Migrate Database to AWS

1. Create an EC2 instance in AWS (We will consider this instance as on Premise instance for this Exercise)
2. Login to the instance
3. Install Mysql on the Instance

```
4. [ec2-user ~]$ sudo yum install -y mysql55-server php56-mysqlnd
```

To secure the MySQL server

The default installation of the MySQL server has several features that are great for testing and development, but they should be disabled or removed for production servers. The**mysql_secure_installation** command walks you through the process of setting a root password and removing the insecure features from your installation. Even if you are not planning on using the MySQL server, performing this procedure is a good idea.

1. Start the MySQL server.

```
[ec2-user ~]$ sudo service mysqld start
Initializing MySQL database:  Installing MySQL system tables...
OK
Filling help tables...
OK

To start mysqld at boot time you have to copy
support-files/mysql.server to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
...

Starting mysqld:                                           [  OK  ]
```

2. Run **mysql_secure_installation**.

```
[ec2-user ~]$ sudo mysql_secure_installation
```

   a. When prompted, enter a password for the `root` account.

        i. Enter the current `root` password. By default, the `root` account does not have a password set, so press **Enter**.
        ii. Type **Y** to set a password, and enter a secure password twice. For more information about creating a secure password, go tohttp://www.pctools.com/guides/password/. Make sure to store this password in a safe place.
    b. Type **Y** to remove the anonymous user accounts.
    c. Type **Y** to disable remote `root` login.
    d. Type **Y** to remove the test database.
    e. Type **Y** to reload the privilege tables and save your changes
3. If you want the MySQL server to start at every boot, enter the following command.

```
[ec2-user ~]$ sudo chkconfig mysqld on
```

Install phpMyAdmin

phpMyAdmin is a web-based database management tool that you can use to view and edit the MySQL databases on your EC2 instance. Follow the steps below to install and configure phpMyAdmin on your Amazon Linux instance.

1. Enable the Extra Packages for Enterprise Linux (EPEL) repository from the Fedora project on your instance.

```
[ec2-user ~]$ sudo yum-config-manager --enable epel
```

2. Install the `phpMyAdmin` package.

```
[ec2-user ~]$ sudo yum install -y phpMyAdmin
```

Note

Answer `y` to import the GPG key for the EPEL repository when prompted.

3. Configure your `phpMyAdmin` installation to allow access from your local machine. By default, `phpMyAdmin` only allows access from the server that it is running on, which is not very useful because Amazon Linux does not include a web browser.
    a. Find your local IP address by visiting a service such as whatismyip.com.

b. Edit the `/etc/httpd/conf.d/phpMyAdmin.conf` file and replace the server IP address (127.0.0.1) with your local IP address with the following command, replacing *`your_ip_address`* with the local IP address you identified in the previous step.

```
[ec2-user ~]$ sudo sed -i -e 's/127.0.0.1/your_ip_address/g'
/etc/httpd/conf.d/phpMyAdmin.conf
```

4. Restart the Apache web server to pick up the new configuration.

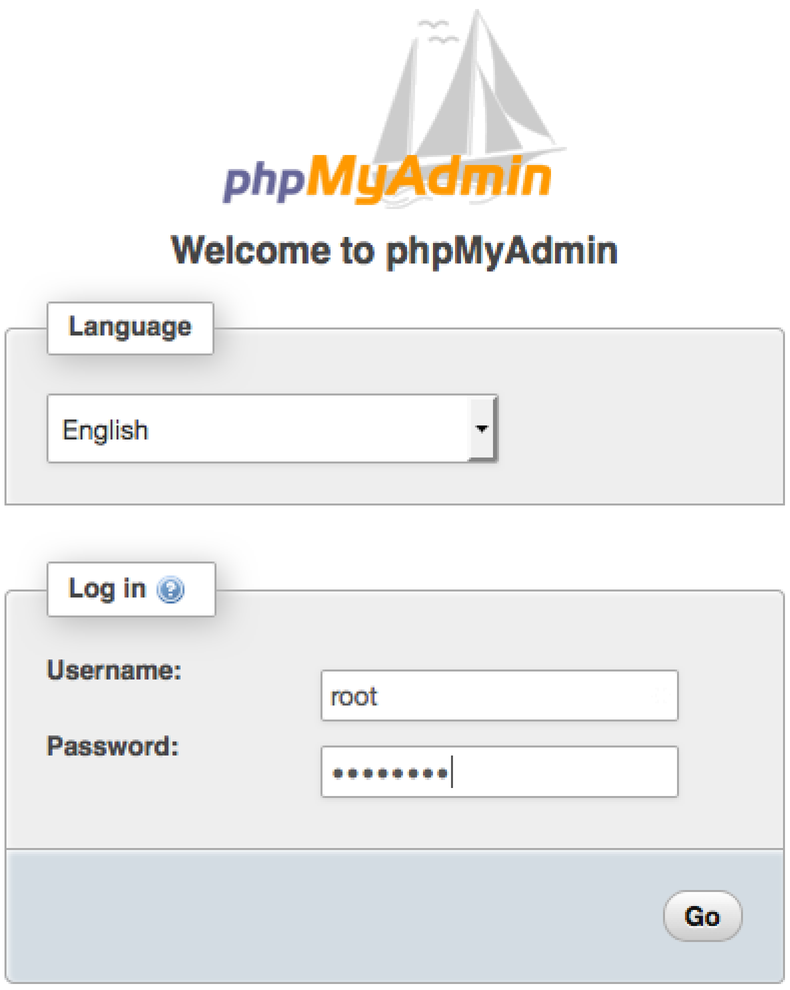
```
5. [ec2-user ~]$ sudo service httpd restart
6. Stopping httpd:                                  [  OK  ]
```

```
Starting httpd:                                  [  OK  ]
```

7. In a web browser, enter the URL of your `phpMyAdmin` installation. This URL is the public DNS address of your instance followed by a forward slash and phpmyadmin. For example:

```
http://my.public.dns.amazonaws.com/phpmyadmin
```

You should see the phpMyAdmin login page.



Note

If you get a `403 Forbidden` error, verify that you have set the correct IP address in the `/etc/httpd/conf.d/phpMyAdmin.conf` file. You can view the Apache access log to see what IP address the Apache server is actually getting your requests from with the following command:

```
[ec2-user ~]$ sudo tail -n 1 /var/log/httpd/access_log | awk '{ print $1 }'
205.251.233.48
```

Repeat Step 3.b, replacing the incorrect address that you previously entered with the address returned here; for example:

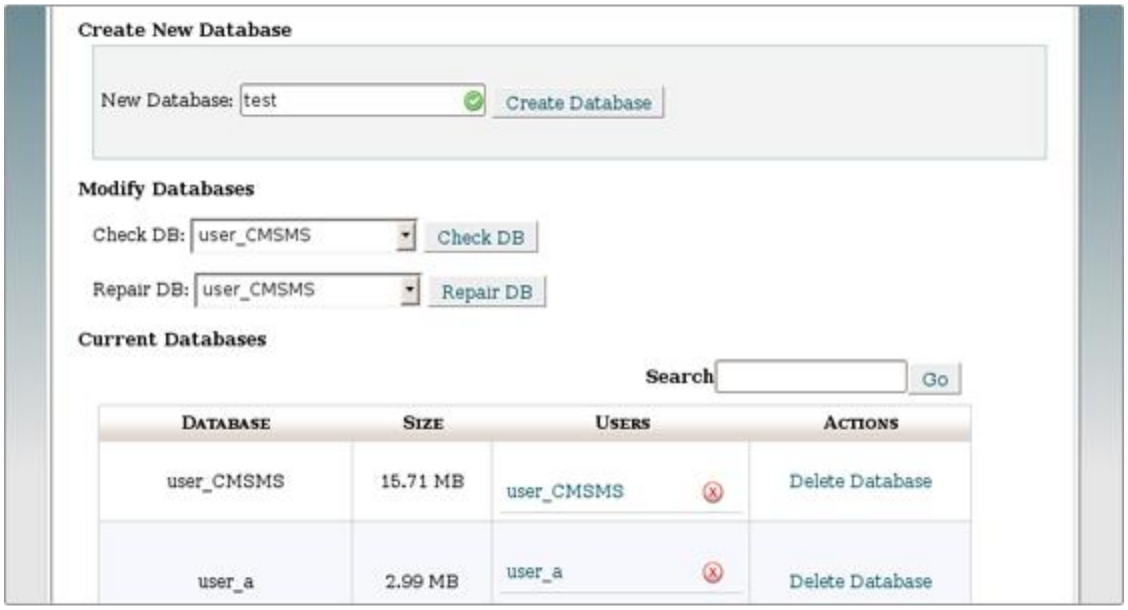
```
[ec2-user ~]$ sudo sed -i -e 's/previous_ip_address/205.251.233.48/g' /etc/httpd/conf.d/phpMyAdmin.conf
```

After you've replaced the IP address, restart the `httpd` service with Step 4.

8. Log into your `phpMyAdmin` installation with the `root` user name the MySQL root password you created earlier

9. How to Create a MySQL Database?

10. Please note that you can not create a database directly through cPanel->PhpMyAdmin due to the lack of user privileges. However, you can easily create a new database from your cPanel->MySQL Databases. Navigate to the Create New Database box. Enter the database name in the New Database text field and click on the Create Database button.



Create New Database

New Database: test

Modify Databases

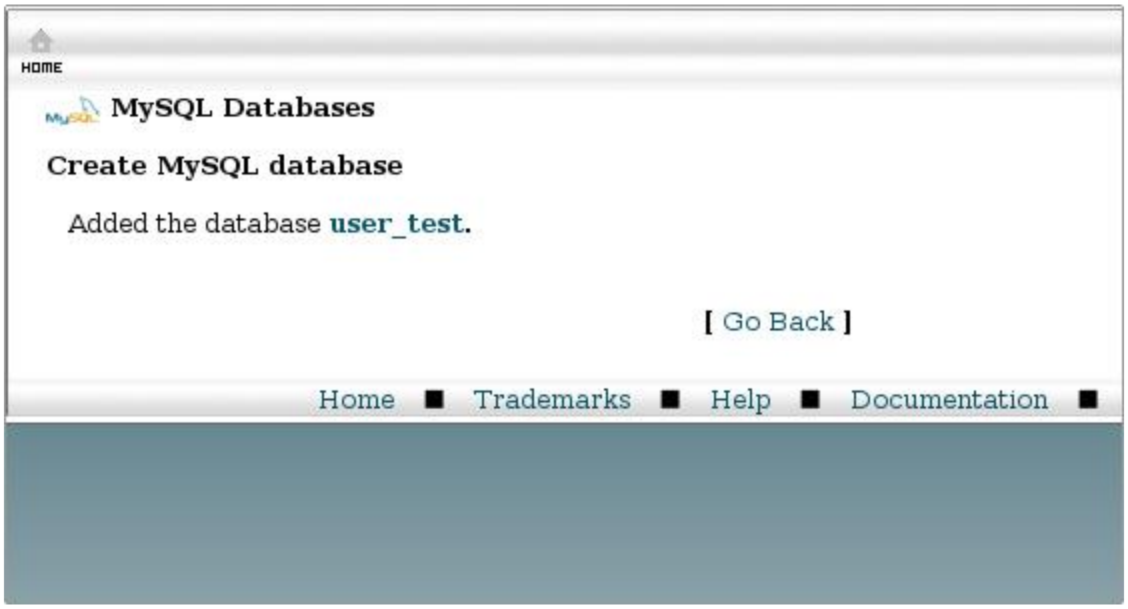
Check DB: user_CMSMS

Repair DB: user_CMSMS

Current Databases

Database	Size	Users	Actions
user_CMSMS	15.71 MB	user_CMSMS	Delete Database
user_a	2.99 MB	user_a	Delete Database

11. The database name will be preceded by the cPanel username. For example, if your cPanel user name is user and you want to have a database named test, the actual database name will be user_test. You will get a confirmation message.



HOME

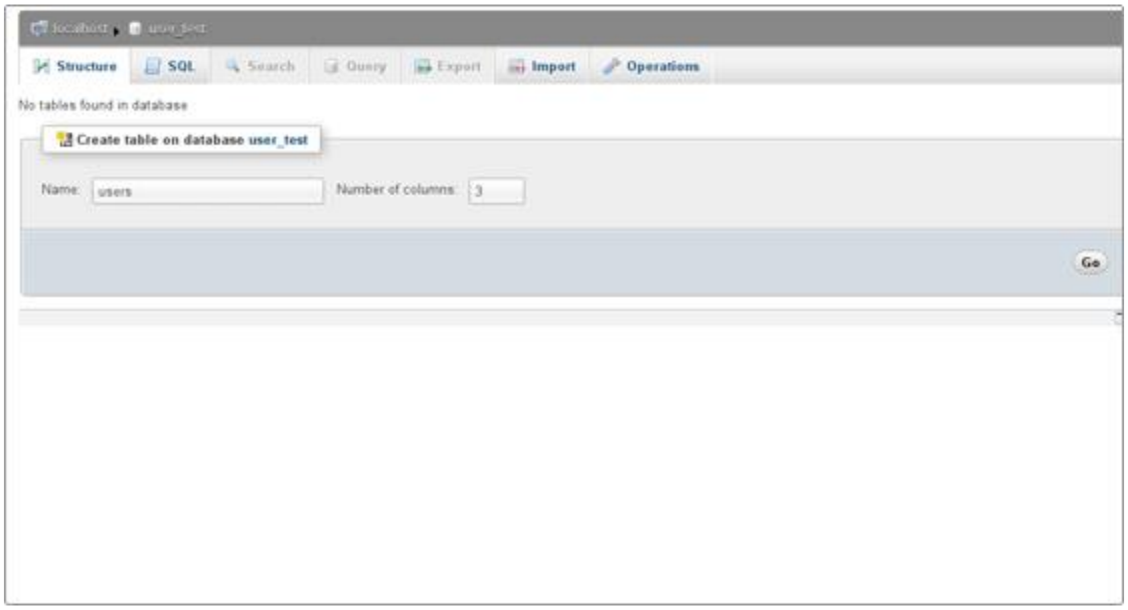
MySQL Databases

Create MySQL database

Added the database user_test.

12. How to Add MySQL Database Tables?

Navigate to your cPanel->PhpMyAdmin tool and open the newly create database. It is empty and there are no tables.



No tables found in database

Create table on database user_test

Name: users

Number of columns: 3

13. Enter the table name and the number of fields. Click on the Go button to create the table.
14. On the next screen you should enter the fields' names and the corresponding properties. The properties are:

Type

Here you should pick the type of the data, which will be stored in the corresponding field. More details about the possible choices can be found in the official MySQL Data Types documentation.

Length/Values

Here you should enter the length of the field. If the field type is "enum" or "set", enter the values using the following format: 'a','b','c'...

Collation

Pick the data collation for each of the fields.

Attributes

The possible attributes' choices are:

BINARY - the collation for the field will be binary, for example utf8_bin;

UNSIGNED - the field numeric values will be positive or 0;

UNSIGNED ZEROFILL - the field numeric values will be positive or 0 and leading zeros will be added to a number;

ON UPDATE CURRENT_TIMESTAMP - the value for a data type field has the current timestamp as its default value, and is automatically updated;

Null

Here you define whether the field value can be NULL. More about the NULL value can be found in the corresponding MySQL documentation.

Default

This property allows you to set the default value for the field.

Extra

In the Extra property you can define whether the field value is auto-increment.

The radio buttons that come below define whether there is an Index defined for the particular field and specify the Index type.

Comments

Here you can add comments, which will be included in the database sql code.

At the end you can include Table comments and pick the MySQL Storage Engine and the Collation.

Once you are ready, click on the Save button.

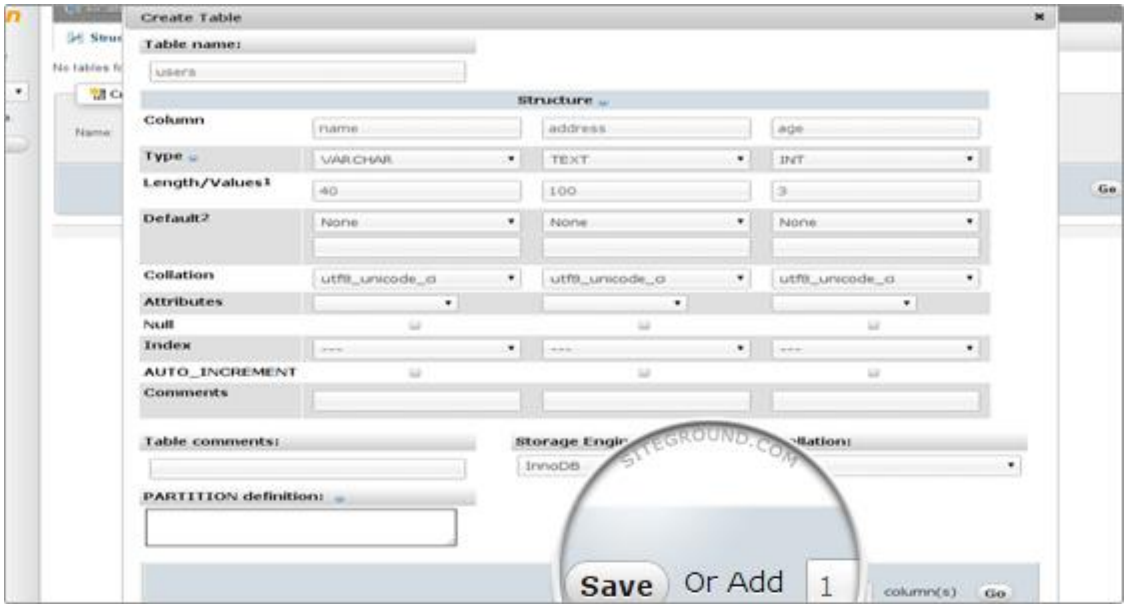

Create Table

Table name: users

Structure			
Column	name	address	age
Type	VARCHAR	TEXT	INT
Length/Values1	40	100	3
Default2	None	None	None
Collation	utf8_unicode_ci	utf8_unicode_ci	utf8_unicode_ci
Attributes			
Null			
Index	---	---	---
AUTO_INCREMENT			
Comments			

Table comments:

Storage Engine: InnoDB

PARTITION definition:

15. If you want to add more fields you should specify their number and click on the Go button instead of Save.
16. The database table will be created and you will see the corresponding MySQL query.

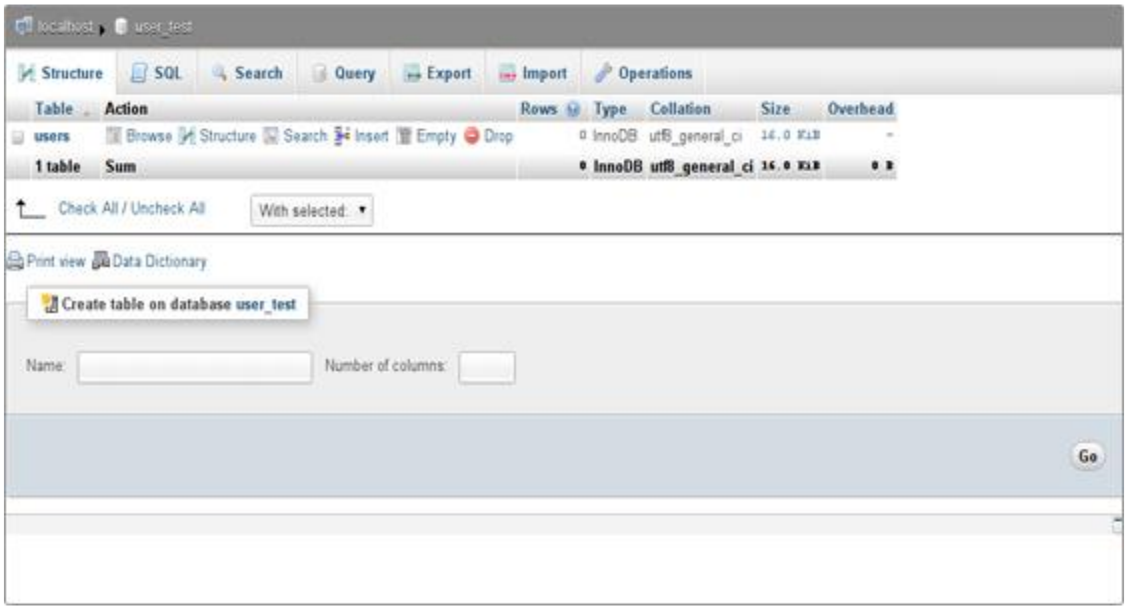
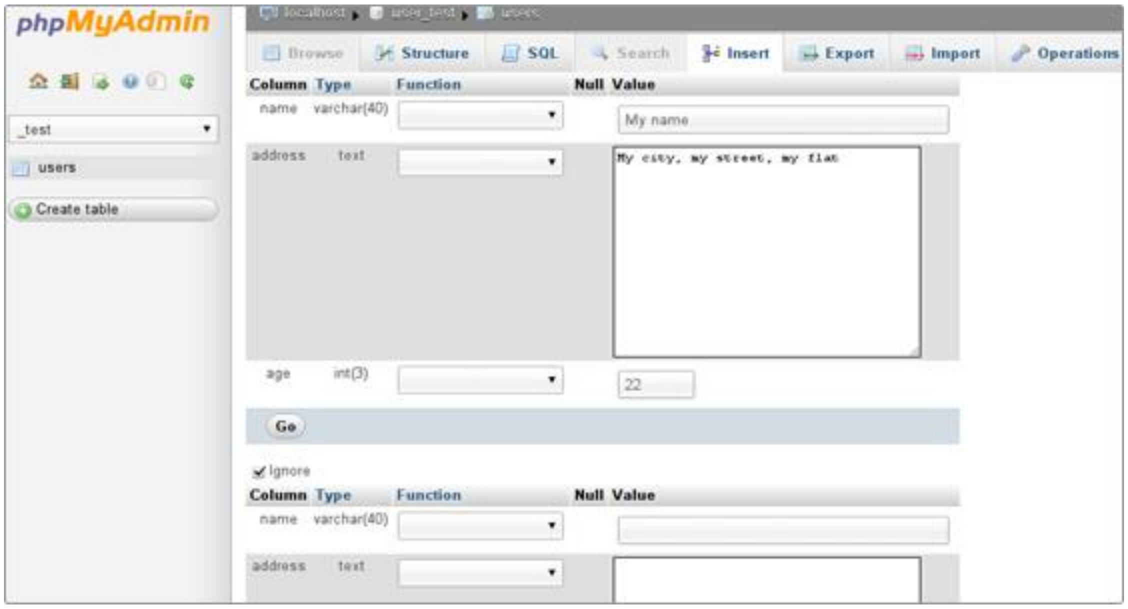


Table	Action	Rows	Type	Collation	Size	Overhead
users	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8_general_ci	16.0 KiB	-
1 table	Sum	0	InnoDB	utf8_general_ci	16.0 KiB	0 B

17. Now we will proceed with the populating of the table with data.

18. How to Add Content in a Database Table?

19. In order to add records in a database table click on the Insert tab.

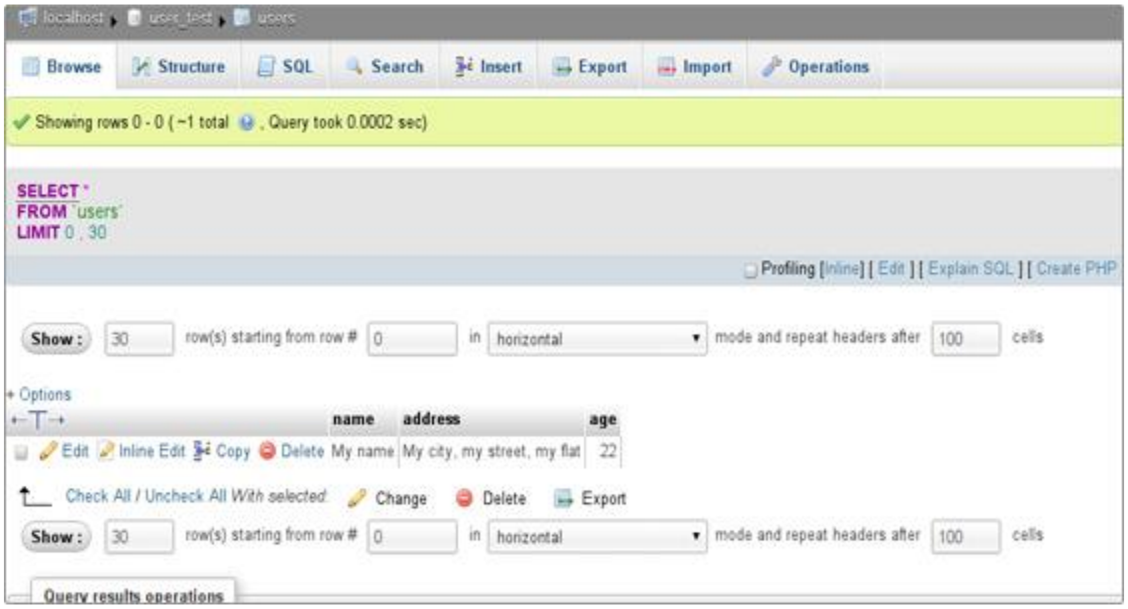


Column	Type	Function	Null	Value
name	varchar(40)			My name
address	text			My city, my street, my flat
age	int(3)			22

✓ Ignore

Column	Type	Function	Null	Value
name	varchar(40)			
address	text			

20. Enter the data in the corresponding fields and click on the Go button to store it.
21. At the bottom of the page you will see a drop-down menu labelled Restart insertion with x rows . There you can pick the number of the rows that you can populate with data and insert at once. By default the value is 2.
22. The Ignore check box will allow you to ignore the data entered below it. It will not be added.
23. You can see the newly inserted record by clicking on the Browse tab.


localhost ▸ user_test ▸ users
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Profiling [Inline] [Edit] [Explain SQL] [Create PHP
+ Options
Check All / Uncheck All With selected:
Change
Delete
Export
Query results operations

✓ Showing rows 0 - 0 (~1 total , Query took 0.0002 sec)

```
SELECT *
FROM `users`
LIMIT 0 , 30
```

Show : 30 row(s) starting from row # 0 in horizontal mode and repeat headers after 100 cells

	name	address	age
Edit Inline Edit Copy Delete	My name	My city, my street, my flat	22

Show : 30 row(s) starting from row # 0 in horizontal mode and repeat headers after 100 cells

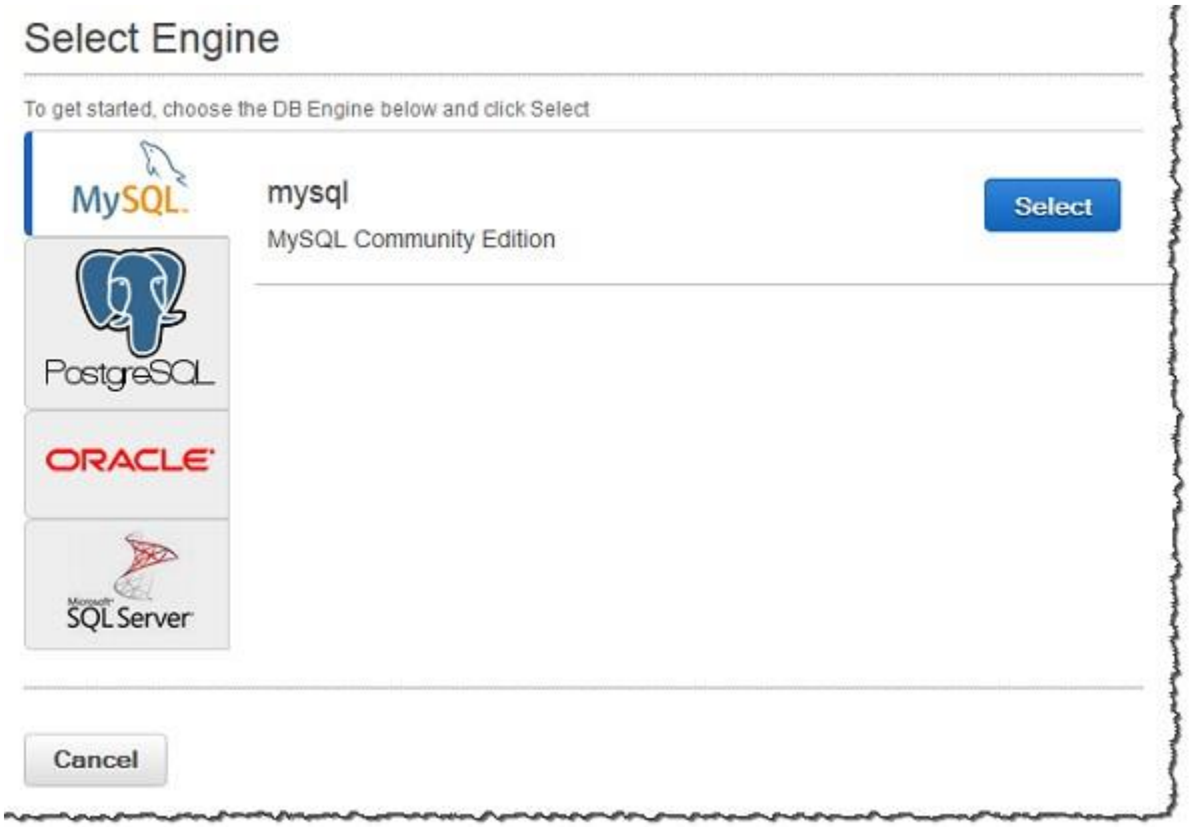
24. You can edit or delete the record by clicking on the corresponding icons.
25. To insert more records, return to the Insert tab and repeat the procedure.

Create a Destination Database in AWS RDS

To launch a MySQL DB instance

1. login to the AWS Management Console and open the Amazon RDS console athttps://console.aws.amazon.com/rds/.
2. In the top right corner of the AWS Management Console, select the region in which you want to create the DB instance.
3. In the navigation pane, click **Instances**.
4. Click **Launch DB Instance** to start the **Launch DB Instance Wizard**.

   The wizard opens on the **Select Engine** page.



Select Engine

To get started, choose the DB Engine below and click Select

mysql

MySQL Community Edition

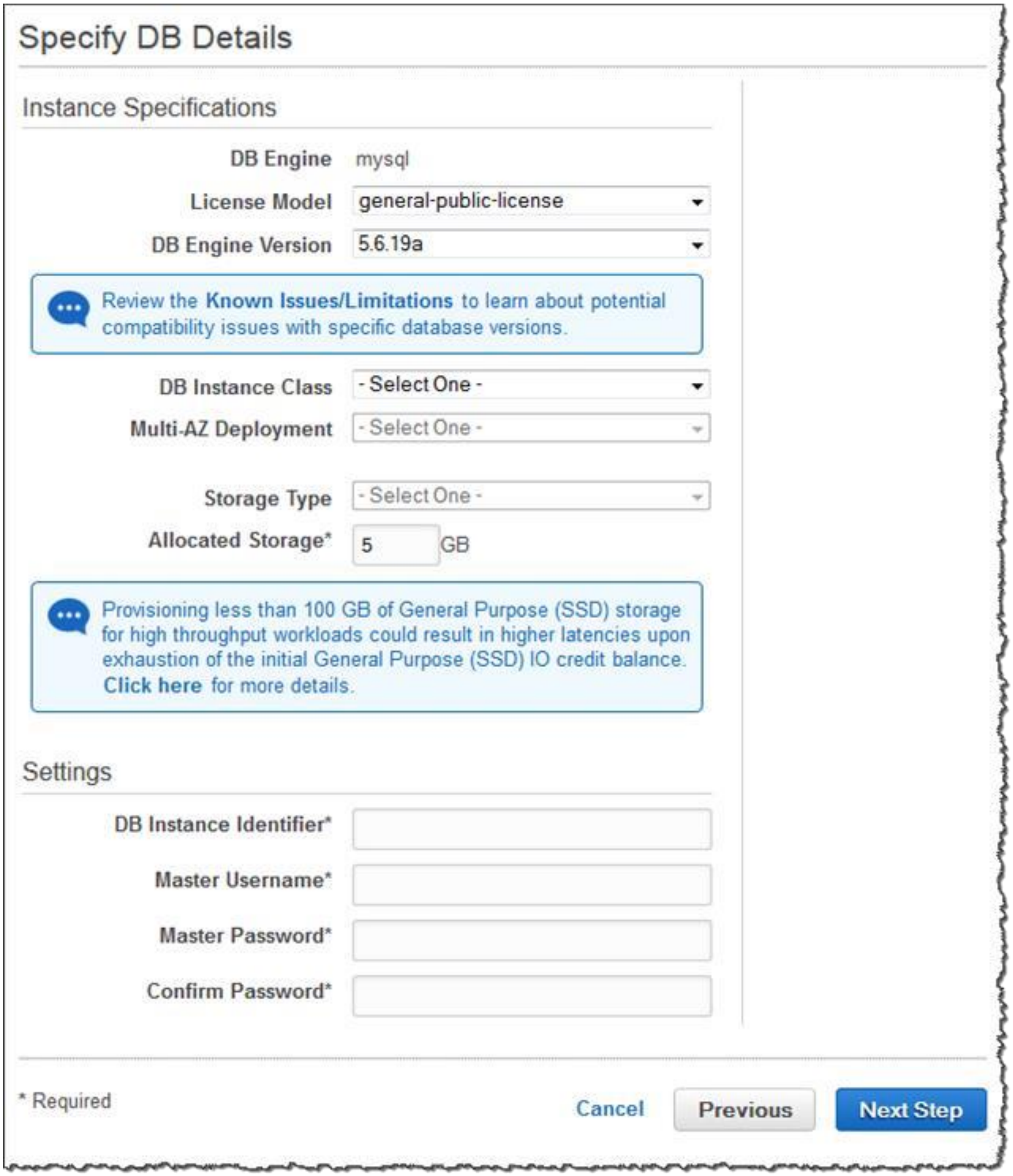
5. In the **Launch DB Instance Wizard** window, click the **Select** button for the MySQL DB engine.
6. The next step asks if you are planning to use the DB instance you are creating for production. If you are, select **Yes**. By selecting **Yes**, the failover option **Multi-AZ** and the **Provisioned IOPS** storage option will be preselected in the following step.

Click **Next** when you are finished.

7. On the **Specify DB Details** page, specify your DB instance information. The following table shows settings for an example DB instance. Click **Next** when you are finished.

For this parameter...	...Do this:
License Model	MySQL has only one license model. Select the default, `General-Public-License`, to use the general license agreement for MySQL.
DB Engine Version	Select the version of MySQL that you want to work with. Note that Amazon RDS supports several versions of MySQL.
DB Instance Class	Select a DB instance class that defines the processing and memory requirements for the DB instance. Select t2.micro
Multi-AZ Deployment	Unselect this Option
Allocated Storage	Type a value to allocate storage for your database (in gigabytes
Storage Type	Select the storage type you want to use. Select Generic ssd
DB Instance Identifier	Type a name for the DB instance that is unique for your account in the region you selected. for example `mysql-instance1`.
Master Username	Type a name using alphanumeric characters that you will use as the master user name to log on to your DB instance. The default privileges granted to the master user name account include: create, drop, references, event, alter, delete, index, insert, select, update, create temporary tables, lock tables, trigger, create view, show view, alter routine, create routine, execute, create user, process, show databases, grant option.

Master Password	Type a password that contains from 8 to 16 printable ASCII characters (excluding /,", and @) for your master user password.
Confirm Password	Re-type the Master Password for confirmation.



Specify DB Details

Instance Specifications

DB Engine	mysql
License Model	general-public-license
DB Engine Version	5.6.19a

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

DB Instance Class	- Select One -
Multi-AZ Deployment	- Select One -
Storage Type	- Select One -
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

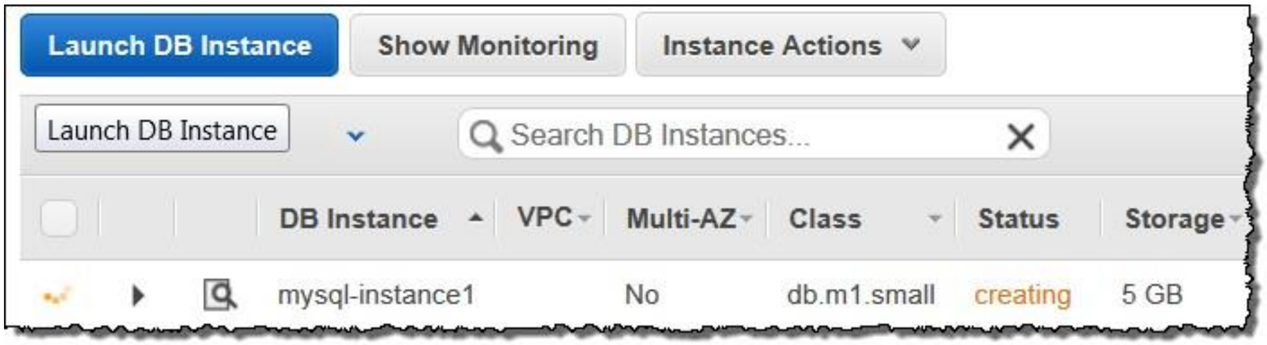
DB Instance Identifier*	
Master Username*	
Master Password*	
Confirm Password*	

* Required

8. Click **Launch DB Instance** to create your MySQL DB instance.
9. On the final page of the wizard, click **Close**.

10. On the Amazon RDS console, the new DB instance appears in the list of DB instances. The DB instance will have a status of **creating** until the DB instance is created and ready

for use. When the state changes to **available**, you can connect to the DB instance. Depending on the DB instance class and store allocated, it could take several minutes for the new instance to be available.



DB Instance	VPC	Multi-AZ	Class	Status	Storage
mysql-instance1		No	db.m1.small	creating	5 GB

Migrate Source Database to Destination Database

The simplest way to import data from an existing MySQL or MariaDB database to an Amazon RDS MySQL or MariaDB DB instance is to copy the database with mysqldump and pipe it directly into the Amazon RDS MySQL or MariaDB DB instance.

```
sudo mysqldump -u <local_user> \
    --databases world \
    --single-transaction \
    --compress \
    --order-by-primary  \
    -p <local_password> | mysql -u <RDS_user_name> \
        --port=3306 \
        --host=hostname \
        -p <RDS_password>
```

Start a Database Migration with AWS Database Migration Service

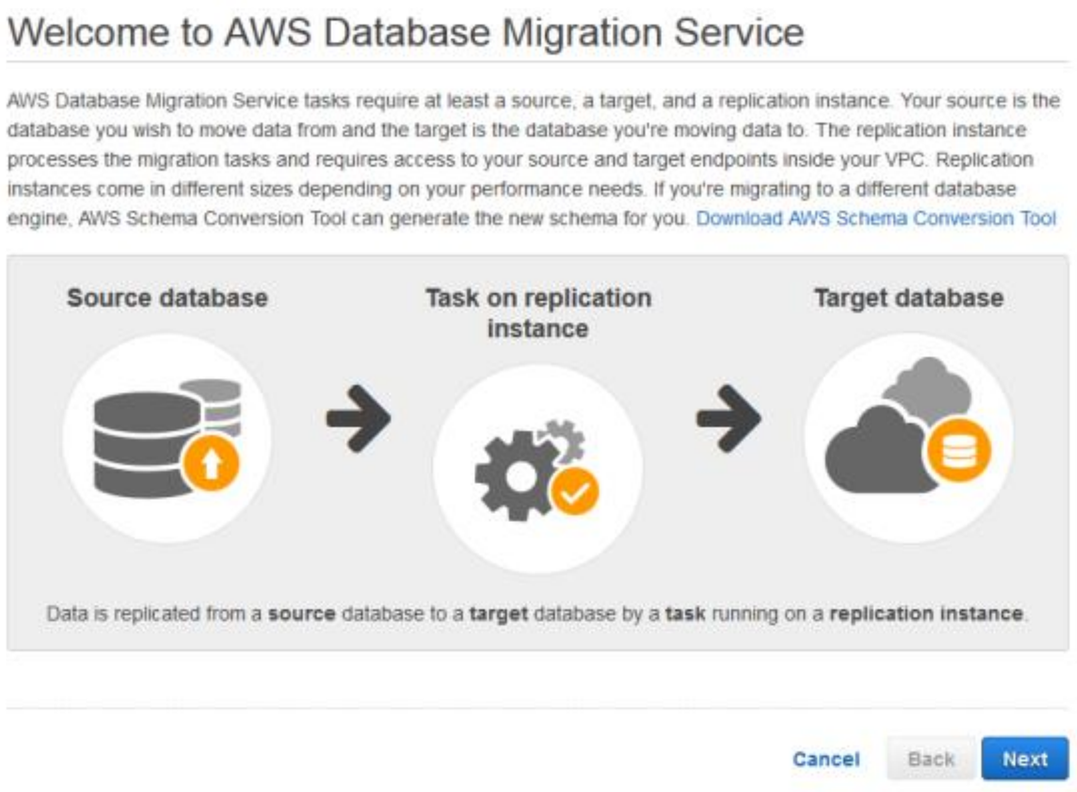
On the **Dashboard** page in the AWS DMS console, you can use a wizard to help create your first data migration. Following the wizard process, you allocate a replication instance that performs all the processes for the migration, specify a source and a target database, and then create a task or set of tasks to define what tables and replication processes you want to use. AWS DMS then creates your replication instance and performs the tasks on the data being migrated.

To start an AWS DMS database migration by using the console

1. Sign in to the AWS Management Console and choose AWS DMS. Note that 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. On the **Dashboard** page, choose **Create Migration**.

Step 1: Welcome

You start migration from the console's Welcome page, shown following, which explains the process of database migration using AWS DMS.


Welcome to AWS Database Migration Service

AWS Database Migration Service tasks require at least a source, a target, and a replication instance. Your source is the database you wish to move data from and the target is the database you're moving data to. The replication instance processes the migration tasks and requires access to your source and target endpoints inside your VPC. Replication instances come in different sizes depending on your performance needs. If you're migrating to a different database engine, AWS Schema Conversion Tool can generate the new schema for you. Download AWS Schema Conversion Tool

Data is replicated from a **source** database to a **target** database by a **task** running on a **replication instance**.

To start a database migration from the console's Welcome page

- Choose **Next**.

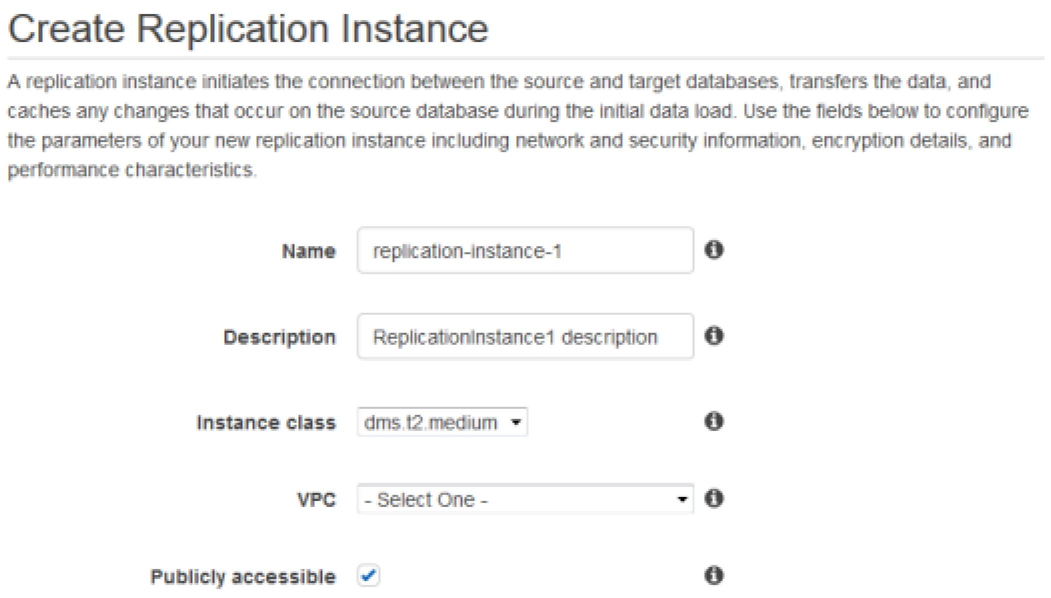
Step 2: Create a Replication Instance

Your first task in migrating a database is to create a replication instance that has sufficient storage and processing power to perform the tasks you assign and migrate data from your source database to the target database. The required size of this instance varies depending on the amount of data you need to migrate and the tasks that you need the instance to perform. For more information about replication instances, see Replication Instances for AWS Database Migration Service.

The procedure following assumes that you have chosen AWS DMS from the AWS Management Console and started the replication task wizard.

To create a replication instance by using the AWS console

1. On the **Create replication instance** page, specify your replication instance information. The following table describes the settings.



Create Replication Instance

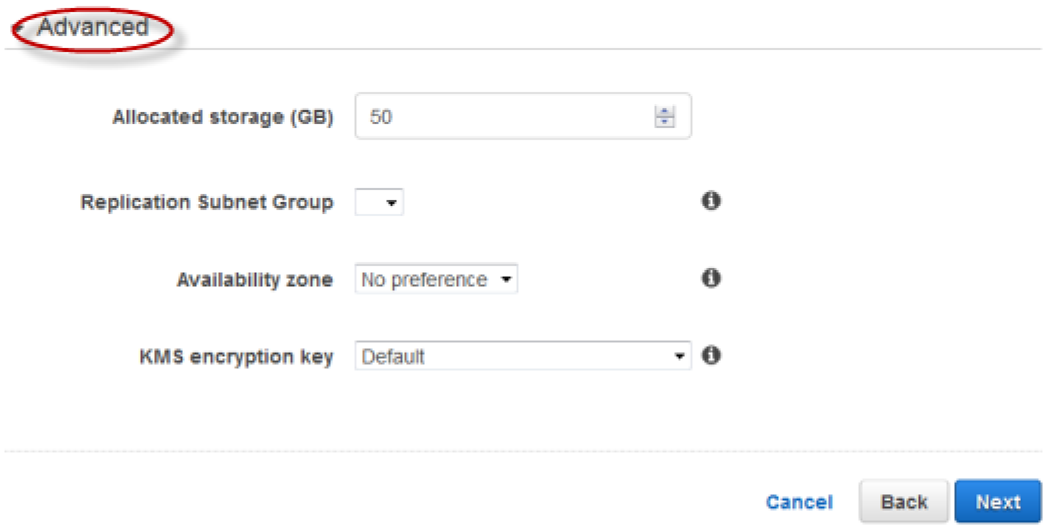
A replication instance initiates the connection between the source and target databases, transfers the data, and caches any changes that occur on the source database during the initial data load. Use the fields below to configure the parameters of your new replication instance including network and security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

Name	replication-instance-1
Description	ReplicationInstance1 description
Instance class	dms.t2.medium
VPC	- Select One -
Publicly accessible	✓

For This Option	Do This
Name	Type a name for the replication instance that contains from 8 to 16 printable ASCII characters (excluding /,", and @). The name should be unique for your account for the region you selected. You can choose to add some intelligence to the name, such as including the region and task you are performing, for example `west2-mysql2mysql-instance1`.
Description	Type a brief description of the replication instance.
Instance class	Choose an instance class with the configuration you need for your migration. Keep in mind that the instance must have enough storage, network, and processing power to successfully complete your migration. For more information on how to determine which instance class is best for your migration, see Replication Instances for AWS Database Migration Service.
VPC	Choose the Amazon Virtual Private Cloud (Amazon VPC) you want to use. If your source or your target database is in an VPC, choose that VPC. If your source and your target databases are in different VPCs,

For This Option	Do This
	ensure that they are both in public subnets and are publicly accessible, and then choose the VPC where the replication instance is located. The replication instance must be able to access the data in the source VPC. If neither your source nor your target database is in a VPC, leave this option blank.
Publicly accessible	Choose this option if you want the replication instance to be accessible from the Internet.

2. Choose the **Advanced** tab, shown following, to set values for network and encryption settings if you need them.


Allocated storage (GB) 50

Replication Subnet Group

Availability zone No preference

KMS encryption key Default

3. Specify the additional settings. The following table describes the settings.

For This Option	Do This
Allocated storage (GB)	Choose the amount of storage, in gigabytes, to allocate for the replication instance.
Replication Subnet Group	Choose the replication subnet group in your selected VPC where you want the replication instance to be created. If your source database is in a VPC, choose the subnet group that contains the source database as

For This Option	Do This
	the location for your replication instance. For more information about replication subnet groups, see Creating a Replication Subnet Group.
Availability zone	Choose the Availability Zone where your source database is located.
KMS encryption key	Choose the encryption key to use to encrypt replication storage and connection information. If you choose **None**, the default AWS Key Management Service (AWS KMS) key associated with your account and region is used. For more information on using the encryption key, seeSetting an Encryption Key for AWS Database Migration Service.

4. Choose **Next**.

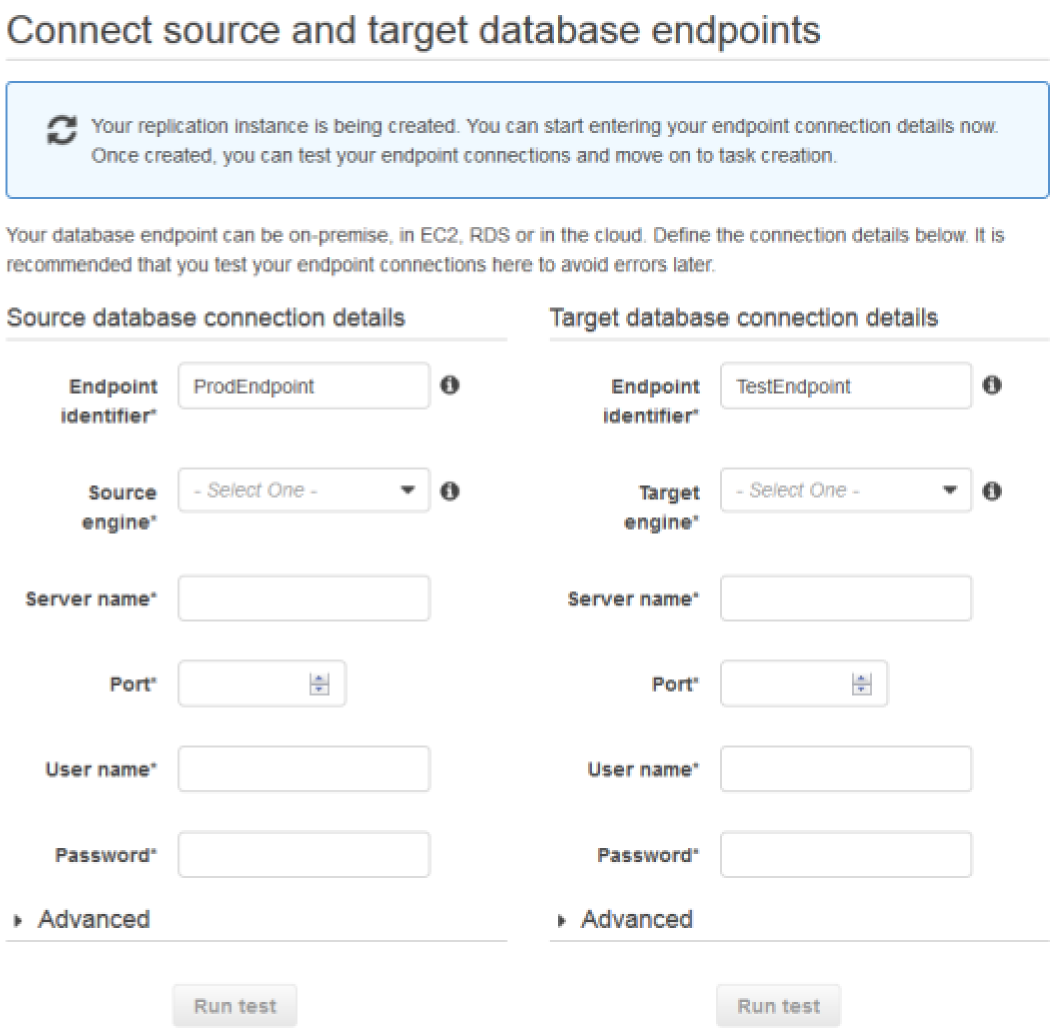
Step 3: Specify Database Endpoints

While your replication instance is being created, you can specify the source and target databases. The source and target databases can be on an Amazon Elastic Compute Cloud (Amazon EC2) instance, an Amazon Relational Database Service (Amazon RDS) DB instance, or an on-premises database.

The procedure following assumes that you have chosen AWS DMS from the AWS Management Console and specified your replication instance information in the replication task wizard.

To specify source and target database endpoints using the AWS console

1. On the **Connect source and target database endpoints** page, specify your connection information for the source and target databases. The following table describes the settings.



Connect source and target database endpoints

Your replication instance is being created. You can start entering your endpoint connection details now. Once created, you can test your endpoint connections and move on to task creation.

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*	ProdEndpoint
Source engine*	- Select One -
Server name*	
Port*	
User name*	
Password*	

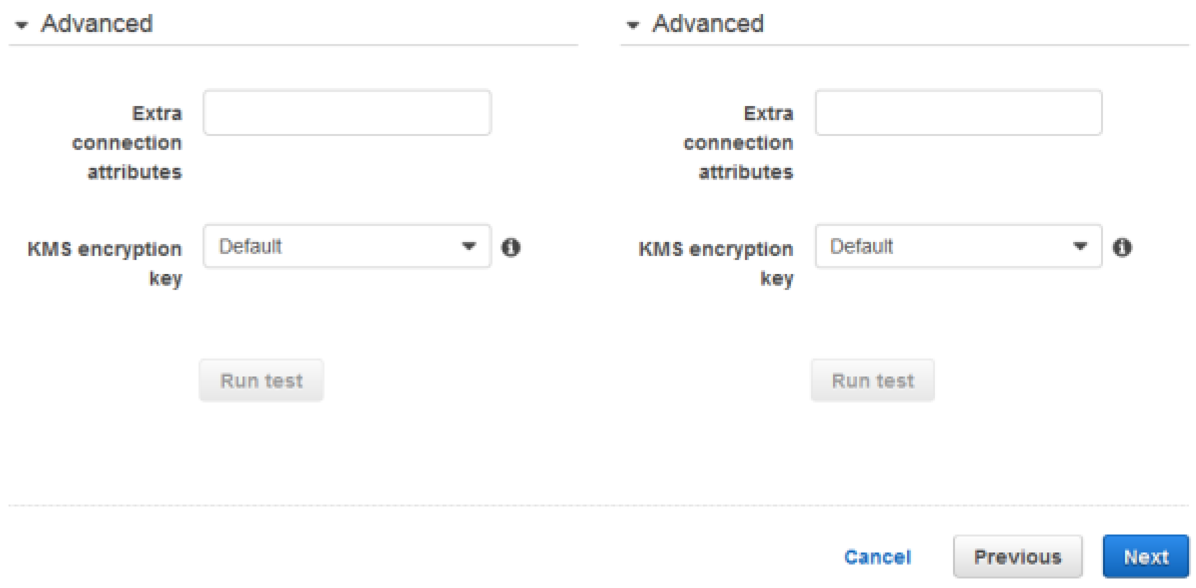
Target database connection details

Endpoint identifier*	TestEndpoint
Target engine*	- Select One -
Server name*	
Port*	
User name*	
Password*	

For This Option	Do This
Endpoint Identifier	Type the name you want to use to identify the endpoint. You might want to include in the name the type of endpoint, such as `oracle-source` or`PostgreSQL-target`. The name must be unique for all replication instances.
Source Engine or**Target Engine**	Choose the type of database that is the endpoint.
Server address	Type the server address.
Port	Type the port used by the database.

For This Option	Do This
User name	Type the user name with the permissions required to allow data migration. For information on the permissions required, see the security section for the source database in this user guide.
Password	Type the password for the account with the required permissions.
Database name/SID	Type the name of the database to use.

2. Choose the **Advanced** tab, shown following, to set values for connection string and encryption key if you need them. You can test the endpoint connection by choosing**Run test**.



For This Option	Do This
Extra connection attributes	Type any additional connection parameters here. For more information about extra connection attributes, see Using Extra Connection Attributes with AWS Database Migration Service.
KMS encryption key	Choose the encryption key to use to encrypt replication storage and connection information. If you choose **None**, the default AWS Key Management Service (AWS KMS) key associated with your account

For This Option	Do This
	and region is used. For more information on using the encryption key, seeSetting an Encryption Key for AWS Database Migration Service.

Step 4: Create a Task

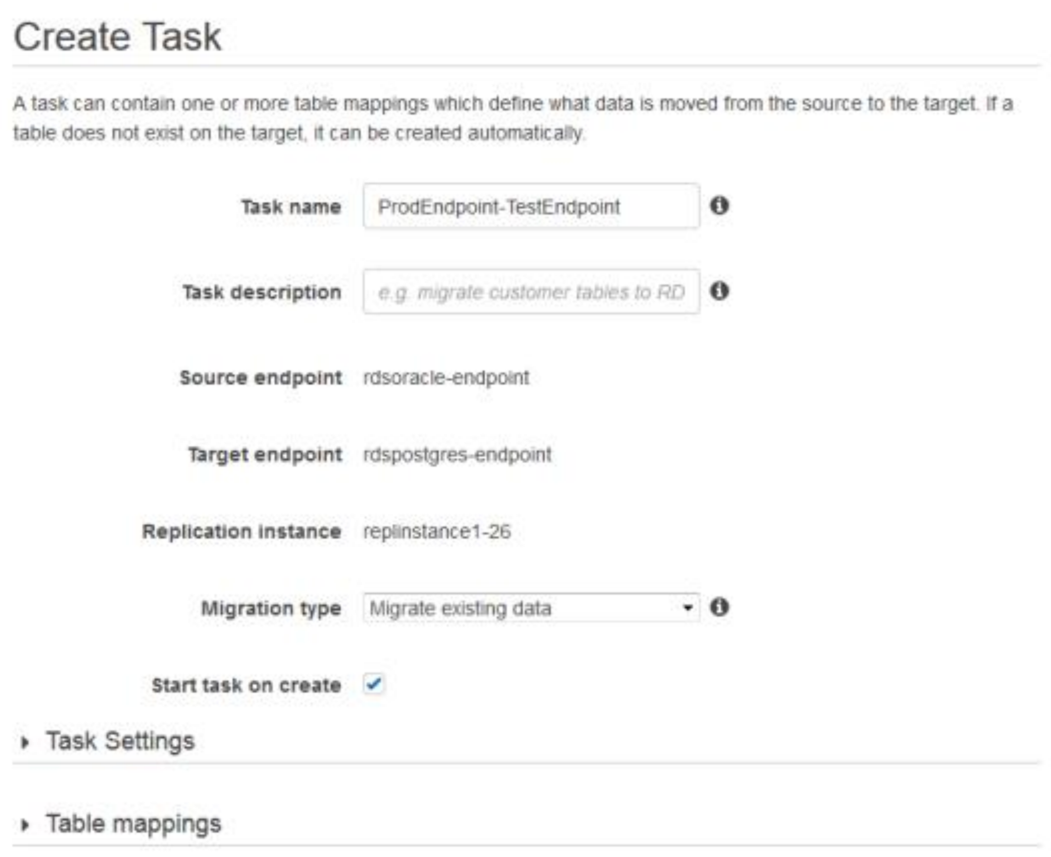
Create a task to specify what tables to migrate, to map data using a target schema, and to create new tables on the target database. As part of creating a task, you can choose the type of migration: to migrate existing data, migrate existing data and replicate ongoing changes, or replicate data changes only.

Using AWS DMS, you can specify precise mapping of your data between the source and the target database. Before you specify your mapping, make sure you review the documentation section on data type mapping for your source and your target database.

The procedure following assumes that you have chosen AWS DMS from the AWS Management Console and specified replication instance information and endpoints in the replication task wizard.

To create a task

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



Create Task

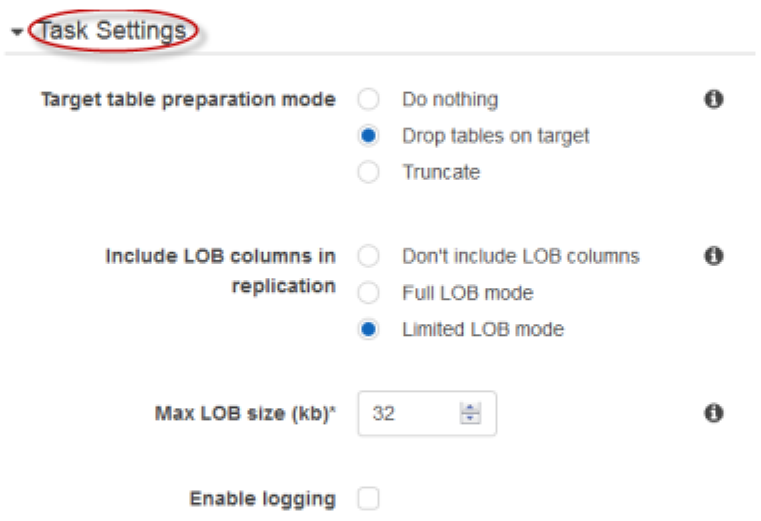
A task can contain one or more table mappings which define what data is moved from the source to the target. If a table does not exist on the target, it can be created automatically.

Task name	ProdEndpoint-TestEndpoint
Task description	e.g. migrate customer tables to RD
Source endpoint	rdsoracle-endpoint
Target endpoint	rdspostgres-endpoint
Replication instance	replinstance1-26
Migration type	Migrate existing data
Start task on create	✓

For This Option	Do This
Task name	Type a name for the task.
Task description	Type a description for the task.
Source endpoint	Shows the source endpoint that will be used.
Target endpoint	Shows the target endpoint that will be used.
Replication instance	Shows the replication instance that will be used.
Migration type	Choose the migration method you want to use. You can choose to have just the existing data migrated to the target database or have ongoing changes sent to the target database in addition to the migrated data.

For This Option	Do This
Start task on create	When this option is selected, the task begins as soon as it is created.

2. Choose the **Task Settings** tab, shown following, and specify values for your target table, LOB support, and to enable logging. The task settings shown depend on the**Migration type** value you select. For example, when you select **Migrate existing data**, the following options are shown:



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]

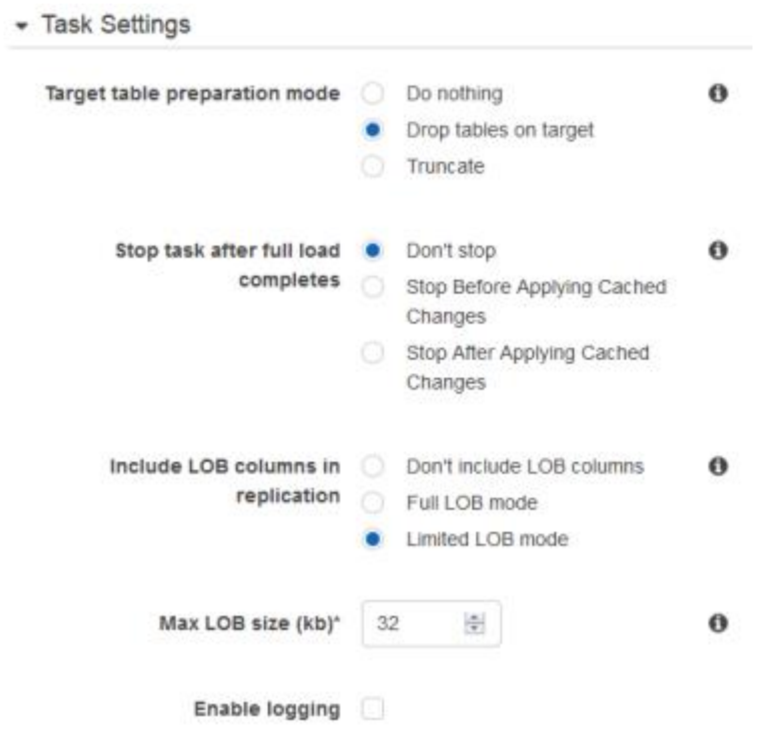
Max LOB size (kb)*: 32

Enable logging: ☐

For This Option	Do This
Target table preparation mode	**Do nothing** - Data and metadata of the target tables are not changed. **Drop tables on target** - The tables are dropped and new tables are created in their place. **Truncate** - Tables are truncated without affecting table metadata.
Include LOB columns in replication	**Don't include LOB columns** - LOB columns will be excluded from the migration.

For This Option	Do This
	Full LOB mode - Migrate complete LOBs regardless of size. LOBs are migrated piecewise in chunks controlled by the LOB chunk size. This method is slower than using Limited LOB Mode. **Limited LOB mode** - Truncate LOBs to 'Max LOB Size' This method is faster than using Full LOB Mode. For more information about LOB support in AWS DMS, see AWS Database Migration Service LOB Support
Max LOB size (kb)	In **Limited LOB Mode**, LOB columns which exceed the setting of **Max LOB Size** will be truncated to the specified Max LOB Size.
Enable logging	Enables logging by Amazon CloudWatch.

When you select **Migrate existing data and replicate** for **Migration type**, the following options are shown:


Task Settings

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

Stop task after full load completes: [✓ Don't stop] Stop Before Applying Cached Changes / Stop After Applying Cached Changes

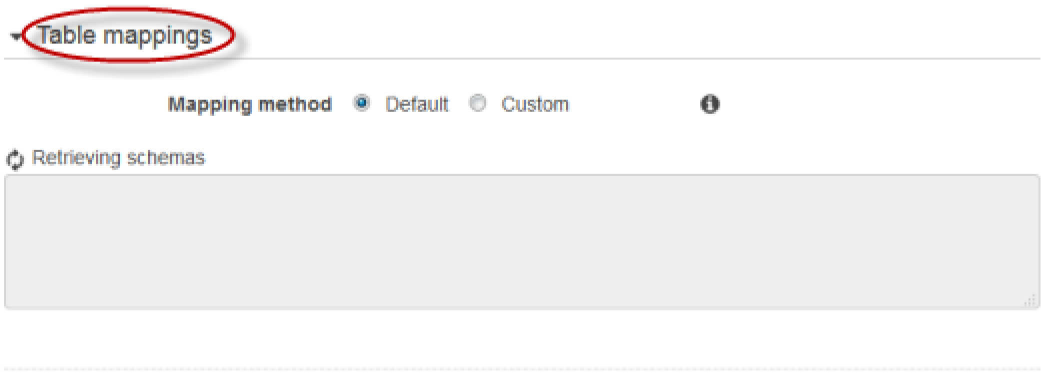
Include LOB columns in replication: Don't include LOB columns / Full LOB mode / [✓ Limited LOB mode]

Max LOB size (kb)*: 32

Enable logging: ☐

For This Option	Do This
Target table preparation mode	**Do nothing** - Data and metadata of the target tables are not changed. **Drop tables on target** - The tables are dropped and new tables are created in their place. **Truncate** - Tables are truncated without affecting table metadata.
Stop task after full load completes	**Don't stop** - Do not stop the task, immediately apply cached changes and continue on. **Stop before applying cached changes** - Stop the task prior to the application of cached changes. This will allow you to add secondary indexes which may speed the application of changes. **Stop after applying cached changes** - Stop the task after cached changes have been applied. This will allow you to add foreign keys, triggers etc. if you are using Transactional Apply.
Include LOB columns in replication	**Don't include LOB columns** - LOB columns will be excluded from the migration. **Full LOB mode** - Migrate complete LOBs regardless of size. LOBs are migrated piecewise in chunks controlled by the LOB chunk size. This method is slower than using Limited LOB Mode. **Limited LOB mode** - Truncate LOBs to 'Max LOB Size' This method is faster than using Full LOB Mode. For more information about LOB support in AWS DMS, see AWS Database Migration Service LOB Support
Max LOB size (kb)	In **Limited LOB Mode**, LOB columns which exceed the setting of **Max LOB Size** will be truncated to the specified Max LOB Size.
Enable logging	Enables logging by Amazon CloudWatch.

3. Choose the **Table mappings** tab, shown following, to set values for schema mapping and the mapping method. If you choose **Custom**, you can specify the target schema and table values. For more information about table mapping, see Using Table Mapping with an AWS Database Migration Service Task.

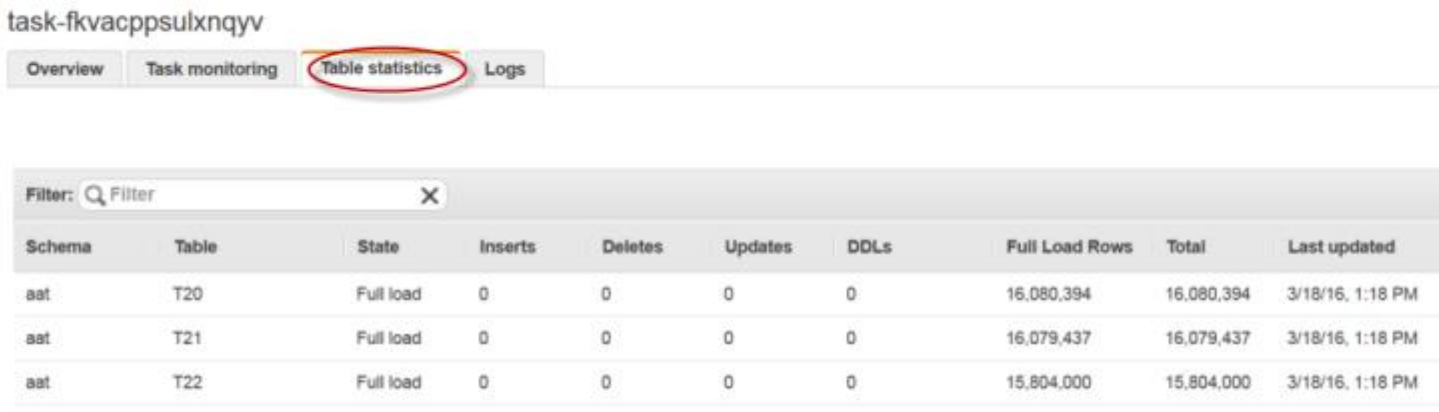


Mapping method [✓ Default] Custom

4. Once you have finished with the task settings, choose **Create task**.

Monitor Your Task

If you select **Start task on create** when you create a task, your task begins immediately to migrate your data when you choose **Create task**. You can view statistics and monitoring information for your task by choosing the running task from the AWS Management Console. The following screenshot shows the table statistics of a database migration.



Schema	Table	State	Inserts	Deletes	Updates	DDLs	Full Load Rows	Total	Last updated
aat	T20	Full load	0	0	0	0	16,080,394	16,080,394	3/18/16, 1:18 PM
aat	T21	Full load	0	0	0	0	16,079,437	16,079,437	3/18/16, 1:18 PM
aat	T22	Full load	0	0	0	0	15,804,000	15,804,000	3/18/16, 1:18 PM