Using AWS Data Migration Service with RDS

INTRODUCTION

AWS's Database Migration Service (DMS) is a managed service to help migrate existing data and replicate changes from on-premise databases to AWS's Relational Database Service (RDS). DMS supports homogenous database engine migrations for many platforms and also heterogeneous migrations to convert database engines or versions using the AWS Schema Conversion Tool. The Database Migration Service can be used to support: simple migrations to AWS RDS, continuous data replication for cloud based failover, migrating to open source solutions, database consolidation, and data warehouse size processing. This guide will cover the requirements, configurations, migration preparation, and deployment.

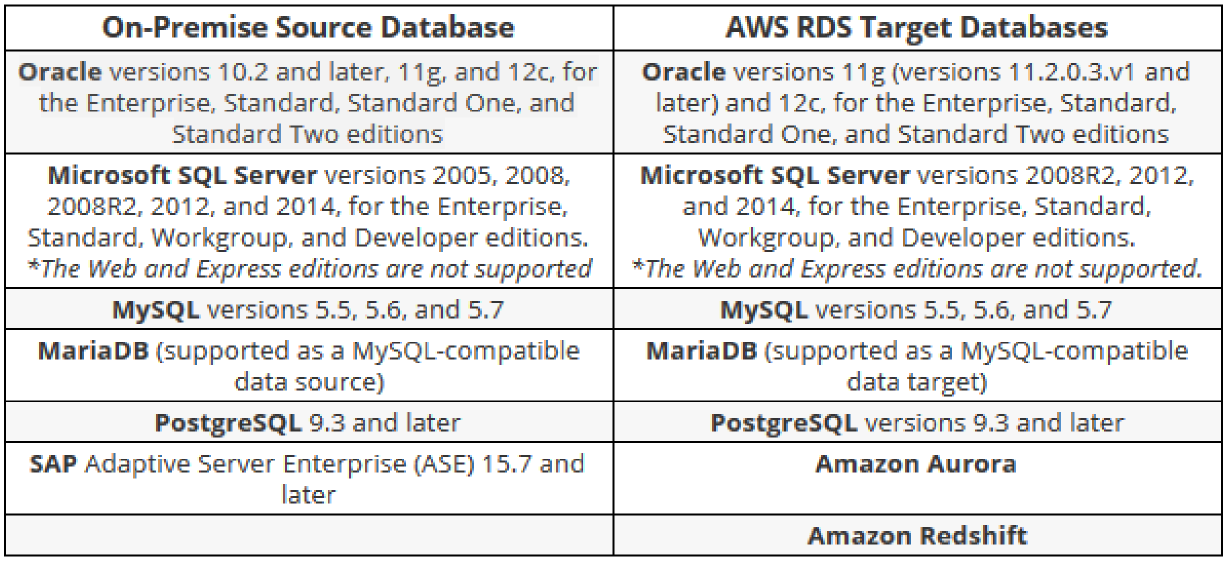
REQUIREMENTS

DATABASE PLATFORMS & VERSIONS

DMS supports multiple source and target database engines. Migrating between on-premise to RDS can be homogenous (ie. Oracle to Oracle) or heterogenous (ie. Oracle to PostgreSQL) to support the needs of your organization. The following table has a list of options for support.

**Note on-premise to on-premise, EC2 to RDS, and RDS to RDS migrations available. See Source Migrations and Target Migrations.*

List of Supported Database Engines

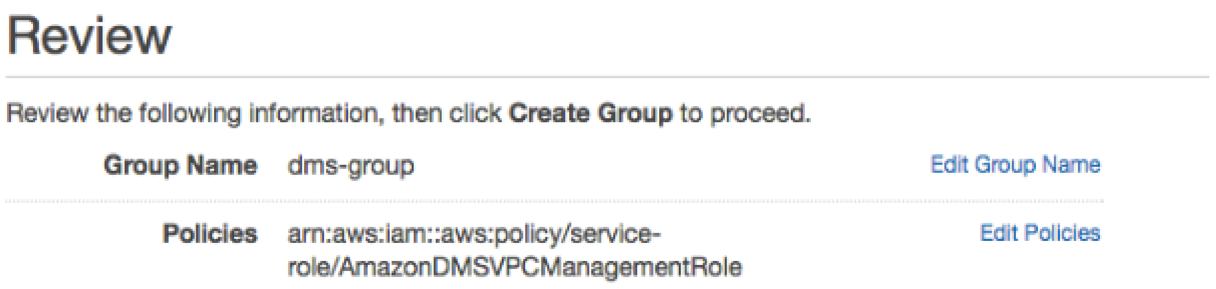


On-Premise Source Database	AWS RDS Target Databases
Oracle versions 10.2 and later, 11g, and 12c, for the Enterprise, Standard, Standard One, and Standard Two editions	**Oracle** versions 11g (versions 11.2.0.3.v1 and later) and 12c, for the Enterprise, Standard, Standard One, and Standard Two editions
Microsoft SQL Server versions 2005, 2008, 2008R2, 2012, and 2014, for the Enterprise, Standard, Workgroup, and Developer editions. **The Web and Express editions are not supported*	**Microsoft SQL Server** versions 2008R2, 2012, and 2014, for the Enterprise, Standard, Workgroup, and Developer editions. **The Web and Express editions are not supported.*
MySQL versions 5.5, 5.6, and 5.7	**MySQL** versions 5.5, 5.6, and 5.7
MariaDB (supported as a MySQL-compatible data source)	**MariaDB** (supported as a MySQL-compatible data target)
PostgreSQL 9.3 and later	**PostgreSQL** versions 9.3 and later
SAP Adaptive Server Enterprise (ASE) 15.7 and later	**Amazon Aurora**
	Amazon Redshift

CONFIGURING NETWORK CONNECTIONS

For DMS replication, you will need to confirm or setup the proper IAM permissions, Security Group rules, on-premise database access, and VPC access for your network. The default DMS deployment should create the correct IAM permissions for your VPC cloud replication. Your on-premise database will need to accessible remotely by DMS and you may need to create IAM roles, Security Groups, and endpoint access depending on your cloud infrastructure.

The IAM permissions will need to be set to allow DMS to deploy replication instances and endpoints. Depending on your cloud infrastructure, you will need to create an IAM group to allow DMS access. Go to IAM > Groups and click Create New Group name the group dms-group and click Next Step. Select the AmazonDMSVPCManagementRole from the AWS predefined policies and click Next Step. Before you click Create Group make sure it looks similar to this example review.

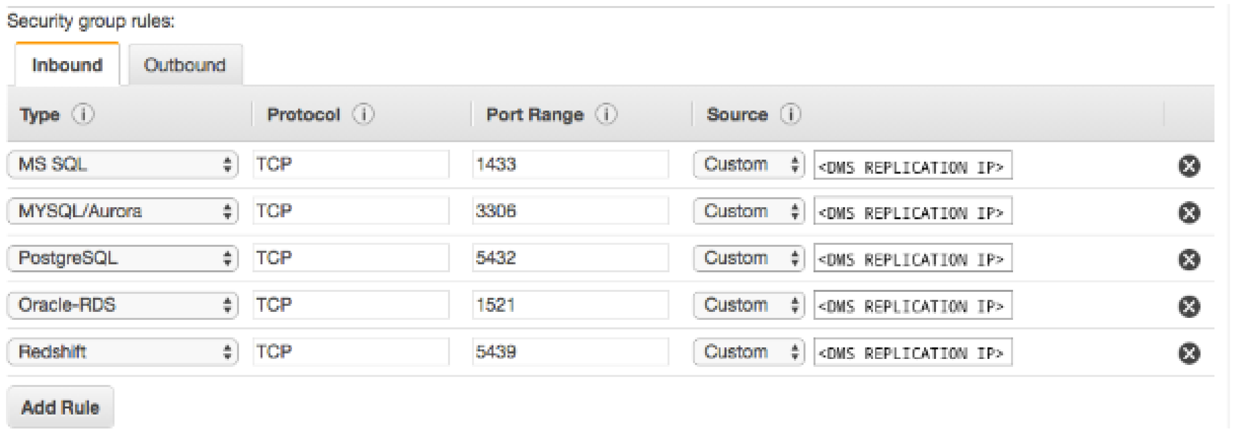


Review

Review the following information, then click **Create Group** to proceed.

Group Name	dms-group	Edit Group Name
Policies	arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole	Edit Policies

Depending on your target database, you will need to setup Security Group rules to allow access to the proper inbound database ports from DMS replication server IP Address or Security Group. From the AWS Console, go to EC2 > Security Groups and click Create Security Group. The following is an example configuration with a variety of default database ports.



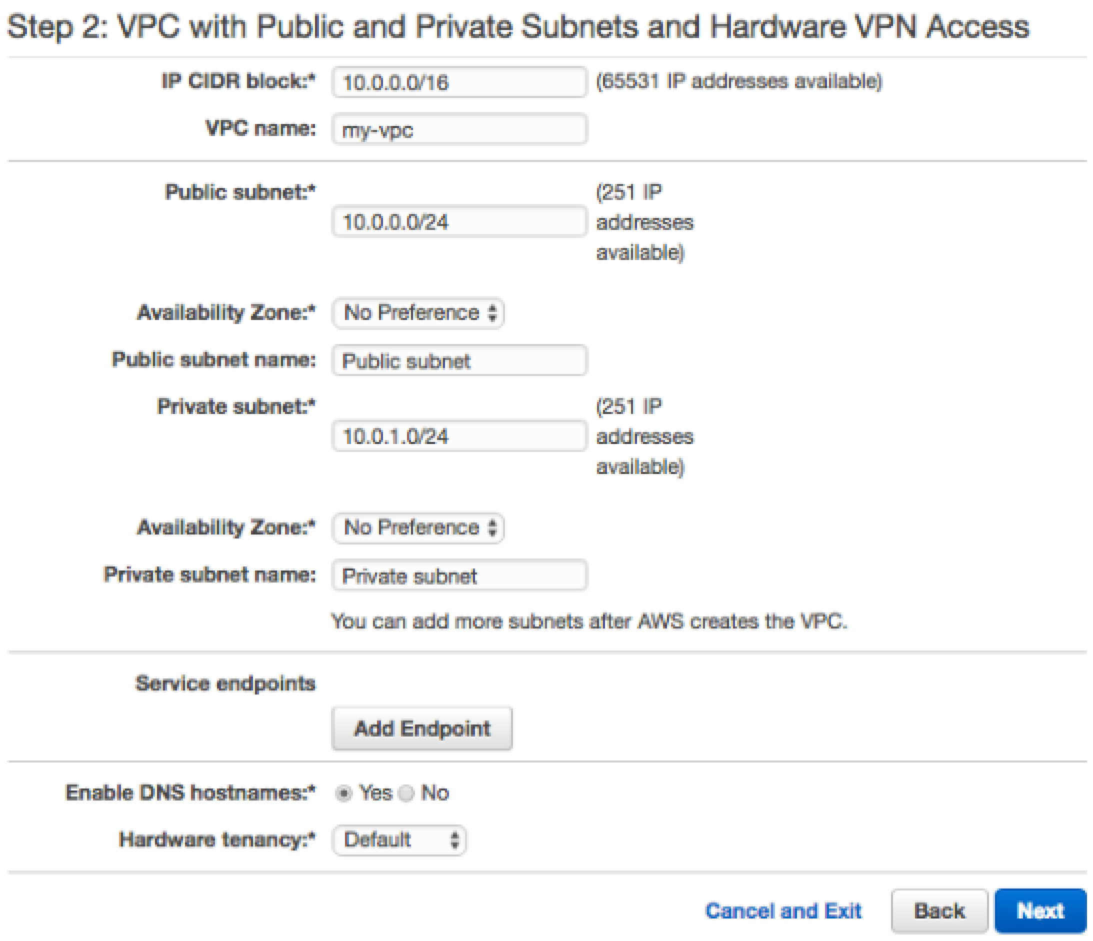
Security group rules:

Inbound | Outbound

Type	Protocol	Port Range	Source	
MS SQL	TCP	1433	Custom	<DMS REPLICATION IP>
MYSQL/Aurora	TCP	3306	Custom	<DMS REPLICATION IP>
PostgreSQL	TCP	5432	Custom	<DMS REPLICATION IP>
Oracle-RDS	TCP	1521	Custom	<DMS REPLICATION IP>
Redshift	TCP	5439	Custom	<DMS REPLICATION IP>

Add Rule

The Virtual Private Cloud (VPC) which will contain your DMS and RDS instances must have a public endpoint or virtual private network (VPN) connection to access your on-premise database instance. To setup a new VPC from the AWS Console go to VPC and click Start VPC Wizard. Select VPC with Public and Private Subnets and Hardware VPN Access and click Select. Create the Public and Private Subnets According to your desired configuration.


Step 2: VPC with Public and Private Subnets and Hardware VPN Access

IP CIDR block:*	10.0.0.0/16	(65531 IP addresses available)
VPC name:	my-vpc	
Public subnet:*	10.0.0.0/24	(251 IP addresses available)
Availability Zone:*	No Preference	
Public subnet name:	Public subnet	
Private subnet:*	10.0.1.0/24	(251 IP addresses available)
Availability Zone:*	No Preference	
Private subnet name:	Private subnet	

You can add more subnets after AWS creates the VPC.

Service endpoints

Enable DNS hostnames:* [✓ Yes] No

Hardware tenancy:* Default

After configuring your subnets, click Next. Now configure your VPN connection with the correct IP address, names and routing (routing can be a dynamic or a static IP address). Click Create VPC to complete the creation.

**For more in depth documentation on configuring your migration network, refer the AWS documentation.*

PREPARING THE MIGRATION

Migrations using DMS can be homogeneous or heterogeneous. Homogeneous migrations require no schema conversion and your database tools can apply the source schema to the target. For heterogenous migrations, Amazon provides AWS Schema Conversion Tool to summarize, generate, and apply schema conversions from the source to the target. The schema conversion tool is stand alone application that must be downloaded and installed on a machine that can connect to the source and target databases. AWS provides a variety installations for Windows, Mac, Ubuntu, and Fedora with links to the supporting JDBC database drivers.

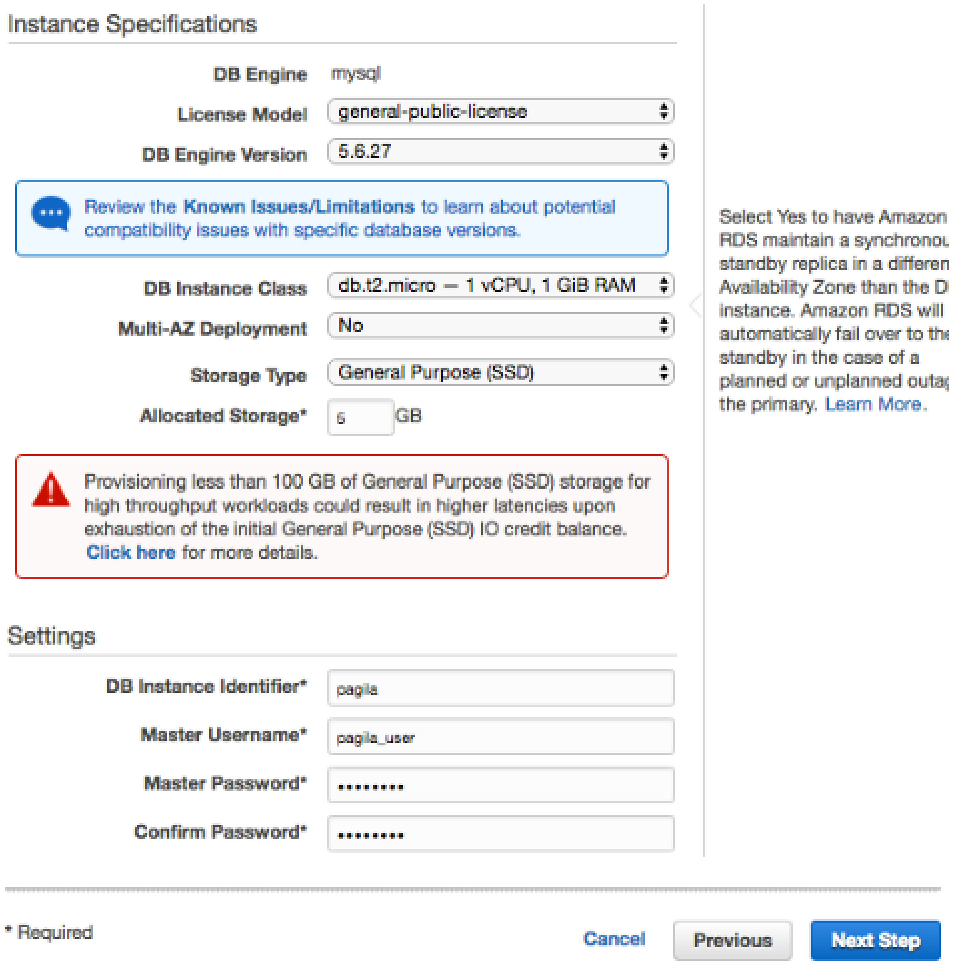
Migration planning is crucial to successfully migrate to RDS and many considerations should be considered. Some requirements you should consider are: why are you migrating to a new version or database engine, will your source database continue to be in use, will the source to target replication be a one time data export or continuing change replication, will the source need to be highly available, does all of the data need to be migrated, and what are the database, compute, and network constraints? A strong understanding of your source database, schema, and tables will help to make the migration a success.

DMS supports basic schema migration with the creation of tables, primary keys, and some unique indexes but does not automatically create secondary indexes, foreign keys, user accounts, and etc for the target database. Tools like Oracle SQL Developer, MySQL Workbench, or pgAdmin III can be used to convert or export schema for homogeneous migrations. The AWS Schema Conversion tool can generate schemas for tables, indexes, and views to the target database engine for heterogeneous migrations. For more information, reference the AWS Schema Conversion Tool user guide.

**The following example is migrating pagila sample database from PostgreSQL source to MySQL target. You can recreate this example migration by loading the pagila sample database into a source PostgreSQL database.*

CREATE A TARGET RDS DATABASE

For the following example we will need to create a MySQL RDS instance. In the AWS Console, go to RDS dashboard and select Instances > Launch DB Instance > MySQL and click Select. Choose Dev/Test MySQL and click Next Step. Use the default settings and enter the DB Instance Identifier as “pagila”, Master Username as “pagila_user”, enter and confirm a password, and click Next Step.



Instance Specifications

DB Engine	mysql
License Model	general-public-license
DB Engine Version	5.6.27

Review the **Known Issues/Limitations** to learn about potential compatibility issues with specific database versions.

DB Instance Class	db.t2.micro — 1 vCPU, 1 GiB RAM
Multi-AZ Deployment	No
Storage Type	General Purpose (SSD)
Allocated Storage*	5 GB

Provisioning less than 100 GB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. **Click here** for more details.

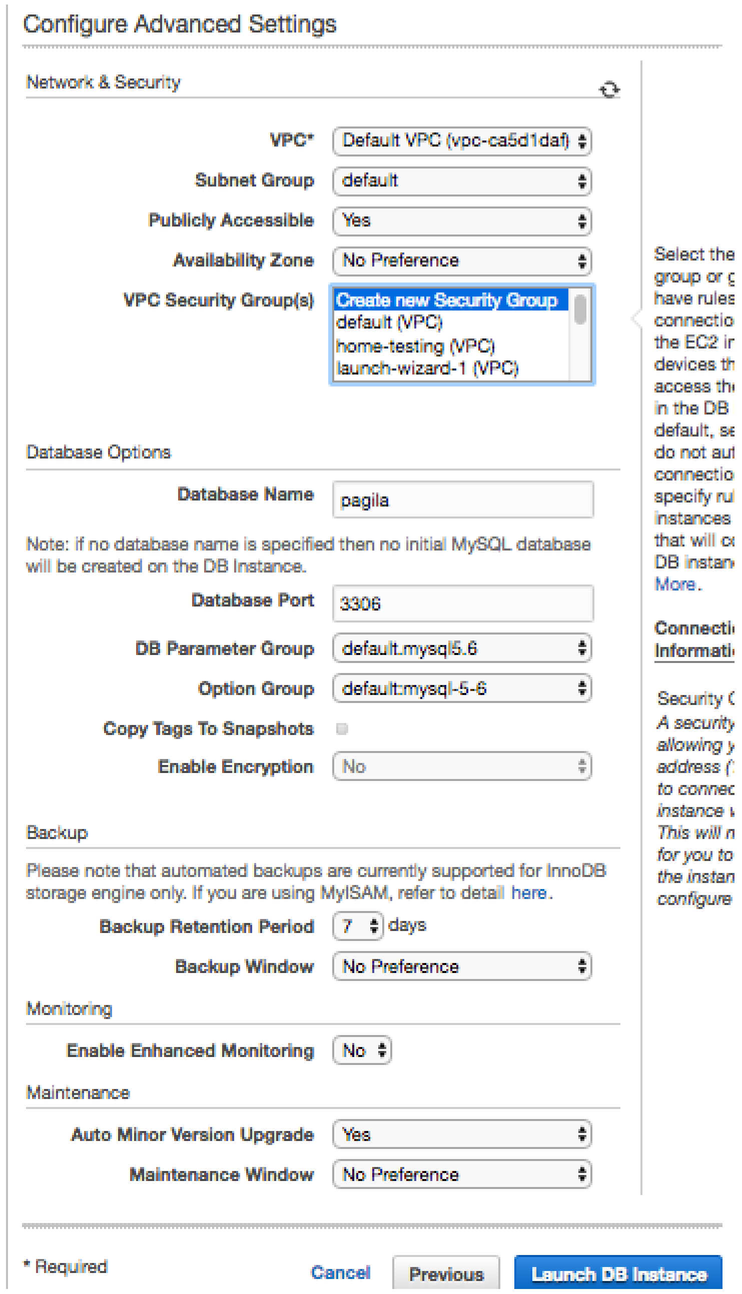
Settings

DB Instance Identifier*	pagila
Master Username*	pagila_user
Master Password*	••••••••
Confirm Password*	••••••••

Select Yes to have Amazon RDS maintain a synchronou standby replica in a differen Availability Zone than the D instance. Amazon RDS will automatically fail over to the standby in the case of a planned or unplanned outag the primary. Learn More.

* Required

Configure the advanced settings. Use the default settings for the example migration. Name the database “pagila” and select Create New Security Group within the VPC Security Groups selection box. Click Launch DB Instance.


Configure Advanced Settings

Network & Security

VPC*	Default VPC (vpc-ca5d1daf)
Subnet Group	default
Publicly Accessible	Yes
Availability Zone	No Preference
VPC Security Group(s)	Create new Security Group default (VPC) home-testing (VPC) launch-wizard-1 (VPC)

Select the group or g have rules connectio the EC2 ir devices th access th in the DB default, se do not aut connectio specify ru instances that will c DB instan More.

Connecti Informati

Security (A security allowing y address (to connec instance v This will n for you to the instan configure

Database Options

Database Name	pagila

Note: if no database name is specified then no initial MySQL database will be created on the DB Instance.

Database Port	3306
DB Parameter Group	default.mysql5.6
Option Group	default:mysql-5-6
Copy Tags To Snapshots	☐
Enable Encryption	No

Backup

Please note that automated backups are currently supported for InnoDB storage engine only. If you are using MyISAM, refer to detail here.

Backup Retention Period	7 days
Backup Window	No Preference

Monitoring

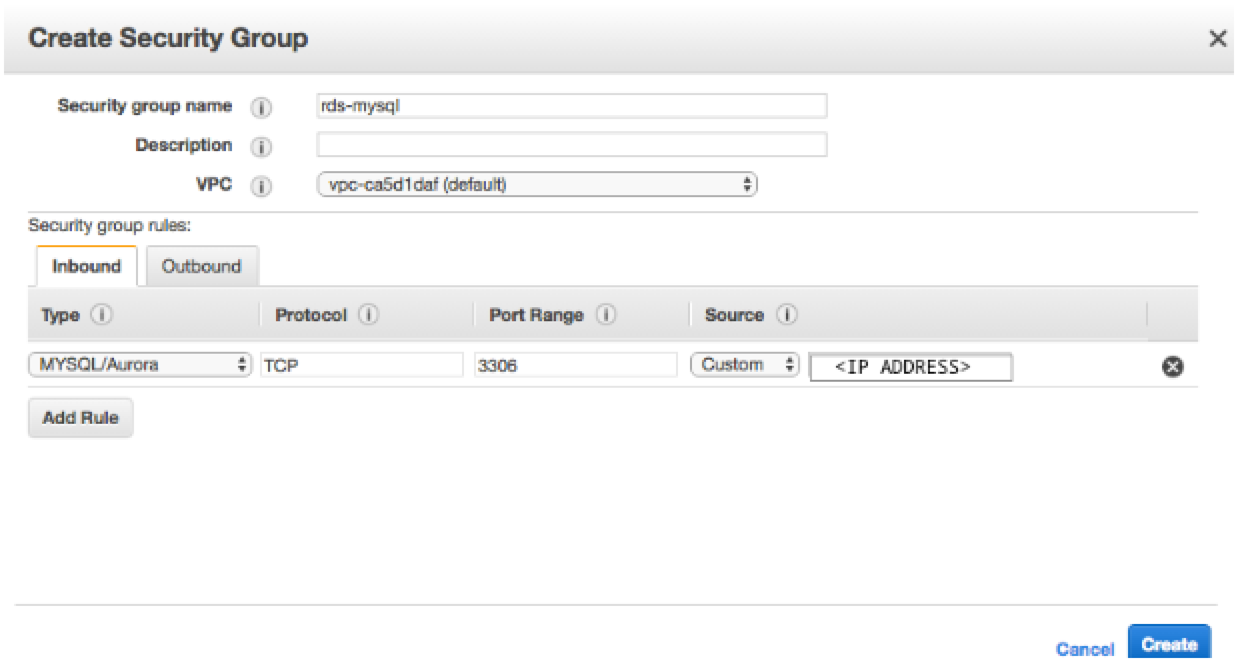
Enable Enhanced Monitoring	No

Maintenance

Auto Minor Version Upgrade	Yes
Maintenance Window	No Preference

* Required

Next, we will create and assign a new security group that will allow the schema conversion tool to connect to the RDS instance. Go to the EC2 dashboard, select Security Groups > Create Security Group. Name the security group as “rds-mysql”, add the MYSQL/Aurora rule and enter your custom IP Address or Security Group in the Source field and click Create.



Create Security Group

Security group name	rds-mysql
Description	
VPC	vpc-ca5d1daf (default)

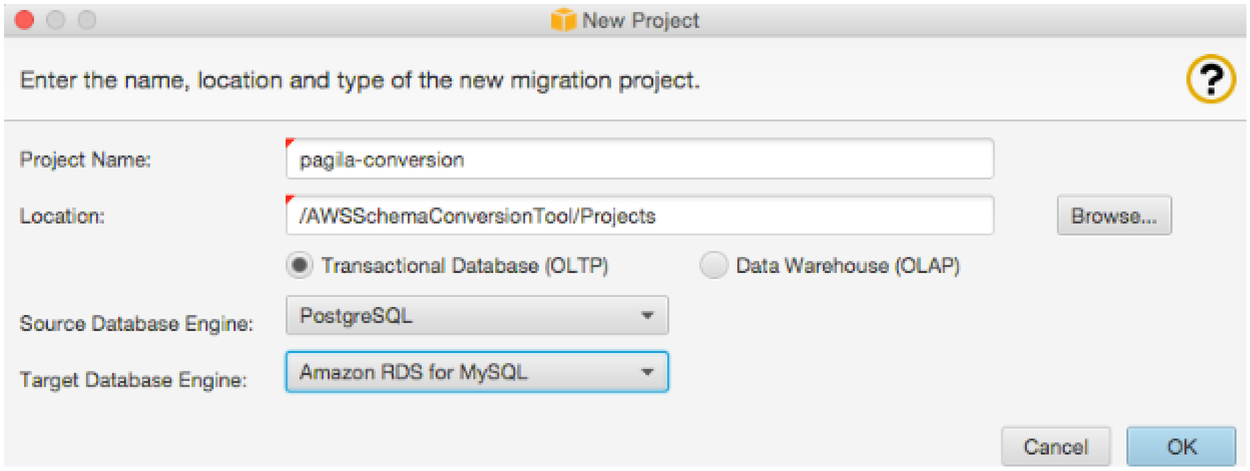
Security group rules:

Type	Protocol	Port Range	Source	
MYSQL/Aurora	TCP	3306	Custom	<IP ADDRESS>

Go to RDS > Instances and select the recently created “pagila” instance and click Instance Actions > Modify. In the Security Group field, select the “rds-mysql” security group, check the Apply Immediately checkbox at the bottom, and click Continue. You can review the settings and then click Modify DB Instance.

USING THE SCHEMA CONVERSION TOOL

-After successfully installing the AWS Schema Conversion Tool, it will be able to analyze, plan, and convert your source database. Open AWS Scheme Conversion Tool and go to File > New Project. Name project, set project location, set Source Database Engine to PostgreSQL, set Target Database Engine to Amazon RDS for MySQL and click OK.

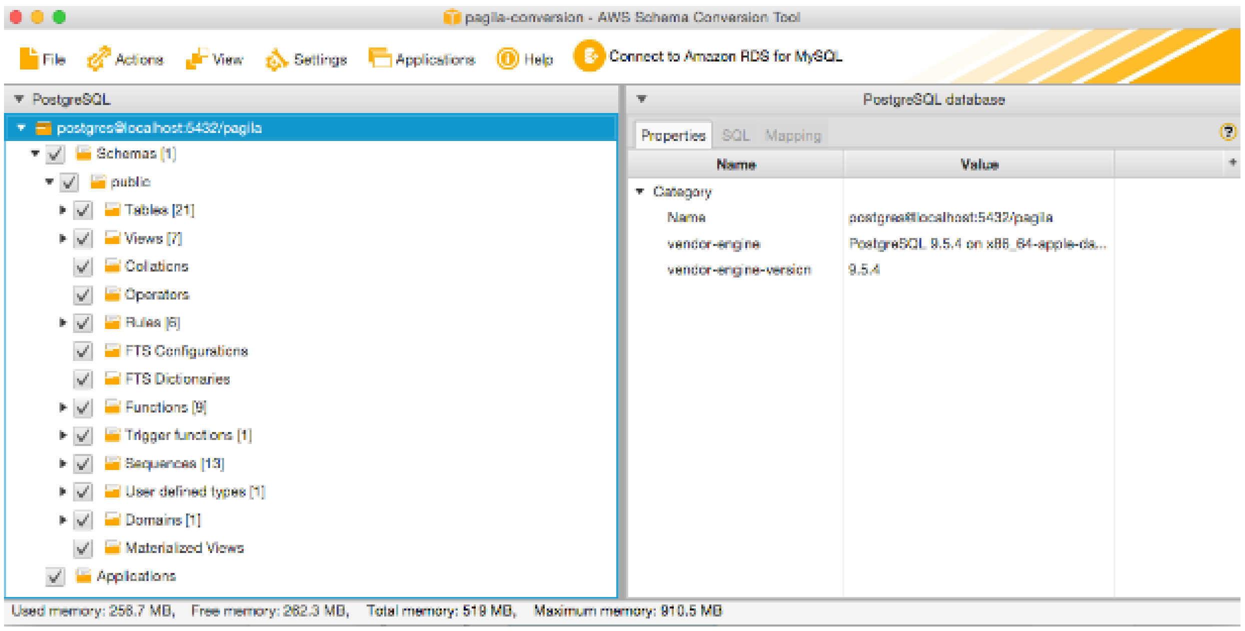


Enter the name, location and type of the new migration project.

Project Name:	pagila-conversion
Location:	/AWSSchemaConversionTool/Projects
	[● Transactional Database (OLTP)] Data Warehouse (OLAP)
Source Database Engine:	PostgreSQL
Target Database Engine:	Amazon RDS for MySQL

Select Connect to PostgreSQL in the top toolbar and enter connection information (host, name, user, password) and click OK.

**Note: You will need to download the corresponding Java Database Connectivity (JDBC) drivers for your source and target database engines and load them into the conversion tool. Amazon's list of JDBC drivers.*

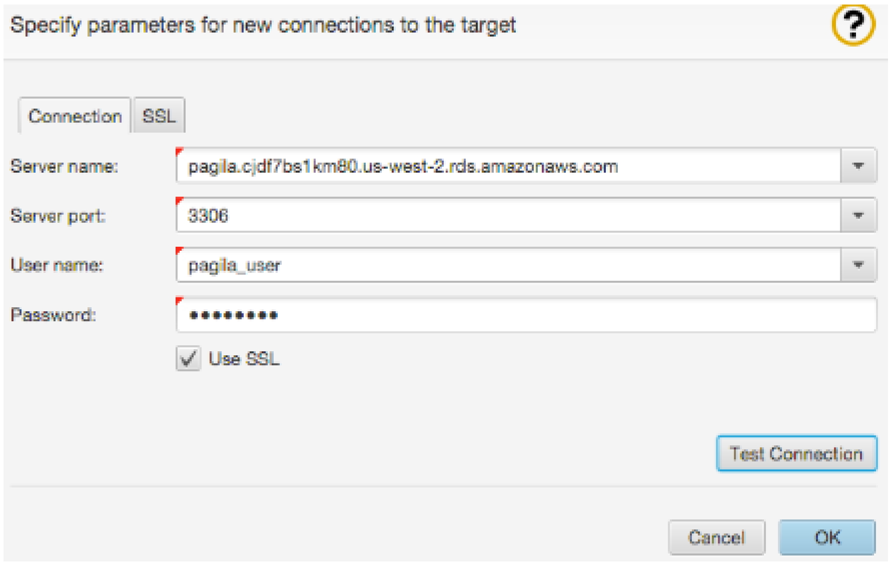


```
▼ postgres@localhost:5432/pagila
  ▼ ✓ Schemas [1]
    ▼ ✓ public
      ▶ ✓ Tables [21]
      ▶ ✓ Views [7]
        ✓ Collations
        ✓ Operators
      ▶ ✓ Rules [6]
        ✓ FTS Configurations
        ✓ FTS Dictionaries
      ▶ ✓ Functions [9]
      ▶ ✓ Trigger functions [1]
      ▶ ✓ Sequences [13]
      ▶ ✓ User defined types [1]
      ▶ ✓ Domains [1]
        ✓ Materialized Views
  ✓ Applications
```

Name	Value
▼ Category	
Name	postgres@localhost:5432/pagila
vendor-engine	PostgreSQL 9.5.4 on x86_64-apple-da...
vendor-engine-version	9.5.4

Used memory: 256.7 MB, Free memory: 262.3 MB, Total memory: 519 MB, Maximum memory: 910.5 MB

Once the source database has been loaded. Select Connect to Amazon RDS for MySQL in the top toolbar and enter the connection information from your earlier RDS deployment.



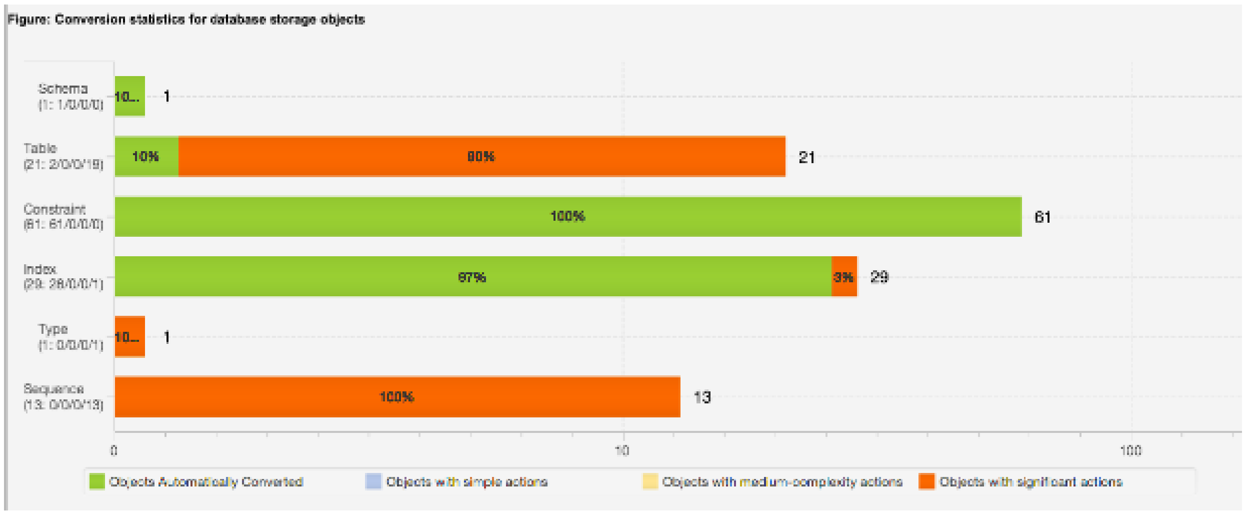
Specify parameters for new connections to the target

Connection | SSL

Server name:	pagila.cjdf7bs1km80.us-west-2.rds.amazonaws.com
Server port:	3306
User name:	pagila_user
Password:	●●●●●●●●●

☑ Use SSL

The schema conversion tool can now generate a summary report regarding the conversion. For example, select the Public schema from the source database and then in the top toolbar click Actions > Create Report. This report summarizes schema conversion statistics and possible issues regarding tables, views, data types, domains, indexes, triggers, etc.



Graph of database object conversion actions

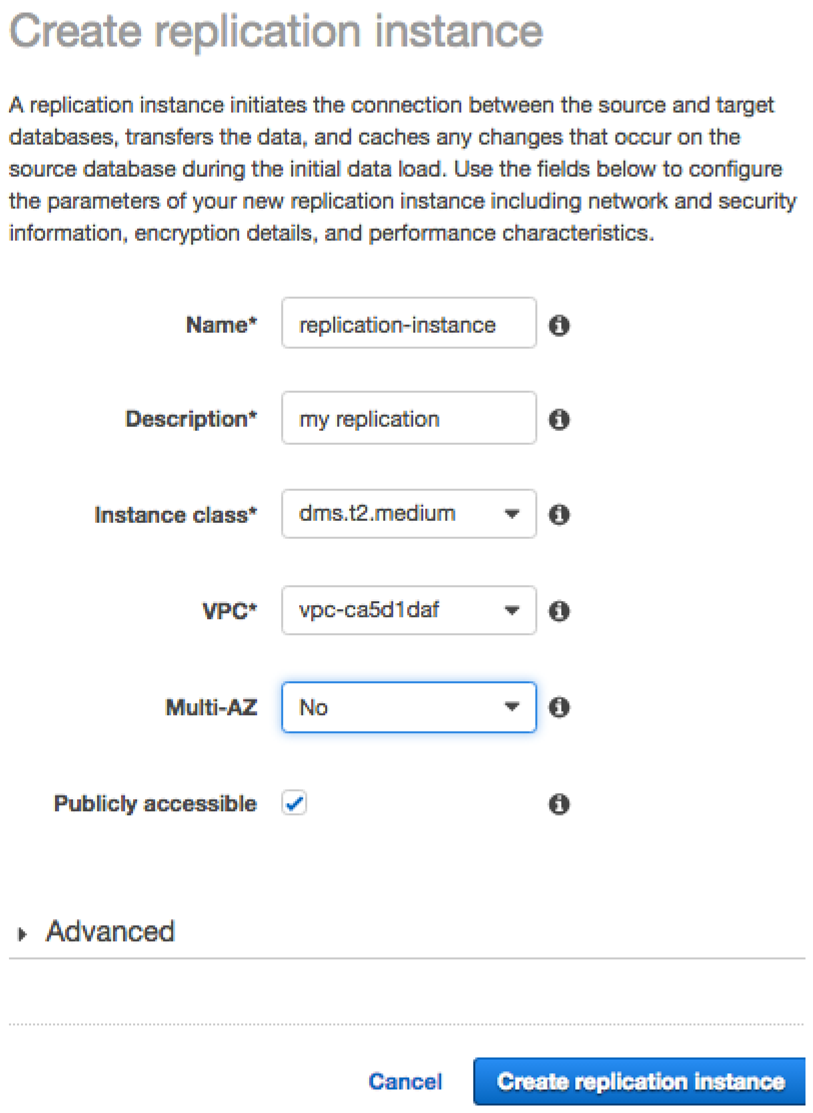
The report has also created the schema conversion from the source to the target. Depending migration goals, the tool provides schema Mapping tools to convert specific subsets, tables, data types, functions, etc of the database. Now, you are able to create a new DMS replication instance to start migrations tasks.

MIGRATING THE DATA

AWS DMS will deploy a replication server to run migrations, create endpoints to connect to source and target databases, and create tasks to migrate data and tables.

DEPLOYING REPLICATION INSTANCE

Go to DMS dashboard > Replication Instance > Create Replication Instance. Add a name, description, instance type, and VPC and then click Create Replication Instance.



Create replication instance

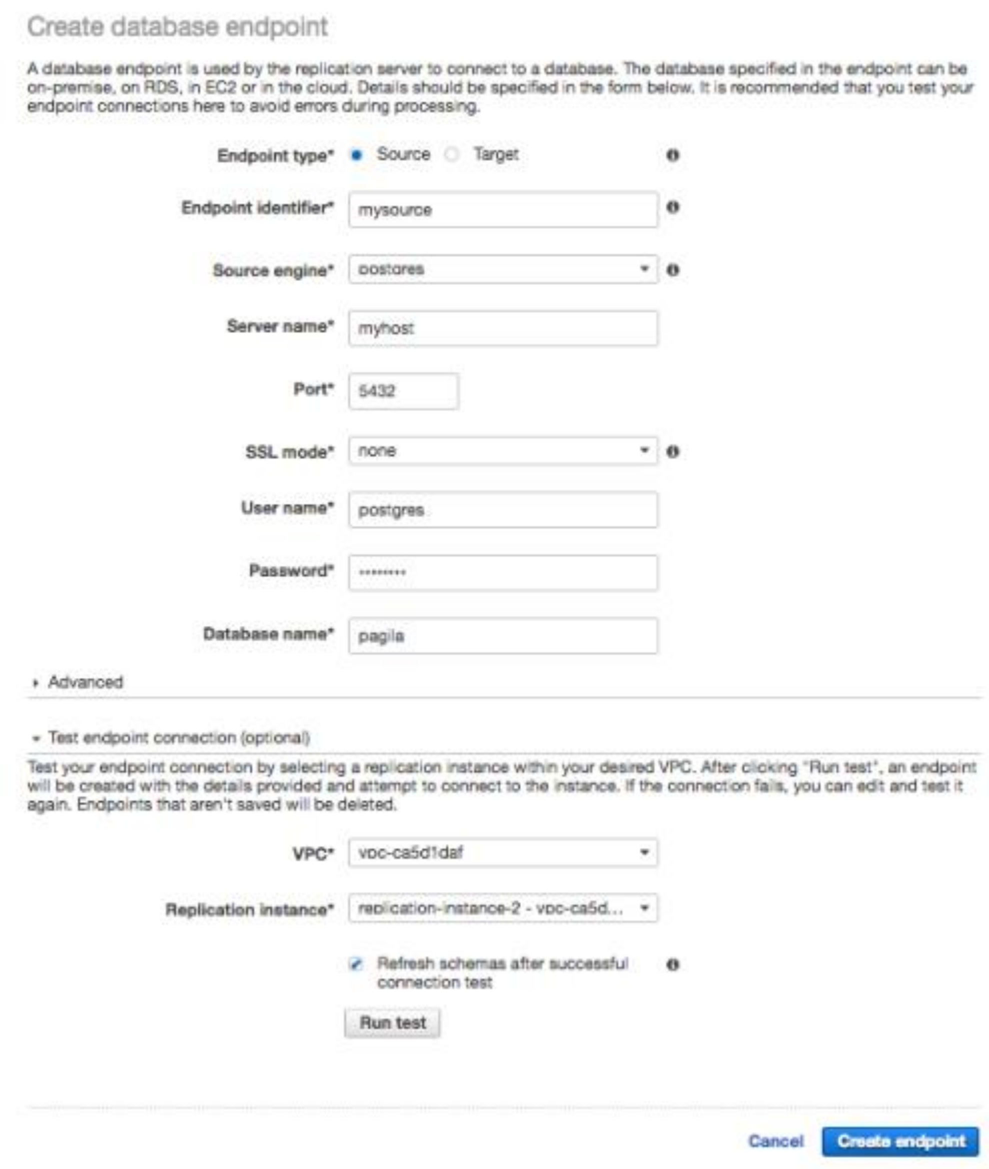
A replication instance initiates the connection between the source and target databases, transfers the data, and caches any changes that occur on the source database during the initial data load. Use the fields below to configure the parameters of your new replication instance including network and security information, encryption details, and performance characteristics.

Name*	replication-instance
Description*	my replication
Instance class*	dms.t2.medium
VPC*	vpc-ca5d1daf
Multi-AZ	No
Publicly accessible	✓

Depending on the size of your source database, you will want to choose the appropriate instance type and allocate the necessary storage space. DMS replication stores most of the conversion objects in RAM but will use disk storage for large transactions. Table sizes and data types will be a major determining factor.

CREATING SOURCE AND TARGET ENDPOINTS

Go to DMS Dashboard > Endpoints > Create Endpoint. Select Endpoint Type Source, enter an endpoint identifier name, the host, database type, and credentials for the example PostgreSQL instance. Click Run Test to verify connection then click Create Endpoint.



Create database endpoint

A database endpoint is used by the replication server to connect to a database. The database specified in the endpoint can be on-premise, on RDS, in EC2 or in the cloud. Details should be specified in the form below. It is recommended that you test your endpoint connections here to avoid errors during processing.

Endpoint type*	[✓ Source] Target
Endpoint identifier*	mysource
Source engine*	postgres
Server name*	myhost
Port*	5432
SSL mode*	none
User name*	postgres
Password*	*********
Database name*	pagila

▸ Advanced

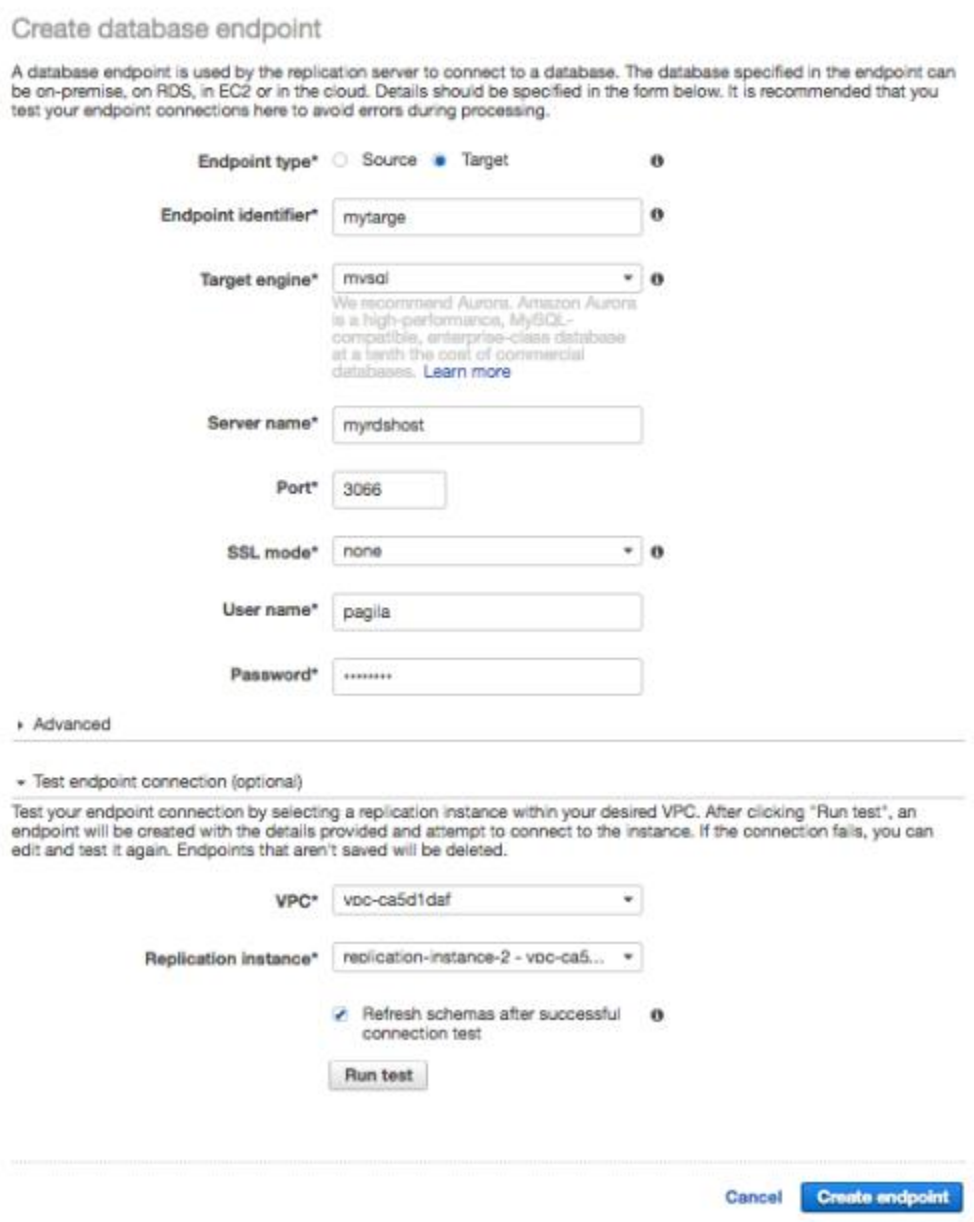
▾ Test endpoint connection (optional)

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-ca5d1daf
Replication instance*	replication-instance-2 - vpc-ca5d...

[✓] Refresh schemas after successful connection test

Repeat the last step but create an endpoint for the target MySQL RDS database.



Create database endpoint

A database endpoint is used by the replication server to connect to a database. The database specified in the endpoint can be on-premise, on RDS, in EC2 or in the cloud. Details should be specified in the form below. It is recommended that you test your endpoint connections here to avoid errors during processing.

Endpoint type*	Source [✓ Target]
Endpoint identifier*	mytarge
Target engine*	mysql We recommend Aurora. Amazon Aurora is a high-performance, MySQL-compatible, enterprise-class database at a tenth the cost of commercial databases. Learn more
Server name*	myrdshost
Port*	3066
SSL mode*	none
User name*	pagila
Password*	•••••••••

▸ Advanced

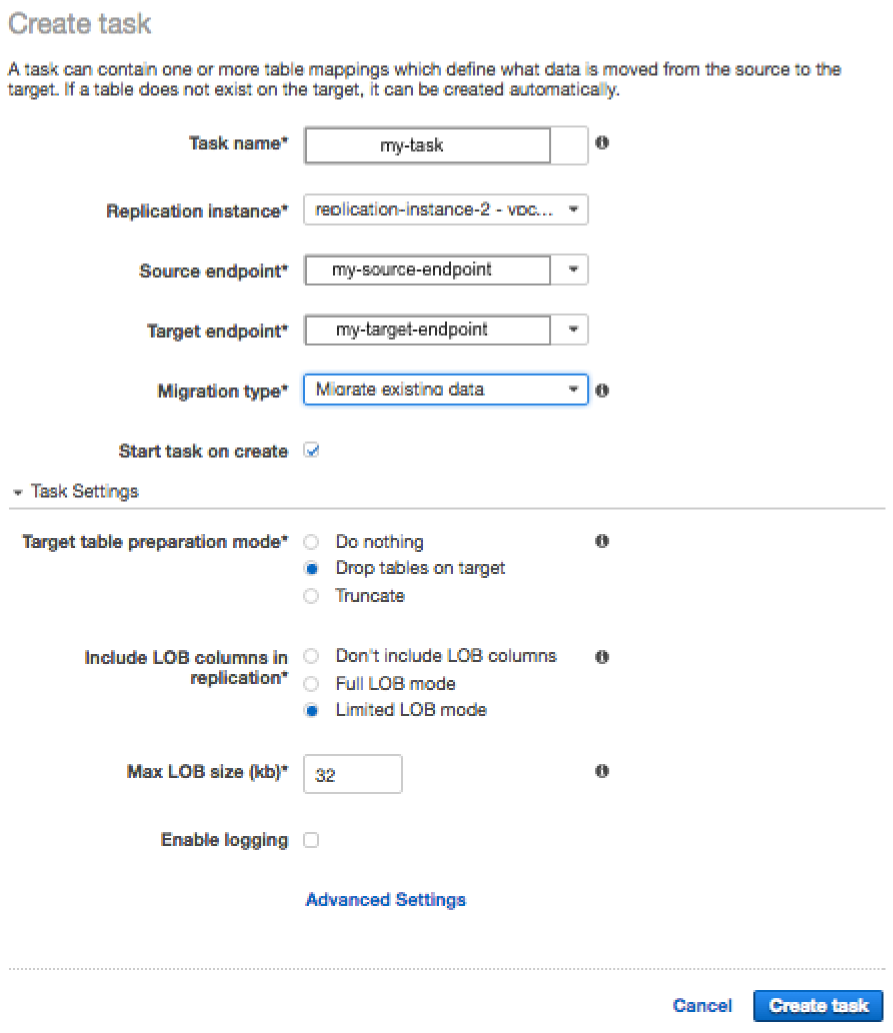
▾ Test endpoint connection (optional)

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-ca5d1daf
Replication instance*	replication-instance-2 - vpc-ca5...
	✓ Refresh schemas after successful connection test

CREATING AND RUNNING THE TASKS

Go to DMS Dashboard > Tasks > Create Task. Name the task, specify the replication instance, enter the source and target endpoints, and migration type. DMS supports three migrations type: Migrate Existing Data, Migrate Existing Data and replicate ongoing changes, and Replicate ongoing changes. Click Create Task and by default DMS will start the selected migration on task create.



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*: my-task

Replication instance*: replication-instance-2 - vpc...

Source endpoint*: my-source-endpoint

Target endpoint*: my-target-endpoint

Migration type*: Migrate existing data

Start task on create: ✓

Task Settings

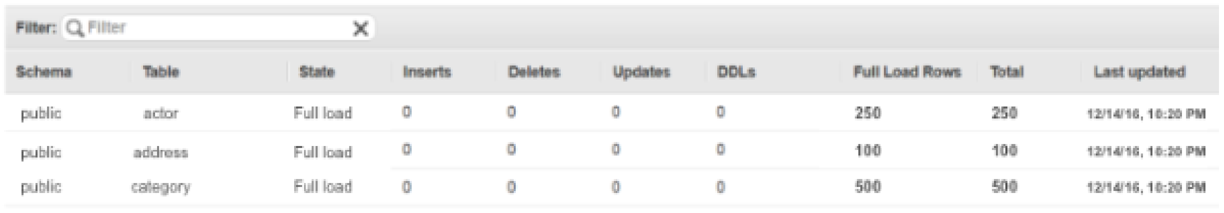
Target table preparation mode*: Do nothing [✓ Drop tables on target] Truncate

Include LOB columns in replication*: Don't include LOB columns Full LOB mode [✓ Limited LOB mode]

Max LOB size (kb)*: 32

Enable logging: ☐

Once the tasks complete. A report of the records migrated by schema and table can be found in the Table Statistics tab.



Filter: Filter

Schema	Table	State	Inserts	Deletes	Updates	DDLs	Full Load Rows	Total	Last updated
public	actor	Full load	0	0	0	0	250	250	12/14/16, 10:20 PM
public	address	Full load	0	0	0	0	100	100	12/14/16, 10:20 PM
public	category	Full load	0	0	0	0	500	500	12/14/16, 10:20 PM

DMS also provides an Overview, Monitoring, and Logs to view CPU and disk storage and individual record conversions. After all tasks are complete, your RDS instance is now up to date.

FINAL THOUGHTS

Database migrations will have pain points and estimating completion time comes down to connectivity and machine resources. Conversions between database engines are always going to be more difficult and defining migration task scope will help to alleviate issues. You'll be successful using DMS when you understand your source data, its constraints, and define attainable goals.

ADDITIONAL RESOURCES

AWS DMS Getting Started

FAQs

User Guide

AWS Schema Conversion Tool