aws training and certification

Migrating from PostgreSQL to Amazon RDS

A step-by-step guide for migrating a self-managed PostgreSQL database instance to a fully managed database on Amazon Relational Database Service (Amazon RDS)

Steps:

1. Create a PostgreSQL database instance in Amazon RDS

In this step, you will create a PostgreSQL database instance in Amazon RDS. This instance will be used as your primary database after you copy your existing data into it by using AWS Database Migration Service, or AWS DMS.

To get started, navigate to the Amazon RDS console. On the main page, choose **Create database**.
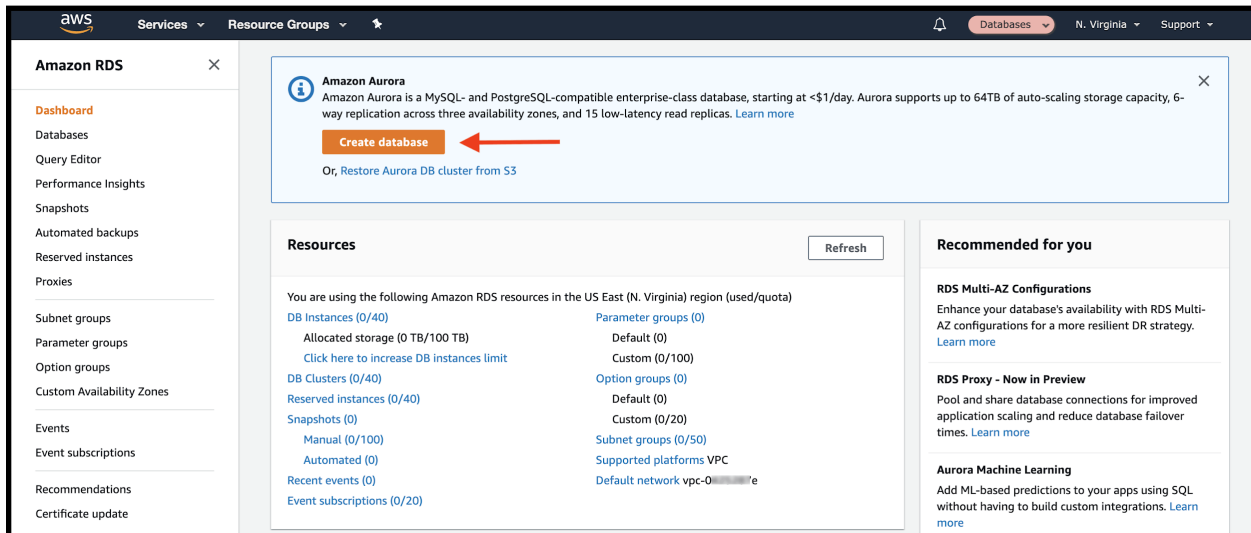
This initiates the database creation wizard. In the **Engine options** section, choose **PostgreSQL** as your **Engine type**. Then, choose the version of PostgreSQL you want to use. Note that AWS DMS supports PostgreSQL versions 9.4 and later (9.x), 10.x, and 11.x.

Choose a database creation method Info

○ **Standard Create**
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

○ **Easy Create**
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type   Info

○ Amazon Aurora

○ MySQL

○ MariaDB

● PostgreSQL

○ Oracle

○ Microsoft SQL Server

Version   Info

PostgreSQL 11.5-R1 ▼

ⓘ If you want to create PostgreSQL 12 in the Preview environment, click here.

The database creation wizard includes templates to make it easier to configure the settings for your Amazon RDS database. If you are creating this database to be used in production, you should choose the production template.

In the **Settings** section, give your database a name, and set the master username and password. Make sure you write these down because you will need them to connect to your database and create additional users.

Settings

DB instance identifier *Info*

Type a name for your DB instance. The name must be unique cross all DB instances owned by your AWS account in the current AWS Region.

```
my-database
```

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ **Credentials Settings**

Master username *Info*

Type a login ID for the master user of your DB instance.

```
postgres
```

1 to 16 alphanumeric characters. First character must be a letter

☐ Auto generate a password
   Amazon RDS can generate a password for you, or you can specify your own password

Master password *Info*

```
••••••••
```

Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), "(double quote) and @ (at sign).

Confirm password *Info*

```
••••••••
```

Next, choose the database instance size. You should choose this based on your estimated capacity. If you are managing your own database on [Amazon Elastic Cloud Compute](#), or Amazon EC2, you can compare your current Amazon EC2 instance size to an Amazon RDS instance size.

If you want to increase or decrease your database instance size in the future, Amazon RDS enables you to do that easily. However, you may incur some downtime to change your database size.

DB instance size

DB instance class Info

Choose a DB instance class that meets your processing power and memory requirements. The DB instance class options below are limited to those supported by the engine you selected above.

○ Standard classes (includes m classes)
○ Memory Optimized classes (includes r and x classes)
○ Burstable classes (includes t classes)

db.m5.large
2 vCPUs   8 GiB RAM   EBS: 3500 Mbps   ▼

⬤ Include previous generation classes

Next, configure the storage options for your Amazon RDS database. There are two storage options in Amazon RDS: general purpose and provisioned IOPS. With general purpose storage, you receive 3 IOPS (I/O operations per second) per GiB of storage allocated. Thus, 100 GiB of storage would have 300 IOPS. Additionally, you receive burst capacity up to 3,000 IOPS.

With provisioned storage, you provision IOPS separately from your storage capacity. These two items are charged separately, allowing you to fine-tune your storage and operations settings to fit your needs.

Storage

Storage type Info

General Purpose (SSD)   ▼

Allocated storage

100   GiB

(Minimum: 20 GiB, Maximum: 65536 GiB) Higher allocated storage **may improve** IOPS performance.

Storage autoscaling Info

Provides dynamic scaling support for your database's storage based on your application's needs.

☑ Enable storage autoscaling
Enabling this feature will allow the storage to increase once the specified threshold is exceeded.

Maximum storage threshold Info

Charges will apply when your database autoscales to the specified threshold

1000   GiB

Minimum: 101 GiB, Maximum: 65536 GiB

Next, decide whether to create a standby instance. A standby instance is a replica of your data that is available in the event of failure. It is located in a different Availability Zone in the same AWS Region as your Amazon RDS database to limit the impact of infrastructure failures. If you are running a production database in which uptime is essential, a standby instance is recommended.

Availability & durability

Multi-AZ deployment **Info**

● **Create a standby instance (recommended for production usage)**
   Creates a standby in a different Availability Zone (AZ) to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.

○ **Do not create a standby instance**

The next section in the Amazon RDS database creation wizard is about connectivity. You must specify the Amazon Virtual Private Cloud, or Amazon VPC, in which your database will reside, as well as the network subnet and security groups for your database instance.

If you are migrating from a self-managed database instance on Amazon EC2, you can use the same Amazon VPC and security groups as your existing database.

If you are migrating from a database that is not hosted on Amazon Web Services (AWS) but your application is hosted on AWS, choose the same Amazon VPC that is used for your application. Then create a new security group for your database instance.

Connectivity

Virtual Private Cloud (VPC) Info
VPC that defines the virtual networking environment for this DB instance.

Default VPC (vpc-0   e) ▼

Only VPCs with a corresponding DB subnet group are listed.

ⓘ After a database is created, you can't change the VPC selection.

▼ **Additional connectivity configuration**

Subnet group Info
DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.

default-vpc-0   e ▼

Publicly accessible Info

○ Yes
  Amazon EC2 instances and devices outside the VPC can connect to your database. Choose one or more VPC security groups that specify which EC2 instances and devices inside the VPC can connect to the database.

● No
  RDS will not assign a public IP address to the database. Only Amazon EC2 instances and devices inside the VPC can connect to your database.

VPC security group
Choose one or more RDS security groups to allow access to your database. Ensure that the security group rules allow incoming traffic from EC2 instances and devices outside your VPC. (Security groups are required for publicly accessible databases.)

○ **Choose existing**	● **Create new**
Choose existing VPC security groups	Create new VPC security group

New VPC security group name

postgresql-database

Database port Info
TCP/IP port the database will use for application connections.

5432

Finally, choose the database authentication methods to allow in your database. PostgreSQL databases traditionally allow for username and password authentication. With PostgreSQL on Amazon RDS, you also can choose to allow for authentication using AWS Identity and Access Management, or IAM. This integrates easily with your application and removes the need for credential rotation.

It is recommended that you allow for password and IAM database authentication. If you only want to start with password authentication, you can add IAM database

authentication later, but it will result in some downtime for your Amazon RDS database instance.

Database authentication

Database authentication options   Info

○ **Password authentication**
Authenticates using database passwords.

● **Password and IAM database authentication**
Authenticates using the database password and user credentials through AWS IAM users and roles.

○ **Password and Kerberos authentication**
Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.

You can configure some additional configuration options, including settings on backups, monitoring, maintenance, and automated upgrades. The default settings work for most situations, but you should review them to ensure they work for your needs.

▶ **Additional configuration**
Database options, encryption enabled, backup enabled, backtrack disabled, Performance Insights enabled, Enhanced Monitoring enabled, maintenance, CloudWatch Logs, delete protection enabled

The end of the database creation wizard shows you the estimated monthly costs for your database instance. Choose **Create database** to create your database instance.

Estimated monthly costs

DB instance	129.94 USD
Storage	11.50 USD
Multi-AZ standby instance	129.94 USD
Total	**271.38 USD**

This billing estimate is based on on-demand usage as described in Amazon RDS Pricing ⬀. Estimate does not include costs for backup storage, IOs (if applicable), or data transfer.

Estimate your monthly costs for the DB Instance using the AWS Simple Monthly Calculator ⬀.

ⓘ You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

Create database

Amazon RDS begins creating your database. As Amazon RDS is provisioning your infrastructure and initializing your database, its **Status** is *Creating*.

When your database is ready to use, its **Status** is *Available*.

In this step, you created a fully managed, production-ready PostgreSQL database instance in Amazon RDS. In the next step, you will create a replication instance in AWS DMS.

In this step, you will create a replication instance in AWS DMS.

AWS DMS is a service you can use to copy data from an existing database into a fully managed AWS database. A replication instance is an Amazon EC2 instance that can host replication tasks in AWS DMS. In the next step, you will create endpoints.

Before using AWS DMS to migrate your existing PostgreSQL database to a PostgreSQL database in Amazon RDS, you should review the AWS DMS documentation about using a PostgreSQL database as a source and using a PostgreSQL database as a target. In certain cases, it might make sense to use native tools such as pg_dump and pg_restore to migrate your database.

To create a replication instance, navigate to the Replication instances section of the AWS DMS console. Choose **Create replication instance** to begin the replication instance creation wizard.
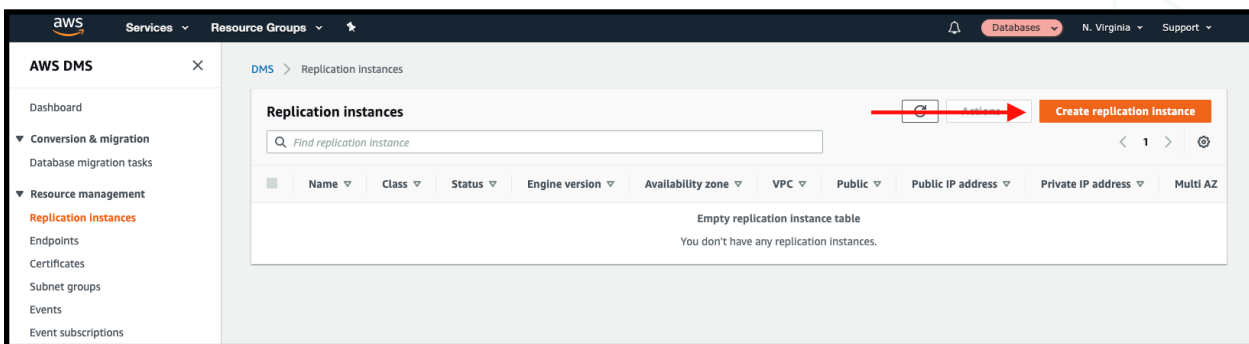
In the **Replication instance configuration** section, give your replication instance a name and description. Then choose your instance class. The instance class you use depends on the size of your existing database and the amount of data flowing through it.

Then choose an engine version for AWS DMS. Finally, choose the amount of allocated storage for your replication instance.

Replication instance configuration

Name
The name must be unique among all of your replication instances in the current AWS region.

myReplicationInstance

Replication instance name must not start with a numeric value

Description

Replication instance for moving to managed PostgreSQL in RDS

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.t2.medium ▼

Billing is based on DMS pricing ⤴.

Engine version
Choose an AWS DMS version to run on your replication instance.

3.3.1 ▼

Allocated storage (GiB)
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

As you continue in the **Replication instance configuration** section, you need to choose an Amazon VPC for your replication instance. Choose the same Amazon VPC in which you provisioned your Amazon RDS database to ease network access for the replication instance.

You can choose to have a Multi-AZ setup for your replication instance for redundancy. If you are using AWS DMS to keep two databases in sync over a long period of time, you might want to use a Multi-AZ setup. If you are performing a one-time migration of your data from an existing database to a fully managed database in Amazon RDS, you likely don't need a Multi-AZ setup.

Finally, choose whether your replication instance should be publicly accessible. If your existing database is in the same Amazon VPC as your new database and your replication instance, you don't need your replication instance to be publicly accessible. If your existing database is not in the same VPC, you might need your replication instance to be publicly accessible. If possible, you should avoid having your replication instance publicly accessible to limit security concerns.

VPC

Choose an Amazon Virtual Private Cloud (VPC) where your replication instance should run.

vpc-0⬛⬛⬛'e ▼

☐ **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.

Next, open the **Advanced security and network configuration** section. For the **VPC security group(s)** configuration, choose the same security group that you attached to your Amazon RDS database. This allows your replication instance to access your Amazon RDS database.

aws training and certification

You can also edit the **Maintenance** and **Tags** settings.

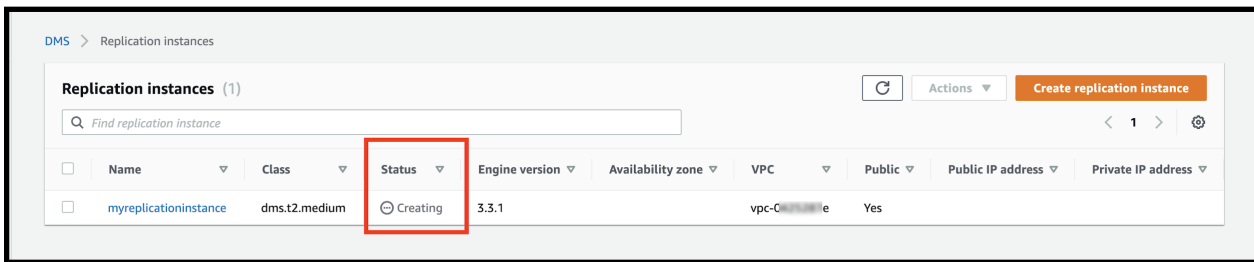When you're ready, choose **Create** to create your replication instance in AWS DMS.

After you choose **Create**, AWS provisions your replication instance. A **Status** of *Creating* is displayed while AWS is provisioning and initializing your instance.

When your replication instance is ready to go, its status is *Available*.
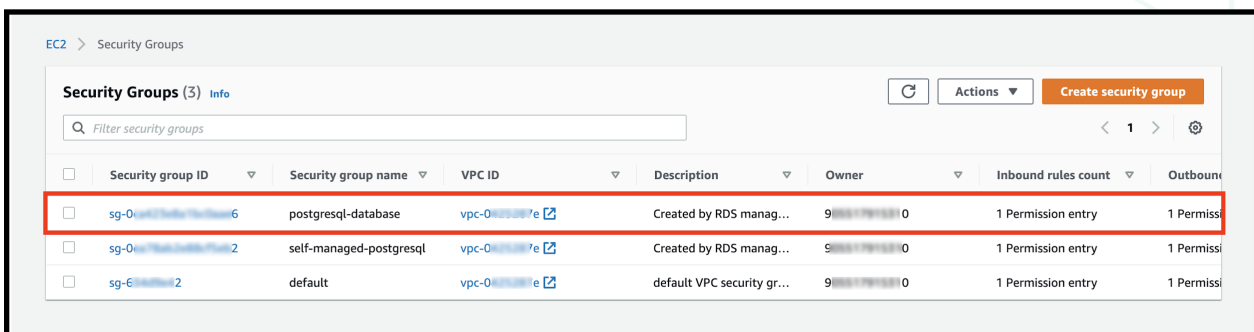
While you are waiting for your replication instance to be available, go to the Security Groups section of the Amazon EC2 console. You need to add a rule to your security group to allow your replication instance to access your database.
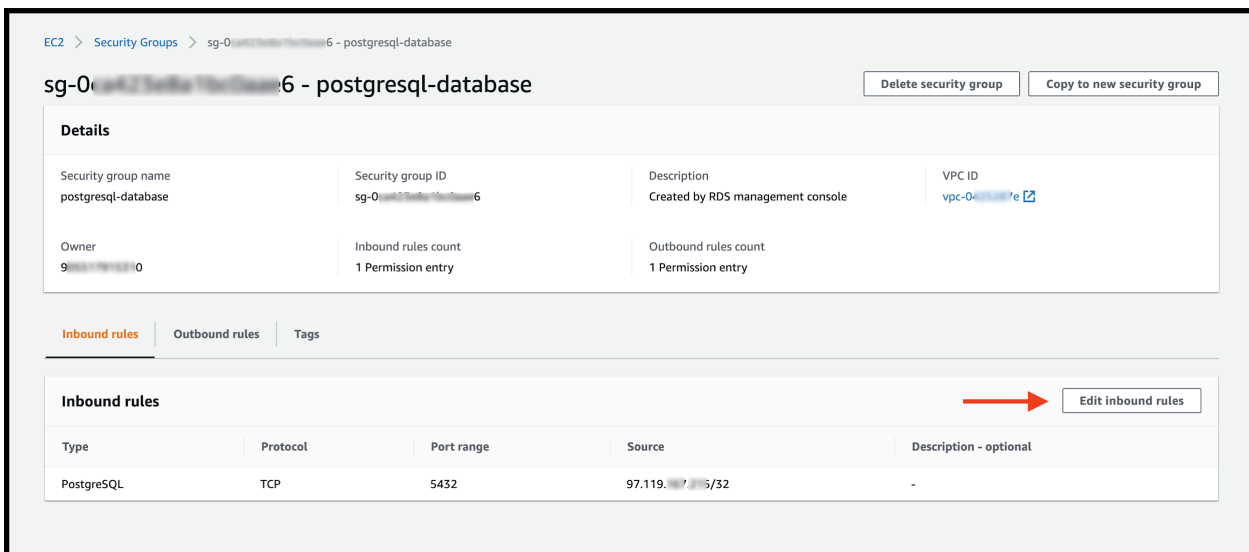
In the **Security Groups** section, find the security group you attached to your PostgreSQL database instance and your replication instance, and choose it.

Choose to **Edit inbound rules** for your security group.

Your security group has an existing rule that allows for access to your PostgreSQL instance from the IP address you used to create the database. Remove the existing IP address and enter the name of the security group used for your Amazon RDS database instance and replication instance.

Your screen should look like the following.
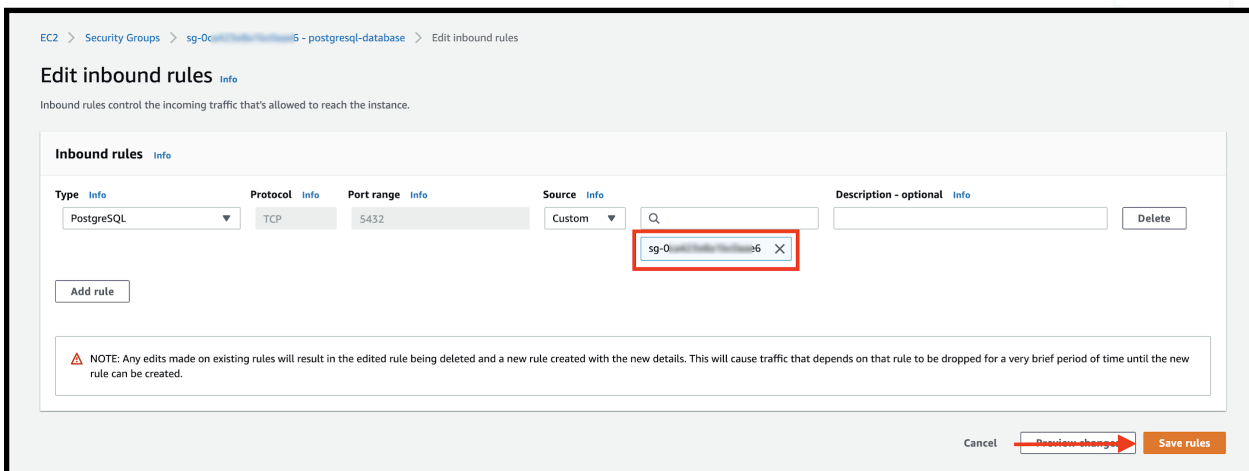
Choose **Save rules** to save the updated rules for your security group.

When your replication instance is available and you have updated the rules for your security group, you can move to the next step.

In this step, you created a replication instance in AWS DMS. The replication instance hosts the replication tasks that migrate data from an existing database to a fully managed database in Amazon RDS. You also updated a security group to allow access from your replication instance to your PostgreSQL database instance in Amazon RDS.

In the next step, you will create endpoints for your source and target databases in Amazon RDS.

3. Create endpoints in AWS DMS

In this step, you will create source and target endpoints for a replication task in AWS DMS.

A replication task is a job that migrates data from one database to another using AWS DMS. Before creating a replication task, you must register endpoints for your source and target databases. An endpoint describes the connection address, credentials, and other information required to connect to a database.

First, create the endpoint for your target database. This is the database you created in Amazon RDS.

Navigate to the Endpoints section of the AWS DMS console. Choose **Create endpoint** to create a new endpoint.
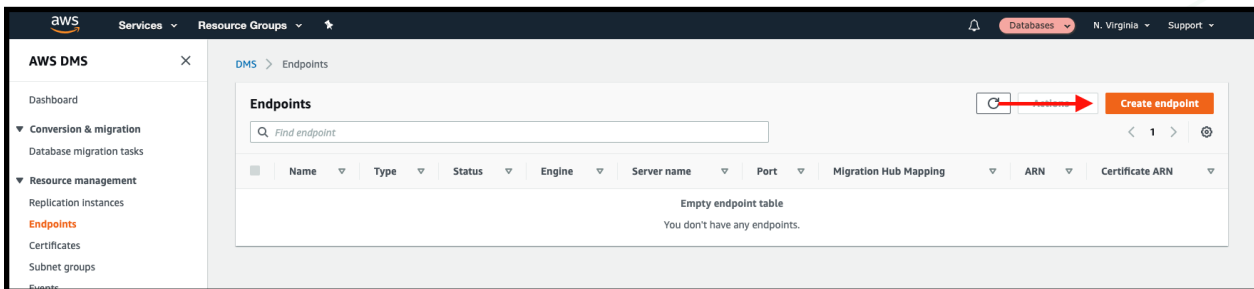
In the endpoint creation wizard, choose to create a **Target endpoint**. Select **Select RDS DB instance**, and choose your newly created Amazon RDS database in the dropdown.
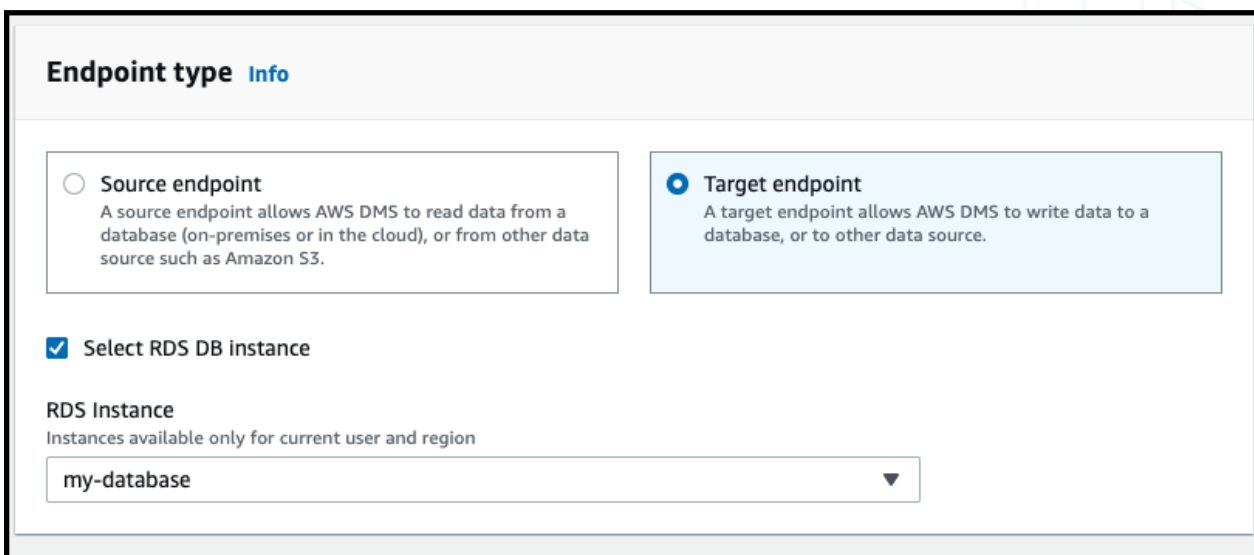
This should fill in most of the **Endpoint configuration** details for you. You need to enter your password and a database name near the bottom of the section.

Endpoint configuration

Endpoint identifier Info
A label for the endpoint to help you identify it.

```
my-database
```

Target engine
The type of database engine this endpoint is connected to.

```
postgres                                                    ▼
```

Server name

```
my-database.c          s.us-east-1.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

```
postgres
```

Password Info

```
••••••••
```

Database name

```
postgres
```

Before you save your endpoint, you should test the connection to ensure it was configured correctly. Open the **Test endpoint connection** section to test your connection.

Choose the replication instance you want to use, and then choose **Run test**. After a few seconds, you should see a **Status** of *successful*. This indicates that you configured your security group and endpoint correctly. Choose **Create endpoint** to save your endpoint.

Follow these steps again to create an endpoint for your source database. Unlike the target database, you need to fill out the connection endpoint, port, and credentials yourself.

You also need to ensure that your replication instance has network access to your source database. If your source database is hosted on Amazon EC2, allow traffic from your replication instance security group to the source database security group. If your source database is not hosted on Amazon EC2, you need to configure the network settings according to the location of your source database.

Before moving to the next step, you should have two endpoints configured: one for your source database and one for your target database. Make sure that you have tested both endpoints and can successfully connect to both databases. Then move to the next step.

In this step, you created your endpoints to connect to your databases. In the next

step, you will use those endpoints to create a replication task that copies data from your source database to your target database.

4. Create a replication task in AWS DMS

In this step, you will create a replication task in AWS DMS.

A replication task is responsible for migrating data from a source database to a target database. In your case, you are moving data from an existing database to your newly created database in Amazon RDS.

To get started, navigate to the Database migration tasks section the AWS DMS console, and choose **Create task** to create a new replication task.
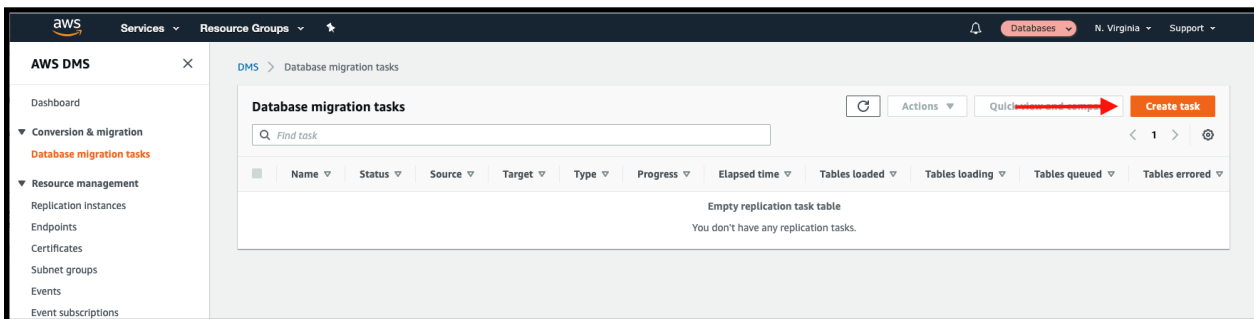
In the **Task configuration** section, set up the parameters of your replication task. Give your task a name, and choose the replication instance you created in an earlier step. Then choose the source endpoint for your existing database and your target endpoint for your fully managed database in Amazon RDS.

You also need to select a migration type. There are two migration types:

1. **Migrate existing data**, which performs a one-time process to copy data from your source database to your target database

2. **Replicate data changes**, which copies all ongoing operations from your source database to your target database

If you are migrating your application from using a self-managed database to using a fully managed database, you should use both types. The first type copies all data in your database, and the second type ensures that all additional updates are replicated to your new database until you switch your application to use the new database.

For the migration type, choose **Migrate existing data and replicate ongoing changes**. Note that this requires you to have logical replication enabled on your source database.

Task configuration

Task identifier

```
postgresql-migration
```

Replication instance

```
myreplicationinstance - vpc-0      7e          ▼
```

Source database endpoint

```
self-managed-postgresql                         ▼
```

Target database endpoint

```
my-database                                     ▼
```

Migration type **Info**

```
Migrate existing data and replicate ongoing changes  ▼
```

> ⓘ 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

In the **Table mappings** section, select the tables you want to copy over. Enter the name of the schemas and tables you want to copy. You can use % as a wildcard character to copy multiple tables or schemas.

In this walkthrough, you will migrate your tables exactly as they exist in the source database. If you have more complex requirements,
AWS DMS supports filters and transformations of your source data before writing to your target database.

When you are ready, choose **Create task** to start your migration task. After you create your task, your task is shown in the **Database migration tasks** section with a **Status** of *Creating*.

After the task is initialized, its **Status** is *Starting*.

After the migration of existing data is complete, it shows a **Status** of *Load complete, replication ongoing.* Any updates to your source database at this point are copied to your target database.

In this step, you created a replication task in AWS DMS to migrate your existing data and sync ongoing changes from your previous database to your new database in Amazon RDS.

In the next step, you will learn about cleaning up the resources you created and next steps.

If you followed all the steps in this lesson, you created a new, fully managed database in Amazon RDS and created a migration task to copy data from your source database to your new database. In this final step, you will learn the steps to complete your migration and clean up your AWS DMS resources.

When your initial migration is complete and all data is synced to your new database, you are ready to use your new database as your primary database.

There are two different ways you can handle this:

1. If you feel confident in the correctness of the migration, you can change the database configuration in your application to use your new database. This helps ensure that all reads and writes go to your new database.

2. If you want to follow a more cautious approach, you can read from and write to both databases for a period of time. This enables you to compare the results from each database for correctness while still maintaining the correct data in your existing database. Additionally, you can use the validation feature in AWS DMS to compare the source and target records.

Although we used AWS DMS in this lesson, there are other options to migrate PostgreSQL databases. You can use the pg_dump and pg_restore client applications to create a data export file and then load it into your new database. Additionally, you can use native logical PostgreSQL replication to migrate ongoing changes to a new database.

After you have switched to using your primary database and are confident in the results, you might want to delete your AWS DMS infrastructure.

First, stop and delete the database migration task to replicate your data. Navigate to the Database migration tasks section of the AWS DMS console. Choose the task you want to remove, and choose **Stop**.
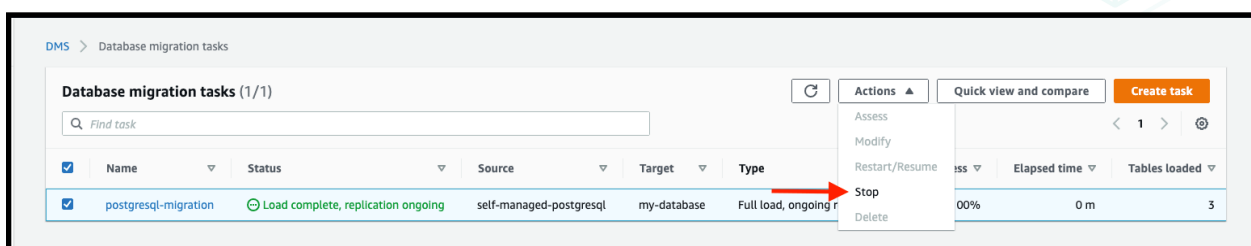
It will take a few moments to stop the task. When the task has stopped, choose it again, and choose **Delete**.
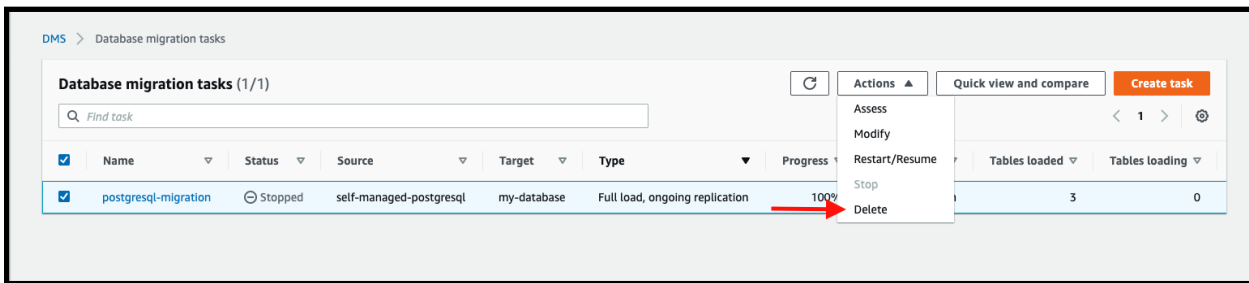
Next, navigate to the Endpoints section of the AWS DMS console. Choose your source endpoint and your target endpoint, and then choose **Delete**.
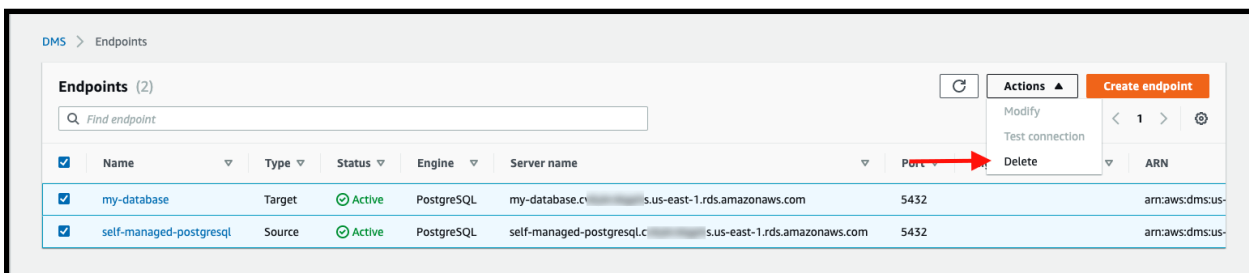

Then, go to the Replication instances section of the AWS DMS console. If your replication instance is not being used for any other replication tasks, choose it, and choose **Delete**.
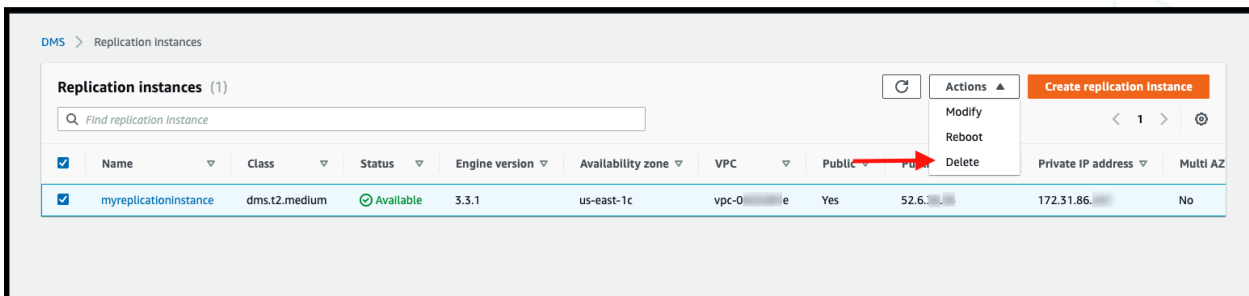
If you do not want to keep the target Amazon RDS database that you created for this lesson, you should delete that as well.

Navigate to the Amazon RDS console. Choose your database instance, and then choose **Delete** from the **Actions** dropdown.
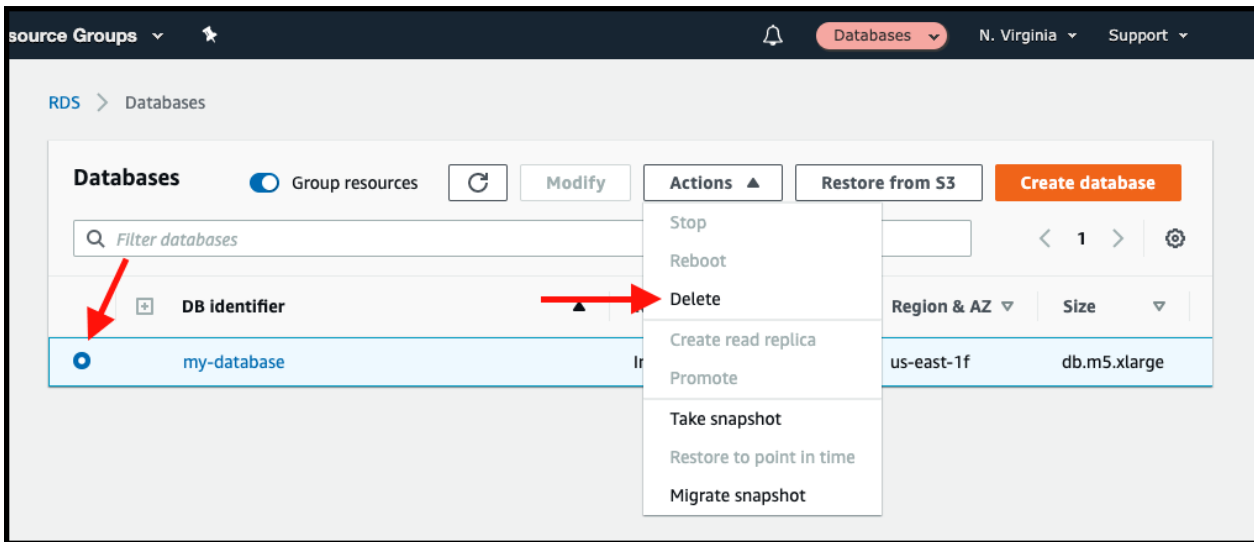
Confirm that you want to delete your database. You can choose not to create a final snapshot or retain backups because you used the database only for this walkthrough.

Delete my-database instance?   ✕

Are you sure you want to Delete the **my-database** DB Instance?

☐ Create final snapshot?
  Determines whether a final DB Snapshot is created before the DB instance is deleted.

☐ Retain automated backups
  Determines whether retaining automated backups for 7 days after deletion

☑ I acknowledge that upon instance deletion, automated backups, including system snapshots and point-in-time recovery, will no longer be available.

To confirm deletion, type *delete me* into the field

delete me ⬅

⚠ We strongly recommend taking a final snapshot before instance deletion since after your instance is deleted, automated backups will no longer be available.

Cancel   **Delete** ⬅

Finally, you might want to terminate your source database because it is no longer being used. If your source database is running on Amazon EC2, you can terminate the Amazon EC2 instance. If your source database is running elsewhere, follow the proper procedures to terminate it.

In this step, you learned how to migrate your application to use your new database. You also learned how to clean up AWS DMS resources when you are done using them.

In this lesson, you migrated an existing, self-managed PostgreSQL database to a fully managed PostgreSQL database in Amazon RDS by using AWS DMS. This migration enables you to free your developers to focus on the innovation that is core to your

business. By using AWS DMS, you can automate the delicate task of migrating data to a new database.