Database Migration Workshop

AWS Schema Conversion Tool (SCT)

Oracle to Aurora PostgreSQL

July 2019

Rev July-02-2019

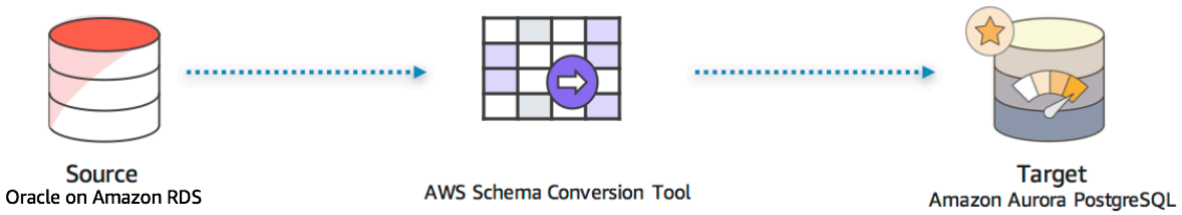
Table of Contents

Overview 3
Connecting to your Environment 4
Schema Conversion Tool (CST) 5
Log on the EC2 instance 5
Install the Schema Conversion Tool on the server 6
Create a Database Migration Project 8
Convert the Schema Using the Schema Conversion Tool 13
Summary 18

Overview

AWS Schema Conversion Tool (SCT) helps you convert your commercial database and data warehouse schemas to open-source engines or AWS-native services such as Amazon Aurora and Redshift. In addition, it can move your table DDL, views, and stored procedure DML to a different platform. The tool generates an assessment report which lists the objects that can be automatically converted and recommends manual changes were needed.

In this lab you will use the AWS SCT to convert an Oracle schema to Amazon Aurora PostgreSQL. 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.



Connecting to your Environment

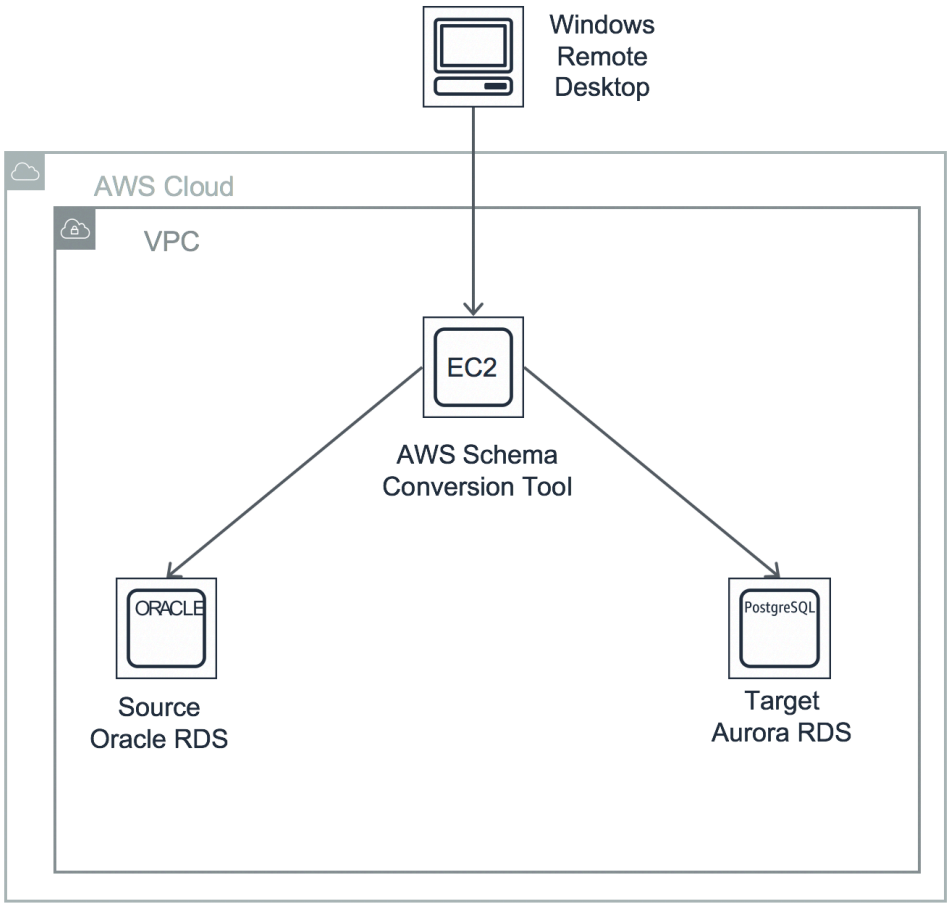
Before launching this lab, make sure you have completed the instructions in the lab environment guide: https://bit.ly/2xL3SMw

The environment for this lab consists of:

- An EC2 instance used to run the AWS Schema Conversion Tool (SCT).
- 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



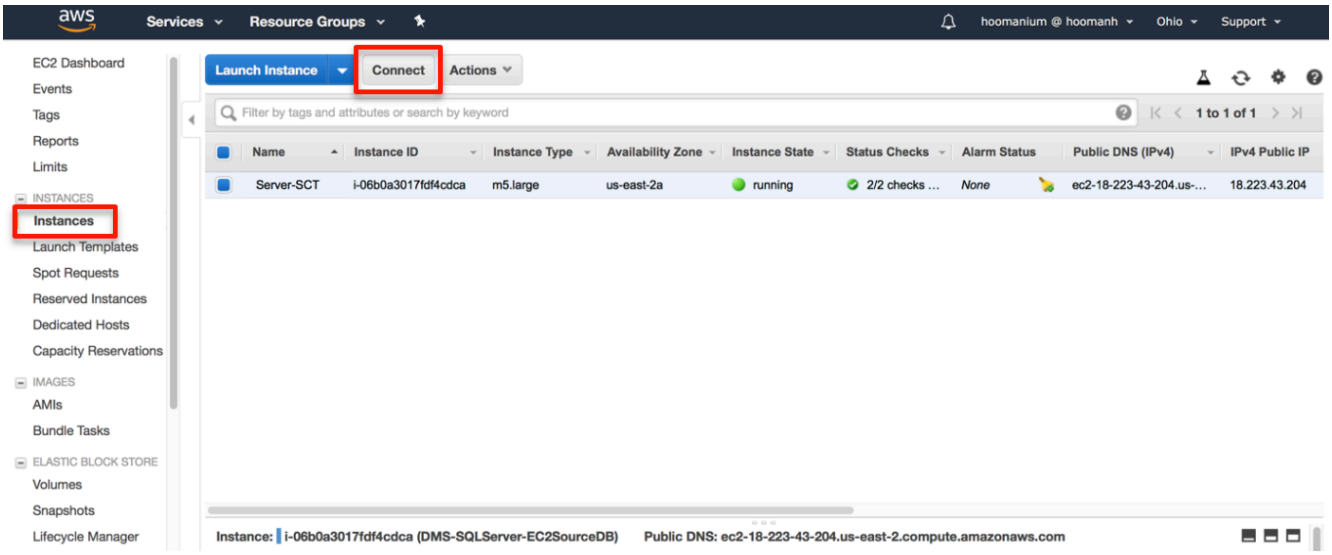
Schema Conversion Tool (CST)

The following steps provide instructions for converting an Oracle database to an Amazon Aurora PostgreSQL database. In this exercise, you perform the following tasks:

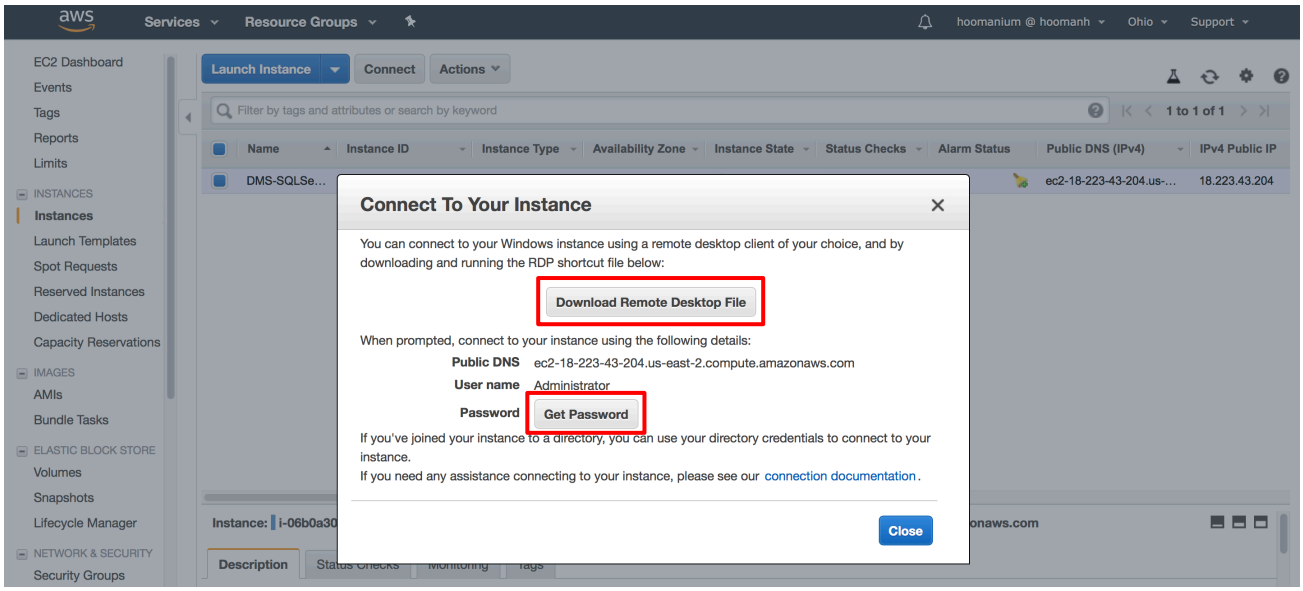
- Log on the Windows EC2 instance
- Install the Schema Conversion Tool on the server
- Use the Schema Conversion Tool to create a database migration project
- Use SCT to convert the Oracle schema to PostgreSQL schema and analyze schema conversion issues
- Apply the converted schema to the Aurora PostgreSQL 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 **<StackName>-EC2Instance** 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.



Install the Schema Conversion Tool on the server

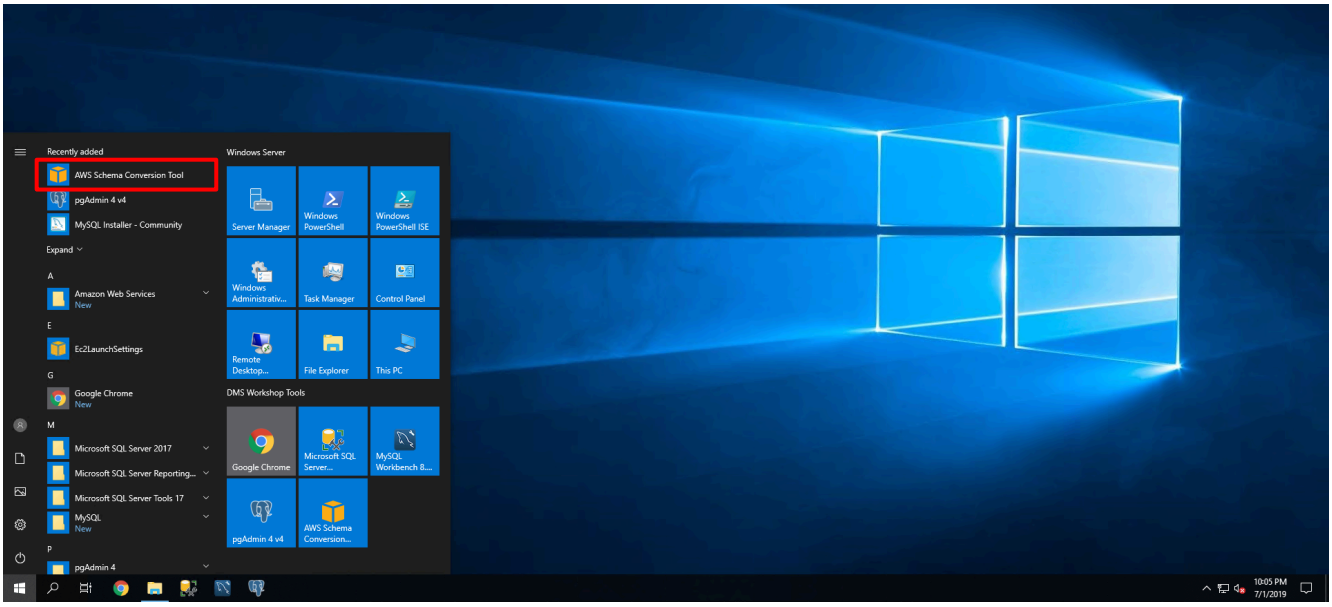
Now that you are connected to the EC2 instance, you are going to install the AWS Schema Conversion tool on the server. Downloading the file and installing it will give you the latest version of the AWS Schema Conversion Tool.

4. On the EC2 server, open the **DMS Workshop** folder. Then, double-click on **AWS Schema Conversion Tool Download** to get the latest version of the software.

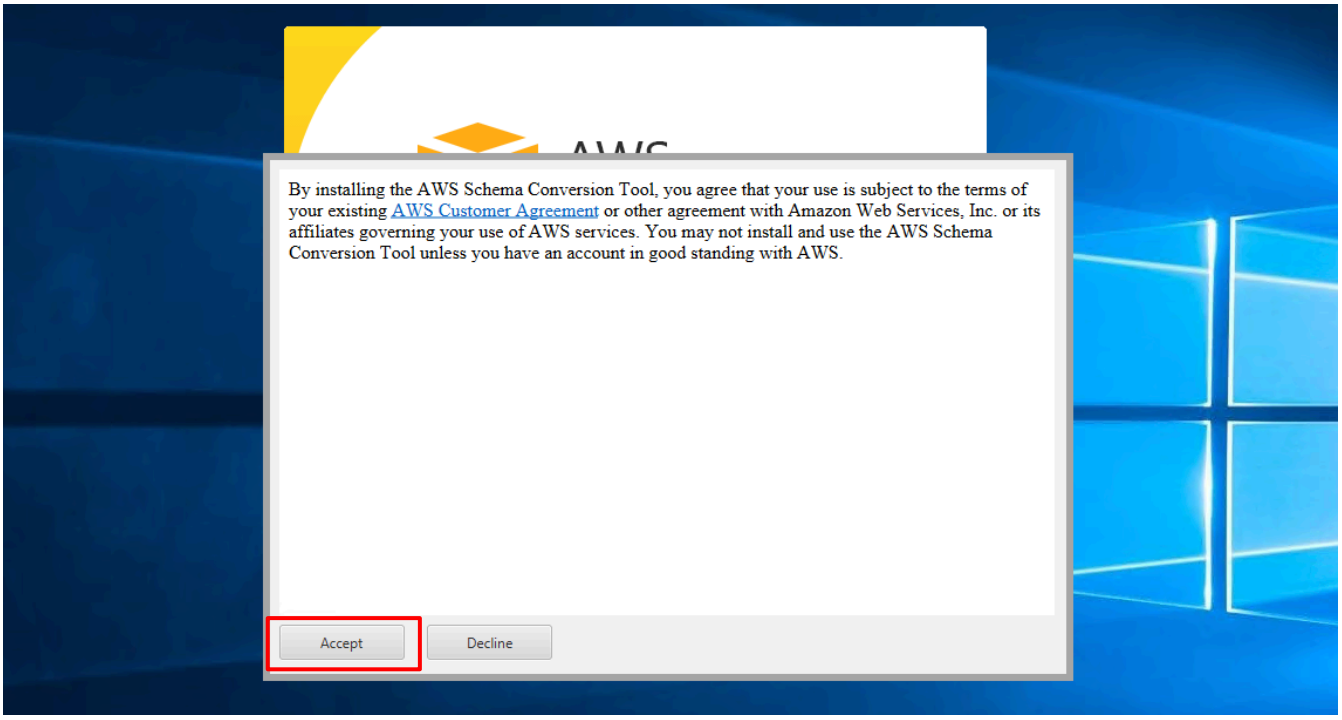
5. When the download is complete, unzip the content and install the AWS Schema Conversion Tool.

    NOTE: When the installer is complete the installation dialog will disappear. There is no other notification.

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



7. **Accept** the terms and Conditions.

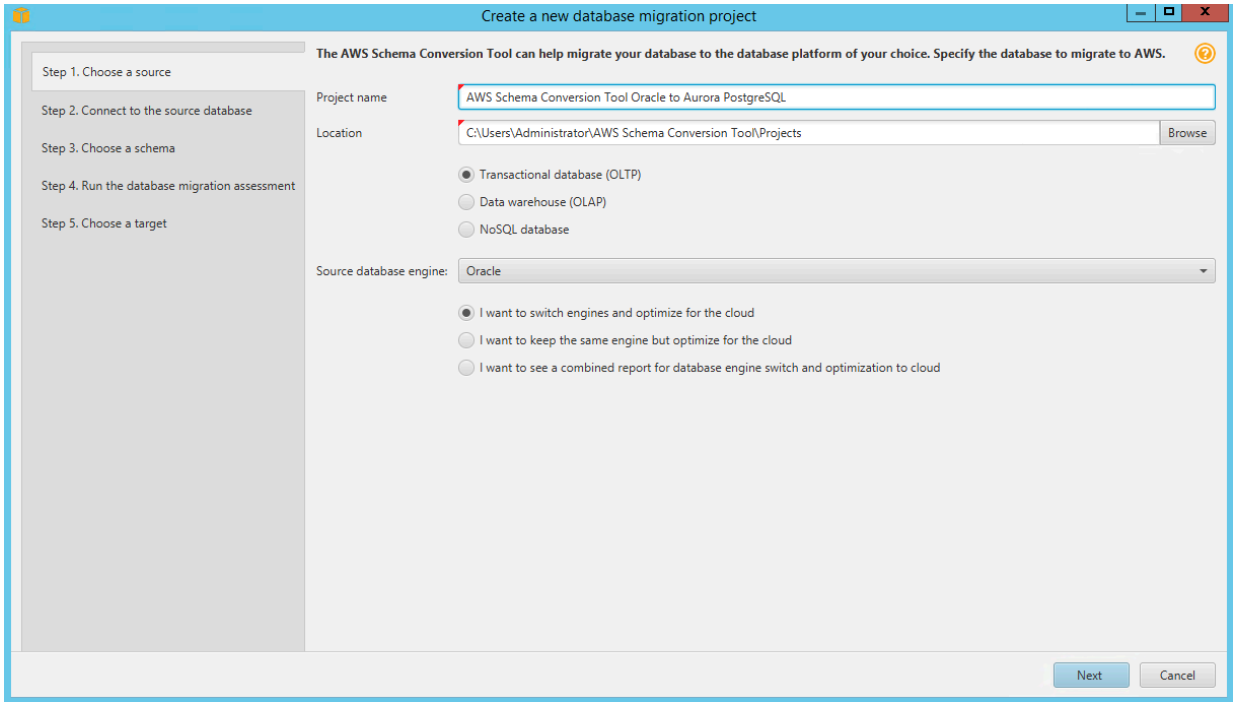


Create a Database Migration Project

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

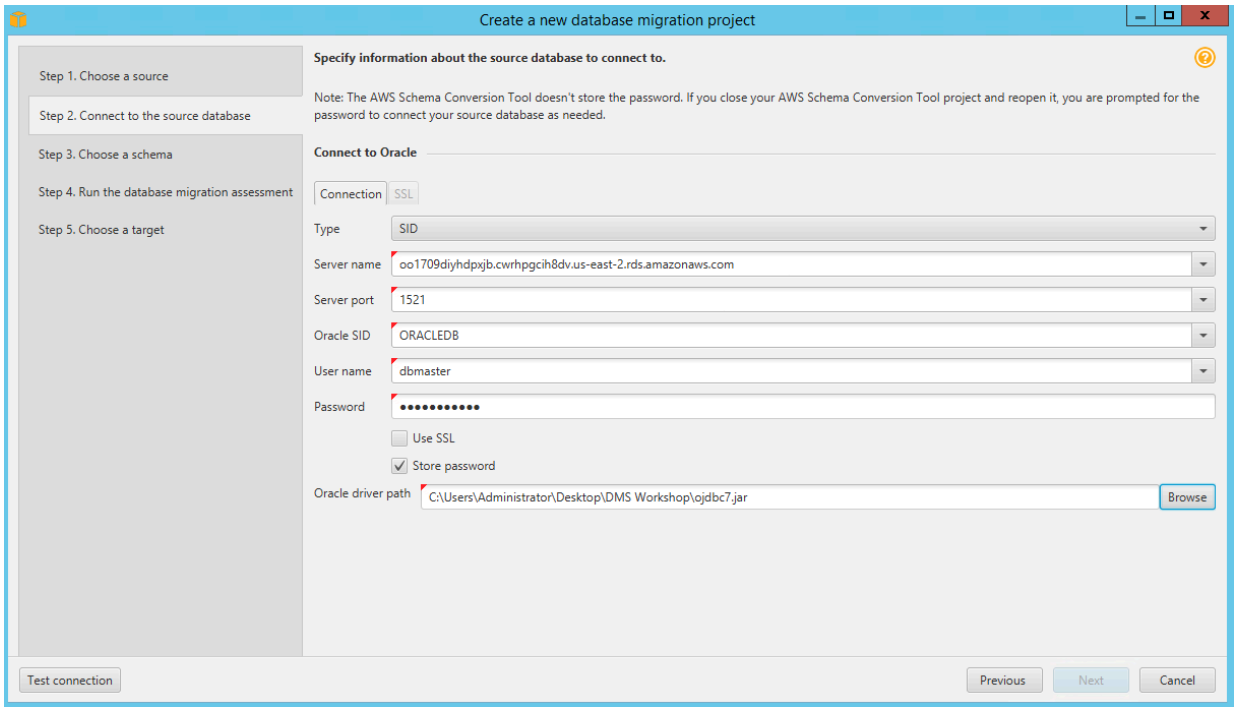
8. With the Schema Conversion Tool, enter the following values into the form and then click **Next.**

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

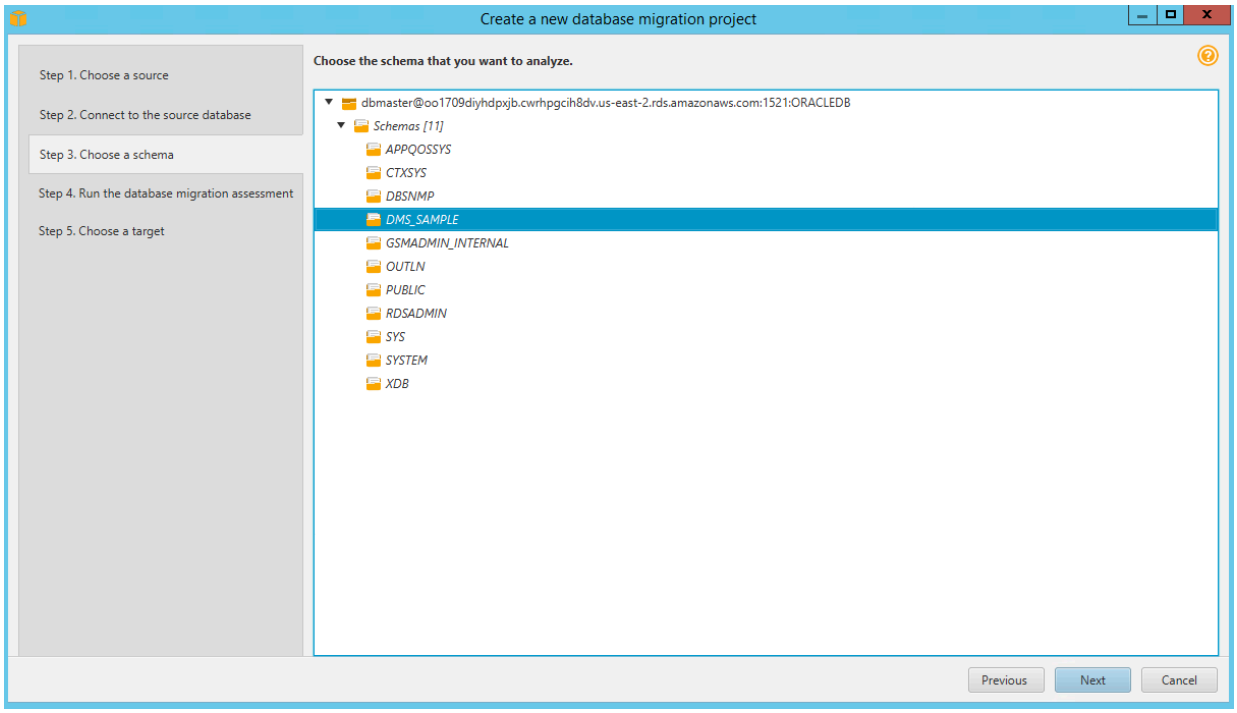


9. Specify the source database configurations in the form, and click **Test Connection**. Once the connection is successfully tested, click **Next.**

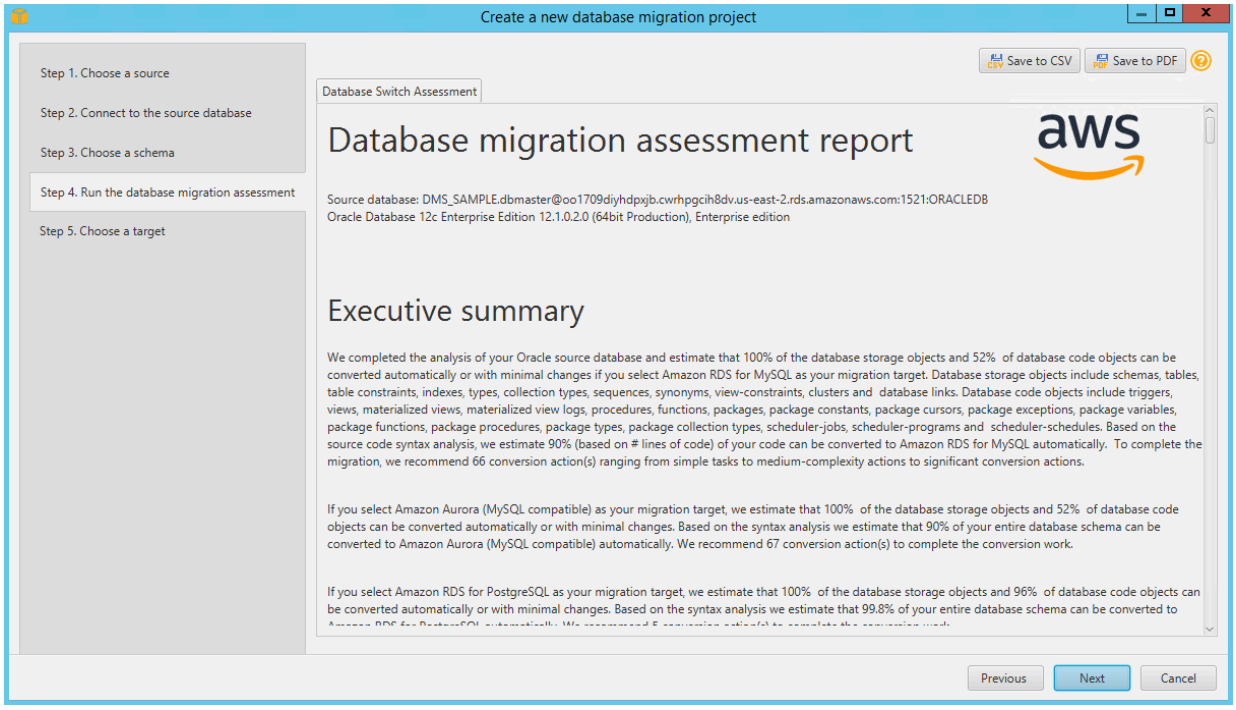
Parameter	Value
Type	SID
Server Name	**< SourceOracleEndpoint >**
Server Port	1521
Oracle SID	ORACLEDB
User Name	dbmaster
Password	dbmaster123
Use SSL	Unchecked
Save Password	Checked
Oracle Driver Path	C:\Users\Administrator\Desktop\DMS Workshop\JDBC\ojdbc7.jar



10. Select the **DMS_SAMPLE** database, then click **Next**.



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



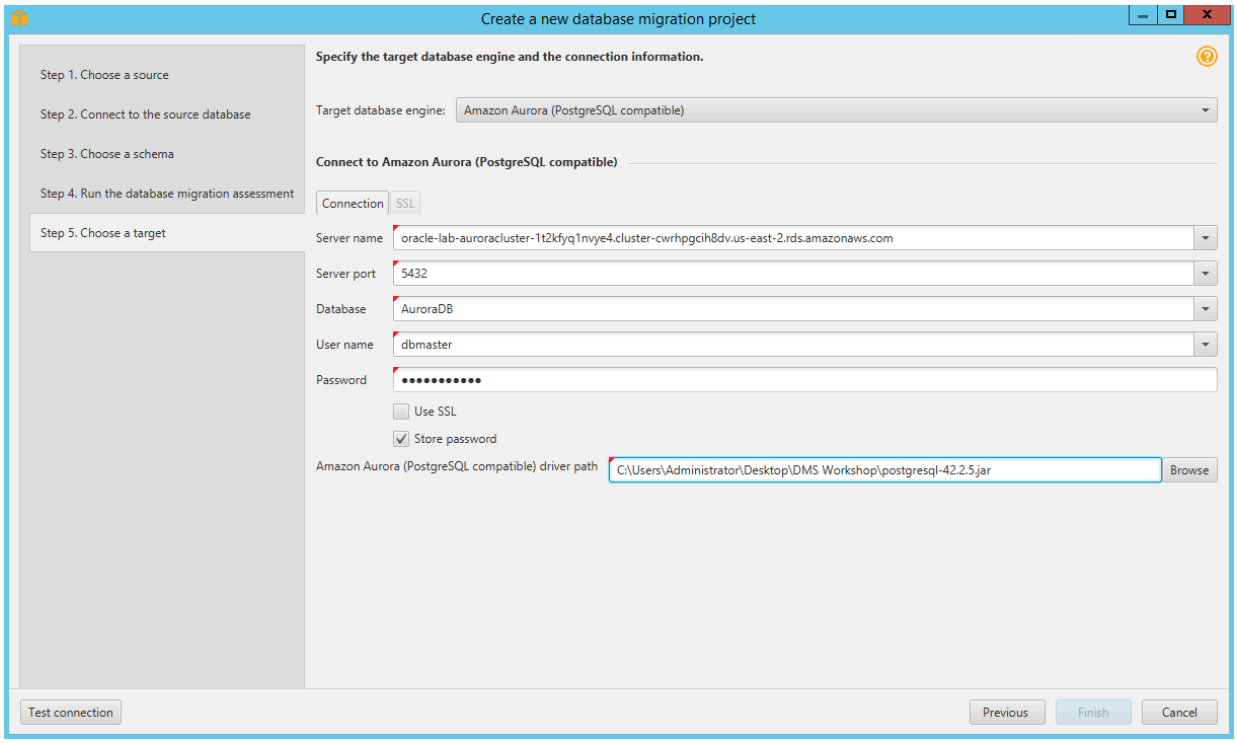
SCT will examine in detail all of the objects in the schema of the source database. It will convert as much as possible automatically and provides detailed information about items it could not convert.

Generally, packages, procedures, and functions are more likely to have conversion issues because they contain the most custom PL/SQL code. AWS SCT specifies how much manual change is needed for each object, and provides hints on successfully converting the objects to the target schema.

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

13. Next, 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	AuroraDB
User Name	dbmaster
Password	dbmaster123
Use SSL	Unchecked
Save Password	Checked
Amazon Aurora Driver Path	C:\Users\Administrator\Desktop\DMS Workshop\JDBC\postgresql-42.2.6.jar



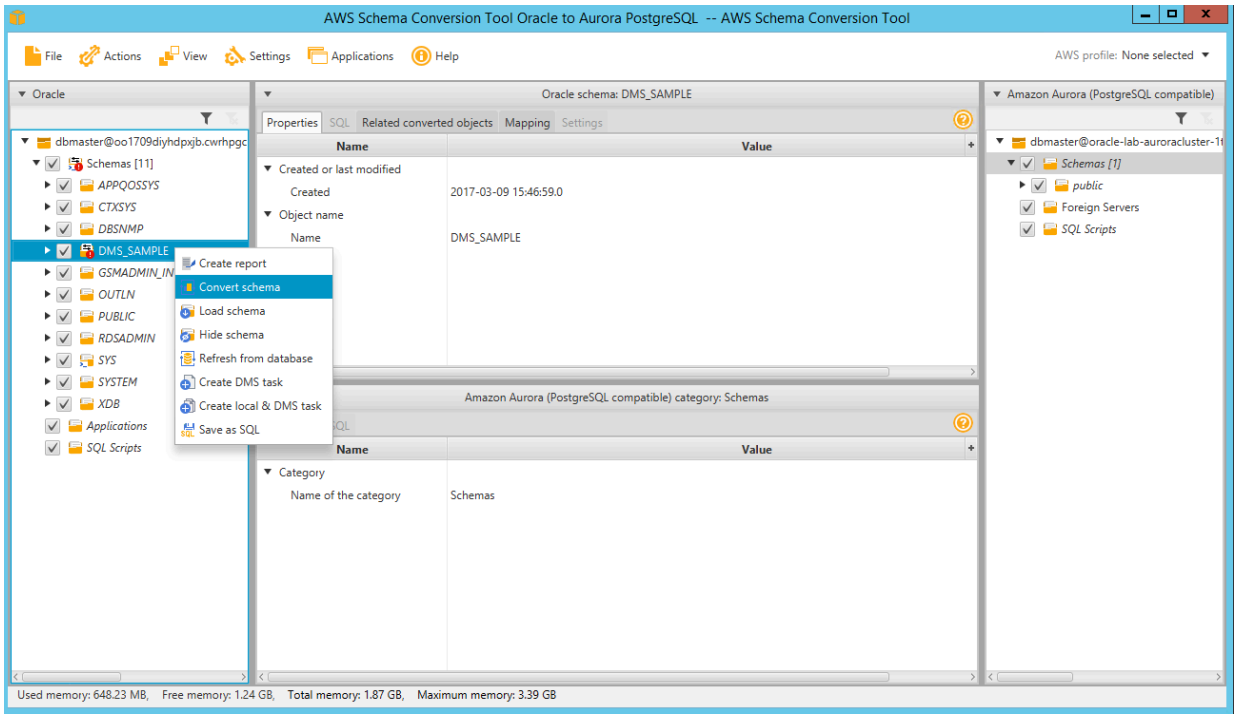
Convert the Schema Using the Schema Conversion Tool

Now that you have created a new Database Migration Project, the next step is to convert the Oracle schema of the source database to that of Amazon Aurora PostgreSQL database.

14. 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.

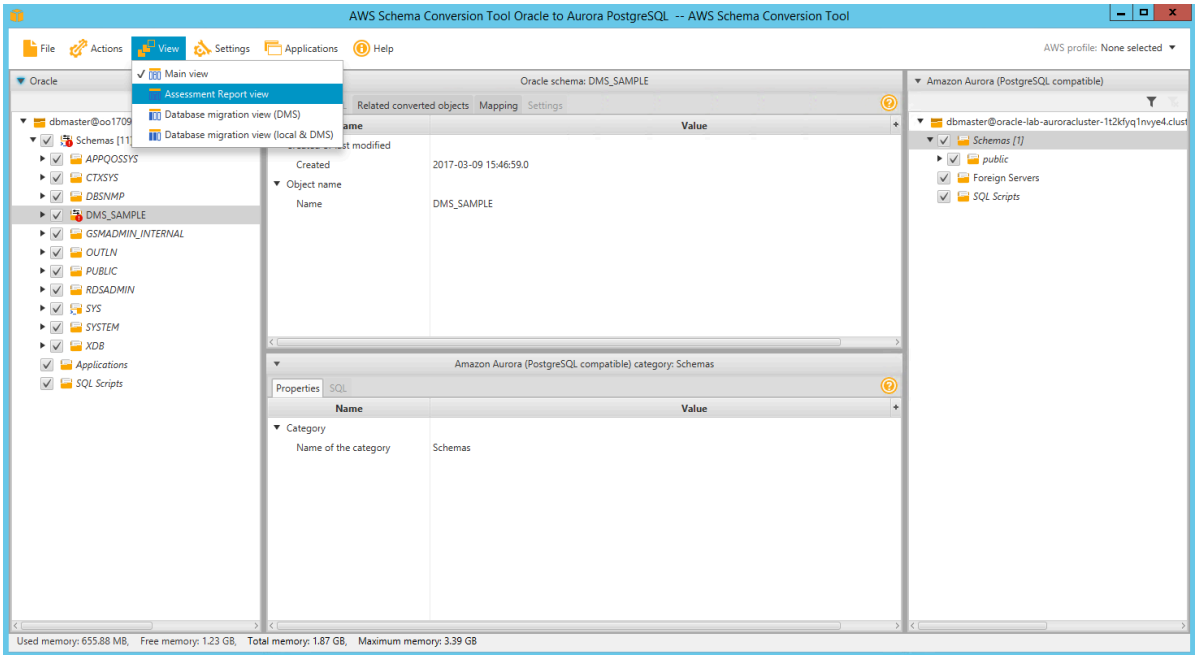
    You can view the generated DDL in the project console, and edit it before applying it to the target database. You can also choose to save it as an .sql file for application later.

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

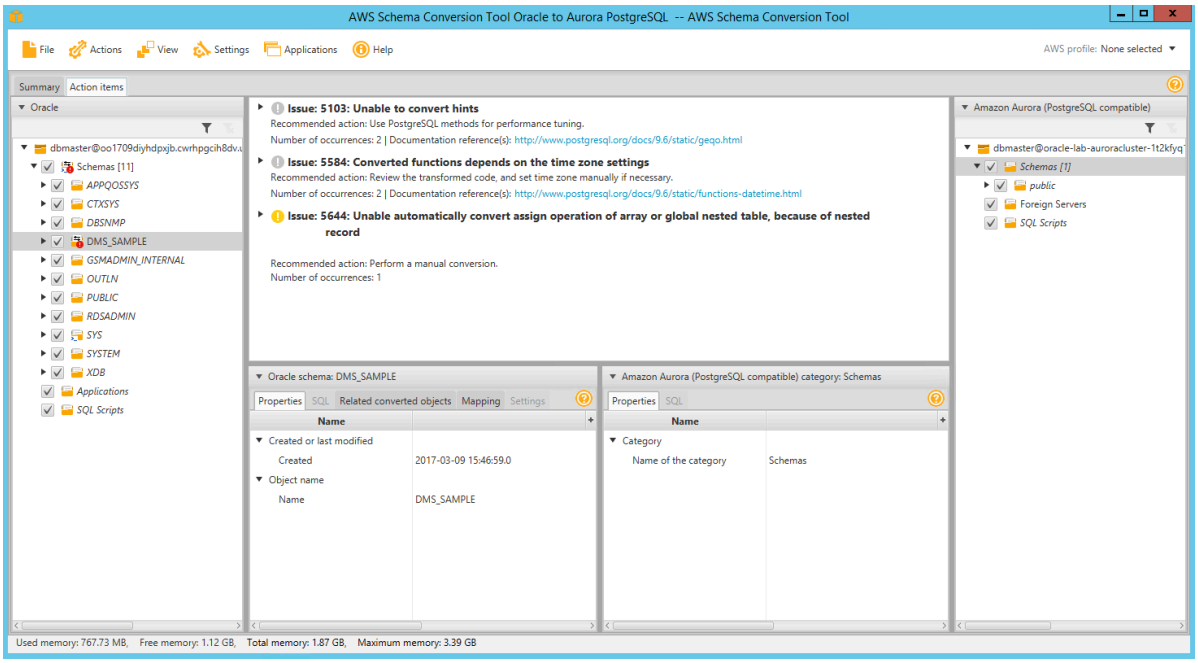


AWS SCT analyses the schema and creates a database migration assessment report for the conversion to PostgreSQL. Items with a red exclamation mark next to them cannot be directly translated from the source to the target. This includes Stored Procedures, and Packages.

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



16. Next, navigate to the **Action Items** tab in the report to see the items that have issues converting to the new schema.

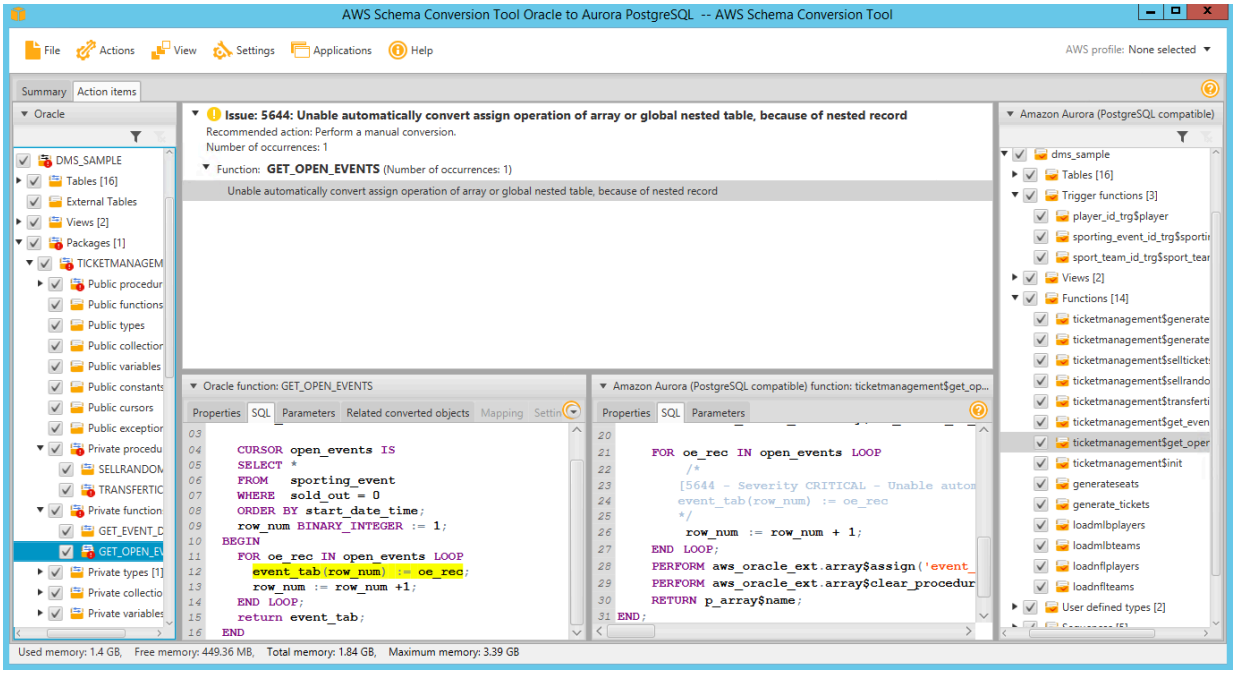


Check each of the issues listed and compare the contents under the source Oracle panel and the target Aurora PostgreSQL panel. Are the issues resolved? And how?

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.

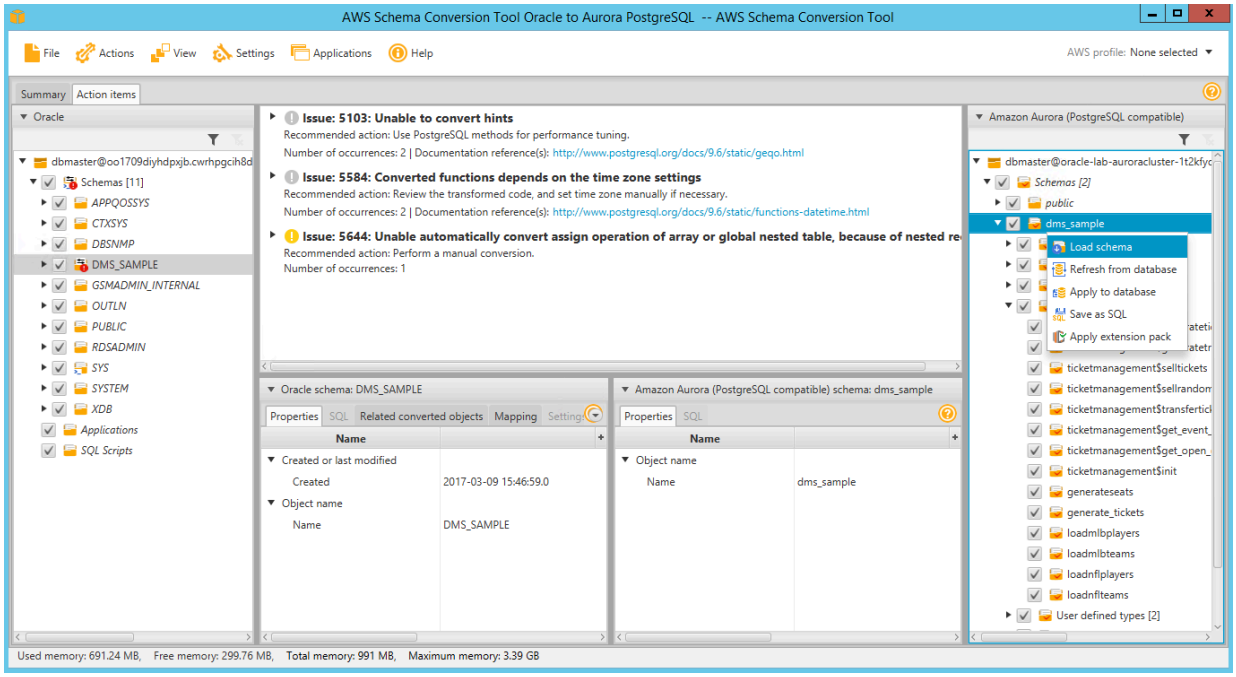
Notice the issue highlighted in the private function named **GET_OPEN_EVENTS**. You'll see that SCT is unable to automatically convert the assign operation. You can complete one of the following actions to fix the issue:

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

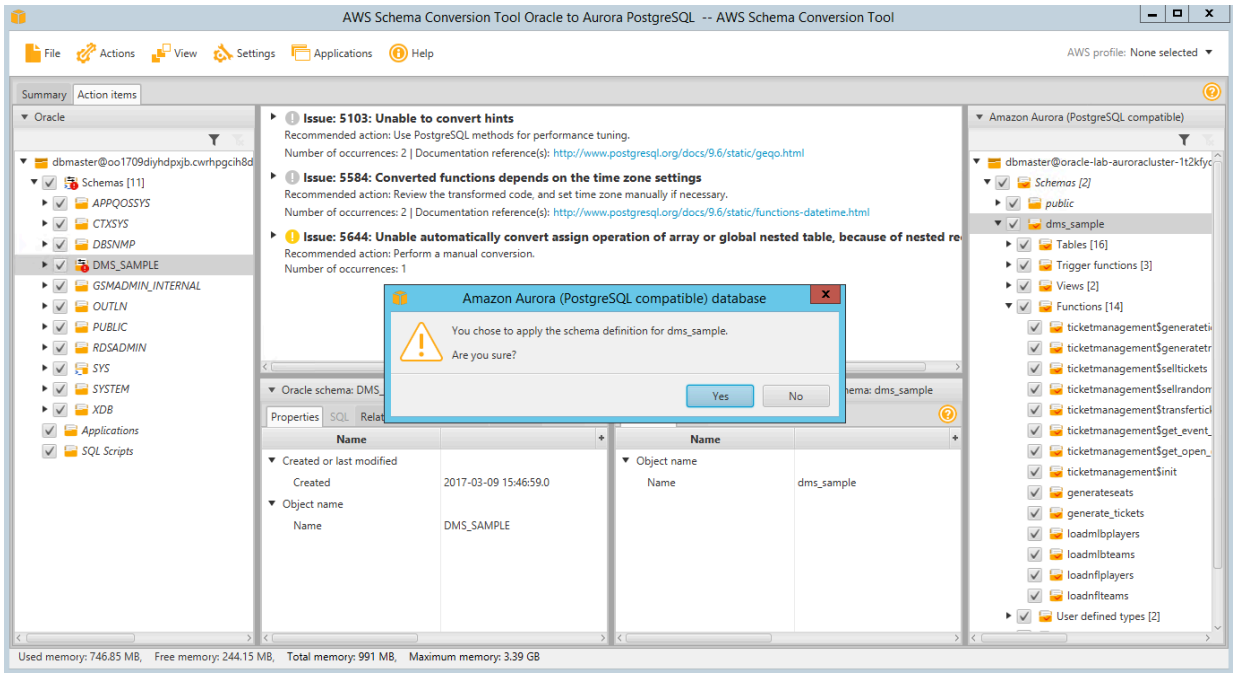


17. [Optional] Manually fix the schema issue. Then, right-click on the **DMS_SAMPLE** schema, and choose **Create Report**. Observe that the schema of the source database is now fully compatible with the target database.

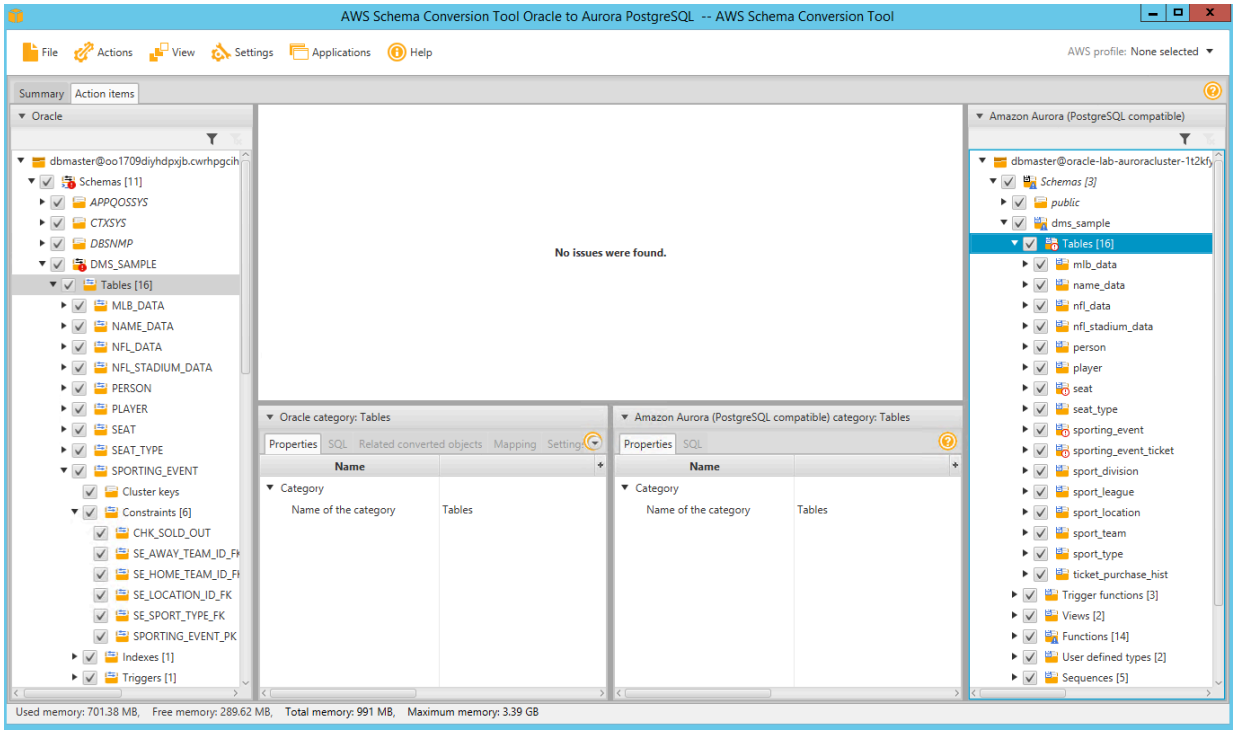
18. Right click on the **dms_sample** schema in the right-hand panel, and click **Apply to database**.



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



20. 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 lab we will drop foreign key target database.

Summary

The first part of the lab demonstrated how easy it is to migrate the schema of an Oracle database into Amazon Aurora (PostgreSQL) using the AWS Schema Conversion Tool (SCT). 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.

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