Building Data Analytics Solutions Using Amazon Redshift

Lab 3 - Data Transformation and Querying in Amazon Redshift

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at *AWS Training and Certification*.

Lab overview

When building a data lake solution in Amazon Simple Storage Service (Amazon S3), you might find that you have various data types, such as structured, unstructured, and semi-structured. When that raw data needs to be analyzed, especially at a petabyte scale, you can load it into Amazon Redshift to build your analytical solutions.

You might also want to use two common data transformation techniques to control how the data is loaded and queried:

- Extract, transform, and load (ETL)
- Extract, load, and transform (ELT)

For example, you might perform an ETL operation to transform data before it is loaded into Amazon Redshift to retain only the parts of the raw data that you need. In another scenario, you might perform an ELT operation to load all of the raw data from a source, and then use Amazon Redshift to create custom transformations, such as a materialized view.

In this lab, you load stock market data (stored in Amazon S3) into Amazon Redshift. You first query the data as is. Then, you create a materialized view to transform the data (to better suit your needs) and query that view. Finally, you create a scheduled task to query the materialized view at a set interval, and you learn how to retrieve the results of a scheduled query.

OBJECTIVES

By the end of this lab, you will be able to:

- Perform an ELT operation with materialized views and stored procedures.
- Use Amazon Redshift scheduled queries.
- Query data directly from the source using Amazon Redshift data sharing.

TECHNICAL KNOWLEDGE PREREQUISITES

- Experience with Cloud platforms.
- Basic navigation of the AWS Management Console.
- Basic knowledge of Amazon Redshift.

DURATION

This lab requires approximately *45* minutes to complete.

ICON KEY

Various icons are used throughout this lab to call attention to certain aspects of the guide. The following list explains the purpose for each one:

- **Command:** A command that you must run.
- **Expected output:** A sample output that you can use to verify the output of a command or edited file.
- **Note:** A hint, tip, or important guidance.
- **Learn more:** Where to find more information.
- **Copy edit:** A time when copying a command, script, or other text to a text editor (to edit specific variables within it) might be easier than editing directly in the command line or terminal.

Start lab

1. To launch the lab, at the top of the page, choose Start lab.

Caution: You must wait for the provisioned AWS services to be ready before you can continue.

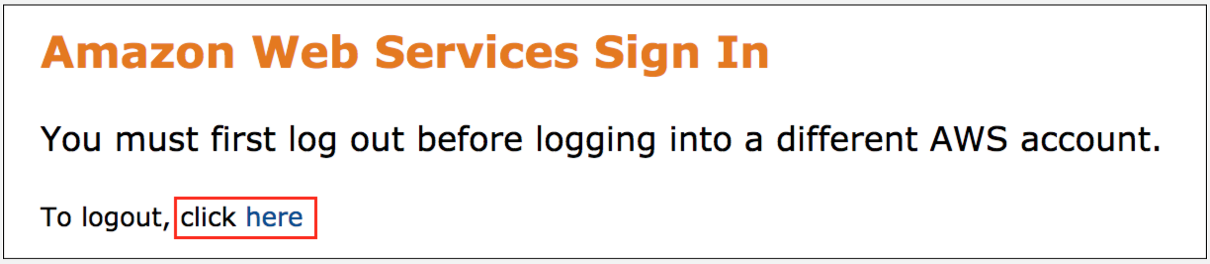
2. To open the lab, choose Open Console.

You are automatically signed in to the AWS Management Console in a new web browser tab.

WARNING: Do not change the Region unless instructed.

COMMON SIGN-IN ERRORS

Error: You must first sign out



Amazon Web Services Sign In

You must first log out before logging into a different AWS account.

To logout, click here

If you see the message, **You must first log out before logging into a different AWS account:**

- Choose the **click here** link.
- Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.

- Choose Open Console again.

Error: Choosing Start Lab has no effect

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

- Add the lab domain name to your pop-up or script blocker's allow list or turn it off.
- Refresh the page and try again.

AWS SERVICES NOT USED IN THIS LAB

AWS service capabilities used in this lab are limited to what the lab requires. Expect errors when accessing other services or performing actions beyond those provided in this lab guide.

Task 1: Explore the lab environment

In this task, you review the lab environment to gain a better understanding of the resources you work with through this lab.

LAB ARCHITECTURE

First, examine the lab architecture.

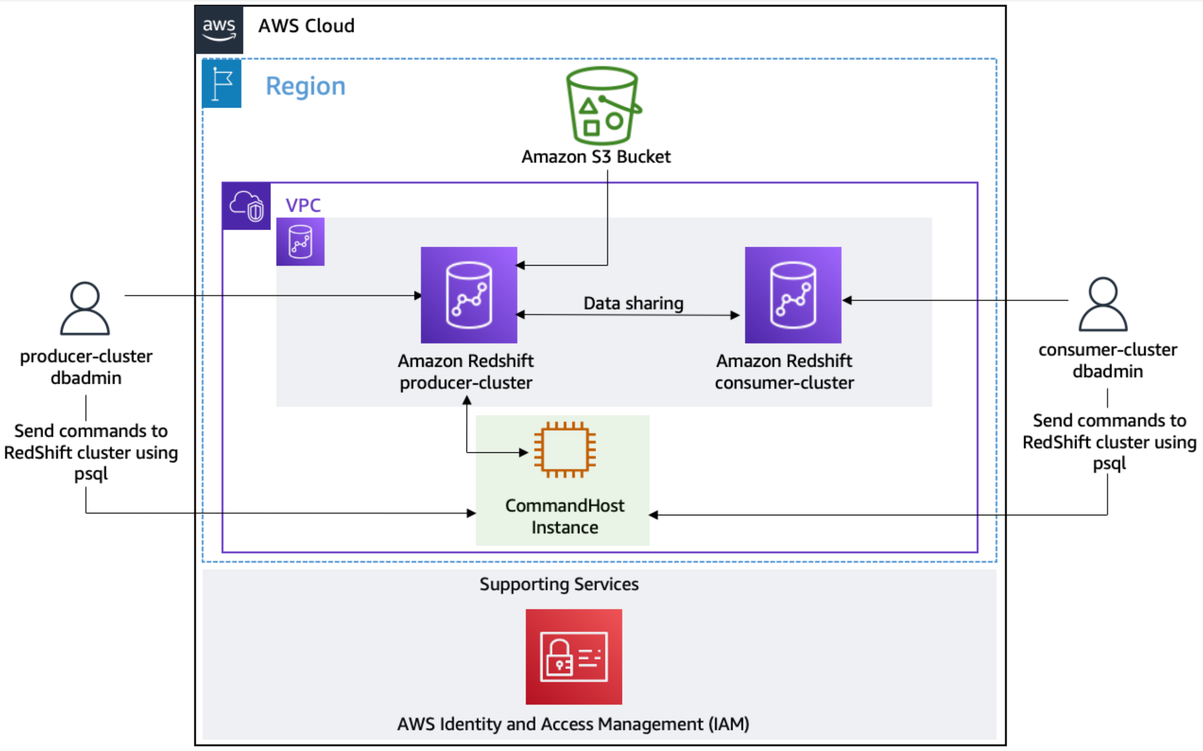

AWS Cloud
Region
Amazon S3 Bucket
VPC
Data sharing
producer-cluster dbadmin
Send commands to RedShift cluster using psql
Amazon Redshift producer-cluster
Amazon Redshift consumer-cluster
consumer-cluster dbadmin
Send commands to RedShift cluster using psql
CommandHost Instance
Supporting Services
AWS Identity and Access Management (IAM)

Image description: The preceding diagram depicts the connection between the producer-cluster admin user and the producer Redshift cluster, along with the connection between the consumer-cluster admin user and the consumer Redshift cluster. Also, both the admin users send commands to the Redshift clusters using psql commands. Lastly, the diagram depicts IAM as the supporting service for this architecture.

During the lab deployment process, the following resources are created for you:

- One VPC
- Two Amazon Redshift clusters, **producer-cluster** and **consumer-cluster**, in a private subnet
- An Amazon S3 bucket that contains the data that you load into Amazon Redshift
- An Amazon Elastic Compute Cloud (Amazon EC2) instance (CommandHost instance) for you to use during the lab.

Connect to this instance through the **CommandHostUrl** value found in the left pane of these instructions. This instance serves as the host to:

- Connect to the Redshift cluster database and run queries using **psql**
- Run AWS CLI commands required for this lab

REVIEW THE SAMPLE DATA IN THE AMAZON S3 BUCKET

One Comma Separated Values (CSV) file was uploaded to an S3 bucket during the lab environment build process. The file contains stock trading data for a number of companies from the year 2001 through 2021, with some simulated data added for future dates.

3. If you have not already done so, follow the steps in the Start Lab section to log into the AWS Management Console.
4. At the top-right corner of the page, verify that the **AWS Region** matches the **Region** listed to the left of these instructions.

Choose the **Region** drop-down menu to display the list of AWS Regions and their associated codes. For example, **US West (Oregon)** has a code of **us-west-2**.

5. At the top of the page, in the unified search bar, search for and choose

   `S3`.
6. Choose the link for the bucket with **databucket** in the name.

Notice that there is one folder in the bucket named **data/**.

7. Choose the **data/** link to open the folder.

There is one CSV file in the folder named **stock_prices.csv** which contains actual trading data from January 2, 2001 through September 14, 2021. The data for September 15, 2021 and beyond is a copy of the 2020 data that is used to provide simulated data when querying relative to today's date.

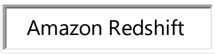
The file contains data similar to this:

Trade_Date	Ticker	High	Low	Open_value	Close	Volume	Adj_Close
2020-01-02	aapl	75.15	73.79	74.05	75.08	135480400.0	74.20

Trade_Date	Ticker	High	Low	Open_value	Close	Volume	Adj_Close
2020-01-02	sq	64.05	62.95	62.99	63.83	5264700	63.83
2020-01-02	amzn	1898.01	1864.15	1875.01	1898.01	4029000	1898.01
2020-01-02	ge	11.96	11.23	11.23	11.93	87421800.0	11.86
2020-01-02	m	17.27	16.39	17.18	16.52	26388100.0	15.86
2020-01-02	tsla	86.14	84.34	84.90	86.05	47660500.0	86.05
2020-01-02	msft	160.73	158.33	158.78	160.62	22622100.0	158.20

VERIFY THAT THE AMAZON REDSHIFT CLUSTERS ARE RUNNING

8. At the top of the page, in the unified search bar, search for and choose


Amazon Redshift

9. On the **Amazon Redshift dashboard**, in the **Cluster overview** section, find two clusters named **consumer-cluster** and **producer-cluster**, and verify that the status of each is Available.

Before moving to task 2, copy the cluster endpoints for both the clusters and paste it on a notepad.

10. From the **Cluster overview** section, choose the link for **producer-cluster**.

On the **producer-cluster** page, in the **General information** section, you find the cluster endpoint. The endpoint should be similar to: **producer-cluster.c6vwej8fodej.us-west-2.redshift.amazonaws.com:5439/producer_stocks**

11. **Copy edit:** Copy the endpoint value on a notepad and remove the **:5439/producer_stocks** portion from the end. Save the remaining endpoint URL for use in the next task.

The final endpoint should be similar to: **producer-cluster.c6vwej8fodej.us-west-2.redshift.amazonaws.com**

12. In the navigation breadcrumbs at the top of the page, choose the **Clusters** link to return to the **Clusters** page.
13. From the **Cluster overview** section, choose the link for **consumer-cluster**.

On the **consumer-cluster** page, in the **General information** section, you find the cluster endpoint. The endpoint should be similar to: **consumer-cluster.c6vwej8fodej.us-west-2.redshift.amazonaws.com:5439/consumer_stocks**

14. **Copy edit:** Copy the endpoint value on a notepad and remove the **:5439/consumer_stocks** portion from the end. Save the remaining endpoint URL for use in the next task.

The final endpoint should be similar to: **consumer-cluster.c6vwej8fodej.us-west-2.redshift.amazonaws.com**

Now that you have reviewed the lab environment, and noted the cluster endpoints for both the clusters, you're ready to begin!

Task 2: Create an external table

In this task, you create an external schema and table in Amazon Redshift that houses the data from the CSV file in the S3 bucket.

DIRECTIONS FOR CONNECTING TO THE COMMAND HOST TO USE PSQL

15. **Copy edit:** Copy the **CommandHostUrl** value found in the left pane of these instructions into a new browser tab to access the command host terminal.
16. **Command:** Run the following commands on the command host:

- Replace the string **<INSERT_PASSWORD>** with **AdministratorPassword** provided to the left of these instructions. (Be sure to keep the single quote marks.)
- Replace **<INSERT_REDSHIFT_CLUSTER_ENDPOINT>** with the value you recorded in the previous task for the **producer-cluster**. Make sure that you have removed the **:5439/producer_stocks** portion from the end before running the command.

```
cd ~
export PGPASSWORD='<INSERT_PASSWORD>'
psql -U dbadmin -h '<INSERT_REDSHIFT_CLUSTER_ENDPOINT>' -d producer_stocks -p
5439
```

Expected output: Your values differ from what is seen below.

```
******************************
**** This is OUTPUT ONLY. ****
******************************

sh-4.2$ cd ~
sh-4.2$ export PGPASSWORD='seo5pzwXx%J>El3'
sh-4.2$ psql -U dbadmin -h producer-cluster.c6vwej8fodej.us-west-
2.redshift.amazonaws.com -d producer_stocks -p 5439
psql (9.2.24, server 8.0.2)
WARNING: psql version 9.2, server version 8.0.
         Some psql features might not work.
SSL connection (cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.

producer_stocks=#
```

This should log you into the **producer_stocks** database within the **producer-cluster** and give you a prompt where you can enter **SQL** commands used in this task.

CREATE AN EXTERNAL SCHEMA AND TABLE

Before you can query the data in the S3 bucket, you must first create an external schema and table that contains the data.

17. Using the psql prompt, enter the following query to create an external schema named **spectrum**:

- Replace the **INSERT_REDSHIFT_ROLE** placeholder value with the **RedshiftRole** value listed to the left of these instructions. (Be sure to keep the single quote marks.)

```
CREATE EXTERNAL SCHEMA spectrum
FROM DATA CATALOG
DATABASE spectrumdb
IAM_ROLE 'INSERT_REDSHIFT_ROLE'
CREATE EXTERNAL DATABASE IF NOT EXISTS;
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

INFO:  External database "spectrumdb" created
CREATE SCHEMA
producer_stocks=#
```

18. Using the psql prompt, enter the following query to create an external table with the **spectrum** schema named **stocksummary**:

- Replace the **INSERT_DATA_BUCKET** placeholder value with the **DataBucket** value listed to the left of these instructions. (Be sure to keep the single quote marks.)

```
DROP TABLE IF EXISTS spectrum.stocksummary;
CREATE EXTERNAL TABLE spectrum.stocksummary(
    Trade_Date VARCHAR(15),
    Ticker VARCHAR(5),
    High DECIMAL(8,2),
    Low DECIMAL(8,2),
    Open_value DECIMAL(8,2),
    Close DECIMAL(8,2),
    Volume DECIMAL(15),
    Adj_Close DECIMAL(8,2)
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION 's3://INSERT_DATA_BUCKET/data/';
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

INFO:  External table "stocksummary" does not exist and will be skipped
DROP TABLE
CREATE EXTERNAL TABLE
producer_stocks=#
```

QUERY THE DATA

Next, query the external table to verify that the data imported correctly.

19. Using the psql prompt, enter the following query to display stock information from January 3, 2020:

```
SELECT * FROM spectrum.stocksummary
    WHERE trade_date = '2020-01-03'
    ORDER BY trade_date ASC, ticker ASC;
```

Expected output:

Notice that the query takes approximately 7 seconds to complete. The query result should display information about 25 stocks, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
trade_date | ticker | high    |   low   | open_value |  close  |  volume   |
adj_close
------------+--------+---------+---------+------------+---------+-----------+-
----------
 2020-01-03 | aal    |   28.29 |   27.34 |      28.27 |   27.65 |  14008900 |
27.55
 2020-01-03 | aapl   |   75.14 |   74.13 |      74.29 |   74.36 | 146322800 |
73.38
 2020-01-03 | amzn   | 1886.20 | 1864.50 |    1864.50 | 1874.97 |   3764400 |
1874.97
 2020-01-03 | ba     |  334.89 |  330.30 |     330.63 |  332.76 |   3875900 |
330.79
 2020-01-03 | bac    |   35.15 |   34.76 |      34.98 |   34.90 |  50357900 |
33.51
 2020-01-03 | c      |   80.52 |   79.45 |      79.80 |   79.70 |  12437400 |
74.88
 2020-01-03 | chwy   |   29.40 |   28.53 |      29.00 |   29.34 |   2205300 |
29.34
 2020-01-03 | coke   |  287.36 |  277.48 |     279.77 |  285.76 |     37500 |
283.91
 2020-01-03 | dis    |  147.90 |  146.05 |     146.40 |  146.50 |   7320200 |
146.50
 2020-01-03 | f      |    9.37 |    9.15 |       9.31 |    9.21 |  45040800 |
9.06
 2020-01-03 | ge     |   96.00 |   92.24 |      92.56 |   95.76 |  10735725 |
95.13
 2020-01-03 | gs     |  232.61 |  230.30 |     231.60 |  231.58 |   2274500 |
223.41
 2020-01-03 | hsy    |  145.89 |  143.76 |     143.97 |  145.26 |    770900 |
140.04
 2020-01-03 | intc   |   60.70 |   59.81 |      59.81 |   60.10 |  15293900 |
57.56
 2020-01-03 | kodk   |    4.19 |    3.92 |       4.00 |    4.03 |    242900 |
4.03
 2020-01-03 | m      |   16.61 |   16.21 |      16.32 |   16.53 |  12026100 |
15.87
 2020-01-03 | ma     |  302.42 |  298.60 |     299.46 |  300.43 |   2501300 |
297.74
 2020-01-03 | msft   |  159.95 |  158.06 |     158.32 |  158.62 |  21116200 |
155.94
 2020-01-03 | nke    |  102.00 |  100.31 |     100.59 |  101.92 |   4541800 |
100.38
```

```
 2020-01-03 | pg       | 123.53 | 121.86 |       122.16 | 122.58 |    7970500 |
117.41
 2020-01-03 | pypl     | 110.42 | 108.76 |       109.49 | 108.76 |    7098300 |
108.76
 2020-01-03 | sq       |  63.27 |  62.33 |        62.59 |  63.00 |    5087100 |
63.00
 2020-01-03 | tsla     |  90.80 |  87.38 |        88.10 |  88.60 |   88892500 |
88.60
 2020-01-03 | v        | 190.96 | 187.92 |       188.41 | 189.60 |    4899700 |
187.62
 2020-01-03 | wmt      | 118.79 | 117.59 |       118.27 | 117.89 |    5399200 |
114.60
(25 rows)

producer_stocks=#
```

Congratulations! You have successfully created an external schema and table, and then queried the data in the S3 bucket.

Task 3: Create and query a materialized view

Suppose you would like to query your stock data to determine the top three stocks, based on volume, over a 7-day span. However, you only want to retrieve the date, stock ticker, and volume. You don't need to pull the additional fields.

In this task, you create a materialized view to compile only the fields from the base table that you are interested in querying. (If you had data in multiple tables, you could use a materialized view to combine data from them all into a single, queryable view.)

For more information on materialized views, refer to *Creating Materialized Views in Amazon Redshift* in the **Additional Resources** section at the end of this lab.

CREATE A MATERIALIZED VIEW

20. Using the psql prompt, enter the following query to create a materialized view named **stocks_mv** that includes only the **trade_date**, **ticker**, and **volume** columns:

```
DROP MATERIALIZED VIEW IF EXISTS stocks_mv;
CREATE MATERIALIZED VIEW stocks_mv AS
    SELECT trade_date, ticker, volume FROM spectrum.stocksummary;
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

INFO:  Materialized View "stocks_mv" does not exist and will be skipped
DROP MATERIALIZED VIEW
WARNING:  An incrementally maintained materialized view could not be created,
reason: External tables other than Elastic Views are unsupported. The
```

```
materialized view created, stocks_mv,will be recomputed from scratch for every
REFRESH.
CREATE MATERIALIZED VIEW
producer_stocks=#
```

QUERY THE DATA IN THE MATERIALIZED VIEW

Next, query the data in the materialized view to discover how it has transformed when compared to the source data.

21. Using the psql prompt, enter the following query to display stock information in the materilized view from January 3, 2020:

```
SELECT * FROM stocks_mv
    WHERE trade_date = '2020-01-03'
    ORDER BY trade_date ASC, ticker ASC;
```

You ran a similar query in the previous task. Notice the **FROM** value for this query is the materialized view, **stocks_mv**.

Expected output:

The query result should display information about 25 stocks, but with only three columns, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
trade_date | ticker |  volume
------------+--------+-----------
 2020-01-03 | aal    |  14008900
 2020-01-03 | aapl   | 146322800
 2020-01-03 | amzn   |   3764400
 2020-01-03 | ba     |   3875900
 2020-01-03 | bac    |  50357900
 2020-01-03 | c      |  12437400
 2020-01-03 | chwy   |   2205300
 2020-01-03 | coke   |     37500
 2020-01-03 | dis    |   7320200
 2020-01-03 | f      |  45040800
 2020-01-03 | ge     |  10735725
 2020-01-03 | gs     |   2274500
 2020-01-03 | hsy    |    770900
 2020-01-03 | intc   |  15293900
 2020-01-03 | kodk   |    242900
 2020-01-03 | m      |  12026100
 2020-01-03 | ma     |   2501300
 2020-01-03 | msft   |  21116200
 2020-01-03 | nke    |   4541800
 2020-01-03 | pg     |   7970500
 2020-01-03 | pypl   |   7098300
 2020-01-03 | sq     |   5087100
 2020-01-03 | tsla   |  88892500
 2020-01-03 | v      |   4899700
 2020-01-03 | wmt    |   5399200
(25 rows)

producer_stocks=#
```

Notice that the data displayed is the same as in the previous task, but it only includes the columns that you are most concerned with.

QUERY FOR THE MOST POPULAR STOCKS

Now that you have limited the data, you would like to query the materialized view for the top three most popular stocks over a 7-day span.

22. Using the psql prompt, enter the following query to display the top three stocks by volume from February 10th, 2020 to February 16th, 2020:

```
WITH tmp_variables AS (
SELECT
   '2020-02-16'::DATE AS StartDate
)

SELECT
    ticker,
    SUM(volume) AS sum_volume
FROM stocks_mv
WHERE trade_date BETWEEN (SELECT StartDate FROM tmp_variables)-6 AND (SELECT
StartDate FROM tmp_variables)
GROUP BY ticker
ORDER BY sum_volume DESC
LIMIT 3;
```

Expected output:

The query result should display the sum of the trades for a given stock, sorted from most to least, and limited to the top three results, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

ticker | sum_volume
--------+------------
aapl    | 492263600
tsla    | 451961000
f       | 377544700
(3 rows)

producer_stocks=#
```

Keep the **CommandHostUrl** Session Manager session running.

Congratulations! You have successfully created and queried an Amazon Redshift materialized view.

Task 4: Use Amazon Redshift data sharing for faster data access between clusters

In a situation where you might need to share data between Redshift clusters, you could use scheduled ETL operations to copy relevant data from one cluster to another. However, that would mean the data could potentially be out of date until the next ETL operation runs. Because you are creating a copy of the same data, you would also incur additional data storage costs for the copies.

With Amazon Redshift data sharing, you can grant access directly to specific data and objects in one Redshift cluster to entities in another cluster. For example, with the stock-related data you have been working with, you might have two departments in an organization that require access to the data.

In this example scenario, let's say the data is stored with the Trading department, which owns the **producer-cluster** cluster and **producer_stocks** database. The Risk Assessment and Volatility (RAV) department owns the **consumer-cluster** cluster and **consumer_stocks** database. The RAV department must be able to run reports as needed to track the trading volume of specific stocks over a set time frame. They are frustrated that they must wait for data to be copied to their database and are unable to access the live data that the Trading department maintains.

In this task, you create an Amazon Redshift datashare to share data from the **producer_cluster** to the **consumer_cluster**. You then query the data in the **producer_cluster** from the **consumer_cluster** to verify that you can access the data. Lastly, you revoke the datashare permissions for the **consumer_cluster** and attempt to access the data again.

For more information about Amazon Redshift datashares, refer to *Overview of Data Sharing in Amazon Redshift* in the **Additional Resources** section at the end of this lab.

CREATE A DATASHARE FROM THE PRODUCER CLUSTER

23. Switch back to the browser tab open to the AWS Management console.
24. At the top of the page, in the unified search bar, search for and choose

    Amazon Redshift.
25. Select the **Clusters** drop down section.

In the **Clusters** section, you find the cluster namespace.

26. Copy this namespace value for

    consumer-cluster &

    producer-cluster to a notepad. You use it in the following steps.

The cluster namespace should be similar to: **8a8fcb18-29b8-480f-b19f-49ad0b18e421**

You use each namespace when working with the datashare throughout this task.

27. Switch back to the browser tab open to the AWS Systems Manager - Session Manager console.

28. Using the psql prompt, enter the following query to create a datashare named **stocks_share**:

```
CREATE DATASHARE stocks_share;
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
CREATE DATASHARE
producer_stocks=#
```

29. Using the psql prompt, enter the following query to add the **public** schema and **stocks_mv** materialized view to the **stocks_share** datashare:

```
ALTER DATASHARE stocks_share ADD SCHEMA public;
```

```
ALTER DATASHARE stocks_share ADD TABLE public.stocks_mv;
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
ALTER DATASHARE
ALTER DATASHARE
producer_stocks=#
```

30. Using the psql prompt, enter the following query to grant permission for the **consumer_cluster** to access the **stocks_share** datashare:

- Replace the **INSERT_CONSUMER_NAMESPACE_ID** placeholder value with the **Cluster namespace** ID for the **consumer_cluster** cluster that you made note of previously. (Be sure to keep the single quote marks.)

```
GRANT USAGE ON DATASHARE stocks_share TO NAMESPACE
'INSERT_CONSUMER_NAMESPACE_ID';
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
GRANT
producer_stocks=#
```

VERIFY THAT THE DATASHARE WAS CREATED AND CONFIGURED CORRECTLY

Next, verify that the datashare was created and configured successfully.

31. Using the psql prompt, enter the following query to display all datashares:

```
SELECT * FROM svv_datashares;
```

Expected output:

The query result should display the details of the **stocks_share** datashare, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

  share_name   | share_id | share_owner | source_database | consumer_database |
share_type |      createdate       | is_publicaccessible | share_acl |
producer_account |                         producer_namespace
--------------+----------+-------------+-----------------+-------------------
+------------+---------------------+---------------------+-----------+--------
----------+------------------------------------------
----------------------+------------
 stocks_share |    106400 |         100 | producer_stocks |                   |
OUTBOUND    | 2023-02-27 17:01:08 | f                    |           |
407561589472     | 393e8eb3-27d2-433b-8d36-6c76098d65d2
(1 row)

producer_stocks=#
```

Notice that the **share_type** is **OUTBOUND** because the data is being shared from the cluster that you are currently connected to.

32. Using the psql prompt, enter the following query to display all objects that have been added to the **stocks_share** datashare:

```
SELECT * FROM svv_datashare_objects;
```

Expected output:

The query result should display the objects in the **stocks_share** datashare, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

 share_type |  share_name  |    object_type    |   object_name    |
producer_account |          producer_namespace          | include_new
------------+--------------+-------------------+------------------+-----------
-------+--------------------------------------+-------------
 OUTBOUND   | stocks_share | materialized view | public.stocks_mv |
407561589472     | 393e8eb3-27d2-433b-8d36-6c76098d65d2 |
 OUTBOUND   | stocks_share | schema            | public           |
407561589472     | 393e8eb3-27d2-433b-8d36-6c76098d65d2 | f
(2 rows)

producer_stocks=#
```

Notice that the **share_type** is **OUTBOUND** and that the object types and names match the **schema** and **materialized view** that you added.

33. Using the psql prompt, enter the following query to display Amazon Redshift clusters that have been granted access to view data in the **stocks_share** datashare:

```
SELECT * FROM svv_datashare_consumers;
```

Expected output:

The query result should display one consumer, and the **consumer_namespace** should match the **cluster namespace** of the **consumer-cluster** cluster, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

  share_name   | consumer_account |          consumer_namespace          |
share_date
--------------+------------------+--------------------------------------+-----
----------------
 stocks_share |                  | 8a8fcb18-29b8-480f-b19f-49ad0b18e421 |
2023-02-27 17:04:40
(1 row)

producer_stocks=#
```

VERIFY DATASHARE ACCESS FROM THE CONSUMER CLUSTER

Now that you have confirmed the datashare was created and configured successfully, verify that you can access it from the consumer cluster.

Before proceeding further, you disconnect from the **producer-cluster** and connect to the **consumer-cluster**.

34. Using the psql prompt, enter the following command to disconnect from the **producer-cluster**:

```
\q
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

producer_stocks=# \q
sh-4.2$
```

35. **Command:** Run the following commands on the command host:

- Replace **<INSERT_REDSHIFT_CLUSTER_ENDPOINT>** with the value you recorded in the previous task for the **consumer-cluster**. Make sure that you have removed the **:5439/consumer_stocks** portion from the end before running the command.

```
psql -U dbadmin -h '<INSERT_REDSHIFT_CLUSTER_ENDPOINT>' -d consumer_stocks -p
5439
```

Expected output: Your values differ from what is seen below.

```
******************************
**** This is OUTPUT ONLY. ****
******************************

sh-4.2$ cd ~
psql -U dbadmin -h consumer-cluster.c6vwej8fodej.us-west-
2.redshift.amazonaws.com -d consumer_stocks -p 5439
psql (9.2.24, server 8.0.2)
WARNING: psql version 9.2, server version 8.0.
         Some psql features might not work.
SSL connection (cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.

consumer_stocks=#
```

This should log you into the **consumer_stocks** database within the **consumer-cluster** and give you a prompt where you can enter **SQL** commands used in this task.

36. Using the psql prompt, enter the following query to display all datashares:

```
SELECT * FROM svv_datashares;
```

Expected output:

The query result should display the details of the **stocks_share** datashare, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

  share_name   | share_id | share_owner | source_database | consumer_database |
share_type | createdate | is_publicaccessible | share_acl | producer_account |
producer_namespace
            | managed_by
--------------+----------+-------------+-----------------+-------------------
+------------+------------+---------------------+-----------+------------------
-+-------------------------------------------------------------------+---------
---
 stocks_share |          |             |                 |                   |
INBOUND    |            | f                   |           | 407561589472     |
393e8eb3-27d2-433b-8d36-6c76098d65d2
            |
(1 row)

consumer_stocks=#
```

Notice that the **share_type** is **INBOUND** because the data is being shared *from* another cluster.

37. Using the psql prompt, enter the following query to display all objects that have been added to the **stocks_share** datashare:

```
SELECT * FROM svv_datashare_objects;
```

Expected output:

The query result should display the objects in the **stocks_share** datashare, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

 share_type |  share_name  |    object_type    |   object_name    |
producer_account |          producer_namespace            | include_new
------------+--------------+-------------------+------------------+-----------
-------+--------------------------------------+-------------
 INBOUND    | stocks_share | materialized view | public.stocks_mv |
407561589472     | 393e8eb3-27d2-433b-8d36-6c76098d65d2 |
 INBOUND    | stocks_share | schema            | public           |
407561589472     | 393e8eb3-27d2-433b-8d36-6c76098d65d2 |
(2 rows)

consumer_stocks=#
```

Notice that the **share_type** is **INBOUND** and that the object types and names match the **schema** and **materialized view** that you added.

QUERY THE DATA IN THE DATASHARE

Now that you have verified that you can access the datashare objects from the **consumer-cluster** cluster, you can create a new database from the datashare and then query the data.

38. Using the psql prompt, enter the following query to create a new local database named **stock_summary** to reference the shared objects:

- Replace the **INSERT_PRODUCER_NAMESPACE_ID** placeholder value with the **Cluster namespace** ID for the **producer_cluster** cluster that you made note of previously. (Be sure to keep the single quote marks.)

```
CREATE DATABASE stock_summary FROM DATASHARE stocks_share of NAMESPACE
'INSERT_PRODUCER_NAMESPACE_ID';
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

CREATE DATABASE
consumer_stocks=#
```

Next, run a query against the database that you just created to retrieve information that is stored in the datashare.

39. Using the psql prompt, enter the following query to retrieve stock data on January, 3, 2020:

```
SELECT * FROM stock_summary.public.stocks_mv
    WHERE trade_date = '2020-01-03'
    ORDER BY trade_date ASC, ticker ASC;
```

Notice that the materialized view query takes approximately 1 second to complete, which is significantly faster when compared to 7 seconds for the full query.

Expected output:

The query result should display 25 stock tickers and associated trade volume from January 3, 2020, similar to this:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

trade_date | ticker |  volume
-----------+--------+-----------
 2020-01-03 | aal    |  14008900
 2020-01-03 | aapl   | 146322800
 2020-01-03 | amzn   |   3764400
 2020-01-03 | ba     |   3875900
 2020-01-03 | bac    |  50357900
 2020-01-03 | c      |  12437400
 2020-01-03 | chwy   |   2205300
 2020-01-03 | coke   |     37500
 2020-01-03 | dis    |   7320200
 2020-01-03 | f      |  45040800
 2020-01-03 | ge     |  10735725
 2020-01-03 | gs     |   2274500
 2020-01-03 | hsy    |    770900
 2020-01-03 | intc   |  15293900
 2020-01-03 | kodk   |    242900
 2020-01-03 | m      |  12026100
 2020-01-03 | ma     |   2501300
 2020-01-03 | msft   |  21116200
 2020-01-03 | nke    |   4541800
 2020-01-03 | pg     |   7970500
 2020-01-03 | pypl   |   7098300
 2020-01-03 | sq     |   5087100
 2020-01-03 | tsla   |  88892500
 2020-01-03 | v      |   4899700
 2020-01-03 | wmt    |   5399200
(25 rows)

consumer_stocks=#
```

REVOKE ACCESS TO THE DATASHARE

You might find yourself in a situation where a department should no longer have access to the data in a datashare. In such a scenario, you can revoke access to the datashare.

Before proceeding further, you disconnect from the **consumer-cluster**.

40. Using the psql prompt, enter the following command to disconnect from the **consumer-cluster**:

```
\q
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

consumer_stocks=# \q
sh-4.2$
```

41. **Command:** Run the following commands on the command host:

- Replace **<INSERT_REDSHIFT_CLUSTER_ENDPOINT>** with the value you recorded in the previous task for the **producer-cluster**. Make sure that you have removed the **:5439/producer_stocks** portion from the end before running the command.

```
psql -U dbadmin -h '<INSERT_REDSHIFT_CLUSTER_ENDPOINT>' -d producer_stocks -p
5439
```

Expected output: Your values differ from what is seen below.

```
******************************
**** This is OUTPUT ONLY. ****
******************************

psql (9.2.24, server 8.0.2)
WARNING: psql version 9.2, server version 8.0.
         Some psql features might not work.
SSL connection (cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.

producer_stocks=#
```

This should log you into the **producer_stocks** database within the **producer-cluster** and give you a prompt where you can enter **SQL** commands to modify the datashare settings.

42. Using the psql prompt, enter the following query to revoke access to the **stocks_share** datashare for the **consumer-cluster** cluster:

- Replace the **INSERT_CONSUMER_NAMESPACE_ID** placeholder value with the **Cluster namespace** ID for the **consumer_cluster** cluster that you made note of previously. (Be sure to keep the single quote marks.)

```
REVOKE USAGE ON DATASHARE stocks_share FROM NAMESPACE
'INSERT_CONSUMER_NAMESPACE_ID';
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
```

```
******************************
```

```
REVOKE
producer_stocks=#
```

Before proceeding further, you disconnect from the **producer-cluster**.

43. Using the psql prompt, enter the following command to disconnect from the **producer-cluster**:

```
\q
```

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
producer_stocks=# \q
sh-4.2$
```

VERIFY THAT THE CONSUMER CLUSTER CAN NO LONGER ACCESS THE DATASHARE

44. **Command:** Run the following commands on the command host to connect to the **consumer-cluster**:

- Replace **<INSERT_REDSHIFT_CLUSTER_ENDPOINT>** with the value you recorded in the previous task for the **consumer-cluster**. Make sure that you have removed the **:5439/consumer_stocks** portion from the end before running the command.

```
psql -U dbadmin -h '<INSERT_REDSHIFT_CLUSTER_ENDPOINT>' -d consumer_stocks -p
5439
```

Expected output: Your values differ from what is seen below.

```
******************************
**** This is OUTPUT ONLY. ****
******************************
```

```
psql (9.2.24, server 8.0.2)
WARNING: psql version 9.2, server version 8.0.
         Some psql features might not work.
SSL connection (cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.

consumer_stocks=#
```

This should log you into the **consumer_stocks** database within the **consumer-cluster** and give you a prompt where you can enter **SQL** commands.

Next, run the same query against the datashare that you ran previously.

45. Using the psql prompt, enter the following query to retrieve stock data on January, 3, 2020:

```
SELECT * FROM stock_summary.public.stocks_mv
    WHERE trade_date = '2020-01-03'
    ORDER BY trade_date ASC, ticker ASC;
```

Expected output:

The query should result in the following error message, which signifies that the consumer cluster no longer has access to the datashare:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

ERROR:  The requested data share doesn't exist or is not accessible.
consumer_stocks=#
```

Congratulations! You have successfully created a datashare, accessed data between clusters, and then revoked access for a cluster.

Conclusion

Congratulations! You now have successfully:

- Performed an ELT operation with materialized views and stored procedures.
- Used Amazon Redshift scheduled queries.
- Queried data directly from the source using Amazon Redshift data sharing.

End lab

Follow these steps to close the console and end your lab.

46. Return to the **AWS Management Console**.
47. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
48. Choose End lab and then confirm that you want to end your lab.