[*Version* 1.0.23]

Storing and Analyzing Data by Using Amazon Redshift

Lab overview and objectives

Of the members of the data science team, Mary most often works with large datasets and columnar data stores. You have decided to explore using Amazon Redshift for your next proof of concept (POC) to address these use cases for her.

You ask Mary if she has a sample dataset that you can use for your POC. She suggests that you use a large music ticket sales dataset that she has worked with before. The dataset is available from a public Amazon Simple Storage Service (Amazon S3) bucket.

Data analysts often work with large datasets. For example, some data warehouses can contain multiple *petabytes* of data, and data lakes can be even larger. Amazon Redshift is designed to handle large datasets. Unlike traditional relational database management systems, Amazon Redshift uses columnar data storage. Columnar storage improves query performance and saves on storage costs.

In this lab, you will learn how to implement a Redshift cluster as part of your analytics pipeline. You will retrieve data from a dataset that is stored in Amazon S3, extract the data to a Redshift database, and then query the data for analysis.

After completing this lab, you should be able to do the following:

- Review an AWS Identity and Access Management (IAM) role's permissions to access and configure Amazon Redshift.
- Create and configure a Redshift cluster.
- Create a security group for a Redshift cluster.
- Create a database, schemas, and tables with the Redshift cluster.
- Load data into tables in a Redshift cluster.
- Query data within a Redshift cluster by using the Amazon Redshift console.
- Query data within a Redshift cluster by using the API and AWS Command Line Interface (AWS CLI) in AWS Cloud9.
- Review an IAM policy with permissions to run queries on a Redshift cluster.
- Confirm that a data science team member can run queries on a Redshift cluster.

Duration

This lab requires **90 minutes** to complete. The lab will remain active for **180 minutes** so that you can complete all steps.

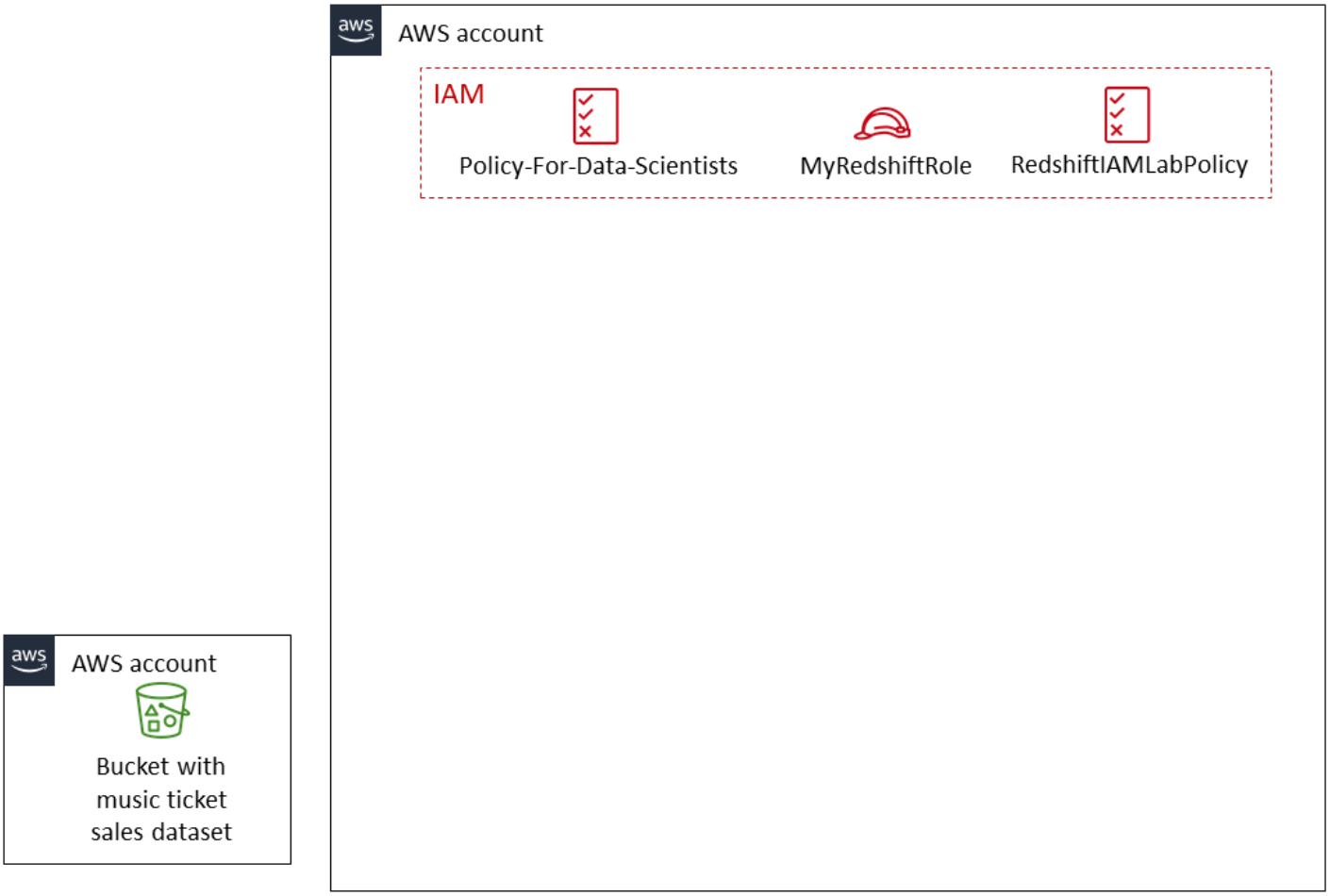
AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that are described in this lab.

Scenario

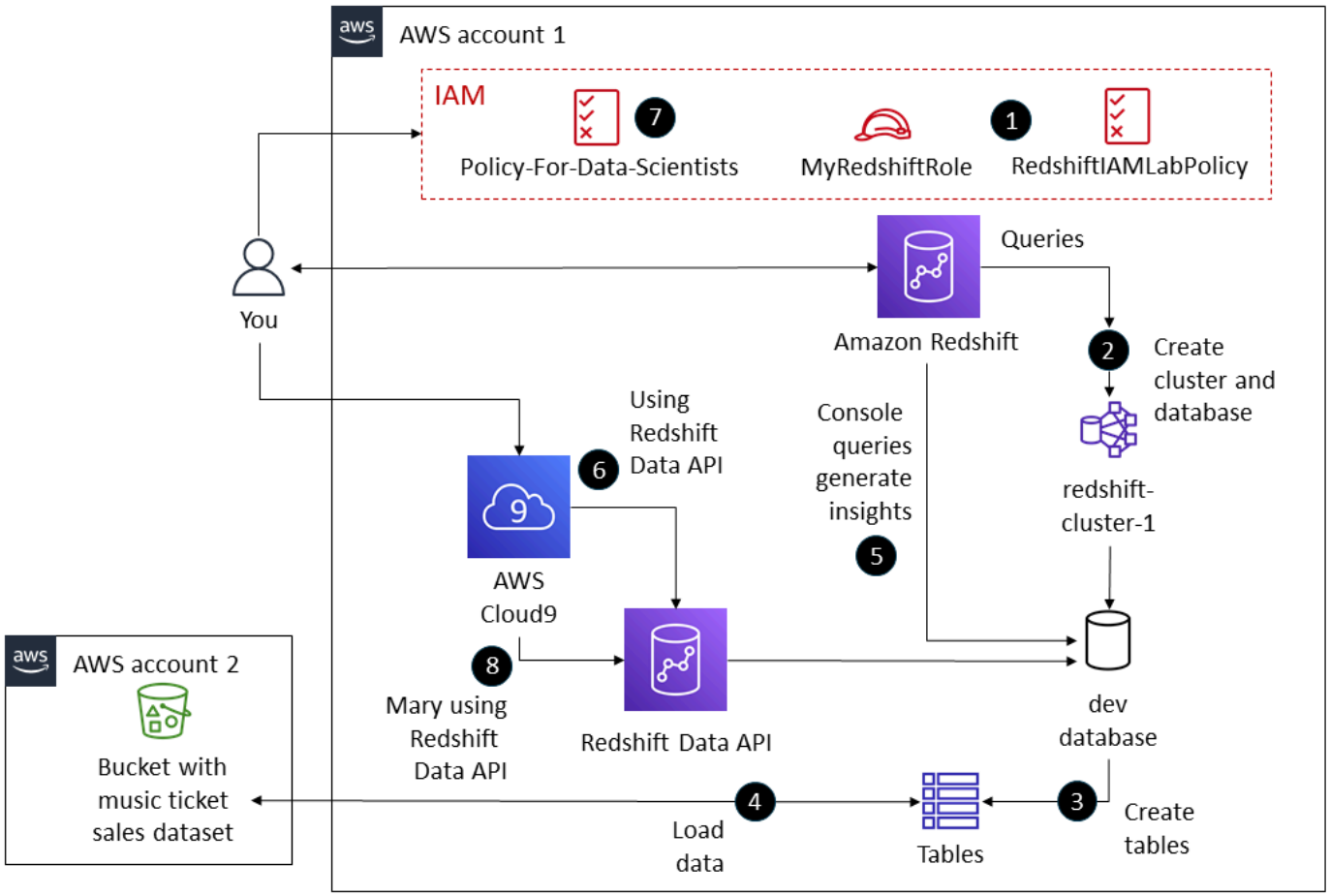
In this lab, you will build a POC to use Amazon Redshift to address the need to query large datasets. You will use SQL queries for a music ticket sales dataset. These data files are already stored in Amazon S3 in another account, and you will load them into a Redshift database.

The following diagram illustrates the environment that was created for you in AWS at the *beginning* of the lab.



Tip: To review the CloudFormation template that built this environment, after logging in to the AWS Management Console, navigate to the CloudFormation console. In the navigation pane, choose **Stacks**.

By the *end* of the lab, you will have created the additional architecture that is shown in the following diagram. The table after the diagram provides a detailed explanation of the architecture.



Numbered Task	Detail
1	You will review the *MyRedshiftRole* IAM role, which has the *RedshiftIAMLabPolicy*
2	You will use the Redshift console to create a Redshift cluster, called *redshift-cluste*
3	You will create tables in the Redshift *dev* database.
4	You will load the music ticket sales dataset data into the *dev* database from an S3
5	You will query the *dev* database by using the Amazon Redshift console.
6	You will also query the *dev* database by using the Redshift Data API and the AWS
7	You will review the *Policy-For-Data-Scientists* IAM policy, which is assigned to the
8	You will test the *Mary* user's ability to access Amazon Redshift with the *Policy-For* from the AWS CLI in the Cloud9 terminal.

Accessing the AWS Management Console

1. At the top of these instructions, choose **Start Lab**.
   - The lab session starts.
   - A timer displays at the top of the page and shows the time remaining in the session.

     Tip: To refresh the session length at any time, choose **Start Lab** again before the timer reaches 00:00.
   - Before you continue, wait until the circle icon to the right of the AWS link in the upper-left corner turns green.
2. To connect to the AWS Management Console, choose the **AWS** link in the upper-left corner.
   - A new browser tab opens and connects you to the console.

     Tip: If a new browser tab does not open, a banner or icon is usually at the top of your browser with the message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and then choose **Allow pop-ups**.

Task 1: Reviewing the IAM role to access and configure Amazon Redshift

To access the Amazon Redshift console, you must have the appropriate permissions. This access is controlled through an IAM policy that you are assigned when the lab environment launches.

In addition to having permissions to use Amazon Redshift, you will need to configure Amazon Redshift to access Amazon S3 on your behalf. This is accomplished with an IAM role, and the role *MyRedshiftRole* has been provided in the lab environment.

In this task, you will do the following:

- Access IAM and get the ARN for the *MyRedshiftRole* IAM role.
- Review the permissions in the *AmazonS3ReadOnlyAccess* managed IAM policy.
- Review the permissions in the *RedshiftIAMLabPolicy* IAM policy.

3. Access the IAM role and get the Amazon Resource Name (ARN).
   - In the AWS Management Console, in the search box to the right of **Services**, search for and choose **IAM** to open the IAM console.
   - In the navigation pane, choose **Roles**.
   - In the roles list, search for `MyRedshiftRole` and then choose the link for **MyRedshiftRole** when it displays.

     The **Summary** section at the top of the page displays the ARN for the role.
   - Copy the role's ARN to a text editor to use later.

On the lower part of the page, you can see which permissions policies are attached to the role. Notice that two IAM policies are attached to this role: *AmazonS3ReadOnlyAccess*, which is an AWS managed policy, and *RedshiftIAMLabPolicy*.

4. Review the policies that are associated with the role.
    - Expand and review the *AmazonS3ReadOnlyAccess* policy.

      This policy includes permissions that allow Amazon Redshift to access and read S3 buckets and the objects contained within them.
    - Now expand and review the *RedshiftIAMLabPolicy* policy.

      This policy was included so that Amazon Redshift can describe Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Virtual Private Cloud (Amazon VPC) resources. For example, the permissions in this policy allow Amazon Redshift to get details of the VPC configuration and the subnets associated with the VPC, and create and delete a network interface in the VPC.

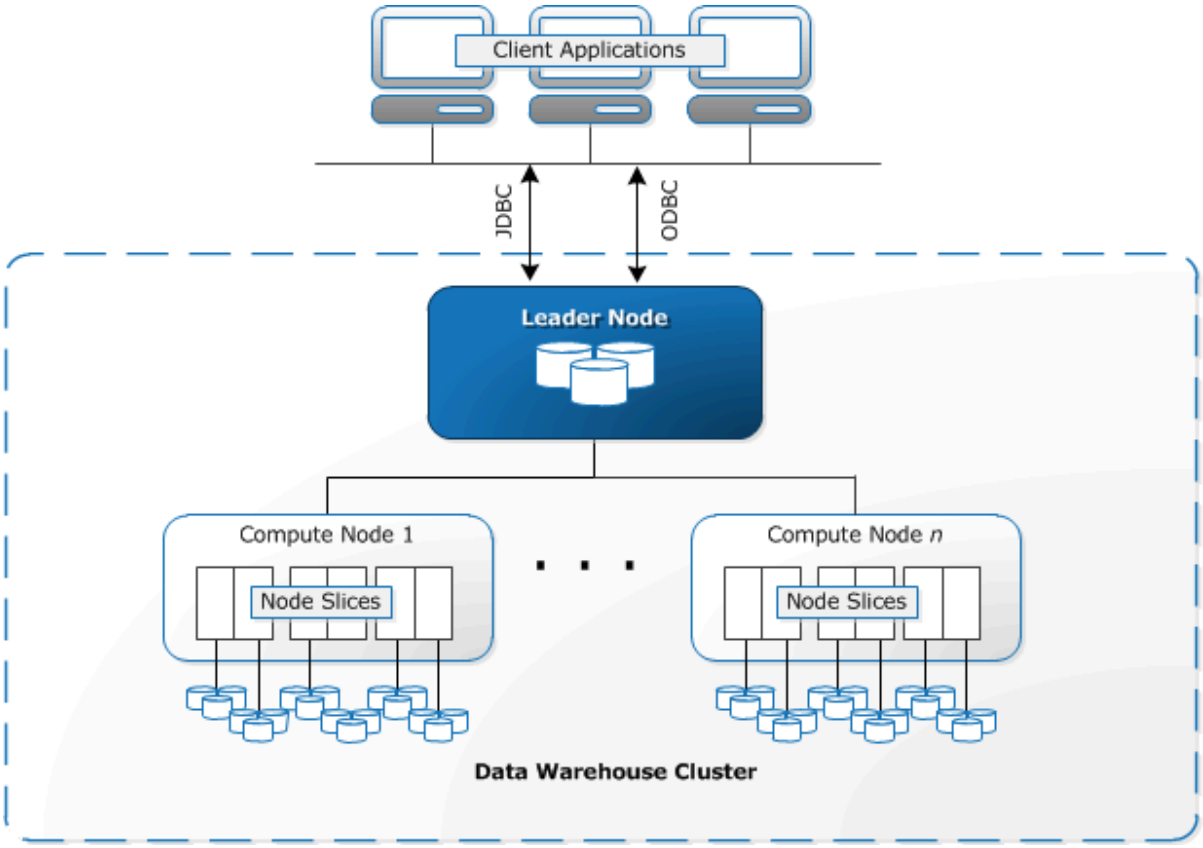
Task 1 summary

In this task, you reviewed a role that allows Amazon S3 to have read-only access to Amazon Redshift. You also reviewed a role that allows Amazon Redshift to access Amazon S3, Amazon EC2, and Amazon VPC resources. Keep these permissions in mind as you complete this lab.

Task 2: Creating and configuring a Redshift cluster

Clusters are the main infrastructure component of an Amazon Redshift data warehouse. A *cluster* is made up of one or more compute nodes. If a cluster has more than one node, then one of the nodes is the *leader node*, and the other nodes are *compute nodes*. Client applications interact with the leader node.

The following diagram illustrates the overall Amazon Redshift infrastructure:



In this task, you will do the following:

- Access the Amazon Redshift console.
- Create and configure a new Redshift cluster.

5. Access Amazon Redshift and initiate the Redshift cluster creation process.
   - In the search box to the right of **Services**, search for and choose **Amazon Redshift** to open the Amazon Redshift console.

     Note: If you see an error that states that you are not authorized, ignore the error. This is because of security restrictions in the lab environment.
   - In the navigation pane, choose **Clusters**.

     Note: If the navigation pane is collapsed, choose the menu icon in the upper-left corner of the page to open the pane.
   - Choose **Create cluster**.
6. Configure the following settings for the cluster.

   Note: Some settings are prefilled with the correct value.
   - In the **Cluster configuration** section, configure the following:
     - **Cluster identifier:** Enter `redshift-cluster-1`
     - Choose **Production**.
     - **Node type:** Choose **dc2.large**.
     - **Number of nodes:** Enter `2`

       Note: The **Configuration summary** section indicates the cost of the configuration per month and the total compressed storage that is available to the nodes.
   - In the **Database configurations** section, configure the following:

- **Admin user name:** Enter `awsuser`
- **Admin user password:** Enter `Passw0rd1`

  Note: The password includes a zero, not a capital letter O.
- To ensure that the password is correct, select **Show password**.

- In the **Cluster permissions** section, configure the following:
  - Choose **Manage IAM roles** > **Associate IAM roles**.
  - In the pop-up window, select **MyRedshiftRole**.
  - Choose **Associate IAM roles**.
  - On the cluster creation page, in the **Associated IAM roles** section, select **MyRedshiftRole** and verify that **(1/1)** displays to the right of the **Associated IAM roles** heading.
- At the bottom of the page, choose **Create cluster**.

  Wait a few minutes for the Amazon Redshift to create the cluster.
- At the bottom of the page, in the **Clusters** section, choose the link for the **redshift-cluster-1** cluster.

  Note: Ignore any messages about changing the size of the instance to improve performance or using query editor v2.

Now you will configure the VPC that hosts your Redshift cluster so that it allows traffic through port *5439* (the default port for Amazon Redshift). To do this, you will configure a security group for your VPC and add the Redshift cluster to this security group.

Note: Amazon Redshift is deployed in a three-tier architecture. The cluster is placed within a private subnet of your VPC with web application servers in the public subnet. Users access the web application through a NAT gateway. The Redshift cluster is not available to the public internet, but port 5439 is configured within the security group to allow the application servers to access the Redshift database. You could add a bastion host to the public subnet to manage the cluster through SSH.

7. Create the security group and configure access to the Redshift database.
   - In the search box to the right of **Services**, search for and choose **EC2** to open the Amazon EC2 console.
   - In the navigation pane, under **Network & Security**, choose **Security Groups**.
   - Choose **Create security group**, and configure the following:
     - **Basic details** section:
       - **Security group name:** Enter `Redshift security group`
       - **Description:** Enter `Security group for my Redshift cluster`
     - **Inbound rules** section:
       - Choose **Add rule**.
       - **Type:** Choose **Redshift**. Note that the protocol and port range automatically update to TCP and 5439.
       - **Source:** Choose **Anywhere-IPv4**. Note that **0.0.0.0/0** is automatically added.
       - **Description:** Enter `Redshift inbound rule`
   - At the bottom of the page, choose **Create security group**.

You now need to add the cluster to your security group.

8. Configure your Redshift cluster.
    - Navigate to the Amazon Redshift console.
    - In the navigation pane, choose **Clusters**.

      Important: Before proceeding to the next step, check that the **Status** is *Available* for the *redshift-cluster-1* cluster. It might take up to 5 minutes for the cluster to be available after creation. To refresh the status, choose the refresh icon in the upper-right corner of this section.
    - When the **Status** is *Available* for the cluster that you created, choose the link for the **redshift-cluster-1** cluster.
    - Choose the **Properties** tab.
    - In the **Network and security settings** section, choose **Edit**.
    - For **VPC security groups**, select **Redshift security group** AND clear **default**.
    - Choose **Save changes**.

      Notice that the **Status** of the cluster changes to *Modifying*.

      Important: Wait about 5 minutes for the **Status** to change to *Available* before proceeding to the next step. To refresh the status, refresh the webpage using your browser's refresh function.

Task 2 summary

In this task, you created a Redshift cluster and associated the *MyRedshiftRole* IAM role to it. This role allows Amazon Redshift to access Amazon S3 resources to load a dataset. You also created a security group for the cluster so that other resources, such as Amazon EC2 instances, can access it through port 5439. You then added the cluster to the VPC by associating it with the security group that you created.

Task 3: Creating tables in a database

Now that you have created a Redshift cluster for your POC, you can load data into it directly from Amazon S3.

Note: Amazon Redshift can store data from a variety of formats, including the following:

- Text
- Apache Avro
- JSON
- Apache Parquet
- Optimized Row Columnar (ORC)

Mary advises you that the music dataset is composed of simple text files. Some of the files use the pipe (|) character as a delimiter, and other files use tab (\t) as a delimiter. To load this data, you will first need to create a few tables (*users*, *date*, and *sales*) with the CREATE TABLE command. Then, you will copy data into each table with the COPY command.

In this task, you will do the following:

- Connect to the Redshift cluster.
- Create the *users*, *date*, and *sales* tables in the *dev* database.
- Load data into these tables from text files that are stored in Amazon S3.

9. Connect to the Redshift cluster and access the query editor.
    - In the navigation pane, choose **Clusters**.
    - Select the **redshift-cluster-1** cluster, and choose **Query data** > **Query in query editor**. **Note:** You will see two options here, Query in query editor and Query in query editor v2. In this lab you will use Query in query editor, which uses version 1 of the Redshift query editor.
    - Choose **Connect to database** and configure the following:
        - **Connection:** Choose **Create a new connection**.
        - **Authentication:** Choose **Temporary credentials**.
        - **Database name:** Enter `dev`
        - **Database user:** Enter `awsuser`

          Note: If you receive an error message that says the query editor is not available for the cluster that you selected, refresh the page and try to connect to the database again. The connection will eventually be established.
    - In the **Resources** pane of the query editor, for **Select schema**, choose **public**.
10. To create the *users* table, copy and paste the following text into a query tab, and choose **Run**:

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

    After the query completes, the **Query results** tab on the lower part of the page displays *Completed*.
11. To create the *date* table, copy and paste the following text into a new query tab, and choose **Run**:

```
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'));
```

12. To create the *sales* table, copy and paste the following text into a new query tab, and choose **Run**:

```
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,
qtysold smallint not null,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp);
```

Task 3 summary

In this task, you accessed the Amazon Redshift console and created the *users*, *date*, and *sales* tables in the *dev* database.

Now it is time to add data to these tables.

Task 4: Loading data from Amazon S3

To load the data from the S3 bucket, you will use the COPY command three times, once for each table in the *dev* database.

13. To load data to the *users* table, run the following query. Replace *<REDSHIFT-ROLE-ARN>* with the ARN that you copied for the *MyRedshiftRole* IAM role earlier in the lab:

```
copy users from 's3://aws-tc-largeobjects/CUR-TF-200-ACDSCI-
1/Lab4/allusers_tab.txt'
credentials 'aws_iam_role=<REDSHIFT-ROLE-ARN>'
delimiter '\t' region 'us-west-2';
```

After you replace the ARN, the query will look similar to the following:

```
copy users from 's3://aws-tc-largeobjects/CUR-TF-200-ACDSCI-1/Lab4/allusers_tab.txt'
credentials 'aws_iam_role=arn:aws:iam::123456789012:role/MyRedshiftRole'
delimiter '\t' region 'us-west-2';
```

After the query completes, the **Query results** tab on the lower part of the page displays *Completed*.

This query copied text from a tab-delimited file called *allusers_tab.txt*, which is located in an S3 bucket, into the *users* table. To accomplish this, Amazon Redshift assumed the *MyRedshiftRole* so that it had the appropriate IAM permissions to complete the action.

Tip: To see the data that was loaded into the table, in the **Resources** pane, to the right of **users**, choose the ellipsis (three dot) icon, and choose **Preview data**. The data displays on the **Table details** tab on the lower part of the page.

14. To load data to the *date* table, run the following query. Replace *<REDSHIFT-ROLE-ARN>* with the ARN that you copied for the *MyRedshiftRole* IAM role earlier in the lab:

```
copy date from 's3://aws-tc-largeobjects/CUR-TF-200-ACDSCI-1/Lab4/date2008_pipe.txt'
credentials 'aws_iam_role=<REDSHIFT-ROLE-ARN>'
delimiter '|' region 'us-west-2';
```

After you replace the ARN, the query will look similar to the following:

```
copy date from 's3://aws-tc-largeobjects/CUR-TF-200-ACDSCI-1/Lab4/date2008_pipe.txt'
credentials 'aws_iam_role=arn:aws:iam::123456789012:role/MyRedshiftRole'
delimiter '|' region 'us-west-2';
```

This query copied text from a pipe-delimited file called *date2008_pipe.txt*, which is located in an S3 bucket, into the *date* table.

15. To load data to the *sales* table, run the following query. Replace *<REDSHIFT-ROLE-ARN>* with the ARN that you copied for the *MyRedshiftRole* IAM role earlier in the lab:

```
copy sales from 's3://aws-tc-largeobjects/CUR-TF-200-ACDSCI-1/Lab4/sales_tab.txt'
credentials 'aws_iam_role=<REDSHIFT-ROLE-ARN>'
delimiter '\t' timeformat 'MM/DD/YYYY HH:MI:SS' region 'us-west-2';
```

After you replace the ARN, the query will look similar to the following:

```
copy sales from 's3://aws-tc-largeobjects/CUR-TF-200-ACDSCI-1/Lab4/sales_tab.txt'
credentials 'aws_iam_role=arn:aws:iam::123456789012:role/MyRedshiftRole'
delimiter '\t' timeformat 'MM/DD/YYYY HH:MI:SS' region 'us-west-2';
```

This query copied text from a tab-delimited file called *sales_tab.txt*, which is located in an S3 bucket, into the *sales* table. The text dataset includes time data, so the query specifies the timeformat parameter. This ensures that each timestamp in the records is formatted accordingly.

Task 4 summary

In this task, you loaded a dataset into your Redshift cluster. To accomplish this, you copied data from text files in Amazon S3 into the tables by using the Amazon Redshift query editor and the SQL COPY command. You learned how to configure a delimiter in the COPY command (as an example, you used **\t** as the delimiter for the first query) to ensure that data is loaded properly and each record matches the schema for its table.

Task 5: Querying the data

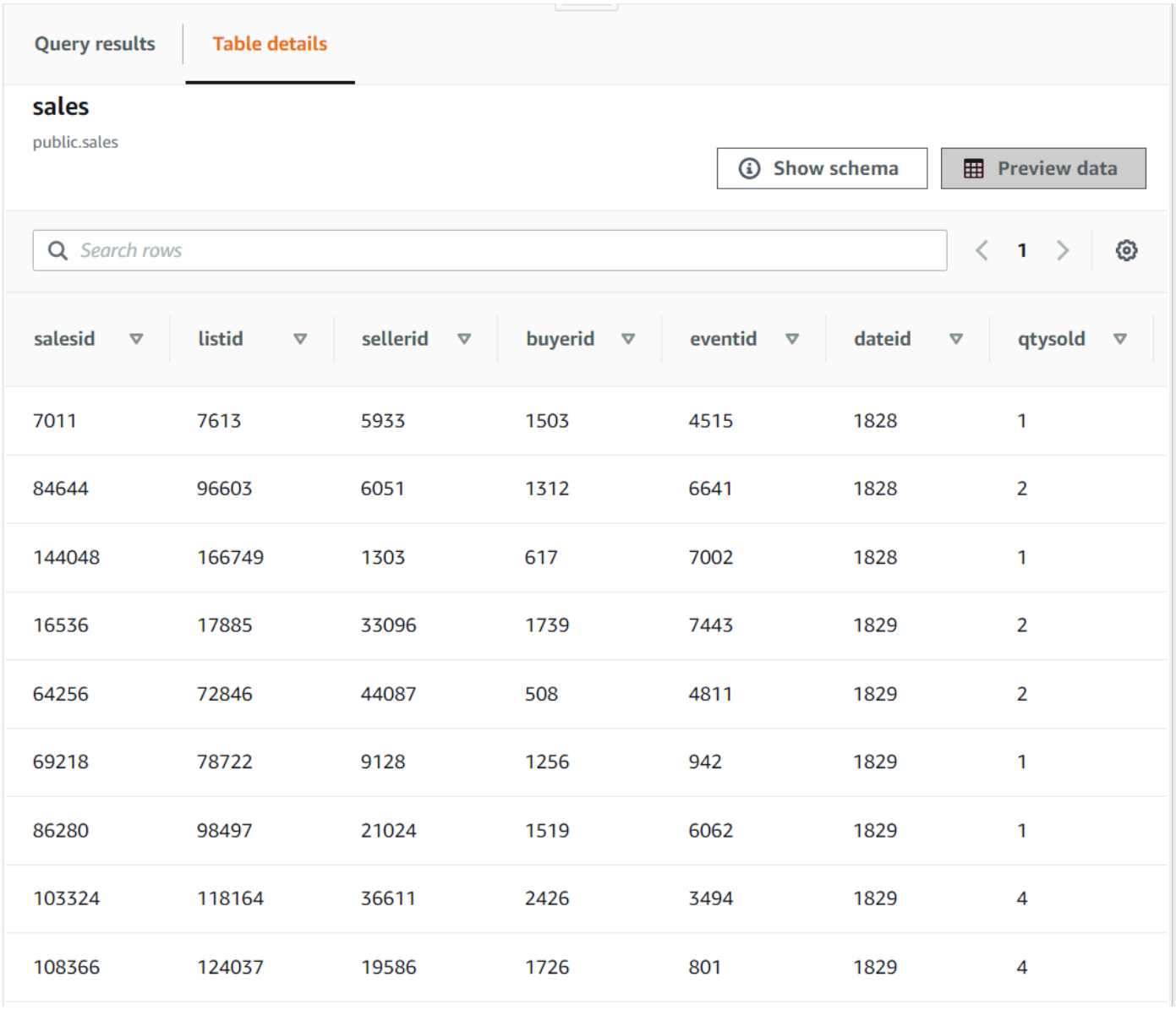
Now that the dataset is imported successfully into the Redshift cluster, you can write queries to generate the reports that Mary needs. You ask Mary what typical queries on this data look like so that you can use them in your POC.

Mary provides a query that provides total number of items that were sold on a particular date. Before you try the full query, check that a simple scan works on the data.

16. Preview the *sales* table.

    In the **Resources** pane, to the right of **sales**, choose the ellipsis (three dot) icon, and choose **Preview data**.

    The data displays on the **Table details** tab on the lower part of the page and looks similar to the following:



salesid	listid	sellerid	buyerid	eventid	dateid	qtysold
7011	7613	5933	1503	4515	1828	1
84644	96603	6051	1312	6641	1828	2
144048	166749	1303	617	7002	1828	1
16536	17885	33096	1739	7443	1829	2
64256	72846	44087	508	4811	1829	2
69218	78722	9128	1256	942	1829	1
86280	98497	21024	1519	6062	1829	1
103324	118164	36611	2426	3494	1829	4
108366	124037	19586	1726	801	1829	4

Now that everything is configured correctly, you can query the cluster by using Marys first query.

17. Run the following query in the query editor:

```
SELECT sum(qtysold)
FROM sales, date
WHERE sales.dateid = date.dateid
AND caldate = '2008-01-05';
```

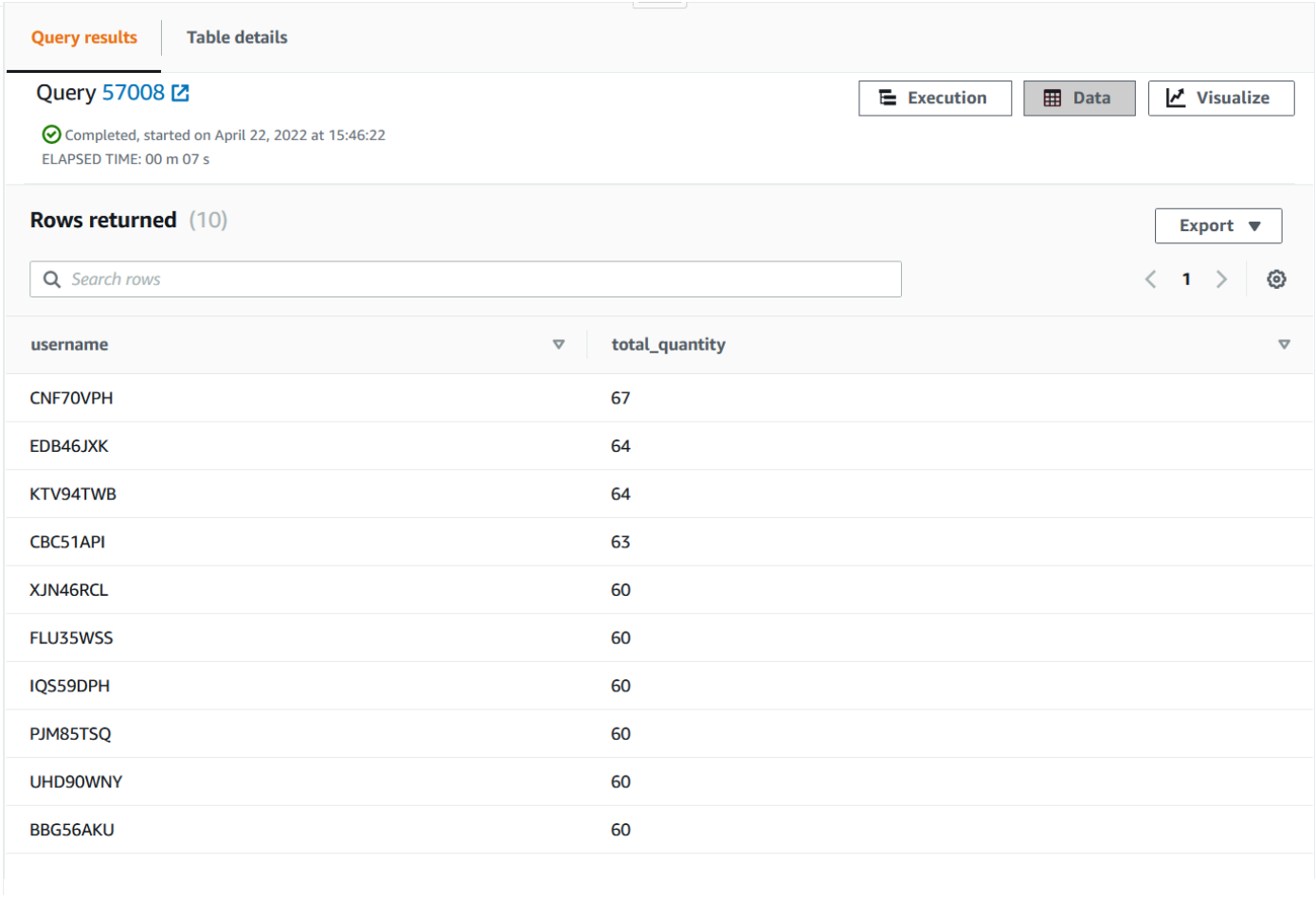
The query returns one row with the value *210*. This is the total number of items that were sold on a particular date.

Mary also sent you a query to find the top 10 buyers by quantity. Try running that query.

18. Run the following query in the query editor:

```
SELECT username, 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;
```

The query returns customer usernames and the total quantity that was sold to each customer, ordered by the **total_quantity** field and limited to 10 results, similar to the following screenshot.



You share the query results with Mary, and she is impressed with the speed. She tries the queries in AWS herself and is happy with the results.

Task 5 summary

In this task, you used SQL to perform queries on the Redshift database. Querying a Redshift database is a broad and deep topic, so you only got an introduction here by using some example queries. Consider using Amazon Redshift for your data analysis use cases because it uses SQL queries. This might help a team to adopt the service and optimize their workflow quickly.

Task 6: Running queries from the AWS CLI

In addition to using the console to run queries on data in Amazon Redshift, you can also use the Amazon Redshift API, AWS SDK libraries, and AWS CLI to perform actions.

In this task, you will use the AWS Cloud9 terminal to perform actions against your Redshift cluster.

19. Navigate to the AWS Cloud9 integrated development environment (IDE).
    - In the AWS Management Console, in the search box next to **Services**, search for and choose **Cloud9** to open the AWS Cloud9 console.

      AWS Cloud9 environments are listed.
    - For the environment named **Cloud9 Instance**, choose **Open IDE**.

      A new browser tab opens and displays the AWS Cloud9 IDE.

First, you need a database user. You will use the *awsuser* user, which you used previously. For the first query, you will select one record from the *users* table in the *dev* database.

20. Query records from the dev database.
    - Run the following command in the AWS Cloud9 terminal.

```
aws redshift-data execute-statement --region us-east-1 --db-user awsuser --cluster-identifier redshift-cluster-1 --database dev --sql "select * from users limit 1"
```

    - The output is similar to the following:

```
{
"ClusterIdentifier": "redshift-cluster-1",
"CreatedAt": 1649864498.714,
"Database": "dev",
"DbUser": "awsuser",
"Id": "ce0ec95c-596e-4cd9-86d4-5d9847dacd94"
}
```

    - Notice the information that is provided in the output:
      - **ClusterIdentifier:** Identifies the Redshift cluster that you ran the query on
      - **CreatedAt:** Gives the time that the query was created
      - **Database:** Identifies the database that you ran the query on
      - **DbUser:** Identifies the user who ran the query
      - **Id:** Gives the ID for the query

        Note: The *DbUser* will never be an IAM user. Amazon Redshift requires a database admin user, which is what you used to run the query.

Note: To get more information about the query, including the result, you can use the query ID to retrieve information by using the AWS CLI or by using the AWS SDK within your application.

- Copy the **Id** value to a text editor to use in the next command.

21. To retrieve the results of the query, run the following get-statement-result command. Replace *<QUERY-ID>* with the query ID that you copied.

```
aws redshift-data get-statement-result  --id <QUERY-ID> --region us-east-1
```

The output is similar to the following:


```
{
    "Records": [
        [
            {
                "longValue": 3
            },
            {
                "stringValue": "IFT66TXU"
            },
            {
                "stringValue": "Lars"
            },
            {
                "stringValue": "Ratliff"
            },
            {
                "stringValue": "High Point"
            },
            {
                "stringValue": "ME"
            },
            {
                "stringValue": "amet.faucibus.ut@condimentumegetvolutpat.ca"
            },
            {
                "stringValue": "(624) 767-2465"
            },
            {
                "stringValue": "true"
            },
            {
                "stringValue": "false"
            },
            {
                "isNull": true
            },
            {
                "stringValue": "false"
            },
            {
                "isNull": true
            },
            {
                "stringValue": "false"
            },
            {
                "stringValue": "true"
            },
            {
                "isNull": true
            },
            {
                "isNull": true
            },
```

```
            {
                "stringValue": "true"
            }
        ]
    ],
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "userid",
            "length": 0,
            "name": "userid",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "users",
            "typeName": "int4"
        },
        ...  # Note: You will see more information in this response.
    ],
    "TotalNumRows": 1
}
```

This is the JSON equivalent to issuing the query directly in the query editor.

Task 6 summary

In this task, you learned how to use the AWS CLI to query a Redshift database.

Using the console to submit queries is helpful to discover data and test. However, your applications or background processes will likely need programmatic access to perform actions with AWS services. Therefore, it is important to understand how queries are processed in the AWS backend, such as receiving a JSON response for a query. From there, you can do further processing; for example, displaying the result to a user in an app. You could also store the result for another workload to process later.

Task 7: Reviewing the IAM policy for Amazon Redshift access

Now that you have tested your ability to query the Redshift database, you will review an IAM policy that allows other users to perform Amazon Redshift actions in production.

22. Review the *Policy-For-Data-Scientists* IAM policy, which was created for you.
    - Navigate to the IAM console.
    - In the navigation pane, choose **Users**.

      Notice the *Mary* user that is listed. This user is part of the *DataScienceGroup* IAM group.
    - Choose the link for the **DataScienceGroup** IAM group.
    - Choose the **Permissions** tab.

      The policy that is attached to this group is displayed.
    - Choose the link for the **Policy-For-Data-Scientists** policy.

      The permissions for this policy display. Review the permissions.

      Note: The permissions provide limited access to the Redshift Data API.

      Tip: To look more closely at the policy details, choose **{ } JSON**. The JSON version of the policy shows the details for the allowed and denied actions.

Task 7 summary

In this task, you reviewed the IAM policy that is attached to the *DataScienceGroup* IAM group. The policy contains permissions for limited access to the Redshift Data API. The policy can be assigned to users who intend to use the API to query a Redshift database. As with all services in AWS, a user must have the appropriate permissions to perform any action.

Now that you have reviewed an example policy, you will be able to apply permissions accordingly for the Redshift Data API and other use cases.

Task 8: Confirming that users can run queries on the Redshift database

Now you will confirm that Mary can run queries on the Redshift database. As an administrator user, occasionally you might need to confirm that your team members have the appropriate access to AWS resources and can perform certain tasks.

In the following steps, you will determine if Mary can access the data in Redshift. To test her access, you will issue AWS CLI commands to query the database and retrieve the results. When you issue the commands, you will pass the *Mary* user credentials as bash variables with the commands. The API will then try to perform the commands as the specified user.

23. Retrieve the credentials for the *Mary* IAM user, and store these as bash variables.
    - In the search box next to **Services**, search for and choose **CloudFormation**.
    - In the navigation pane, choose **Stacks**.
    - Choose the link for the stack that created the lab environment. The stack name includes a random string of letters and numbers, and the stack should have the oldest creation time.
    - On the stack details page, choose the **Outputs** tab.

      Note: When you create a CloudFormation template, you can choose to output information about the resources that the template will create. The CloudFormation template that created the resources in your lab environment output the access key and secret access key for the *mary* user.
    - Copy the value of **MarysAccessKey** to your clipboard.
    - Return to the AWS Cloud9 terminal.
    - To create a variable for the access key, run the following command. Replace *<ACCESS-KEY>* with the value from your clipboard.

      ```
      AK=<ACCESS-KEY>
      ```

    - Return to the CloudFormation console, and copy the value of **MarysSecretAccessKey** to your clipboard.
    - Return to the AWS Cloud9 terminal.
    - To create a variable for the secret access key, run the following command. Replace *<SECRET-ACCESS-KEY>* with the value from your clipboard.

      ```
      SAK=<SECRET-ACCESS-KEY>
      ```

24. Use the execute-statement command in the Redshift Data API to query the database.
    - Run the following command in the AWS Cloud9 terminal.

```
AWS_ACCESS_KEY_ID=$AK AWS_SECRET_ACCESS_KEY=$SAK aws redshift-data
execute-statement --region us-east-1 --db-user awsuser --cluster-
identifier redshift-cluster-1 --database dev --sql "select * from users
limit 1"
```

The output is similar to the following and looks like the output that was displayed after you ran the command earlier:


```
{
    "ClusterIdentifier": "redshift-cluster-1",
    "CreatedAt": 1649864498.714,
    "Database": "dev",
    "DbUser": "awsuser",
    "Id": "ce0ec95c-596e-4cd9-86d4-5d9847dacd94"
}
```


Now, test the *Mary* user ability to get the results of the query by using the Redshift Data API get-statement-result command.

25. To retrieve the query results, copy the following command, replace the placeholder with the Id in the response above, then run the command in the terminal:

```
AWS_ACCESS_KEY_ID=$AK AWS_SECRET_ACCESS_KEY=$SAK aws redshift-data get-
statement-result  --id <QUERY-ID> --region us-east-1
```

The output is similar to the following and looks like the output that was displayed after you ran the command earlier:


```
{
    "Records": [
        [
            {
                "longValue": 3
            },
            {
                "stringValue": "IFT66TXU"
            },
            {
                "stringValue": "Lars"
            },
            {
                "stringValue": "Ratliff"
            },
            {
                "stringValue": "High Point"
            },
            {
                "stringValue": "ME"
            },
            {
                "stringValue": "amet.faucibus.ut@condimentumegetvolutpat.ca"
            },
            {
                "stringValue": "(624) 767-2465"
            },
            {
                "stringValue": "true"
            },
            {
                "stringValue": "false"
```

```
            },
            {
                "isNull": true
            },
            {
                "stringValue": "false"
            },
            {
                "isNull": true
            },
            {
                "stringValue": "false"
            },
            {
                "stringValue": "true"
            },
            {
                "isNull": true
            },
            {
                "isNull": true
            },
            {
                "stringValue": "true"
            }
        ]
    ],
    "ColumnMetadata": [
        {
            "isCaseSensitive": false,
            "isCurrency": false,
            "isSigned": true,
            "label": "userid",
            "length": 0,
            "name": "userid",
            "nullable": 0,
            "precision": 10,
            "scale": 0,
            "schemaName": "public",
            "tableName": "users",
            "typeName": "int4"
        },
        ...  # Note: You will see more information in this response.
    ],
    "TotalNumRows": 1
}
```

Task 8 summary

The results from this task confirm that Mary has the ability to run SQL statements on a Redshift database by using the AWS CLI. This will simplify her work so that she can address other challenges, such as automating tasks in Amazon Redshift and building applications that use Redshift queries by using the Amazon Redshift SDK.

Update from the team

Congratulations! You have learned how to create a Redshift database and load data into it from a dataset that is stored in Amazon S3.

Mary and the data science team are happy with the workflow that you created by using Amazon Redshift. By using this workflow, the team can use AWS services in a way that follows best practices and simplifies their workloads.

Submitting your work

26. To record your progress, choose **Submit** at the top of these instructions.
27. When prompted, choose **Yes**.

    After a couple of minutes, the grades panel appears and shows you how many points you earned for each task. If the results do not display after a couple of minutes, choose **Grades** at the top of these instructions.

Important: Some of the checks made by the submission process in this lab will only give you credit if it has been at least 5 minutes since you completed the action. If you do not receive credit the first time you submit, you may need to wait a couple minutes and the submit again to receive credit for these items.

Tip: You can submit your work multiple times. After you change your work, choose **Submit** again. Your last submission is recorded for this lab.

28. To find detailed feedback about your work, choose **Submission Report**.

Lab complete

Congratulations! You have completed the lab.

29. At the top of this page, choose **End Lab**, and then choose Yes to confirm that you want to end the lab.

    A message panel indicates that the lab is terminating.

30. To close the panel, choose **Close** in the upper-right corner.

Additional resources

For more information about the services and concepts that were covered in this lab, see the following resources:

- Columnar Storage
- Authorizing Amazon Redshift to Access Other AWS Services on Your Behalf
- Amazon Redshift System and Architecture Overview
- Amazon Redshift Clusters
- Security Group Rules for Different Use Cases
- Modular Architecture for Amazon Redshift
- Tools to Build on AWS
- Cloud9 User Guide
- Actions, Resources, and Condition Keys for Amazon Redshift Data API
- Using the Amazon Redshift Data API
- Amazon Redshift Data API Reference