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

Lets'sUpgrade AWS Advance Program

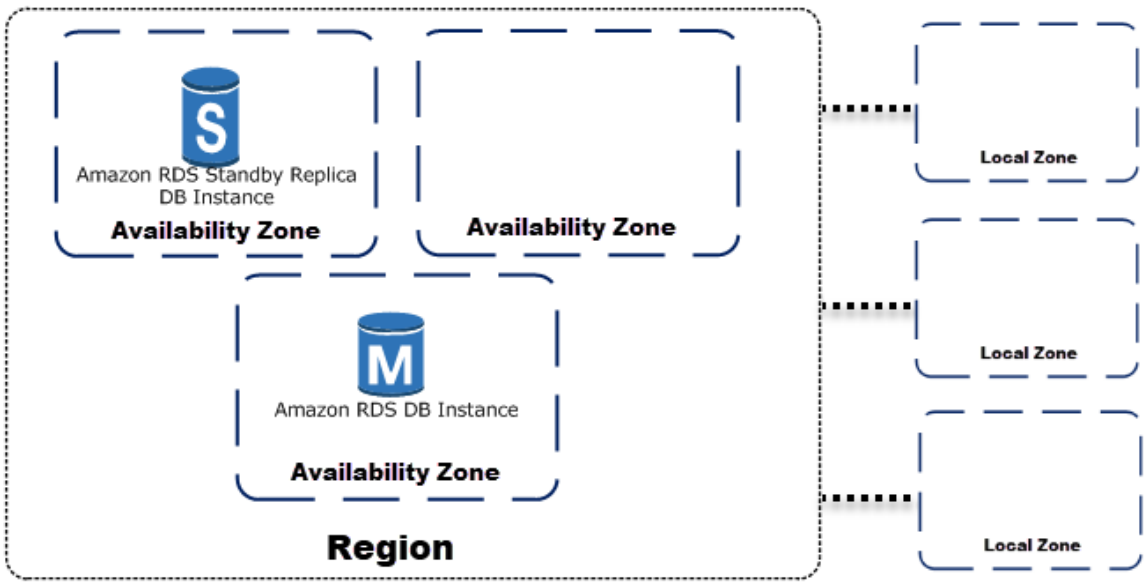
Help by Mrs. Vinolin Jermiah

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**

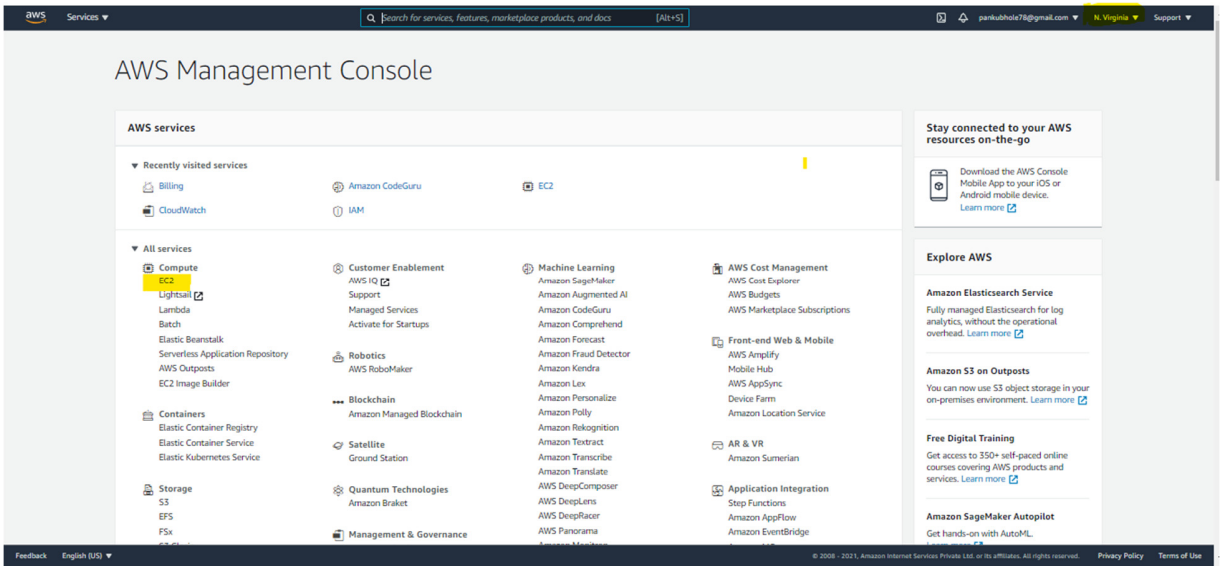
Lab Tasks

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.

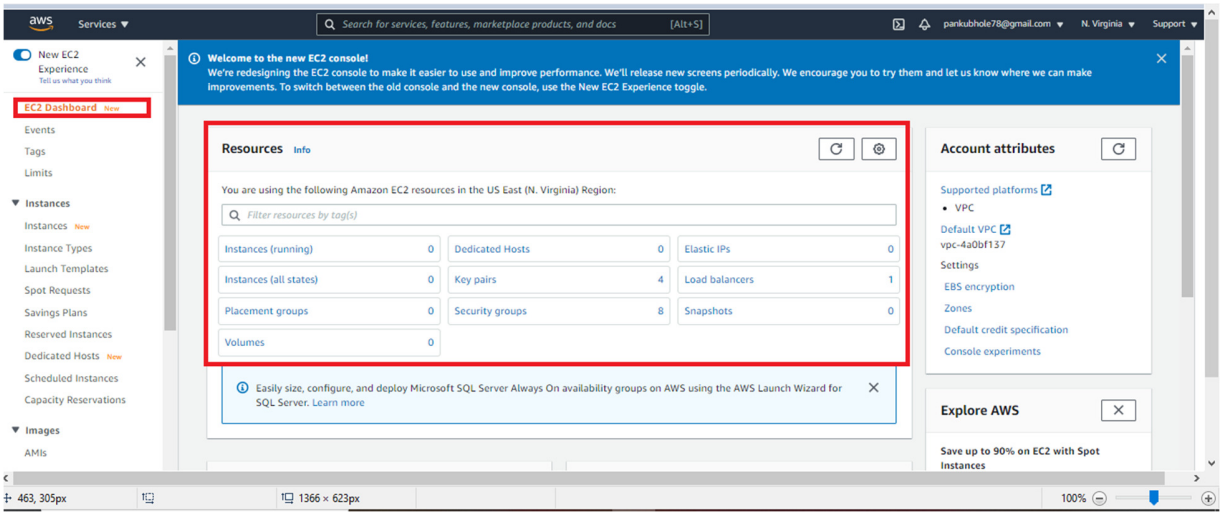
Steps

Creating an EC2 Instance

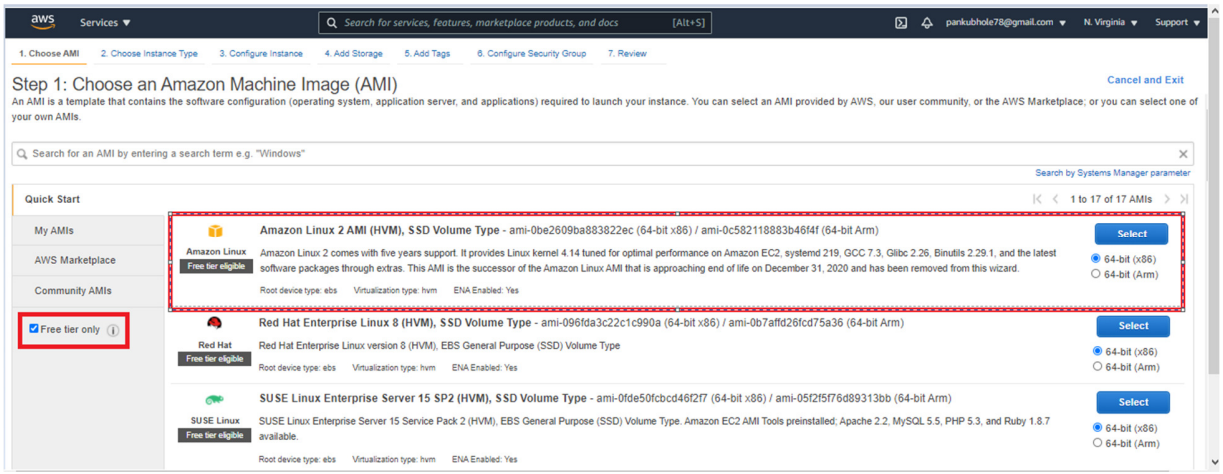
1. Click on AWS Management Console and set region N.Virginia



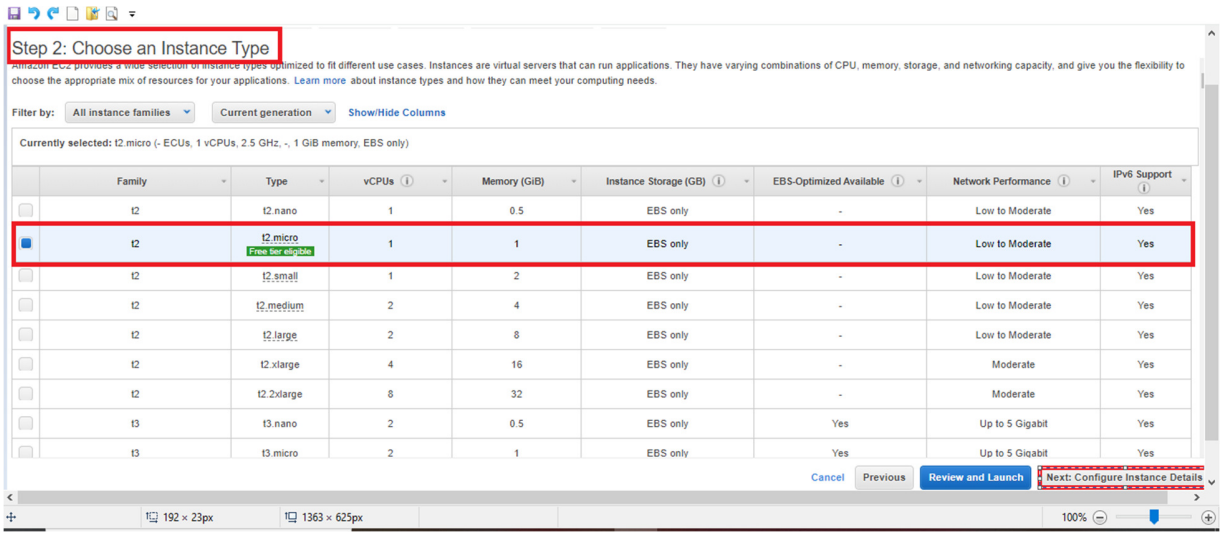
And select Ec2 from in compute and choose the EC2 Dashboard in this dashboard shown the resources working status.



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



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



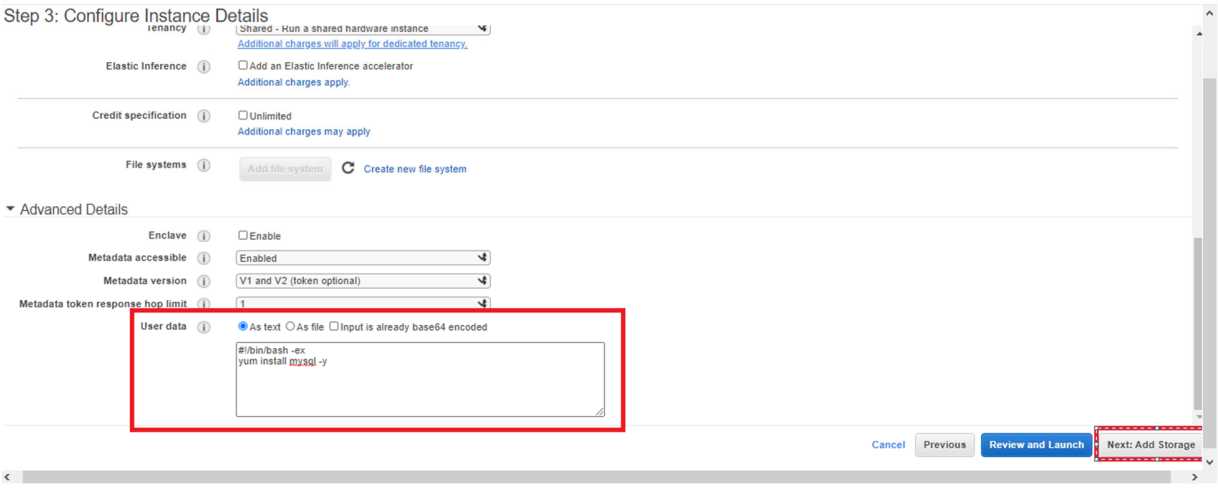
4. On the **Configure Instance Details** page:

- 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):

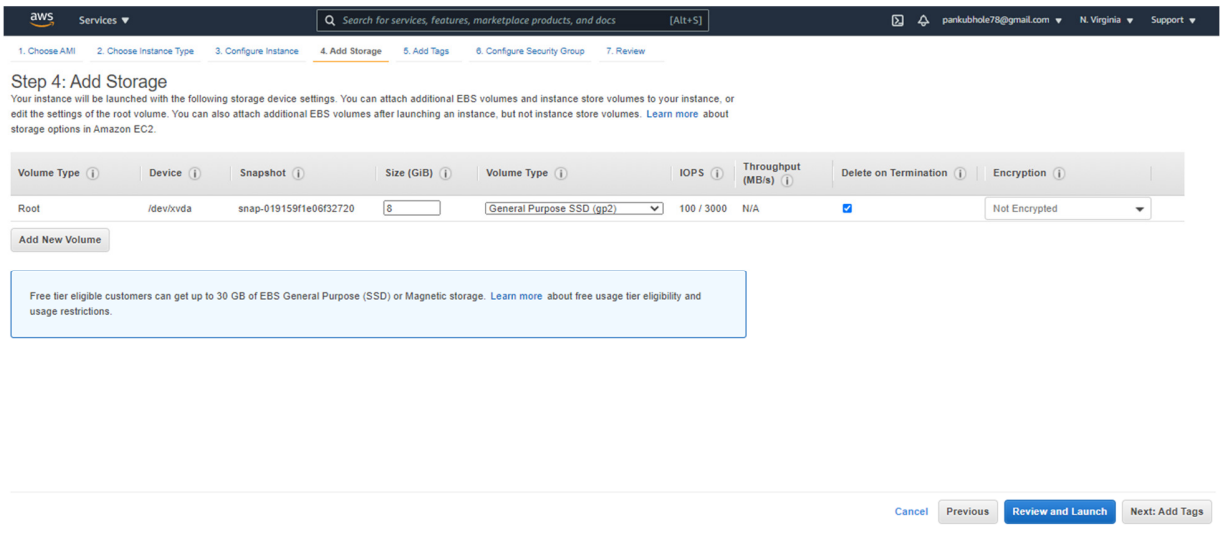
#!/bin/bash -ex

yum install mysql –y



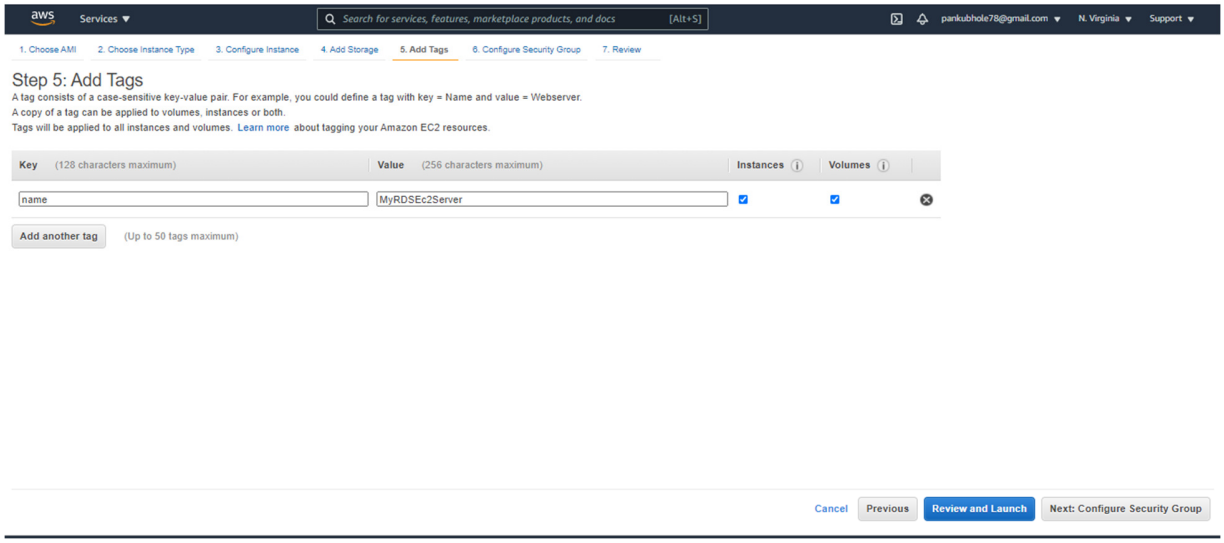
- Now click on Next Add storage Page.

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**

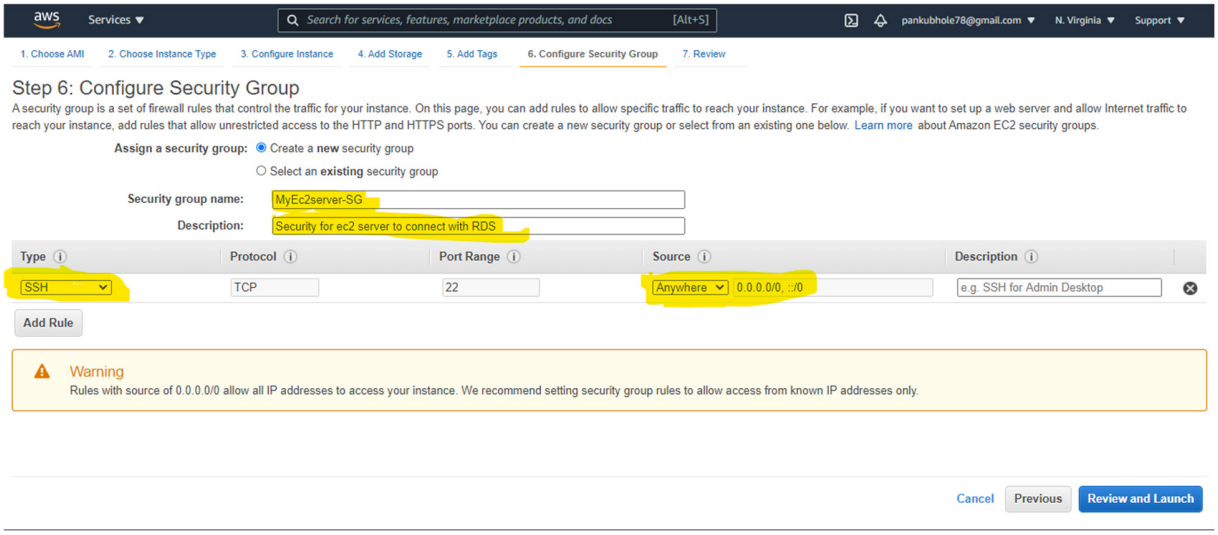


- Click on Configure Security group button

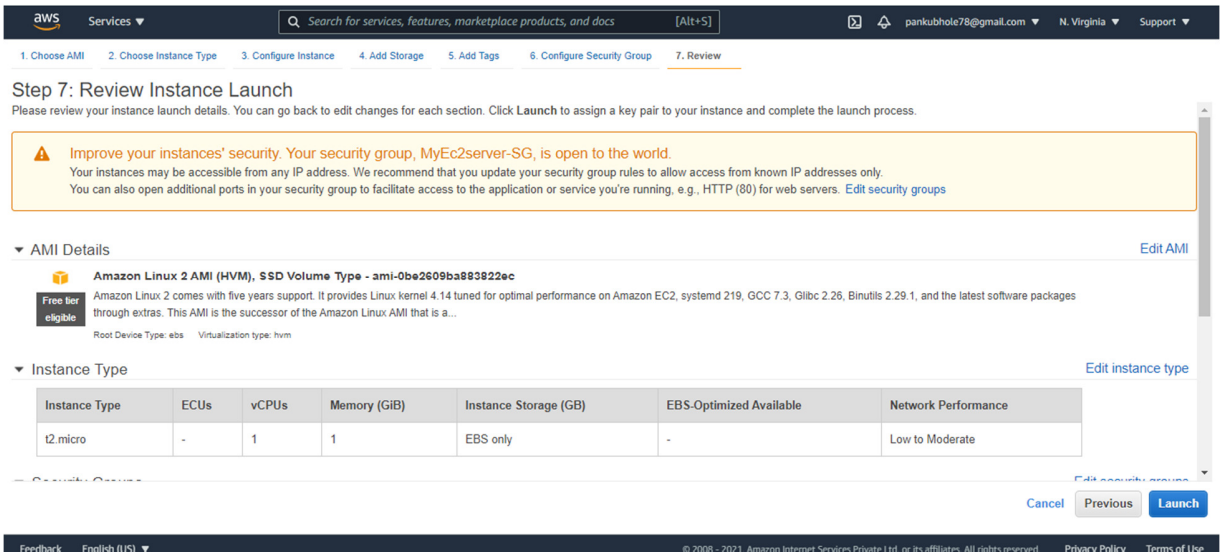
7. On the **Configure Security Group** page:

- **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:**

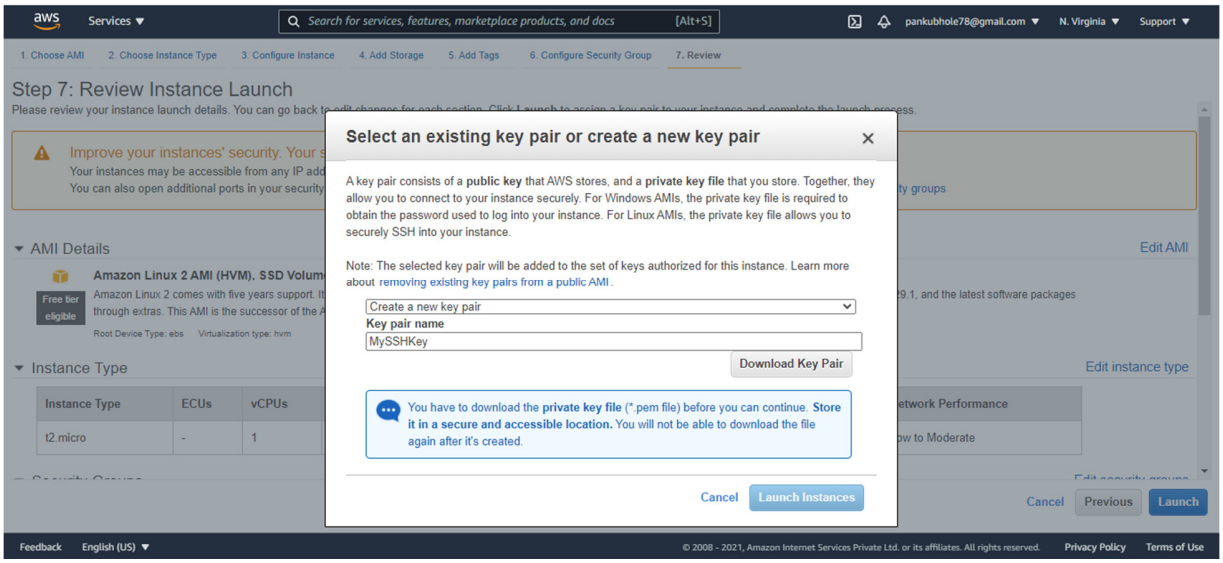
  - Choose Type:
  - **Source**: **Custom** Enter **0.0.0.0/0** in the textbox or select **anywhere**.



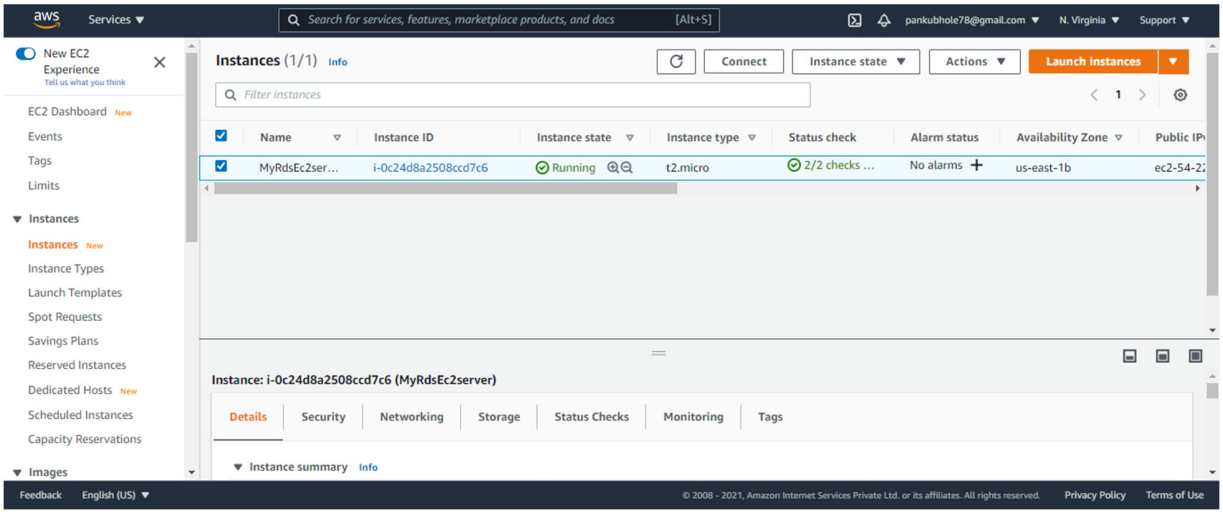
8. **Review and Launch:** Review all your settings and click on Launch button.



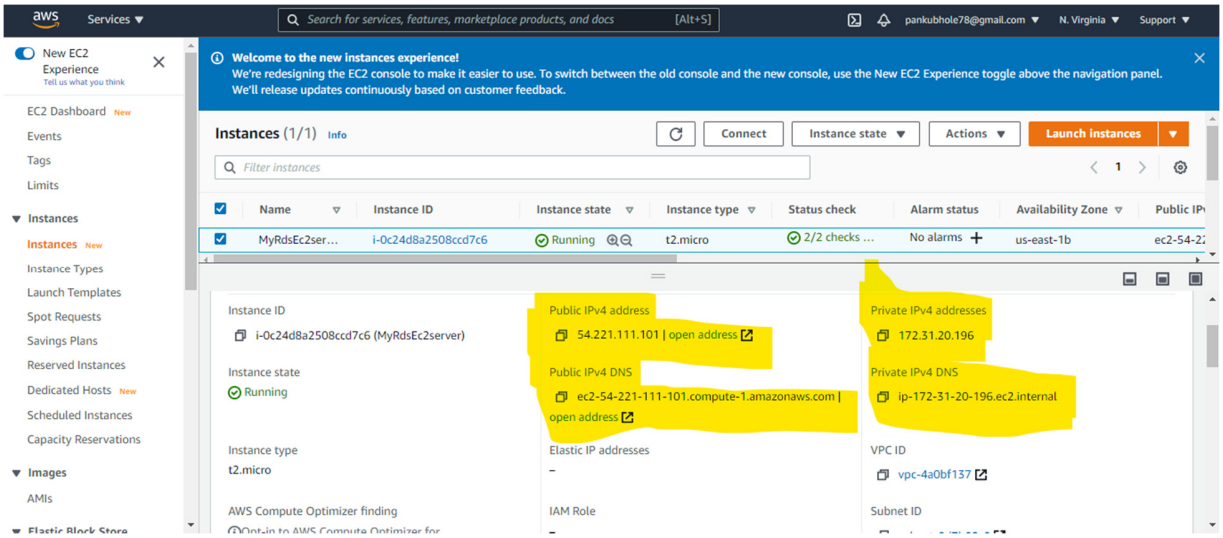
9. **Key Pair** - Select **Create a new key Pair** with name **MySSHKey,** click on **Launch instance button** and save it to your local machine.
10. Once the download is complete, click on **Download Key Pair button.**



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

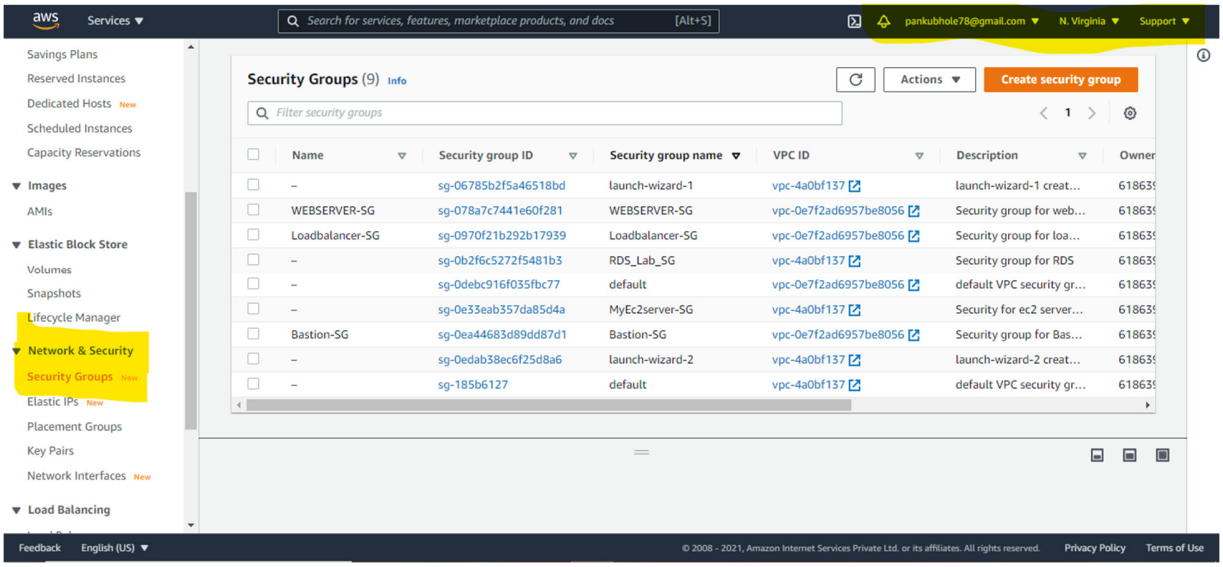


12. Note down **IPv4 Private IP address** in your text editor, navigate to the EC2 Dashboard and look in the instance details.

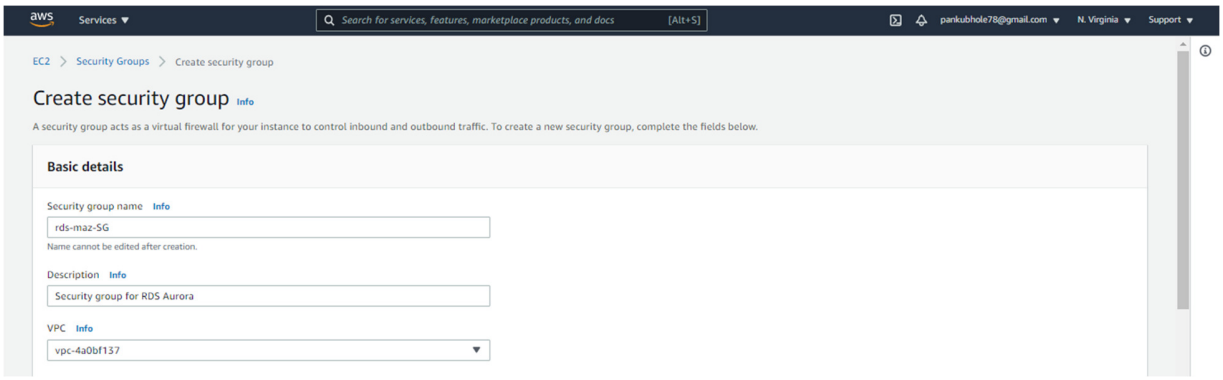


Create a Security Group for RDS instance

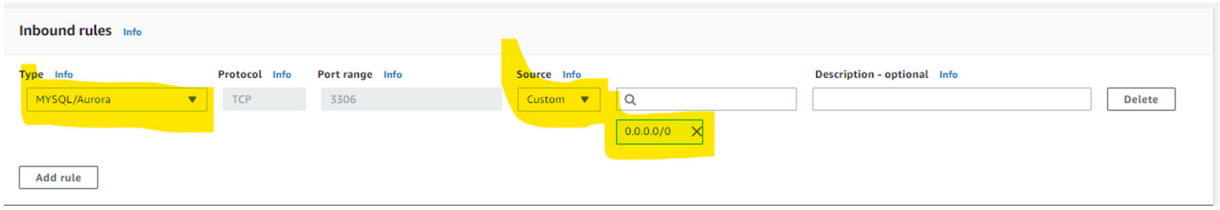
1. Make sure you are in the **N.Virginia** Region.
2. Navigate to EC2 by clicking on the menu available under the section.
3. On the left panel menu, select the security group under the Network **& Security** section.



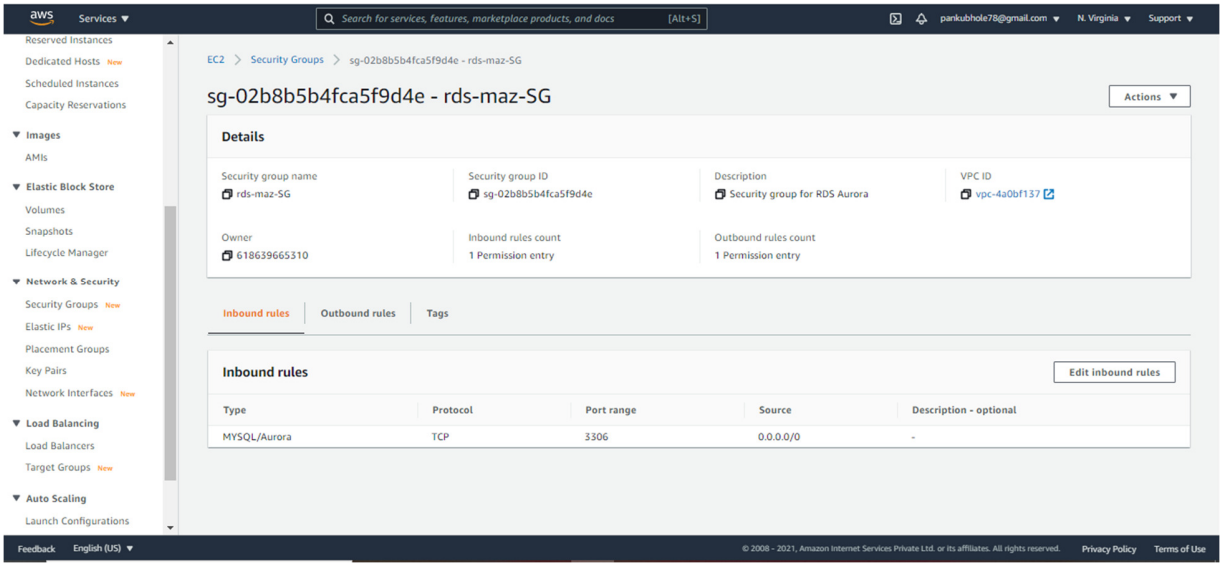
4. Click on the **Create Security Group** button.
5. We are going to create a Security group for RDS with 3306 port number enabled.
    - Security group name : Enter ***rds-maz-SG***
    - Description : Enter ***Security group for RDS Aurora***
    - VPC : Select Default VPC



- 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**

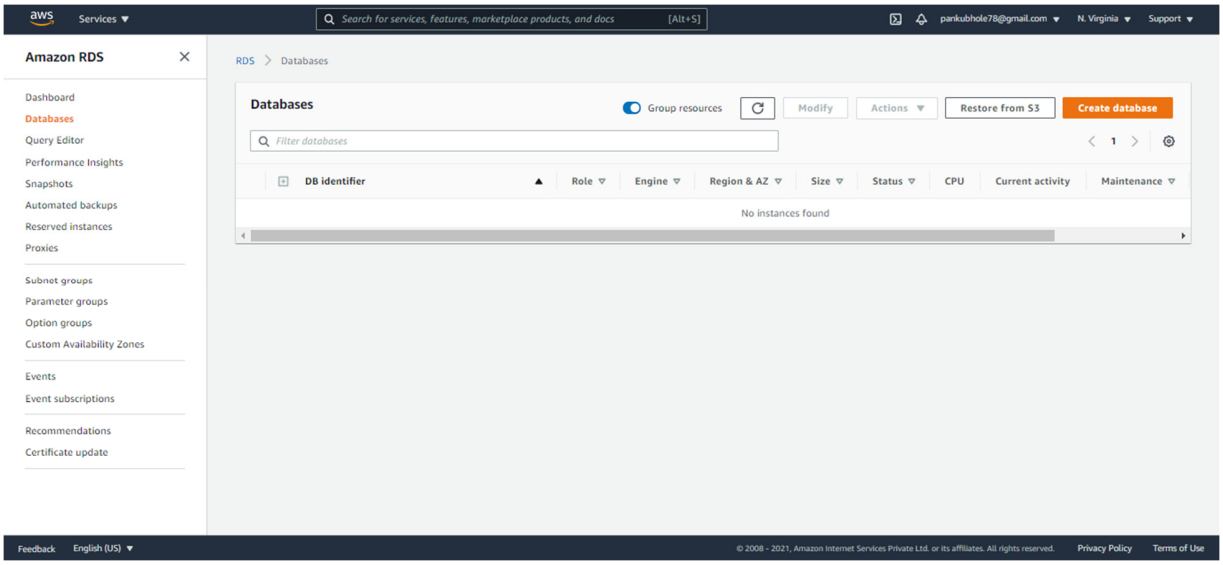


6. Leave everything as default and click on the **Create Security Group** button.

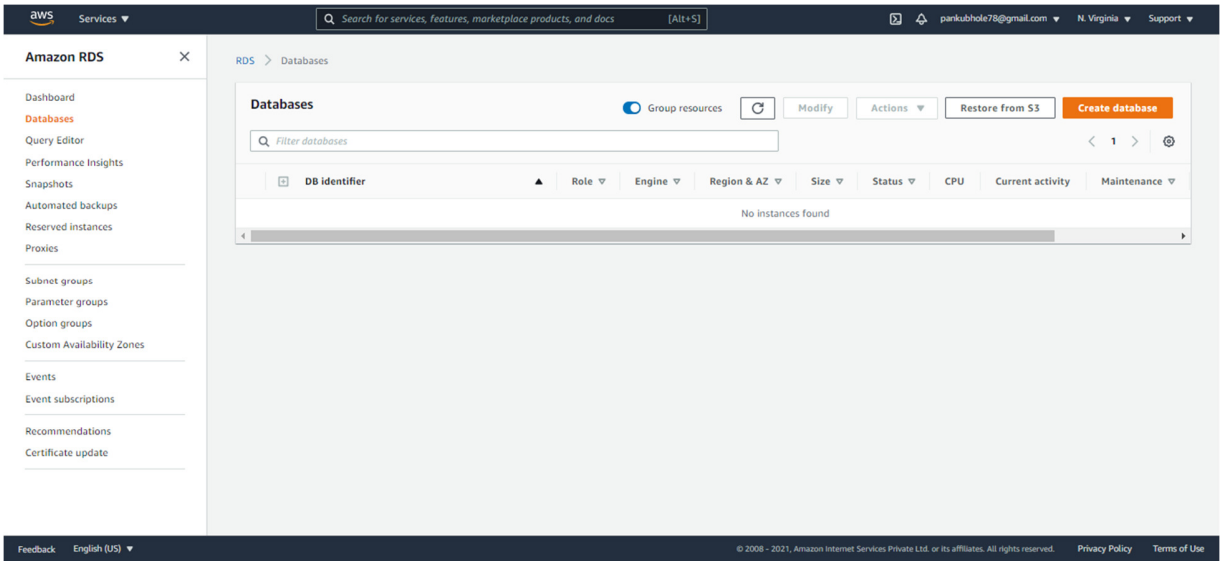


Create an Amazon Aurora database with Multi-AZ enabled

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.

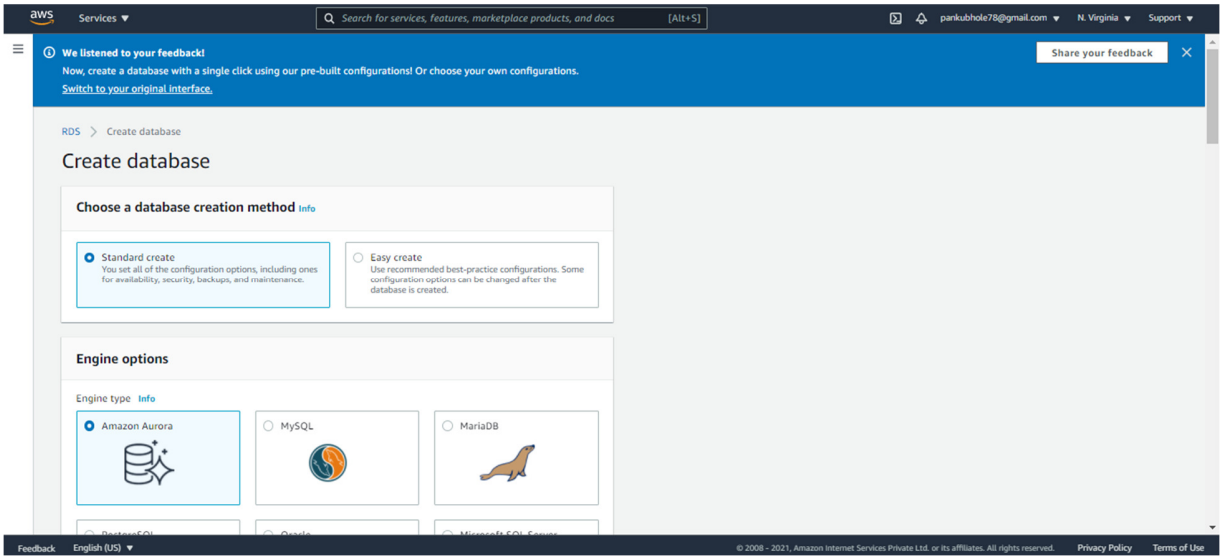


4. Click on **Create database**.
5. Ignore the warning mentioned below and continue with the next steps.



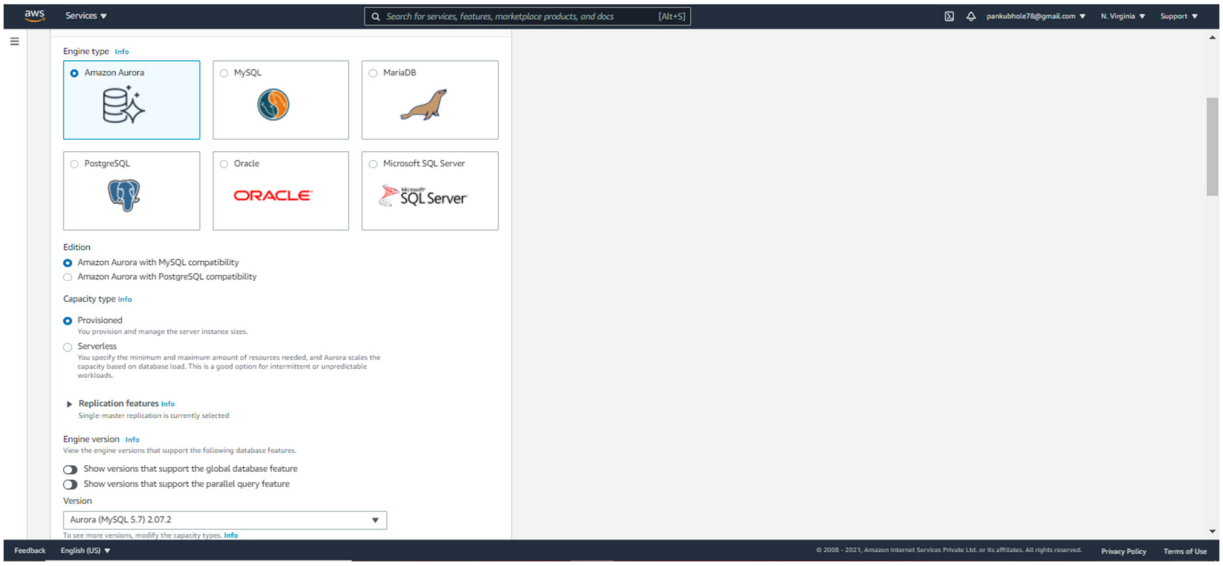
Note: If you are in the original interface, make sure to click on the **Switch to the new database creation flow** as shown below.

6. Next we'll configure the database on the **Create Database** Page
    - Choose a **Database Creation Method:**
        - Select **Standard Create**

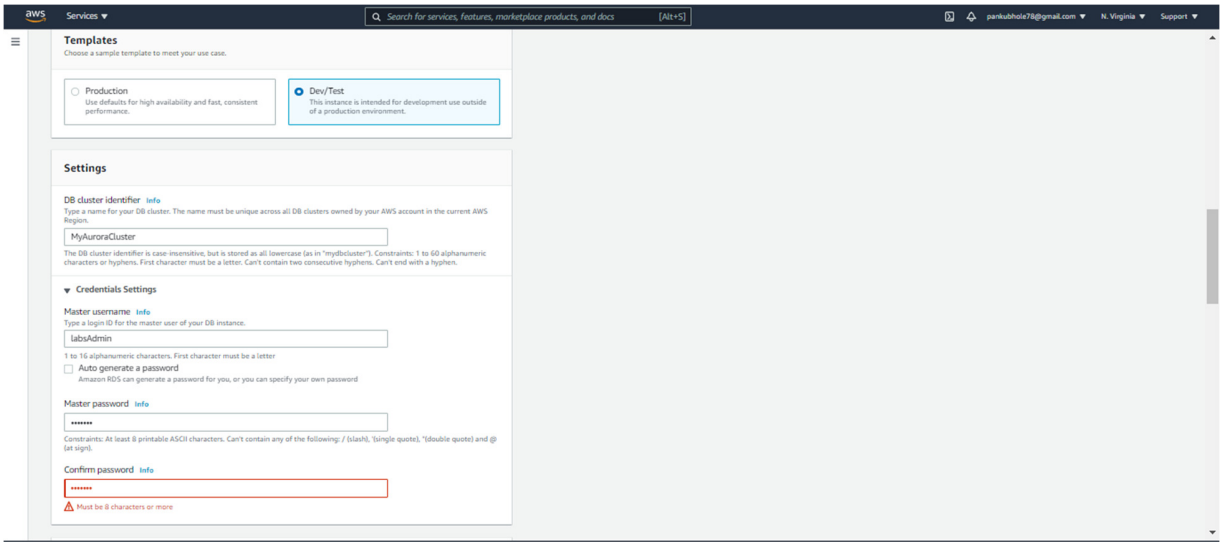


- In **Engine options:**

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

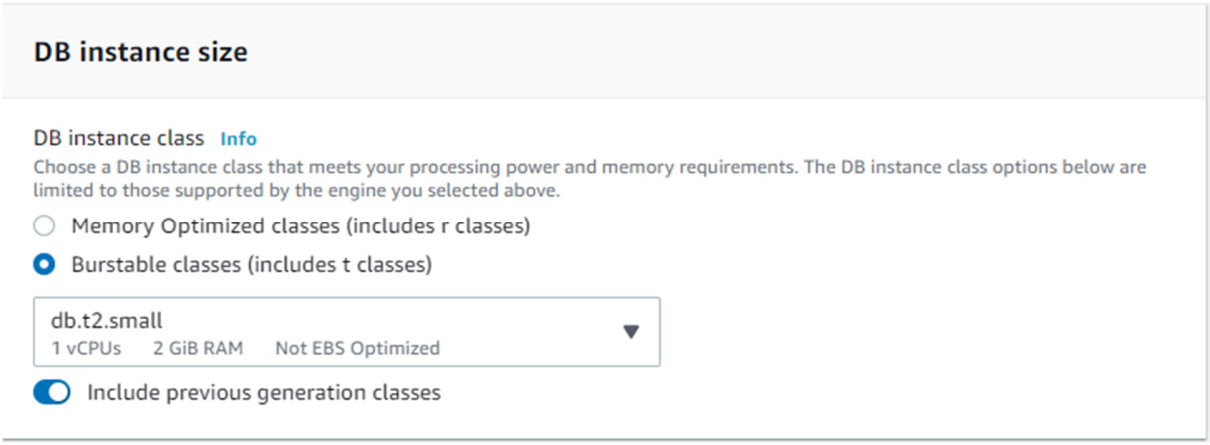


- Choose **Template: Dev/Test**
- Fill in the required details for the database (Aurora Cluster Settings)
  - **DB cluster identifier:** Specify cluster name *MyAuroraCluster*
  - Give the following details in the **credential settings**
    - **Master Username:** *labsAdmin*
    - **Master password:**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



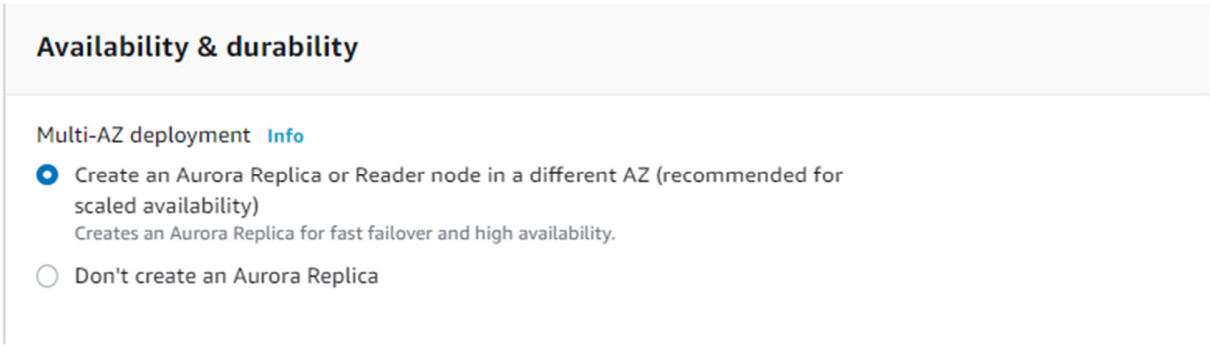
  - **Note:** This is the username and password used to log into your database. Please make note of them.
  - Choosing **DB instance size**
  - DB instance class: Choose **Burstable classes (includes t classes)**

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

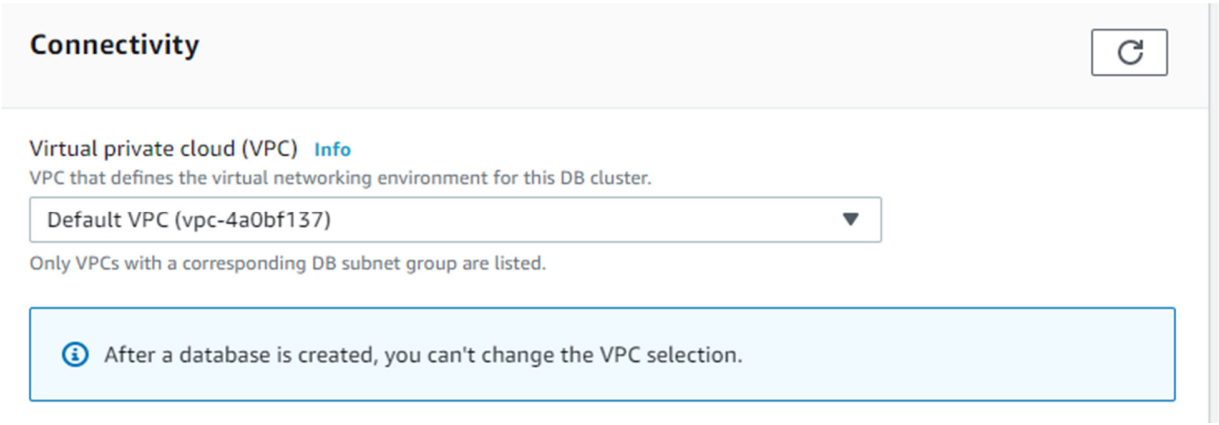


- **Note: Incase, db.t2.small is not available, we have added support for db.t3.small also, meaning you can go ahead with any of the two types.**

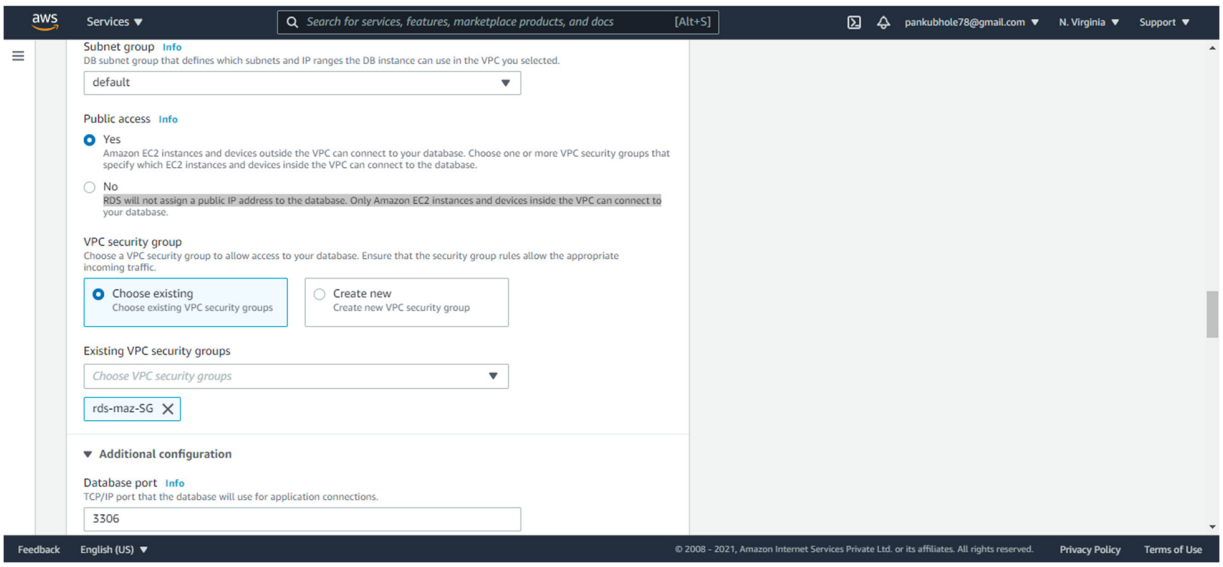
- **Availability and Durability**: Choose Multi-AZ deployment: **Create Aurora Replica or Reader node in a different AZ** as shown below:



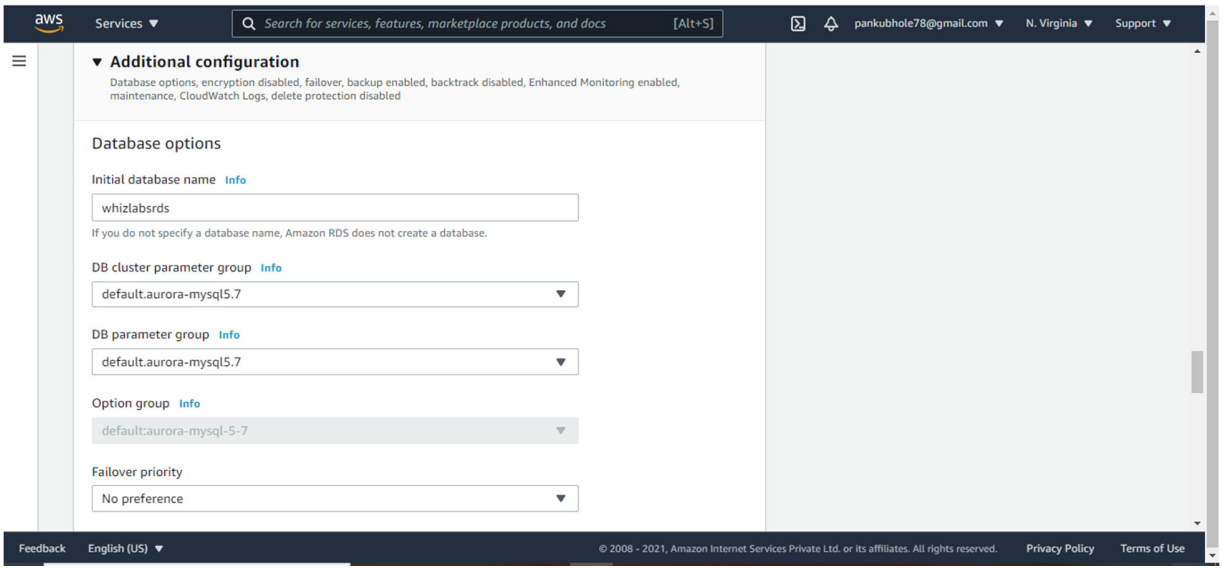
- **Connectivity**
  - Choose the Default VPC



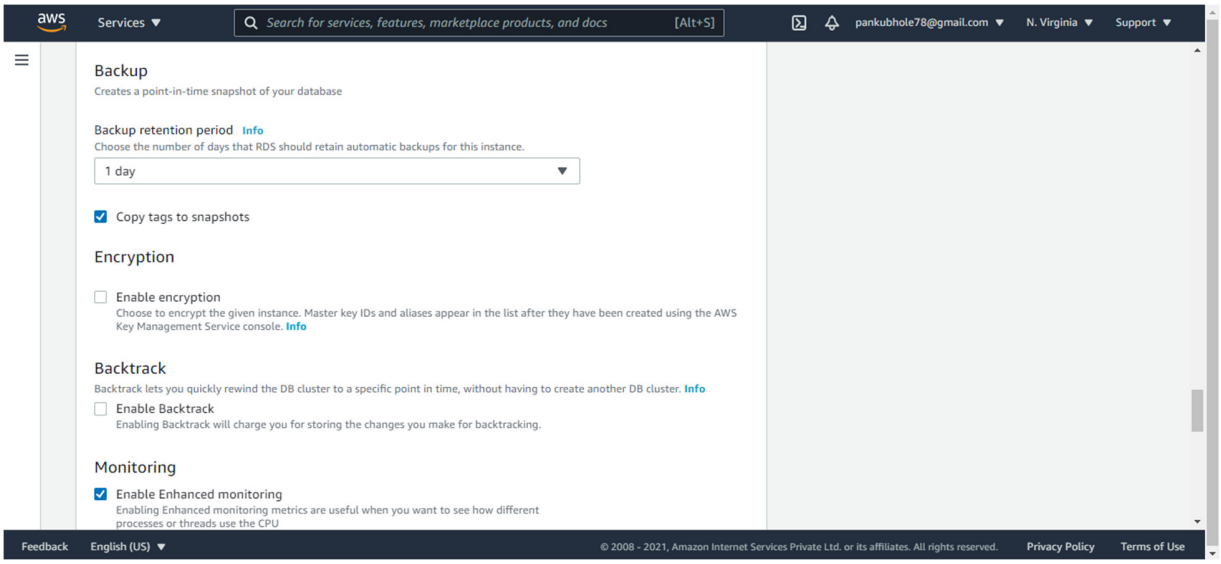
- Additional connectivity configuration
  - Subnet group : **Default**
  - Publicly accessible : **Yes**
  - Existing VPC security groups :
    - Remove the Default security group, which is selected by default.
    - Select **rds-maz-SG** for the dropdown.(This is the security group which you have created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



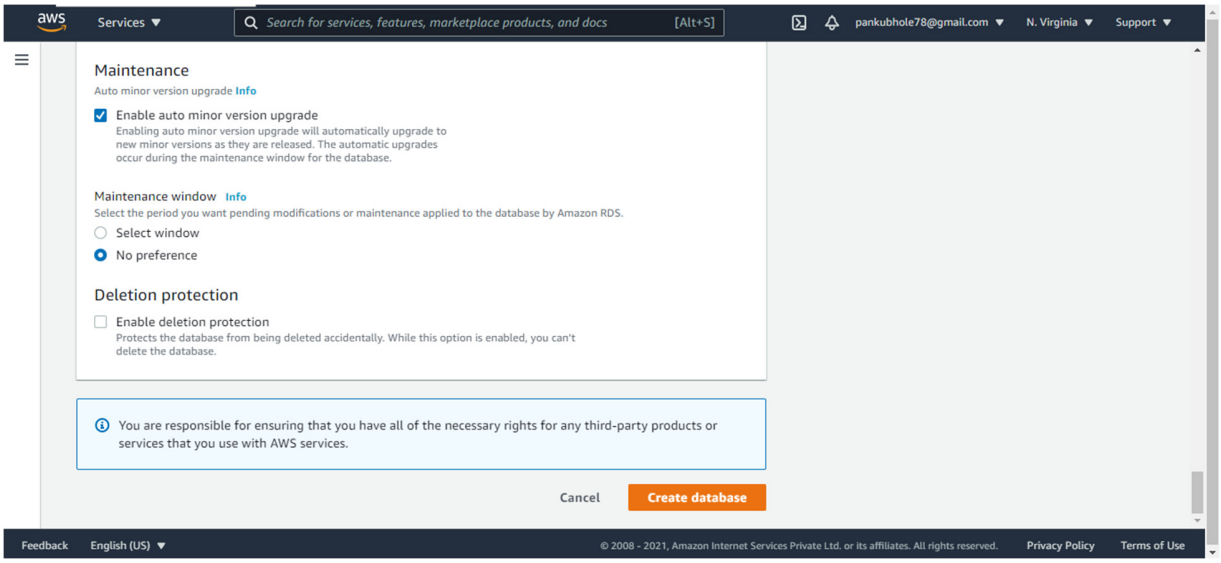
- **Additional configuration:**
  - Database options
    - Initial database name: **whizlabsrds**



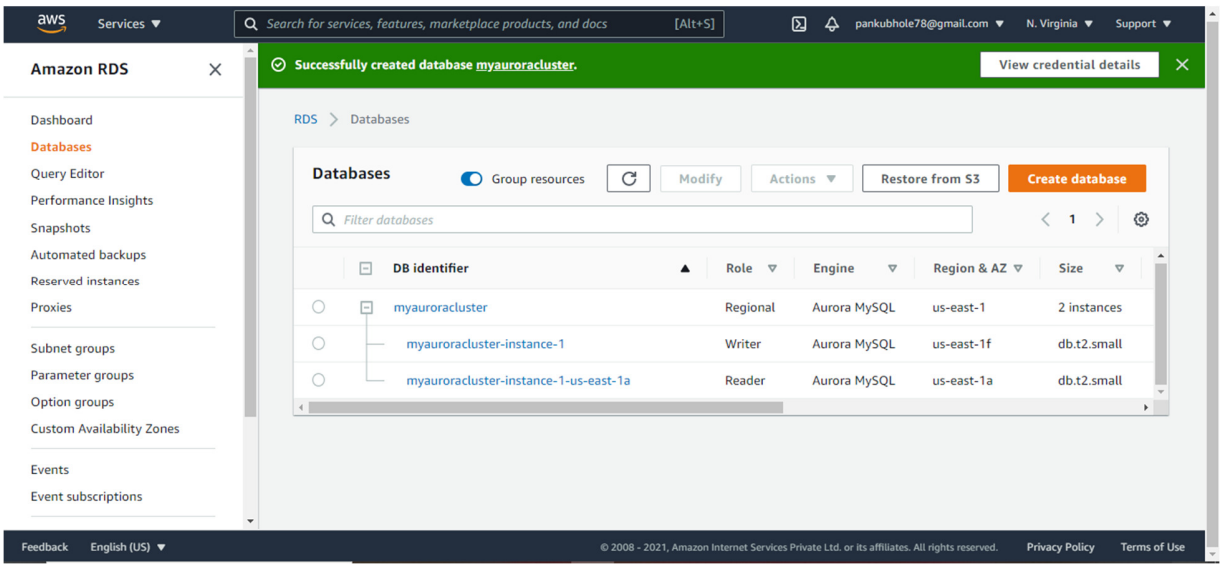
- **Encryption : Uncheck**



- Leave other settings **as default**



6. One 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:

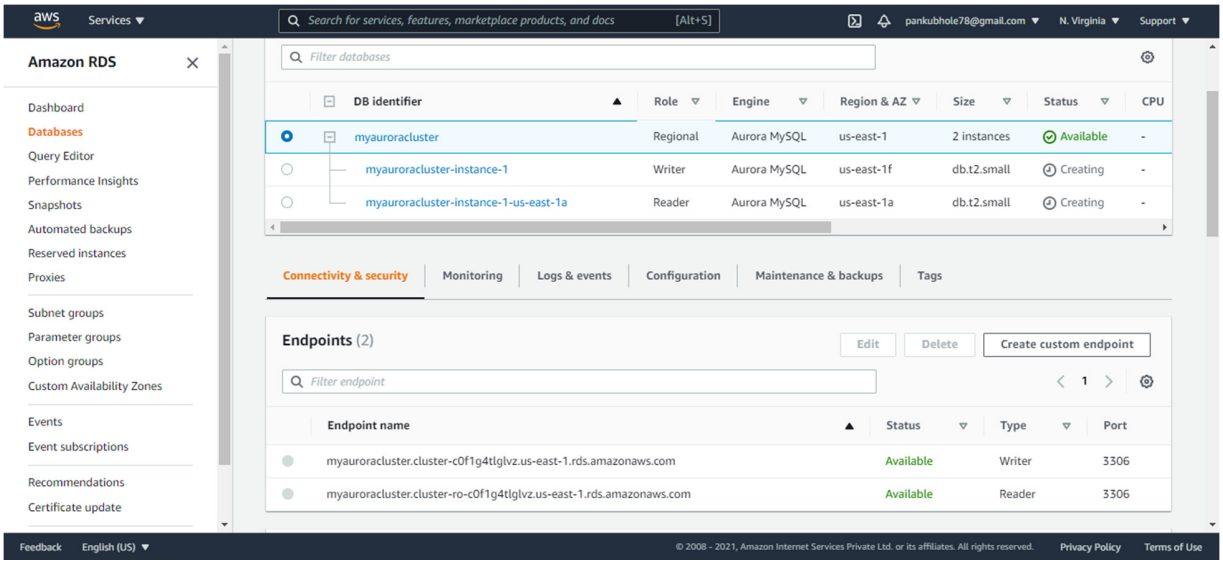


Similar to the screenshot, you should be able to see that our database launched in multiple AZs, namely **us-east-1d and us-east-1e**

Connecting to the Aurora (MySQL) database on RDS

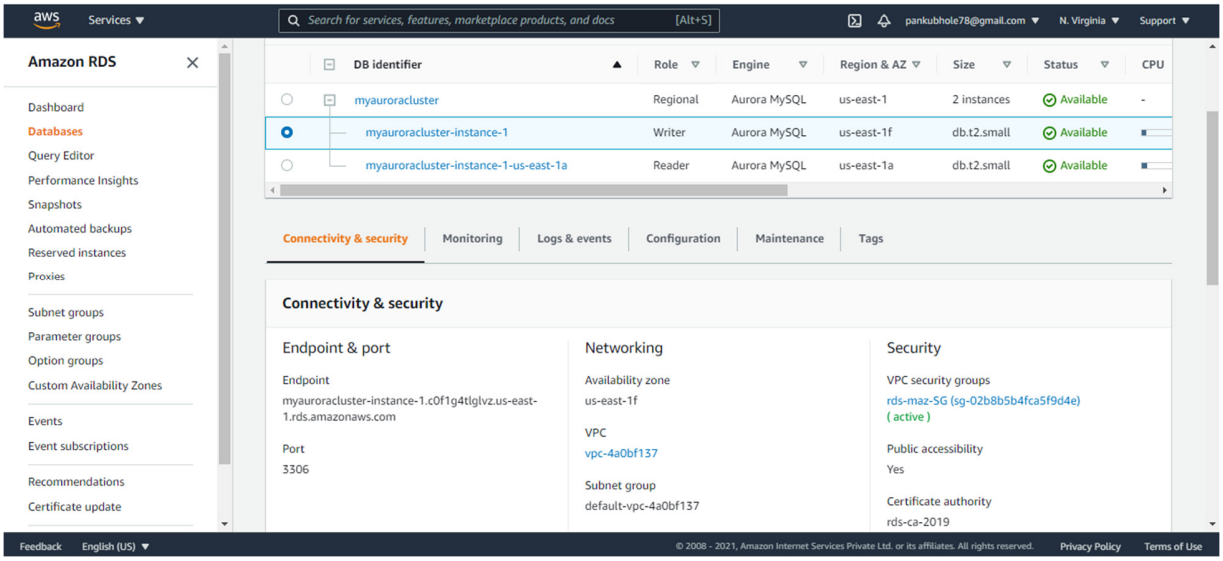
Now we have successfully launched Aurora RDS with Multi-AZ enabled. To connect to the new Aurora database, we need the **endpoint.**

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.

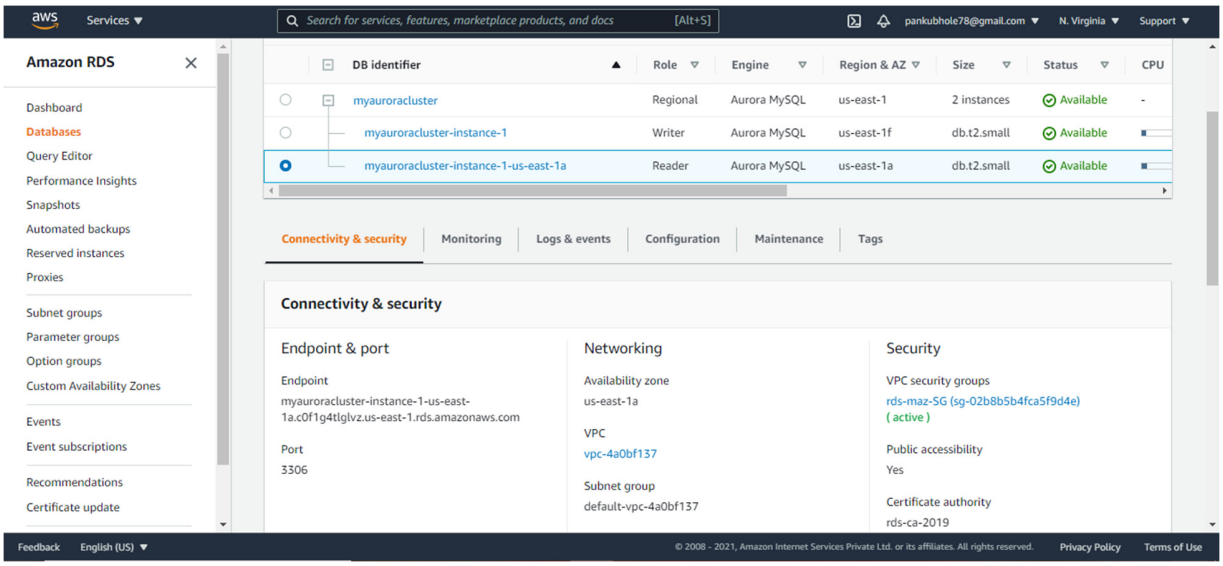


- The endpoints you see to be similar to these examples:

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



Reader: myauroracluster.cluster-ro-c0f1g4tlglvz.us-east-1.rds.amazonaws.com

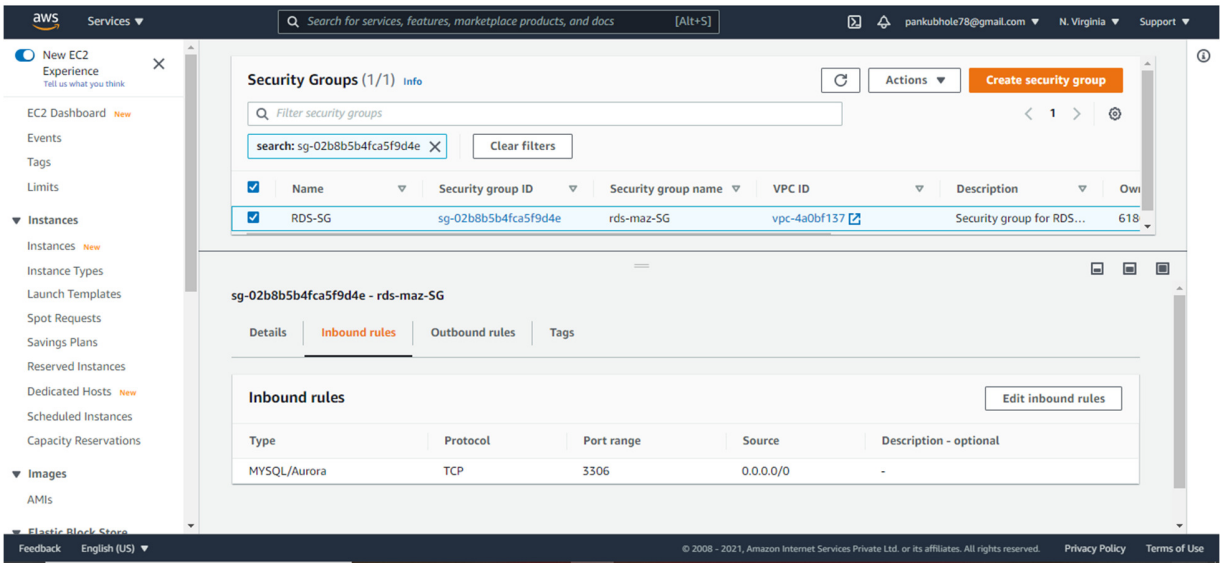


Note: Please carefully look at the role of the DB instance (**Reader vs Master (Writer)**) and their respective availability zones. (Here **us-east-1d** and **us-east-1e)**

2. 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:

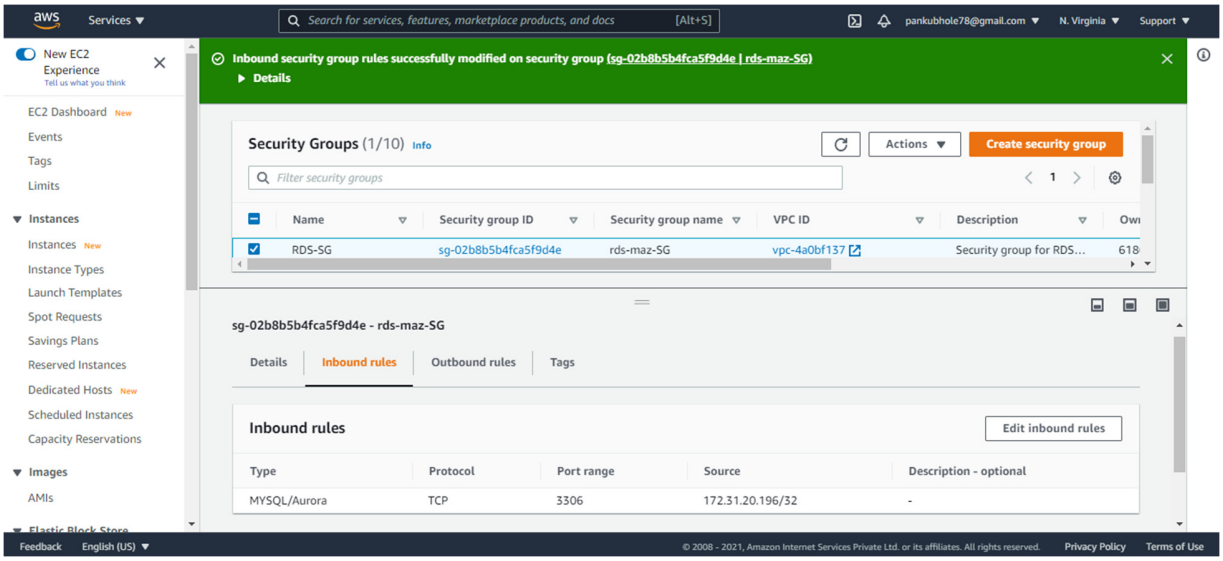
Connecting the EC2 Server to RDS:

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:



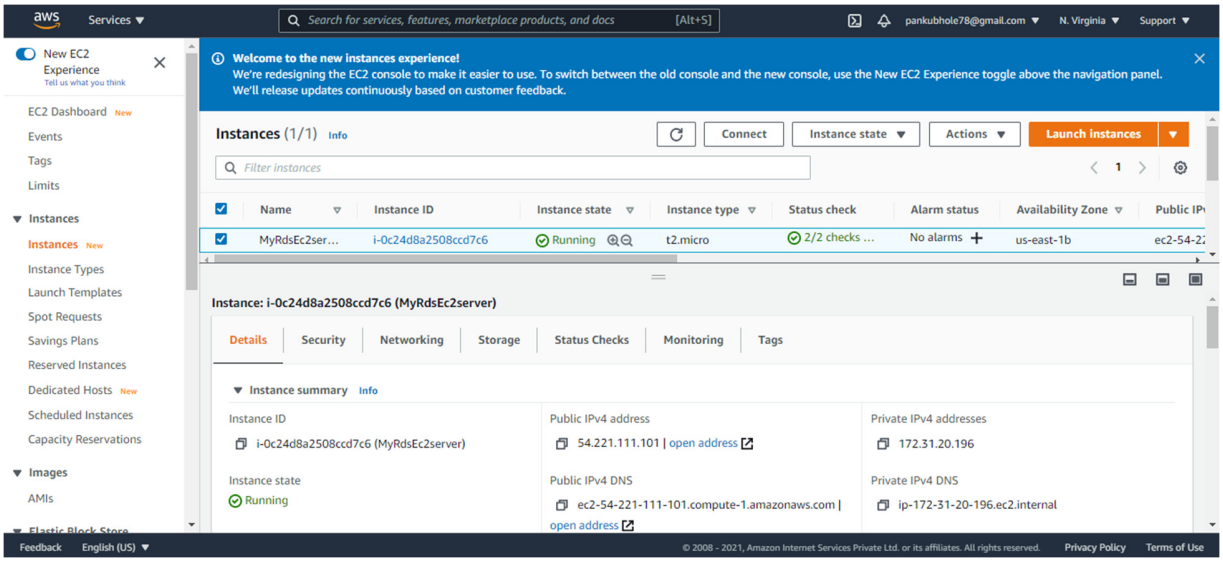
4. It will open the Security Group page. Click on **InBound.**

- 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:

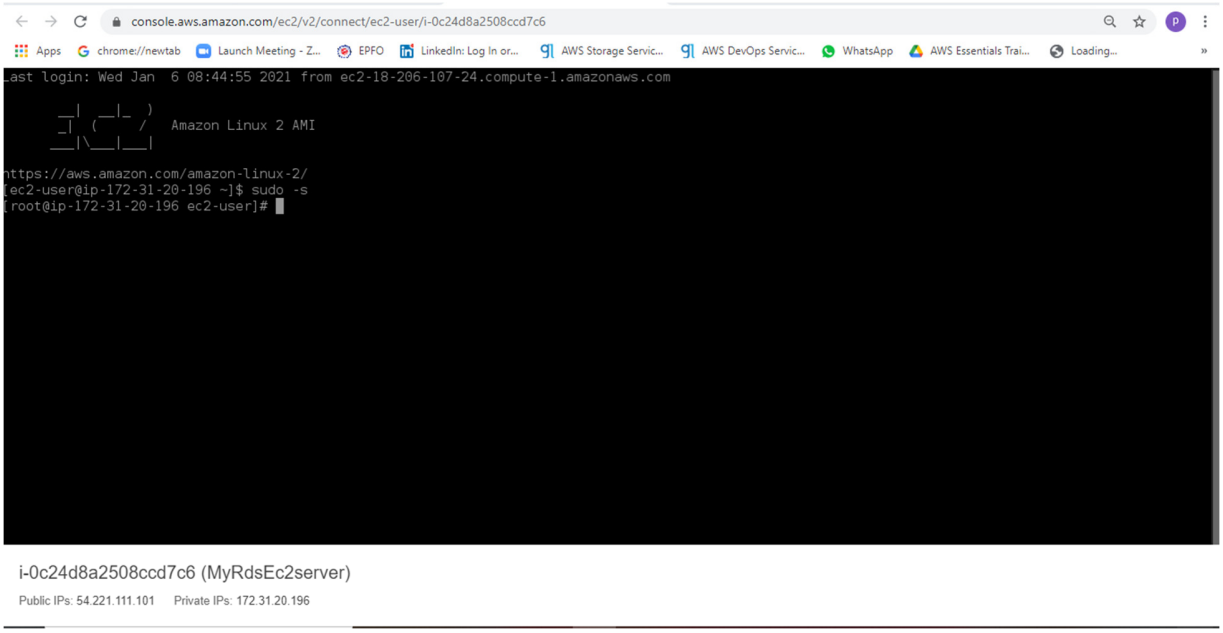


Execute Database Operations via SSH

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



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**

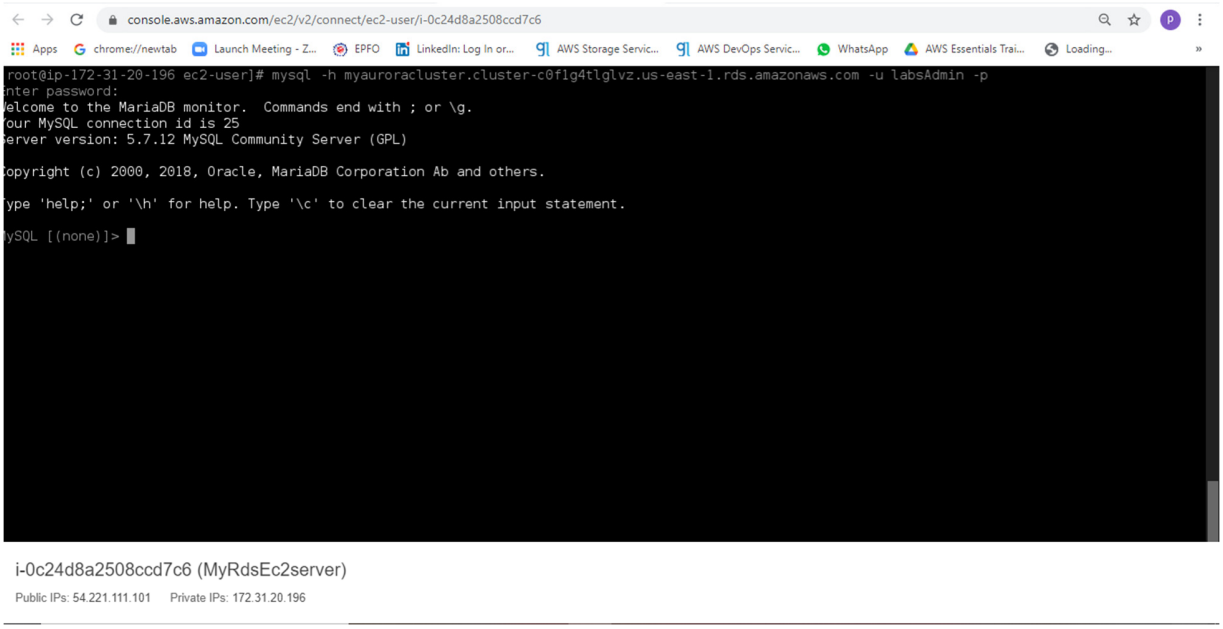


4. Log into the RDS instance using the below command:

- **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:



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:

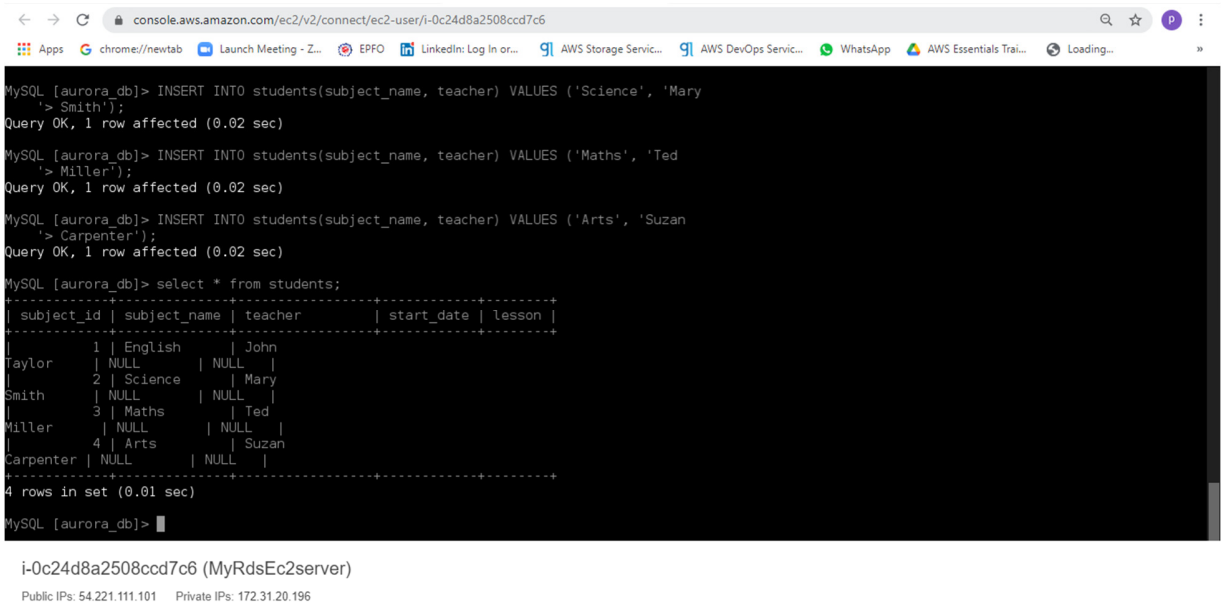
- **CREATE TABLE students (subject_id INT AUTO_INCREMENT, subject_name VARCHAR(255) NOT NULL,teacher VARCHAR(255),start_date DATE, lesson TEXT,PRIMARY KEY (subject_id));**

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;**



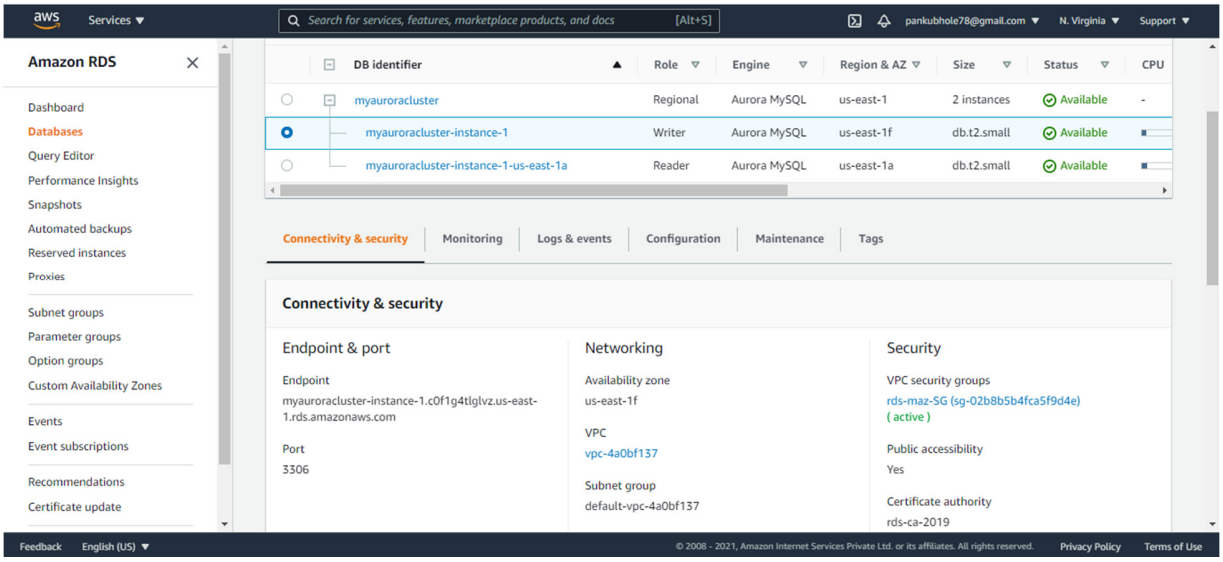
i-0c24d8a2508ccd7c6 (MyRdsEc2server)

Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

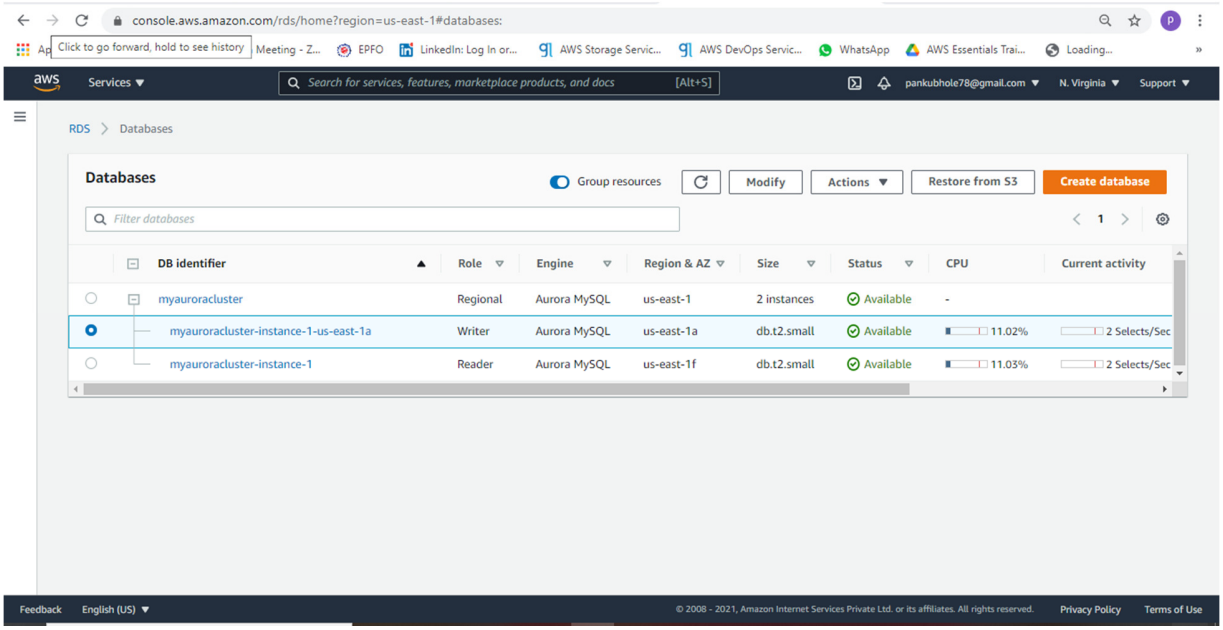
10. Exit from mysql console use the below command:
    - **exit**

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.



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



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

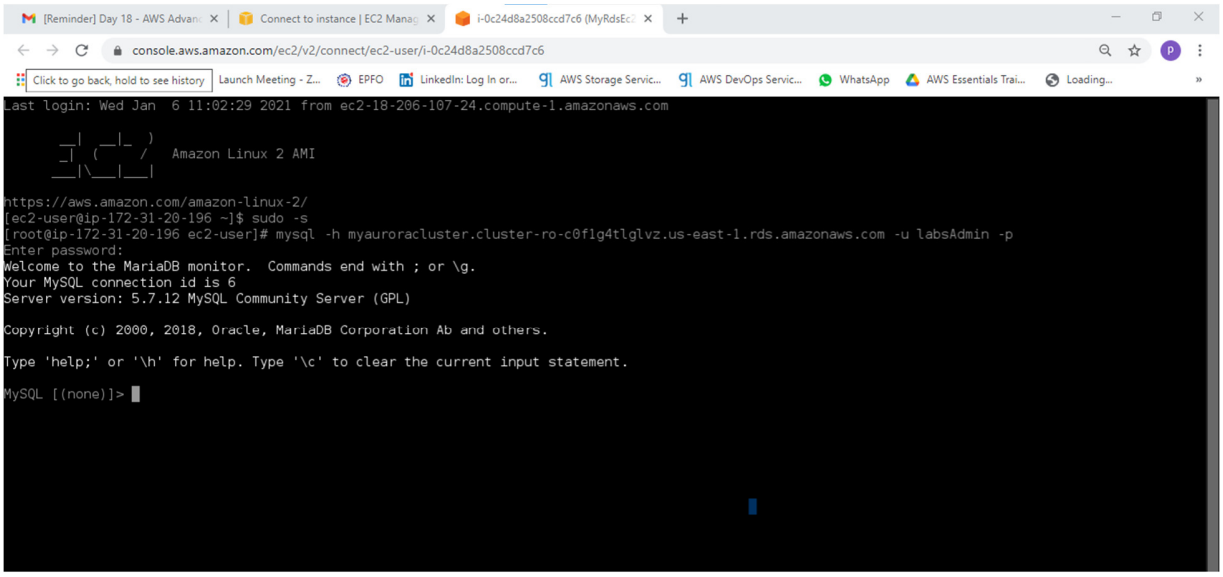
Testing the Failover Condition

1. Now connect to RDS with new Master endpoint
    - 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]**

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

  Password**: labs123**



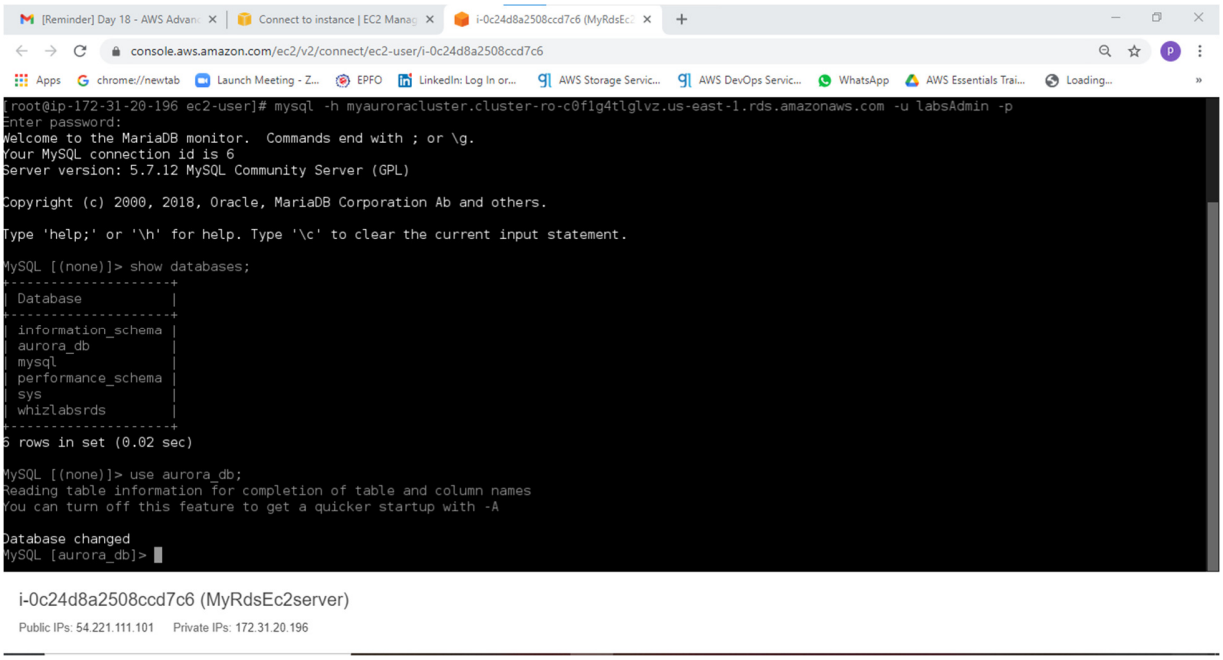
i-0c24d8a2508ccd7c6 (MyRdsEc2server)

Public IPs: 54.221.111.101 Private IPs: 172.31.20.196

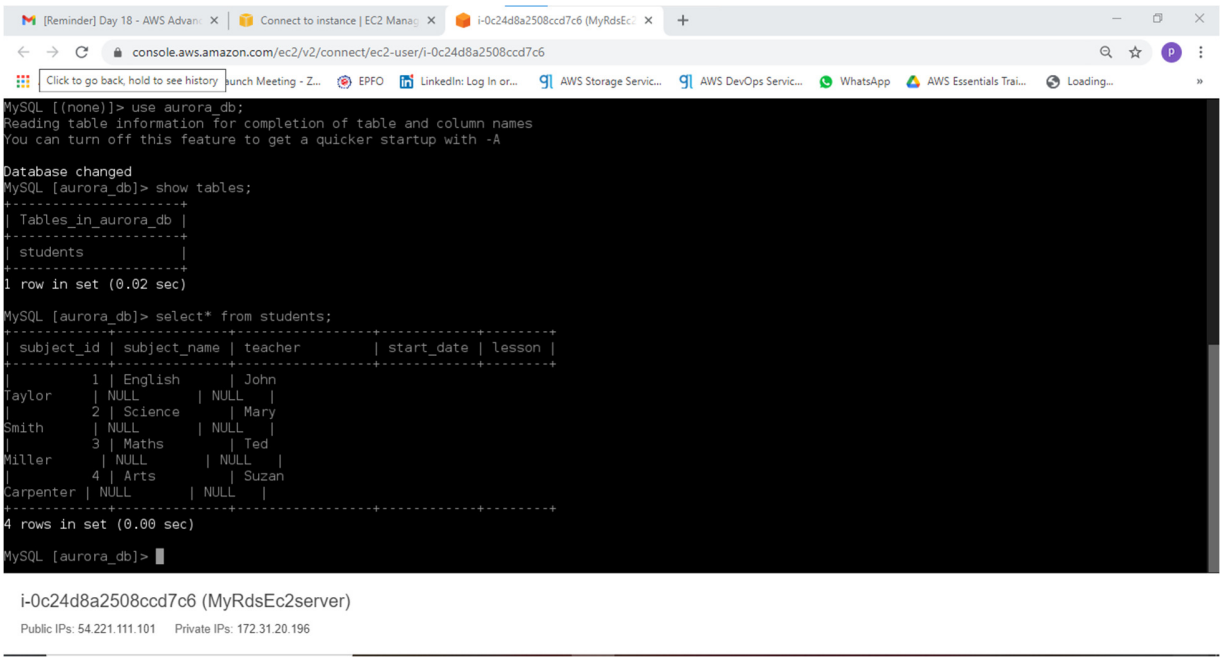
2. You will be **able to Log** into MySQL and check for the database and table created in the master DB instance before the failover.

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

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



3. Now check the existence of table named **students and data** (that we created earlier in the lab):
    - **show tables;**
    - **select * from students;**



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