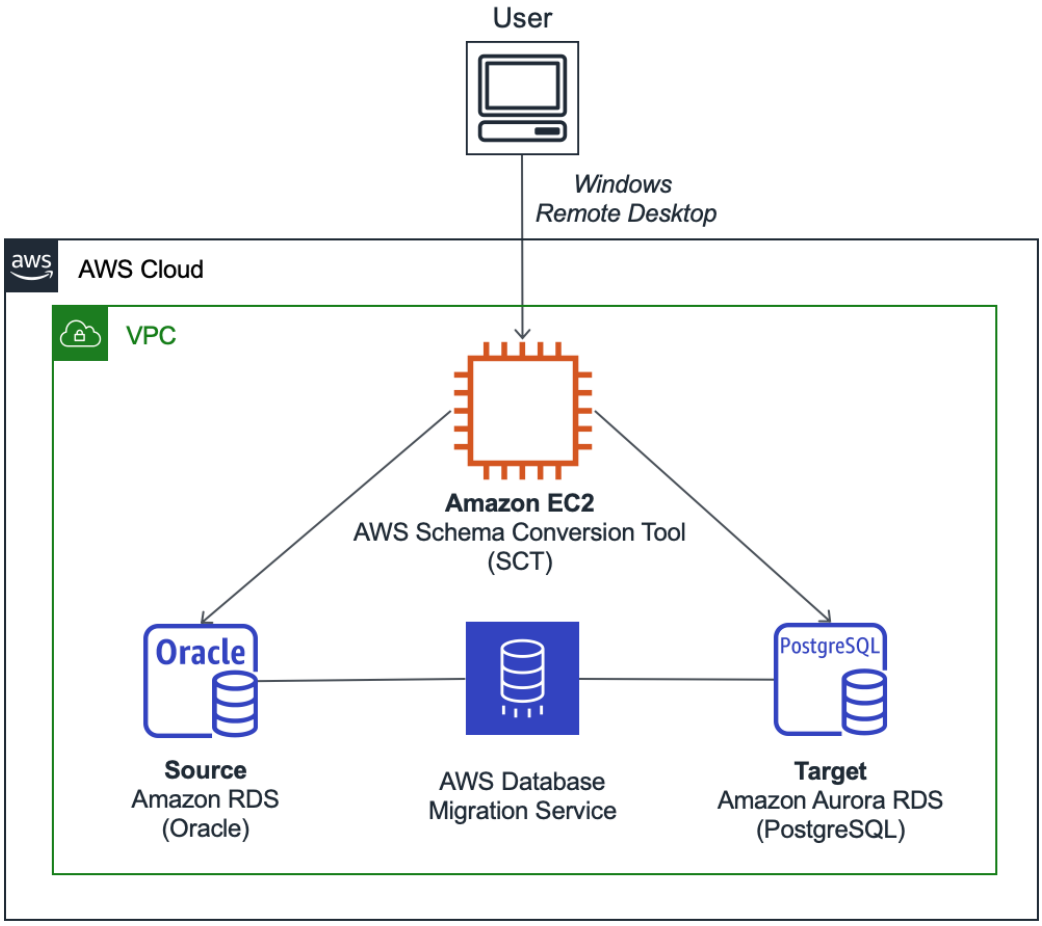
ORACLE TO AMAZON AURORA (POSTGRESQL)

Now that you have completed setting up the workshop enviornment, you are ready to migrate a sample data base.

This step-by-step guide demonstrates how you can use AWS Database Migration Service (DMS) and AWS Schema Conversion Tool (AWS SCT) to migrate data from an Oracle database to Amazon Aurora (PostgreSQL). Additionally, you will use AWS DMS to continually replicate database changes from the source database to the target database.

The environment for this lab consists of:

- An Amazon 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.

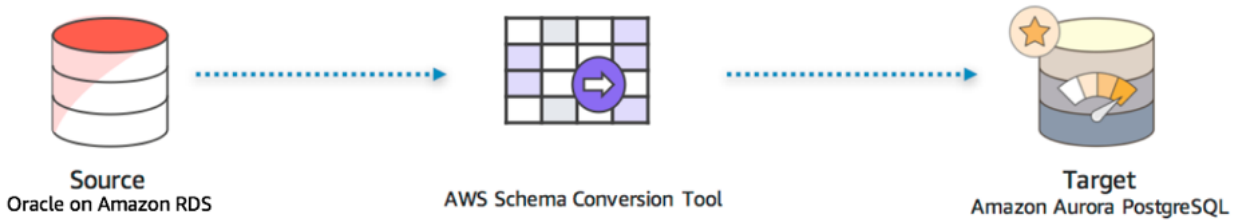


Note

Before proceeding further, make sure you have completed the instructions in the Getting Started section that preceeded this chapter.

PART 1: SCHEMA CONVERSION

This section demonstrates how to use the AWS Schema Conversion Tool for converting an Oracle database schema to an Amazon Aurora (PostgreSQL) database. Additionally, you will observe how AWS SCT helps you spot the differences between the two dialects; and, provides you with tips about how you can modify procedural code when needed to successfully migrate all database objects.

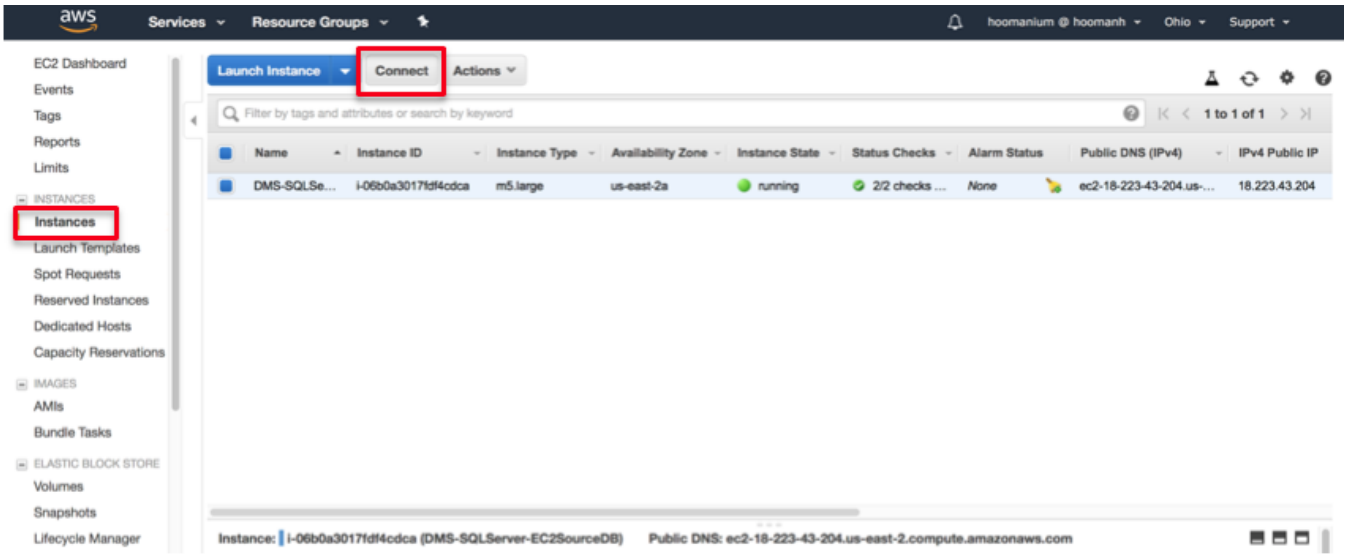


In this exercise, you perform the following tasks:

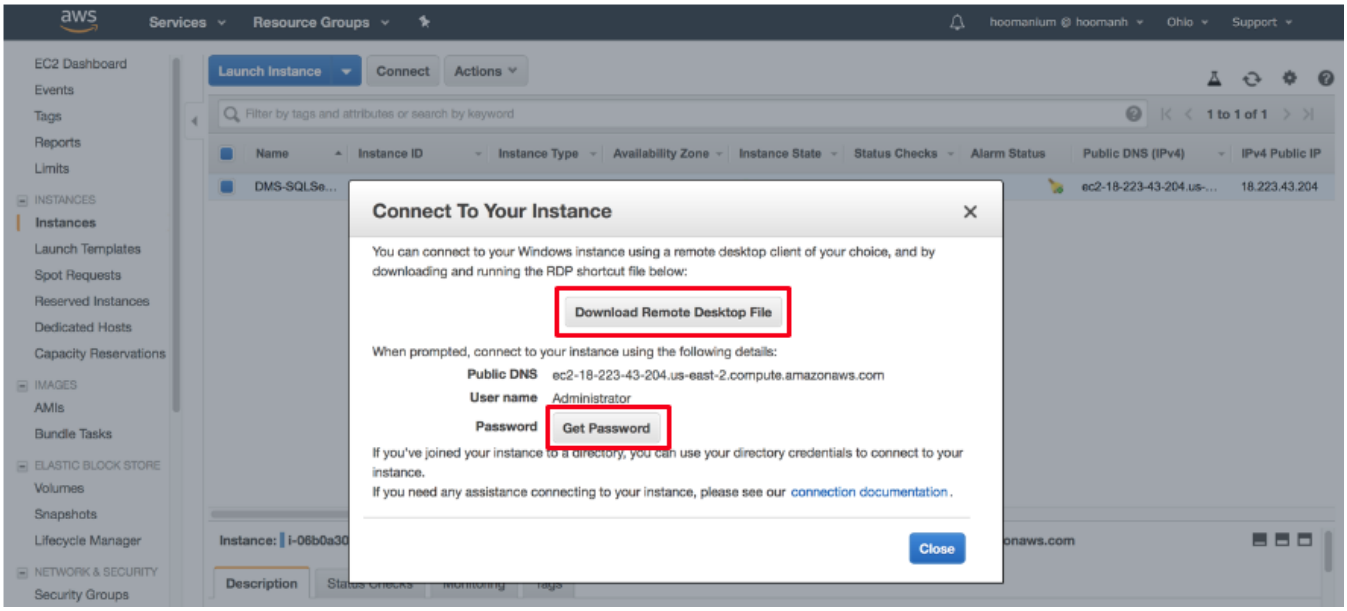
- Connect to the EC2 Instance
- Install the AWS Schema Conversion Tool (AWS SCT)
- Create a Database Migration Project
- Convert the Schema

CONNECT TO THE EC2 INSTANCE

1. Go to the AWS EC2 console and click on **Instances** in the left column.
2. Select the instance with the name **<StackName>-EC2Instance** and then click the **Connect** button.

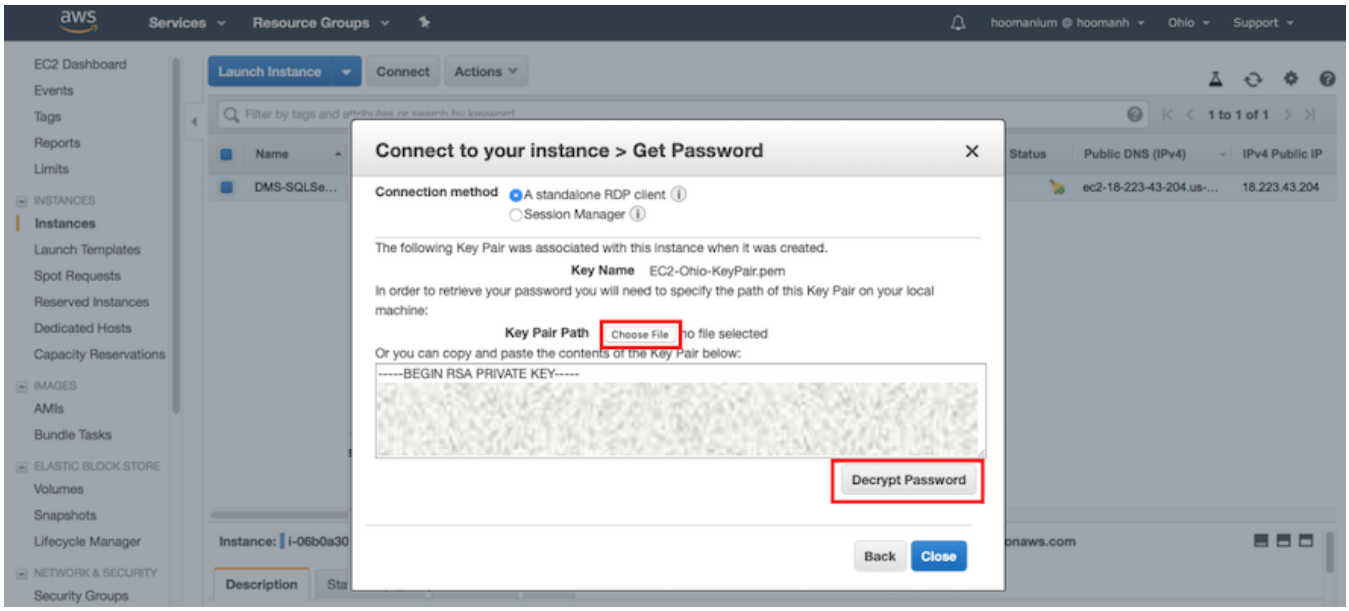


3. Click the **Get Password** button.



4. Upload the Key Pair file that you downloaded earlier.

5. Click on Decrypt Password.



6. Take note of the EC2 console generated administrator password.

7. Click on Download Remote Desktop File to download the RDP file to access this EC2 instance.

8. Connect to the EC2 instance using an RDP client.

INSTALL THE AWS SCHEMA CONVERSION TOOL (AWS SCT)

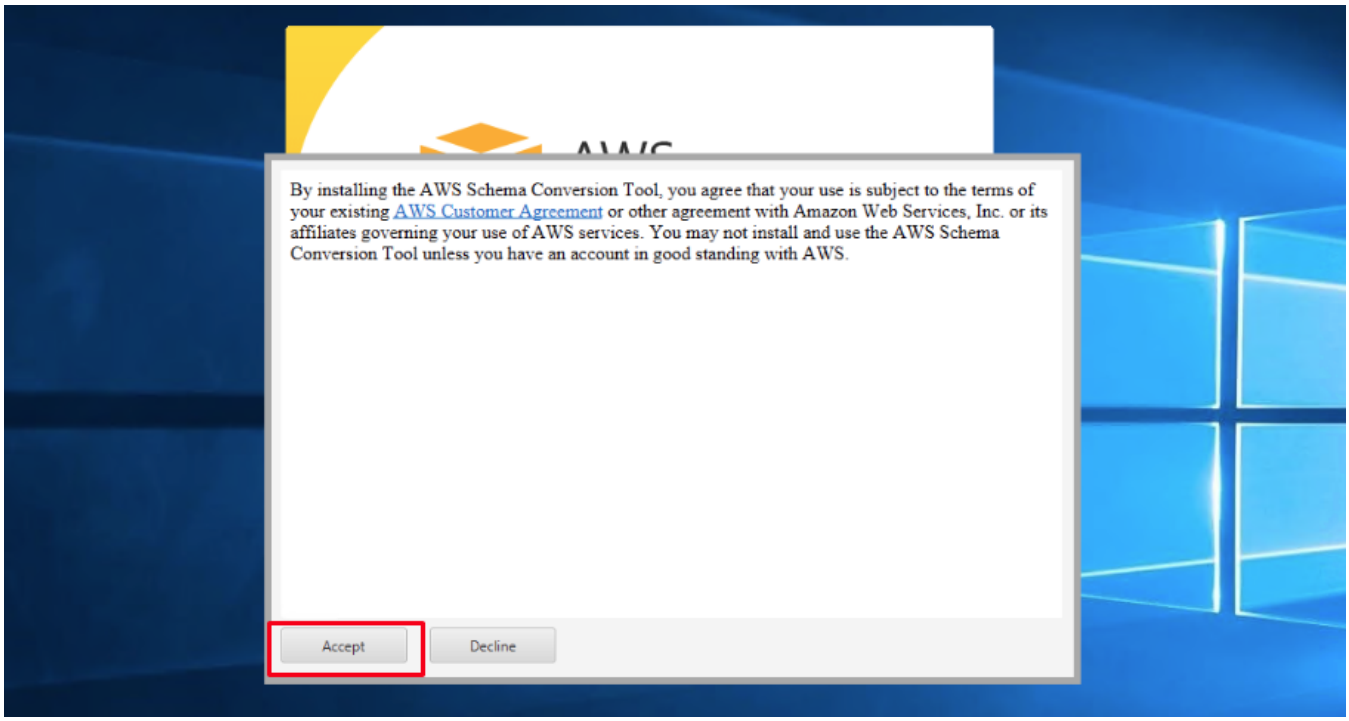
Now that you are connected to the EC2 instance, you are going to install the AWS Schema Conversion tool on the server. Downloading the file and installing it will give you the latest version of the AWS Schema Conversion Tool.

9. On the EC2 server, open the **DMS Workshop** folder that is on the **Desktop**. Then, double-click on **AWS Schema Conversion Tool Download** to get the latest version of the software.
10. When the download is complete, unzip the content and install the AWS Schema Conversion Tool.

> **Note**
>
> When the installer is complete the installation dialog will disappear. There is no other notification.

11. Once the installation is complete, go to the **Start Menu** and launch the AWS Schema Conversion Tool.

12. **Accept** the terms and Conditions.

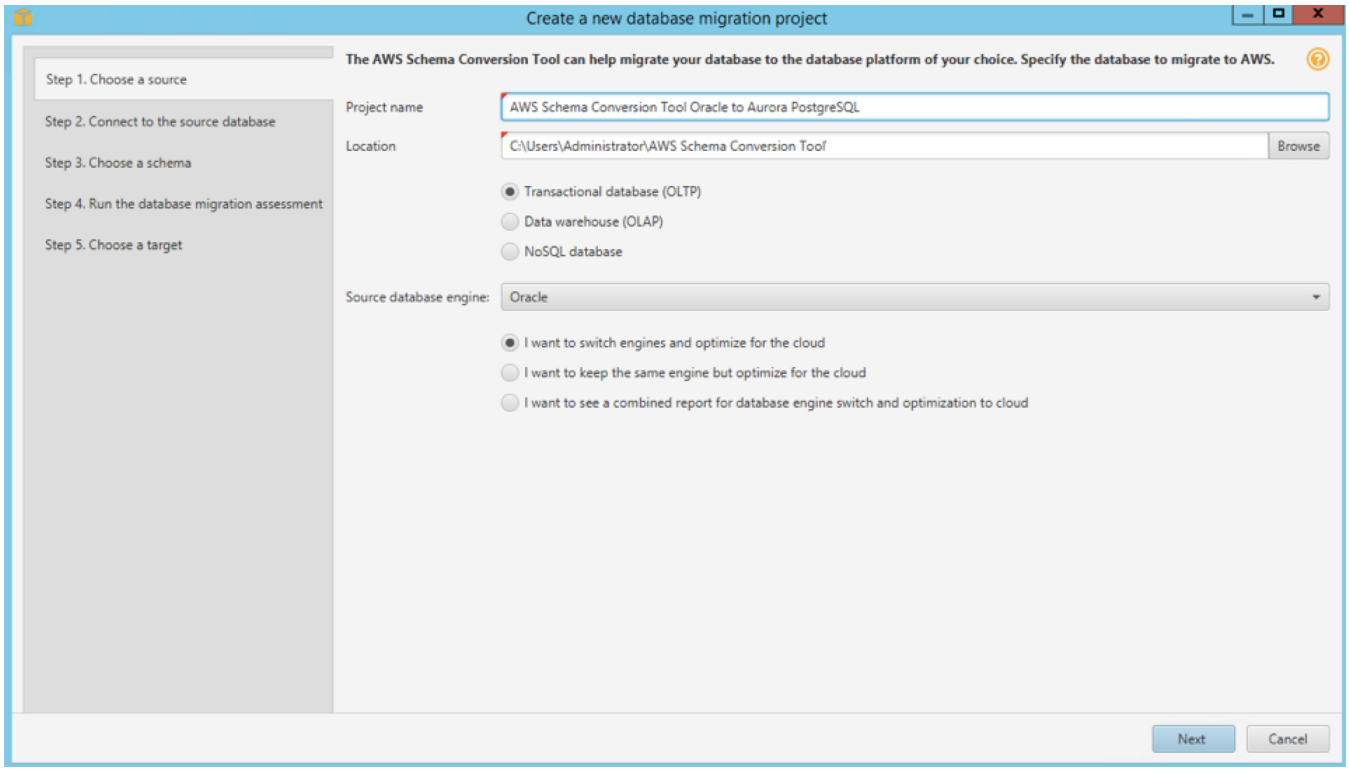


CREATE A DATABASE MIGRATION PROJECT

Now that you have installed the AWS Schema Conversion Tool, the next step is to create a Database Migration Project using the tool.

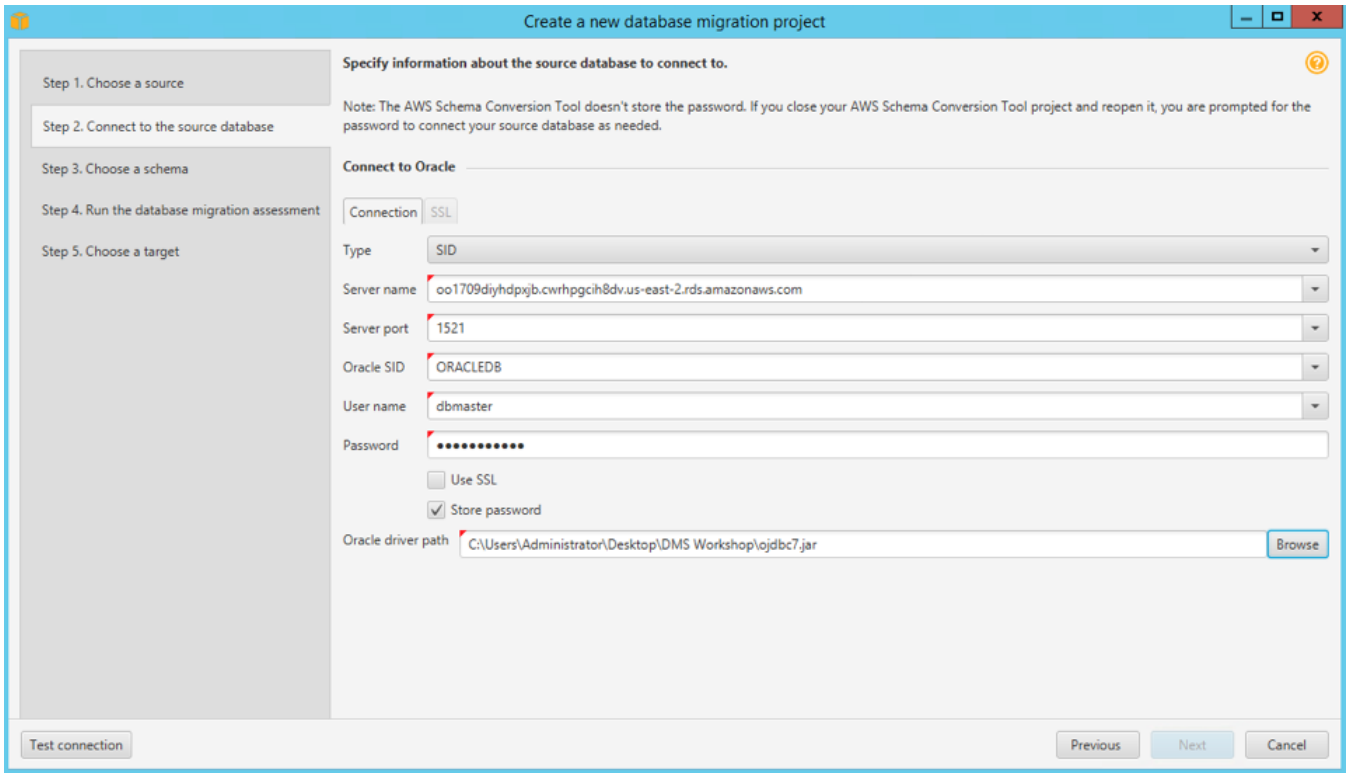
13. Within the Schema Conversion Tool, enter the following values into the form and then click **Next**.

Parameter	Value
Project Name	AWS Schema Conversion Tool Oracle to Aurora PostgreSQL
Location	C:\Users\Administrator\AWS Schema Conversion Tool
Database Type	Transactional Database (OLTP)
Source Database Engine	Oracle / I want to switch engines and optimize for the cloud

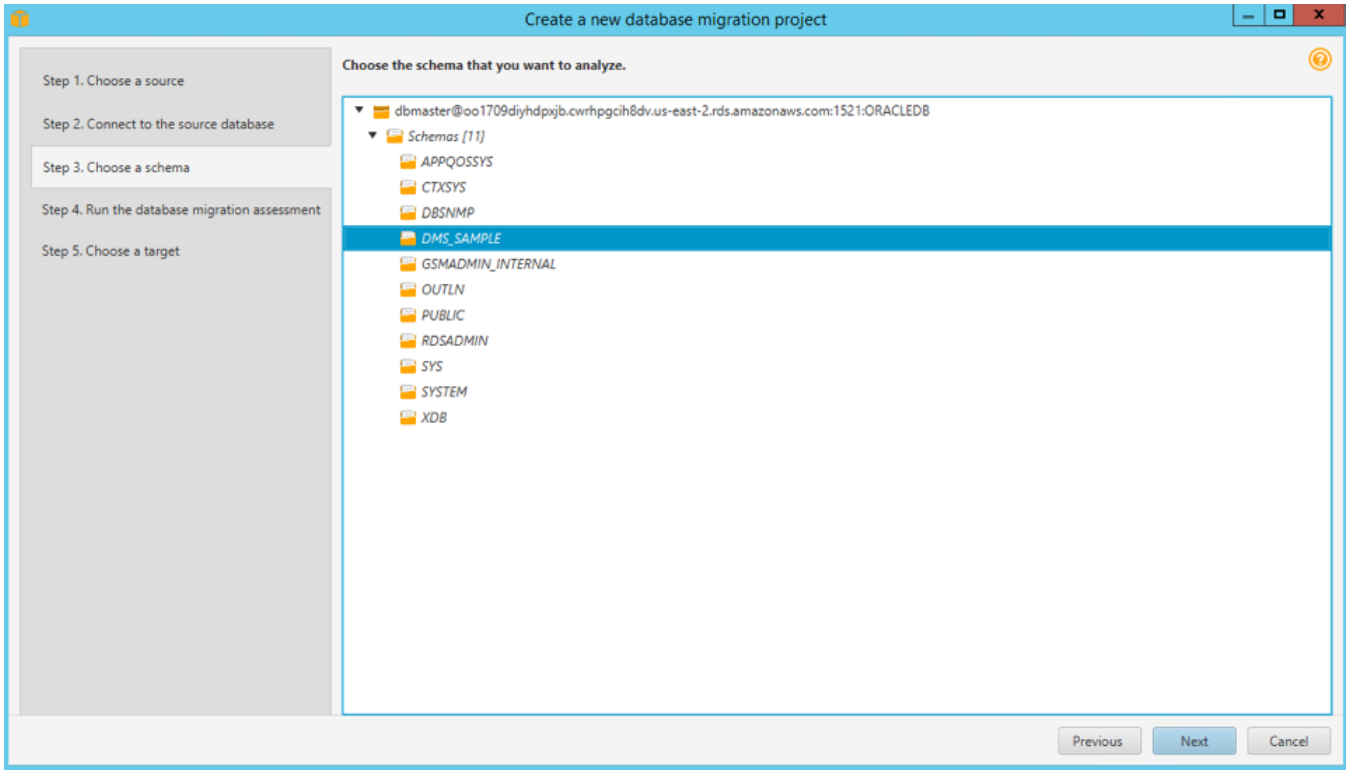


14. Specify the source database configurations in the form, and click **Test Connection**. Once the connection is successfully tested, click **Next**.

Parameter	Value
Type	SID
Server Name	< SourceOracleEndpoint >
Server Port	1521
Oracle SID	ORACLEDB
User Name	dbmaster
Password	dbmaster123
Use SSL	Unchecked
Store Password	Checked
Oracle Driver Path	C:\Users\Administrator\Desktop\DMS Workshop\JDBC\ojdbc8.jar



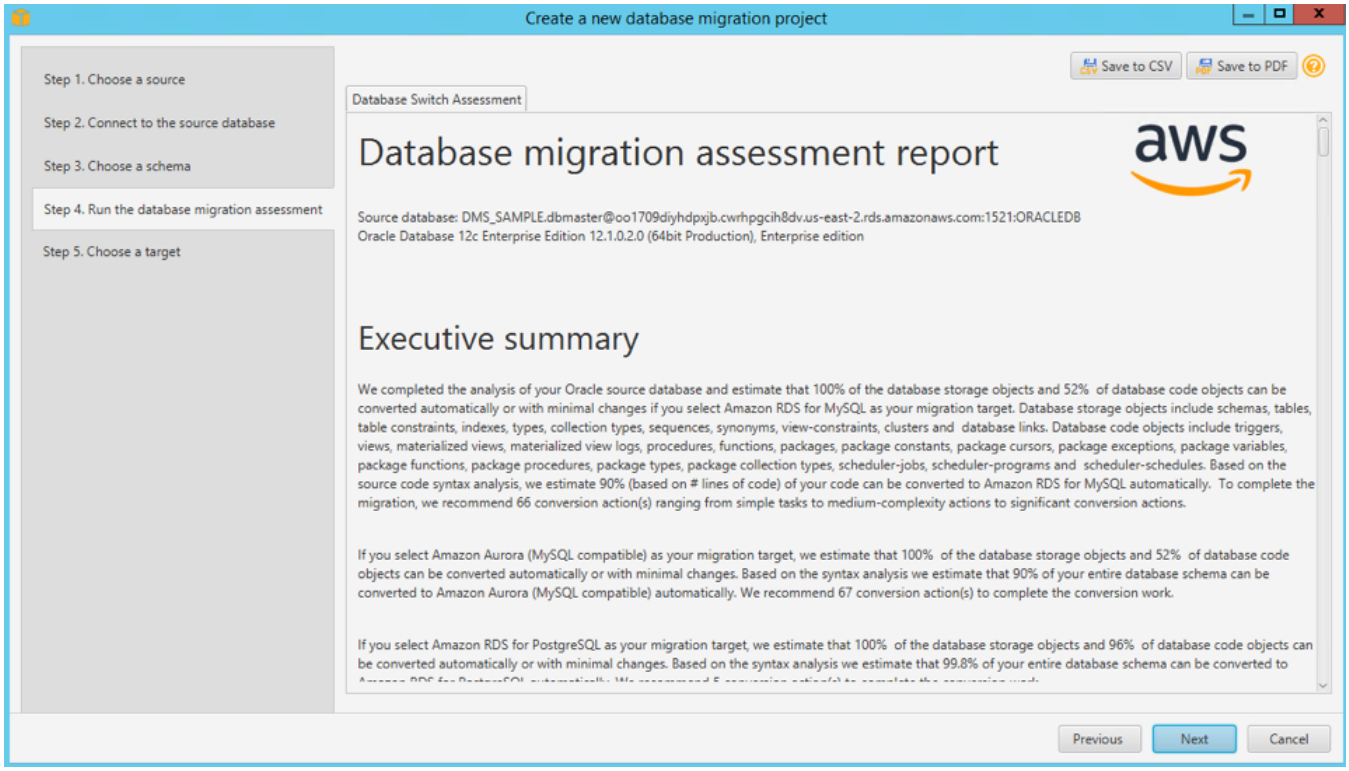
15. Select the DMS_SAMPLE database, then click Next.



Note

After hitting **Next** and loading metadata, you may get a warning message saying: **Metadata loading was intrupted because of data fetching issues.** You can ignore this message as it doesn't affect us in this workshop.

16. Review the **Database Migration Assessment Report**.



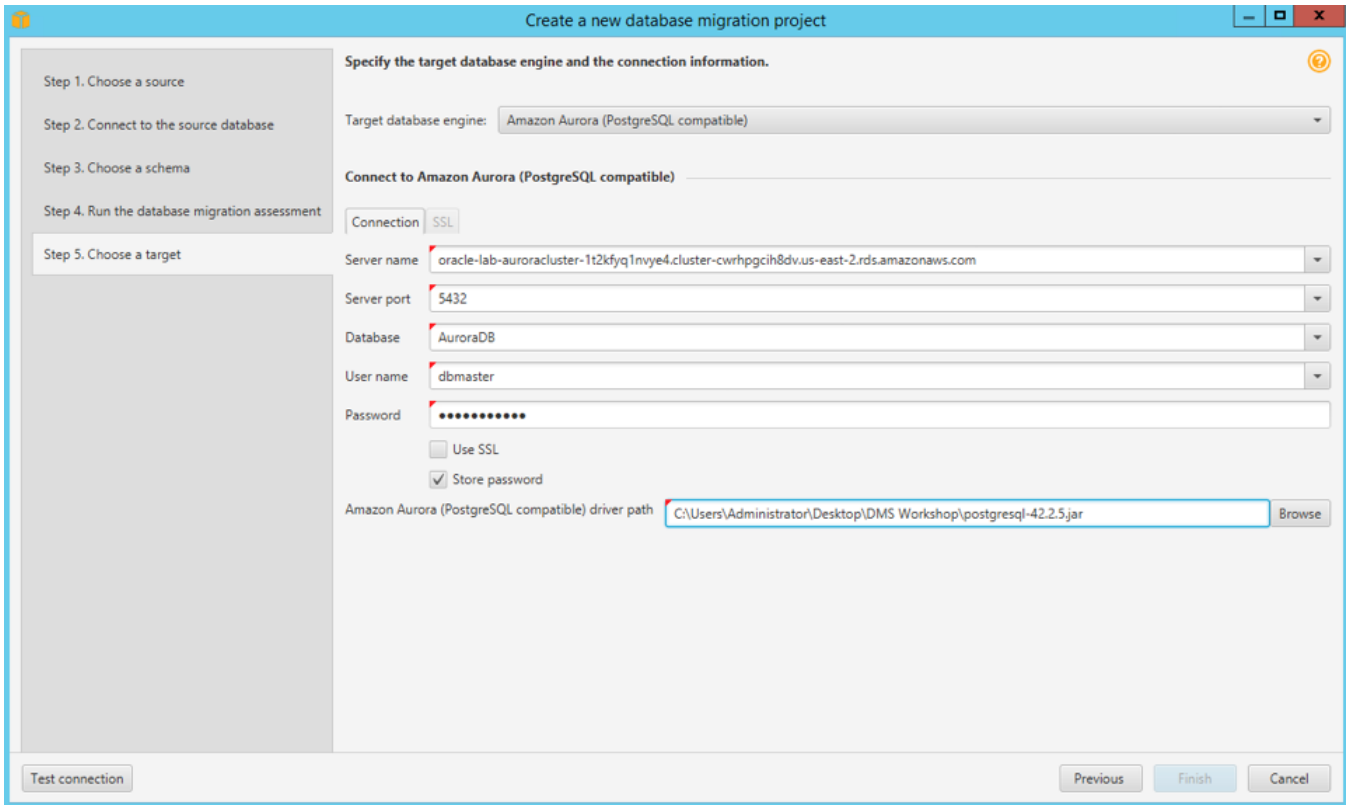
SCT will examine in detail all of the objects in the schema of source database. It will convert as much as possible automatically and provides detailed information about items it could not convert.

Generally, packages, procedures, and functions are more likely to have some issues to resolve because they contain the most custom or proprietary SQL code. AWS SCT specifies how much manual change is needed to convert each object type. It also provides hints about how to adapt these objects to the target schema successfully.

17. After you are done reviewing the database migration assessment report, click **Next**.

18. Specify the target database configurations in the form, and then click **Test Connection**. Once the connection is successfully tested, click **Finish**.

Parameter	Value
Target Database Engine	Amazon Aurora (PostgreSQL compatible)
Server Name	< TargetAuroraPostgreSQLEndpoint >
Server Port	5432
Database Name	AuroraDB
User Name	dbmaster
Password	dbmaster123
Use SSL	Unchecked
Save Password	Checked
Amazon Aurora Driver Path	C:\Users\Administrator\Desktop\DMS Workshop\JDBC\postgresql-42.2.9.jar



Note

You may see a warning message saying database version that you connected to is 9.x.x.x which is less than the recommended PostgreSQL 10.1. You can ignore the warning.

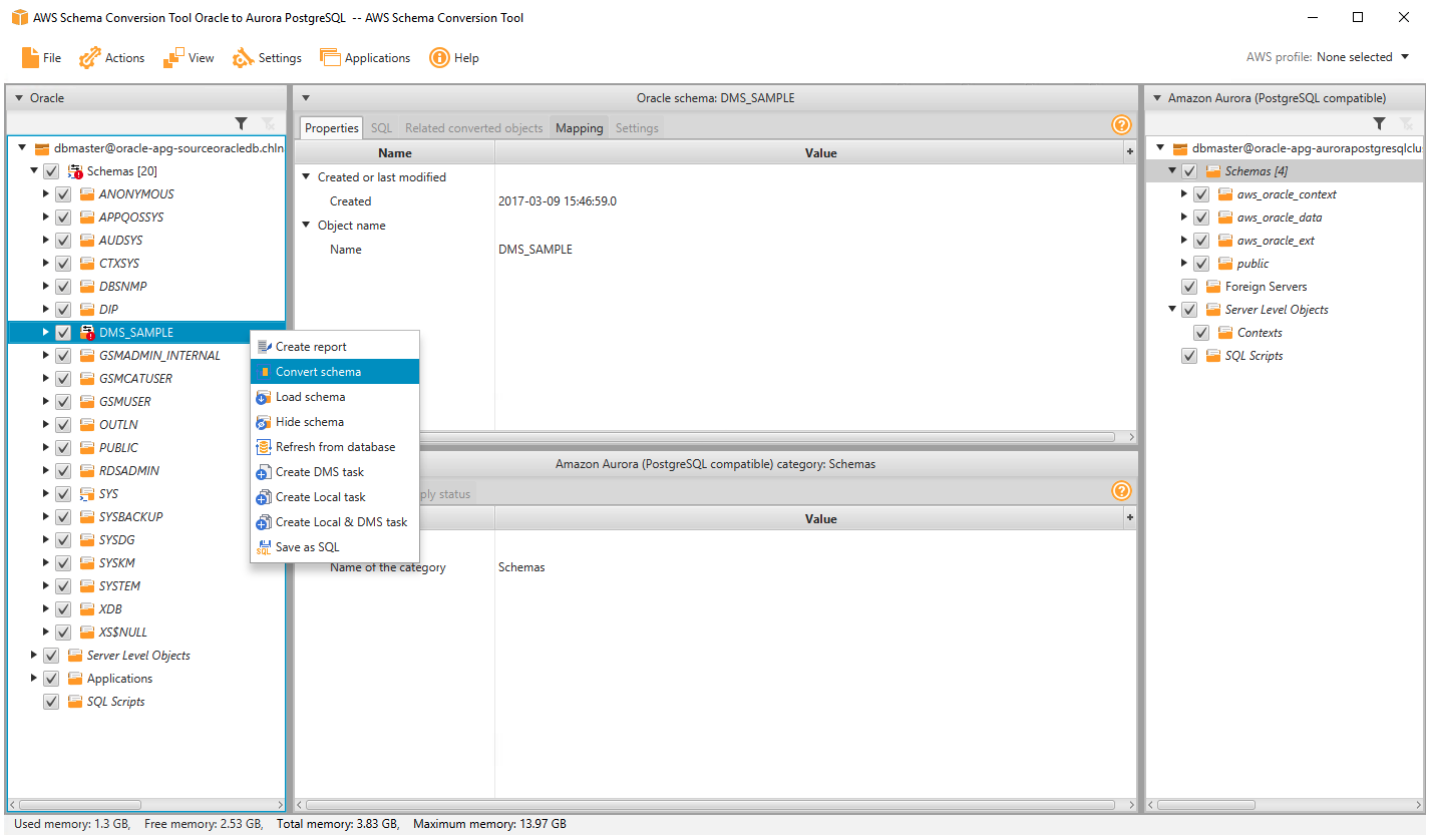
CONVERT THE SCHEMA

19. Right-click on the **DMS_SAMPLE** schema from Oracle source and select **Convert Schema** to generate the data definition language (DDL) statements for the target database.

You can view the generated DDL in the project console, and edit it before applying it to the target database. You can also choose to save it as an .sql file for application later.

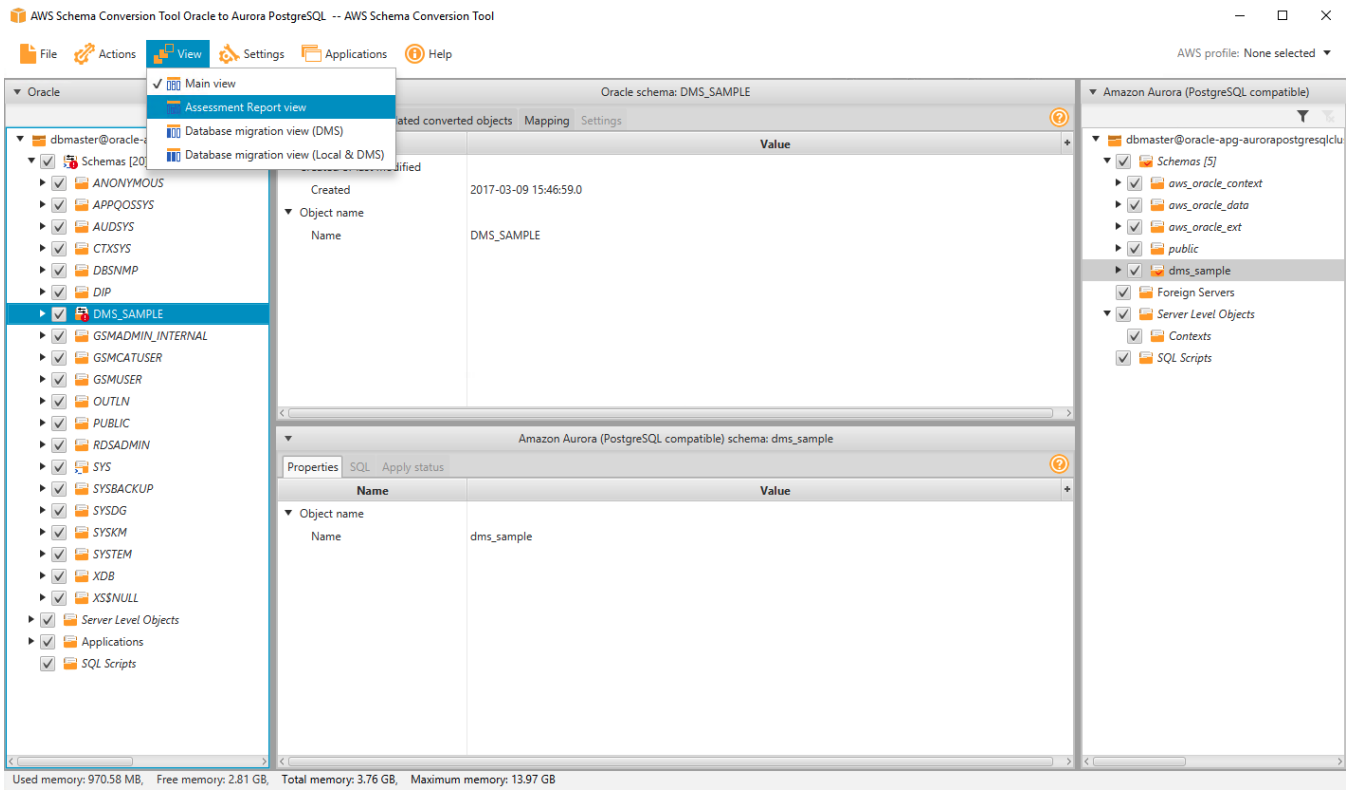
Warning

You may be prompted with a dialog box "Object may already exist in the target database, replace?" Select **Yes** and conversion will start.

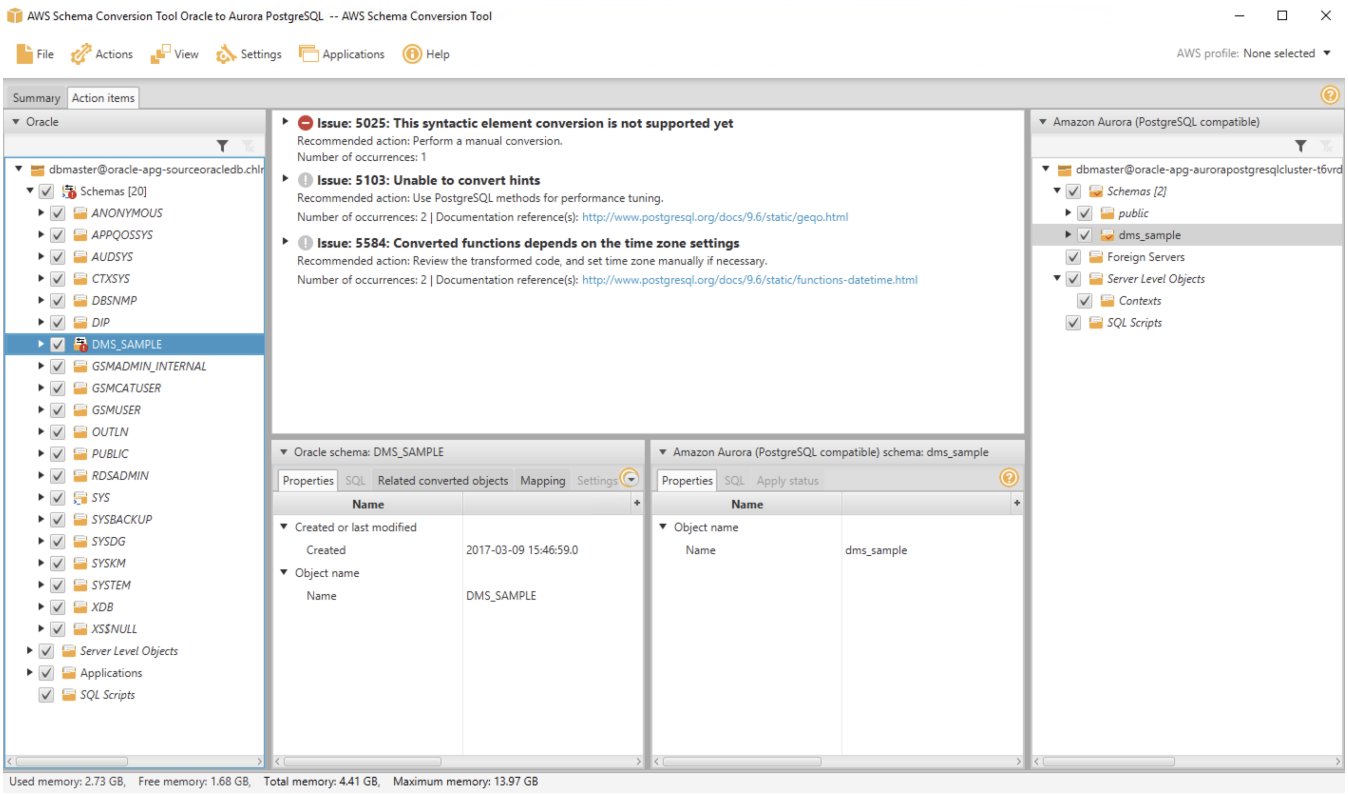


AWS SCT analyses the schema and creates a database migration assessment report for the conversion to PostgreSQL. Items with a red exclamation mark next to them cannot be directly translated from the source to the target. This includes Stored Procedures, and Packages.

20. Click on the **View** button, and choose **Assessment Report view**.



21. Next, navigate to the **Action Items** tab in the report to see the items that the tool could not convert, and find out how much manual changes you need to make.



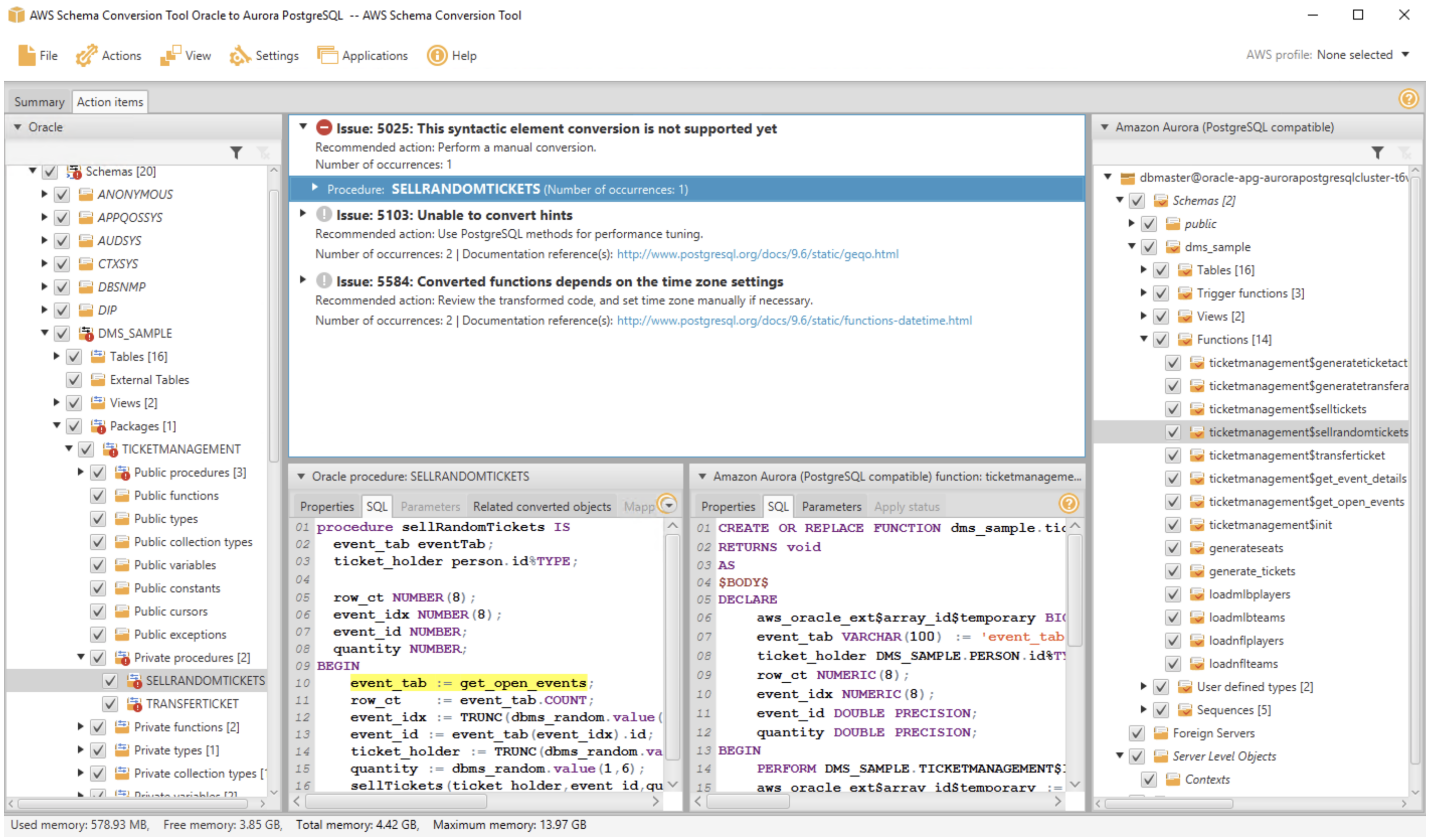
Check each of the issues listed and compare the contents under the source Oracle panel and the target Aurora PostgreSQL panel. Are the issues resolved? And how?

SCT has proposed resolutions by generating equivalent PostgreSQL DDL to convert the objects. Additionally, SCT highlights each conversion issue where it cannot automatically generate a conversion, and provides you with hints on how you can successfully convert the database object.

Notice the issue highlighted in the Private procedures named SELLRANDOMTICKETS. You'll see that SCT is unable to automatically convert the assign operation. You can complete one of the following actions to fix the issue:

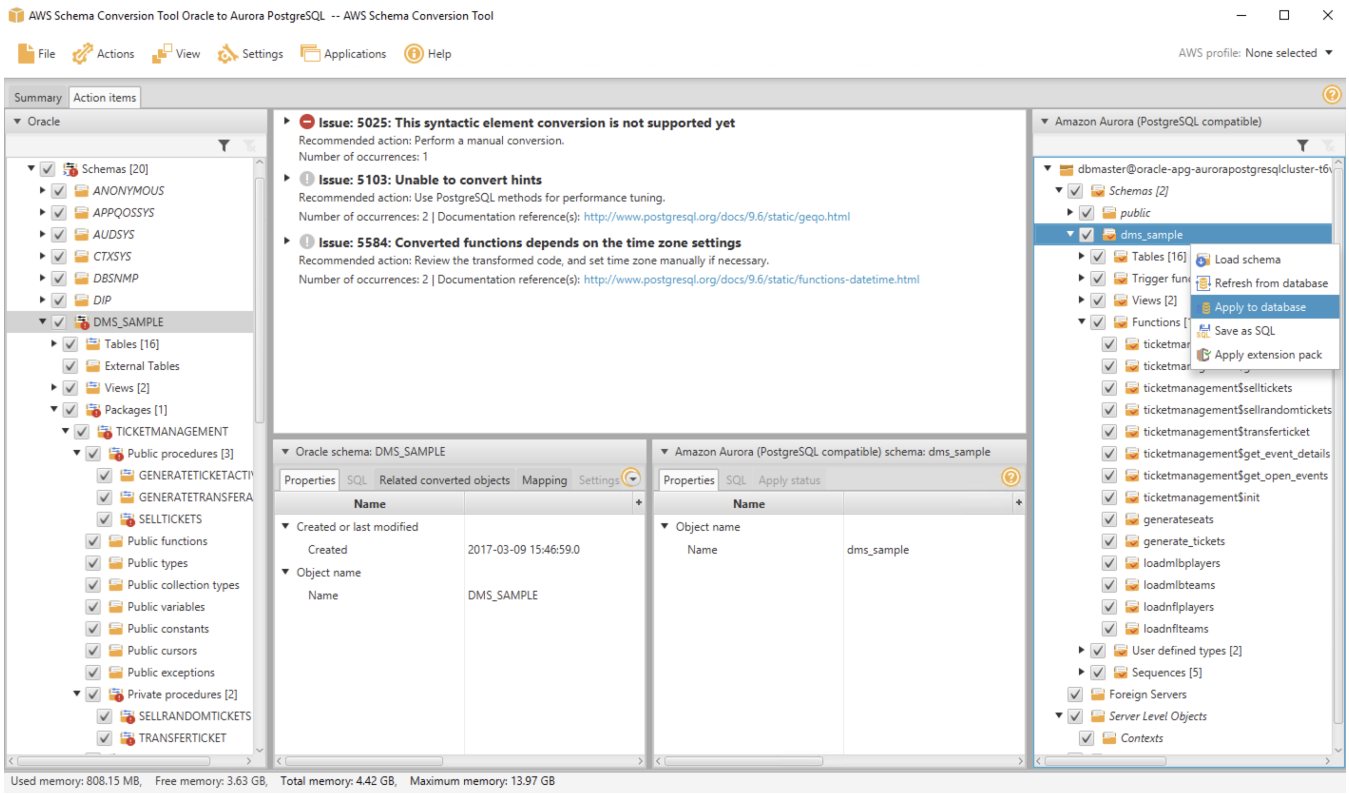
1. Modify the objects on the source Oracle database so that AWS SCT can convert the objects to the target Aurora PostgreSQL database.
2. Instead of modifying the source schema, modify scripts that AWS SCT generates before applying the scripts on the target Aurora PostgreSQL database.

For the sake of time, we skip modifying all the objects that could not be automatically converted. Instead, as an example, you will manually modify one of the stored procedures from within SCT to make it compatible with the target database.

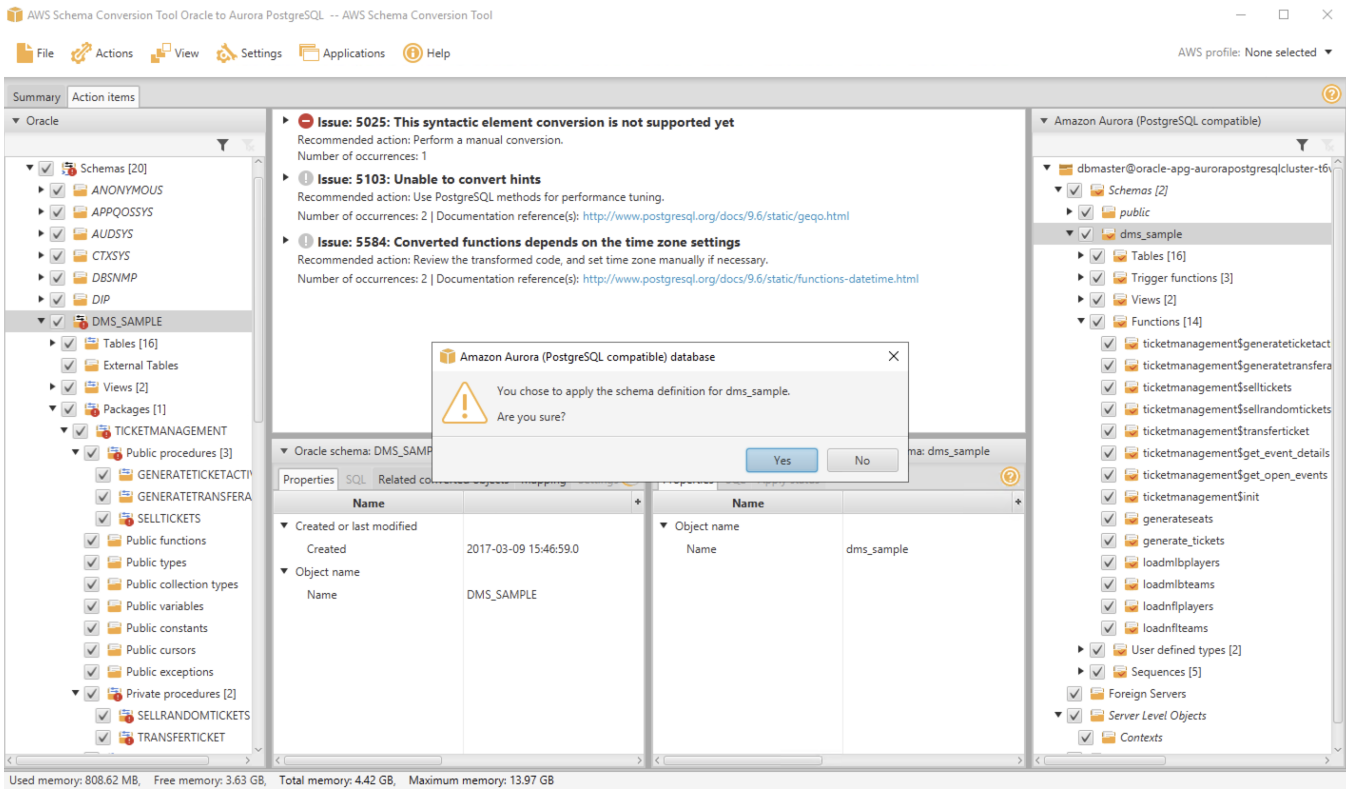


22. [Optional] Manually fix the schema issue. Then, right-click on the DMS_SAMPLE schema, and choose Create Report. Observe that the schema of the source database is now fully compatible with the target database.

23. Right click on the dms_sample schema in the right-hand panel, and click Apply to database.



24. When prompted if you want to apply the schema to the database, click **Yes**.



25. At this point, the schema has been applied to the target database. Expand the **dms_sample** schema to see the tables.

> **Note**
>
> You may see an exclamation mark on certain database objects such as indexes, and foreign key constraints. In the next section we will drop foreign key target database.

You have sucessfully converted the database schema and object from Oracle to the format compatible with Amazon Aurora (PostgreSQL).

This part demonstrated how easy it is to migrate the schema of an Oracle database into Amazon Aurora (PostgreSQL) using the AWS Schema Conversion Tool. Similarly, you learned how the Schema Conversion Tool highlights the differences between different database engine dialects, and provides you with tips on how you can successfully modify the code when needed to migrate procedure and other database objects.

The same steps can be followed to migrate SQL Server and Oracle workloads to other RDS engines including PostgreSQL and MySQL.

The next section describes the steps required to move the actual data using AWS DMS.

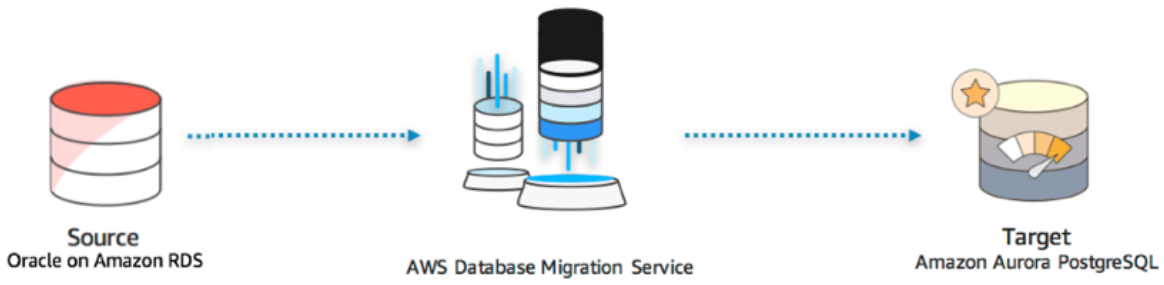
PART 2: DATA MIGRATION

> **Note**
>
> Please note that you need to complete the steps described in Schema Conversion section as a pre-requisite for this part.

This section will demonstrate how you can use the AWS Database Migration Service to migrate data from an Oracle database to an Amazon Aurora (PostgreSQL) instance. Additionally, you will use AWS DMS to continually replicate database changes from the source database to the target database. We do this in two steps:

1. First, you perform a full load migration of source oracle database to target Aurora PostgreSQL database using AWS DMS.
2. Next, you capture data changes (CDC) from the Oracle database, and replicate them automatically to Aurora PostgreSQL instance using AWS DMS.

AWS DMS doesn't migrate your secondary indexes, sequences, default values, stored procedures, triggers, synonyms, views, and other schema objects that aren't specifically related to data migration. To migrate these objects to your Aurora (PostgreSQL) target, we used the AWS Schema Conversion Tool in the previous section.



In this exercise you perform the following tasks:

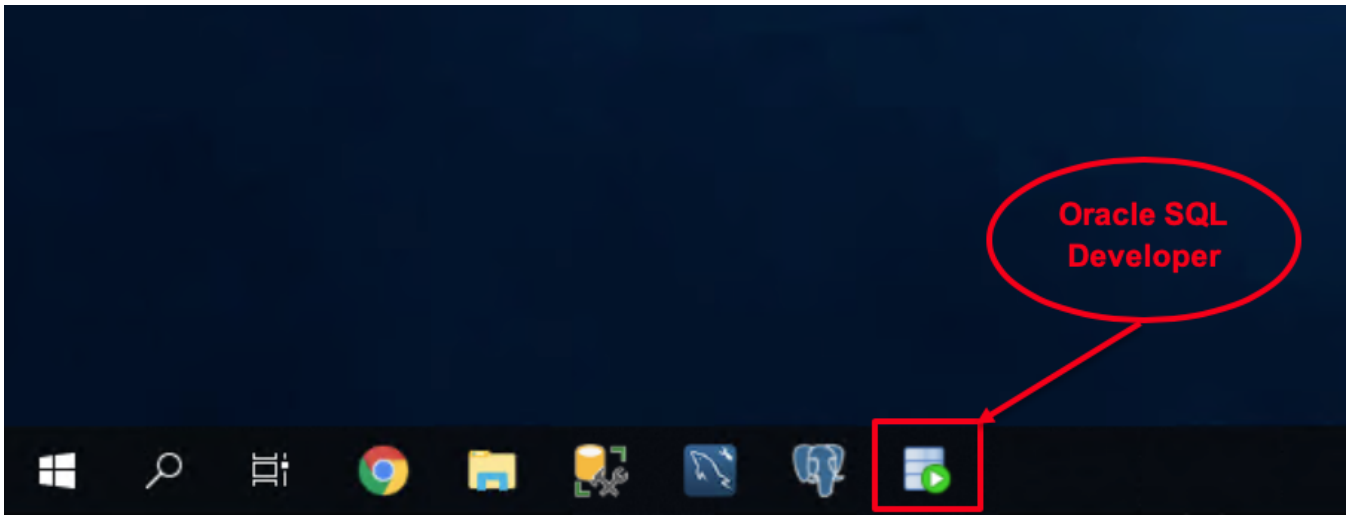
- Connect To The Source Oracle Database
- Configure the Source Database
- Configure the Target Database
- Create a DMS Replication Instance
- Create DMS Source and Target Endpoints
- Create a DMS Migration Task
- Inspect the Content of Target Database
- Replicate Data Changes

CONNECT TO THE SOURCE ORACLE DATABASE

Tip

If you disconnected from the Source EC2 instance, follow the instruction in Connect to the EC2 Instance section from the previous part to RDP to the instance.

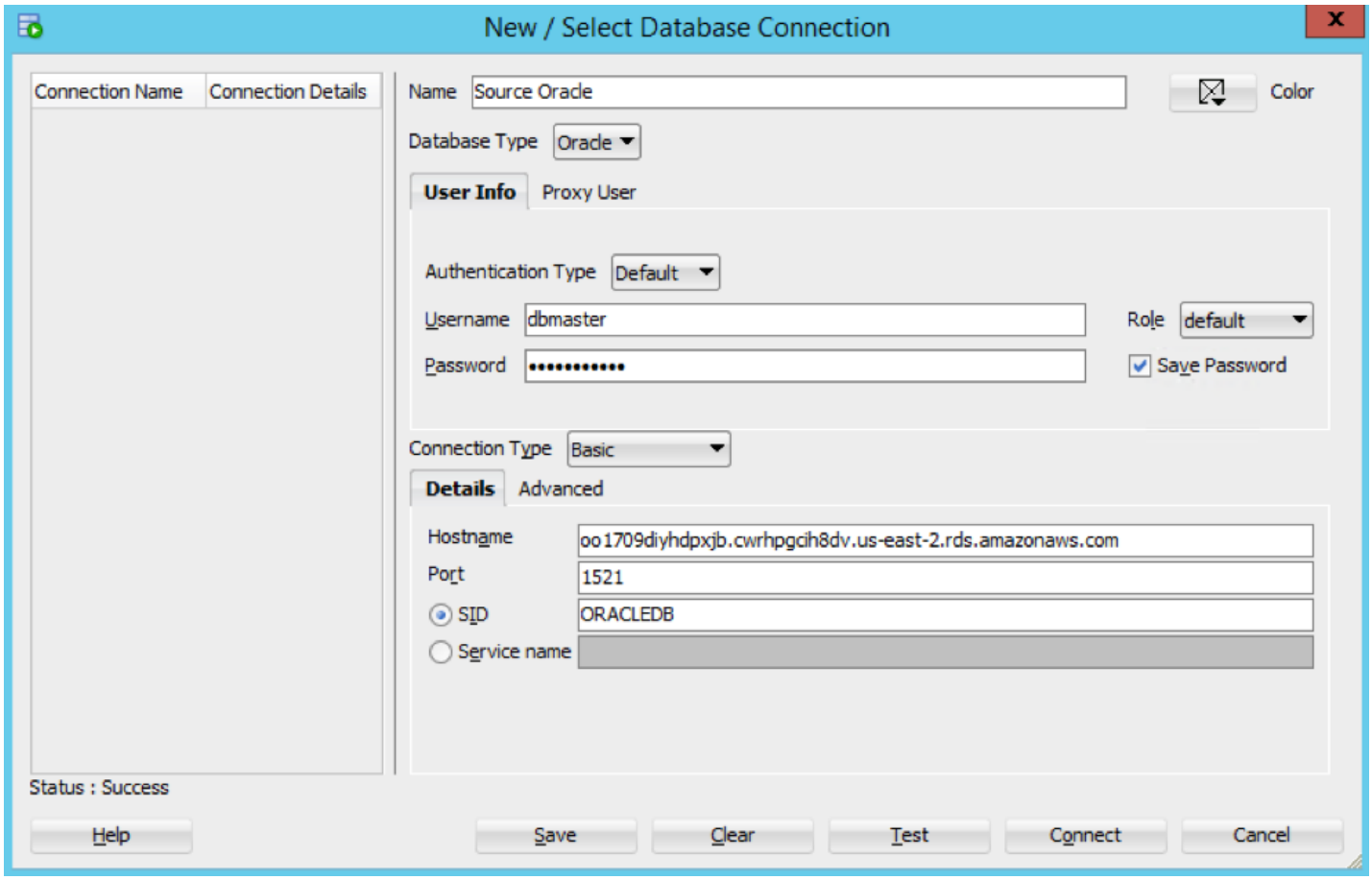
1. Once connected, open **Oracle SQL Developer** from the **Taskbar**.



2. 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

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



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

CONFIGURE THE SOURCE DATABASE

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.

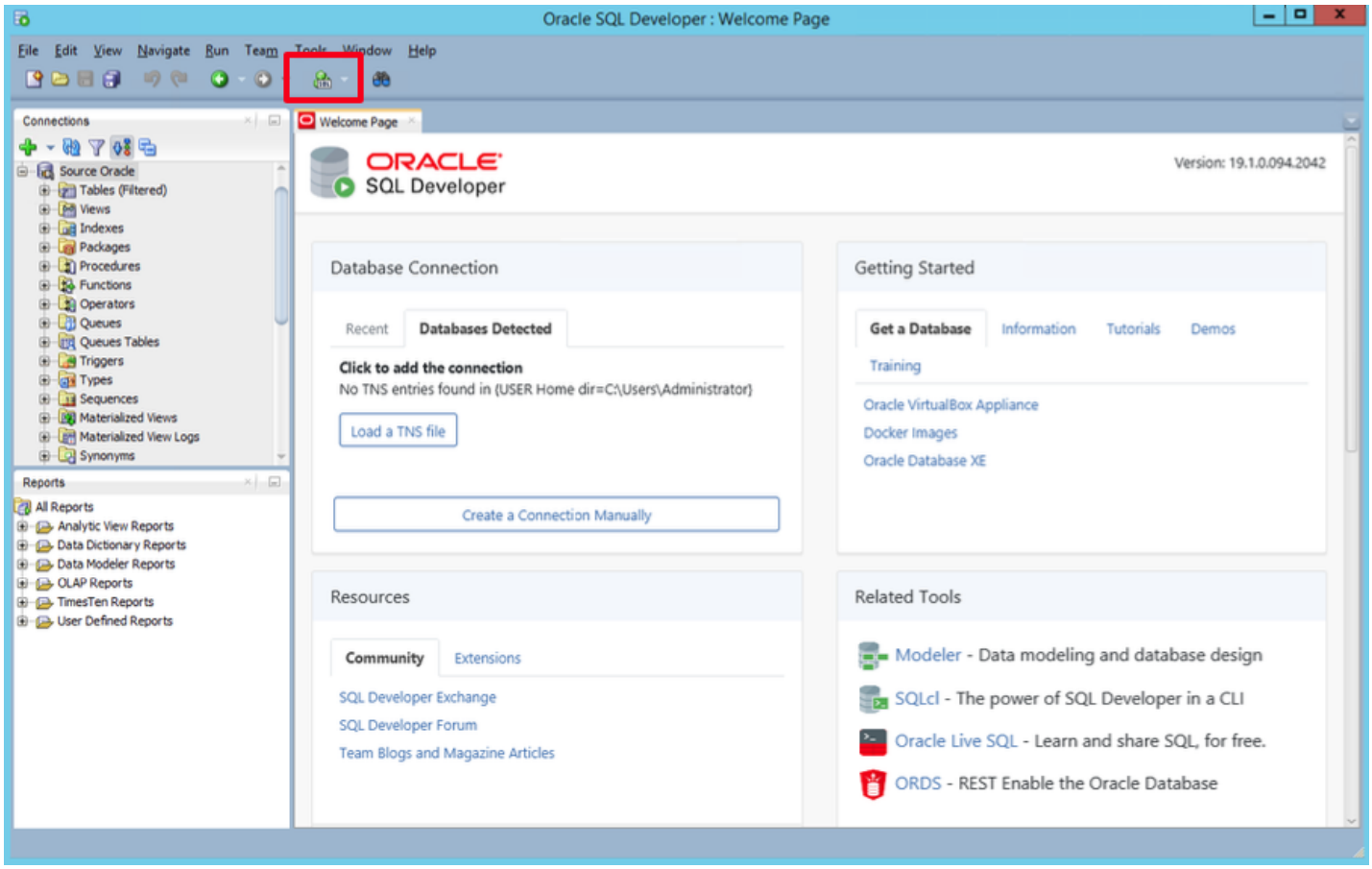
You also need to ensure that ARCHIVELOG MODE is on to provide information to LogMiner. AWS DMS uses LogMiner to read information from the archive logs so that AWS DMS can capture changes.

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.

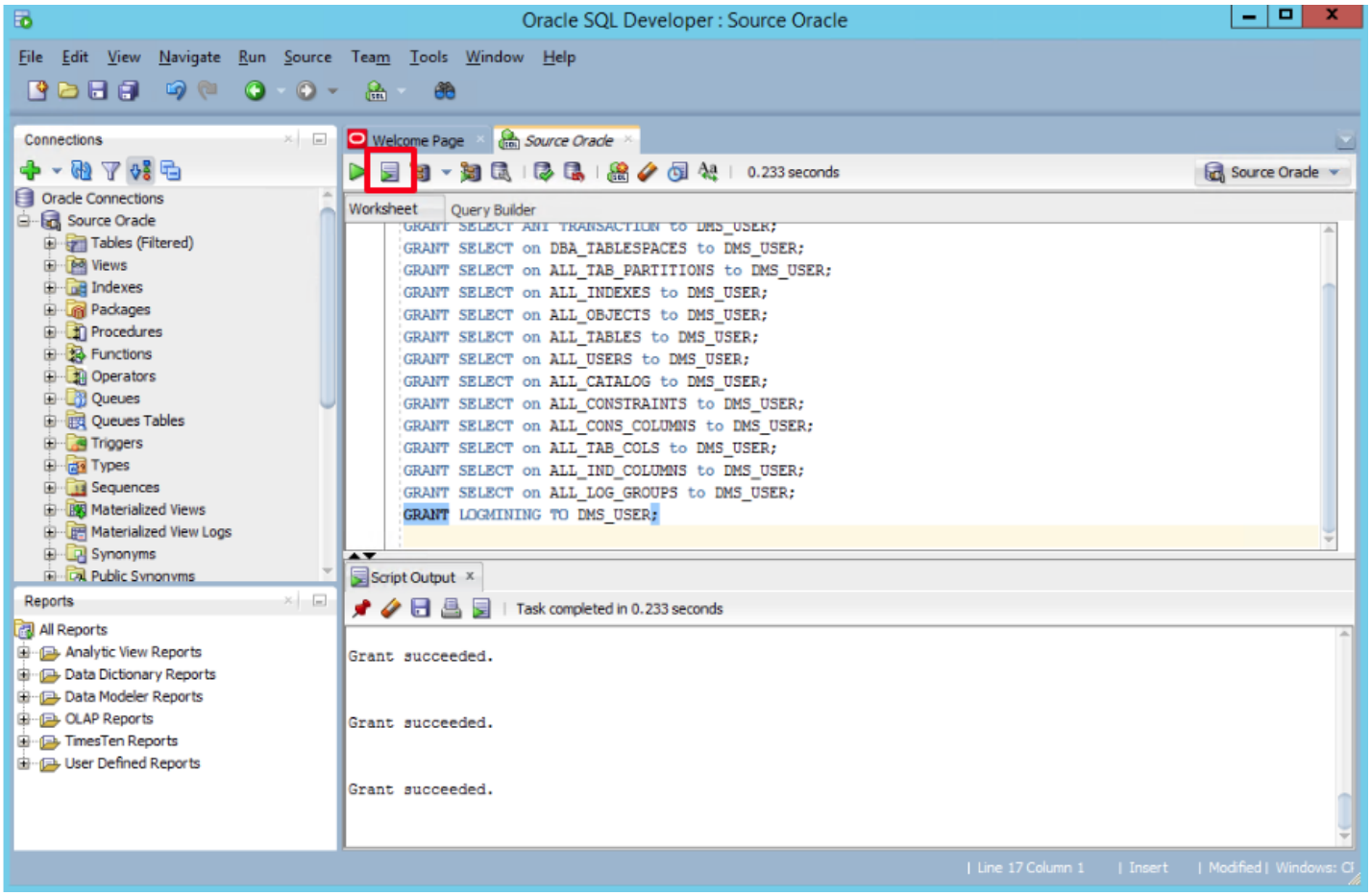
Similarly, you need to enable table-level supplemental logging for each table that you want to migrate.

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



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

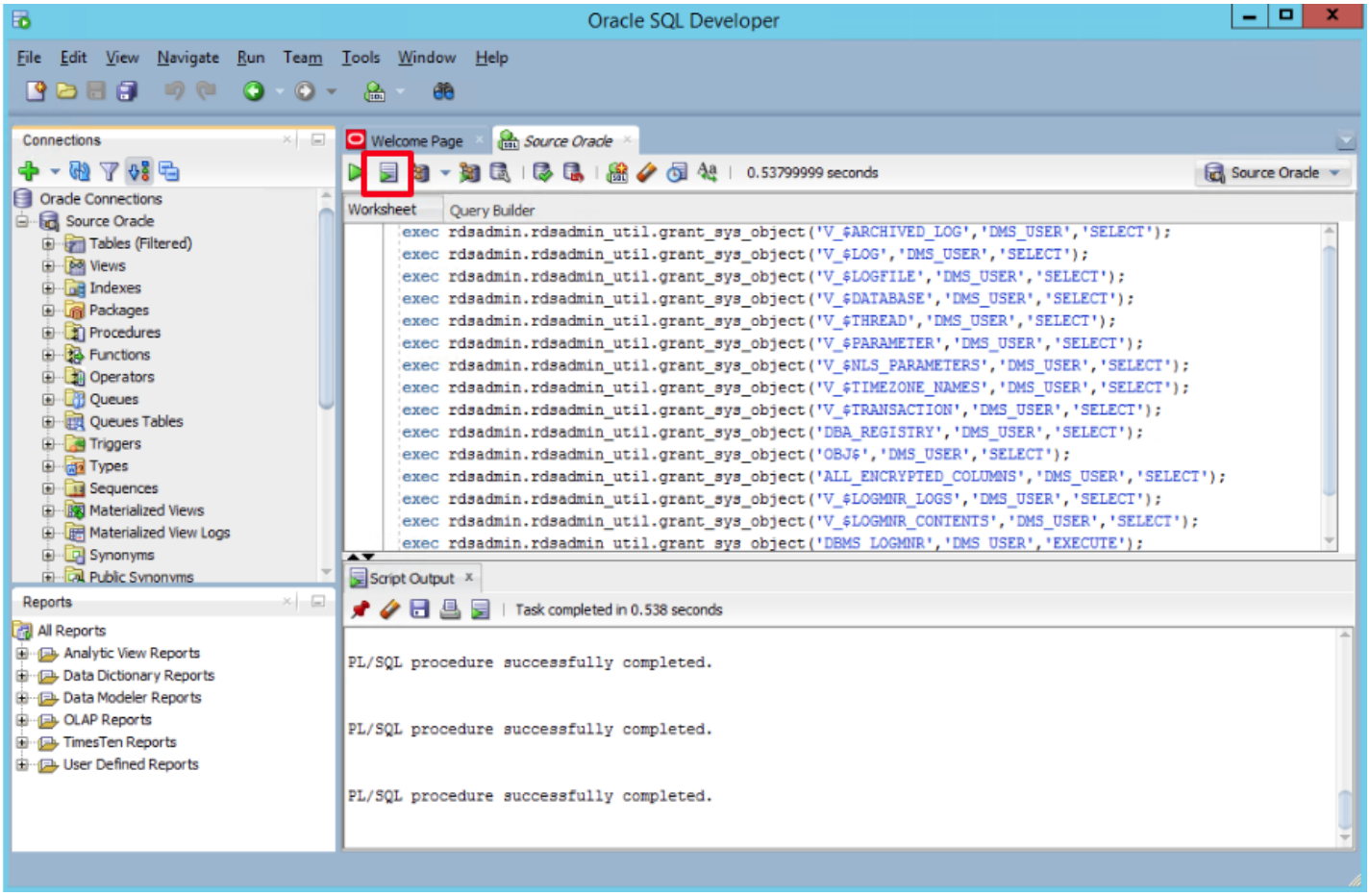
```
GRANT SELECT ANY TABLE to DMS_USER;
GRANT SELECT on ALL_VIEWS to DMS_USER;
GRANT SELECT ANY TRANSACTION to DMS_USER;
GRANT SELECT on DBA_TABLESPACES to DMS_USER;
GRANT SELECT on ALL_TAB_PARTITIONS to DMS_USER;
GRANT SELECT on ALL_INDEXES to DMS_USER;
GRANT SELECT on ALL_OBJECTS to DMS_USER;
GRANT 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;
```



6. 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','SELEC

exec
rdsadmin.rdsadmin_util.grant_sys_object('V_$TIMEZONE_NAMES','DMS_USER','SELEC

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','S

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','SELE

exec
rdsadmin.rdsadmin_util.grant_sys_object('DBMS_LOGMNR','DMS_USER','EXECUTE');
```



7. 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);
```

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

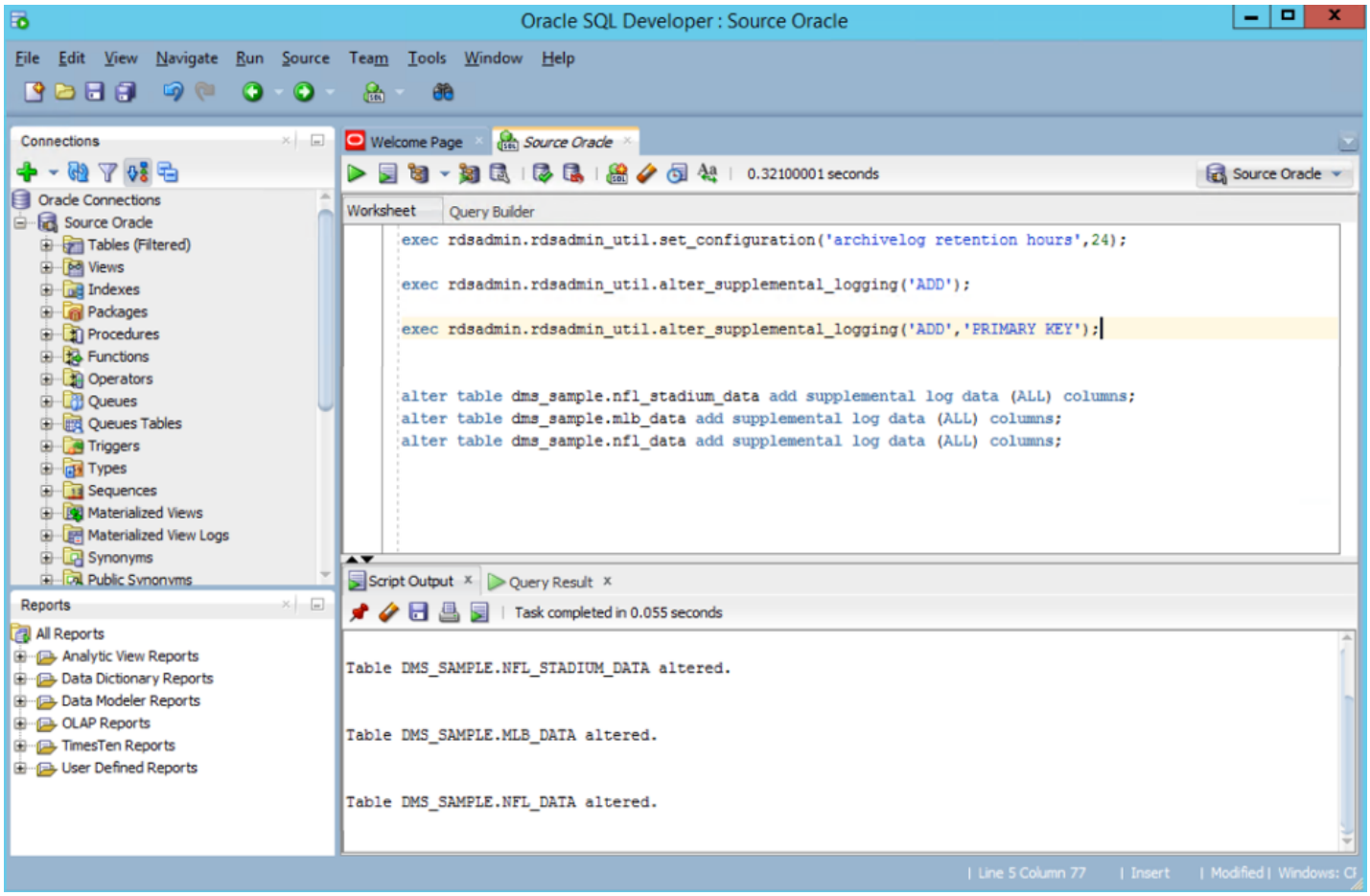
```
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD');
```

9. 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');
```

10. 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;
```

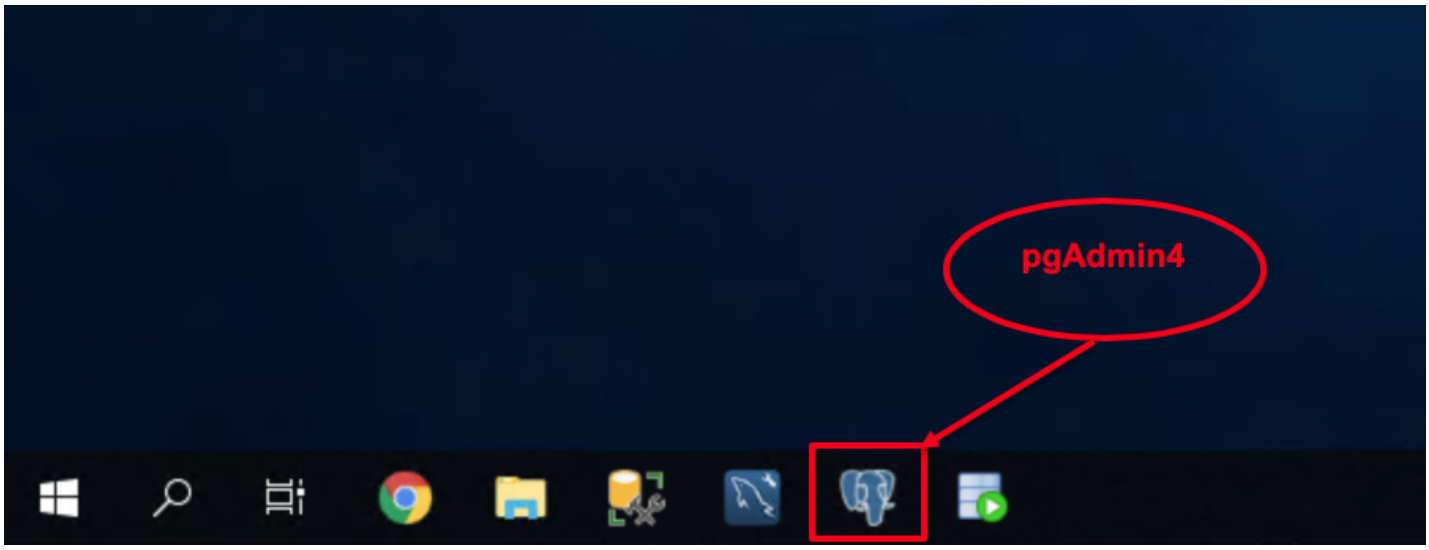

Oracle SQL Developer : Source Oracle
File Edit View Navigate Run Source Team Tools Window Help
Connections
Oracle Connections
Source Oracle
Tables (Filtered)
Views
Indexes
Packages
Procedures
Functions
Operators
Queues
Queues Tables
Triggers
Types
Sequences
Materialized Views
Materialized View Logs
Synonyms
Public Synonyms
Reports
All Reports
Analytic View Reports
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
TimesTen Reports
User Defined Reports
Welcome Page
Source Oracle
0.32100001 seconds
Source Oracle
Worksheet
Query Builder
exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24);
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD');
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD','PRIMARY KEY');
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;
Script Output
Query Result
Task completed in 0.055 seconds
Table DMS_SAMPLE.NFL_STADIUM_DATA altered.
Table DMS_SAMPLE.MLB_DATA altered.
Table DMS_SAMPLE.NFL_DATA altered.
Line 5 Column 77
Insert
Modified
Windows: CF

CONFIGURE THE TARGET DATABASE

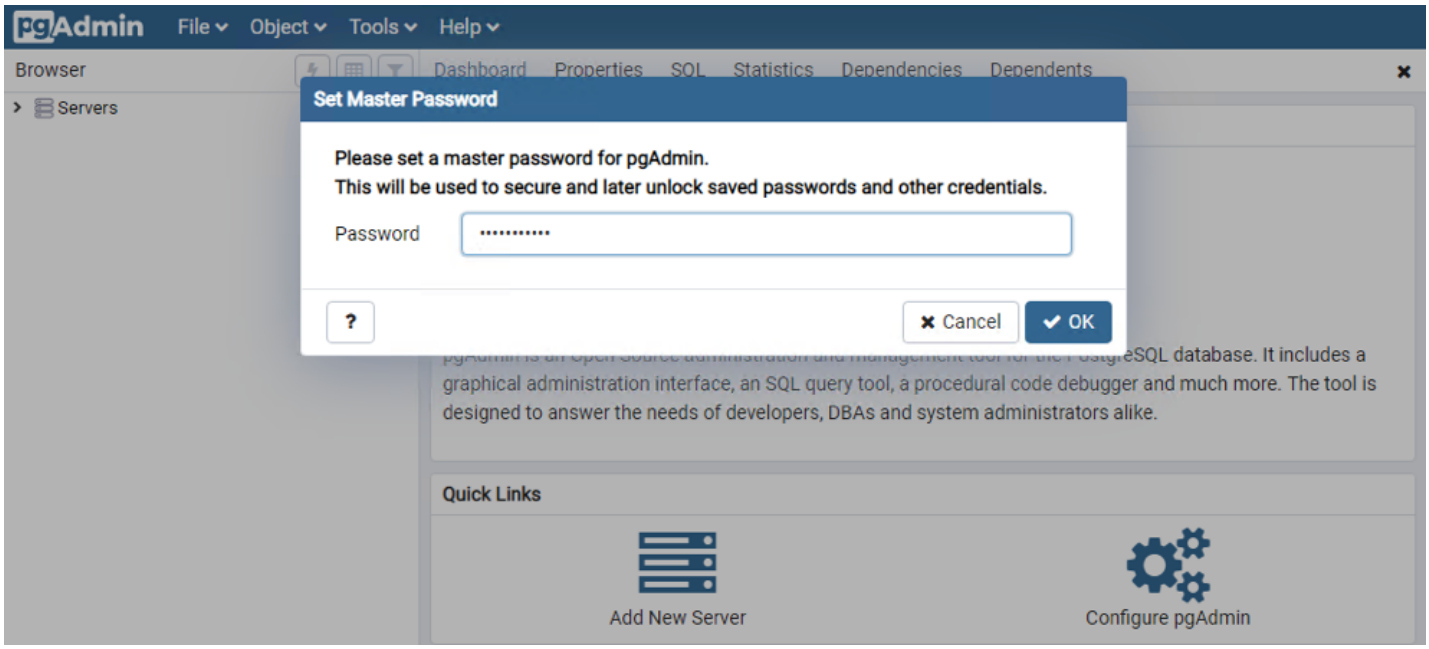
Info

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. To overcome this, we drop the constraints on the target database.

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

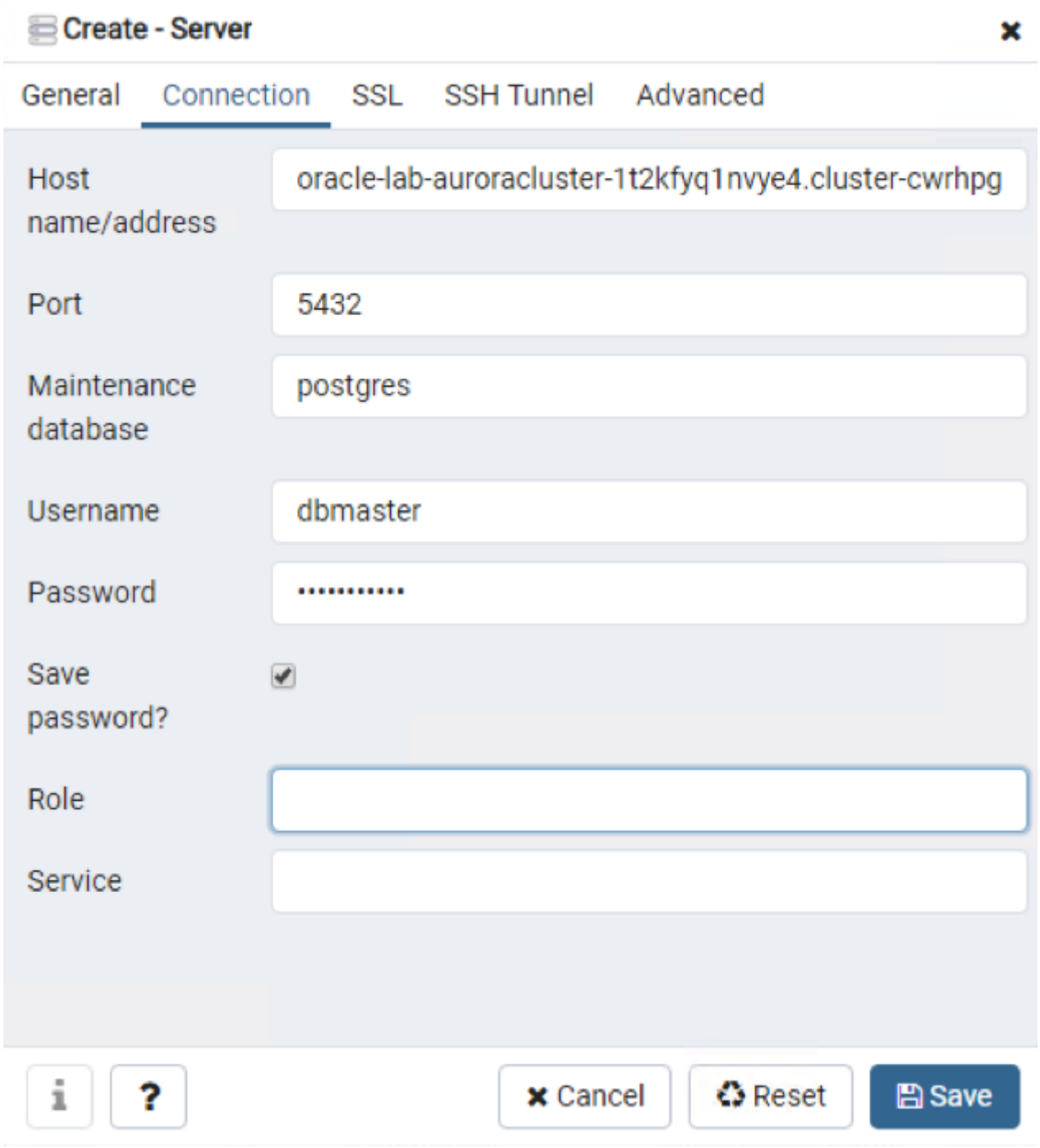


12. You may be prompted to set a **Master Password**. Enter **dbmaster123**, then click, **OK**.

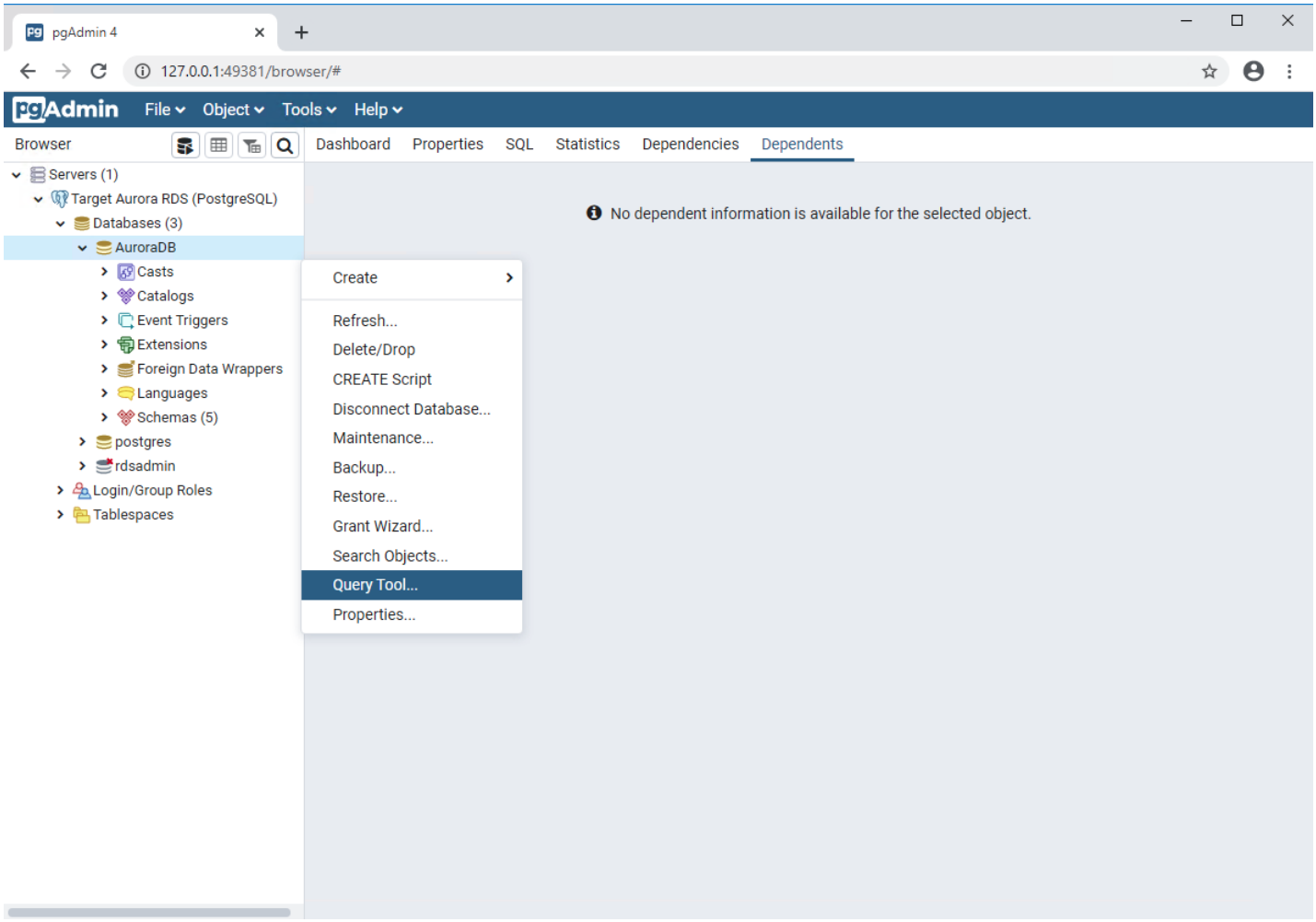


13. Click on the **Add New Server** icon, and enter the following values. Then, press **Save**.

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

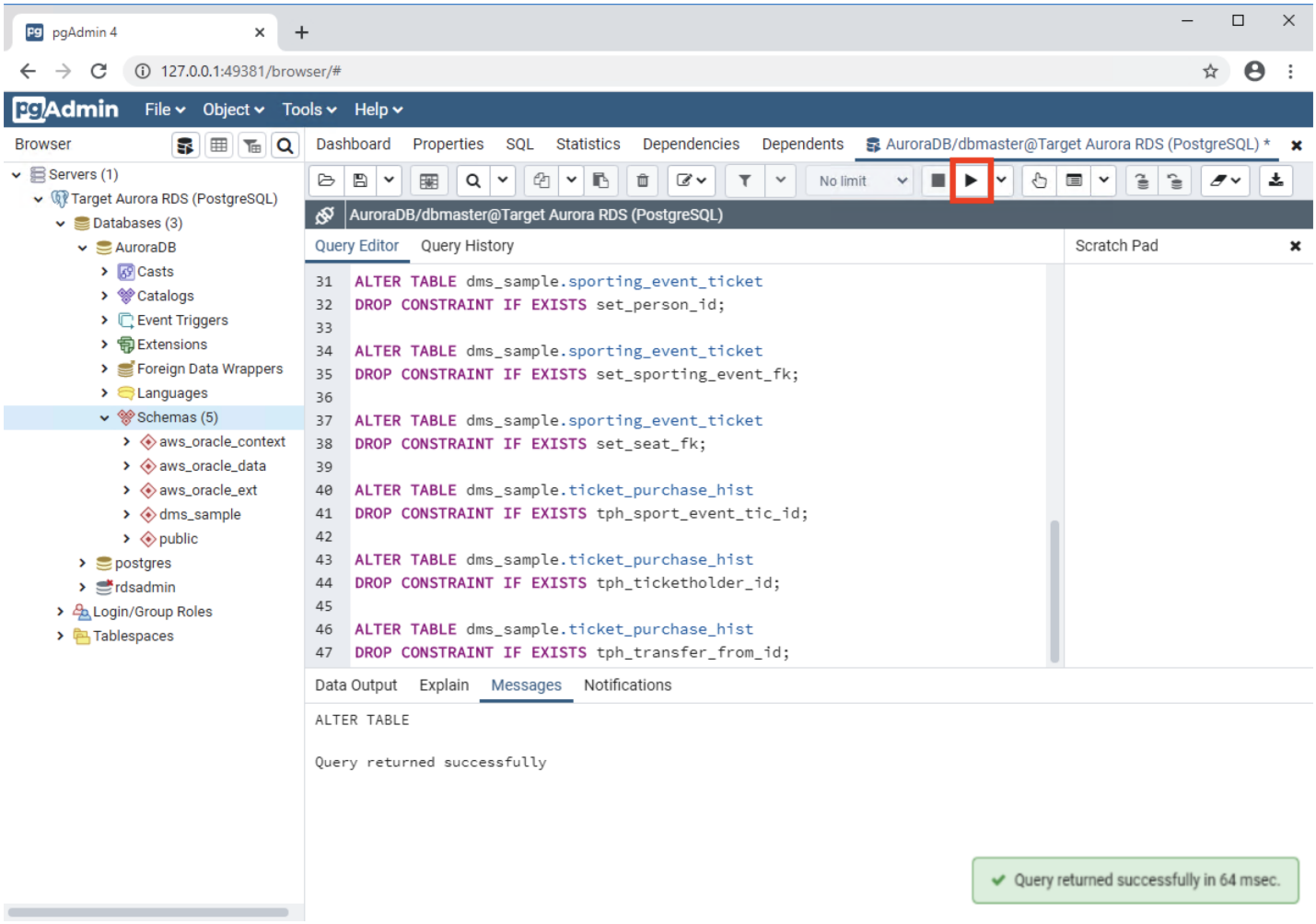


14. Right-click on AuroraDB database from left-hand menu, and then select Query Tool.



15. In this step you are going to drop the foreign key constraint from the target database:

    1. Open DropConstraintsPostgreSQL.sql in your favorite text editor.
    2. Copy the content of the file to the **Query Editor** in **pgAdmin 4**.
    3. **Execute** the script.


pgAdmin 4
127.0.0.1:49381/browser/#
pgAdmin
File
Object
Tools
Help
Browser
Servers (1)
Target Aurora RDS (PostgreSQL)
Databases (3)
AuroraDB
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Schemas (5)
aws_oracle_context
aws_oracle_data
aws_oracle_ext
dms_sample
public
postgres
rdsadmin
Login/Group Roles
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
AuroraDB/dbmaster@Target Aurora RDS (PostgreSQL) *
No limit
AuroraDB/dbmaster@Target Aurora RDS (PostgreSQL)
Query Editor
Query History
Scratch Pad
31 ALTER TABLE dms_sample.sporting_event_ticket
32 DROP CONSTRAINT IF EXISTS set_person_id;
33
34 ALTER TABLE dms_sample.sporting_event_ticket
35 DROP CONSTRAINT IF EXISTS set_sporting_event_fk;
36
37 ALTER TABLE dms_sample.sporting_event_ticket
38 DROP CONSTRAINT IF EXISTS set_seat_fk;
39
40 ALTER TABLE dms_sample.ticket_purchase_hist
41 DROP CONSTRAINT IF EXISTS tph_sport_event_tic_id;
42
43 ALTER TABLE dms_sample.ticket_purchase_hist
44 DROP CONSTRAINT IF EXISTS tph_ticketholder_id;
45
46 ALTER TABLE dms_sample.ticket_purchase_hist
47 DROP CONSTRAINT IF EXISTS tph_transfer_from_id;
Data Output
Explain
Messages
Notifications
ALTER TABLE
Query returned successfully
Query returned successfully in 64 msec.

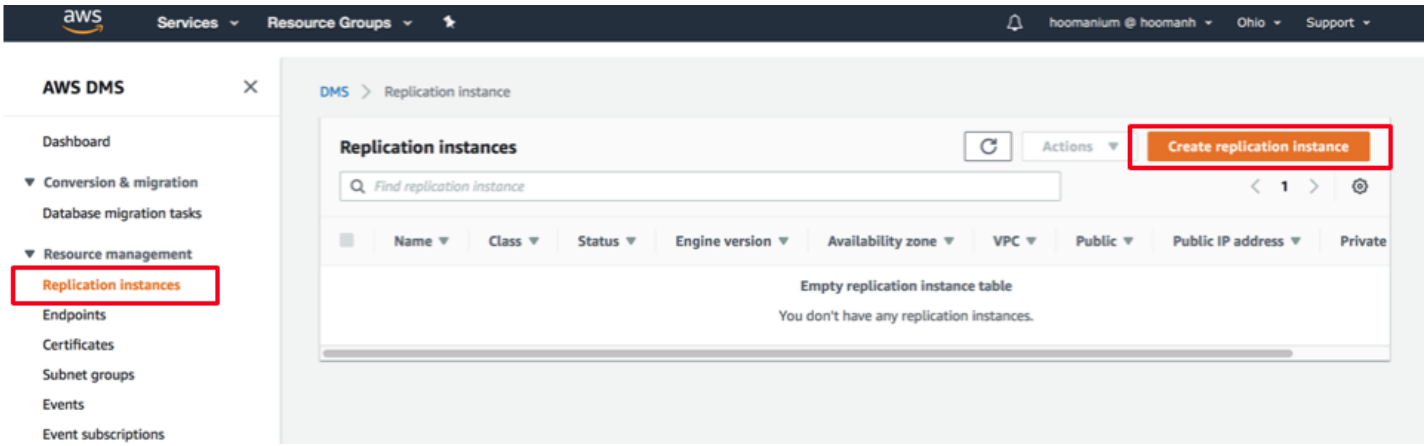
CREATE A DMS REPLICATION INSTANCE

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

Info

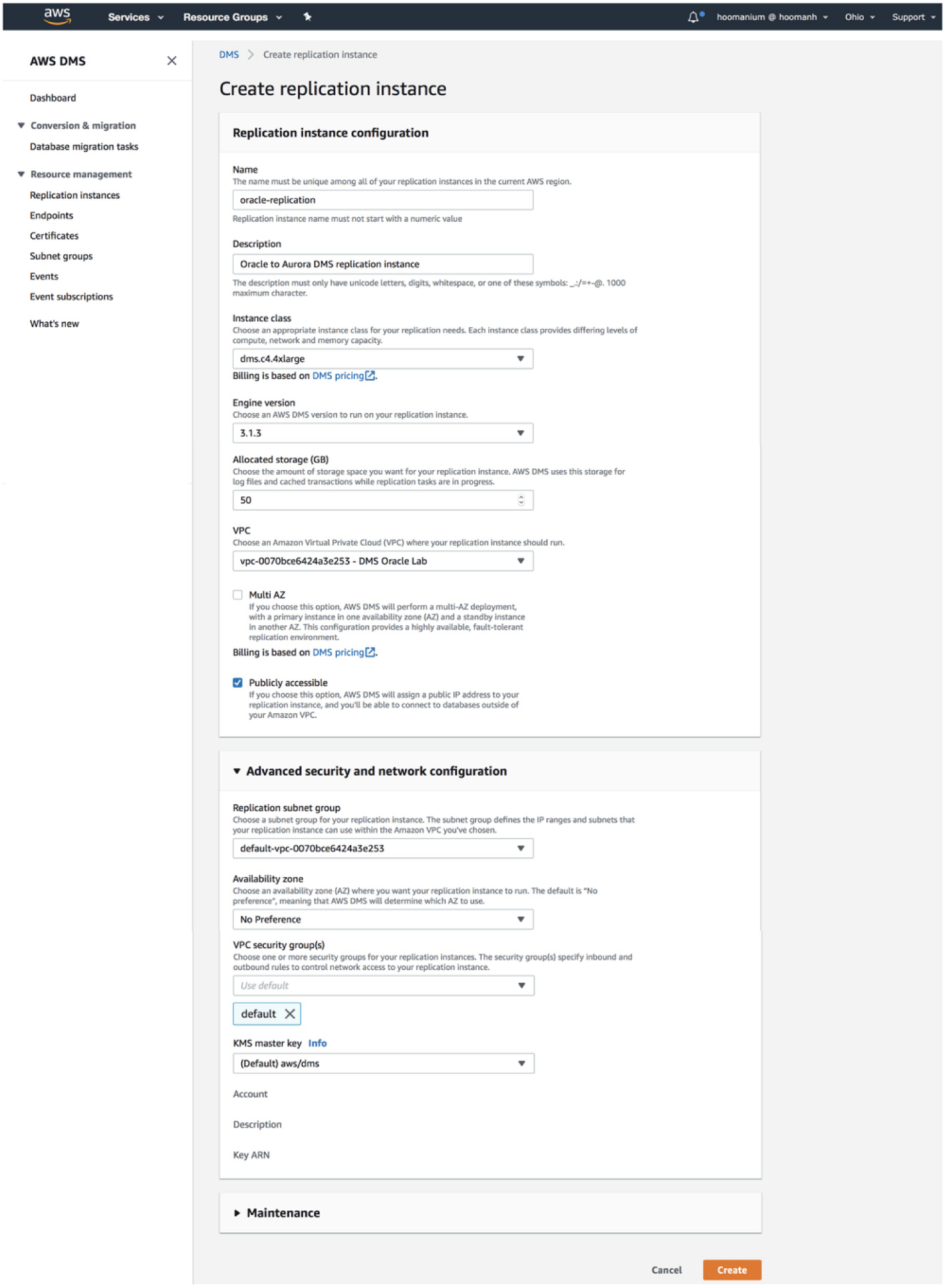
An AWS DMS replication instance performs the actual data migration between source and target. The replication instance also caches the transaction logs during the migration. The amount of CPU and memory capacity of a replication instance influences the overall time that is required for the migration.

16. Navigate to the Database Migration Service (DMS) console.
17. On the left-hand menu click on **Replication Instances**. This will launch the Replication instance screen.
18. Click on the **Create replication instance** button on the top right side.



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

Parameter	Value
Name	DMSReplication
Description	Replication server for Database Migration
Instance Class	dms.c4.xlarge
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


aws
Services
Resource Groups
hoomanium @ hoomanh
Ohio
Support
AWS DMS
Dashboard
Conversion & migration
Database migration tasks
Resource management
Replication instances
Endpoints
Certificates
Subnet groups
Events
Event subscriptions
What's new
DMS > Create replication instance
Create replication instance
Replication instance configuration
Name
The name must be unique among all of your replication instances in the current AWS region.
oracle-replication
Replication instance name must not start with a numeric value
Description
Oracle to Aurora DMS replication instance
The description must only have unicode letters, digits, whitespace, or one of these symbols: _.:/=+-@. 1000 maximum character.
Instance class
Choose an appropriate instance class for your replication needs. Each instance class provides differing levels of compute, network and memory capacity.
dms.c4.4xlarge
Billing is based on DMS pricing.
Engine version
Choose an AWS DMS version to run on your replication instance.
3.1.3
Allocated storage (GB)
Choose the amount of storage space you want for your replication instance. AWS DMS uses this storage for log files and cached transactions while replication tasks are in progress.
50
VPC
Choose an Amazon Virtual Private Cloud (VPC) where your replication instance should run.
vpc-0070bce6424a3e253 - DMS Oracle Lab
Multi AZ
If you choose this option, AWS DMS will perform a multi-AZ deployment, with a primary instance in one availability zone (AZ) and a standby instance in another AZ. This configuration provides a highly available, fault-tolerant replication environment.
Billing is based on DMS pricing.
Publicly accessible
If you choose this option, AWS DMS will assign a public IP address to your replication instance, and you'll be able to connect to databases outside of your Amazon VPC.
Advanced security and network configuration
Replication subnet group
Choose a subnet group for your replication instance. The subnet group defines the IP ranges and subnets that your replication instance can use within the Amazon VPC you've chosen.
default-vpc-0070bce6424a3e253
Availability zone
Choose an availability zone (AZ) where you want your replication instance to run. The default is "No preference", meaning that AWS DMS will determine which AZ to use.
No Preference
VPC security group(s)
Choose one or more security groups for your replication instances. The security group(s) specify inbound and outbound rules to control network access to your replication instance.
Use default
default
KMS master key Info
(Default) aws/dms
Account
Description
Key ARN
Maintenance
Cancel
Create

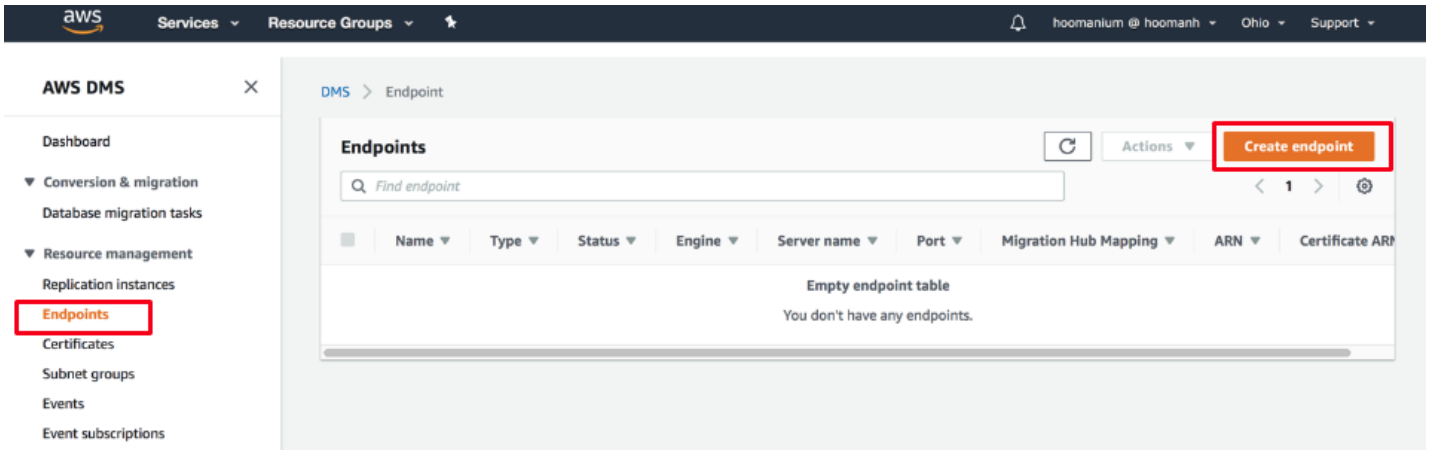
Note

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, test connectivity only after the replication instance has been created, because the replication instance is used in the connection.

CREATE DMS SOURCE AND TARGET ENDPOINTS

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

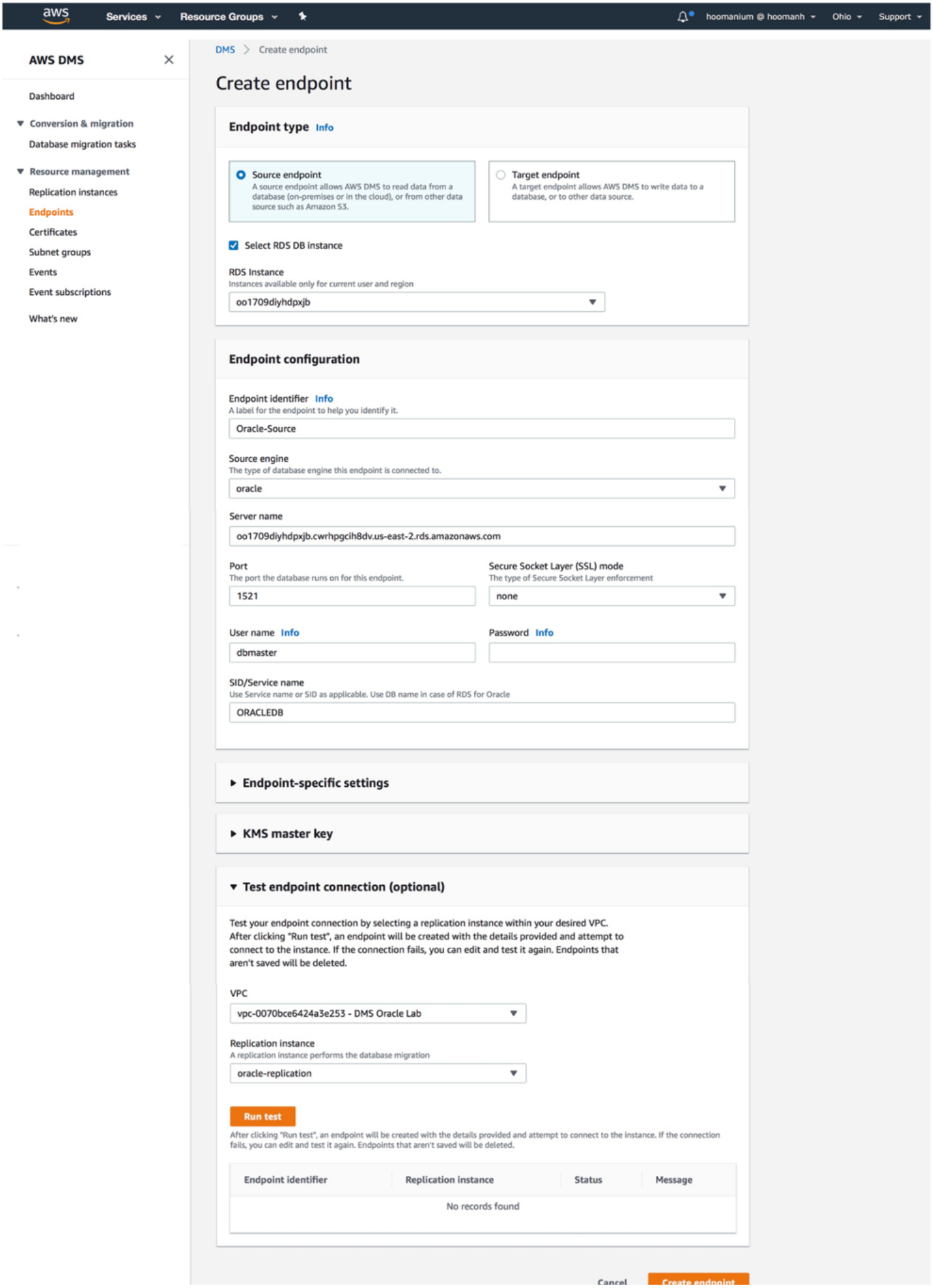
20. Click on the **Endpoints** link on the left, and then click on **Create endpoint** on the top right corner.

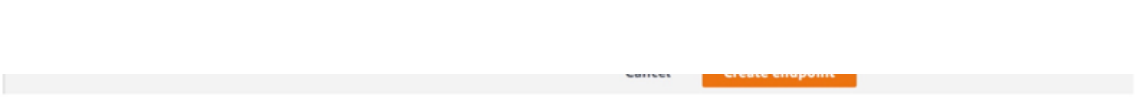


21. Enter the following information to create an endpoint for the source **dms_sample** database:

Parameter	Value
Endpoint Type	Source endpoint
Select RDS DB instance	Check
RDS Instance	<StackName>-SourceOracleDB

Parameter	Value
Endpoint Identifier	oracle-source
Source Engine	oracle
Server Name	< SourceOracleEndpoint >
Port	1521
SSL Mode	none
User Name	dbmaster
Password	dbmaster123
SID/Service Name	ORACLEDB
Test endpoint connection -> VPC	< VPC ID from Environment Setup Step >
Replication Instance	oracle-replication

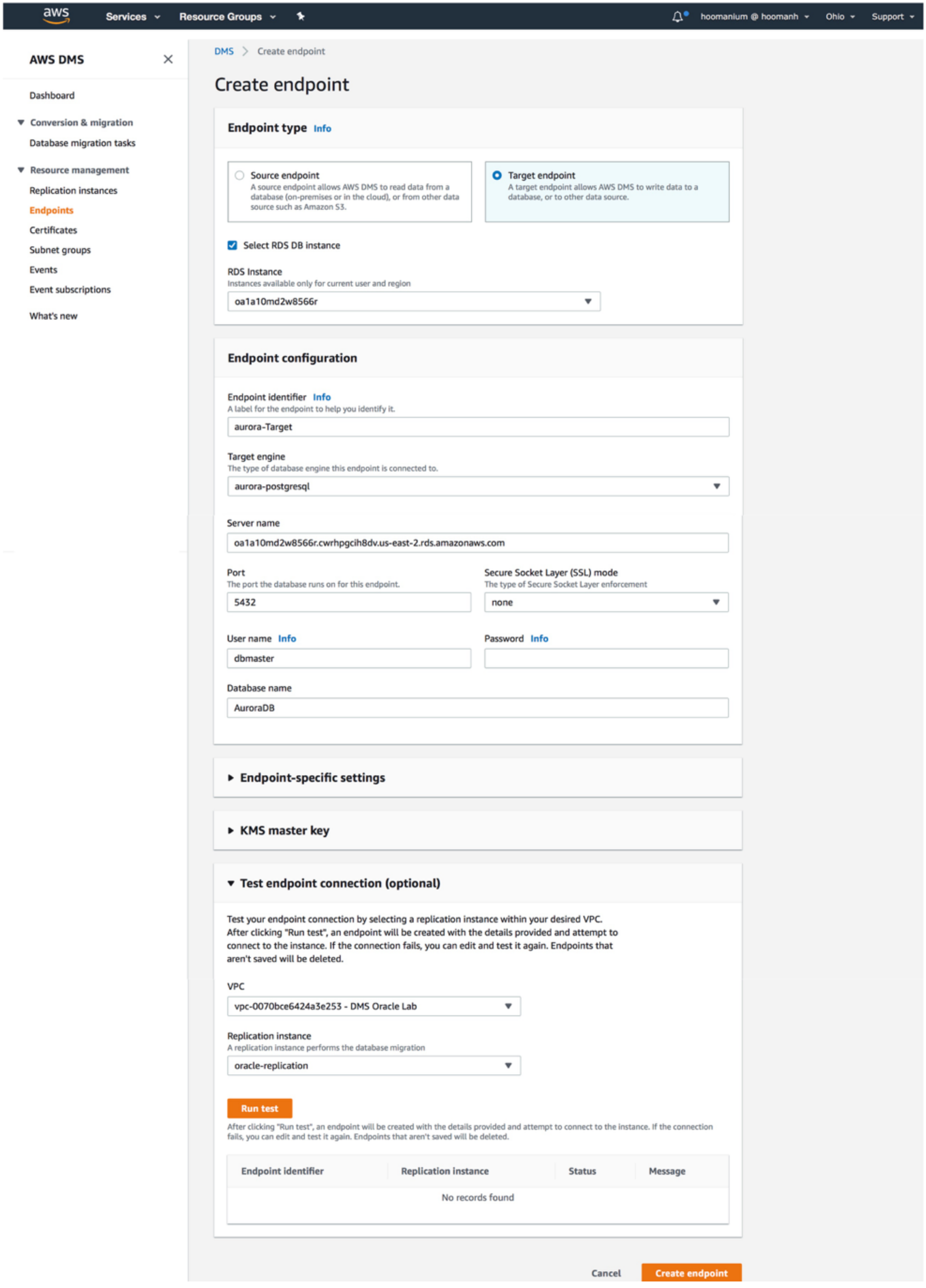

aws
Services
Resource Groups
hoomanium @ hoomanh
Ohio
Support
AWS DMS
Dashboard
Conversion & migration
Database migration tasks
Resource management
Replication instances
Endpoints
Certificates
Subnet groups
Events
Event subscriptions
What's new
DMS
Create endpoint
Create endpoint
Endpoint type Info
Source endpoint
A source endpoint allows AWS DMS to read data from a database (on-premises or in the cloud), or from other data source such as Amazon S3.
Target endpoint
A target endpoint allows AWS DMS to write data to a database, or to other data source.
Select RDS DB instance
RDS Instance
Instances available only for current user and region
oo1709diyhdpxjb
Endpoint configuration
Endpoint identifier Info
A label for the endpoint to help you identify it.
Oracle-Source
Source engine
The type of database engine this endpoint is connected to.
oracle
Server name
oo1709diyhdpxjb.cwrhpgcih8dv.us-east-2.rds.amazonaws.com
Port
The port the database runs on for this endpoint.
1521
Secure Socket Layer (SSL) mode
The type of Secure Socket Layer enforcement
none
User name Info
dbmaster
Password Info
SID/Service name
Use Service name or SID as applicable. Use DB name in case of RDS for Oracle
ORACLEDB
Endpoint-specific settings
KMS master key
Test endpoint connection (optional)
Test your endpoint connection by selecting a replication instance within your desired VPC. After clicking "Run test", an endpoint will be created with the details provided and attempt to connect to the instance. If the connection fails, you can edit and test it again. Endpoints that aren't saved will be deleted.
VPC
vpc-0070bce6424a3e253 - DMS Oracle Lab
Replication instance
A replication instance performs the database migration
oracle-replication
Run test
After clicking "Run test", an endpoint will be created with the details provided and attempt to connect to the instance. If the connection fails, you can edit and test it again. Endpoints that aren't saved will be deleted.
Endpoint identifier
Replication instance
Status
Message
No records found
Cancel
Create endpoint



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

23. 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	<StackName>-AuroraPostgreSQLInstance
Endpoint Identifier	aurora-target
Source Engine	aurora-postgresql
Server Name	< TargetAuroraPostgreSQLEndpoint >
Port	5432
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

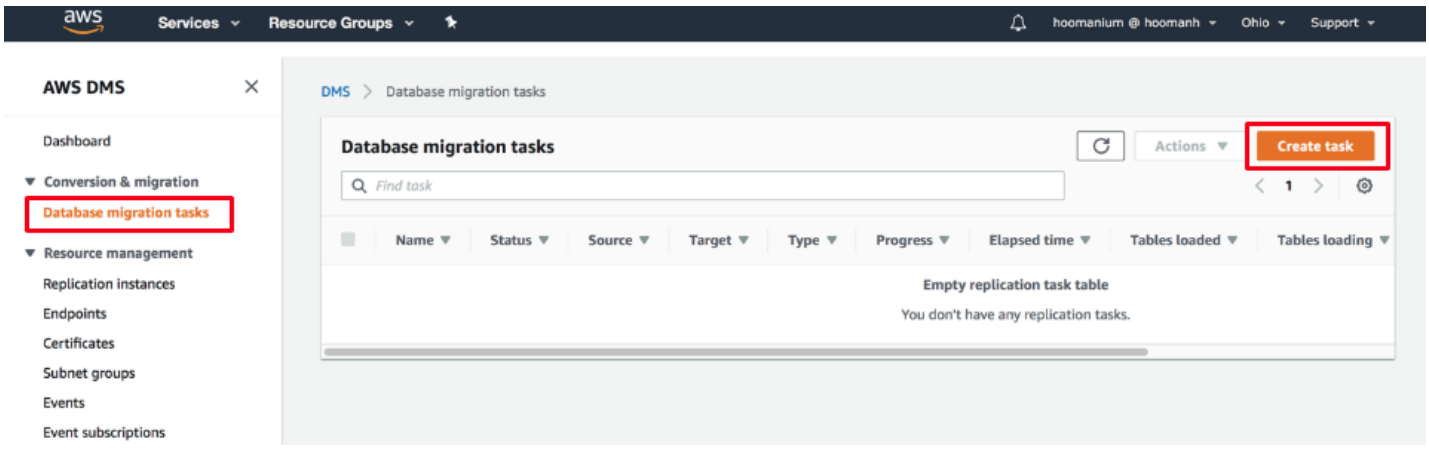

aws
Services
Resource Groups
hoomanium @ hoomanh
Ohio
Support
AWS DMS
Dashboard
Conversion & migration
Database migration tasks
Resource management
Replication instances
Endpoints
Certificates
Subnet groups
Events
Event subscriptions
What's new
DMS > Create endpoint
Create endpoint
Endpoint type Info
Source endpoint
A source endpoint allows AWS DMS to read data from a database (on-premises or in the cloud), or from other data source such as Amazon S3.
Target endpoint
A target endpoint allows AWS DMS to write data to a database, or to other data source.
Select RDS DB instance
RDS Instance
Instances available only for current user and region
oa1a10md2w8566r
Endpoint configuration
Endpoint identifier Info
A label for the endpoint to help you identify it.
aurora-Target
Target engine
The type of database engine this endpoint is connected to.
aurora-postgresql
Server name
oa1a10md2w8566r.cwrhpgcih8dv.us-east-2.rds.amazonaws.com
Port
The port the database runs on for this endpoint.
5432
Secure Socket Layer (SSL) mode
The type of Secure Socket Layer enforcement
none
User name Info
dbmaster
Password Info
Database name
AuroraDB
Endpoint-specific settings
KMS master key
Test endpoint connection (optional)
Test your endpoint connection by selecting a replication instance within your desired VPC. After clicking "Run test", an endpoint will be created with the details provided and attempt to connect to the instance. If the connection fails, you can edit and test it again. Endpoints that aren't saved will be deleted.
VPC
vpc-0070bce6424a3e253 - DMS Oracle Lab
Replication instance
A replication instance performs the database migration
oracle-replication
Run test
After clicking "Run test", an endpoint will be created with the details provided and attempt to connect to the instance. If the connection fails, you can edit and test it again. Endpoints that aren't saved will be deleted.
Endpoint identifier
Replication instance
Status
Message
No records found
Cancel
Create endpoint

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

CREATE A DMS MIGRATION TASK

AWS DMS uses **Database Migration Task** to migrate the data from source to the target database. For this migraiton, 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.

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



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

Parameter	Value
Task identifier	oracle-migration-task
Replication instance	dmsreplication

Parameter	Value
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

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

28. Click on **Add new selection rule** button and enter the following values in the form:

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.

29. Next, expand the **Transformation rules** section, and click on **Add new transformation rule** using the following values:

Rule 1:

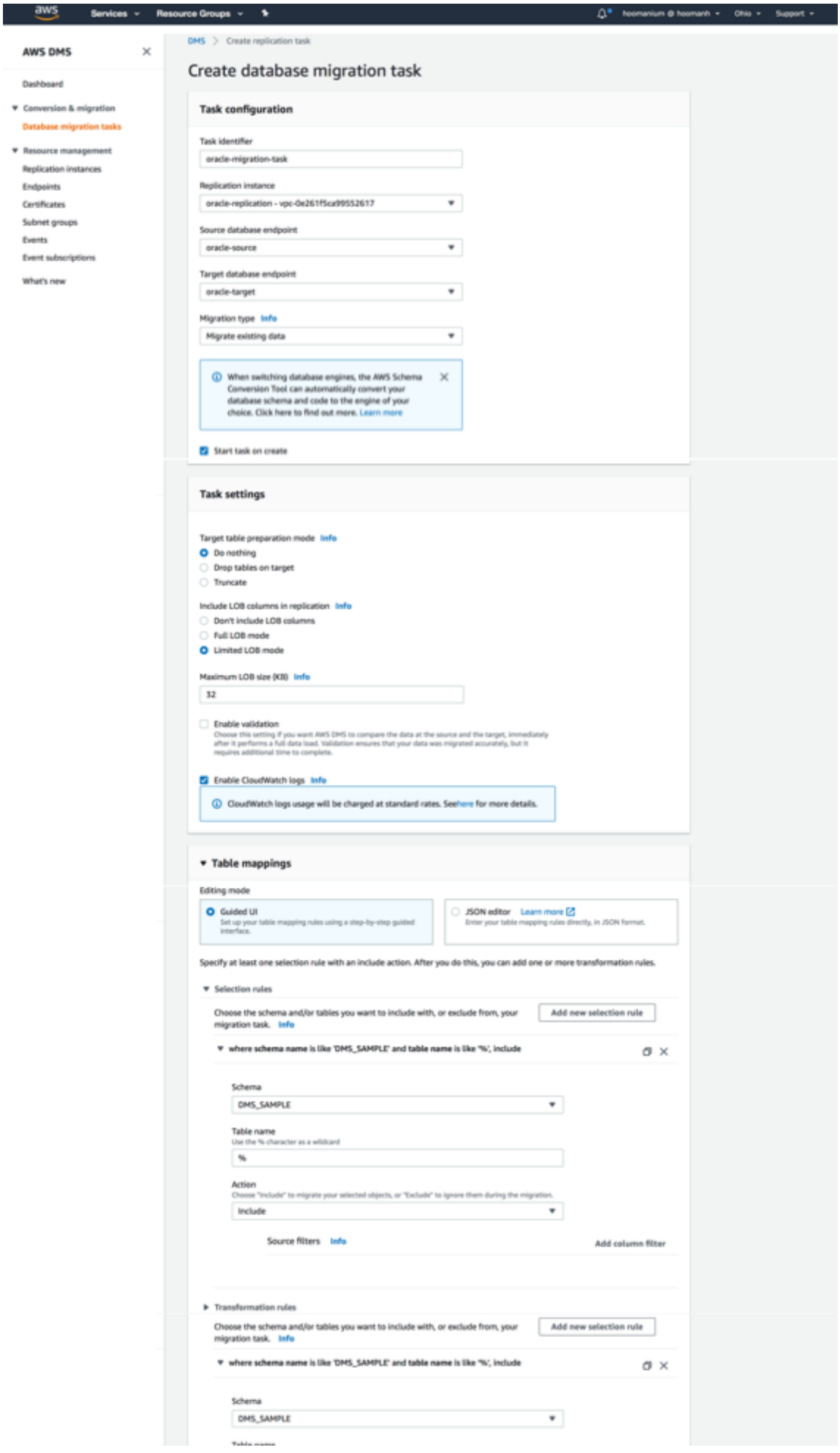
Parameter	Value
Target	Schema
Schema Name	DMS_SAMPLE
Action	Make lowercase

Rule 2:

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

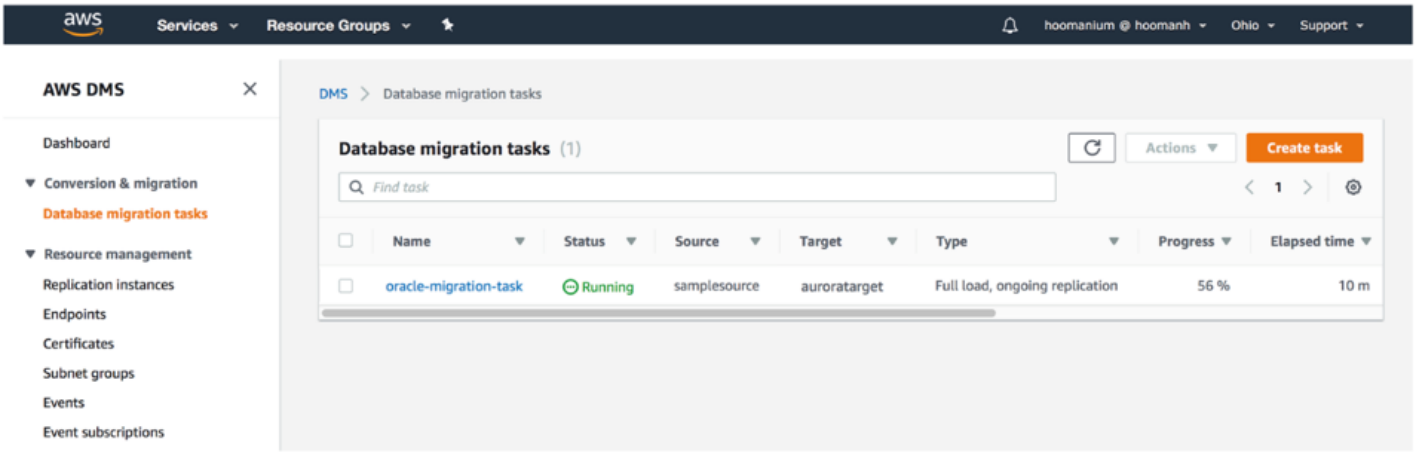
Rule 3:

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


aws
Services
Resource Groups
hoomanium @ hoomanh
Ohio
Support
AWS DMS
Dashboard
Conversion & migration
Database migration tasks
Resource management
Replication instances
Endpoints
Certificates
Subnet groups
Events
Event subscriptions
What's new
DMS > Create replication task
Create database migration task
Task configuration
Task identifier
oracle-migration-task
Replication instance
oracle-replication - vpc-0e261f5ca99552617
Source database endpoint
oracle-source
Target database endpoint
oracle-target
Migration type Info
Migrate existing data
When switching database engines, the AWS Schema Conversion Tool can automatically convert your database schema and code to the engine of your choice. Click here to find out more. Learn more
Start task on create
Task settings
Target table preparation mode Info
Do nothing
Drop tables on target
Truncate
Include LOB columns in replication Info
Don't include LOB columns
Full LOB mode
Limited LOB mode
Maximum LOB size (KB) Info
32
Enable validation
Choose this setting if you want AWS DMS to compare the data at the source and the target, immediately after it performs a full data load. Validation ensures that your data was migrated accurately, but it requires additional time to complete.
Enable CloudWatch logs Info
CloudWatch logs usage will be charged at standard rates. Seehere for more details.
Table mappings
Editing mode
Guided UI
Set up your table mapping rules using a step-by-step guided interface.
JSON editor Learn more
Enter your table mapping rules directly, in JSON format.
Specify at least one selection rule with an include action. After you do this, you can add one or more transformation rules.
Selection rules
Choose the schema and/or tables you want to include with, or exclude from, your migration task. Info
Add new selection rule
where schema name is like 'DMS_SAMPLE' and table name is like '%', include
Schema
DMS_SAMPLE
Table name
Use the % character as a wildcard
%
Action
Choose "Include" to migrate your selected objects, or "Exclude" to ignore them during the migration.
Include
Source filters Info
Add column filter
Transformation rules
Choose the schema and/or tables you want to include with, or exclude from, your migration task. Info
Add new selection rule
where schema name is like 'DMS_SAMPLE' and table name is like '%', include
Schema
DMS_SAMPLE

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

31. At this point, the task should start running and replicating data from the **DMS_SAMPLE** Oracle database to the Amazon Aurora RDS (PostgreSQL) instance.



32. As the rows are being transferred, you can monitor the task progress:
    1. 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.
    2. If there is an error, the status color changes from green to red. Click on View logs link for the logs to debug.

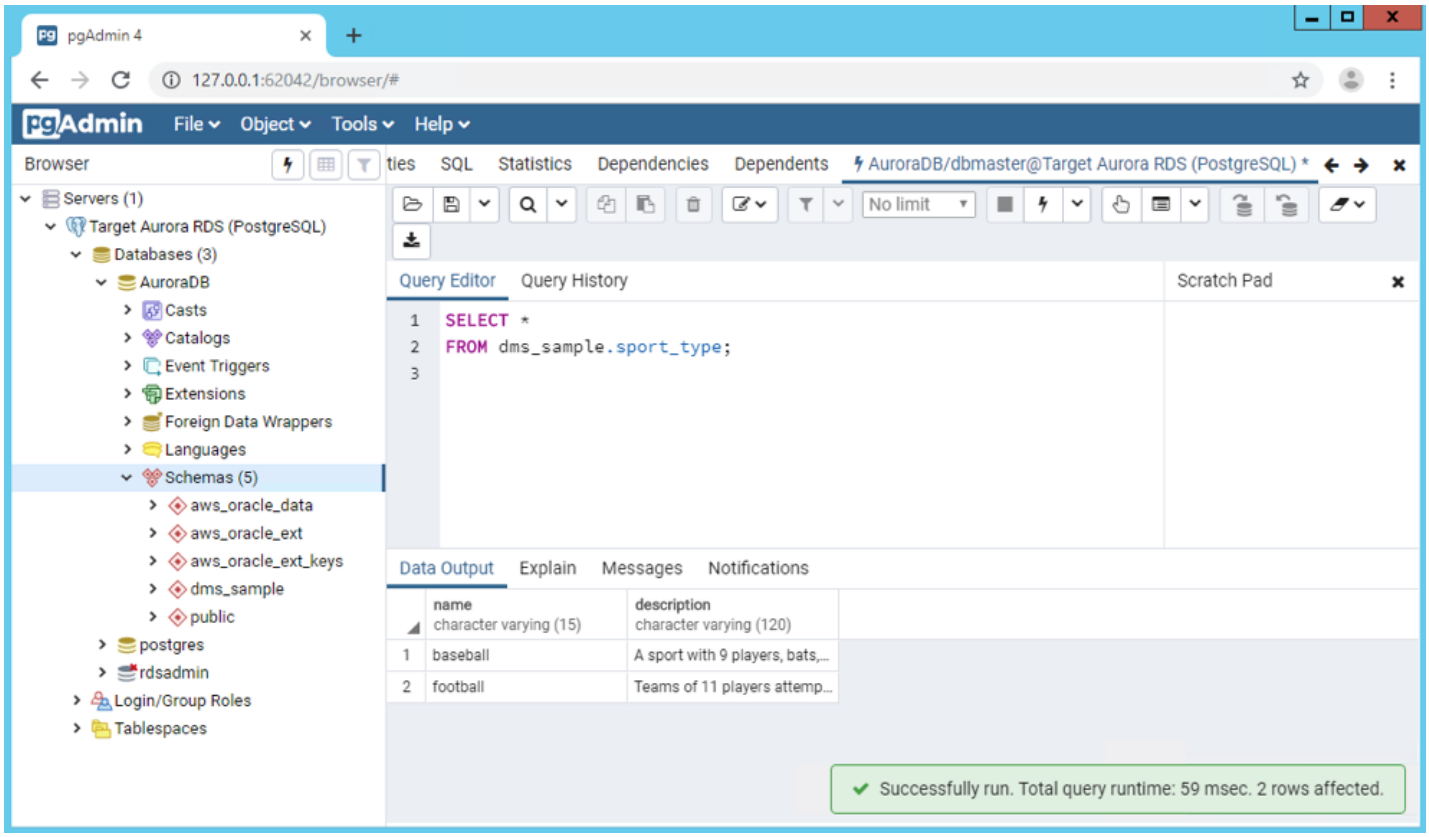
INSPECT THE CONTENT OF TARGET DATABASE

Tip

If you disconnected from the EC2 instance, follow the instruction in Connect to the EC2 Instance section from the previous part to RDP to the instance.

33. Open pgAdmin4 from within the EC2 server, and then connect to the Target Aurora RDS (PostgreSQL) database connection that you created earlier.
34. 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;
```



Note

Baseball, and football are the only two sports that are currently listed in this table. In the next section you will insert several new records to the source database with information about other sport types. DMS will automatically replicate these new records from the source database to the target database.

35. Now, use the following script to enable the foreign key constraints that we dropped earlier:
    1. Open AddConstraintsPostgreSQL.sql in your favorite text editor.
    2. Copy the content of the file to the **Query Editor** in **pgAdmin 4**.
    3. **Execute** the script.

REPLICATE DATA CHANGES

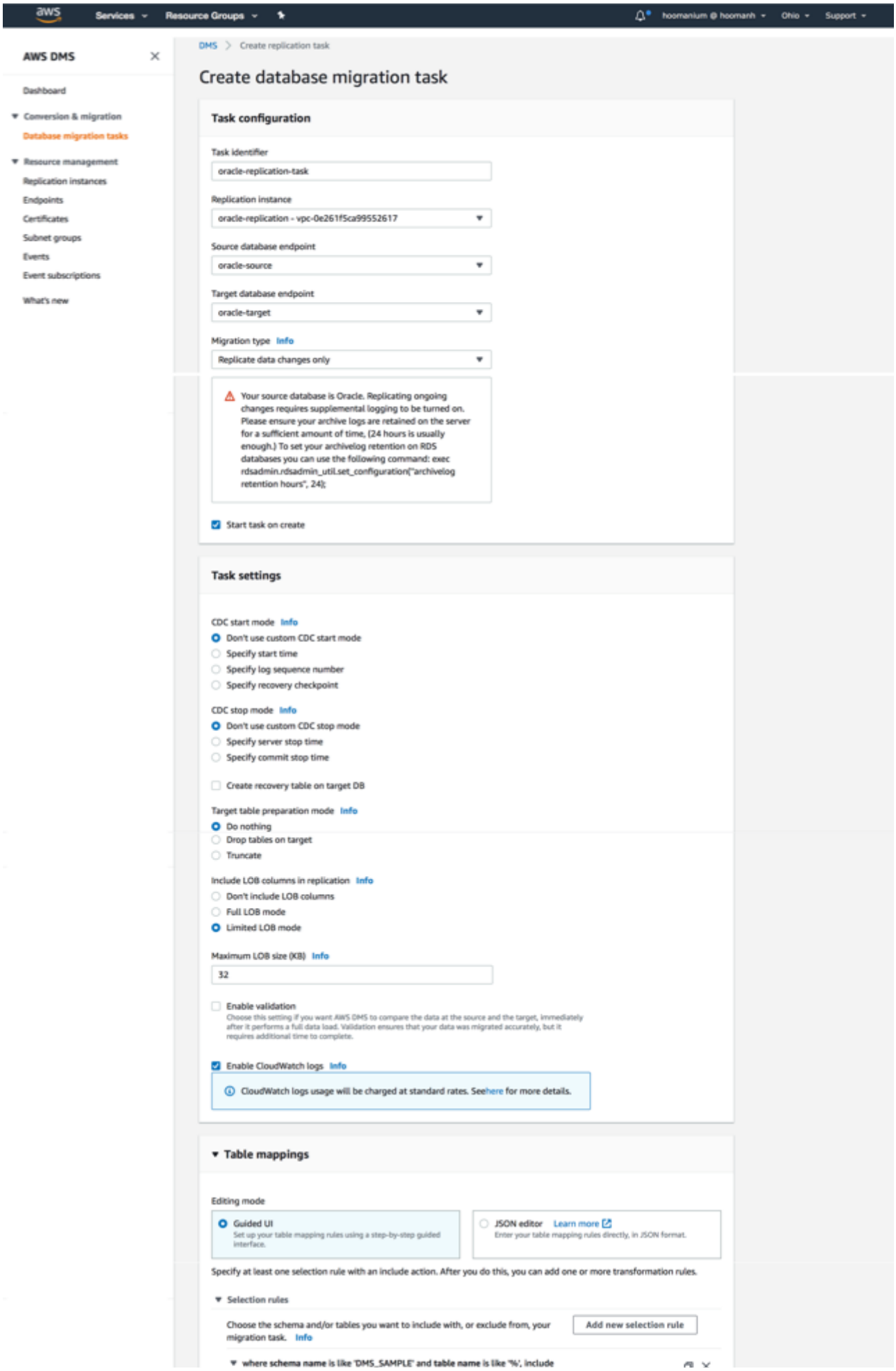
Now you are going to simulate a transaction to the source database by updating the **sport_type** table. The Database Migration Service will automatically detect and replicate these changes to the target database.

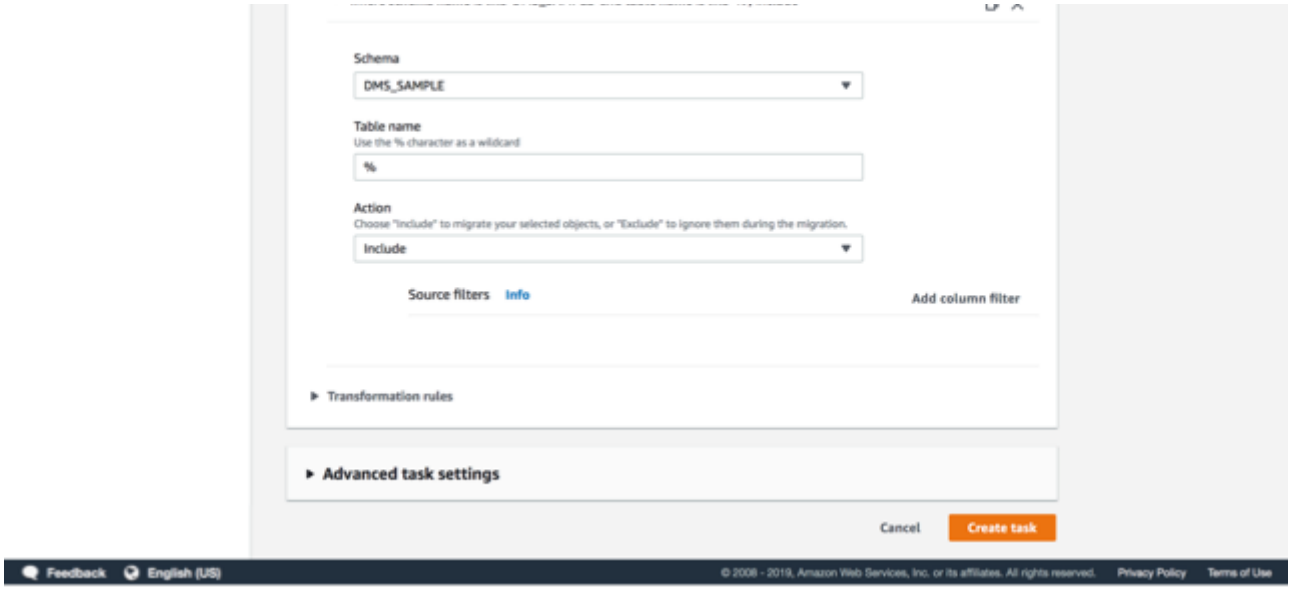
36. Create another **Data Migration Task** with the following values for capturing data changes to the source Oracle database, and replicating the changes to the target Aurora RDS instance.

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

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

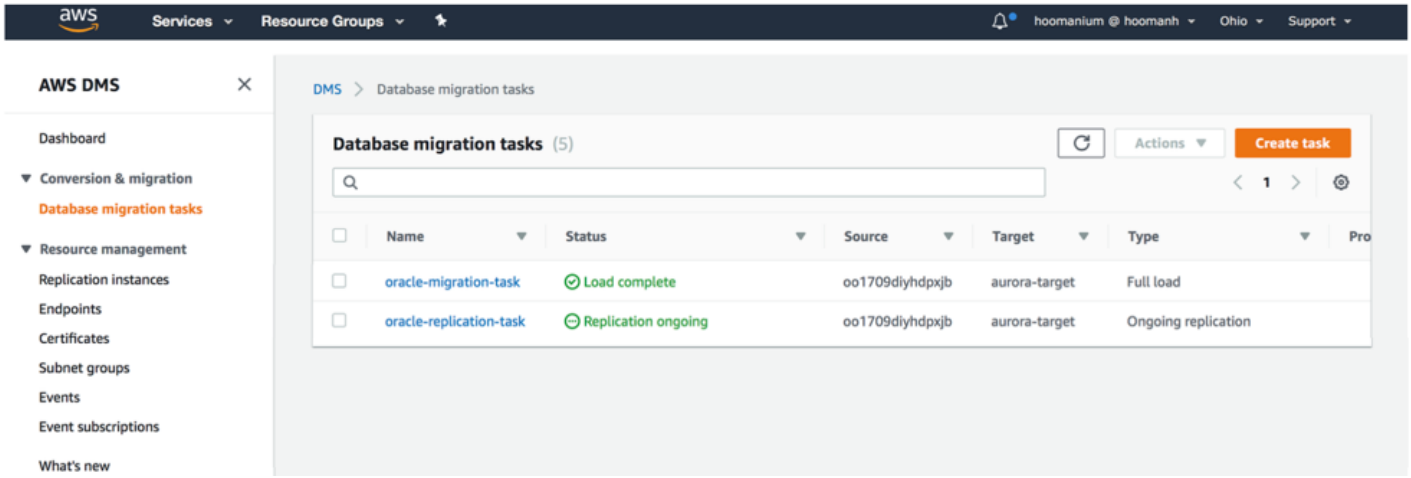
38. Add the same **Selection**, and **Transformation** rules as specified in steps 20, and 21 that we described earlier.


aws
Services
Resource Groups
hoomanium @ hoomanh
Ohio
Support
AWS DMS
Dashboard
Conversion & migration
Database migration tasks
Resource management
Replication instances
Endpoints
Certificates
Subnet groups
Events
Event subscriptions
What's new
DMS
Create replication task
Create database migration task
Task configuration
Task identifier
oracle-replication-task
Replication instance
oracle-replication - vpc-0e261f5ca99552617
Source database endpoint
oracle-source
Target database endpoint
oracle-target
Migration type Info
Replicate data changes only
Your source database is Oracle. Replicating ongoing changes requires supplemental logging to be turned on. Please ensure your archive logs are retained on the server for a sufficient amount of time, (24 hours is usually enough.) To set your archivelog retention on RDS databases you can use the following command: exec rdsadmin.rdsadmin_util.set_configuration("archivelog retention hours", 24);
Start task on create
Task settings
CDC start mode Info
Don't use custom CDC start mode
Specify start time
Specify log sequence number
Specify recovery checkpoint
CDC stop mode Info
Don't use custom CDC stop mode
Specify server stop time
Specify commit stop time
Create recovery table on target DB
Target table preparation mode Info
Do nothing
Drop tables on target
Truncate
Include LOB columns in replication Info
Don't include LOB columns
Full LOB mode
Limited LOB mode
Maximum LOB size (KB) Info
32
Enable validation
Choose this setting if you want AWS DMS to compare the data at the source and the target, immediately after it performs a full data load. Validation ensures that your data was migrated accurately, but it requires additional time to complete.
Enable CloudWatch logs Info
CloudWatch logs usage will be charged at standard rates. Seehere for more details.
Table mappings
Editing mode
Guided UI
Set up your table mapping rules using a step-by-step guided interface.
JSON editor Learn more
Enter your table mapping rules directly, in JSON format.
Specify at least one selection rule with an include action. After you do this, you can add one or more transformation rules.
Selection rules
Choose the schema and/or tables you want to include with, or exclude from, your migration task. Info
Add new selection rule
where schema name is like 'DMS_SAMPLE' and table name is like '%', include



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

40. 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.

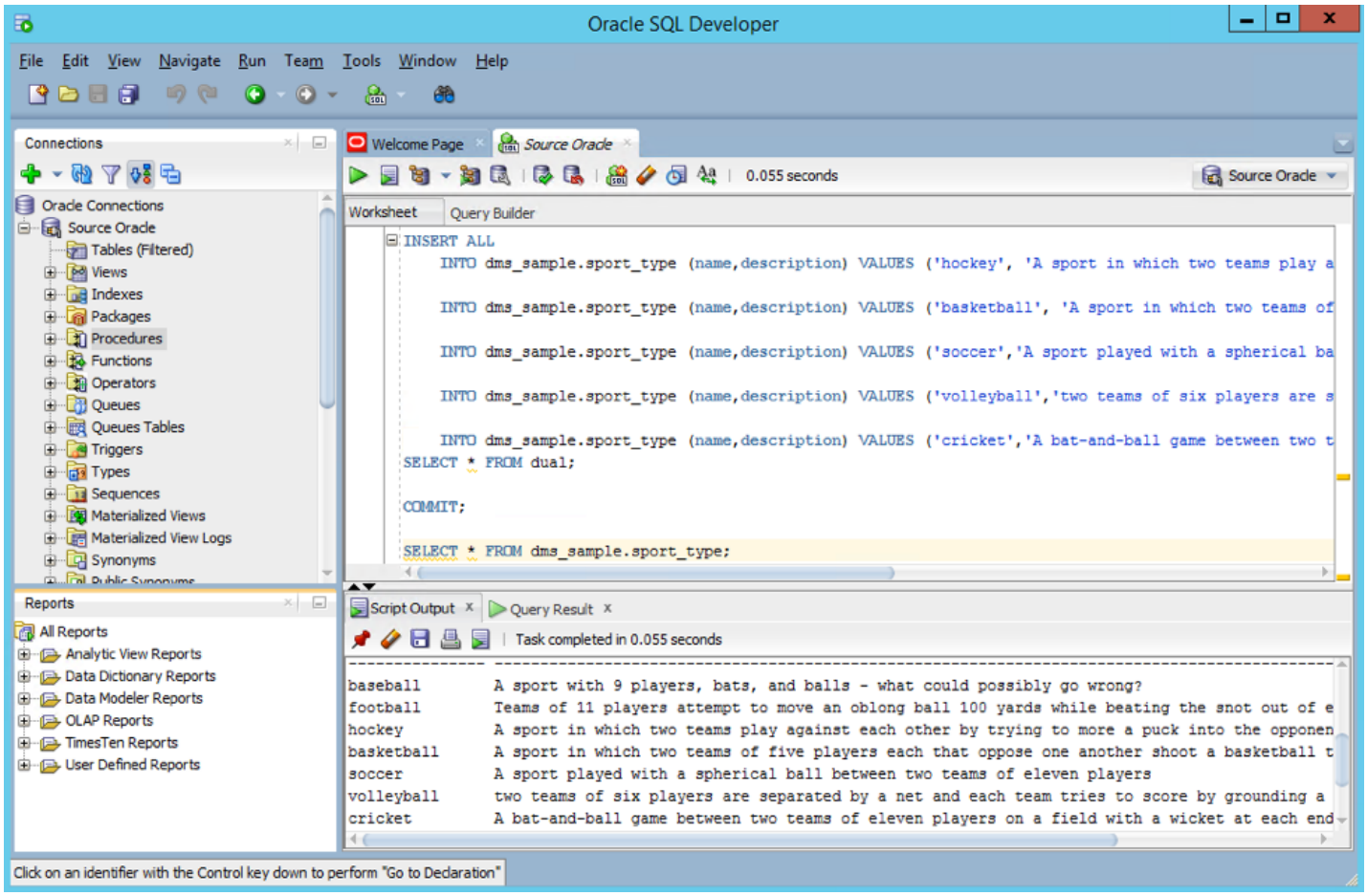


Now you are going to simulate a transaction to the source database by updating the **sport_type** table. The Database Migration Service will automatically detect and replicate these changes to the target database.

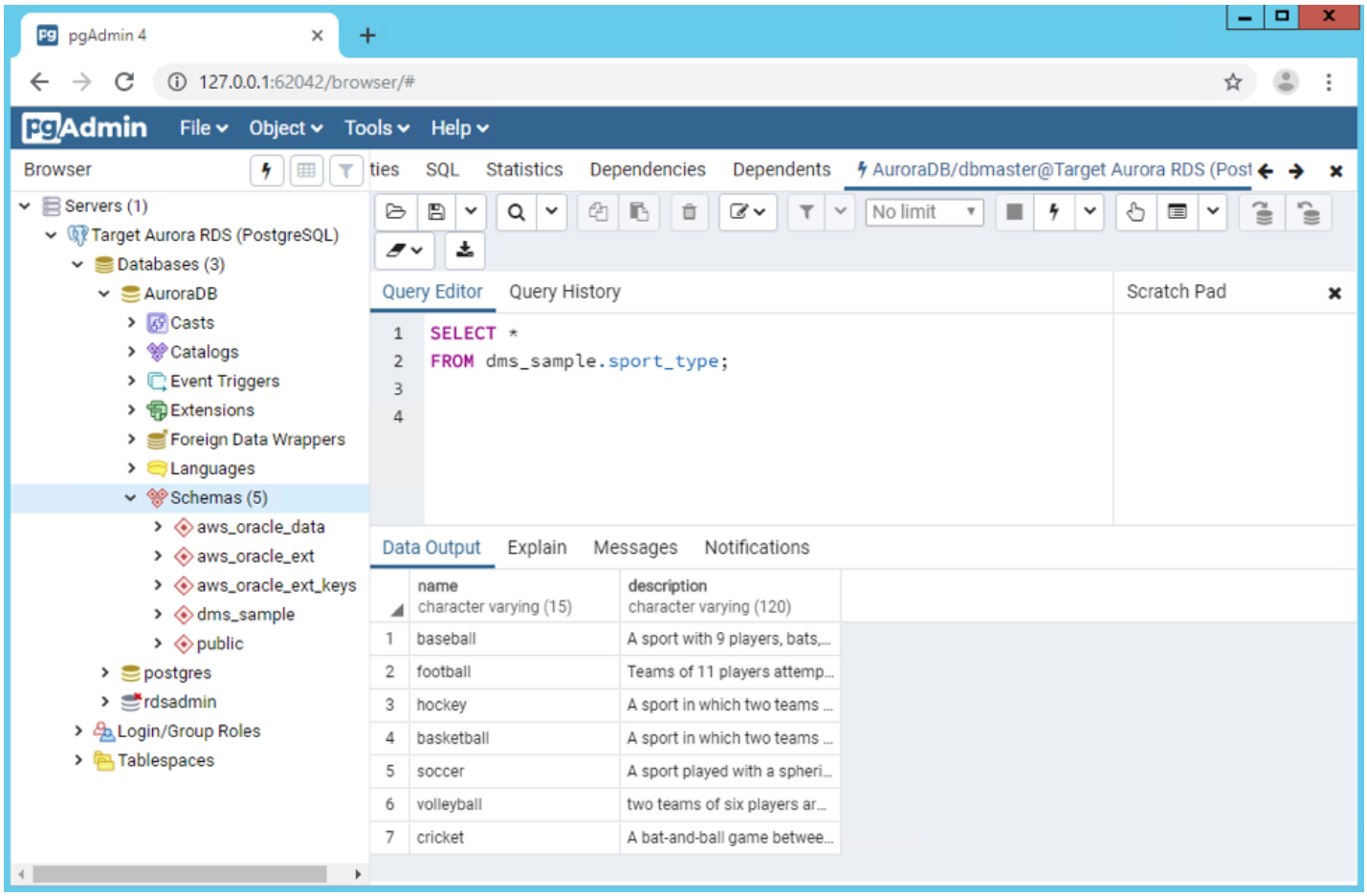
41. Use **Oracle SQL Developer** connect to the source Oracle RDS.

42. 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;
```



43. Repeat steps 33 and 34 as described earlier to inspect the content of **sport_type** table in the target database.



Note

The new records for that you added for basketball, cricket, hockey, soccer, volleyball to the **sports_type** table in the source database have been replicated to your **dms_sample** 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.

Info

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

In the first part of this tutorial we saw how easy it is to convert the database schema from an Oracle database into Amazon Aurora (PostgreSQL) using the AWS Schema Conversion Tool (AWS SCT). In the second part, we used the AWS Database Migration Service (AWS DMS) to migrate the data from our source to target database with no downtime. Similarly, we 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.

Warning

The resources provisined as part of this workshop will incur charges. Follow the instructions in the Environment Cleanup section to remove these resources.