Database Migration Workshop

AWS Database Migration Service (DMS)

Oracle to Aurora PostgreSQL

July 2019

Rev July-02-2019

Table of Contents

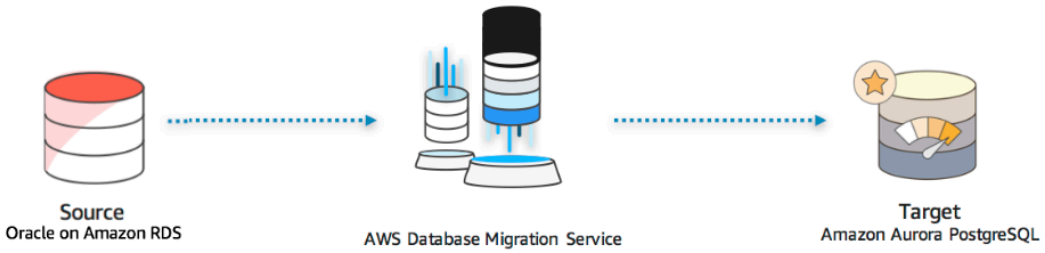
Overview

AWS Database Migration Service (DMS) helps you migrate databases to AWS easily and securely. The source database remains fully operational during the migration, minimizing downtime to applications that rely on the database. AWS DMS can migrate your data to and from most widely used commercial and open-source databases. The service supports homogenous migrations such as Oracle Server to Oracle Server or PostgreSQL to Aurora PostgreSQL, as well as heterogeneous migrations between different database platforms, such as SQL Server to Amazon Aurora or Oracle to PostgreSQL. AWS DMS can also be used for continuous data replication with high-availability.

In this lab session we will perform database migration from a source Oracle Server on an Amazon RDS instance to a target Amazon Aurora (PostgreSQL) database.

This lab will walk you through:

a) Performing a full load migration of source oracle server to target Aurora PostgreSQL database using AWS DMS.
b) Capturing change data set (CDC) from the Oracle Server EC2 instance to an RDS Aurora PostgreSQL instance using AWS DMS.



Connecting to your Environment

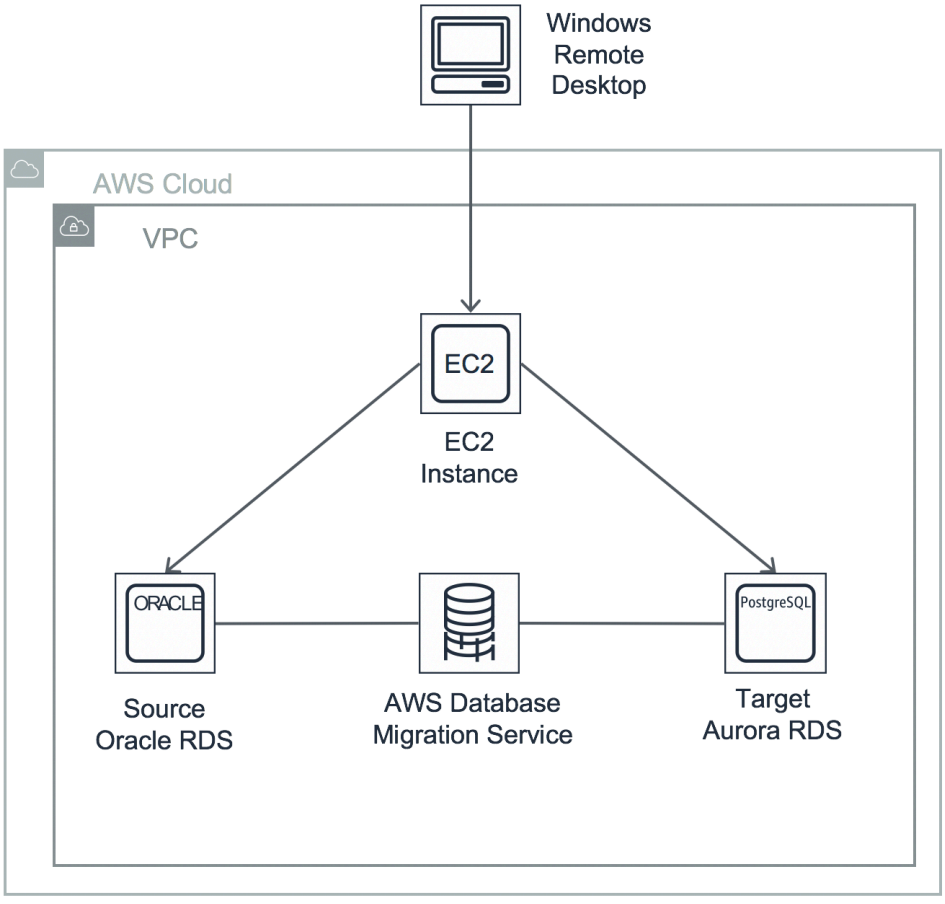
Before launching this lab, make sure you have completed the instructions in the following guide to setup your lab environment: https://bit.ly/2xL3SMw

The environment for this lab consists of:

- An EC2 instance used to run the AWS Schema Conversion Tool (SCT) as well as other applications needed to complete the lab.
- An Amazon RDS instance used to host the source Oracle database
- An Amazon RDS Aurora PostgreSQL instance used as the target database

Once you have completed the instructions in the above referenced document, take special note of the following output values:

- SourceEC2PublicDNS
- SourceOracleEndpoint
- TargetAuroraPostgreSQLEndpoint



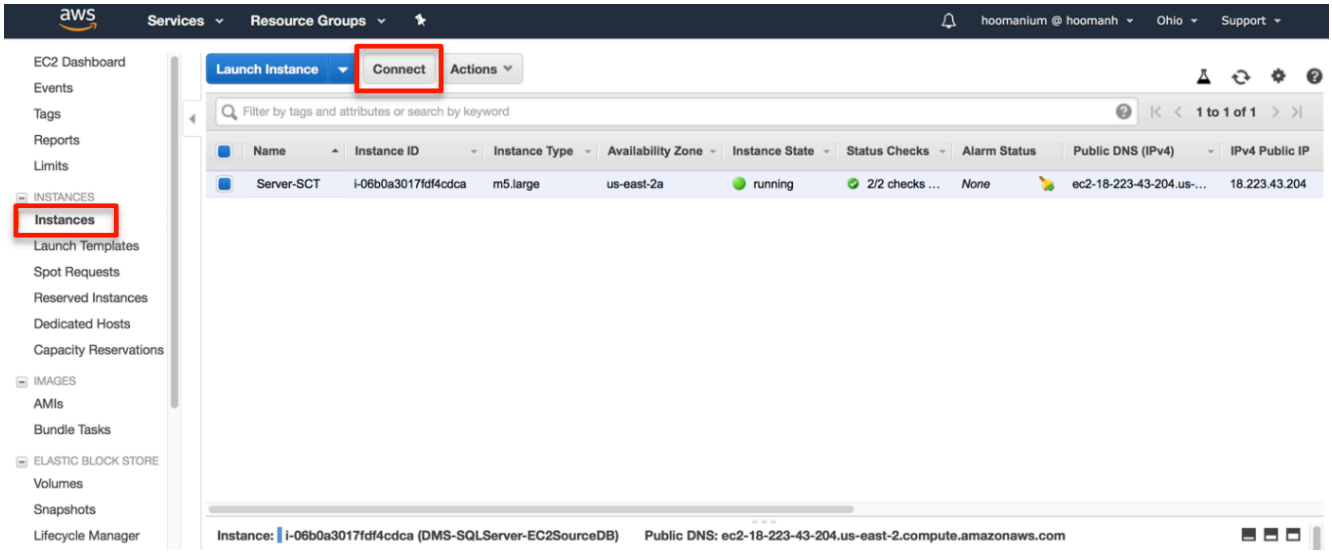
Database Migration Service (DMS)

The following steps provide instructions to migrate existing data from the source Oracle running on an Amazon RDS instance to a target Amazon Aurora (PostgreSQL) database. In this exercise you perform the following tasks:

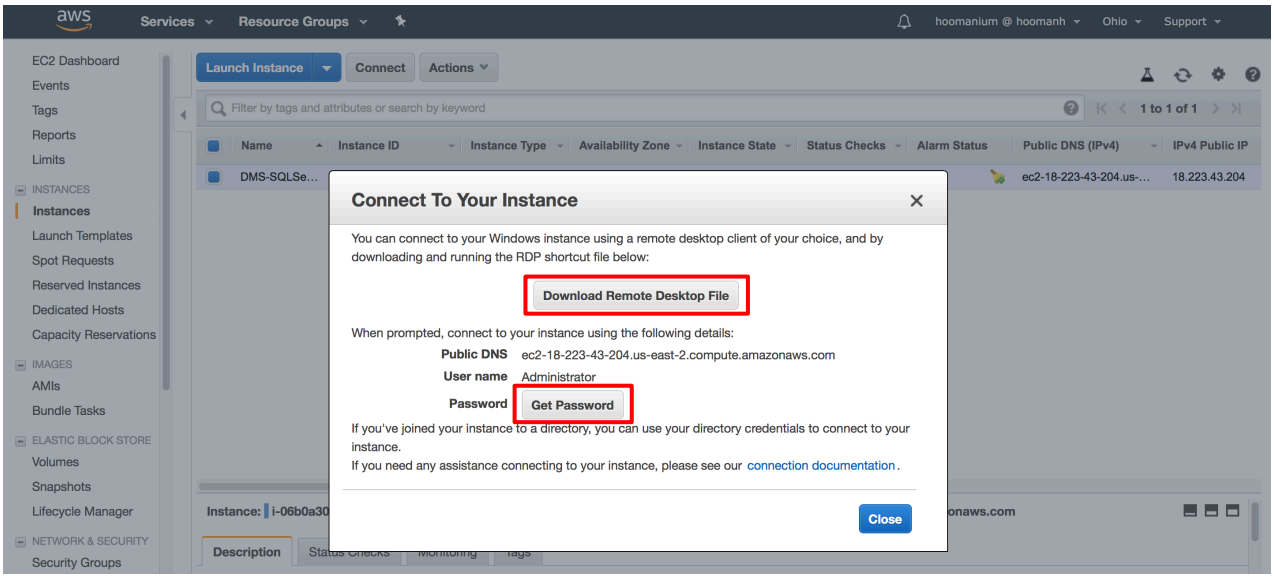
- Connect the source Oracle database
- Configure the source database for replication
- Configure the target database for replication
- Create an AWS DMS Replication Instance
- Create AWS DMS source and target endpoints
- Create and run your AWS DMS migration task
- Replicate ongoing changes from the source to the target database

Log on the EC2 instance

1. Go to the AWS EC2 [redacted] and click on **Instances** in the left column.
2. Select the instance with the name **Server-SCT** and then click the **Connect** button.



3. In this step, you perform 3 tasks:
    a. Click the **Get Password** button and upload the **Key Pair** file that you downloaded earlier. Please take note of the EC2 console generated administrator password.
    b. Click on **Download Remote Desktop File** to download the RDP file to access this EC2 instance.
    c. Connect to the EC2 instance using the RDP.

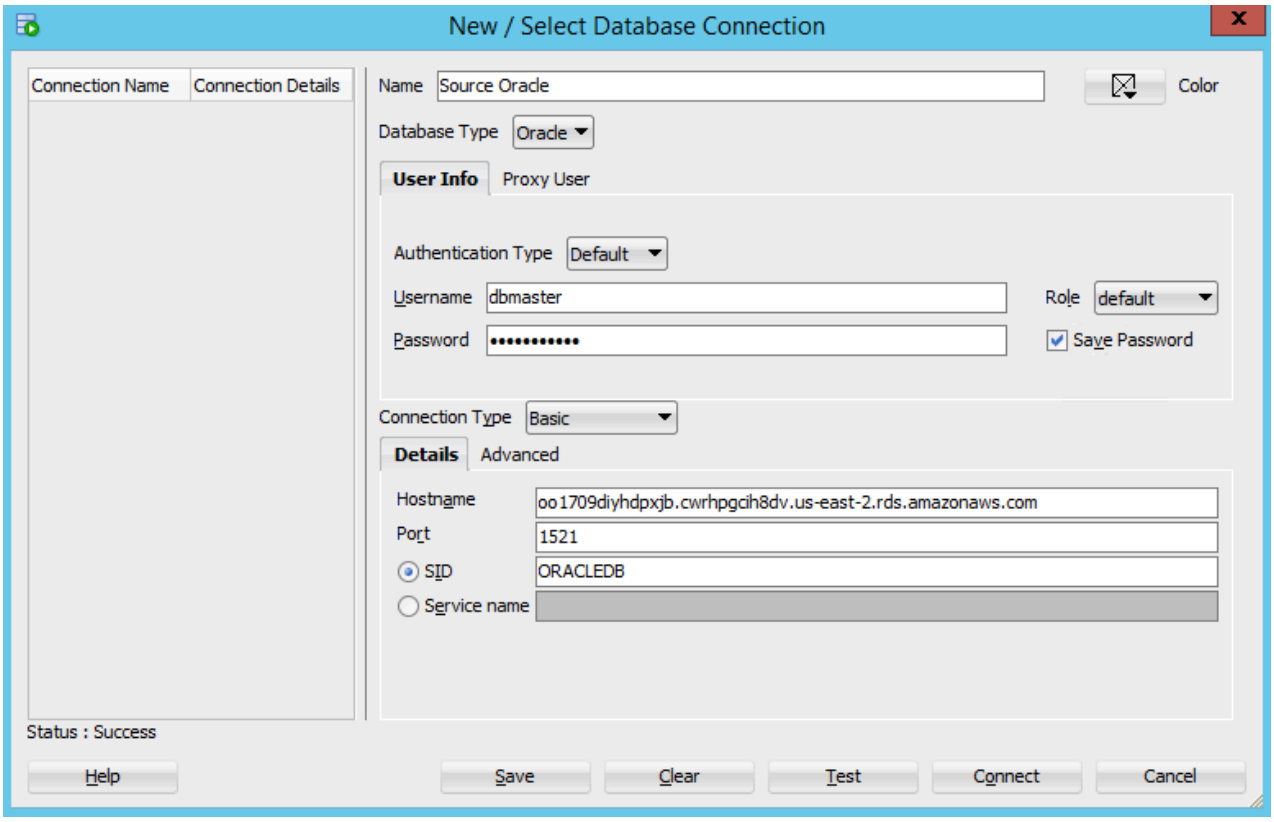


Connect to the Source Oracle on Amazon RDS

4. Once you have connected to the EC2 instance, open **Oracle SQL Developer** from the **Taskbar**.

5. Click on the **plus sign** from the left-hand menu to create a **New Database Connection** using the following values, then click **Connect**.

Parameter	Value
Connection Name	Source Oracle
Username	dbmaster
Password	dbmaster123
Save Password	Check
Hostname	**< SourceOracleEndpoint >**
Port	1521
SID	ORACLEDB



6. After the you see the test status as **Successful**, click **Connect.**

Configure the Source Oracle database for Replication

To use Oracle as a source for AWS Database Migration Service (AWS DMS), you must first provide a user account (DMS user) with read and write privileges on the Oracle database.

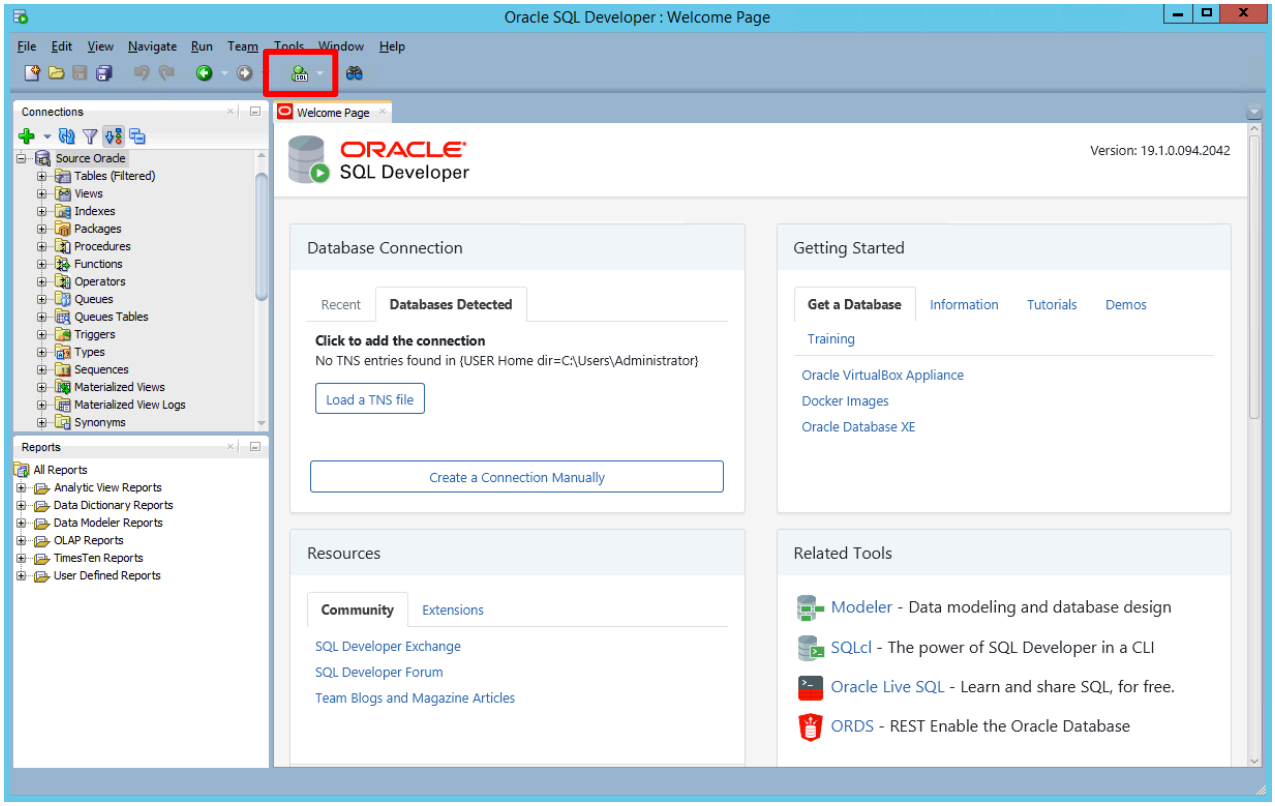
You also need to ensure that ARCHIVELOG MODE is on to provide information to LogMiner. AWS DMS uses LogMiner to read information from the archive logs so that AWS DMS can capture changes.

For AWS DMS to read this information, make sure the archive logs are retained on the database server as long as AWS DMS requires them. Retaining archive logs for 24 hours is usually sufficient.

To capture change data, AWS DMS requires database-level supplemental logging to be enabled on your source database. Doing this ensures that the LogMiner has the minimal information to support various table structures such as clustered and index-organized tables.

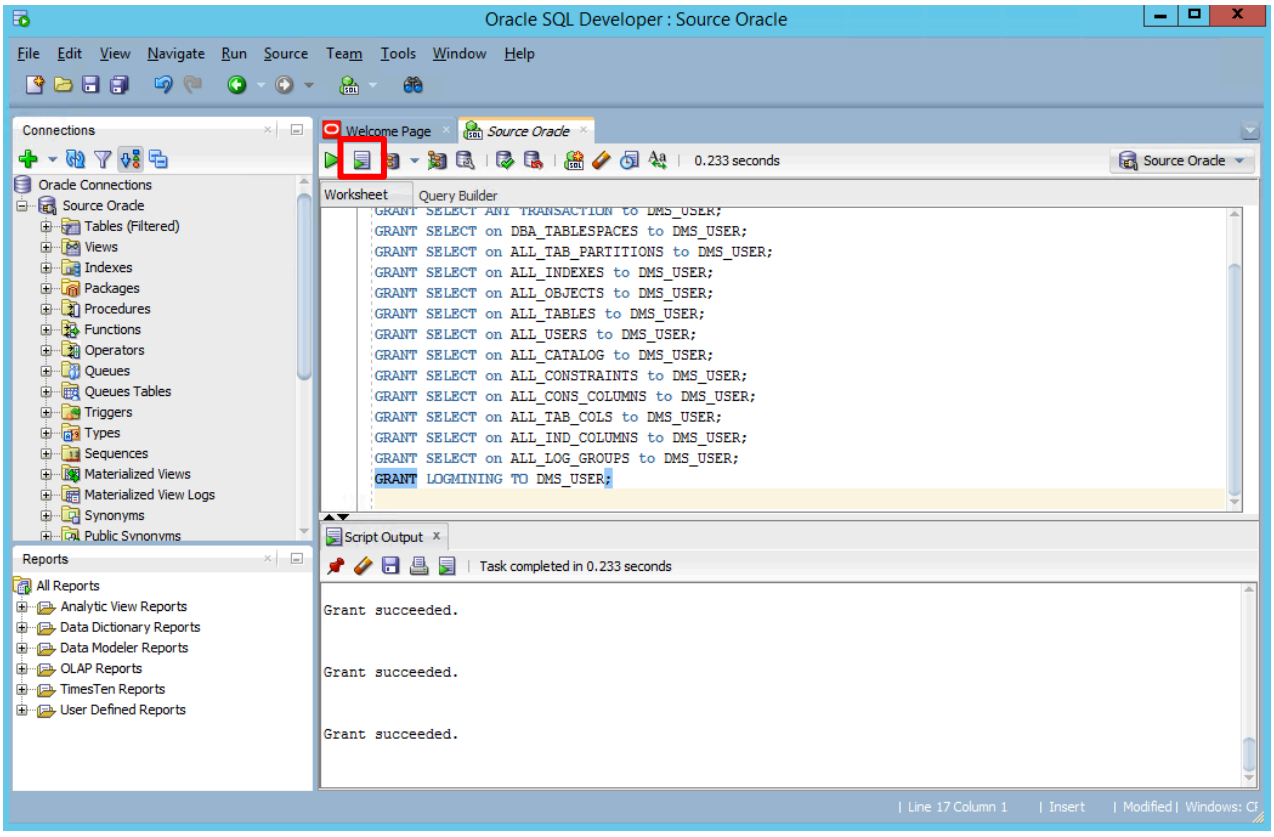
Similarly, you need to enable table-level supplemental logging for each table that you want to migrate.

7. Click on the **SQL Worksheet** icon within **Oracle SQL Developer**, then connect to the **Source Oracle** database.



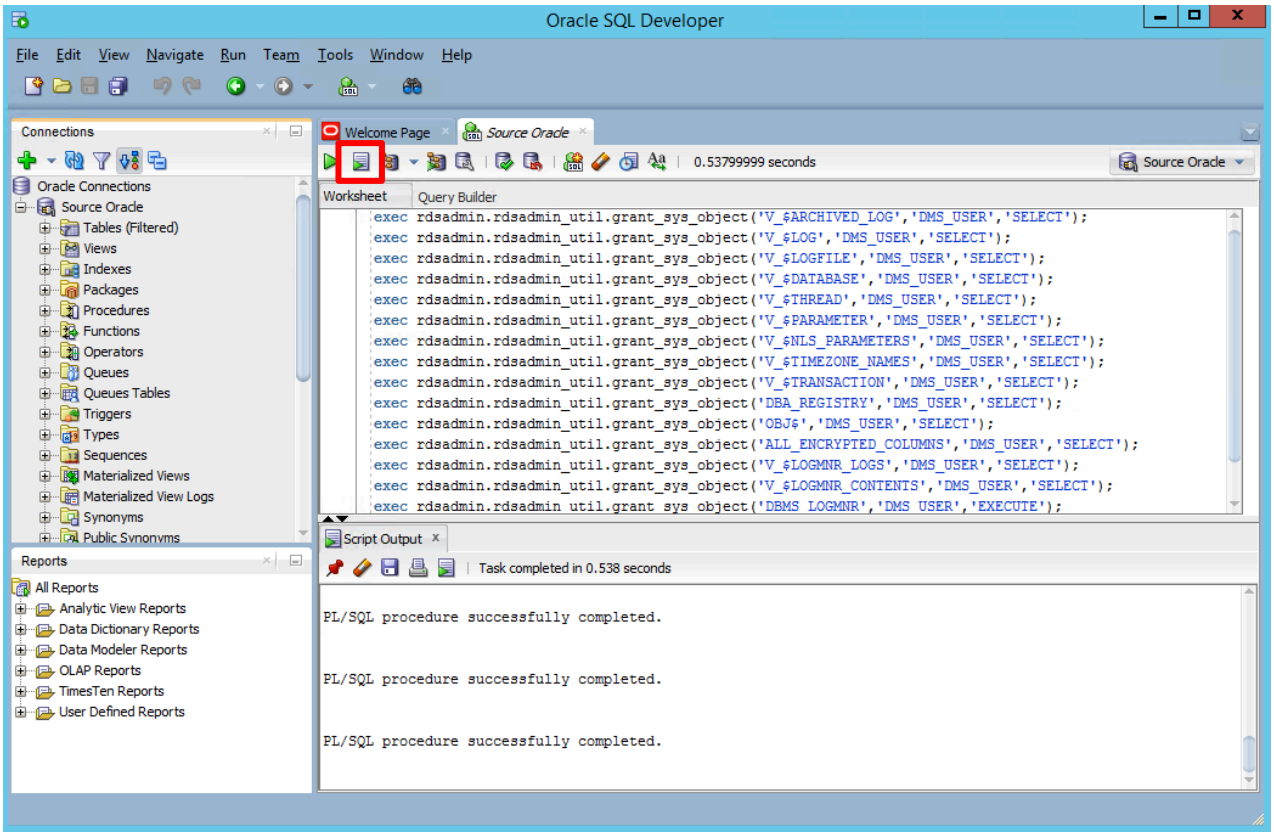
8. Next, execute the below statements to grant the following privileges to the AWS DMS user to access the source Oracle endpoint:

```
GRANT SELECT ANY TABLE to DMS_USER;
GRANT SELECT on ALL_VIEWS to DMS_USER;
GRANT SELECT ANY TRANSACTION to DMS_USER;
GRANT SELECT on DBA_TABLESPACES to DMS_USER;
GRANT SELECT on ALL_TAB_PARTITIONS to DMS_USER;
GRANT SELECT on ALL_INDEXES to DMS_USER;
GRANT SELECT on ALL_OBJECTS to DMS_USER;
GRANT SELECT on ALL_TABLES to DMS_USER;
GRANT SELECT on ALL_USERS to DMS_USER;
GRANT SELECT on ALL_CATALOG to DMS_USER;
GRANT SELECT on ALL_CONSTRAINTS to DMS_USER;
GRANT SELECT on ALL_CONS_COLUMNS to DMS_USER;
GRANT SELECT on ALL_TAB_COLS to DMS_USER;
GRANT SELECT on ALL_IND_COLUMNS to DMS_USER;
GRANT SELECT on ALL_LOG_GROUPS to DMS_USER;
GRANT LOGMINING TO DMS_USER;
```



9. In addition, run the following:

```
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVED_LOG','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOG','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGFILE','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATABASE','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$THREAD','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$PARAMETER','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$NLS_PARAMETERS','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TIMEZONE_NAMES','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TRANSACTION','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_REGISTRY','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('OBJ$','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_ENCRYPTED_COLUMNS','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGMNR_LOGS','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGMNR_CONTENTS','DMS_USER','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_LOGMNR','DMS_USER','EXECUTE');
```



10. Run the following query to retain archived redo logs of the source Oracle database instance for 24 hours:

```
exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24);
```

11. Run the following query to enable database-level supplemental logging:

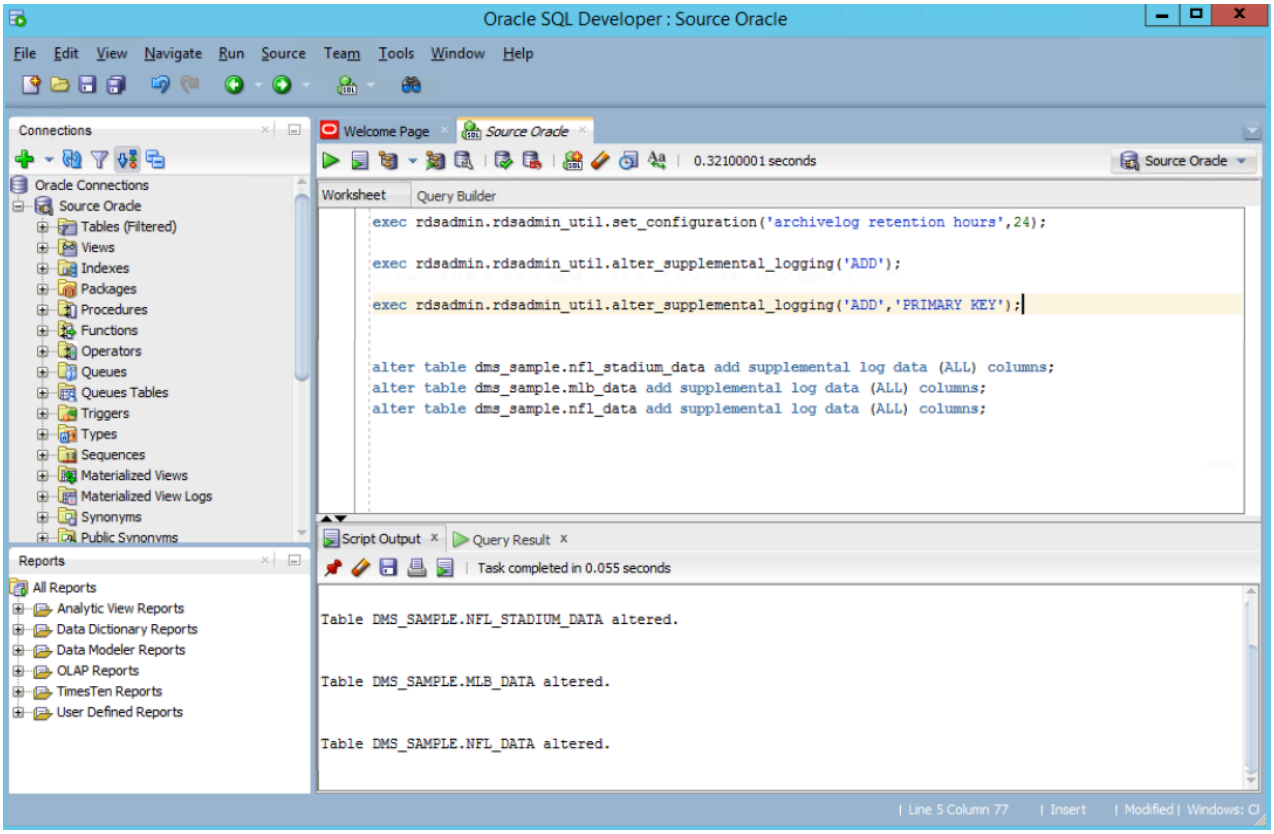
```
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD');
```

12. Run the following query to enable PRIMARY KEY logging for tables that have primary keys:

```
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD','PRIMARY KEY');
```

13. Run the following queries to add supplemental logging for tables that don't have primary keys, use the following command to add supplemental logging:

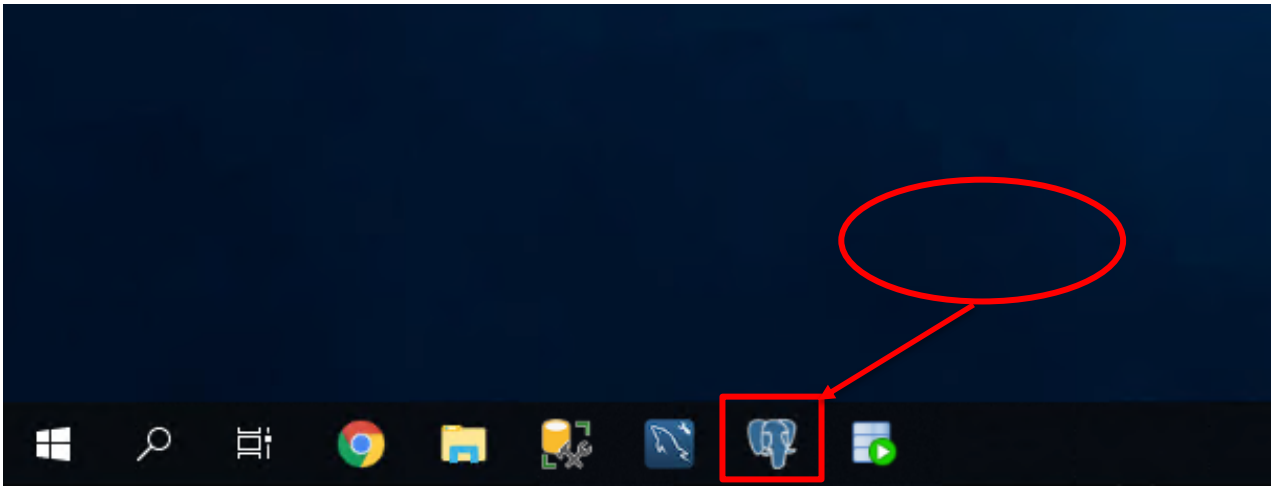
```
alter table dms_sample.nfl_stadium_data add supplemental log data (ALL) columns;
alter table dms_sample.mlb_data add supplemental log data (ALL) columns;
alter table dms_sample.nfl_data add supplemental log data (ALL) columns;
```



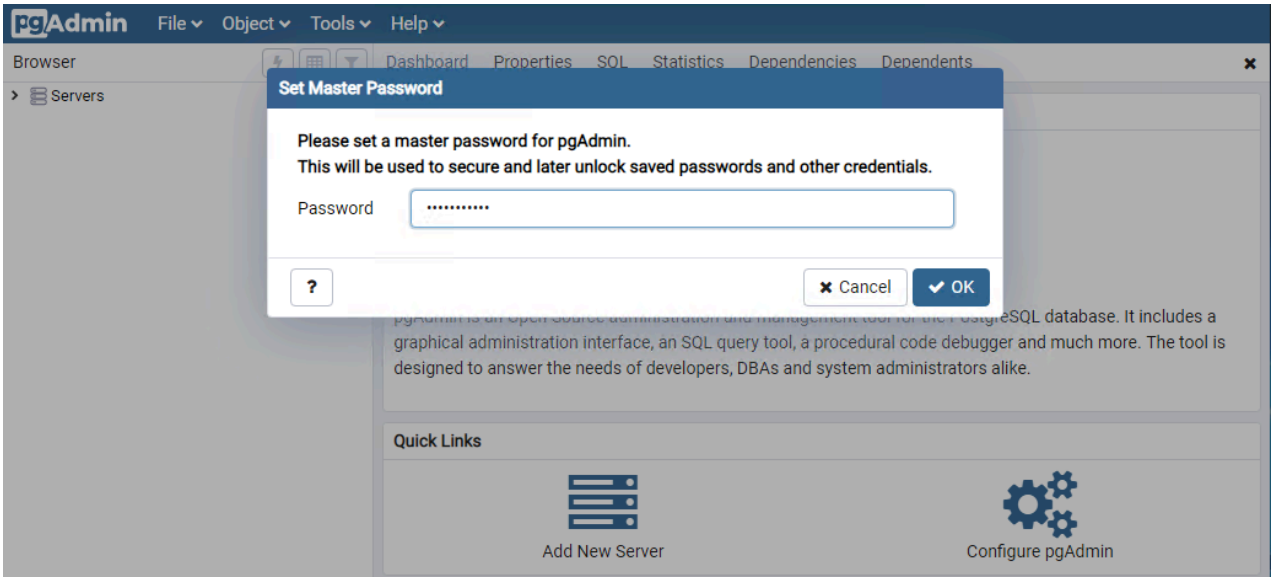
Configure the Target Aurora RDS database for Replication

During the full load process, AWS DMS does not load tables in any particular order, so it might load the child table data before parent table data. As a result, foreign key constraints might be violated if they are enabled. Also, if triggers are present on the target database, they might change data loaded by AWS DMS in unexpected ways.

14. Open **pgAdmin 4** from the **Taskbar** on the EC2 server.

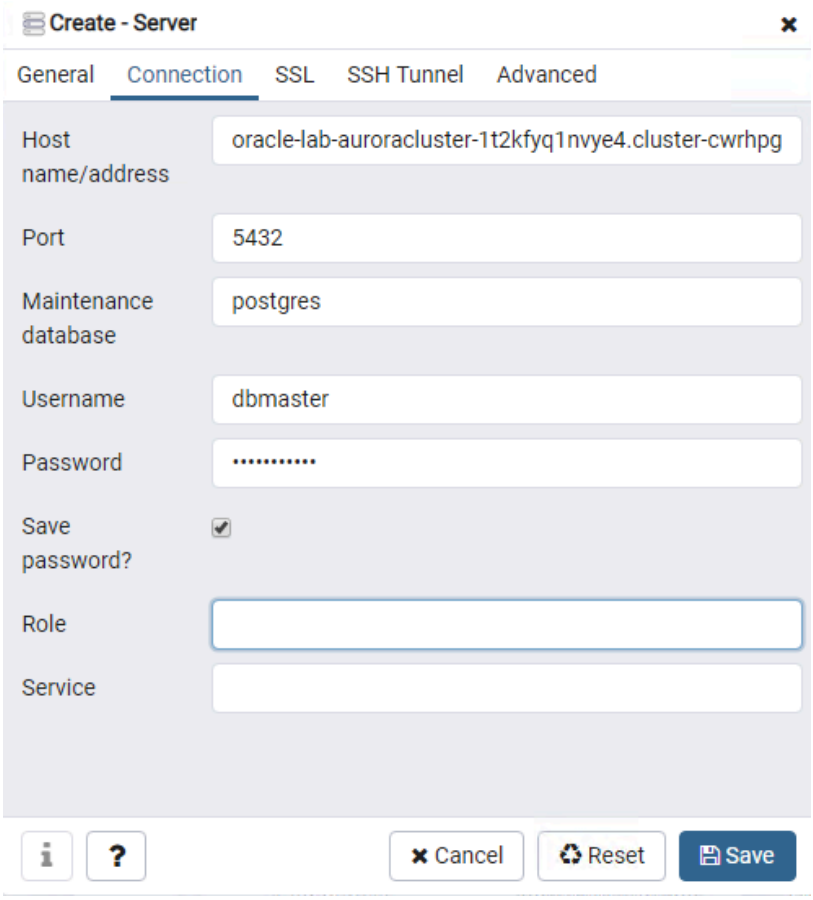


15. You may be prompted to set a **Master Password.** Enter **dbmaster123**, then click, **OK**.

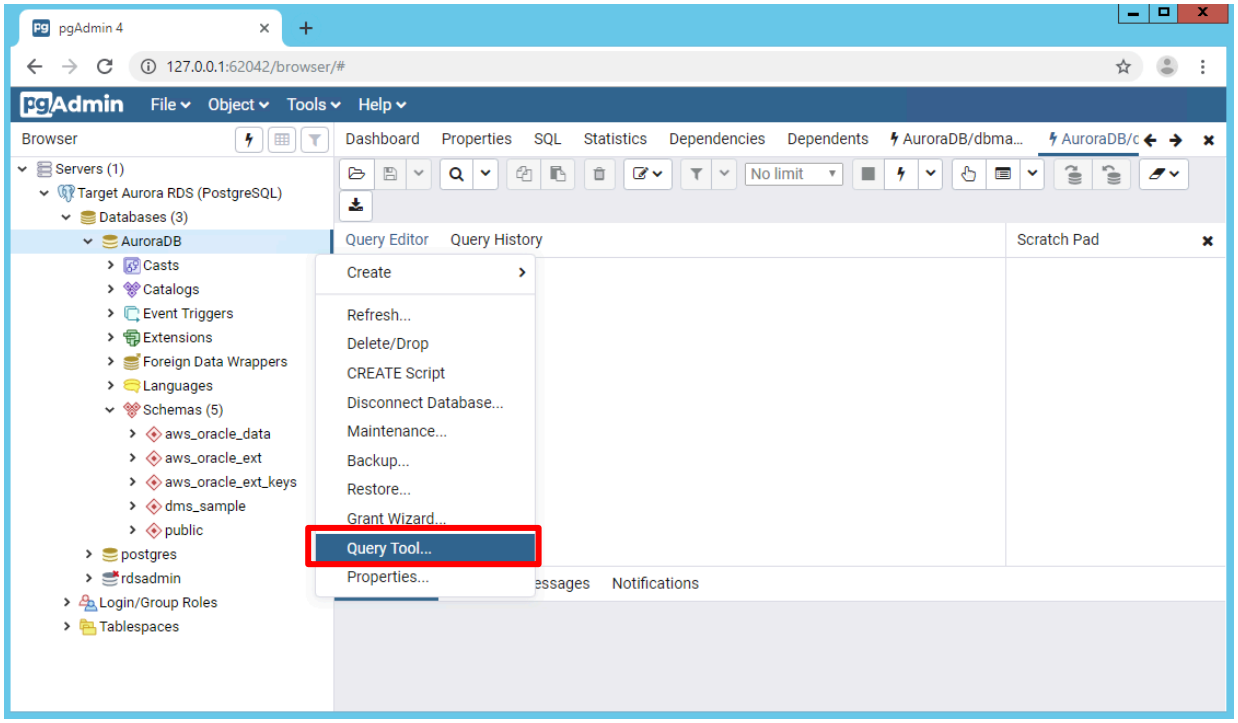


16. Click on the **Add New Server** icon, and enter the following values. Then, press **Save**.

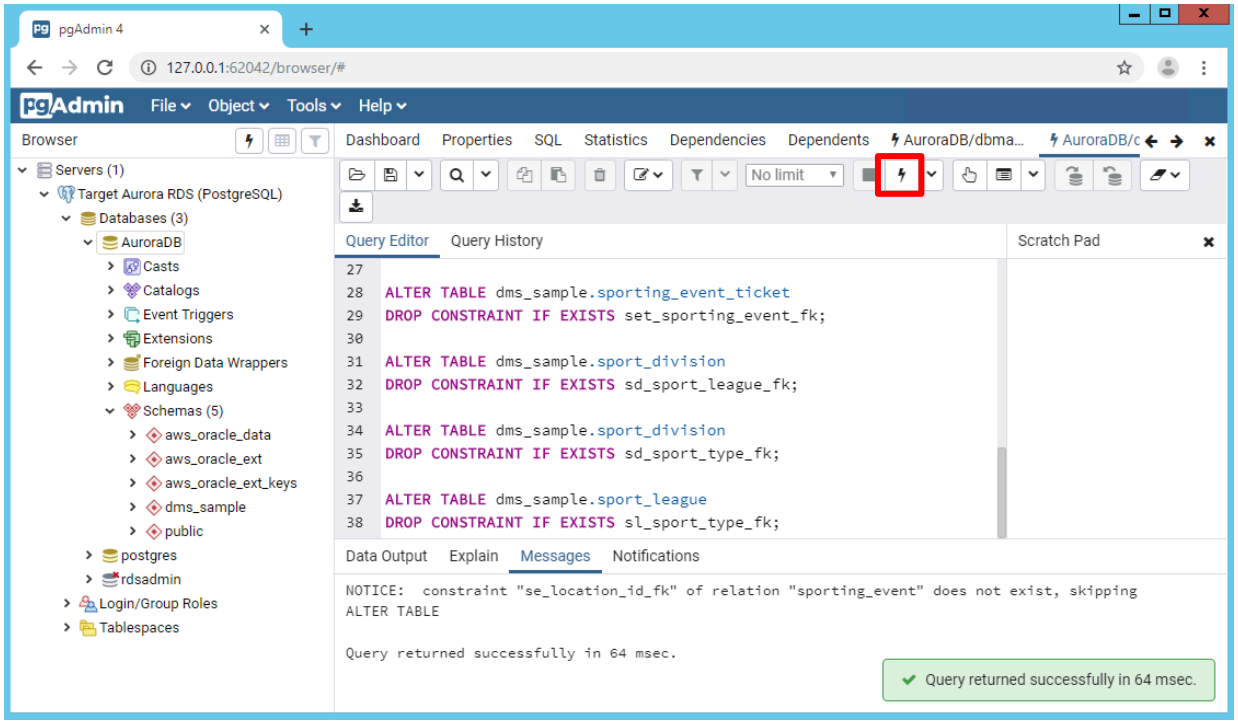
Parameter	Value
General -> Name	Target Aurora RDS (PostgreSQL)
Connection -> Host Name/Address	**< TargetAuroraPostgreSQLEndpoint >**
Connection -> Port	5432
Connection -> Username	dbmaster
Connection -> Password	dbmaster123
Connection -> Save Password	Check



17. Right-click on **AuroraDB** database from left-hand menu, and then select **Query Tools**.



18. In this step you are going to drop the foreign key constraints from the target database:
    a. Open **DropConstraintsPostgreSQL.sql** inside the **DMS Workshop\Scripts** folder in Notepad.
    b. Copy the content of the file to the **Query Editor** in **pgAdmin 4**.
    c. **Execute** the script.



Create Replication Instance

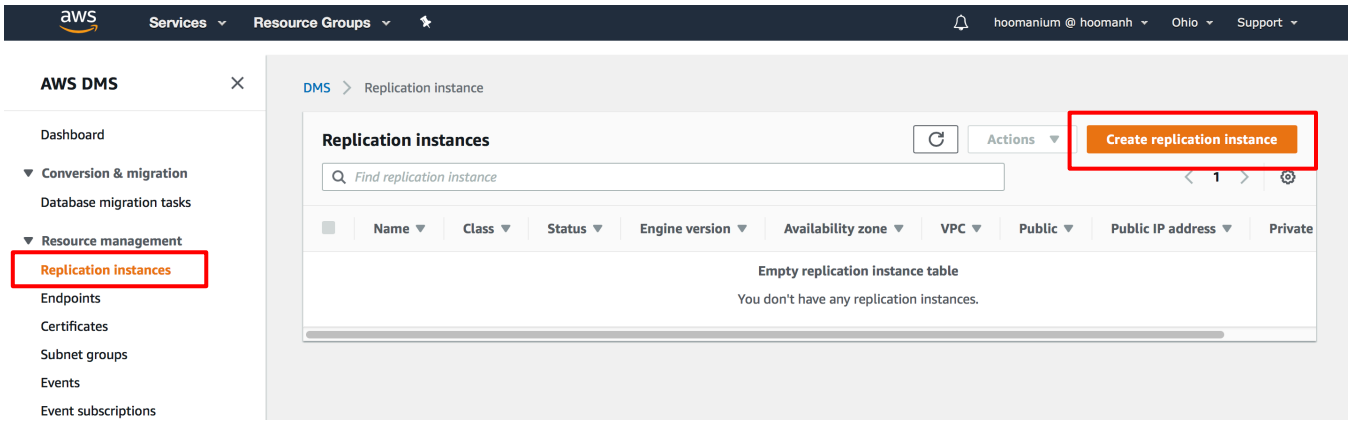
The following illustration shows a high-level view of the migration process.

An AWS DMS replication instance performs the actual data migration between source and target. The replication instance also caches the transaction logs during the migration. The amount of CPU and memory capacity a replication instance has influences the overall time that is required for the migration.

19. Click on https://console.aws.amazon.com/dms/ to launch the Database Migration Service.

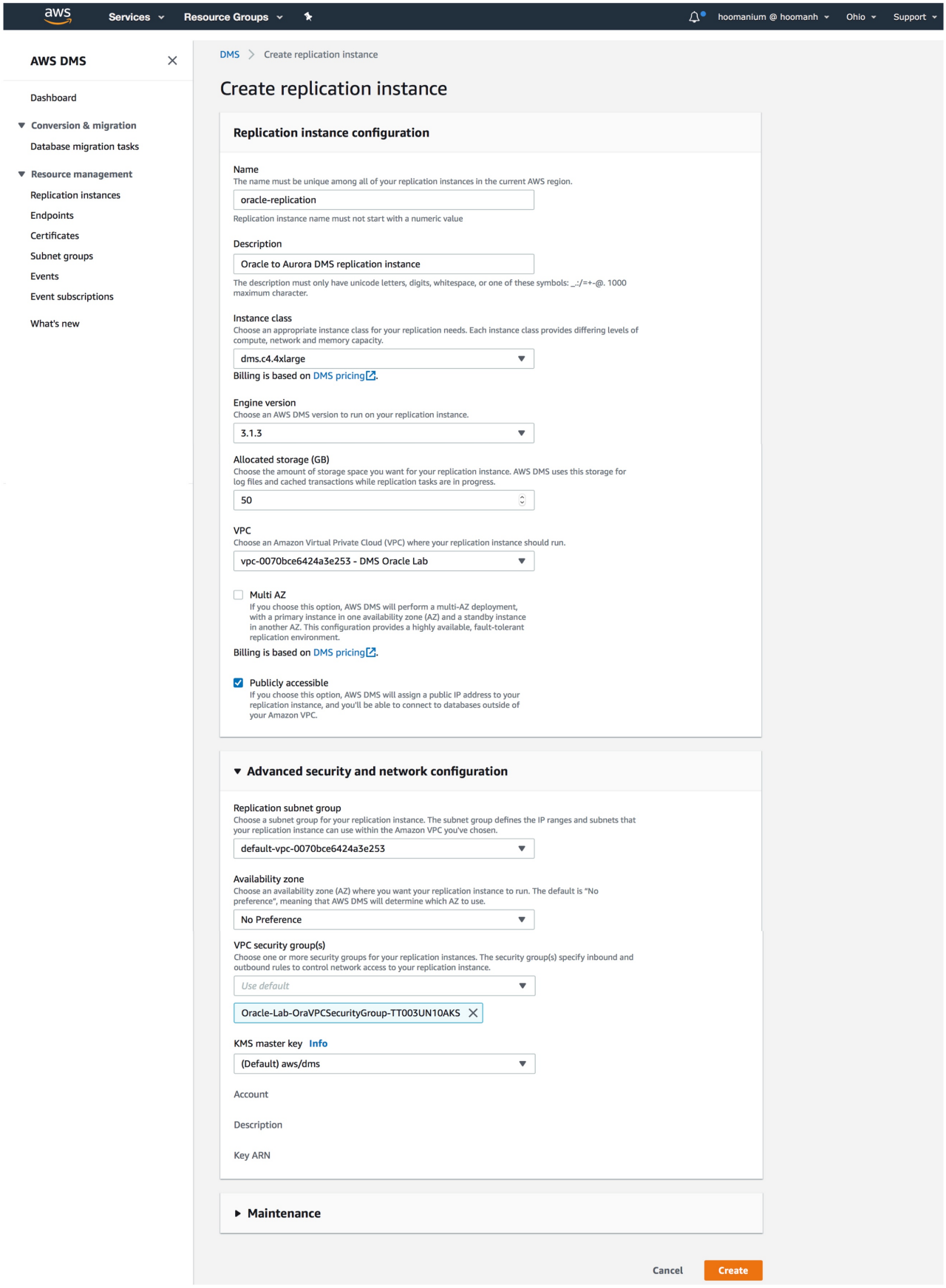
20. On the left-hand menu click on **Replication Instances**. This will launch the Replication instance screen.

21. Click on the **Create replication instance** button on the top right side.



22. Enter the following information for the **Replication Instance**. Then, click on the **Create** button.

Parameter	Value
Name	oracle-replication
Description	Oracle to Aurora DMS replication instance
Instance Class	dms.c4.2xlarge
Engine version	Leave the default value
Allocated storage (GB)	50
VPC	**< VPC ID from Environment Setup Step >**
Multi-AZ	No
Publicly accessible	No
Advanced -> VPC Security Group(s)	default

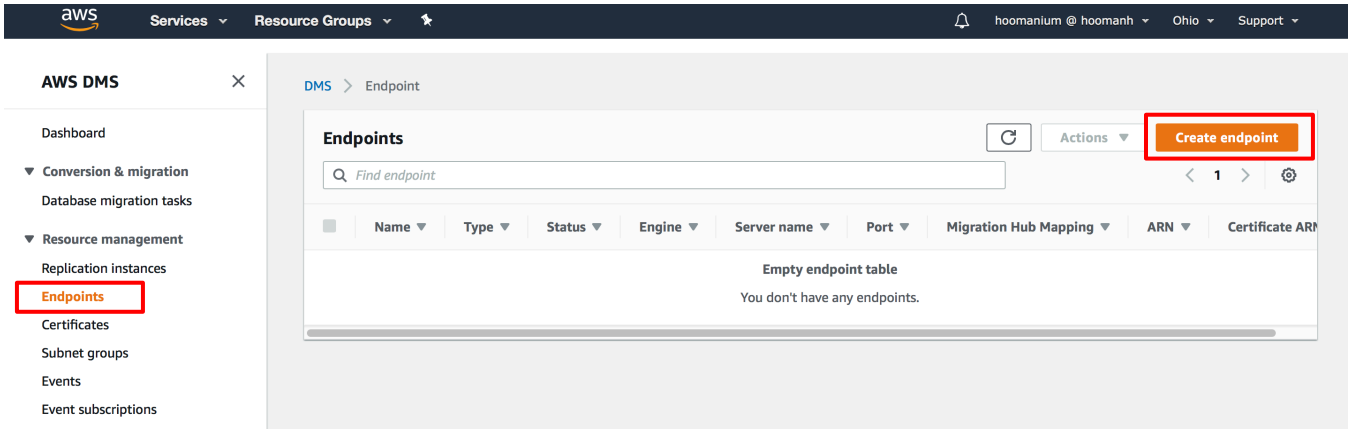


NOTE: Creating replication instance will take several minutes. While waiting for the replication instance to be created, you can specify the source and target database endpoints in the next steps. However, you can test connectivity only after the replication instance has been created, because the replication instance is used in the connection.

Create Source and Target Endpoints

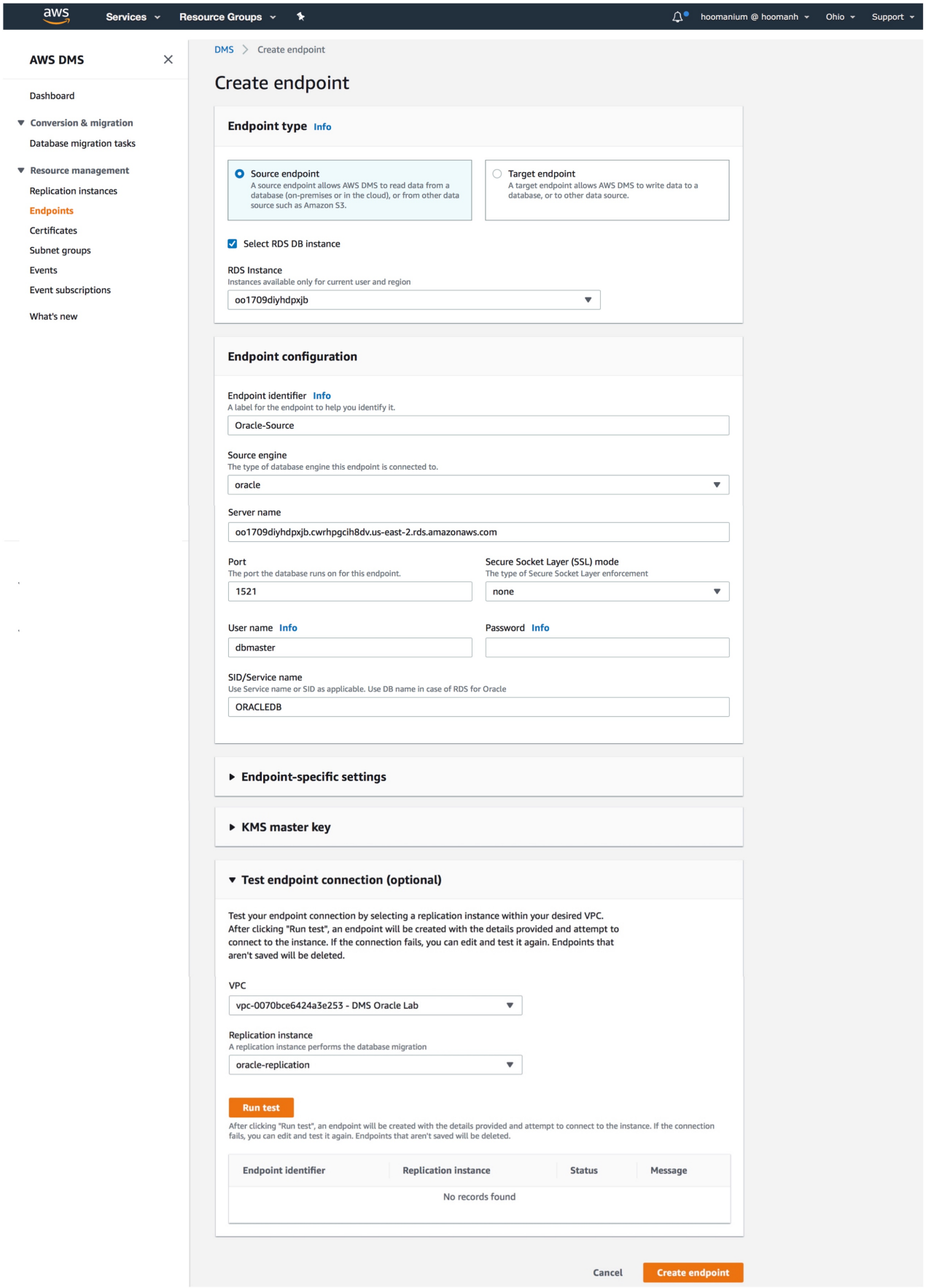
Now that you have a replication instance, you need to create source and target endpoints for the sample database.

23. Click on the **Endpoints** link on the left, and then click on **Create endpoint** on the top right corner.



24. Enter the following information to create an endpoint for the source **dms_sample** database (continued on the next page):

Parameter	Value
Endpoint Type	Source endpoint
Select RDS DB instance	Check
RDS Instance	**<StackName>-SourceOracleDB**
Endpoint Identifier	oracle-source
Source Engine	oracle
Server Name	**< SourceOracleEndpoint >**
Port	1521
SSL Mode	none
User Name	dbmaster
Password	dbmaster123
SID/Service Namee	ORACLEDB
Test endpoint connection -> VPC	**< VPC ID from Environment Setup Step >**
Replication Instance	oracle-replication

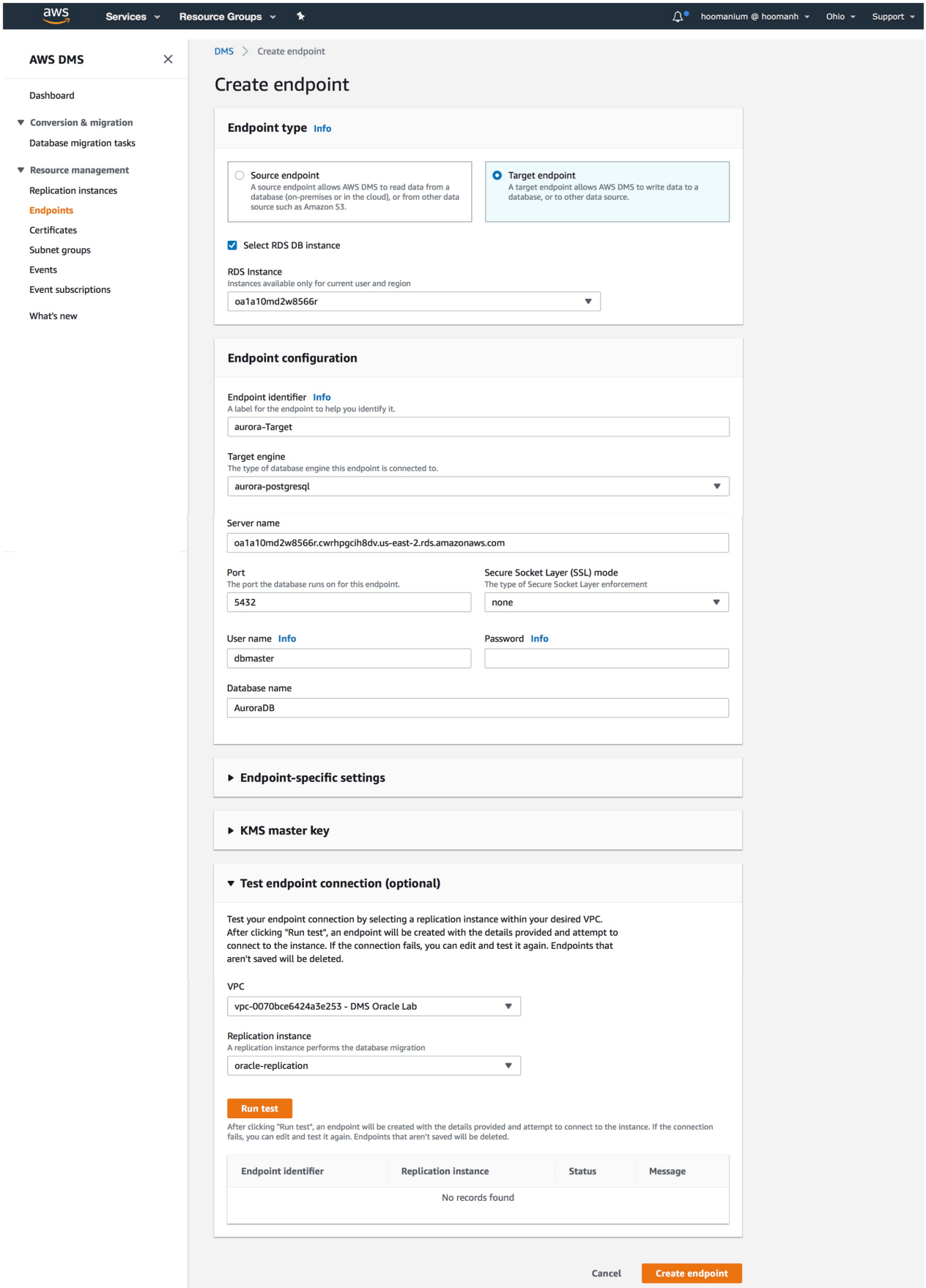


25. Once the information has been entered, click **Run Test**. When the status turns to **successful**, click **Create endpoint**.

26. Follow the same steps to create another endpoint for the **Target Aurora RDS Database** using the following values:

Parameter	Value
Endpoint Type	Target endpoint
Select RDS DB instance	Check
RDS Instance	**<StackName>-AuroraPostgreSQLInstance**
Endpoint Identifier	aurora-target
Source Engine	aurora-postgresql
Server Name	**< TargetAuroraPostgreSQLEndpoint >**
Port	1521
SSL Mode	none
User Name	dbmaster
Password	dbmaster123
Database Name	AuroraDB
Test endpoint connection -> VPC	**< VPC ID from Environment Setup Step >**
Replication Instance	oracle-replication

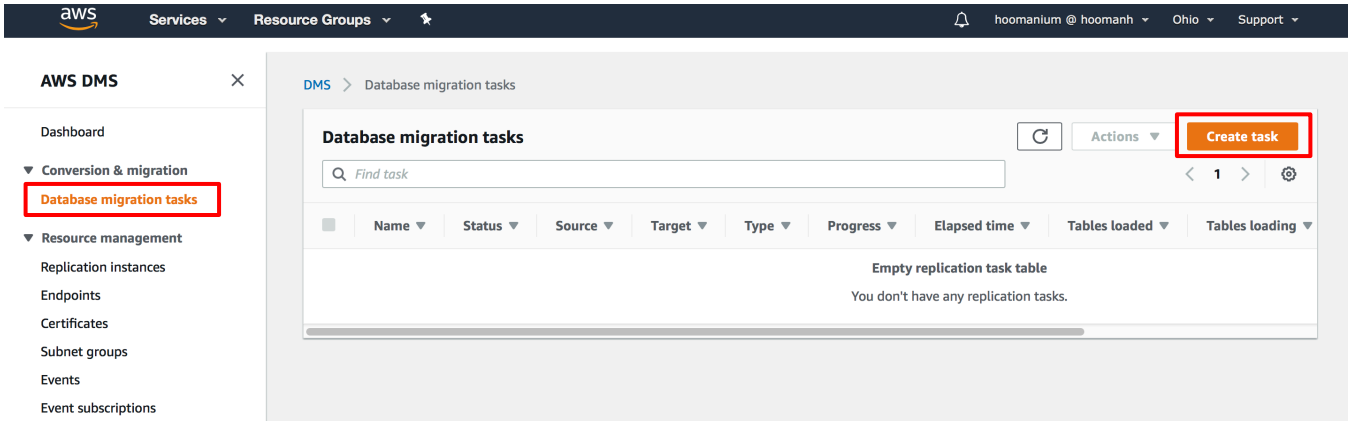
27. Once the information has been entered, click **Run Test**. When the status turns to **successful**, click **Create endpoint**.



Create and Run Your Database Migration Task

AWS DMS uses **Database Migration Task** to migrate the data from source to the target database. In this part of the lab you are going to create two Database Migration Tasks: one for migrating the existing data, and another for capturing data changes on the source database and replicating the changes to the target database.

28. Click on **Database migration tasks** on the left-hand menu, then click on the **Create task** button on the top right corner.



29. Create a data migration task with the following values for migrating the **dms_sample** database.

Parameter	Value
Task identifier	oracle-migration-task
Replication instance	oracle-replication
Source database endpoint	oracle-source
Target database endpoint	aurora-target
Migration type	Migrate existing data
Start task on create	Checked
Target table preparation mode	Do nothing
Include LOB columns in replication	Limited LOB mode
Max LOB size (KB)	32
Enable validation	Unchecked
Enable CloudWatch logs	Checked

30. Expand the **Table mappings** section, and select **Guided UI** for the editing mode

31. Click on **Add new selection rule** button and enter the following values:

Parameter	Value
Schema	DMS_SAMPLE
Table name	%
Action	Include

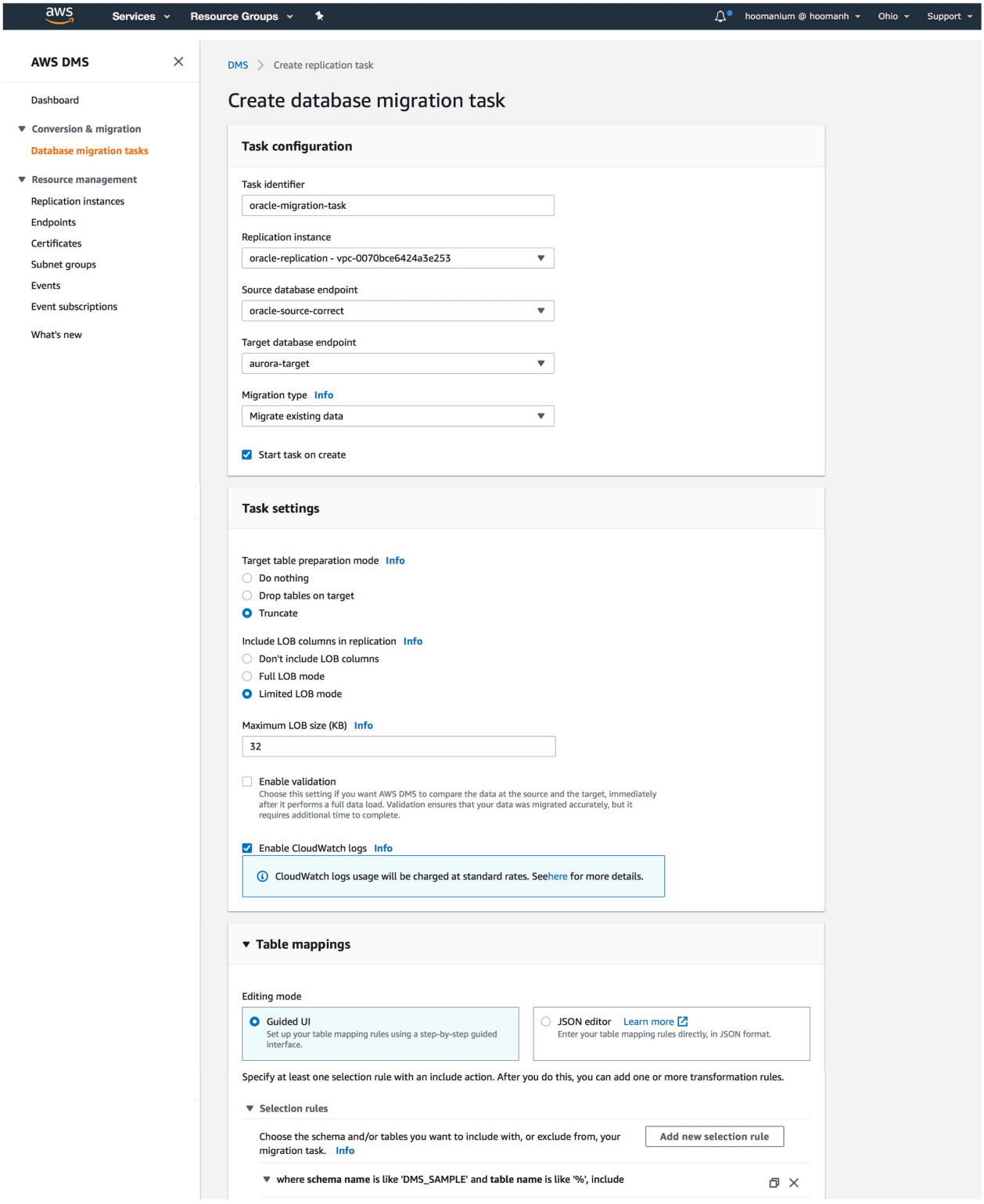
NOTE: If the Create Task screen does not recognize any schemas, make sure to go back to endpoints screen and click on your endpoint. Scroll to the bottom of the page and click on **Refresh Button (⟳)** in the **Schemas** section.
If your schemas still do not show up on the Create Task screen, click on the Guided tab and manually select **DMS_SAMPLE** schema and all tables.

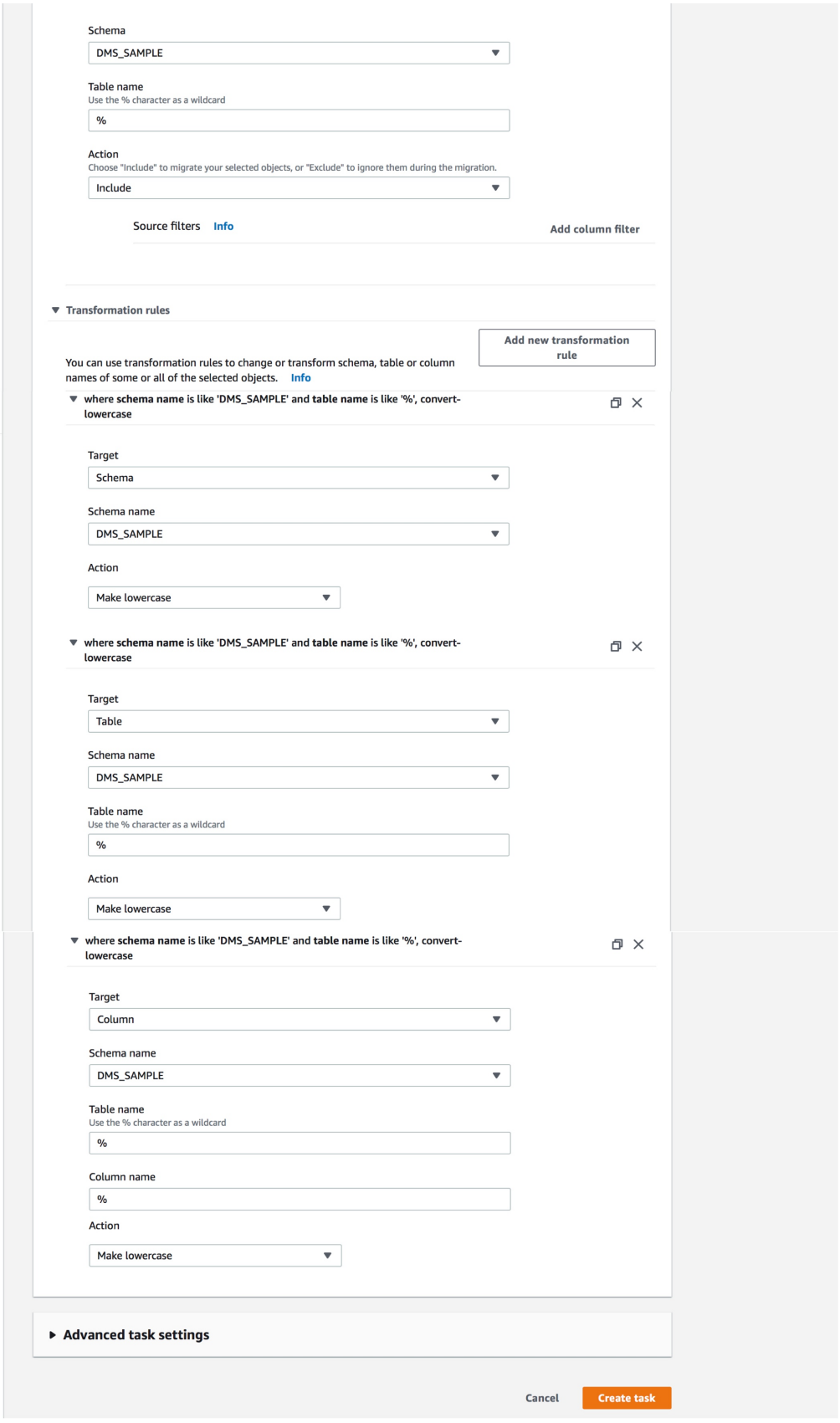
32. Next, expand the **Transformation rules** section, and click on **Add new transformation rule**. Then, create the following rules:

Parameter	Value
Target	Schema
Schema Name	DMS_SAMPLE
Action	Make lowercase

Parameter	Value
Target	Table
Schema Name	DMS_SAMPLE
Table Name	%
Action	Make lowercase

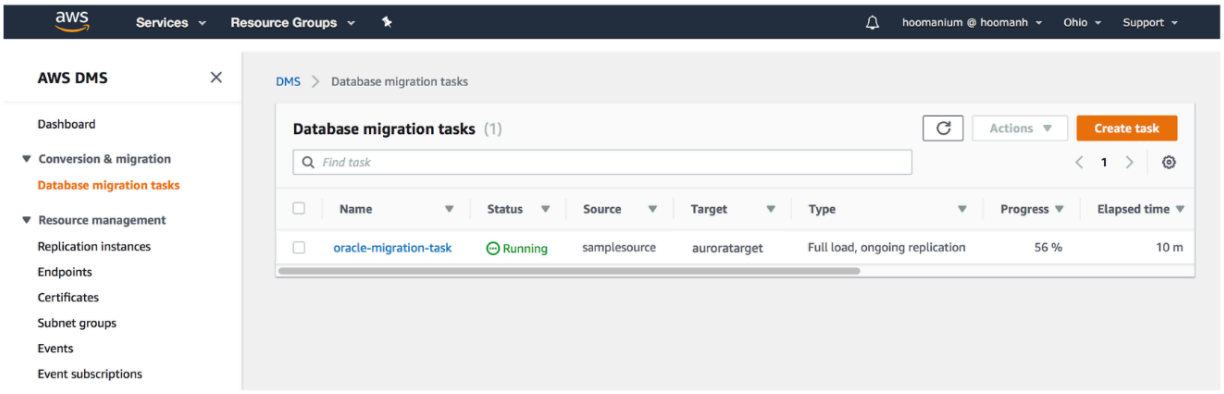
Parameter	Value
Target	Column
Schema Name	DMS_SAMPLE
Table Name	%
Column Name	%
Action	Make lowercase





33. After entering the values click on **Create task**.

34. At this point, the task should start migrating data from the source Oracle database to the Amazon Aurora RDS instance.



35. As the rows are being transferred, you can monitor the task progress:

    d. Click on your task **(oracle-migration-task)** and scroll to the **Table statistics** section to view the table statistics to see how many rows have been moved.
    e. If there is an error, the status color changes from green to reed. Click on **View logs** link for the logs to debug.

36. When the full load is complete, remember to enable the foreign key constraints.

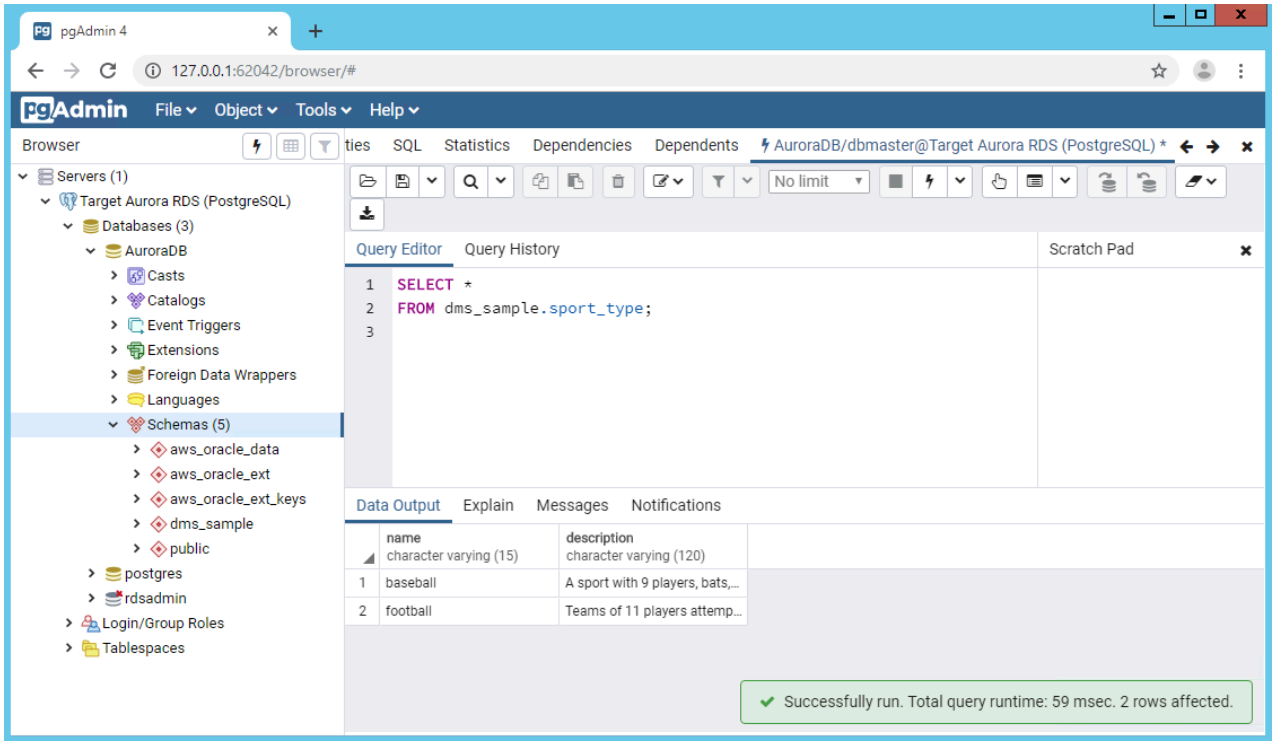
Inspect the Content of Target Database

37. If you already disconnected from the EC2 Server, follow steps 1, 2, and 3 at the top of this lab to connect (RDP) to your EC2 instance.

38. Open **pgAdmin4** from within the EC2 server, and then connect to the Target Aurora RDS (PostgreSQL) database connection that you created earlier.

39. Inspect the migrated data, by querying one of the tables in the target database. For example, the following query should return a table with two rows:

```
SELECT *
FROM dms_sample.sport_type;
```



Baseball, and football are the only two sports that are currently listed in this table.

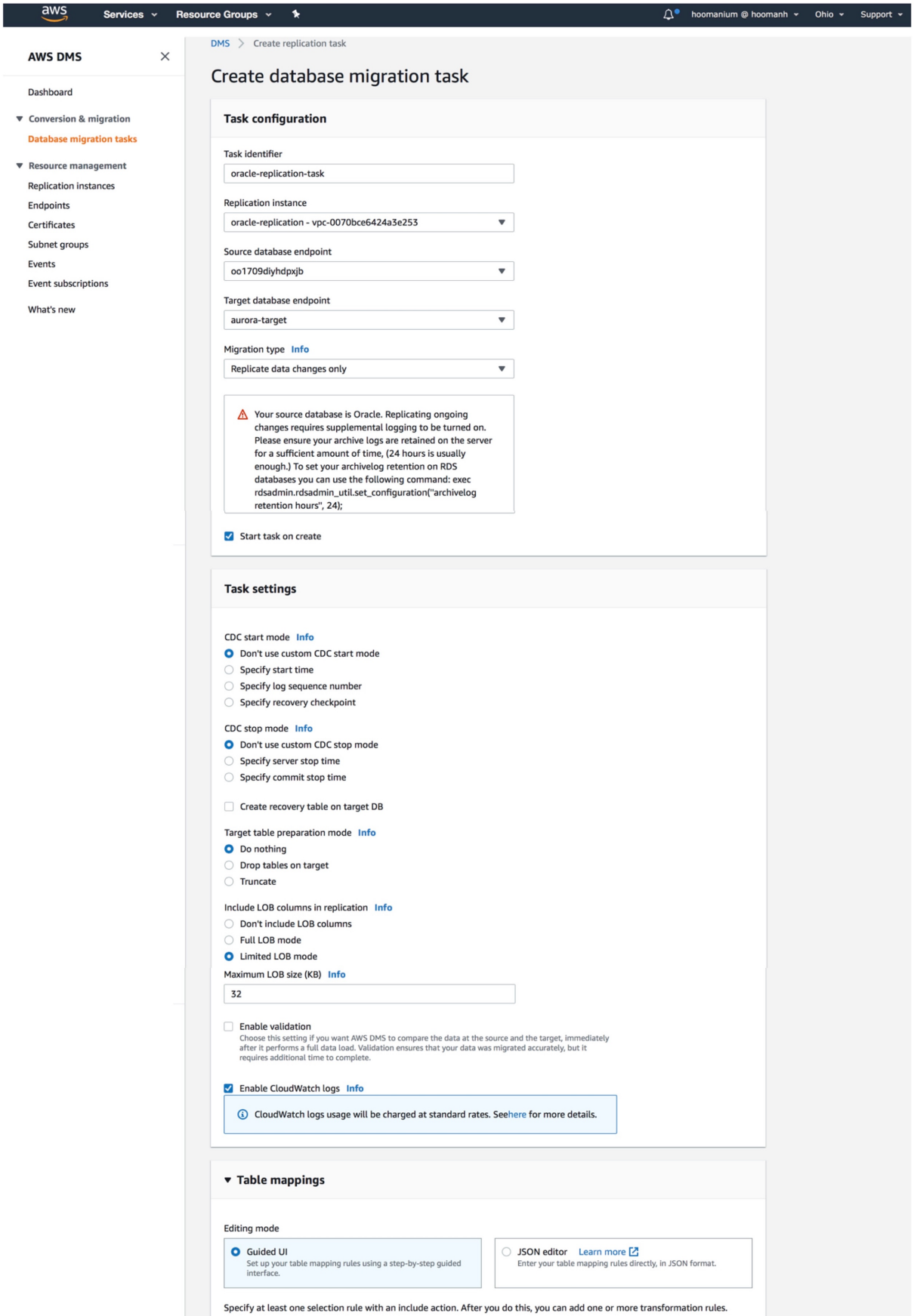
Capture and Replicate Data Changes

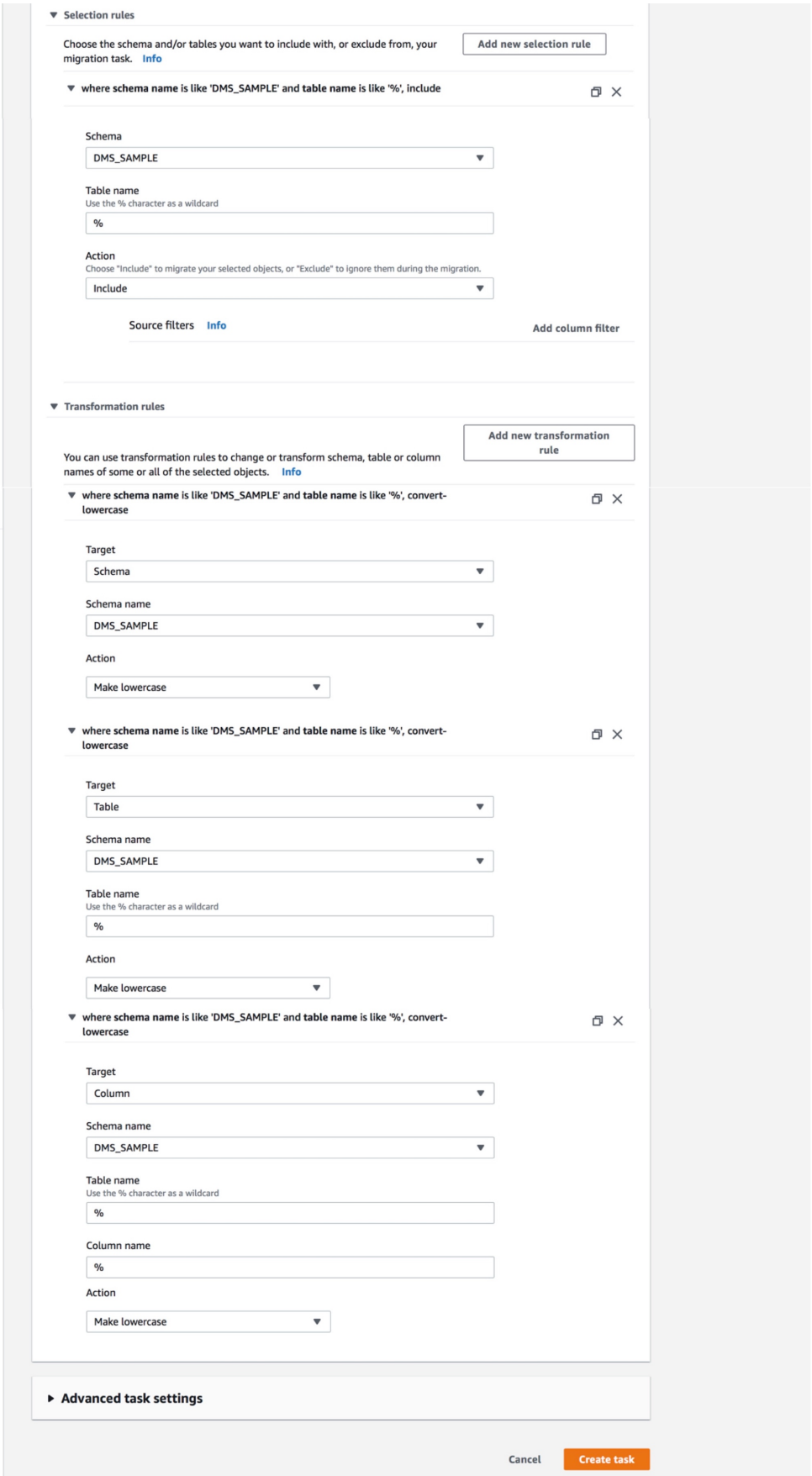
40. Create another **Data Migration Task** with the following values for capturing data changes to the source Oracle database, and replicating the changes to the target Aurora RDS instance.

Parameter	Value
Task identifier	oracle-replication-task
Replication instance	oracle-replication
Source database endpoint	oracle-source
Target database endpoint	aurora-target
Migration type	Replicate data changes only
Start task on create	Checked
CDC stop mode	Don’t use custom CDC stop mode
Target table preparation mode	Do nothing
Stop task after full load completes	Don’t stop
Include LOB columns in replication	Limited LOB mode
Max LOB size (KB)	32
Enable validation	Unchecked
Enable CloudWatch logs	Checked

41. Expand the **Table mappings** section, and select **Guided UI** for the editing mode.

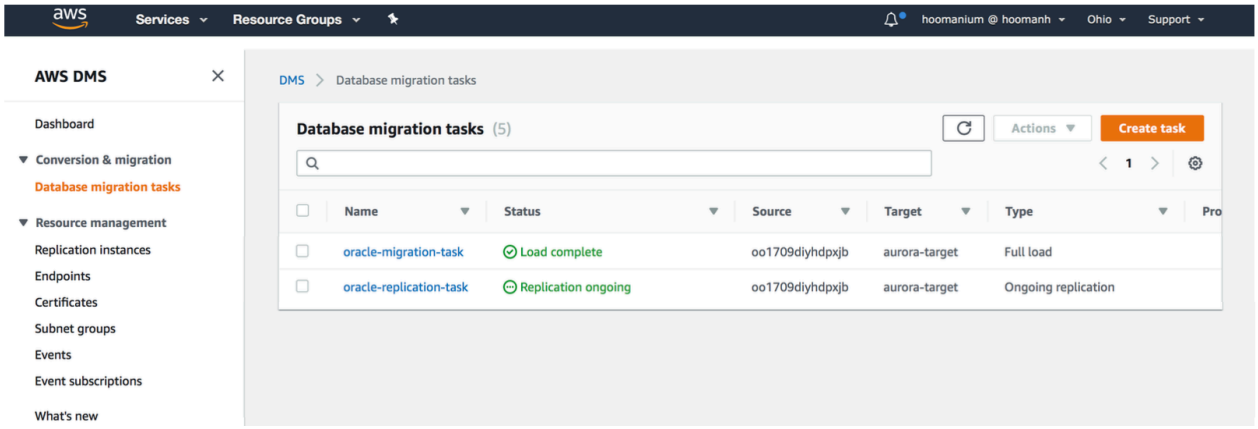
42. Add the same **Selection,** and **Transformation** rules as specified in steps 28, and 29.





43. After entering the values click on **Create task**.

44. At this point, the new migration task is ready to replicate ongoing data changes from the source Oracle RDS to the Amazon Aurora RDS (PostgreSQL) database.



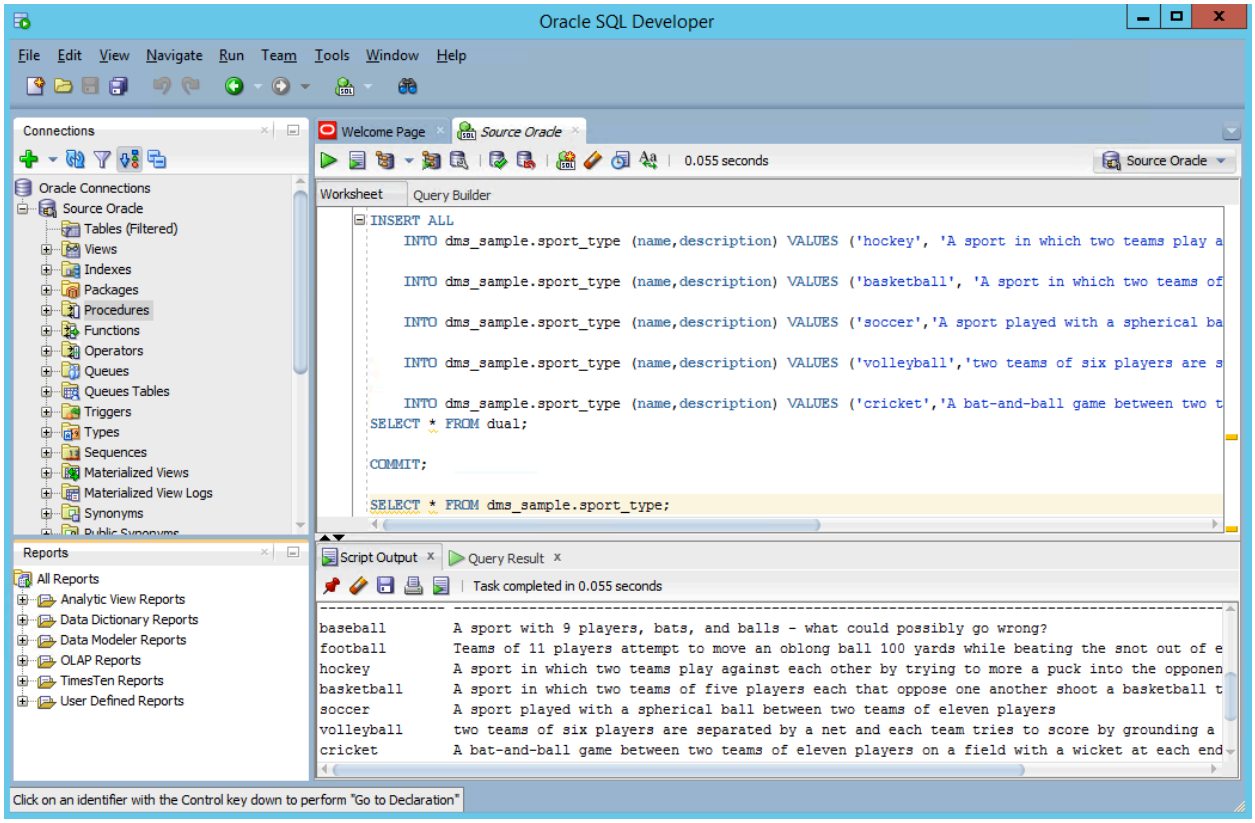
Replicating Changes from Source to the Target Database

Now you are going to simulate a transaction to the source database by updating the **sport_type** table. The Database Migration Service will automatically detect and replicate these changes to the target database.

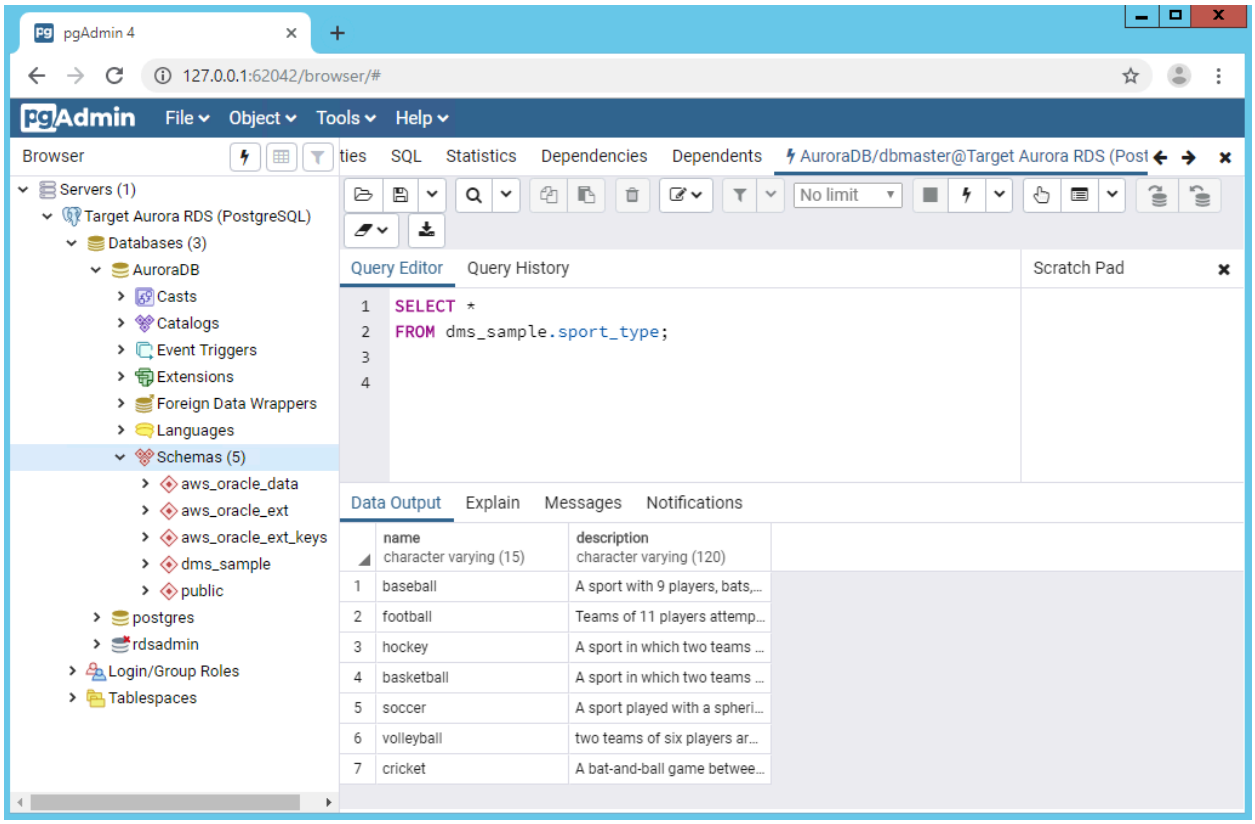
45. Use **Oracle SQL Developer** connect to the source Oracle RDS instance (described in steps 4, and 5.)

46. Open a **New Query** window and **execute** the following statement to insert 5 new sports into the **sport_type** table:

```
INSERT ALL

INTO dms_sample.sport_type (name,description) VALUES ('hockey', 'A sport in which two teams play against each other by trying to more a puck into the opponents goal using a hockey stick')

INTO dms_sample.sport_type (name,description) VALUES ('basketball', 'A sport in which two teams of five players each that oppose one another shoot a basketball through the defenders hoop')

INTO dms_sample.sport_type (name,description) VALUES ('soccer','A sport played with a spherical ball between two teams of eleven players')

INTO dms_sample.sport_type (name,description) VALUES ('volleyball','two teams of six players are separated by a net and each team tries to score by grounding a ball on the others court')

INTO dms_sample.sport_type (name,description) VALUES ('cricket','A bat-and-ball game between two teams of eleven players on a field with a wicket at each end')

SELECT * FROM dual;

COMMIT;

SELECT * FROM dms_sample.sport_type;
```



47. Repeat steps 35 and 36 to inspect the content of **sport_type** table in the target database.



Notice the new records for: basketball, cricket, hokey, soccer, and volleyball that you added to the **sports_type** table in the source database have been replicated to your **dms_reporting** database. You can further investigate the number of inserts, deletes, updates, and DDLs by viewing the **Table statistics** of your **Database migration tasks** in AWS console.

The AWS DMS task keeps the target Aurora PostgreSQL database up to date with source database changes. AWS DMS keeps all the tables in the task up to date until it's time to implement the application migration. The latency is close to zero when the target has caught up to the source.

Summary

The first part of the lab demonstrated how easy it is to migrate the data from an Oracle database running on an Amazon RDS instance to an Amazon Aurora (PostgreSQL) instance using the AWS Database Migration Service (DMS). Similarly, you observed how DMS automatically replicates new transactions on the source to target database.

You can follow the same steps to migrate SQL Server and Oracle workloads to other RDS engines including PostgreSQL and MySQL.