**Oracle to postgresSQL Migration:**

https://docs.aws.amazon.com/dms/latest/sbs/CHAP\_RDSOracle2PostgreSQL.Steps.ConvertSchema.html

AWS Database Migration Service is a web service you can use to migrate data from your database that is on-premises, on an Amazon Relational Database Service (Amazon RDS) DB instance, or in a database on an Amazon Elastic Compute Cloud (Amazon EC2) instance to a database on an AWS service. These services can include a database on Amazon RDS or a database on an Amazon EC2 instance. You can also migrate a database from an AWS service to an on-premises database. You can migrate data between heterogeneous or homogenous database engines.

You can use AWS Database Migration Service (AWS DMS) to migrate your data to and from most widely used commercial and open-source databases such as Oracle, PostgreSQL, Microsoft SQL Server, Amazon Redshift, Amazon Aurora, MariaDB, and MySQL. The service supports homogeneous migrations such as Oracle to Oracle, and also heterogeneous migrations between different database platforms, such as Oracle to MySQL or MySQL to Amazon Aurora with MySQL compatibility. The source or target database must be on an AWS service.

AWS Database Migration Service (AWS DMS) helps you migrate databases to AWS efficiently and securely. The source database can remain fully operational during the migration, minimizing downtime to applications that rely on the database. AWS DMS can migrate your Oracle data to the most widely used commercial and open-source databases on AWS.

AWS DMS migrates data, tables, and primary keys to the target database. All other database elements are not migrated.

The AWS Schema Conversion Tool (SCT) makes heterogeneous database migrations easy by automatically converting the source database schema and a majority of the custom code, including views, stored procedures, and functions, to a format compatible with the target database. Any code that cannot be automatically converted is clearly marked so that it can be manually converted. You can use this tool to convert your source Oracle databases to an Amazon Aurora MySQL, MySQL, or PostgreSQL target database on either Amazon RDS or EC2.

It is important to understand that DMS and SCT are two different tools and serve different needs and they don’t interact with each other in the migration process.

* AWS DMS takes a minimalist approach and creates only those objects required to efficiently migrate the data for example tables with primary key – therefore, we will use DMS to load the tables with data without any foreign keys or constraints. (We can also use the SCT to generate the table scripts and create it on the target before performing the load via DMS).
* We will leverage SCT:
  + To identify the issues, limitations and actions for the schema conversion
  + To generate the target schema scripts including foreign key and constraints
  + To convert code such as procedures and views from source to target and apply it on target

The size and type of Oracle database migration you want to do greatly determines the tools you should use. For example, a heterogeneous migration, where you are migrating from an Oracle database to a different database engine on AWS, is best accomplished using AWS DMS. A homogeneous migration, where you are migrating from an Oracle database to an Oracle database on AWS, is best accomplished using native Oracle tools.

[Migrating an On-Premises Oracle Database to Amazon Aurora MySQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_On-PremOracle2Aurora.html)

[Migrating an Amazon RDS Oracle Database to Amazon Aurora MySQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2Aurora.html)

[Migrating a SQL Server Database to Amazon Aurora MySQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_SQLServer2Aurora.html)

[Migrating an Oracle Database to PostgreSQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.html)

[Migrating an Amazon RDS for Oracle Database to Amazon Redshift](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2Redshift.html)

[Migrating MySQL-Compatible Databases to AWS](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_MySQL.html)

[Migrating a MySQL-Compatible Database to Amazon Aurora MySQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_MySQL2Aurora.html)

**AWS DMS doesn't migrate your secondary indexes, sequences, default values, stored procedures, triggers, synonyms, views, and other schema objects not specifically related to data migration. To migrate these objects to your PostgreSQL target, use AWS SCT.**

* [Prerequisites](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Prerequisites.html)
* [Step-by-Step Migration](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.html)
* [Rolling Back the Migration](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_Oracle2PostgreSQL.Rollback.html)
* [Troubleshooting](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_Oracle2PostgreSQL.Troubleshooting.html)
* Understand the supported data type conversion options for Oracle and PostgreSQL
* Understand the supported features and limitations of AWS DMS.
* Size your target PostgreSQL database host. DBAs should be aware of the load profile of the current source Oracle database host. Consider CPU, memory, and IOPS. With RDS, you can size up the target database host, or reduce it, after the migration. If this is the first time you are migrating to
* PostgreSQL, then we recommend that you have extra capacity to account for performance issues and tuning opportunities.
* Audit your source Oracle database. For each schema and all the objects under each schema, determine if any of the objects are no longer being used. Deprecate these objects on the source Oracle database, because there's no need to migrate them if they are not being used.
* If load capacity permits, then get the max size (kb) for each LOB type on the source database, and keep this information for later.
* If possible, move columns with BLOB, CLOB, NCLOB, LONG, LONG RAW, and XMLTYPE to S3, Dynamo DB, or another data store. Doing so simplifies your source Oracle database for an easier migration. It will also lower the capacity requirements for the target PostgreSQL database.
* [Step 1: Install the SQL Drivers and AWS Schema Conversion Tool on Your Local Computer](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.InstallSCT.html)
* [Step 2: Configure Your Oracle Source Database](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_Oracle2PostgreSQL.Steps.ConfigureOracle.html)
* [Step 3: Configure Your PostgreSQL Target Database](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_Oracle2PostgreSQL.Steps.ConfigurePostgreSQL.html)
* [Step 4: Use the AWS Schema Conversion Tool (AWS SCT) to Convert the Oracle Schema to PostgreSQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.ConvertSchema.html)
* [Step 5: Create an AWS DMS Replication Instance](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.CreateReplicationInstance.html)
* [Step 6: Create AWS DMS Source and Target Endpoints](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.CreateSourceTargetEndpoints.html)
* [Step 7: Create and Run Your AWS DMS Migration Task](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.CreateMigrationTask.html)
* [Step 8: Cut Over to PostgreSQL](https://docs.aws.amazon.com/dms/latest/sbs/CHAP_RDSOracle2PostgreSQL.Steps.CutOver.html)