Amazon Redshift: Getting Started Guide
Copyright © 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

Getting started with Amazon Redshift ................................................................. 1
  Prerequisites ........................................................................................................ 1
    Sign up for AWS ............................................................................................... 1
    Determine firewall rules .................................................................................. 2
  Getting started with clusters and data loading .................................................. 3
    Using a sample dataset .................................................................................... 3
      Step 1: Create a sample Amazon Redshift cluster ...................................... 3
      Step 2: Try example queries using the query editor .................................... 5
  Bringing your own data to Amazon Redshift ....................................................... 6
    Step 1: Create an IAM role ............................................................................ 6
    Step 2: Create a cluster ............................................................................... 7
    Step 3: Configure inbound rules for SQL clients ........................................... 9
    Step 4: Grant access to the query editor and run queries ......................... 9
    Step 5: Load sample data ........................................................................... 11
    Step 6: Try example queries using the query editor ............................... 14
    Step 7: Reset your environment .............................................................. 15
  Getting started with common database tasks .................................................... 16
    Task 1: Create a database .......................................................................... 16
    Task 2: Create a user ................................................................................ 16
    Task 3: Create a schema ........................................................................ 17
    Task 4: Create a table ............................................................................. 18
      Insert data rows into a table .................................................................. 19
      Select data from a table .................................................................... 19
    Task 5: Load sample data ........................................................................... 19
    Task 6: Query the system tables ................................................................. 19
      View a list of table names .................................................................. 20
      View users ......................................................................................... 21
      View recent queries ........................................................................ 22
      Determine the process ID of a running query .................................... 22
    Task 7: Cancel a query ........................................................................... 22
      Cancel a query from another session ................................................ 24
      Cancel a query using the superuser queue ....................................... 24
    Task 8: Clean up your resources .............................................................. 24
  Getting started querying data lakes ................................................................. 26
  Getting started querying remote data sources .................................................. 27
  Getting started accessing data in other clusters ................................................. 28
  Getting started training ML models with Redshift data ................................. 29
  Additional resources .................................................................................... 30
  Document history ......................................................................................... 31
Getting started with Amazon Redshift

Welcome to the Amazon Redshift Getting Started Guide. Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the AWS Cloud. 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.

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

- Amazon Redshift management overview – In this topic, you can find an overview of Amazon Redshift.
- Service highlights and pricing – On this product detail page, you can find details about Amazon Redshift service highlights and pricing.
- Amazon Redshift Getting Started (this guide) – In this guide, you can find a tutorial of using Amazon Redshift to create a sample cluster and work with sample data.

In this guide, you can find tutorials that walk you through one of the following:

- Getting started with Amazon Redshift clusters and data loading (p. 3)
- Getting started with common database tasks (p. 16)
- Getting started querying your data lake (p. 26)
- Getting started querying data on remote data sources (p. 27)
- Getting started accessing data in other Amazon Redshift clusters (p. 28)
- Getting started training machine learning models with Amazon Redshift data (p. 29)

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

Prerequisites

Before you begin setting up an Amazon Redshift cluster, make sure that you complete the following prerequisites:

- Sign up for AWS (p. 1)
- Determine firewall rules (p. 2)

Sign up for AWS

If you don’t already have an AWS account, sign up for one. If you already have an account, you can skip this prerequisite and use your existing account.

2. Follow the online instructions.
Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

**Determine firewall rules**

As part of this tutorial, you specify a port when you launch your Amazon Redshift cluster. You also create an inbound ingress rule in a security group to allow access through the port to your cluster.

If your client computer is behind a firewall, make sure that you know an open port that you can use. Using this open port, you can connect to the cluster from a SQL client tool and run queries. If you don’t know an open port, work with someone who understands your network firewall rules to determine an open port in your firewall. Though Amazon Redshift uses port 5439 by default, the connection doesn’t work if that port isn’t open in your firewall. You can’t change the port number for your Amazon Redshift cluster after it’s created. Thus, make sure that you specify an open port that works in your environment during the launch process.

This prerequisite is only applicable when you bring your own data to Amazon Redshift. For more information, see [Bringing your own data to Amazon Redshift](p. 6).
Getting started with Amazon Redshift clusters and data loading

In this section, you can find two tutorials that walk you through the process of creating a sample Amazon Redshift cluster. In one, you use a sample dataset, and in the other you bring your own dataset.

Make sure that you have the prerequisites for the Amazon Redshift Getting Started Guide before getting started. For more information, see Prerequisites (p. 1).

Topics
- Using a sample dataset (p. 3)
- Bringing your own data to Amazon Redshift (p. 6)

Using a sample dataset

In this tutorial, you walk through the process to create an Amazon Redshift cluster by using a sample dataset. Amazon Redshift automatically loads the sample dataset when you are creating a new cluster and connects your new cluster to the query editor. You can immediately query the data when the cluster is created.

Before you begin setting up an Amazon Redshift cluster, make sure that you complete the Sign up for AWS (p. 1) and Determine firewall rules (p. 2).

In this tutorial, you perform the steps shown in the following diagram.

Topics
- Step 1: Create an Amazon Redshift cluster (p. 3)
- Step 2: Try example queries using the query editor (p. 5)

Important
The sample cluster that you create runs in a live environment. The on-demand rate is $0.25 per hour for using the sample cluster that is designed in this tutorial until you delete it. For more pricing information, see Amazon Redshift pricing. If you have questions or get stuck, you can contact the Amazon Redshift team by posting on our Discussion forum.

This tutorial isn't meant for production environments and doesn't discuss options in depth. After you complete the steps in this tutorial, you can use Additional resources (p. 30) to find more in-depth information. This information can help you plan, deploy, and maintain your clusters, and work with the data in your data warehouse.

Step 1: Create an Amazon Redshift cluster

When you have the prerequisites completed, you can start creating your Amazon Redshift cluster, based on a sample dataset.

To create an Amazon Redshift cluster based on a sample dataset

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. To create a cluster, do one of the following:
   - On the Amazon Redshift service page, choose Create cluster. The Create cluster page appears.
   - On the https://console.aws.amazon.com/redshift/, choose DASHBOARD, then choose Create cluster.
   - On the https://console.aws.amazon.com/redshift/, choose CLUSTERS, then choose Create cluster.

3. In the Cluster configuration section, specify a Cluster identifier. This identifier must be unique. The identifier must be from 1–63 characters using as valid characters a–z (lowercase only) and -(hyphen).
   
   Enter examplecluster for this tutorial.

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

   The console displays your selection, as shown in the example following.

   If you later choose another node type, your organization is no longer eligible for the free trial.

   After you choose your node type, do one of the following:

   - In Sample data, choose Load sample data to load the sample dataset to your Amazon Redshift cluster. Amazon Redshift loads the sample dataset Tickit to the default dev database and public schema. You can start using the query editor to query data.
   - To bring your own data to your Amazon Redshift cluster, choose Production. Then in Sample data, choose Load sample data. For information about bringing your own data, see Bringing your own data to Amazon Redshift (p. 6).

   Amazon Redshift automatically loads the sample dataset to your sample Amazon Redshift cluster.

5. In the Database configuration section, specify values for Admin user name and Admin user password. Or choose Generate password to use a password generated by Amazon Redshift.

   For this tutorial, use these values:

   - Admin user name: Enter awsuser.
   - Admin user password: Enter a value for the password.

6. Choose Create cluster.
This tutorial uses the Amazon Redshift query editor, which you can use to query data immediately as Amazon Redshift connects your new cluster to the query editor during cluster creation.

You can also choose other SQL client tools that support JDBC or ODBC drivers to work with data in your cluster. For more information, see Connecting to an Amazon Redshift cluster using SQL client tools in the Amazon Redshift Cluster Management Guide.

**Step 2: Try example queries using the query editor**

When Amazon Redshift is creating your Amazon Redshift cluster, it automatically uploads the sample dataset Tickit. Cluster creation might take a few minutes to complete. After creation completes, the cluster status becomes ACTIVE. You can view the sample Tickit tables in the query editor by choosing the dev database and public schema.

After the Amazon Redshift cluster is created, in Connect to Amazon Redshift clusters, choose Query data.

The Amazon Redshift query editor appears. Amazon Redshift establishes connection with the new cluster that is loaded with the sample dataset.

By default, Amazon Redshift shows the database created during cluster creation in Select database and the default schema named public. To view the individual data files of the sample dataset, go to the query editor and choose the dev database and public schema.

Try some example queries in the query editor, as shown following. For more information on working with the SELECT command, see SELECT in the Amazon Redshift Database Developer Guide.

```
-- Find total sales on a given calendar date.
SELECT sum(qtysold)
FROM   sales, date
WHERE  sales.dateid = date.dateid
AND    caldate = '2008-01-05';

-- Find top 10 buyers by quantity.
SELECT firstname, lastname, total_quantity
FROM   (SELECT buyerid, sum(qtysold) total_quantity
        FROM  sales
        GROUP BY buyerid
        ORDER BY total_quantity desc limit 10) Q, users
WHERE Q.buyerid = userid
ORDER BY Q.total_quantity desc;

-- Find events in the 99.9 percentile in terms of all time gross sales.
SELECT eventname, total_price
FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price desc) as percentile
      FROM (SELECT eventid, sum(pricepaid) total_price
```
Bringing your own data to Amazon Redshift

In this tutorial, you walk through the process to create an Amazon Redshift cluster by bringing your own dataset to Amazon Redshift. You can use this sample cluster to evaluate the Amazon Redshift service.

Before you begin setting up an Amazon Redshift cluster, make sure that you complete the Sign up for AWS (p. 1) and Determine firewall rules (p. 2).

In this tutorial, you perform the steps shown in the following diagram.

Topics
- Step 1: Create an IAM role (p. 6)
- Step 2: Create a sample Amazon Redshift cluster (p. 7)
- Step 3: Configure inbound rules for SQL clients (p. 9)
- Step 4: Grant access to the query editor and run queries (p. 9)
- Step 5: Load sample data from Amazon S3 (p. 11)
- Step 6: Try example queries using the query editor (p. 14)
- Step 7: Reset your environment (p. 15)

Important
The sample cluster that you create runs in a live environment. The on-demand rate is $0.25 per hour for using the sample cluster that is designed in this tutorial until you delete it. For more pricing information, go to the Amazon Redshift pricing page. If you have questions or get stuck, you can contact the Amazon Redshift team by posting on our Discussion forum.

This tutorial isn’t meant for production environments and doesn’t discuss options in depth. After you complete the steps in this tutorial, you can use Additional resources (p. 30) to find more in-depth information. This information can help you plan, deploy, and maintain your clusters, and work with the data in your data warehouse.

Step 1: Create an IAM role

For any operation that accesses data on another AWS resource, your cluster needs permission to access the resource and the data on the resource on your behalf. An example is using a COPY command to load

```sql
FROM sales
GROUP BY eventid)) Q, event E
WHERE Q.eventid = E.eventid
AND percentile = 1
ORDER BY total_price desc;
```
data from Amazon Simple Storage Service (Amazon S3). You provide those permissions by using AWS Identity and Access Management (IAM). You can do this through an IAM role that is attached to your cluster. Or you can provide the AWS access key for an IAM user that has the necessary permissions. For more information about credentials and access permissions, see Credentials and access permissions.

To best protect your sensitive data and safeguard your AWS access credentials, we recommend creating an IAM role and attaching it to your cluster. For more information about providing access permissions, see Permissions to access other AWS resources.

In this step, you create a new IAM role that enables Amazon Redshift to load data from Amazon S3 buckets. An IAM role is an IAM identity that you can create in your account that has specific permissions. In the next step, you attach the role to your cluster.

**To create an IAM role for Amazon Redshift**

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, choose Roles.
3. Choose Create role.
4. In the AWS Service group, choose Redshift.
5. Under Select your use case, choose Redshift - Customizable, then choose Next: Permissions.
6. On the Attach permissions policies page, choose AmazonS3ReadOnlyAccess. You can leave the default setting for Set permissions boundary. Then choose Next: Tags.
8. For Role name, enter a name for your role. For this tutorial, enter myRedshiftRole.
9. Review the information, and then choose Create Role.
10. Choose the role name of the role that you just created.
11. Copy the Role ARN value to your clipboard—this value is the Amazon Resource Name (ARN) for the role that you just created. You use that value when you use the COPY command to load data in Step 5: Load sample data from Amazon S3 (p. 11).

Now that you have created the new role, your next step is to attach it to your cluster. You can attach the role when you launch a new cluster or you can attach it to an existing cluster. In the next step, you attach the role to a new cluster.

**Step 2: Create a sample Amazon Redshift cluster**

Now that you have the prerequisites completed, you can launch your Amazon Redshift cluster.

The cluster that you are about to create is live (and not running in a sandbox). You incur the standard Amazon Redshift usage fees for the cluster until you delete it. If you complete the tutorial described here in one sitting and delete the cluster when you are finished, the total charges are minimal.

**To create an Amazon Redshift cluster**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
   
   **Important**

   If you use IAM user credentials, ensure that you have the necessary permissions to perform the cluster operations. For more information, see Controlling access to IAM users in the Amazon Redshift Cluster Management Guide.

2. At upper right, choose the AWS Region where you want to create the cluster.
3. On the navigation menu, choose **CLUSTERS**, then choose **Create cluster**. The **Create cluster** page appears.

4. In the **Cluster configuration** section, specify values for **Cluster identifier**, **Node type**, and **Nodes**:
   - **Cluster identifier**: Enter `examplecluster` for this tutorial. This identifier must be unique. The identifier must be from 1–63 characters using as valid characters a–z (lowercase only) and - (hyphen).
   - Choose one of the following methods to size your cluster:
     - **Note**
       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 Cluster Management Guide**.
     - If your AWS Region supports RA3 node types, choose either **Production** or **Free trial** to answer the question **What are you planning to use this cluster for?**
       - When you choose **Production**, do one of the following:
         - If your organization is eligible, you might be able to create a cluster under the Amazon Redshift free trial program. For information about creating clusters using the free trial program, see Using a sample dataset (p. 3). To do this, choose **Free trial** to create a configuration with the dc2.large node type. For more information about choosing a free trial, see Amazon Redshift free trial.
         - If you don't know how large to size your cluster, choose **Help me choose**. Doing this starts a sizing calculator that asks you questions about the size and query characteristics of the data that you plan to store in your data warehouse.
         - If you know the required size of your cluster (that is, the node type and number of nodes), choose **I'll choose**. Then choose the **Node type** and number of **Nodes** to size your cluster for the proof of concept.
   - Choose **Node type**: `dc2.large` with **Nodes**: 2 for this tutorial.
   - If you have chosen **Production** for your cluster, then do one of the following:
     - To use the sample dataset Amazon Redshift provides, in **Sample data**, choose **Load sample data**. Amazon Redshift loads the sample dataset Tickit to the default `dev` database and `public` schema.
     - To bring your own data to Amazon Redshift, continue with the rest of the tutorial.

5. In the **Database configuration** section, specify values for **Database name (optional)**, **Database port (optional)**, **Admin user name**, and **Admin user password**. Or choose **Generate password** to use a password generated by Amazon Redshift.

   In this tutorial, use these values:
   - **Database name (optional)**: Enter `dev`.
   - **Database port (optional)**: Enter `5439`.
   - **Admin user name**: Enter `awsuser`.
   - **Admin user password**: Enter a value for the password.

6. (Optional) In the **Cluster permissions** section, for **Available IAM roles** choose the IAM role that you previously created, `myRedshiftRole`. Then choose **Associate IAM role**.

   If you created your cluster with the **Load sample data** option, associate an IAM role to the cluster.

7. (Optional) In the **Additional configurations** section, turn off **Use defaults** to modify **Network and security**, **Database configuration**, **Maintenance**, **Monitoring**, and **Backup** settings.

8. Choose **Create cluster**.
Step 3: Configure inbound rules for SQL clients

Later in this tutorial, you access your cluster from within a virtual private cloud (VPC) based on the Amazon VPC service. However, if you use an SQL client from outside your firewall to access the cluster, make sure that you grant inbound access.

You can skip this step if you plan to access the cluster with the Amazon Redshift query editor from within your VPC.

To check your firewall and grant inbound access to your cluster

1. Check your firewall rules if your cluster needs to be accessed from outside a firewall. For example, your client might be an Amazon Elastic Compute Cloud (Amazon EC2) instance or an external computer.
2. To access from an Amazon EC2 external client, add an ingress rule to the security group attached to your cluster that allows inbound traffic. You add Amazon EC2 security group rules in the Amazon EC2 console. For example, a CIDR/IP of 192.0.2.0/24 allows clients in that IP address range to connect to your cluster. Find out the correct CIDR/IP for your environment.

Step 4: Grant access to the query editor and run queries

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.
- Connect to your cluster through an SQL client tool, such as SQL Workbench/J. For more information about using SQL Workbench/J, see Connect to your cluster by using SQL Workbench/J in the Amazon Redshift Cluster Management Guide.

Using the Amazon Redshift query editor is the easiest way to run queries on databases hosted by your Amazon Redshift cluster. After creating your cluster, you can immediately run queries using the Amazon Redshift console. For details about considerations when using the Amazon Redshift query editor, see Querying a database using the query editor in the Amazon Redshift Cluster Management Guide.

Topics

- Grant access to the query editor (p. 9)
- Use the query editor (p. 10)

Grant access to the query editor

To use the Amazon Redshift query editor, you need permission. To set access, attach the AmazonRedshiftQueryEditor and AmazonRedshiftReadOnlyAccess IAM policies to the IAM user that you use to access your cluster.

If you have already created an IAM user to access Amazon Redshift, you can attach the AmazonRedshiftQueryEditor and AmazonRedshiftReadOnlyAccess policies to that user. If you haven’t created an IAM user yet, create one and attach the policies to the IAM user.
To attach the required IAM policies for the query editor

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. Choose Users.
3. Choose the user that needs access to the query editor.
4. Choose Add permissions.
5. Choose Attach existing policies directly.
6. For Policy names, choose AmazonRedshiftQueryEditor and AmazonRedshiftReadOnlyAccess.
7. Choose Next: Review.
8. Choose Add permissions.

Use the query editor

In the following example, you use the query editor to perform the following tasks:

- Run SQL commands.
- View details about how queries run.
- Save a query.
- Download a query result set.

To use the query editor

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 EDITOR, then connect to a database in your cluster.

   On the Connect to database page, there are two ways to authenticate, namely, Temporary credentials and AWS Secrets Manager. For this tutorial, choose Create a new connection and Temporary credentials, then enter the values that you used when you created the cluster, as follows:

   - Cluster: Choose examplecluster.
   - Database name: Enter dev.
   - Database user: Enter awsuser.

   Then choose Connect.
3. For Schema, choose public to create a new table based on that schema.
4. Enter the following in the query editor window, and choose Run to create a new table.

   ```
   create table shoes(
       shoetype varchar (10),
       color varchar(10));
   ```

5. Choose Clear.
6. Enter the following command in the query editor window, and choose Run to add rows to the table.

   ```
   insert into shoes values
   ('loafers', 'brown'),
   ('sandals', 'black');
   ```

7. Choose Clear.
8. Enter the following command in the query editor window, and choose **Run** to query the new table.

```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>
<tr>
<td>loafers</td>
<td>brown</td>
</tr>
</tbody>
</table>

9. Choose **Execution** to view the run details.
10. Choose **Export** to download the query results as a file. The supported file formats are CSV, TXT, and HTML.

### Step 5: Load sample data from Amazon S3

At this point, you have a database called *dev* and you are connected to it. Next, you create some tables in the database, upload data to the tables, and try a query. For your convenience, the sample data that you load is available in an Amazon S3 bucket.

**Note**

If you're using a SQL client tool, ensure that your SQL client is connected to the cluster.

After you complete this step, you can do the following:

- Try example queries at Step 6: Try example queries using the query editor (p. 14).
- Reset your environment at Step 7: Reset your environment (p. 15).
- Find more information about Amazon Redshift at Additional resources (p. 30).

**Note**

To try querying data in the query editor without loading your own data, choose **Load sample data** in Sample data. If you do, Amazon Redshift loads its sample dataset to your Amazon Redshift cluster automatically during cluster creation.

### To load sample data from Amazon S3

1. **Create tables.**

    If you are using the Amazon Redshift query editor, individually copy and run the following create table statements to create tables in the dev database. For more information about the syntax, see CREATE TABLE in the Amazon Redshift Database Developer Guide.

    ```sql
create table users(
    userid integer not null distkey sortkey,
    username char(8),
    firstname varchar(30),
    lastname varchar(30),
    city varchar(30),
    state char(2),
    email varchar(100),
    phone char(14),
    likesports boolean,
    liketheatre boolean,
    likeconcerts boolean,
    likejazz boolean,
    . . .
```
likeclassical boolean,
likeopera boolean,
likerock boolean,
likevegas boolean,
likebroadway boolean,
likemusicals boolean);

create table venue(
    venueid smallint not null distkey sortkey,
    venuename varchar(100),
    venuocity varchar(30),
    venuestate char(2),
    venueats integer);

create table category(
    catid smallint not null distkey sortkey,
    categroup varchar(10),
    catname varchar(10),
    catedesc varchar(50));

create table date(
    dateid smallint not null distkey sortkey,
    caldate date not null,
    day character(3) not null,
    week smallint not null,
    month character(5) not null,
    qtr character(5) not null,
    year smallint not null,
    holiday boolean default('N'));

create table event(
    eventid integer not null distkey,
    venueid smallint not null,
    catid smallint not null,
    dateid smallint not null sortkey,
    eventname varchar(200),
    startime timestamp);

create table listing(
    listid integer not null distkey,
    sellerid integer not null,
    eventid integer not null,
    dateid smallint not null sortkey,
    numtickets smallint not null,
    priceperticket decimal(8,2),
    totalprice decimal(8,2),
    listtime timestamp);

create table sales(
    salesid integer not null,
    listid integer not null distkey,
    sellerid integer not null,
    buyerid integer not null,
    eventid integer not null,
    dateid smallint not null sortkey,
    qtsold smallint not null,
    pricepaid decimal(8,2),
Load sample data from Amazon S3 by using the COPY command.

**Note**
We recommend using the COPY command to load large datasets into Amazon Redshift from Amazon S3 or Amazon DynamoDB. For more information about COPY syntax, see COPY in the Amazon Redshift Database Developer Guide.

a. Download the file `tickitdb.zip`, which contains individual sample data files.

b. Unzip and load the individual files to a `tickit` folder in your Amazon S3 bucket in your AWS Region.

c. Edit the COPY commands in this tutorial to point to the files in your Amazon S3 bucket. For information about how to manage files with Amazon S3, see Creating and configuring an S3 Bucket in the Amazon Simple Storage Service Console User Guide.

d. Provide authentication for your cluster to access Amazon S3 on your behalf to load the sample data. You can provide either role-based authentication or key-based authentication. We recommend using role-based authentication. You can use the `myRedshiftRole` IAM role that you created at Step 1: Create an IAM role (p. 6) to enable Amazon Redshift to load data from Amazon S3 buckets. For more information about both types of authentication, see CREDENTIALS in the Amazon Redshift Database Developer Guide.

For this step, you provide authentication by referencing the IAM role that you created and then attached to your cluster in previous steps.

**Note**
If you don’t have proper permissions to access Amazon S3, you receive the following error message when running the COPY command: S3ServiceException: Access Denied. For information about IAM permissions for the COPY command, see COPY in the Amazon Redshift Database Developer Guide.

The COPY commands include a placeholder for the Amazon Resource Name (ARN) for the IAM role, your bucket name, and an AWS Region, as shown in the following example.

```sql
copy users from 's3://<myBucket>/tickit/allusers_pipe.txt'
credentials 'aws_iam_role=<iam-role-arn>'
delimiter '|' region '<aws-region>';
```

To authorize access using an IAM role, replace `<iam-role-arn>` in the CREDENTIALS parameter string with the role ARN for the IAM role that you created in Step 1: Create an IAM role (p. 6).

Your COPY command looks similar to the following example.

```sql
copy users from 's3://<myBucket>/tickit/allusers_pipe.txt'
credentials 'aws_iam_role=arn:aws:iam::123456789012:role/myRedshiftRole'
delimiter '|' region '<aws-region>';
```

To load the sample data, replace `<myBucket>`, `<iam-role-arn>`, and `<aws-region>` in the following COPY commands with your values. If you are using the Amazon Redshift query editor, individually run the following commands.

```sql
copy users from 's3://<myBucket>/tickit/allusers_pipe.txt'
credentials 'aws_iam_role=<iam-role-arn>'
delimiter '|' region '<aws-region>';
```
Step 6: Try example queries using the query editor

Now, try some example queries, as shown following. For more information on working with the SELECT command, see SELECT in the Amazon Redshift Developer Guide.

```sql
-- Get definition for the sales table.
SELECT *
FROM pg_table_def
WHERE tablename = 'sales';

-- Find total sales on a given calendar date.
SELECT sum(qtysold)
FROM sales, date
WHERE sales.dateid = date.dateid
AND caldate = '2008-01-05';

-- Find top 10 buyers by quantity.
SELECT firstname, lastname, total_quantity
FROM (SELECT buyerid, sum(qtysold) total_quantity
      FROM sales
      GROUP BY buyerid
      ORDER BY total_quantity desc limit 10) Q, users
WHERE Q.buyerid = userid
ORDER BY Q.total_quantity desc;

-- Find events in the 99.9 percentile in terms of all time gross sales.
SELECT eventname, total_price
FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price desc) as percentile
      FROM sales)
ORDER BY percentile desc;
```
GROUP BY eventid)) Q, event E
WHERE Q.eventid = E.eventid
AND percentile = 1
ORDER BY total_price desc;

You have successfully created an Amazon Redshift cluster and queried data from your own dataset using the Amazon Redshift query editor.

**Step 7: Reset your environment**

When you have completed this tutorial, we suggest that you reset your environment to the previous state by deleting your sample cluster. You continue to incur charges for the Amazon Redshift service until you delete the cluster.

However, you might want to keep the sample cluster running if you intend to try tasks in other Amazon Redshift guides.

**To delete a cluster**

1. Sign in to the AWS Management Console and open the Amazon Redshift console at https://console.aws.amazon.com/redshift/.
2. On the navigation menu, choose **CLUSTERS** to display your list of clusters.
3. Choose the **examplecluster** cluster. For **Actions**, choose **Delete**. The **Delete cluster** page appears.
4. Confirm the cluster to be deleted, then choose **Delete cluster**.

On the cluster list page, the cluster status is updated as the cluster is deleted.

After you complete this tutorial, you can find more information about Amazon Redshift and next steps in **Additional resources (p. 30)**.
Getting started with common database tasks

Following, you can find a description and walkthrough for common tasks so you can begin using Amazon Redshift databases.

After you connect to the initial cluster dev database, you can create a new database. Independent of whether you choose to use the sample dataset or bring your own data to Amazon Redshift while creating a cluster, Amazon Redshift creates the dev database.

The examples in this section assume the following:

- You have signed up for the Amazon Redshift data warehouse service. For more information, see Prerequisites (p. 1).
- You have created an Amazon Redshift cluster. For more information, see Getting started with Amazon Redshift clusters and data loading (p. 3).
- You have established a connection to the cluster from your SQL client tool, such as the Amazon Redshift console query editor. For more information, see Step 4: Grant access to the query editor and run queries (p. 9).

**Important**

The cluster that you deployed for this exercise runs in a live environment. As long as it's running, it accrues charges to your AWS account. For pricing information, see the Amazon Redshift pricing page.

To avoid unnecessary charges, delete your cluster when you are done with it. The final step of the exercise explains how to do so.

**Task 1: Create a database**

After you verify that your cluster is up and running, you can create your own first database. This database is where you actually create tables, load data, and run queries. A single cluster can host multiple databases. For example, you can have a SALESDB database and an ORDERSDB database on the same cluster.

For example, to create a database named SALESDB, run the following command in your SQL client tool.

```
CREATE DATABASE SALESDB;
```

For this exercise, accept the defaults. For information about more command options, see CREATE DATABASE in the Amazon Redshift Database Developer Guide.

After you have created the SALESDB database, you can connect to the new database from your SQL client. Use the same connection parameters as you used for your current connection, but change the database name to SALESDB.

**Task 2: Create a user**

By default, only the admin user that you created when you launched the cluster has access to the initial database in the cluster. To grant other users access, create one or more accounts. Database user accounts are global across all the databases in a cluster, and not per individual databases.
Use the CREATE USER command to create a new user. When you create a new user, you specify the name of the new user and a password. We recommend that you specify a password for the user. It must have 8–64 characters, and it must include at least one uppercase letter, one lowercase letter, and one numeral.

For example, to create a user named **GUEST** with password **ABCd4321**, run the following command.

```
CREATE USER GUEST PASSWORD 'ABCd4321';
```

To connect to the **SALESDB** database as the **GUEST** user, use the same password when you created the user, such as **ABCd4321**.

For information about other command options, see **CREATE USER** in the Amazon Redshift Database Developer Guide.

### Task 3: Create a schema

After you create a new database, you can create a new schema in the current database. A **schema** is a namespace that contains named database objects such as tables, views, and user-defined functions (UDFs). A database can contain one or multiple schemas, and each schema belongs to only one database. Two schemas can have different objects that share the same name.

You can create multiple schemas in the same database to organize data the way that you want or to group your data functionally. For example, you can create a schema to store all your staging data and another schema to store all the reporting tables. You can also create different schemas to store data relevant to different business groups that are in the same database. Each schema can store different database objects, such as tables, views, and user-defined functions (UDFs). In addition, you can create schemas with the AUTHORIZATION clause. This clause gives ownership to a specified user or sets a quota on the maximum amount of disk space that the specified schema can use.

PostgreSQL automatically creates a schema called **public** for every new database. When you don't specify the schema name while creating database objects, the objects go into the **public** schema.

To access an object in a schema, qualify the object by using the **schema_name.table_name** notation. The qualified name of the schema consists of the schema name and table name separated by a dot. For example, you might have a **sales** schema that has a **price** table and an **inventory** schema that also has a **price** table. When you refer to the **price** table, you must qualify it as **sales.price** or **inventory.price**.

The following example creates a schema named **SALES** for the user **GUEST**.

```
CREATE SCHEMA SALES AUTHORIZATION GUEST;
```

For information about more command options, see **CREATE SCHEMA** in the Amazon Redshift Database Developer Guide.

To view the list of schemas in your database, run the following command.

```
select * from pg_namespace;
```

The output should look similar to the following.

```
nspname | nspowner | nsacl
---------|---------|-----------------------------
```
For more information on how to query catalog tables, see Querying the catalog tables in the Amazon Redshift Database Developer Guide.

Use the GRANT statement to give permissions to users for the schemas.

The following example grants privilege to the GUEST user to select data from all tables or views in the SALES schema using a SELECT statement.

```
GRANT SELECT ON ALL TABLES IN SCHEMA SALES TO GUEST;
```

The following example grants all available privileges at once to the GUEST user.

```
GRANT ALL ON SCHEMA SALES TO GUEST;
```

**Task 4: Create a table**

After you create your new database, create tables to hold your data. Specify any column information when you create the table.

In the following example, the GUEST user logs on to the SALESDB and creates a new table.

For example, to create a table named `DEMO`, run the following command.

```
CREATE TABLE Demo (
    PersonID int,
    City varchar (255)
);
```

You can also create a table using the `schema_name.object_name` notation to create the table in the SALES schema.

```
CREATE TABLE SALES.DEMO (
    PersonID int,
    City varchar (255)
);
```

To view and inspect schemas and their tables, you can use the Amazon Redshift query editor. Or you can see the list of tables in schemas using system views. For more information, see Task 6: Query the system tables (p. 19).

By default, new database objects, such as tables, are created in the default schema named `public` created during cluster creation. You can use another schema to create database objects. For more information about schemas, see Managing database security in the Amazon Redshift Database Developer Guide.
The encoding, distkey, and sortkey columns are used by Amazon Redshift for parallel processing. For more information about designing tables that incorporate these elements, see Amazon Redshift best practices for designing tables.

Insert data rows into a table

After you create a table, insert rows of data into that table.

Note
The INSERT command inserts rows into a table. For standard bulk loads, use the COPY command. For more information, see Use a COPY command to load data.

For example, to insert values into the DEMO table, run the following command.

```sql
INSERT INTO DEMO VALUES (781, 'San Jose'), (990, 'Palo Alto');
```

Select data from a table

After you create a table and populate it with data, use a SELECT statement to display the data contained in the table. The SELECT * statement returns all the column names and row values for all of the data in a table, Using SELECT is a good way to verify that recently added data was correctly inserted into the table.

To view the data that you entered in the DEMO table, run the following command.

```sql
SELECT * from SALES.DEMO;
```

The result will look like this:

<table>
<thead>
<tr>
<th>personid</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>781</td>
<td>San Jose</td>
</tr>
<tr>
<td>990</td>
<td>Palo Alto</td>
</tr>
</tbody>
</table>

(2 rows)

For more information about using the SELECT statement to query tables, see SELECT.

Task 5: Load sample data

Most of the examples in this guide use the TICKIT sample dataset. You can download the file tickitdb.zip, which contains individual sample data files.

The sample data for this tutorial is provided in Amazon S3 buckets. These buckets give read access to all authenticated AWS users, so any valid AWS credentials that permit access to Amazon S3 work.

To load the sample data for your database, first create the tables, then use the COPY command to load the tables with sample data that is stored in an Amazon S3 bucket. For steps to create tables and load sample data, see Step 5: Load sample data from Amazon S3 (p. 11).

Task 6: Query the system tables

In addition to the tables that you create, your database contains a number of system tables. These system tables contain information about your installation and about the various queries and processes
that are running on the system. You can query these system tables to collect information about your
database.

Note
The description for each table in this documentation indicates whether a table is visible to
all users or only to superusers. Log in as a superuser to query tables that are visible only to
superusers.

Amazon Redshift provides access to the following types of system tables:

- **STL tables**
  These system tables are generated from Amazon Redshift log files to provide a history of the system.
  Logging tables have an STL prefix.

- **STV tables**
  These tables are virtual system tables that contain snapshots of the current system data. Snapshot
  tables have an STV prefix.

- **System views**
  System views contain a subset of data found in several of the STL and STV system tables. Systems
  views have an SVV or SVL prefix.

- **System catalog tables**
  The system catalog tables store schema metadata, such as information about tables and columns.
  System catalog tables have a PG prefix.

To retrieve system table information about a query, you might need to specify the process ID associated
with that query. For more information, see Determine the process ID of a running query (p. 22).

### View a list of table names

To view a list of all tables in a schema, you can query the PG_TABLE_DEF system catalog table. You can
first examine the setting for `search_path`.

```
SHOW search_path;
```

The result should look similar to the following,

```
search_path
---------------
$user, public
(1 row)
```

The following example adds the `SALES` schema to the search path and shows all the tables in the `SALES`
schema.

```
set search_path to '$user', 'public', 'sales';
SHOW search_path;
search_path
---------------
"$user", public, sales
(1 row)

select * from pg_table_def where schemaname = 'sales';
```
The following example shows a list of all tables called DEMO in all schemas on the current database.

```sql
select * from pg_table_def where tablename = 'demo';
```

For more information, see PG_TABLE_DEF.

You can also use the Amazon Redshift query editor to view all the tables in a specified schema by first choosing a database that you want to connect to.

**View users**

You can query the PG_USER catalog to view a list of all users, along with the user ID (USESYSID) and user privileges.

```sql
SELECT * FROM pg_user;
```

The user name rdsdb is used internally by Amazon Redshift to perform routine administrative and maintenance tasks. You can filter your query to show only user-defined user names by adding where usesysid > 1 to your SELECT statement.

```sql
SELECT * FROM pg_user;
```
View recent queries

In the previous example, the user ID (USESYSID) for adminuser is 100. To list the five most recent queries run by adminuser, you can query the SVL_QLOG view.

The SVL_QLOG view is a friendlier subset of information from the STL_QUERY table. You can use this view to find the query ID (QUERY) or process ID (PID) for a recently run query. You can also use this view to check how long it took a query to complete. SVL_QLOG includes the first 60 characters of the query string (SUBSTRING) to help you locate a specific query. Use the LIMIT clause with your SELECT statement to limit the results to five rows.

```
SELECT query, pid, elapsed, substring from svl_qlog
WHERE userid = 100
ORDER BY starttime desc
LIMIT 4;
```

The result look something like the following.

<table>
<thead>
<tr>
<th>query</th>
<th>pid</th>
<th>elapsed</th>
<th>substring</th>
</tr>
</thead>
<tbody>
<tr>
<td>892</td>
<td>21046</td>
<td>55868</td>
<td>SELECT query, pid, elapsed, substring from svl_qlog WHERE userid = 100</td>
</tr>
<tr>
<td>620</td>
<td>17635</td>
<td>1296265</td>
<td>SELECT query, pid, elapsed, substring from svl_qlog WHERE userid = 100</td>
</tr>
<tr>
<td>610</td>
<td>17607</td>
<td>82555</td>
<td>SELECT * from DEMO;</td>
</tr>
<tr>
<td>596</td>
<td>16762</td>
<td>226372</td>
<td>INSERT INTO DEMO VALUES (100);)</td>
</tr>
</tbody>
</table>

Determine the process ID of a running query

In the previous example, you learned how to obtain the query ID and process ID (PID) for a completed query from the SVL_QLOG view.

You might need to find the PID for a query that is still running. For example, you need the PID if you need to cancel a query that is taking too long to run. You can query the STV_RECENTS system table to obtain a list of process IDs for running queries, along with the corresponding query string. If your query returns multiple PIDs, you can look at the query text to determine which PID you need.

To determine the PID of a running query, run the following SELECT statement.

```
SELECT pid, user_name, starttime, query
FROM stv_recent
WHERE status='Running';
```

Task 7: Cancel a query

If you run a query that is taking too long or is consuming excessive cluster resources, cancel the query. For example, create a list of ticket sellers that includes the seller's name and quantity of tickets sold. The following query selects data from the SALES table and USERS table and joins the two tables by matching SELLERID and USERID in the WHERE clause.

```
SELECT sellerid, firstname, lastname, sum(qtysold)
FROM sales, users
WHERE sales.sellerid = users.userid
GROUP BY sellerid, firstname, lastname
ORDER BY 4 desc;
```
The result will look something like this:

<table>
<thead>
<tr>
<th>sellerid</th>
<th>firstname</th>
<th>lastname</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>48950</td>
<td>Nayda</td>
<td>Hood</td>
<td>184</td>
</tr>
<tr>
<td>19123</td>
<td>Scott</td>
<td>Simmons</td>
<td>164</td>
</tr>
<tr>
<td>20029</td>
<td>Drew</td>
<td>Mcguire</td>
<td>164</td>
</tr>
<tr>
<td>36791</td>
<td>Emerson</td>
<td>Delacruz</td>
<td>160</td>
</tr>
<tr>
<td>13567</td>
<td>Imani</td>
<td>Adams</td>
<td>156</td>
</tr>
<tr>
<td>9697</td>
<td>Dorian</td>
<td>Ray</td>
<td>156</td>
</tr>
<tr>
<td>41579</td>
<td>Harrison</td>
<td>Durham</td>
<td>156</td>
</tr>
<tr>
<td>15591</td>
<td>Phyllis</td>
<td>Clay</td>
<td>152</td>
</tr>
<tr>
<td>3008</td>
<td>Lucas</td>
<td>Stanley</td>
<td>148</td>
</tr>
<tr>
<td>44956</td>
<td>Rachel</td>
<td>Villarreal</td>
<td>148</td>
</tr>
</tbody>
</table>

Note
This is a complex query. For this tutorial, you don't need to worry about how this query is constructed.

The previous query runs in seconds and returns 2,102 rows.

Suppose that you forget to put in the WHERE clause.

```sql
SELECT sellerid, firstname, lastname, sum(qtysold)
FROM sales, users
GROUP BY sellerid, firstname, lastname
ORDER BY 4 desc;
```

The result set includes all of the rows in the `SALES` table multiplied by all the rows in the `USERS` table (49989*3766). This is called a Cartesian join, and it isn't recommended. The result is over 188 million rows and takes a long time to run.

To cancel a running query, use the CANCEL command with the query's PID.

To find the process ID, start a new session and query the `STV_RECENTS` table, as shown in the previous step. The following example shows how you can make the results more readable. To do this, use the TRIM function to trim trailing spaces and show only the first 20 characters of the query string.

```sql
SELECT pid, trim(user_name), starttime, substring(query,1,20)
FROM stv_recents
WHERE status='Running';
```

The result looks something like the following.

<table>
<thead>
<tr>
<th>pid</th>
<th>btrim</th>
<th>starttime</th>
<th>substring</th>
</tr>
</thead>
<tbody>
<tr>
<td>610</td>
<td>adminuser</td>
<td>2013-03-28 18:39:49.355918</td>
<td>select sellerid, fir</td>
</tr>
</tbody>
</table>

To cancel the query with PID 18764, run the following command.

```sql
CANCEL 610;
```

Note
The CANCEL command doesn't stop a transaction. To stop or roll back a transaction, use the ABORT or ROLLBACK command. To cancel a query associated with a transaction, first cancel the query then stop the transaction.
If the query that you canceled is associated with a transaction, use the ABORT or ROLLBACK command to cancel the transaction and discard any changes made to the data:

```
ABORT;
```

Unless you are signed on as a superuser, you can cancel only your own queries. A superuser can cancel all queries.

### Cancel a query from another session

If your query tool doesn't support running queries concurrently, start another session to cancel the query. For example, the query editor that we use in the Amazon Redshift Getting Started doesn't support multiple concurrent queries. To start another session with the query editor, choose File, New Window and connect using the same connection parameters. Then you can find the PID and cancel the query.

### Cancel a query using the superuser queue

If your current session has too many queries running concurrently, you might not be able to run the CANCEL command until another query finishes. In that case, run the CANCEL command using a different workload management query queue.

By using workload management, you can run queries in different query queues so that you don't need to wait for another query to complete. The workload manager creates a separate queue, called the Superuser queue, that you can use for troubleshooting. To use the Superuser queue, log on a superuser and set the query group to 'superuser' using the SET command. After running your commands, reset the query group using the RESET command.

To cancel a query using the Superuser queue, run these commands.

```
SET query_group TO 'superuser';
CANCEL 610;
RESET query_group;
```

For information about managing query queues, see Implementing workload management.

### Task 8: Clean up your resources

If you deployed a cluster to complete this exercise, when you are finished with the exercise delete the cluster. Deleting the cluster stops it accruing charges to your AWS account.

To delete the cluster, follow the steps in Deleting a cluster in the Amazon Redshift Cluster Management Guide.

If you want to keep the cluster, keep the sample data for reference. Most of the examples in this guide use the tables that you create in this exercise. The size of the data won't have any significant effect on your available storage.

If you want to keep the cluster, but want to clean up the sample data, run the following command to drop the SALESDB database.

```
DROP DATABASE SALESDB;
```

If you didn't create a SALESDB database, or if you don't want to drop the database, run the following commands to drop just the tables.
DROP TABLE DEMO;
DROP TABLE users;
DROP TABLE venue;
DROP TABLE category;
DROP TABLE date;
DROP TABLE event;
DROP TABLE listing;
DROP TABLE sales;
Getting started querying your data lake

You can use Amazon Redshift Spectrum to query data in Amazon S3 files without having to load the data into Amazon Redshift tables. You can query data in many formats, including Parquet, ORC, RCFile, TextFile, SequenceFile, RegexSerde, OpenCSV, and AVRO. You create external schemas and tables to define the structure of the files in Amazon S3. Then, you use an external data catalog such as AWS Glue or your own Apache Hive metastore. Changes to either type of data catalog are immediately available to any of your Amazon Redshift clusters.

After your data is registered with an AWS Glue Data Catalog and enabled with AWS Lake Formation, you can query it by using Redshift Spectrum.

Redshift Spectrum resides on dedicated Amazon Redshift servers that are independent of your cluster. Redshift Spectrum pushes many compute-intensive tasks, such as predicate filtering and aggregation, to the Redshift Spectrum layer. Redshift Spectrum also scales intelligently to take advantage of massively parallel processing.

You can partition the external tables on one or more columns to optimize query performance through partition elimination. You can query and join the external tables with Amazon Redshift tables. You can access external tables from multiple Amazon Redshift clusters and query the Amazon S3 data from any cluster in the same AWS Region. When you update Amazon S3 data files, the data is immediately available for queries from any of your Amazon Redshift clusters.

For more information about Redshift Spectrum, including how to work with Redshift Spectrum and data lakes, see Getting started with Amazon Redshift Spectrum in Amazon Redshift Database Developer Guide.
Getting started querying data on remote data sources

You can join data from an Amazon RDS database, an Amazon Aurora database, or Amazon S3 with data in your Amazon Redshift database using a federated query. You can use Amazon Redshift to query operational data directly (without moving it), apply transformations, and insert data into your Redshift tables. Some of the computation for federated queries is distributed to the remote data sources.

To run federated queries, Amazon Redshift first makes a connection to the remote data source. Amazon Redshift then retrieves metadata about the tables in the remote data source, issues queries, and then retrieves the result rows. Amazon Redshift then distributes the result rows to Amazon Redshift compute nodes for further processing.

For information about setting up your environment for federated queries, see one of the following topics in the Amazon Redshift Database Developer Guide:

- Getting started with using federated queries to PostgreSQL
- Getting started with using federated queries to MySQL (preview)
Getting started accessing data in other Amazon Redshift clusters

Using Amazon Redshift data sharing, you can securely and easily share live data across Amazon Redshift clusters or AWS accounts for read purposes. You can have instant, granular, and high-performance access to data across Amazon Redshift clusters without your needing to manually copy or move it. Your users can see the most up-to-date and consistent information as it’s updated in Amazon Redshift clusters.

Amazon Redshift data sharing is especially useful for these use cases:

- Centralizing business-critical workloads – Use a central extract, transform, and load (ETL) cluster that shares data with multiple business intelligence (BI) or analytic clusters. This approach provides read workload isolation and chargeback for individual workloads.
- Sharing data between environments – Share data among development, test, and production environments. You can improve team agility by sharing data at different levels of granularity.

For more information about data sharing, see Getting started data sharing in the Amazon Redshift Database Developer Guide.
Getting started training machine learning models with Amazon Redshift data

Using Amazon Redshift machine learning (Amazon Redshift ML), you can train a model by providing the data to Amazon Redshift. Then Amazon Redshift ML creates models that capture patterns in the input data. You can then use these models to generate predictions for new input data without incurring additional costs. By using Amazon Redshift ML, you can train machine learning models using SQL statements and invoke them in SQL queries for prediction. You can continue to improve the accuracy of the predictions by iteratively changing parameters and improving your training data.

Amazon Redshift ML makes it easier for SQL users to create, train, and deploy machine learning models using familiar SQL commands. By using Amazon Redshift ML, you can use your data in Amazon Redshift clusters to train models with Amazon SageMaker. You can then localize the models, and predictions then can be made within an Amazon Redshift database.

For more information about Amazon Redshift ML, see Getting started with Amazon Redshift ML in the Amazon Redshift Database Developer Guide.
Additional resources

When you have completed these tutorials, we recommend that you continue to learn more about the concepts introduced in this guide by using the following Amazon Redshift resources:

- **Amazon Redshift Cluster Management Guide:** This guide builds upon this *Amazon Redshift Getting Started*. It provides in-depth information about the concepts and tasks for creating, managing, and monitoring clusters.
- **Amazon Redshift Database Developer Guide:** This guide also builds upon this *Amazon Redshift Getting Started*. It provides in-depth information for database developers about designing, building, querying, and maintaining the databases that make up your data warehouse.
- **SQL reference:** This topic describes SQL commands and function references for Amazon Redshift.
- **System tables and views:** This topic describes system tables and views for Amazon Redshift.
- **Tutorials for Amazon Redshift:** This topic shows tutorials to learn about Amazon Redshift features.
  - **Using spatial SQL functions with Amazon Redshift:** This tutorial demonstrates how to use some of the spatial SQL functions with Amazon Redshift.
  - **Loading data from Amazon S3:** This tutorial describes how to load data into your Amazon Redshift database tables from data files in an Amazon S3 bucket.
  - **Querying nested data with Amazon Redshift Spectrum:** This tutorial describes how to use Redshift Spectrum to query nested data in Parquet, ORC, JSON, and Ion file formats using external tables.
  - **Configuring manual workload management (WLM) queues:** This tutorial describes how to configure manual workload management (WLM) in Amazon Redshift.
- **Feature videos:** These videos help you learn about Amazon Redshift features.
  - To learn how to get started with Amazon Redshift, watch the following video: *Getting stated with Amazon Redshift*.
  - To learn how Amazon Redshift data sharing works, watch the following video: *Amazon Redshift data sharing workflow*.
  - To learn how Amazon Redshift Machine Learning (ML) works, watch the following video: *Amazon Redshift ML*.
  - To learn how to monitor, isolate, and optimize your queries using the query monitoring features on the Amazon Redshift console, watch the following video: *Query Monitoring with Amazon Redshift*.
- **What’s new:** This webpage lists Amazon Redshift new features and product updates.
# Document history

The following table describes the important changes for the *Amazon Redshift Getting Started Guide*.

**Latest documentation update:** June 30, 2021

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Release date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation update</td>
<td>Updated the guide to include new sections about getting started with common database tasks, querying your data lake, querying data on remote sources, sharing data, and training machine learning models with Amazon Redshift data.</td>
<td>June 30, 2021</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated the guide to describe the new sample load procedure.</td>
<td>June 4, 2021</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated the guide to remove the original Amazon Redshift console and improve step flow.</td>
<td>August 14, 2020</td>
</tr>
<tr>
<td>New console</td>
<td>Updated the guide to describe the new Amazon Redshift console.</td>
<td>November 11, 2019</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated the guide to describe the quick-launch cluster procedure.</td>
<td>August 10, 2018</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated the guide to launch clusters from the Amazon Redshift dashboard.</td>
<td>July 28, 2015</td>
</tr>
<tr>
<td>New feature</td>
<td>Updated the guide to use new node type names.</td>
<td>June 9, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated screenshots and procedure for configuring VPC security groups.</td>
<td>April 30, 2015</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Updated screenshots and procedures to match the current console.</td>
<td>November 12, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Moved loading data from Amazon S3 information into its own section and moved next steps section into the final step for better discoverability.</td>
<td>May 13, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>Removed the Welcome page and incorporated the content into the main Getting Started page.</td>
<td>March 14, 2014</td>
</tr>
<tr>
<td>Documentation update</td>
<td>This is a new release of the <em>Amazon Redshift Getting Started Guide</em> that addresses customer feedback and service updates.</td>
<td>March 14, 2014</td>
</tr>
<tr>
<td>New guide</td>
<td>This is the first release of the <em>Amazon Redshift Getting Started Guide</em>.</td>
<td>February 14, 2013</td>
</tr>
</tbody>
</table>