

Migrating an Amazon RDS for Oracle Database to Amazon Redshift

martes, 22 de junio de 2021 23:02

Migrating an Amazon RDS for Oracle Database to Amazon Redshift

This walkthrough gets you started with heterogeneous database migration from Amazon RDS or Oracle to Amazon Redshift using AWS Database Migration Service (AWS DMS) and the AWS Schema Conversion Tool (AWS SCT). This introductory exercise doesn't cover all scenarios but provides you with a good understanding of the steps involved in such a migration.

It is important to understand that AWS DMS and AWS SCT are two different tools and serve different needs. They don't interact with each other in the migration process. At a high level, the steps involved in this migration are the following:

1. Using the AWS SCT to do the following:
 1. Run the conversion report for Oracle to Amazon Redshift to identify the issues, limitations, and actions required for the schema conversion.
 2. Generate the schema scripts and apply them on the target before performing the data load by using AWS DMS. AWS SCT performs the necessary code conversion for objects like procedures and views.
2. Identify and implement solutions to the issues reported by AWS SCT.
3. Disable foreign keys or any other constraints that might impact the AWS DMS data load.
4. AWS DMS loads the data from source to target using the Full Load approach. Although AWS DMS is capable of creating objects in the target as part of the load, it follows a minimalistic approach to efficiently migrate the data so that it doesn't copy the entire schema structure from source to target.
5. Perform postmigration activities such as creating additional indexes, enabling foreign keys, and making the necessary changes in the application to point to the new database.

This walkthrough uses a custom AWS CloudFormation template to create RDS DB instances for Oracle and Amazon Redshift. It then uses a SQL command script to install a sample schema and data onto the RDS Oracle DB instance that you then migrate to Amazon Redshift.

This walkthrough takes approximately two hours to complete. Be sure to follow the instructions to delete resources at the end of this walkthrough to avoid additional charges.

Prerequisites

The following prerequisites are also required to complete this walkthrough: Familiarity with Amazon RDS, Amazon Redshift, the applicable database technologies, and SQL.

The custom scripts that include creating the tables to be migrated and SQL queries for confirming the migration, as listed following:

Oracle_Redshift_For_DMSDemo.template -- an AWS CloudFormation template
Oraclesalesstarschema.sql -- SQL statements to build the SH schema

These scripts are available at the following link: [dms-sbs-RDSOracle2Redshift.zip](#)
Each step in the walkthrough also contains a link to download the file involved or includes the exact query in the step.

An AWS account with AWS Identity and Access Management (IAM) credentials that allow you to launch RDS, AWS Database Migration Service (AWS DMS) instances, and Amazon Redshift clusters in your AWS Region. For information about IAM credentials, see [Creating an IAM User](#).

Basic knowledge of the Amazon Virtual Private Cloud (Amazon VPC) service and of security groups. For information about using Amazon VPC with Amazon RDS, see [Virtual Private Clouds \(VPCs\) and Amazon RDS](#). For information about Amazon RDS security groups, see [Amazon RDS Security Groups](#). For information about using Amazon Redshift in a VPC, see [Managing Clusters in an Amazon Virtual Private Cloud \(VPC\)](#).

An understanding of the supported features and limitations of AWS DMS. For information about AWS DMS, see [What Is AWS Database Migration Service?](#)

Knowledge of the supported data type conversion options for Oracle and Amazon Redshift. For information about data types for Oracle as a source, see [Using an Oracle Database as a Source for AWS Database Migration Service](#). For information about data types for Amazon Redshift as a target, see [Using an Amazon Redshift Database as a Target for AWS Database Migration Service](#).

Migration Architecture

This walkthrough uses AWS CloudFormation to create a simple network topology for

database migration that includes the source database, the replication instance, and the target database in the same VPC. For more information on AWS CloudFormation, see [the CloudFormation documentation](#).

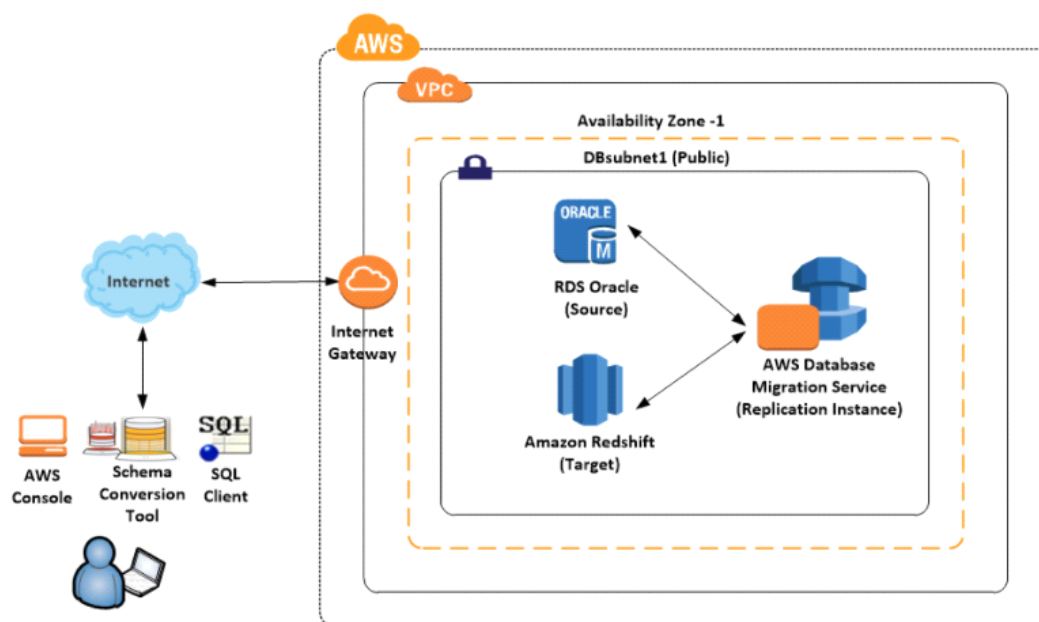
We provision the AWS resources that are required for this AWS DMS walkthrough through AWS CloudFormation. These resources include a VPC and Amazon RDS instance for Oracle and an Amazon Redshift cluster. We provision through CloudFormation because it simplifies the process, so we can concentrate on tasks related to data migration. When you create a stack from the CloudFormation template, it provisions the following resources:

- A VPC with CIDR (10.0.0.0/24) with two public subnets in your region, DBSubnet1 at the address 10.0.0.0/26 in Availability Zone (AZ) 1 and DBSubnet2 at the address 10.0.0.64/26, in AZ 12.
- A DB subnet group that includes DBSubnet1 and DBSubnet2.
- Oracle RDS Standard Edition Two with these deployment options:
 - License Included
 - Single-AZ setup
 - db.m3.medium or equivalent instance class
 - Port 1521
 - Default option and parameter groups
- Amazon Redshift cluster with these deployment options:
 - dc1.large
 - Port 5439
 - Default parameter group
- A security group with ingress access from your computer or 0.0.0.0/0 (access from anywhere) based on the input parameter

We have designed the CloudFormation template to require few inputs from the user. It provisions the necessary AWS resources with minimum recommended configurations. However, if you want to change some of the configurations and parameters, such as the VPC CIDR block and Amazon RDS instance types, feel free to update the template.

We use the AWS Management Console to provision the AWS DMS resources, such as the replication instance, endpoints, and tasks. You install client tools such as SQL Workbench/J and the AWS Schema Conversion Tool (AWS SCT) on your local computer to connect to the Amazon RDS instances.

Following is an illustration of the migration architecture for this walkthrough.



Step by Step Migration

In the following sections, you can find step-by-step instructions for migrating an Amazon RDS for Oracle database to Amazon Redshift. These steps assume that you have already prepared your source database as described in preceding sections.

Step 1: Launch the RDS Instances in a VPC by Using the CloudFormation Template

Before you begin, you'll need to download an AWS CloudFormation template. Follow these instructions:

1. Download the following archive to your computer:
<http://docs.aws.amazon.com/dms/latest/sbs/samples/dms-sbs-RDSOracle2Redshift.zip>
2. Extract the CloudFormation template (**Oracle_Redshift_For_DMSDemo_template**) from the archive.
3. Copy and paste the **Oracle_Redshift_For_DMSDemo_template** file into your current directory.

Now you need to provision the necessary AWS resources for this walkthrough.

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
2. Choose **Create Stack**.
3. On the **Select Template** page, choose **Upload a template to Amazon S3**.
4. Click **Choose File**, and then choose the **Oracle_Redshift_For_DMSDemo_template** file that you extracted from the [dbm-sbs-RDSOracle2Redshift.zip](#) archive.
5. Choose **Next**. On the **Specify Details** page, provide parameter values as shown following.

For This Parameter	Do This
*Stack Name *	Type Oracle to Redshift DW using DMS.
OracleDBName	Provide a unique name for your database. The name should begin with a letter. The default is ORCL.
OracleDBUsername	Specify the admin (DBA) user for managing the Oracle instance. The default is oraadmin.
OracleDBPassword	Provide the password for the admin user. The default is oraadmin123
RedshiftDBName	Provide any unique name for your database. The name should begin with a letter. The default is test.
RedshiftDBUsername	Provide the password for the master user. The default is Redshift#123.
ClientIP	Specify the IP address in CIDR (x.x.x.x/32) format for your local computer. You can get your IP address from whatsmyip.org . Your RDS instances' security group will allow ingress to this IP address. The default is access from anywhere (0.0.0.0/0), which is not recommended; you should use your IP address for this walkthrough.

Create stack

Select Template
Specify Details
Options
Review

Specify Details

Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. [Learn more](#).

Stack name

Parameters

Source Oracle Database Configuration

OracleDBName Enter Oracle Database name

OracleDBUsername Enter database Admin username for Oracle

OracleDBPassword Enter password for Oracle Admin user

Target Redshift Database Configuration

RedshiftDBName Enter Redshift DB name

RedshiftDBUsername Enter master user name for Redshift

RedshiftDBPassword Enter master user password for Redshift

Enter IP address for DB Security group Configuration

ClientIP

The IP address range that can be used to connect to the RDS instances from your local machine. It must be a valid IP CIDR range of the form x.x.x.x/x. Please get your address using [checkip.amazonaws.com](#) or [whatsmyip.org](#)

[Cancel](#) [Previous](#) [Next](#)

- Create stack

8. AWS can take about 20 minutes or more to create the stack with an Amazon RDS Oracle instance and an Amazon Redshift cluster.

Create Stack

Actions

Design template

Filter: Active

By Stack Name

Showing 1 stack

Stack Name	Created Time	Status	Description
<input checked="" type="checkbox"/> OracletoRedshiftDWusingDM	2016-12-22 14:14:02 UTC-0800	CREATE_IN_PROGRESS	This CloudFormation sample template OracleDWtoRedshift_DMS creates a Oracle and R

Overview

Outputs

Resources

Events

Template

Parameters

Tags

Stack Policy

Change Sets

2016-12-22

Status

Type

Logical ID

Status reason

14:14:02 UTC-0800

CREATE_IN_PROGRESS

AWS::CloudFormation::Stack

OracletoRedshiftDWusingDMS

User Initiated

42 more events available to display

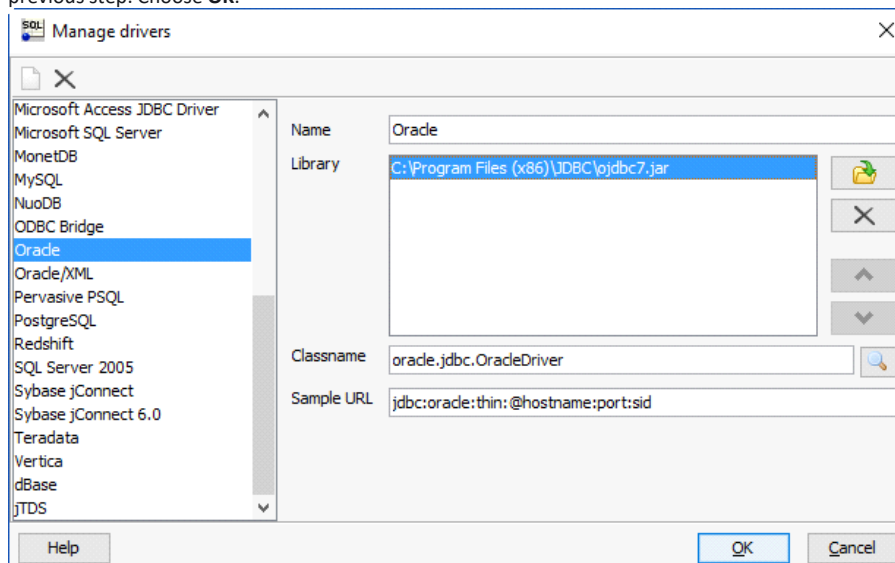
- After the stack is created, select the **OracletoRedshiftDWusingDMS** stack, and then choose the Outputs view. Record the JDBC connection strings, **OracleJDBCConnectionString** and **RedshiftJDBCConnectionString**, for use later in this walkthrough to connect to the Oracle and Amazon Redshift databases.

Step 2: Install the SQL Tools and AWS Schema Conversion Tool on Your Local Computer

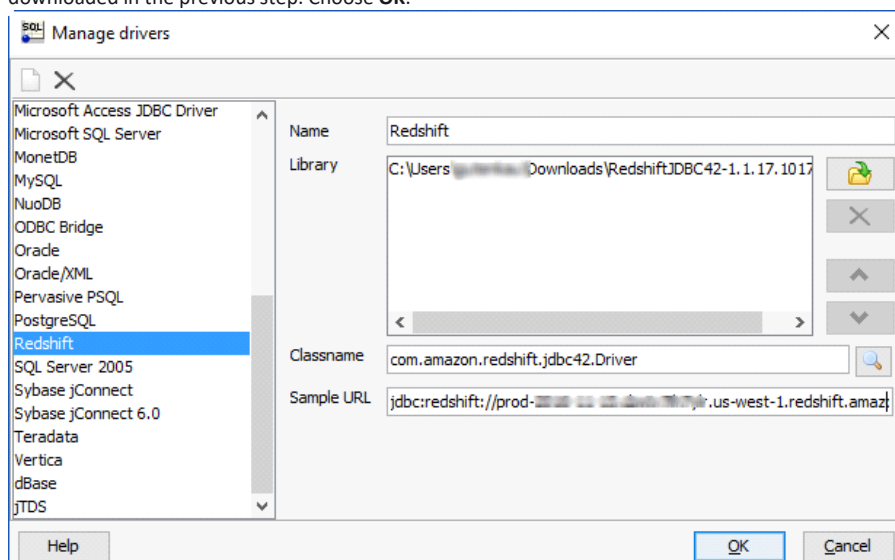
Next, you need to install a SQL client and AWS SCT on your local computer.

This walkthrough assumes you will use the SQL Workbench/J client to connect to the RDS instances for migration validation.

1. Download SQL Workbench/J from [the SQL Workbench/J website](#), and then install it on your local computer. This SQL client is free, open-source, and DBMS-independent.
2. Download the JDBC driver for your Oracle database release. For more information, go to <https://www.oracle.com/jdbc>.
3. Download the Amazon Redshift driver file, **RedshiftJDBC41-1.1.17.1017.jar**, as described following.
 1. Find the Amazon S3 URL to the file in [Previous JDBC Driver Versions](#) of the Amazon Redshift Cluster Management Guide.
 2. Download the driver as described in [Download the Amazon Redshift JDBC Driver](#) of the same guide.
4. Using SQL Workbench/J, configure JDBC drivers for Oracle and Amazon Redshift to set up connectivity, as described following.
 1. In SQL Workbench/J, choose **File**, then choose **Manage Drivers**.
 2. From the list of drivers, choose Oracle.
 3. Choose the **Open icon**, then choose the **ojdbc7.jar** file that you downloaded in the previous step. Choose **OK**.

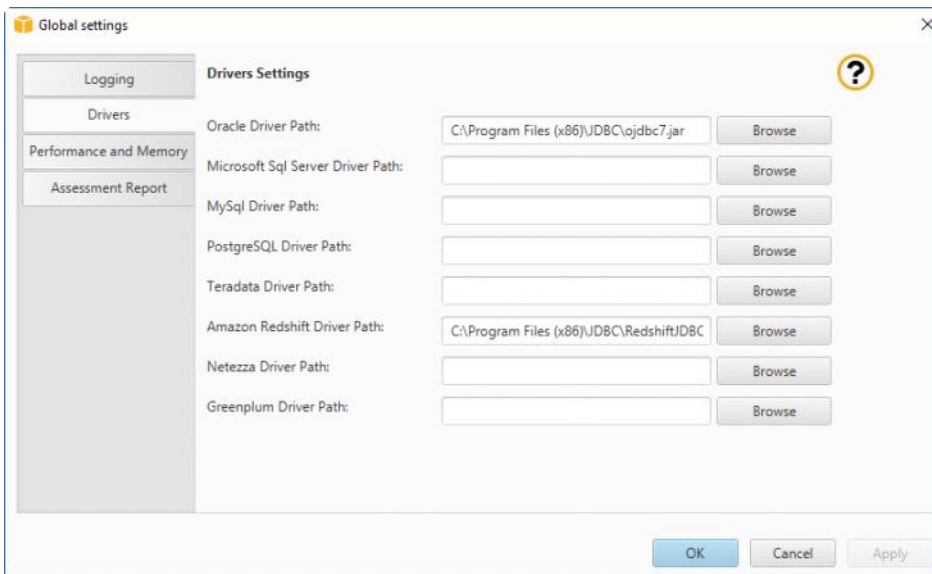


4. From the list of drivers, choose **Redshift**.
5. Choose the **Open icon**, then choose the **Amazon Redshift JDBC** driver that you downloaded in the previous step. Choose **OK**.



Next, install AWS SCT and the required JDBC drivers.

1. Download AWS SCT from [Installing and Updating the AWS Schema Conversion Tool](#) in the AWS Schema Conversion Tool User Guide.
2. Follow the instructions to install AWS SCT. By default, the tool is installed in the C:\Program Files\AWS Schema Conversion Tool\AWS directory.
3. Launch AWS SCT.
4. In AWS SCT, choose **Global Settings** from **Settings**.
5. Choose **Settings, Global Settings**, then choose **Drivers**, and then choose **Browse** for **Oracle Driver Path**. Locate the Oracle JDBC driver and choose **OK**.
6. Choose **Browse** for **Amazon Redshift Driver Path**. Locate the Amazon Redshift JDBC driver and choose **OK**. Choose **OK** to close the dialog box.



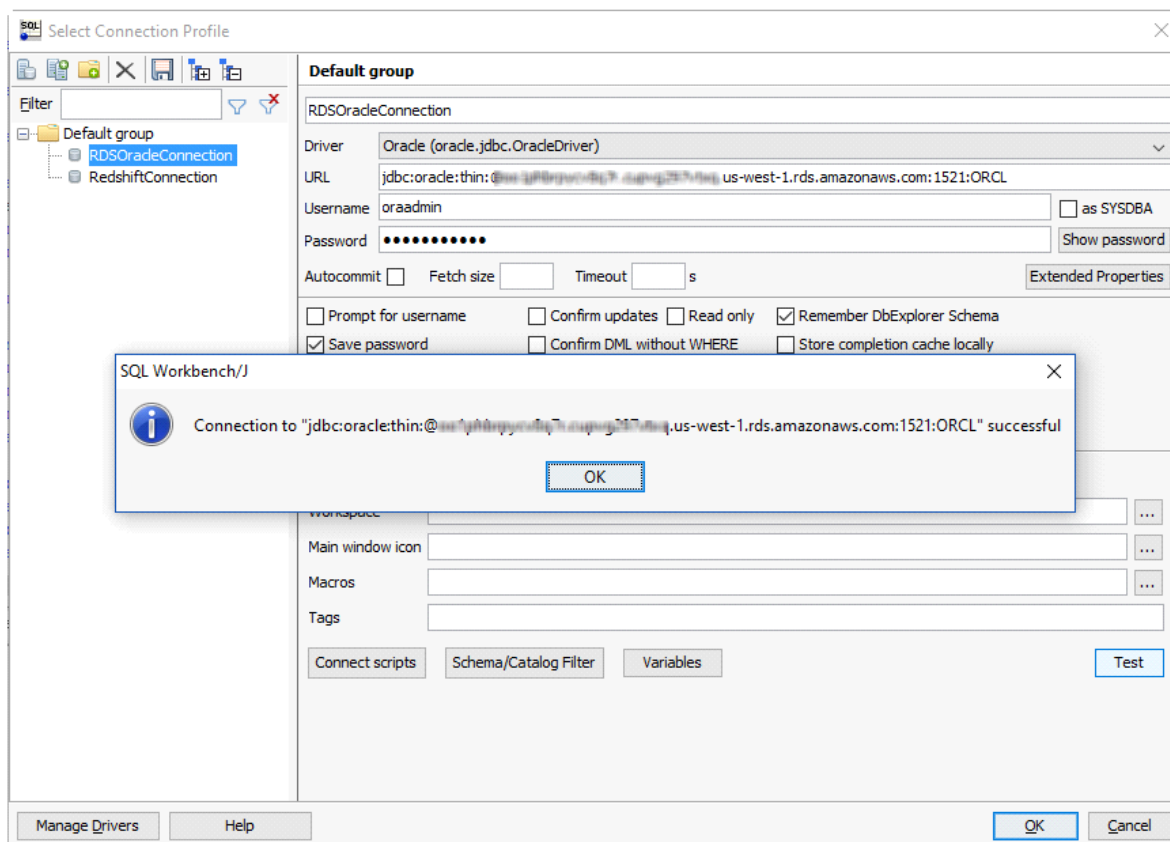
Step 3: Test Connectivity to the Oracle DB Instance and Create the Sample Schema

After the CloudFormation stack has been created, test the connection to the Oracle DB instance by using SQL Workbench/J and then create the HR sample schema.

1. In SQL Workbench/J, choose **File**, then choose **Connect window**. Create a **new connection profile** using the following information.

For This Parameter	Do This
New profile name	Type <code>RDSOracleConnection</code> .
Driver	Choose <code>Oracle (oracle.jdbc.OracleDriver)</code> .
URL	Use the OracleJDBCConnectionString value you recorded when you examined the output details of the DMSdemo stack in a previous step.
Username	Type <code>oraadmin</code> .
Password	Type <code>oraadmin123</code> .

2. Test the connection by choosing **Test**. Choose **OK** to close the dialog box, then choose **OK** to create the connection profile.



Note:

If your connection is unsuccessful, ensure that the IP address you assigned when creating the CloudFormation template is the one you are attempting to connect from. This issue is the most common one when trying to connect to an instance.

3. Create the SH schema you will use for migration using a custom SQL script (Oraclesalesstarschema.sql). To obtain this script, do the following:
 - Download the following archive to your computer: <http://docs.aws.amazon.com/dms/latest/sbs/samples/dms-sbs-RDSOracle2Redshift.zip>
 - Extract the SQL script(Oraclesalesstarschema.sql) from the archive.
 - Copy and paste the Oraclesalesstarschema.sql file into your current directory.
 - a. Open the SQL script in a text editor. Copy the entire script.
 - b. In SQL Workbench/J, paste the SQL script in the Default.wksp window showing **Statement 1**.
 - c. Choose **SQL**, then choose **Execute All**.


```

SQL Workbench/J RDSOracleConnection - Default.wksp
File Edit View Data SQL Macros Workspace Tools Help
User=oraadmin, URL=jdbc:oracle:thin:@oo1phbrpycv8q7r.cupvg297vtbx.us-west-1.

Statement 1
1553 ALTER TABLE "SH"."SALES" MODIFY ("CHANNEL_ID" NOT NULL ENABLE);
1554 ALTER TABLE "SH"."SALES" MODIFY ("TIME_ID" NOT NULL ENABLE);
1555 ALTER TABLE "SH"."SALES" MODIFY ("CUST_ID" NOT NULL ENABLE);
1556 ALTER TABLE "SH"."SALES" MODIFY ("PROD_ID" NOT NULL ENABLE);
1557 -----
1558 -- Ref Constraints for Table SALES
1559 -----
1560
1561 ALTER TABLE "SH"."SALES" ADD CONSTRAINT "SALES_CHANNEL_FK" FOREIGN KEY ("CHANNEL_ID")
1562 REFERENCES "SH"."CHANNELS" ("CHANNEL_ID") ENABLE NOVALIDATE;
1563 ALTER TABLE "SH"."SALES" ADD CONSTRAINT "SALES_CUSTOMER_FK" FOREIGN KEY ("CUST_ID")
1564 REFERENCES "SH"."CUSTOMERS" ("CUST_ID") ENABLE NOVALIDATE;
1565 ALTER TABLE "SH"."SALES" ADD CONSTRAINT "SALES_PRODUCT_FK" FOREIGN KEY ("PROD_ID")
1566 REFERENCES "SH"."PRODUCTS" ("PROD_ID") ENABLE NOVALIDATE;
1567 ALTER TABLE "SH"."SALES" ADD CONSTRAINT "SALES_PROMO_FK" FOREIGN KEY ("PROMO_ID")
1568 REFERENCES "SH"."PROMOTIONS" ("PROMO_ID") ENABLE NOVALIDATE;
1569 -----
1570
1571 -- Collect Statistics of all Tables in SH
1572 -----
1573
Messages
Execution time: 0.05s
Statement 1293 of 1294 finished

0 rows affected
Table SH.SALES analyzed

Execution time: 0.22s
Statement 1294 of 1294 finished

Ready, if you are L:1567 C:1 1m 40s Timeout: 0 Max. Rows: 0

```

- verify the object types and count in SH Schema were created successfully by running the following SQL query.

```
Select OBJECT_TYPE, COUNT(*) from dba_OBJECTS where owner='SH'
GROUP BY OBJECT_TYPE;
```

The results of this query should be similar to the following.

OBJECT_TYPE	COUNT(*)
INDEX PARTITION	40
TABLE PARTITION	8
TABLE	5
INDEX	15

- Verify the total number of tables and number of rows for each table by running the following SQL query.

```
Select table_name, num_rows from dba_tables where owner='SH' order by 1;
```

The results of this query should be similar to the following.

TABLE_NAME	NUM_ROWS
CHANNELS	5
CUSTOMERS	8
PRODUCTS	66
PROMOTIONS	503
SALES	553

- Verify the integrity in tables. Check the number of sales made in different channels by

running the following SQL query.

```
Select b.channel_desc,count(*) from SH.SALES a,SH.CHANNELS b where a.channel_id=b.channel_id
group by b.channel_desc
order by 1;
```

The results of this query should be similar to the following.

CHANNEL_DESC	COUNT(*)
Direct Sales	710
Internet	52
Partners	344

Note:
The preceding examples are representative of validation queries. When you perform actual migrations, you should develop similar queries to validate the schema and the data integrity.

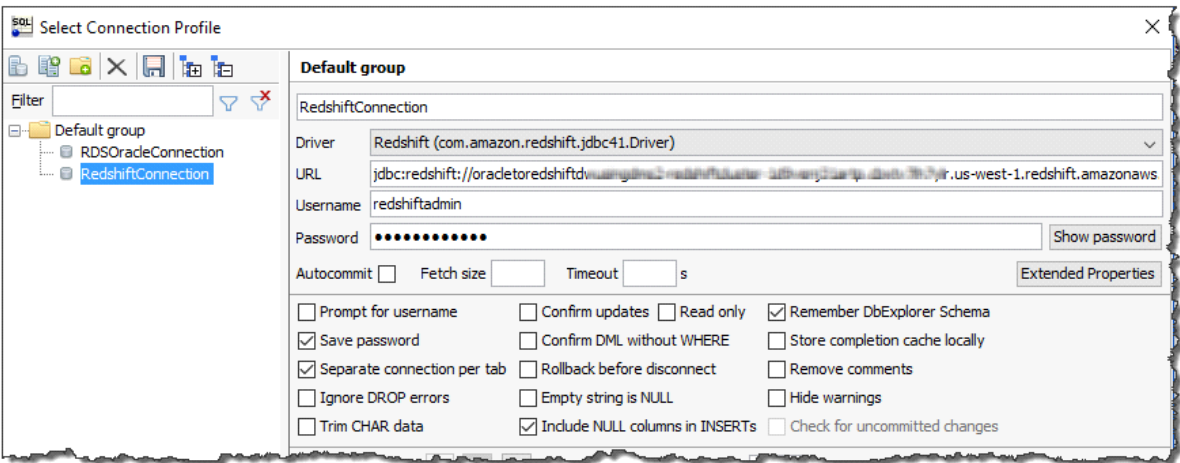
Step 4: Test the Connectivity to the Amazon Redshift Database

Next, test your connection to your Amazon Redshift database.

- 1. In SQL Workbench/J, choose **File**, then choose **Connect window**. Choose the **Create a new connection profile** icon. Connect to the Amazon Redshift database in SQL Workbench/J by using the information shown following.

For This Parameter	Do This
New profile name	Type RedshiftConnection.
Driver	Choose Redshift (com.amazon.redshift.jdbc42.Driver).
URL	Use the RedshiftJDBCCConnectionString value you recorded when you examined the output details of the DMSdemo stack in a previous step.
Username	Type redshiftadmin.
Password	Type Redshift#123.

- 2. Test the connection by choosing **Test**. Choose **OK** to close the dialog box, then choose **OK** to create the connection profile.



Note
If your connection is unsuccessful, ensure that the IP address you assigned when creating the CloudFormation template is the one you are attempting to connect from. This issue is the most common one when trying to connect to an instance.

- 3. Verify your connectivity to the Amazon Redshift DB instance by running a sample SQL

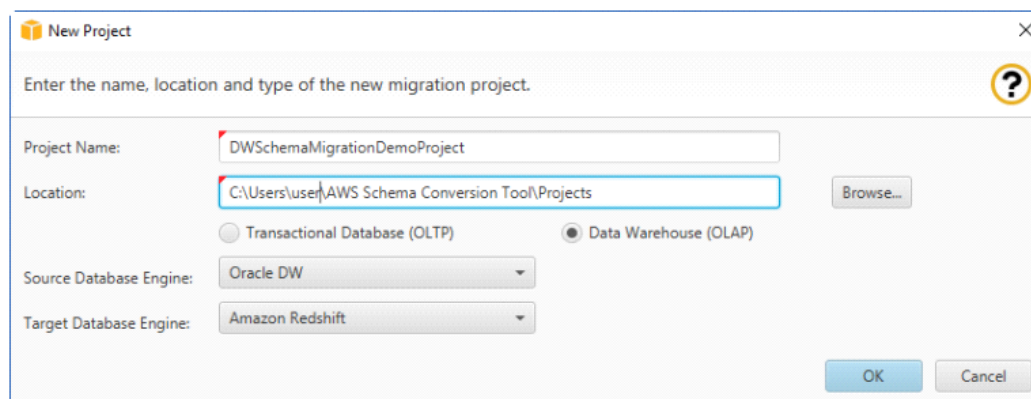
command, such as `select current_date;`.

Step 5: Use AWS SCT to Convert the Oracle Schema to Amazon Redshift

Before you migrate data to Amazon Redshift, you convert the Oracle schema to an Amazon Redshift schema as described following.

1. Launch AWS SCT. In AWS SCT, choose **File**, then choose **New Project**. Create a new project called `DWSchemaMigrationDemoProject`. Enter the following information in the New Project window, and then choose **OK**.

For This Parameter	Do This
Project Name	Type <code>DWSchemaMigrationDemoProject</code> .
Location	Use the default Projects folder and the default Data Warehouse (OLAP) option.
Source Database Engine	Choose Oracle DW .
Target Database Engine	Choose Amazon Redshift .



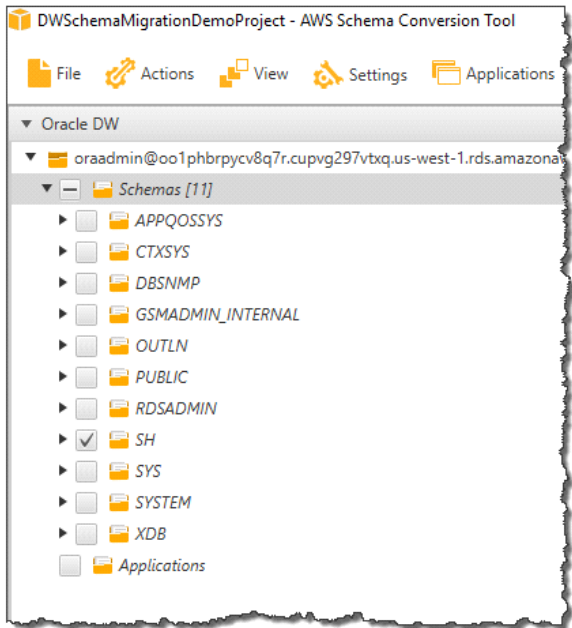
2. Choose **Connect to Oracle**. In the **Connect to Oracle** dialog box, enter the following information, and then choose **Test Connection**.

For This Parameter	Do This
Type	Choose SID .
Server name	Use the OracleJDBCConnectionString value you used to connect to the Oracle DB instance, but remove the JDBC prefix information and the port and database name suffix. For example, a sample connection string you use with SQL Workbench/J might be <code>"jdbc:oracle:thin:@abc12345678.cqi87654abc.us-west-2.rds.amazonaws.com:1521:ORCL"</code> . For the AWS SCT Server name , you remove <code>"jdbc:oracle:thin:@"</code> and <code>":1521:ORCL"</code> and use just the server name: <code>"abc12345678.cqi87654abc.us-west-2.rds.amazonaws.com"</code> .
Server port	Type 1521.
Oracle SID	Type ORCL.
User name	Type oraadmin.
Password	Type oraadmin123.

3. Choose **OK** to close the alert box, then choose **OK** to close the dialog box and to start the connection to the Oracle DB instance. The database structure on the Oracle DB instance is shown following. Select only the SH schema.

Note

If the SH schema does not appear in the list, choose **Actions**, then choose **Refresh from Database**.

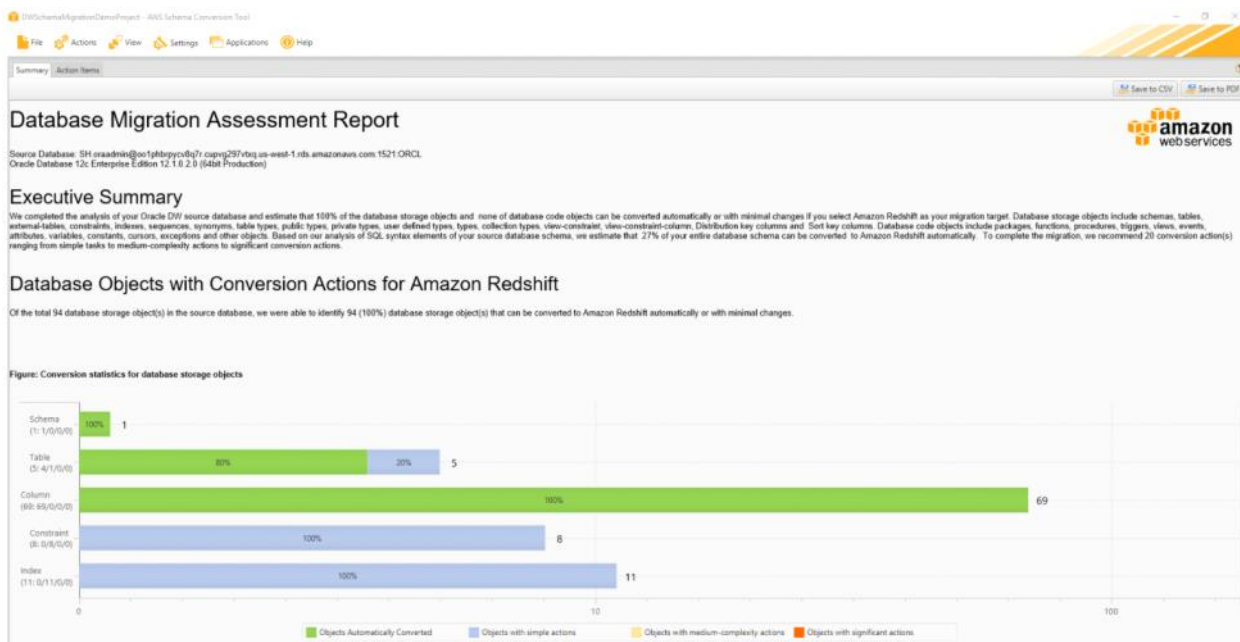


4. Choose **Connect to Amazon Redshift**. In the **Connect to Amazon Redshift** dialog box, enter the following information and then choose **Test Connection**.

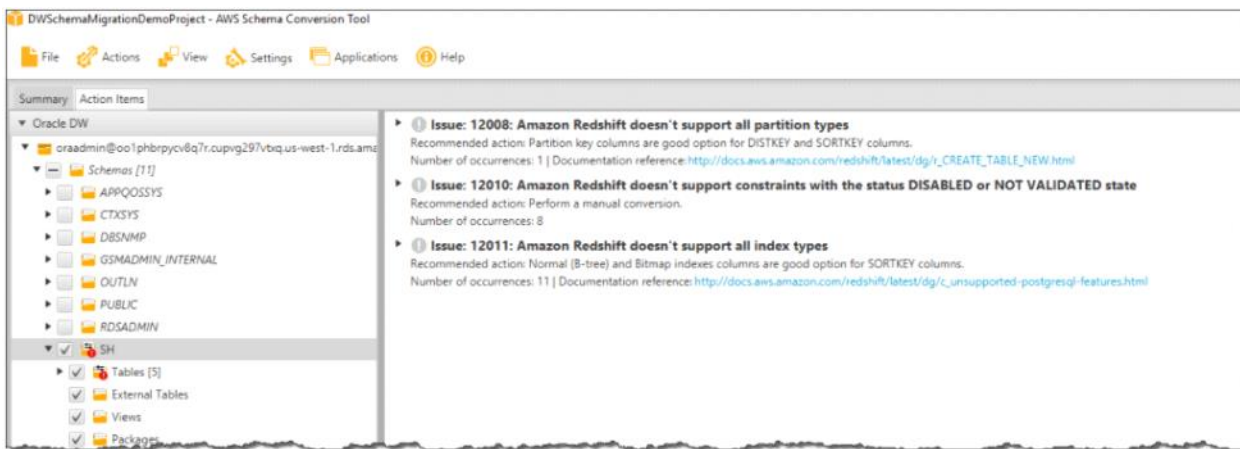
For This Parameter	Do This
Type	Choose SID .
Server name	Use the RedshiftJDBCConnectionString value you used to connect to the Amazon Redshift cluster, but remove the JDBC prefix information and the port suffix. For example, a sample connection string you use with SQL Workbench/J might be "jdbc:redshift://oracletoredshiftdwusingdms-redshiftcluster-abc123567.abc87654321.us-west-2.redshift.amazonaws.com:5439/test". For the AWS SCT Server name , you remove "jdbc:redshift://" and :5439/test" to use just the server name: "oracletoredshiftdwusingdms-redshiftcluster-abc123567.abc87654321.us-west-2.redshift.amazonaws.com"
Server port	Type 5439.
User name	Type redshiftadmin.
Password	Type Redshift#123.

AWS SCT analyzes the SH schema and creates a database migration assessment report for the conversion to Amazon Redshift.

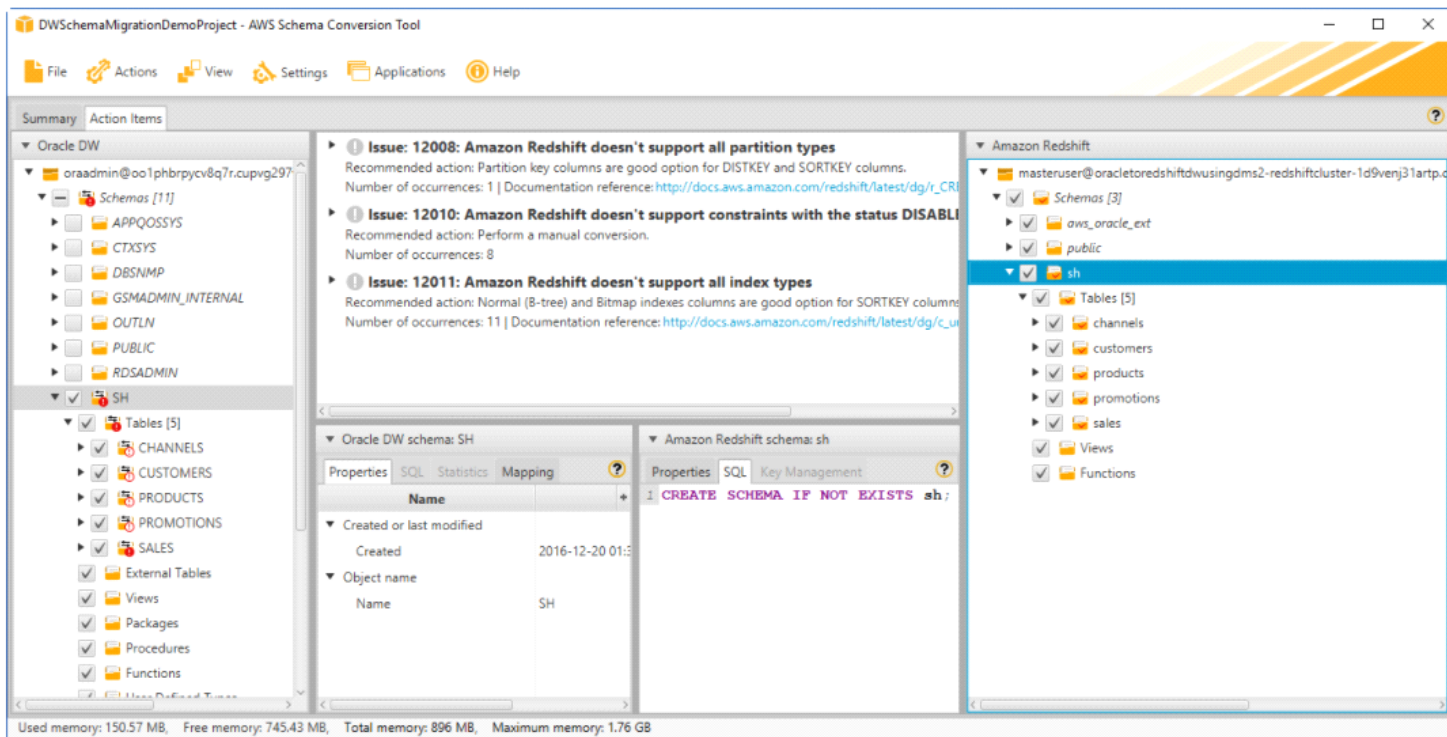
5. Choose **OK** to close the alert box, then choose **OK** to close the dialog box to start the connection to the Amazon Redshift DB instance.
6. In the **Oracle DW** view, open the context (right-click) menu for the **SH** schema and select **Create Report**.
7. Review the report summary. To save the report, choose either **Save to CSV** or **Save to PDF**. The report discusses the type of objects that can be converted by using AWS SCT, along with potential migration issues and actions to resolve these issues. For this walkthrough, you should see something like the following.



- Choose the **Action Items** tab. The report discusses the type of objects that can be converted by using AWS SCT, along with potential migration issues and actions to resolve these issues. For this walkthrough, you should see something like the following.



- Open the context (right-click) menu for the **SH** item in the **Schemas** list, and then choose **Collect Statistics**. AWS SCT analyzes the source data to recommend the best keys for the target Amazon Redshift database. For more information, see [Collecting or Uploading Statistics for the AWS Schema Conversion Tool](#).
- Open the context (right-click) menu for the **SH** schema, and then choose **Convert schema**.
- Choose **Yes** for the confirmation message. AWS SCT then converts your schema to the target database format.



Note

The choice of the Amazon Redshift sort keys and distribution keys is critical for optimal performance. You can use key management in AWS SCT to customize the choice of keys. For this walkthrough, we use the defaults recommended by AWS SCT. For more information, see [Optimizing Amazon Redshift by Using the AWS Schema Conversion Tool](#).

12. In the Amazon Redshift view, open the context (right-click) menu for the **SH** schema, and then choose **Apply to database** to apply the schema scripts to the target Amazon Redshift instance.
13. Open the context (right-click) menu for the **SH** schema, and then choose **Refresh from Database** to refresh from the target database.

The database schema has now been converted and imported from source to target.

Step 6: Validate the Schema Conversion

To validate the schema conversion, you compare the objects found in the Oracle and Amazon Redshift databases using SQL Workbench/J.

1. In SQL Workbench/J, choose **File**, then choose **Connect window**. Choose the **RedshiftConnection** you created in an earlier step. Choose **OK**.
2. Run the following script to verify the number of object types and count in SH schema in the target Amazon Redshift database. These values should match the number of objects in the source Oracle database.

```
SELECT 'TABLE' AS OBJECT_TYPE,
       TABLE_NAME AS OBJECT_NAME,
       TABLE_SCHEMA AS OBJECT_SCHEMA
FROM information_schema.TABLES
WHERE TABLE_TYPE = 'BASE TABLE'
AND OBJECT_SCHEMA = 'sh';
```

The output from this query should be similar to the following.

object_type	object_name	object_schema
TABLE	channels	sh
TABLE	customers	sh
TABLE	products	sh
TABLE	promotions	sh
TABLE	sales	sh

3. Verify the sort and distributions keys that are created in the Amazon Redshift cluster by using the following query.

```

set search_path to '$user', 'public', 'sh';

SELECT tablename,
       "column",
       TYPE,
       encoding,
       distkey,
       sortkey,
       "notnull"
FROM pg_table_def
WHERE (distkey = TRUE OR sortkey <> 0);

```

The results of the query reflect the distribution key (distkey) and sort key (sortkey) choices made by using AWS SCT key management.

tablename	column	type	encoding	distkey	sortkey	notnull
channels	channel_id	numeric(38,18)	none	true	1	true
customers	cust_id	numeric(38,18)	none	false	4	true
customers	cust_gender	character(2)	none	false	1	true
customers	cust_year_of_birth	smallint	none	false	3	true
customers	cust_marital_status	character varying(40)	none	false	2	false
products	prod_id	integer	none	true	4	true
products	prod_subcategory	character varying(100)	none	false	3	true
products	prod_category	character varying(100)	none	false	2	true
products	prod_status	character varying(40)	none	false	1	true
promotions	promo_id	integer	none	true	1	true
sales	prod_id	numeric(38,18)	none	false	4	true
sales	cust_id	numeric(38,18)	none	false	3	true
sales	time_id	timestamp without time zone	none	true	1	true
sales	channel_id	numeric(38,18)	none	false	2	true
sales	promo_id	numeric(38,18)	none	false	5	true

Step 7: Create an AWS DMS Replication Instance

After we validate the schema structure between source and target databases, as described preceding, we proceed to the core part of this walkthrough, which is the data migration. The following illustration shows a high-level view of the migration process.



A DMS replication instance performs the actual data migration between source and target. The replication instance also caches the transaction logs during the migration. How much CPU and memory capacity a replication instance has influences the overall time required for the migration.

To create an AWS DMS replication instance, do the following:

1. Sign in to the AWS Management Console, open the AWS DMS console at <https://console.aws.amazon.com/dms/>, and choose **Create Migration**. If you are signed in as an AWS Identity and Access Management (IAM) user, you must have the appropriate permissions to access AWS DMS. For more information on the permissions required, see [IAM Permissions Needed to Use AWS DMS](#).
2. Choose **Create migration** to start a database migration.
3. On the **Welcome** page, choose **Next**.
4. On the **Create replication instance** page, specify your replication instance information as shown following.

For This Parameter	Do This
Name	Type DMSdemo-repserver.
Description	Type a brief description, such as DMS demo replication server.
Instance class	Choose dms.t2.medium . This instance class is large enough to migrate a small set of tables.
VPC	Choose OracleToRedshiftusingDMS , which is the VPC that was created by the CloudFormation stack.
Multi-AZ	Choose No .
Publicly accessible	Leave this item selected.

5. For the **Advanced** section, leave the default settings as they are, and choose **Next**.

Step 8: Create AWS DMS Source and Target Endpoints

While your replication instance is being created, you can specify the source and target database endpoints using the AWS Management Console. However, you can only test connectivity after the replication instance has been created, because the replication instance is used in the connection.

1. Specify your connection information for the source Oracle database and the target Amazon Redshift database. The following table describes the source settings.

For This Parameter	Do This
Endpoint Identifier	Type <code>Orasource</code> (the Amazon RDS Oracle endpoint).
Source Engine	Choose oracle .
Server name	Provide the Oracle DB instance name. This name is the Server name value that you used for AWS SCT, such as <code>"abc123567.abc87654321.us-west-2.rds.amazonaws.com"</code> .
Port	Type <code>1521</code> .
SSL mode	Choose None .
Username	Type <code>oraadmin</code> .
Password	Type <code>oraadmin123</code> .
SID	Type <code>ORCL</code> .

The following table describes the target settings.

For This Parameter	Do This
Endpoint Identifier	Type <code>Redshifttarget</code> (the Amazon Redshift endpoint).
Target Engine	Choose redshift .
Servename	Provide the Amazon Redshift DB instance name. This name is the Server name value that you used for AWS SCT, such as <code>"oracletoreshiftdwusingdms-redshiftcluster-abc123567.abc87654321.us-west-2.redshift.amazonaws.com"</code> .
Port	Type <code>5439</code> .
SSL mode	Choose None .
Username	Type <code>redshiftadmin</code> .
Password	Type <code>Redshift#123</code> .
Database name	Type <code>test</code> .

The completed page should look like the following.

✓ Replication instance created successfully.

Your database endpoint can be on-premise, in EC2, RDS or in the cloud. Define the connection details below. It is recommended that you test your endpoint connections here to avoid errors later.

Source database connection details

Endpoint identifier* ⓘ

Source engine* ⓘ

Server name*

Port*

SSL mode* ⓘ

User name*

Password*

SID*

► Advanced

Target database connection details

Endpoint identifier* ⓘ

Target engine* ⓘ

Server name*

Port*

SSL mode* ⓘ

User name*

Password*

Database name*

► Advanced

2. Wait for the status to say **Replication instance created successfully**.
3. To test the source and target connections, choose **Run Test** for the source and target connections.
4. Choose **Next**.

Step 9: Create and Run Your AWS DMS Migration Task

Using an AWS DMS task, you can specify what schema to migrate and the type of migration. You can migrate existing data, migrate existing data and replicate ongoing changes, or replicate data changes only. This walkthrough migrates existing data only.

1. On the **Create Task** page, specify the task options. The following table describes the settings.

For This Parameter	Do This
Task name	Type migrateSHschema.
Replication instance	Shows DMSdemo-repserver (the AWS DMS replication instance created in an earlier step).
Source endpoint	Shows orasource (the Amazon RDS for Oracle endpoint).
Target endpoint	Shows redshifttarget (the Amazon Redshift endpoint).
Migration type	Choose the option Migrate existing data .
Start task on create	Choose this option.

The page should look like the following.

Create task

A task can contain one or more table mappings which define what data is moved from the source to the target. If a table does not exist on the target, it can be created automatically.

Task name* ⓘ

Replication instance*

Source endpoint*

Target endpoint*

Migration type* ⓘ

Start task on create ☒

2. On the **Task Settings** section, specify the settings as shown in the following table.

For This Parameter	Do This
Target table preparation mode	Choose Do nothing .
Include LOB columns in replication	Choose Limited LOB mode .
Max LOB size (kb)	Accept the default (32).

The section should look like the following.

▼ Task Settings

Target table preparation mode*

☐ Do nothing

☒ Drop tables on target

☐ Truncate

Include LOB columns in replication*

☐ Don't include LOB columns

☒ Limited LOB mode

Max LOB size (kb)*

Enable logging

☐

[Advanced Settings](#)

3. In the **Selection rules** section, specify the settings as shown in the following table.

For This Parameter	Do This
Schema name is	Choose Enter a schema.
Schema name is like	Type SH%.
Table name is like	Type %.
Action	Choose Include.

The section should look like the following:

▼ Table mappings

GuidedJSON

Selection rules ⓘ

At least one selection rule with an include action is required. Once you have one or more selection rules, you can add transformation rules.

Where ⓘ

Schema name is

Schema name is like

Table name is like

Use % as a wildcard.

Action

Filter ⓘ

[Add column filter](#)

[Add selection rule](#)

4. Choose **Add selection rule**.

5. Choose **Create task**. The task begins immediately. The **Tasks** section shows you the status of the migration task.

AWS página 17

DMS

Dashboard
Get started
Tasks
Endpoints
Certificates
Replication instances
Subnet groups

Create task Modify Start/Resume Stop Delete

Filter:

ID	Status	Source	Target	Type	Complete %	Elapsed time	Tables loaded
migrateschema2	Creating	orasource	redshifttarget	Full Load	0		0
migrateschema	Ready	orasource	redshifttarget	Full Load	0	0m	0

Step 10: Verify That Your Data Migration Completed Successfully

When the migration task completes, you can compare your task results with the expected results.

1. On the navigation pane, choose **Tasks**.
2. Choose your migration task (*migrateschema*).
3. Choose the **Table statistics** tab, shown following.

Create task Modify Start/Resume Stop Delete

Filter:

ID	Status	Source	Target	Type	Complete %	Elapsed time	Tables loaded
migrateschema	Modifying	orasource	redshifttarget	Full Load	100	1m	5

migrateschema

Overview Task monitoring **Table statistics** Logs

Filter:

Schema	Table	State	Inserts	Deletes	Updates	DDLs	Full Load Rows	Total	Last updated
SH	CHANNELS	Table completed	0	0	0	0	5	5	12/26/16, 9:00 PM
SH	CUSTOMERS	Table completed	0	0	0	0	8	8	12/26/16, 9:00 PM
SH	PRODUCTS	Table completed	0	0	0	0	66	66	12/26/16, 9:00 PM
SH	PROMOTIONS	Table completed	0	0	0	0	503	503	12/26/16, 9:00 PM
SH	SALES	Table completed	0	0	0	0	1,106	1,106	12/26/16, 9:00 PM

4. Connect to the Amazon Redshift instance by using SQL Workbench/J, and then check whether the database tables are successfully migrated from Oracle to Amazon Redshift by running the SQL script shown following.

```
select "table", tbl_rows
from svv_table_info
where
SCHEMA = 'sh'
order by 1;
```

Your results should look similar to the following.

table	tbl_rows
channels	5
customers	8
products	66
promotions	503
sales	1106

5. To verify whether the output for tables and number of rows from the preceding query

matches what is expected for RDS Oracle, compare your results with those in previous steps.

6. Run the following query to check the relationship in tables; this query checks the departments with employees greater than 10.

```
Select b.channel_desc,count(*) from SH.SALES a,SH.CHANNELS b where a.channel_id=b.channel_id
group by b.channel_desc
order by 1;
```

The output from this query should be similar to the following.

```
channel_desc | count
-----+-----
Direct Sales |   355
Internet     |    26
Partners     |   172
```

7. Verify column compression encoding.

DMS uses an Amazon Redshift COPY operation to load data. By default, the COPY command applies automatic compression whenever loading to an empty target table. The sample data for this walkthrough is not large enough for automatic compression to be applied. When you migrate larger data sets, COPY will apply automatic compression.

For more details about automatic compression on Amazon Redshift tables, see [Loading Tables with Automatic Compression](#).

To view compression encodings, run the following query.

```
SELECT *
FROM pg_table_def
WHERE schemaname = 'sh';
```

Now you have successfully completed a database from an Amazon RDS for Oracle DB instance to Amazon Redshift.

Step 11: Delete Walkthrough Resources

After you have completed this walkthrough, perform the following steps to avoid being charged further for AWS resources used in the walkthrough. It's necessary that you do the steps in order, because some resources cannot be deleted if they have a dependency upon another resource.

To delete AWS DMS resources, do the following:

1. On the navigation pane, choose **Tasks**, choose your migration task (*migratehrschema*), and then choose **Delete**.
2. On the navigation pane, choose **Endpoints**, choose the Oracle source endpoint (*orasource*), and then choose **Delete**.
3. Choose the Amazon Redshift target endpoint (*redshifttarget*), and then choose **Delete**.
4. On the navigation pane, choose **Replication instances**, choose the replication instance (*DMSdemo-repserver*), and then choose **Delete**.

Next, you must delete your AWS CloudFormation stack, *DMSdemo*. Do the following:

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation> if you are signed in as an IAM user, you must have the appropriate permissions to access AWS CloudFormation.
2. Choose your CloudFormation stack, *OracletoRedshiftDWusingDMS*.
3. For **Actions**, choose **Delete stack**.

The status of the stack changes to DELETE_IN_PROGRESS while AWS CloudFormation cleans up the resources associated with the *OracletoRedshiftDWusingDMS* stack. When AWS CloudFormation is finished cleaning up resources, it removes the stack from the list.

