



Deploying Amazon RDS Multi-AZ and Read Replica, Simulate Failover



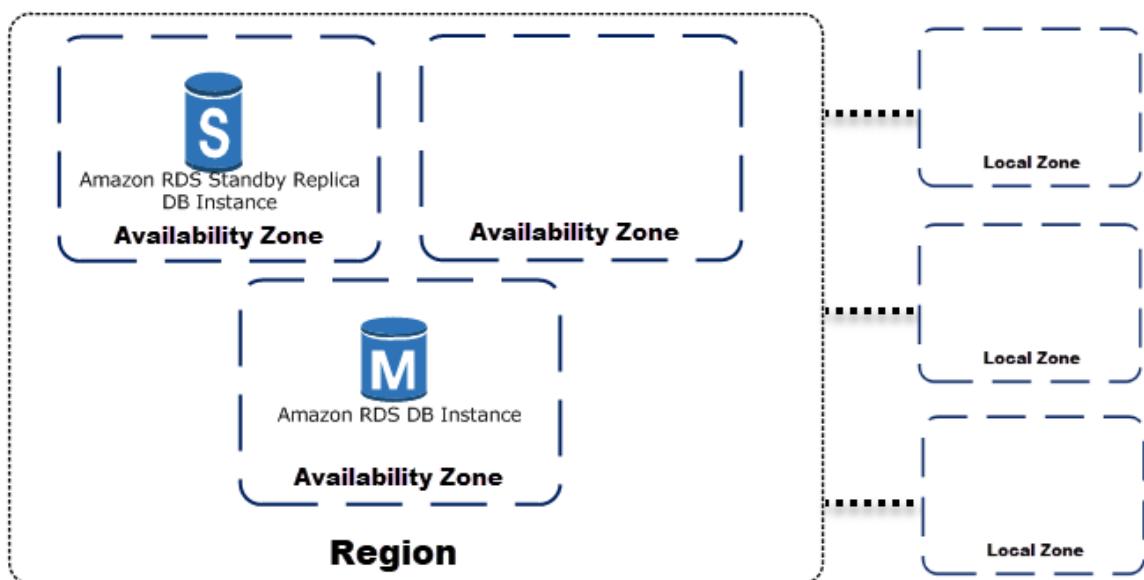
Lets'sUpgrade AWS Advance Program
Help by Mrs. Vinolin Jermiah

Overview of Project Summary:

High availability (Multi-AZ) for Amazon RDS

Amazon RDS provides high availability and failover support for DB instances using Multi-AZ deployments. Amazon RDS uses several different technologies to provide failover support. Multi-AZ deployments for Maria DB, MySQL, Oracle, and PostgreSQL DB instances use Amazon's failover technology. SQL Server DB instances use SQL Server Database Mirroring (DBM) or Always on Availability Groups (AGs).

In a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone. The primary DB instance is synchronously replicated across Availability Zones to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups. Running a DB instance with high availability can enhance availability during planned system maintenance, and help protect your databases against DB instance failure and Availability Zone disruption.



Failover process for Amazon RDS

In the event of a planned or unplanned outage of your DB instance, Amazon RDS automatically switches to a standby replica in another Availability Zone if you have enabled Multi-AZ. The time it takes for the failover to complete depends on the database activity and other conditions at the time the primary DB instance became unavailable. Failover times are typically 60–120 seconds. However, large transactions or a lengthy recovery process can increase failover time. When the failover is complete, it can take additional time for the RDS console to reflect the new Availability Zone.

Amazon RDS handles failovers automatically so you can resume database operations as quickly as possible without administrative intervention. The primary DB instance switches over automatically to the standby replica if any of the following conditions occur:

- ◆ An Availability Zone outage
- ◆ The primary DB instance fails
- ◆ The DB instance's server type is changed
- ◆ The operating system of the DB instance is undergoing software patching
- ◆ A manual failover of the DB instance was initiated using **Reboot with failover**

Lab Details

1. This lab walks you through the steps to launch an Amazon Aurora RDS DB instance with multi-AZ enabled. We will also simulate a database failover from one AZ to another.
2. You will practice using Amazon Aurora.
3. Duration: **1 hour**
4. AWS Region: **US East (N. Virginia) us-east-1**

Lab Tasks

1. In this lab session, first we are going to launch an Amazon Aurora RDS DB instance with Multi-AZ enabled.
2. Connect to the RDS database instance (using its endpoint) from your local machine.
3. Create a test database and table in your Master RDS DB instance.
4. Force the Master DB instance to failover.
5. After Failover, Master will change to Reader and Reader will change to Master
6. Connect to the new Master to test the database replication.

Steps

Creating an EC2 Instance

1. Click on AWS Management Console and set region N.Virginia

The screenshot shows the AWS Management Console homepage. The navigation bar at the top includes the AWS logo, a 'Services' dropdown, a search bar ('Search for services, features, marketplace products, and docs'), and account information ('parkubhole78@gmail.com', 'N. Virginia', 'Support'). The main content area is titled 'AWS Management Console'. On the left, there's a sidebar with 'AWS services' categorized under 'Recently visited services' (Billing, CloudWatch, Amazon CodeGuru, IAM) and 'All services' (Compute, Containers, Storage, etc.). The 'Compute' section is expanded, showing EC2 as the selected service. Other options like Lambda, Batch, Elastic Beanstalk, and Serverless Application Repository are also listed. The right side of the page contains promotional boxes for 'Stay connected to your AWS resources on-the-go', 'Explore AWS' (Amazon Elasticsearch Service, Amazon S3 on Outposts, Free Digital Training, Amazon SageMaker Autopilot), and 'AWS Cost Management'.

And select Ec2 from in compute and choose the EC2 Dashboard in this dashboard shown the resources working status.

The screenshot shows the 'Welcome to the new EC2 console' message. The left sidebar has a 'New EC2 Experience' toggle, followed by sections for 'EC2 Dashboard' (highlighted in red), 'Events', 'Tags', 'Limits', 'Instances' (with sub-options like Instances, Instance Types, Launch Templates, etc.), 'Images' (AMIs), and 'Snapshots'. The main content area is titled 'Resources' and displays a summary of Amazon EC2 resources in the US East (N. Virginia) Region. It shows counts for Instances (running), Dedicated Hosts, Elastic IPs, Instances (all states), Key pairs, Load balancers, Placement groups, Security groups, Snapshots, and Volumes. A callout box highlights the 'Instances (running)' count. To the right, the 'Account attributes' section lists supported platforms (VPC), default VPC (vpc-4a0bf137), settings, EBS encryption, zones, default credit specification, and console experiments. A promotional box for 'Save up to 90% on EC2 with Spot Instances' is also present.

2. **Choose an Amazon Machine Image (AMI):** Search for **Amazon Linux 2 AMI** in the search box and click on the **Select** button

Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search for an AMI by entering a search term e.g. "Windows"

Quick Start

- My AMIs
- AWS Marketplace
- Community AMIs

Free tier only

Amazon Linux Free tier eligible	Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0be2609ba883822ec (64-bit x86) / ami-0c582118883b46f4f (64-bit Arm)	<input type="button" value="Select"/>
Red Hat Free tier eligible	Red Hat Enterprise Linux 8 (HVM), SSD Volume Type - ami-096fda3c22c1c990a (64-bit x86) / ami-0b7affd26fd75a36 (64-bit Arm)	<input type="button" value="Select"/>
SUSE Linux Free tier eligible	SUSE Linux Enterprise Server 15 SP2 (HVM), SSD Volume Type - ami-0fde50fcbcd46f217 (64-bit x86) / ami-05f2f5f76d89313bb (64-bit Arm)	<input type="button" value="Select"/>

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

3. **Choose an Instance Type:** Select and then click on the Configure instance details button.

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance families ▾ Current generation ▾ Show/Hide Columns

Currently selected: t2.micro (~ ECUs, 1 vCPUs, 2.5 GHz, ~ 1 GiB memory, EBS only)

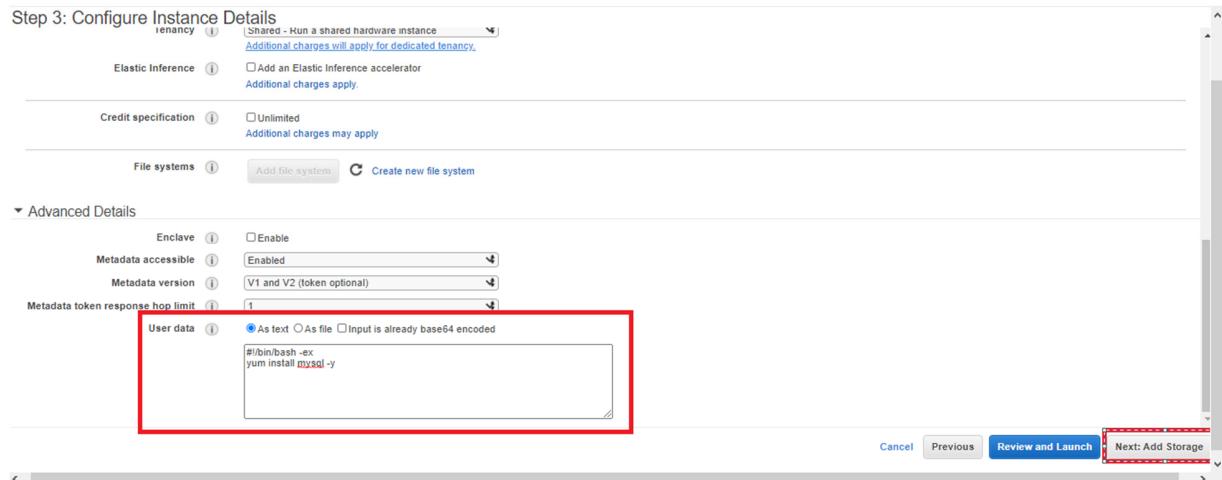
Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
I2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
I2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
I2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
I2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
I2	t2.large	2	8	EBS only	-	Low to Moderate	Yes
I2	t2.xlarge	4	16	EBS only	-	Moderate	Yes
I2	t2.2xlarge	8	32	EBS only	-	Moderate	Yes
I3	t3.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit	Yes
I3	t3.micro	2	1	EBS only	Yes	Up to 5 Gigabit	Yes

4. On the **Configure Instance Details** page:

- ◆ Network: Select default available VPC
- ◆ Subnet : Default selected

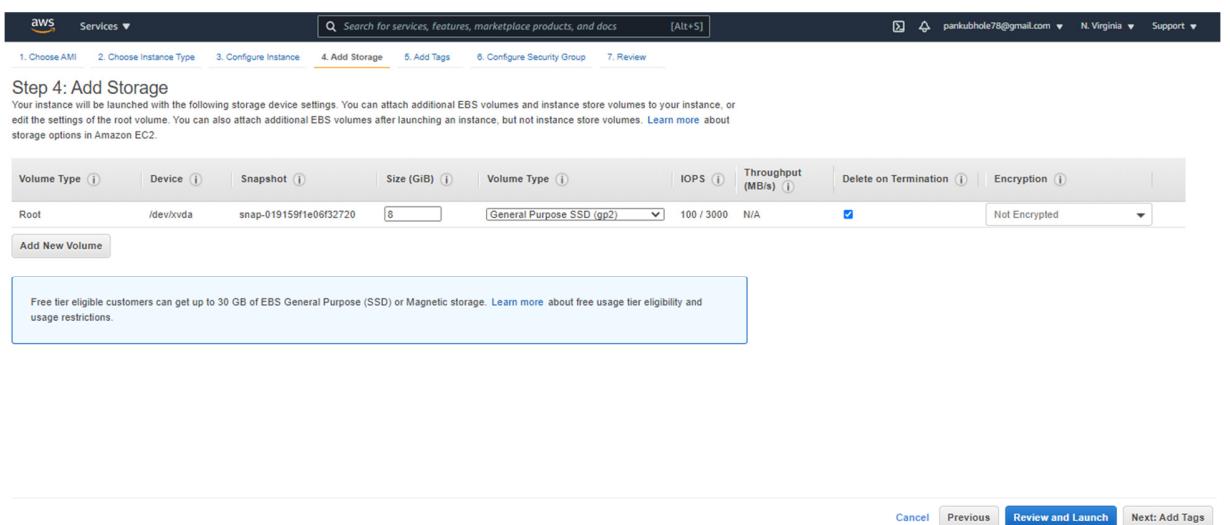
- ◆ Auto-assign Public IP: **Enable** - It should be enabled because the public IP is needed for connecting to EC2 via SSH.
- ◆ Click on
- ◆ Under the **User data** section, enter the following script, (which installs MySQL):

```
#!/bin/bash -ex
yum install mysql -y
```



- ◆ Now click on Next Add storage Page.

5. Add Storage Page: No need to change anything in this step. Click on Next Add tags button.



6. Add Tags Page

- ◆ Click on Add Tags Button
- ◆ Key : **Name**
- ◆ Value : **MyRdsEc2server**

Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.

A copy of a tag can be applied to volumes, instances or both.

Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key	(128 characters maximum)	Value	(256 characters maximum)	Instances	Volumes	Snapshot
name		MyRDSEc2Server		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Add another tag (Up to 50 tags maximum)

Cancel Previous Review and Launch Next: Configure Security Group

- ◆ Click on Configure Security group button

7. On the **Configure Security Group** page:

- ◆ **Assign a security group:** Create a **new** security group
- ◆ Security group name: **MyEc2server-SG**
- ◆ Description: **Security for ec2 server to connect with RDS**
- ◆ To add **SSH:**
 - ◆ Choose Type:
 - ◆ **Source: Custom** Enter **0.0.0.0/0** in the textbox or select **anywhere**.

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: Create a new security group Select an existing security group

Security group name: MyEc2server-SG

Description: Security for ec2 server to connect with RDS

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Anywhere 0.0.0.0/0	e.g. SSH for Admin Desktop

Add Rule

Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel Previous Review and Launch

8. Review and Launch: Review all your settings and click on Launch button.

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

AMI Details Edit AMI

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0be2609ba883822ec
Free tier eligible

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is a...
Root Device Type: ebs Virtualization type: hvm

Instance Type Edit instance type

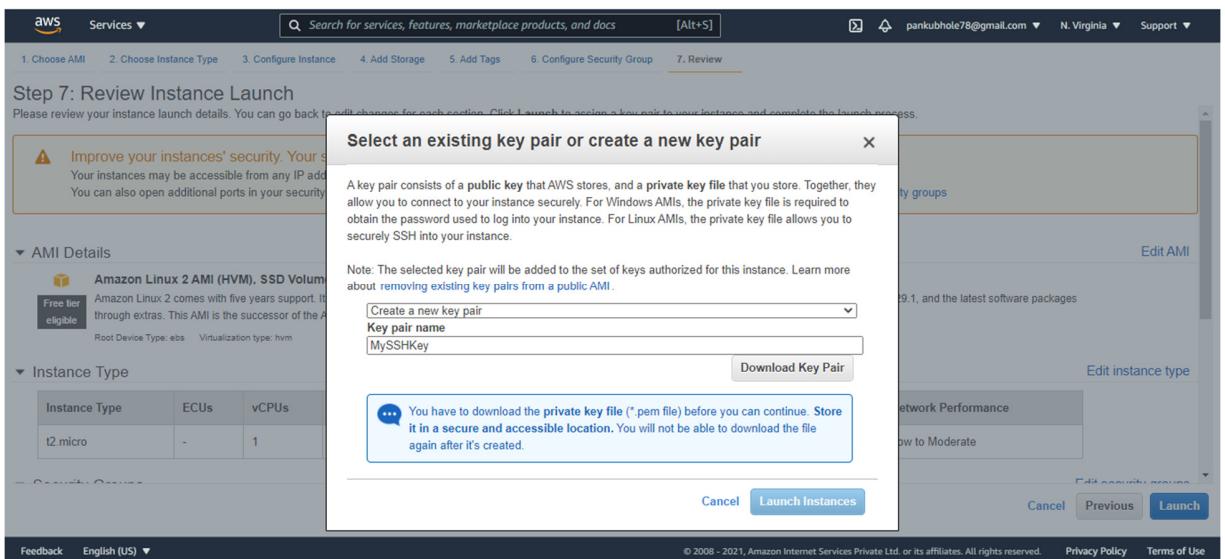
Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2 micro	-	1	1	EBS only	-	Low to Moderate

Feedback English (US) © 2006–2021 Amazon Web Services, Inc., or its affiliates. All rights reserved. Privacy Policy Terms of Use

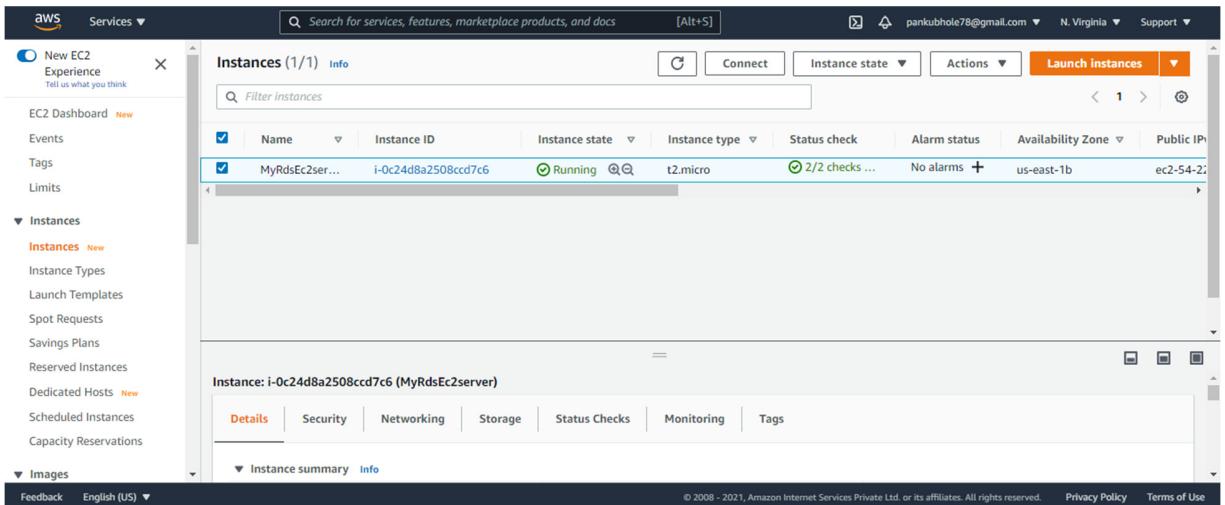
Cancel Previous Launch

9. Key Pair - Select **Create a new key Pair** with name **MySSHKey**, click on **Launch instance button** and save it to your local machine.

10. Once the download is complete, click on **Download Key Pair button**.



11. After 1-2 minutes, the **Instance State** will become **running** as shown below:



12. Note down **IPv4 Private IP address** in your text editor, navigate to the EC2 Dashboard and look in the instance details.

The screenshot shows the AWS EC2 Instances page. A single instance named "MyRdsEc2server" is listed. The instance ID is i-0c24d8a2508cccd7c6. It is currently running. The instance type is t2.micro. The status check is 2/2 checks. The alarm status is no alarms. The availability zone is us-east-1b, and the public IP is ec2-54-221-111-101. The instance details pane shows the instance ID, instance state, instance type, and IAM role. The public IPv4 address (54.221.111.101) is highlighted with a yellow box. The private IPv4 address (172.31.20.196) and private IPv4 DNS (ip-172-31-20-196.ec2.internal) are also visible but not highlighted.

Create a Security Group for RDS instance

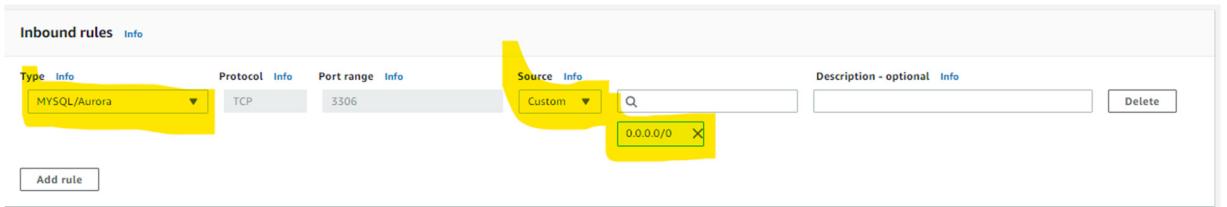
1. Make sure you are in the **N.Virginia** Region.
2. Navigate to EC2 by clicking on the menu available under the section.
3. On the left panel menu, select the security group under the **Network & Security** section.

The screenshot shows the AWS Management Console with the 'Services' dropdown open. The 'Network & Security' section is highlighted with a yellow box. The main content area displays a table titled 'Security Groups (9)'. The columns are: Name, Security group ID, Security group name, VPC ID, Description, and Owner. The table lists various security groups like 'WEB SERVER-SG', 'Loadbalancer-SG', and 'Bastion-SG', along with their respective details.

4. Click on the **Create Security Group** button.
5. We are going to create a Security group for RDS with 3306 port number enabled.
 - o Security group name : Enter **rds-maz-SG**
 - o Description : Enter **Security group for RDS Aurora**
 - o VPC : Select Default VPC

The screenshot shows the 'Create security group' wizard. The first step, 'Basic details', is completed. The 'Security group name' is set to 'rds-maz-SG', the 'Description' is 'Security group for RDS Aurora', and the 'VPC' is set to 'Default VPC'. The next steps in the wizard are 'Inbound rules', 'Outbound rules', and 'Tags'.

- o Click on the **Add Rule** button under **Inbound rules**.
 - ◆ In the textbox add **0.0.0.0/0**
 - ◆ Source : Select **Custom**
 - ◆ Type : Select **MYSQL/Aurora**

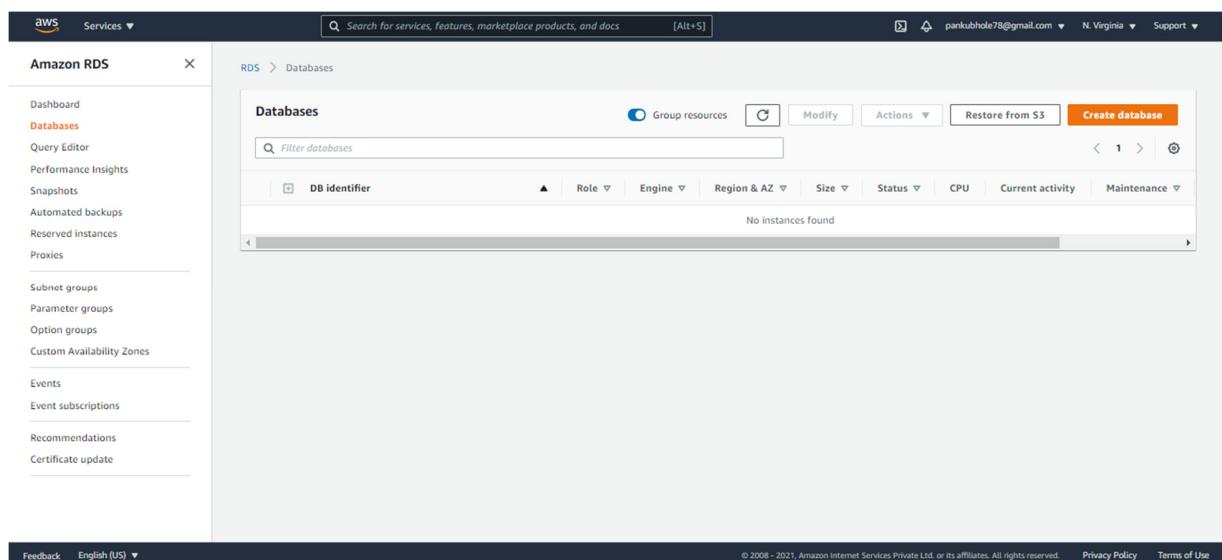


6. Leave everything as default and click on the **Create Security Group** button.

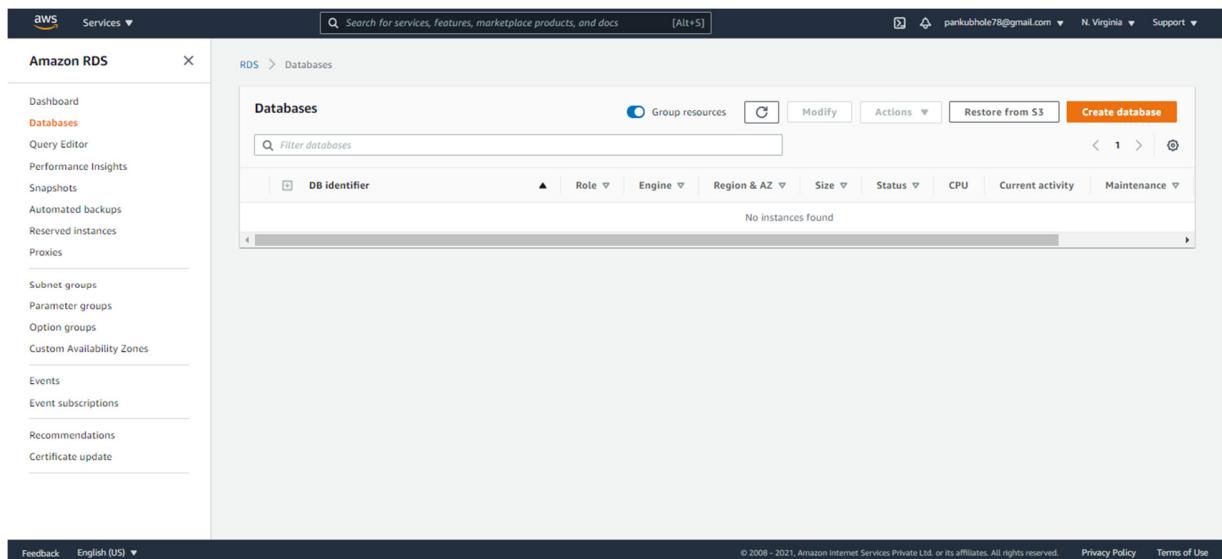
Type	Protocol	Port range	Source	Description - optional
MySQL/Aurora	TCP	3306	0.0.0.0/0	-

Create an Amazon Aurora database with Multi-AZ enabled

1. Make sure you are in **N.Virginia** Region.
2. Navigate to **RDS** under the database section of the **Services** menu
3. Click on **Databases** in the left Panel.



4. Click on **Create database**.
5. Ignore the warning mentioned below and continue with the next steps.

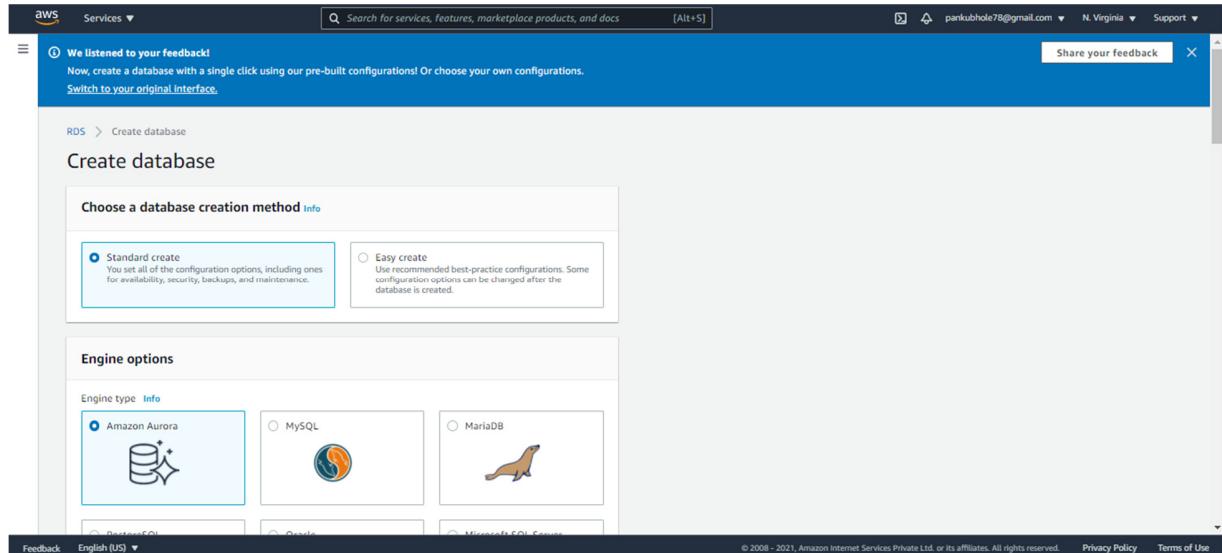


Note: If you are in the original interface, make sure to click on the **Switch to the new database creation flow** as shown below.

6. Next we'll configure the database on the **Create Database Page**

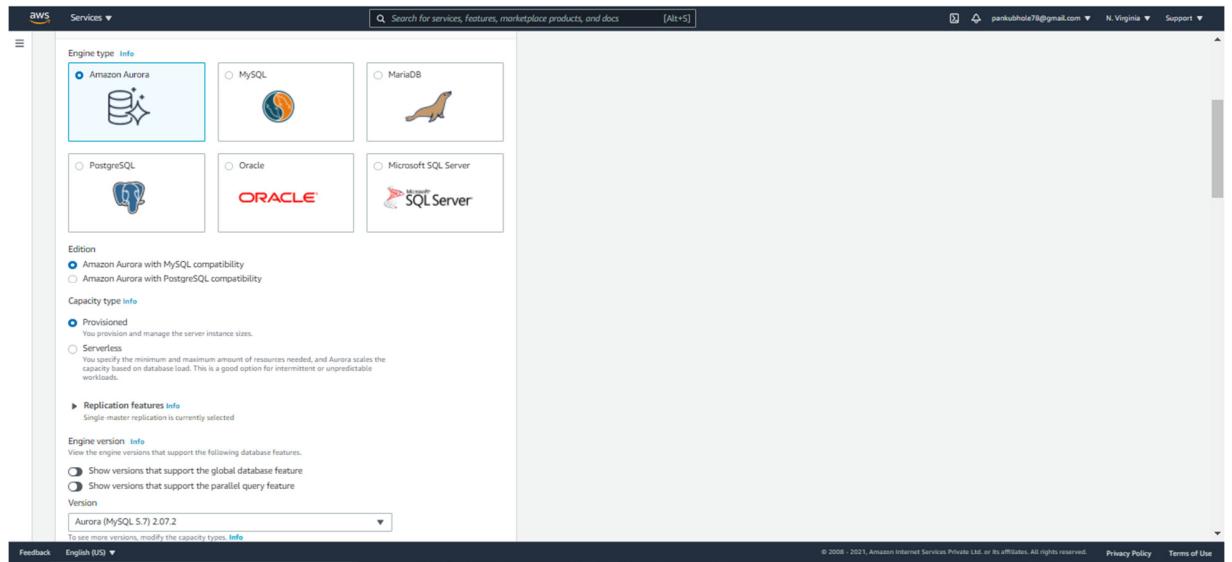
- o Choose a **Database Creation Method**:

- Select **Standard Create**



- o In **Engine options**:

- o Engine type : Choose **Amazon Aurora**
- o Edition : Choose **Amazon Aurora with MySQL compatibility**
- o Capacity Type : **Provisioned**
- o Version : **Default (Aurora (MySQL 5.7) 2.07.2)**
- o Replication features:
 - Select **Single-master (default)**



- Choose **Template: Dev/Test**
- Fill in the required details for the database (Aurora Cluster Settings)
 - **DB cluster identifier:** Specify cluster name *MyAuroraCluster*
 - Give the following details in the **credential settings**
 - **Master Username:** *labsAdmin*
 - **Master password:** *labs123*
 - **Confirm password:** *labs123*

Templates
Choose a sample template to meet your use case.

Production
Use defaults for high availability and fast, consistent performance.

Dev/Test
This instance is intended for development use outside of a production environment.

Settings

DB cluster identifier [Info](#)
Type a name for your DB cluster. The name must be unique across all DB clusters owned by your AWS account in the current AWS Region.

The DB cluster identifier is case-insensitive, but is stored as all lowercase (as in "mydbcluster"). Constraints: 1 to 60 alphanumeric characters or hyphens. 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.

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), ' (single quote), " (double quote) and @ (at sign).

Confirm password [Info](#)

⚠ Must be 8 characters or more

- **Note:** This is the username and password used to log into your database. Please make note of them.
- **Choosing DB instance size**
- **DB instance class:** Choose **Burstable classes (includes t classes)**

- By Default, **db.t3.small** will be selected, please enable the option below of **Include previous generation classes**.
- Select **db.t2.small** instance.

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.

- Memory Optimized classes (includes r classes)
 Burstable classes (includes t classes)

db.t2.small
 1 vCPUs 2 GiB RAM Not EBS Optimized

- Include previous generation classes

- Note: Incase, db.t2.small is not available, we have added support for db.t3.small also, meaning you can go ahead with any of the two types.

- Availability and Durability: Choose Multi-AZ deployment: **Create Aurora Replica or Reader node in a different AZ** as shown below:

Availability & durability

Multi-AZ deployment [Info](#)

- Create an Aurora Replica or Reader node in a different AZ (recommended for scaled availability)
 Creates an Aurora Replica for fast failover and high availability.
- Don't create an Aurora Replica

- o **Connectivity**
 - Choose the Default VPC

The screenshot shows the 'Connectivity' section of the AWS RDS configuration interface. A dropdown menu is open, showing 'Default VPC (vpc-4a0bf137)' as the selected option. Below the dropdown, a message states: 'Only VPCs with a corresponding DB subnet group are listed.' A callout box contains the note: 'After a database is created, you can't change the VPC selection.'

- o Additional connectivity configuration
 - Subnet group : **Default**
 - Publicly accessible : **Yes**
 - Existing VPC security groups :
 - o Remove the Default security group, which is selected by default.
 - o Select **rds-maz-SG** for the dropdown.(This is the security group which you have created in the beginning)
 - o Database port : **3306**

The screenshot shows the 'VPC configuration' section of the AWS RDS configuration interface. Under 'Subnet group', 'default' is selected. Under 'Public access', 'Yes' is selected. Under 'VPC security group', 'Choose existing' is selected, and 'rds-maz-SG' is chosen from the dropdown. Under 'Existing VPC security groups', 'rds-maz-SG' is listed. At the bottom, under 'Additional configuration', the 'Database port' is set to '3306'.

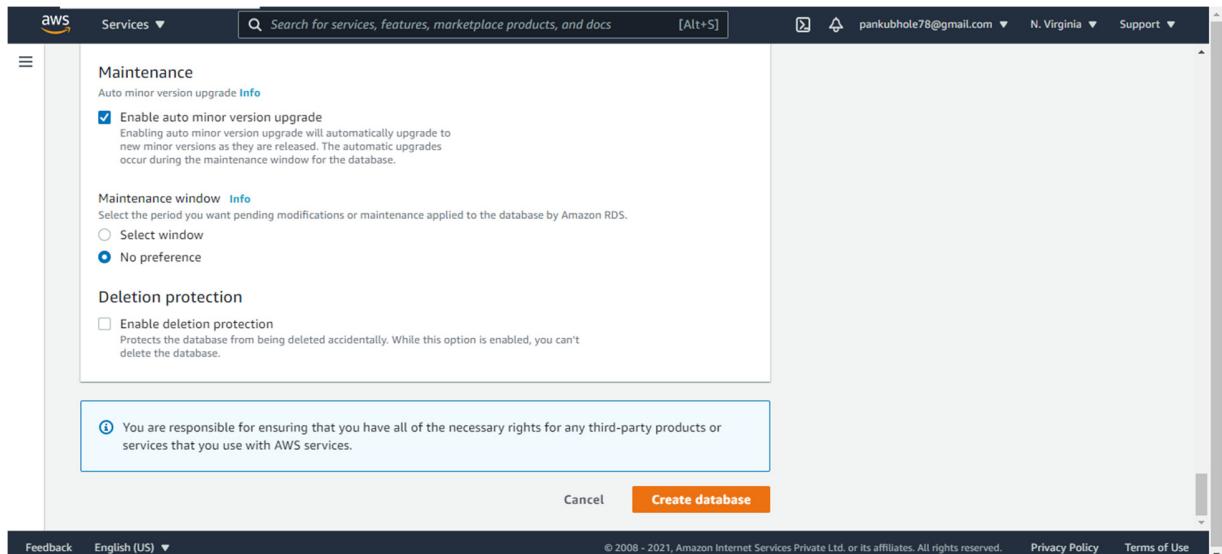
- **Additional configuration:**
 - Database options
 - Initial database name: **whizlabsrds**

The screenshot shows the 'Additional configuration' section of the AWS RDS console. Under 'Database options', the 'Initial database name' is set to 'whizlabsrds'. Below it, a note states: 'If you do not specify a database name, Amazon RDS does not create a database.' Other settings include 'DB cluster parameter group' (set to 'default.aurora-mysql5.7'), 'DB parameter group' (set to 'default.aurora-mysql5.7'), and 'Option group' (set to 'default:aurora-mysql-5.7'). The 'Failover priority' is set to 'No preference'.

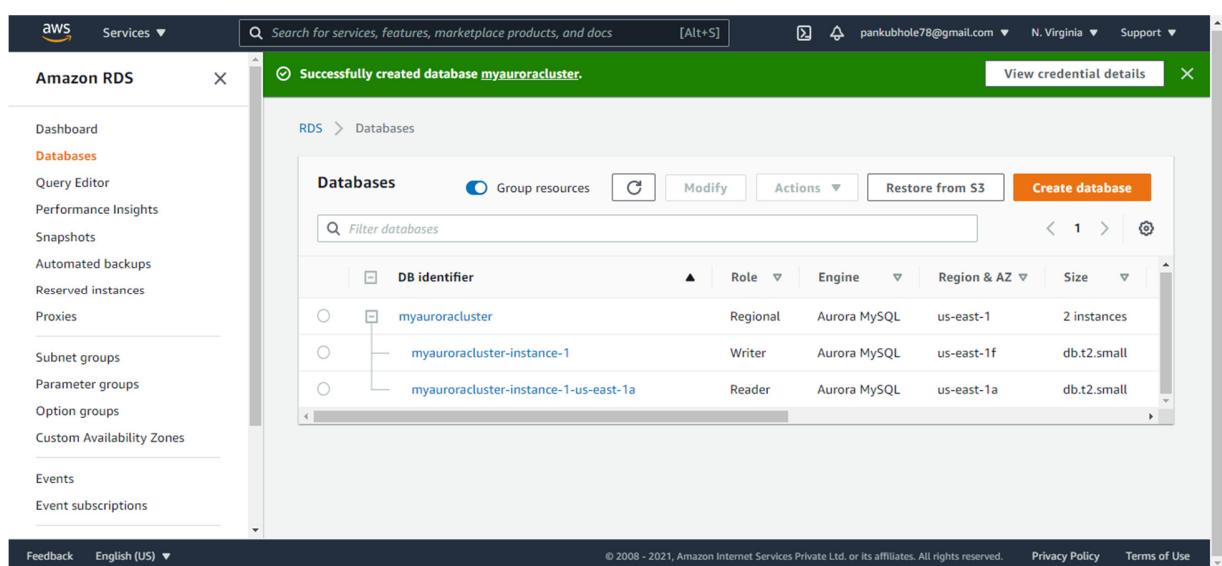
- **Encryption : Uncheck**

The screenshot shows the 'Backup' configuration section of the AWS RDS console. Under 'Encryption', the 'Enable encryption' checkbox is unchecked. A note below it says: 'Choose to encrypt the given instance. Master key IDs and aliases appear in the list after they have been created using the AWS Key Management Service console.' Under 'Backtrack', both 'Enable encryption' and 'Enable Backtrack' checkboxes are unchecked. A note below 'Enable Backtrack' says: 'Enabling Backtrack will charge you for storing the changes you make for backtracking.' Other sections shown include 'Monitoring' where 'Enable Enhanced monitoring' is checked.

- Leave other settings as default



6. Once the details above have been filled in, click on **Create database button**.
7. It will take around **10-15 minutes** for the database to be created. Please wait until database status changes from **creating** to **Available**.
8. Once the database has been created, you should see the following page:



Similar to the screenshot, you should be able to see that our database launched in multiple AZs, namely **us-east-1d** and **us-east-1e**

Connecting to the Aurora (MySQL) database on RDS

Now we have successfully launched Aurora RDS with Multi-AZ enabled. To connect to the new Aurora database, we need the **endpoint**.

1. Click on the **RDS cluster name** and then navigate to **Connectivity & security** to find the endpoint of your **Master (Writer)** and **Reader** instances, with which you can connect to your DB instance.

The screenshot shows the AWS RDS console. On the left, there's a sidebar with various options like Dashboard, Databases, Query Editor, etc. The main area shows a table of databases:

DB identifier	Role	Engine	Region & AZ	Size	Status	CPU
myauroracluster	Regional	Aurora MySQL	us-east-1	2 instances	Available	-
myauroracluster-instance-1	Writer	Aurora MySQL	us-east-1f	db.t2.small	Creating	-
myauroracluster-instance-1-us-east-1a	Reader	Aurora MySQL	us-east-1a	db.t2.small	Creating	-

Below this, there's a tab labeled "Connectivity & security" which is currently selected. It shows the "Endpoints" section with two entries:

Endpoint name	Status	Type	Port
myauroracluster.cluster-c0f1g4tlglvz.us-east-1.rds.amazonaws.com	Available	Writer	3306
myauroracluster.cluster-ro-c0f1g4tlglvz.us-east-1.rds.amazonaws.com	Available	Reader	3306

- The endpoints you see to be similar to these examples:

Master(Writer): myauroracluster.cluster-c0f1g4tlglvz.us-east-1.rds.amazonaws.com

The screenshot shows the AWS RDS console for the 'Amazon RDS' service. On the left, the navigation pane includes options like Dashboard, Databases, Query Editor, Performance Insights, Snapshots, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom Availability Zones, Events, Event subscriptions, Recommendations, and Certificate update. The main content area displays a table of DB identifiers. One row is expanded to show details for 'myauroracluster':

DB identifier	Role	Engine	Region & AZ	Size	Status	CPU
myauroracluster	Regional	Aurora MySQL	us-east-1	2 instances	Available	-
myauroracluster-instance-1	Writer	Aurora MySQL	us-east-1f	db.t2.small	Available	
myauroracluster-instance-1-us-east-1a	Reader	Aurora MySQL	us-east-1a	db.t2.small	Available	

Below the table, there are tabs for Connectivity & security, Monitoring, Logs & events, Configuration, Maintenance, and Tags. The Connectivity & security tab is selected, showing the following details:

Endpoint & port	Networking	Security
Endpoint myauroracluster-instance-1c0f1g4t1glvz.us-east-1.rds.amazonaws.com	Availability zone us-east-1f VPC vpc-4a0bf137	VPC security groups rds-maz-SG (sg-02b8b5b4fca5f9d4e) (active)
Port 3306	Subnet group default-vpc-4a0bf137	Public accessibility Yes Certificate authority rds-ca-2019

Reader: myauroracluster.cluster-ro-c0f1g4t1glvz.us-east-1.rds.amazonaws.com

This screenshot is identical to the one above, showing the AWS RDS console with the same cluster configuration and connectivity details.

Note: Please carefully look at the role of the DB instance (**Reader vs Master (Writer)**) and their respective availability zones. (Here **us-east-1d** and **us-east-1e**)

- To get the endpoint of the RDS instances, click on the name of the cluster. Then you should click on **Endpoints**. This will expose the read and write endpoints for the database. See the example below:

Connecting the EC2 Server to RDS:

1. Now we need to connect the RDS with ec2 server in order to eventually connect with the Aurora database.
2. Navigate to RDS available under the database section of the Services menu.
3. Click on Master (writer) database and click on the security group name in this example it is **rds-maz-SG** under VPC security groups as shown below:

The screenshot shows the AWS Management Console with the search bar set to "Security Groups (1/1)". A filter is applied for the security group ID "sg-02b8b5b4fcfa5f9d4e". The table lists one item: "RDS-SG" with ID "sg-02b8b5b4fcfa5f9d4e", VPC ID "vpc-4a0bf137", and a description "Security group for RDS...". Below the table, the "Inbound rules" tab is selected, showing a single rule: Type "MySQL/Aurora", Protocol "TCP", Port range "3306", and Source "0.0.0.0/0".

4. It will open the Security Group page. Click on **InBound**.
- The MySQL rule will already exists.
- Under source, delete any pre-populated IP Address and enter the **private IP of your MyRdsEc2server** EC2 instance with CIDR /32 (**EC2 instance Private IP**) and then click on **Save Rule** button as shown below:

The screenshot shows the AWS Security Groups console. A green header bar indicates that security group rules have been successfully modified. The main table lists a single security group, RDS-5G, with details such as Security group ID (sg-02b8b5b4fca5f9d4e), Security group name (rds-maz-5G), VPC ID (vpc-4a0bf137), and Description (Security group for RDS...). Below the table, the Inbound rules tab is selected, showing one rule for MySQL/Aurora (TCP port 3306) from 172.31.20.196/32.

Execute Database Operations via SSH

1. Copy the **IPv4 Public IP address**, navigate to the EC2 Dashboard and look in the instance details.

The screenshot shows the AWS EC2 Instances console. A blue header bar welcomes users to the new instances experience. The main table shows one instance, MyRdsEc2server, which is running (Status check: 2/2 checks ...). The Details tab is selected, displaying the instance ID (i-0c24d8a2508cccd7c6), instance state (Running), instance type (t2.micro), and public IP address (54.221.111.101). The Public IPv4 address is also listed as 172.31.20.196.

2. SSH into the EC2 instance we just created through the following steps in [SSH into EC2 Instance](#).
3. Switch to the root user using the command : **sudo -s**

The screenshot shows a terminal session on an Amazon Linux 2 AMI instance. The session starts with a welcome message from the OS, followed by a command to change the user's password (sudo -s), and ends with a root prompt (#). The terminal window is part of the AWS CloudWatch Metrics Insights interface.

```
Last login: Wed Jan  6 08:44:55 2021 from ec2-18-206-107-24.compute-1.amazonaws.com
[ec2-user@ip-172-31-20-196 ~]$ sudo -s
[root@ip-172-31-20-196 ec2-user]#
```

i-0c24d8a2508ccd7c6 (MyRdsEc2server)

Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

4. Log into the RDS instance using the below command:

- **Syntax:** mysql -h <Hostname> -u <username> -p

Note: Make sure to change the above **Master(Writer)Cluster endpoint** and **Username** with your's.

- Host name : **myauroracluster.cluster-c0f1g4tlglvz.us-east-1.rds.amazonaws.com (Master(Writer)cluster endpoint)**
- Username : **labsAdmin**
- Password : **labs123** (Use yours in case you changed the password while creating RDS)
- Database name : **labsrds**
- You should now be able to log into the database, as shown below:

```

root@ip-172-31-20-196 ~# mysql -h myauroracluster.cluster-c0fig4t1glvz.us-east-1.rds.amazonaws.com -u labsAdmin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 25
Server version: 5.7.12 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>

```

i-0c24d8a2508cccd7c6 (MyRdsEc2server)
Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

4. List all Databases:

- **Show databases;**

Now you will see the database **labsrds** created while launching the RDS cluster.

5. Now create the database in the **Master(Writer) RDS** as given in the screenshot. We'll create a demo database named **auroro_db**.

- **Create database auroro_db;**

6. Select the newly-created database:

- **use auroro_db;**

7. Next we'll create a table named **students** and insert few rows of data using list of commands:

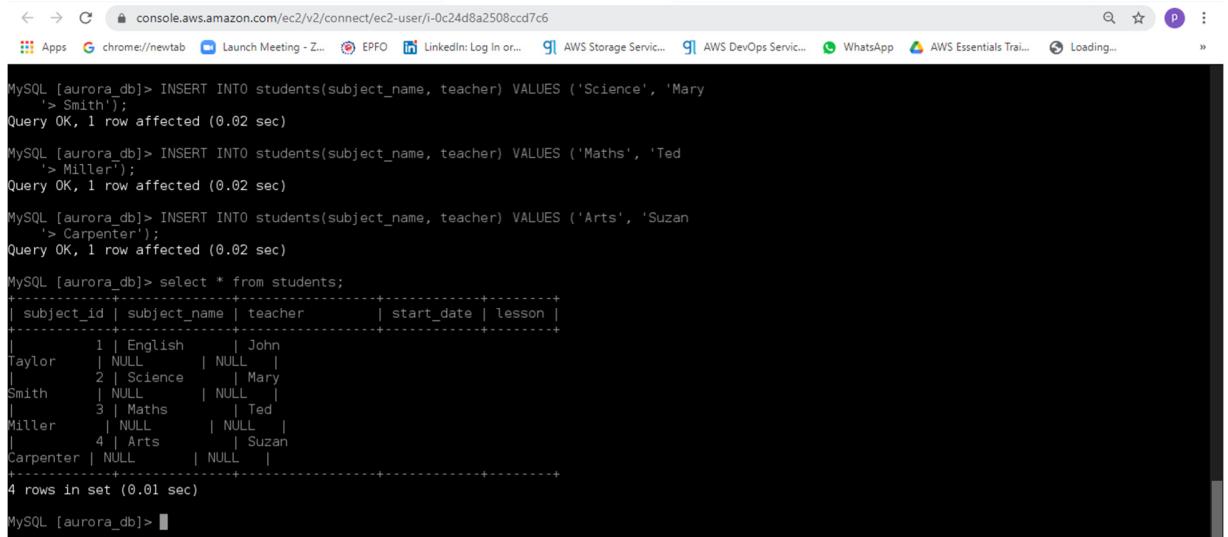
- **CREATE TABLE students (subject_id INT AUTO_INCREMENT, subject_name VARCHAR(255) NOT NULL,teacher VARCHAR(255),start_date DATE, lesson TEXT,PRIMARY KEY (subject_id));**

8. Insert data into the table:

- `INSERT INTO students(subject_name, teacher) VALUES ('English', 'John Taylor');`
- `INSERT INTO students(subject_name, teacher) VALUES ('Science', 'Mary Smith');`
- `INSERT INTO students(subject_name, teacher) VALUES ('Maths', 'Ted Miller');`
- `INSERT INTO students(subject_name, teacher) VALUES ('Arts', 'Suzan Carpenter');`

9. Now you can view the contents of the table student using the below command:

- `select * from students;`



The screenshot shows a browser window with the URL `console.aws.amazon.com/ec2/v2/connect/ec2-user/i-0c24d8a2508cccd7c6`. The page displays a MySQL session on the `aurora_db` database. The user has run three `INSERT` statements to add data to the `students` table:

```
MySQL [aurora_db]> INSERT INTO students(subject_name, teacher) VALUES ('Science', 'Mary
-> Smith');
Query OK, 1 row affected (0.02 sec)

MySQL [aurora_db]> INSERT INTO students(subject_name, teacher) VALUES ('Maths', 'Ted
-> Miller');
Query OK, 1 row affected (0.02 sec)

MySQL [aurora_db]> INSERT INTO students(subject_name, teacher) VALUES ('Arts', 'Suzan
-> Carpenter');
Query OK, 1 row affected (0.02 sec)
```

After the inserts, the user runs a `SELECT * FROM students;` query, which returns the following results:

subject_id	subject_name	teacher	start_date	lesson
1	English	John Taylor	NULL	NULL
2	Science	Mary Smith	NULL	NULL
3	Maths	Ted Miller	NULL	NULL
4	Arts	Suzan Carpenter	NULL	NULL

4 rows in set (0.01 sec)

MySQL [aurora_db]>

At the bottom of the screen, the server information is shown:

i-0c24d8a2508cccd7c6 (MyRdsEc2server)
Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

10. Exit from mysql console use the below command:

- `exit`

Forcing a Failover to Test Multi-AZ

1. To test if Multi-AZ is working, we will create a situation where master fails and the read replica has to become the new Master (Writer).
2. On the next screen, confirm the Failover.

The screenshot shows the AWS RDS console for an Aurora MySQL cluster named "myauroracluster". The cluster has three instances: "myauroracluster-instance-1" (Writer, us-east-1f, db.t2.small), "myauroracluster-instance-1-us-east-1a" (Reader, us-east-1a, db.t2.small), and "myauroracluster-instance-1-us-east-1f" (Reader, us-east-1f, db.t2.small). The "Connectivity & security" tab is selected, displaying network configuration including endpoint, VPC, subnet group, and security groups (rds-maz-SG). The "Security" section also shows public accessibility set to "Yes" and certificate authority "rds-ca-2019".

3. Wait for a few minutes for the RDS instances to failover.

The screenshot shows the AWS RDS console with the "Databases" tab selected. The cluster "myauroracluster" now has two instances: "myauroracluster-instance-1" (Writer, us-east-1a, db.t2.small) and "myauroracluster-instance-1-us-east-1f" (Reader, us-east-1f, db.t2.small). The "myauroracluster-instance-1" instance is highlighted, indicating it is the current writer. The "Current activity" column shows usage statistics for both instances.

- (i.e. Master(Writer) becomes Reader and Reader becomes Master(Writer) as shown below)

Testing the Failover Condition

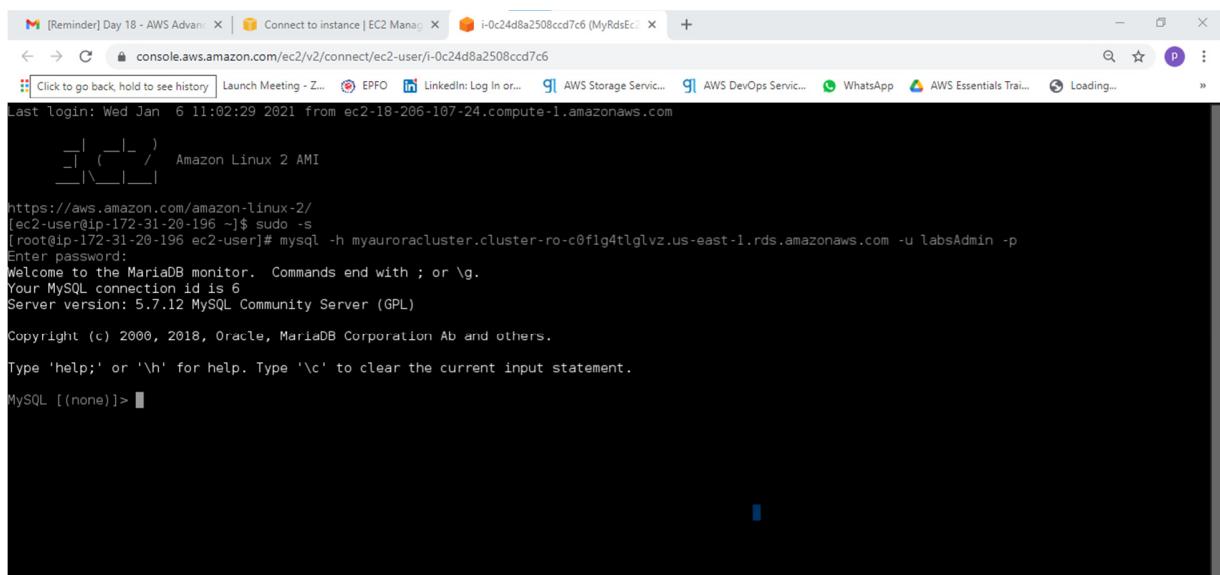
1. Now connect to RDS with new Master endpoint

- o Copy the endpoint of the **new Master (Writer) cluster** and replace it with your endpoint link.

MySQL -h <endpoint> -u <username> -p and press [Enter]

- o **MySQL -h myauroracluster.cluster-cpoz6c7903cx.us-east-1.rds.amazonaws.com -u labsAdmin -p**

Password: labs123



```
[Reminder] Day 18 - AWS Advanced | Connect to instance | EC2 Manager | i-0c24d8a2508cccd7c6 (MyRdsEc2server) +  
Last login: Wed Jan 6 11:02:29 2021 from ec2-18-206-107-24.compute-1.amazonaws.com  
[ec2-user@ip-172-31-20-196 ~]$ sudo -s  
[root@ip-172-31-20-196 ec2-user]# mysql -h myauroracluster.cluster-ro-c0fig4tlglvz.us-east-1.rds.amazonaws.com -u labsAdmin -p  
Enter password:  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MySQL connection id is 6  
Server version: 5.7.12 MySQL Community Server (GPL)  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
MySQL [(none)]>
```

i-0c24d8a2508cccd7c6 (MyRdsEc2server)

Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

2. You will be **able to Log** into MySQL and check for the database and table created in the master DB instance before the failover.

You can notice the resources created on the original master db are present, implying that the Failover worked successfully.

- **Show databases;**
- **use auroro_db;**

```

[Reminder] Day 18 - AWS Advanced | Connect to instance | EC2 Manager | i-0c24d8a2508cccd7c6 (MyRdsEc2server) | +
← → ⌂ 🔍 star p ⋮
console.aws.amazon.com/ec2/v2/connect/ec2-user/i-0c24d8a2508cccd7c6
Apps chrome://newtab Launch Meeting - Z... EPFO LinkedIn: Log In or... AWS Storage Servic... AWS DevOps Servic... WhatsApp AWS Essentials Train... Loading...
root@ip-172-31-20-196 ~% mysql -h myauroracluster.cluster-ro-c0fig4t1glvz.us-east-1.rds.amazonaws.com -u labsAdmin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.7.12 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| aurora_db |
| mysql |
| performance_schema |
| sys |
| whizlabsrds |
+-----+
5 rows in set (0.02 sec)

MySQL [(none)]> use aurora_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [aurora_db]>

i-0c24d8a2508cccd7c6 (MyRdsEc2server)
Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

```

3. Now check the existence of table named **students** and **data** (that we created earlier in the lab):

- **show tables;**
- **select * from students;**

```

[Reminder] Day 18 - AWS Advanced | Connect to instance | EC2 Manager | i-0c24d8a2508cccd7c6 (MyRdsEc2server) | +
← → ⌂ 🔍 star p ⋮
console.aws.amazon.com/ec2/v2/connect/ec2-user/i-0c24d8a2508cccd7c6
Click to go back, hold see history punch Meeting - Z... EPFO LinkedIn: Log In or... AWS Storage Servic... AWS DevOps Servic... WhatsApp AWS Essentials Train... Loading...
MySQL [(none)]> use aurora_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [aurora_db]> show tables;
+-----+
| Tables_in_aurora_db |
+-----+
| students |
+-----+
1 row in set (0.02 sec)

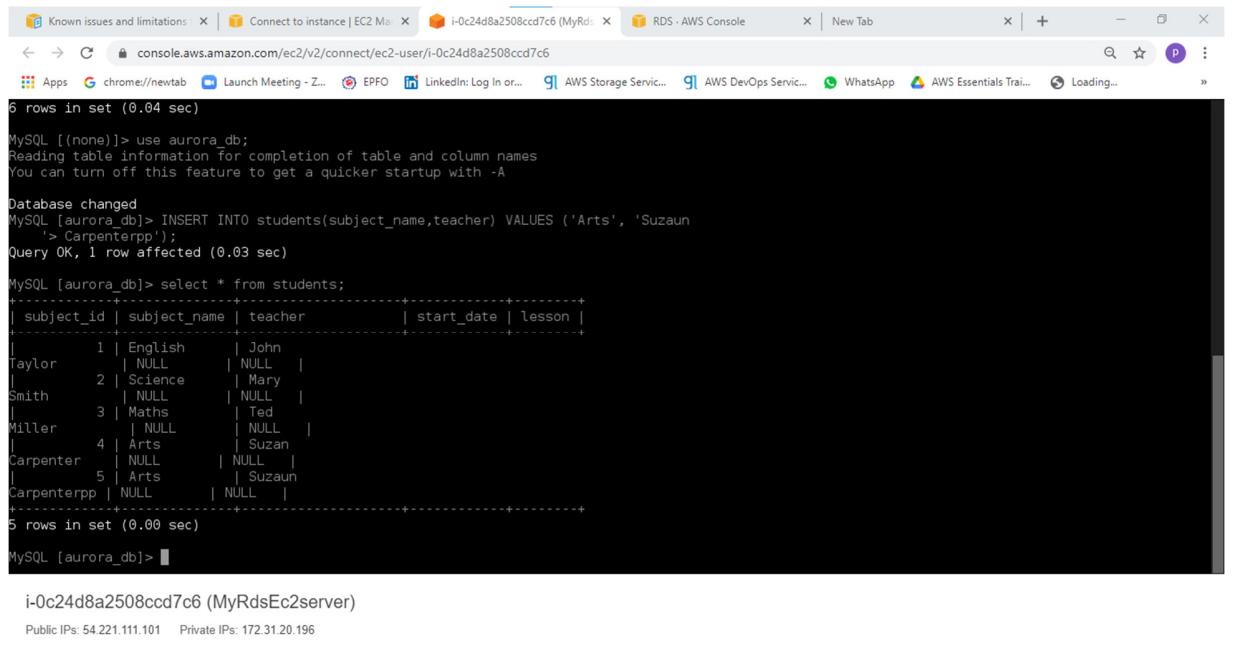
MySQL [aurora_db]> select* from students;
+-----+-----+-----+-----+
| subject_id | subject_name | teacher | start_date | lesson |
+-----+-----+-----+-----+
| 1 | English | John | |
| Taylor | NULL | NULL | Mary |
| 2 | Science | NULL |
| Smith | NULL | NULL | Ted |
| 3 | Maths | NULL |
| Miller | NULL | NULL |
| 4 | Arts | Suzan |
| Carpenter | NULL | NULL |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)

MySQL [aurora_db]>

i-0c24d8a2508cccd7c6 (MyRdsEc2server)
Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

```

4. Now if you want you can insert new data into the table students.
 - o **INSERT INTO students(subject_name, teacher) VALUES ('Spanish', 'Isabella);**
 - o **select * from students;**
5. **INSERT INTO students(subject_name,teacher) VALUES ('Arts', 'Suzaun Carpenterpp');**



```

Known issues and limitations | Connect to instance | EC2 Mail | i-0c24d8a2508cccd7c6 (MyRdsEc2server) | RDS - AWS Console | New Tab | - | + | ×
← → ⌂ console.aws.amazon.com/ec2/v2/connect/ec2-user/i-0c24d8a2508cccd7c6
Apps chrome://newtab Launch Meeting - Z... EPFO LinkedIn: Log In or... AWS Storage Service... AWS DevOps Service... WhatsApp AWS Essentials Training Loading...
6 rows in set (0.04 sec)

MySQL [(none)]> use aurora_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [aurora_db]> INSERT INTO students(subject_name,teacher) VALUES ('Arts', 'Suzaun
    '> Carpenterpp');
Query OK, 1 row affected (0.03 sec)

MySQL [aurora_db]> select * from students;
+-----+-----+-----+-----+
| subject_id | subject_name | teacher | start_date | lesson |
+-----+-----+-----+-----+
| 1 | English | John | NULL | NULL |
| 2 | Science | Mary | NULL | NULL |
| 3 | Maths | Ted | NULL | NULL |
| 4 | Arts | Suzan | NULL | NULL |
| 5 | Arts | Suzaun | NULL | NULL |
| NULL | NULL | NULL | NULL | NULL |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)

MySQL [aurora_db]> ■

```

i-0c24d8a2508cccd7c6 (MyRdsEc2server)
 Public IPs: 54.221.111.101 Private IPs: 172.31.20.196