433).</td>
<td>April 2, 2014</td>
</tr>
<tr>
<td>New feature</td>
<td>Added a section about a new parameter, <code>max_cursor_result_set_size</code>, 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. 389).</td>
<td>March 28, 2014</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 [Amazon Redshift clusters](p. 64).</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 [Resizing a cluster](p. 116).</td>
<td>March 21, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Reorganized and updated [What is Amazon Redshift?](p. 1) and revised [Amazon Redshift provisioned clusters overview](p. 3). 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 [Amazon Redshift clusters](p. 64).</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 [Managing clusters in a VPC](p. 121) and [Creating a cluster in a VPC](p. 123).</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 [Logging with CloudTrail](p. 567).</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 [Database audit logging](p. 559). For more information about database parameters, see [Amazon Redshift parameter groups](p. 389).</td>
<td>December 6, 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>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 <a href="#">Copying snapshots to another AWS Region</a>.</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 <a href="#">Database audit logging</a>.</td>
<td>November 11, 2013</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 <a href="#">Amazon Redshift database encryption</a>, <a href="#">Encryption for Amazon Redshift using hardware security modules</a>, and <a href="#">Encryption key rotation in Amazon Redshift</a>.</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 <a href="#">Amazon Redshift event notifications</a>.</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 <a href="#">Security in Amazon Redshift</a>.</td>
<td>August 9, 2013</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated to describe restore progress metrics. For more information, see <a href="#">Restoring a cluster from a snapshot</a>.</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 <a href="#">Sharing snapshots</a>.</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 <em>Amazon Redshift Management Guide</em> 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 <em>Amazon Redshift Management Guide</em>.</td>
<td>February 14, 2013</td>
</tr>
</tbody>
</table>