PART A

(PART A: TO BE REFFERED BY STUDENTS)

Experiment No. 6

A.1 Aim:

To create database using RDS.

A.2 Prerequisite:

A.3 Outcome:

After successful completion of this course, students will be able to, install and appreciate security features and user management.

A.4 Theory:

What is Amazon Relational Database Service (Amazon RDS)?

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

Overview of Amazon RDS:

Why do you want a managed relational database service? Because Amazon RDS takes over many of the difficult or tedious management tasks of a relational database:

When you buy a server, you get CPU, memory, storage, and IOPS, all bundled together. With Amazon RDS, these are split apart so that you can scale them independently. If you need more CPU, less IOPS, or more storage, you can easily allocate them.

Amazon RDS manages backups, software patching, automatic failure detection, and recovery.

To deliver a managed service experience, Amazon RDS doesn't provide shell access to DB instances, and it restricts access to certain system procedures and tables that require advanced privileges.

You can have automated backups performed when you need them, or manually create your own backup snapshot. You can use these backups to restore a database. The Amazon RDS restore process works reliably and efficiently.

You can get high availability with a primary instance and a synchronous secondary instance that you can fail over to when problems occur. You can also use MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.

You can use the database products you are already familiar with: MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server.

In addition to the security in your database package, you can help control who can access your RDS databases by using AWS Identity and Access Management (IAM) to define users and permissions. You can also help protect your databases by putting them in a virtual private cloud.

Amazon RDS Interfaces:

AWS Management Console

The AWS Management Console is a simple web-based user interface. You can manage your DB instances from the console with no programming required. To access the Amazon RDS console, sign in to the AWS Management Console and open the Amazon RDS console.

Command Line Interface

You can use the AWS Command Line Interface (AWS CLI) to access the Amazon RDS API interactively. To install the AWS CLI, see Installing the AWS Command Line Interface. To begin using the AWS CLI for RDS, see AWS Command Line Interface Reference for Amazon RDS.

Programming with Amazon RDS

If you are a developer, you can access the Amazon RDS programmatically. For more information, see Amazon RDS Application Programming Interface (API) Reference.

For application development, we recommend that you use one of the AWS Software Development Kits (SDKs). The AWS SDKs handle low-level details such as authentication, retry logic, and error handling, so that you can focus on your application logic. AWS SDKs are available for a wide variety of languages. AWS also provides libraries, sample code, tutorials, and other resources to help you get started more easily. For more information, see Sample Code & Libraries.

PART B

(PART B: TO BE COMPLETED BY STUDENTS)

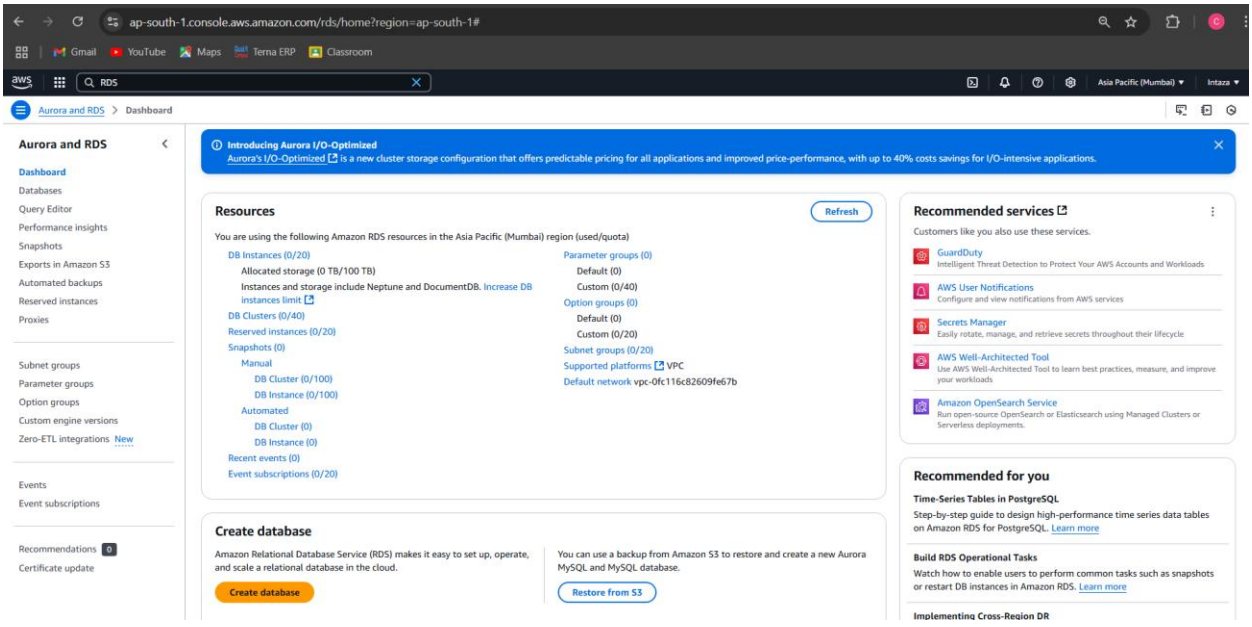
(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

Roll. No.B48	Name: Aryan Unhale
Class:TE B COMPS	Batch:B3
Date of Experiment:27/3/25	Date of Submission:29/3/25
Grade:	

B.1 Software Code written by student:

B.2 Input and Output:

Step 1: Open the Amazon RDS Service(search for **RDS** in the search bar and Click on **Aurora and RDS** to open the service.)

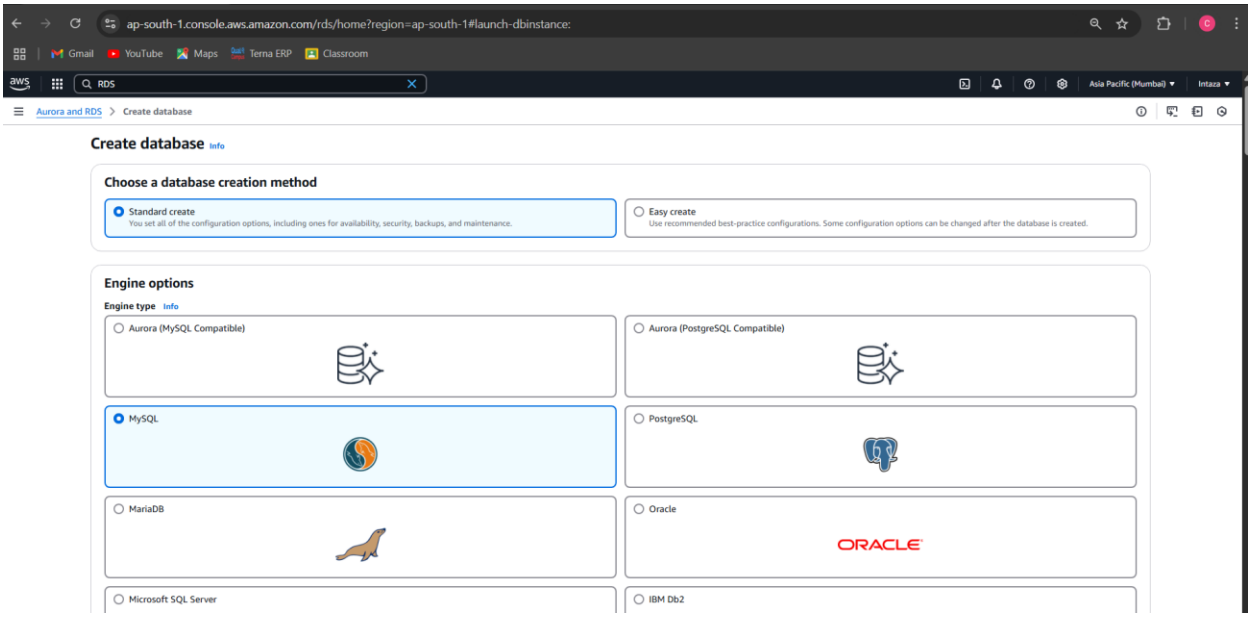


Step 2: Scroll down and Click on **Create database**.

- Choose **Standard create** for full configuration options.

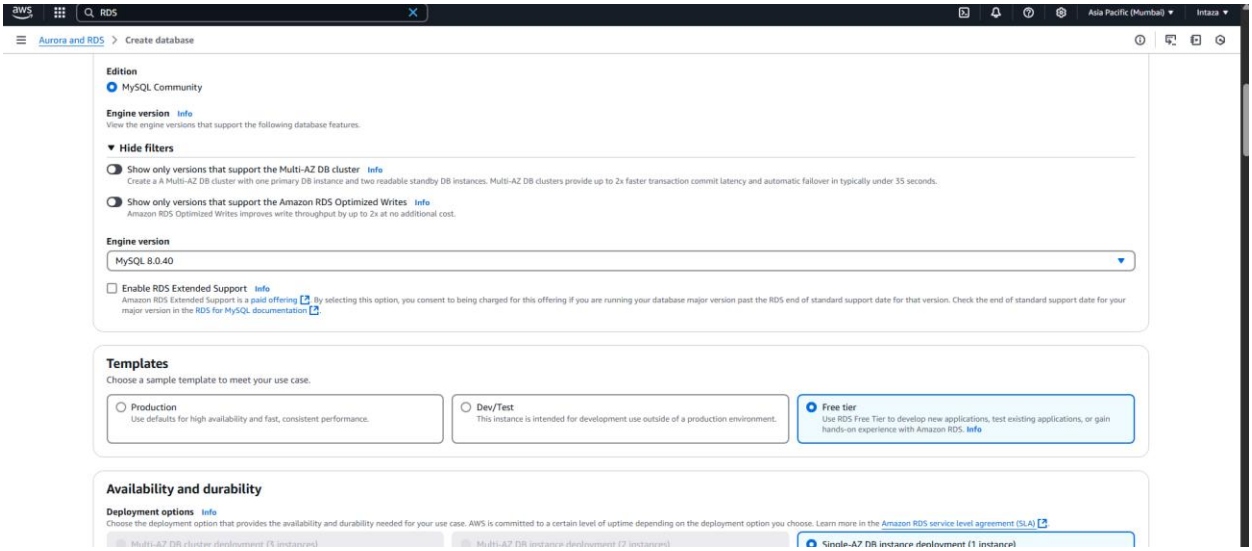
Step 3: Select Database Engine

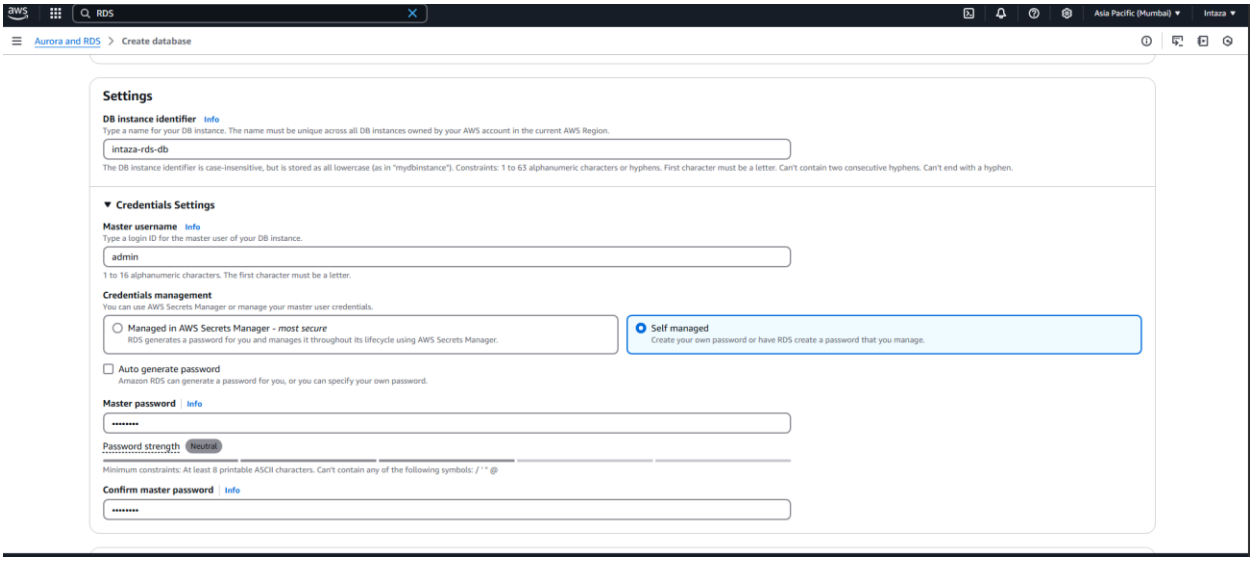
- Under **Engine options**, select a relational database (e.g., **MySQL**).
- Choose the **Engine Version** (default recommended).



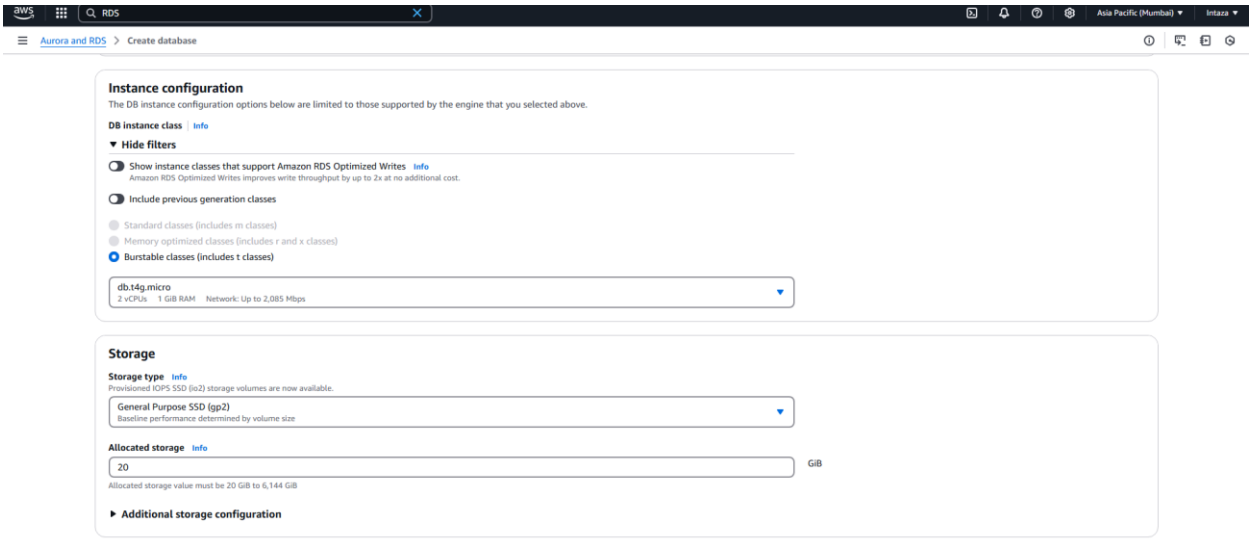
Step 4: Configure Database Settings

- **Templates**: Select **Free Tier** (if eligible) to avoid costs.
- **DB instance identifier**: Enter a unique name (e.g., mydb-instance).
- **Master username**: Enter admin (or a custom username).
- **Master password**: Set a secure password.



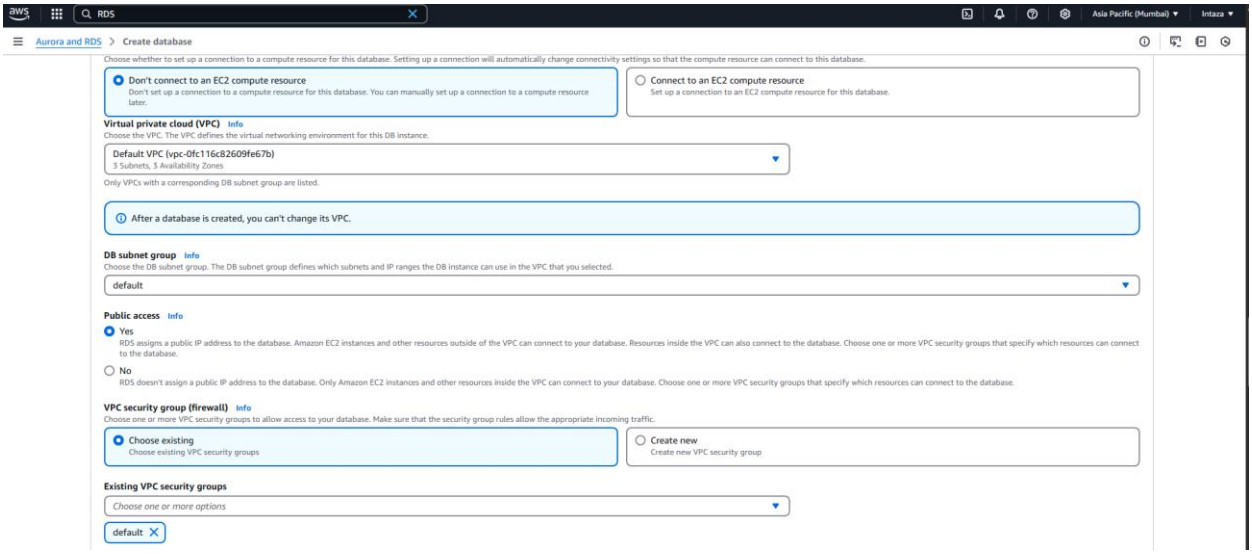


Instance Specifications (default for Free Tier).



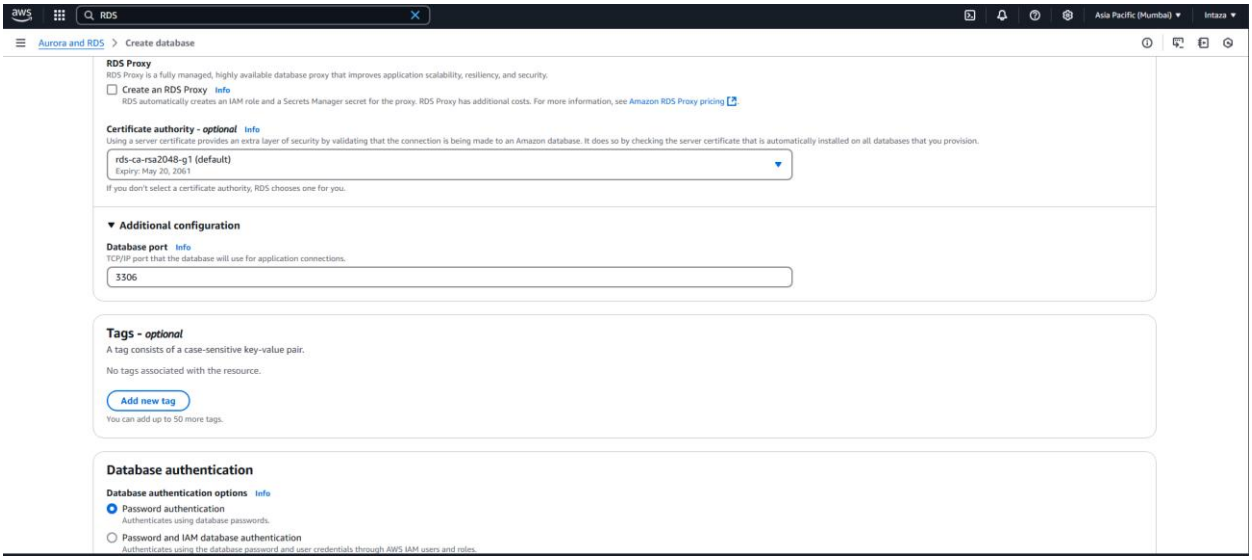
Step 5: Configure Connectivity

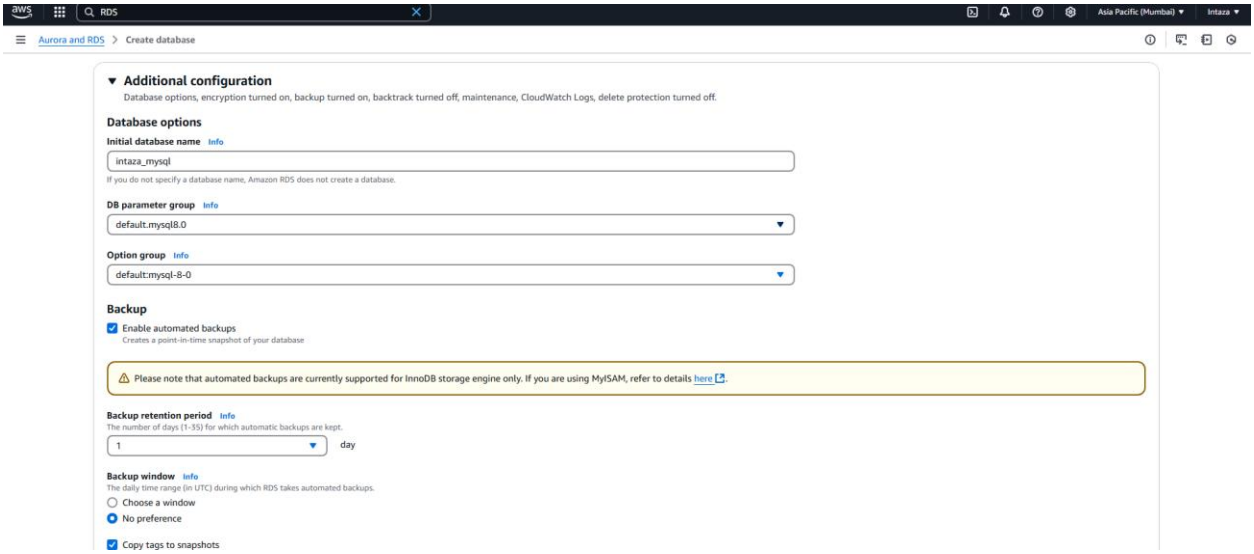
- **Virtual Private Cloud (VPC)**: Use the default VPC.
- **Public access**: Select **Yes** (for lab purposes only).



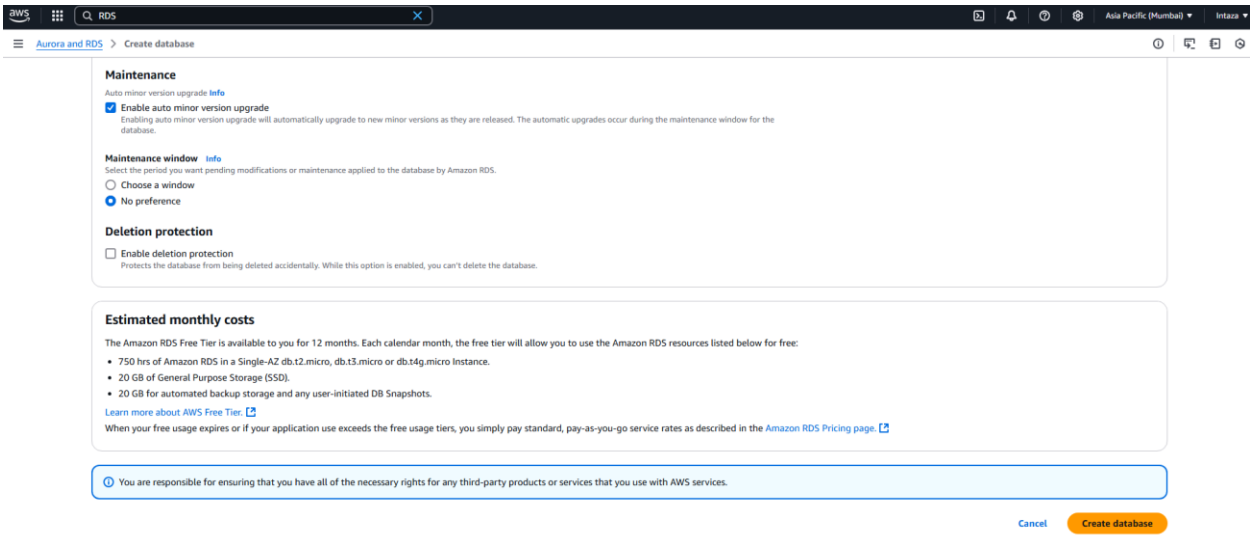
Step 6: Set Initial Database Name

- Under **Additional configuration**, enter a database name (e.g., mydatabase).

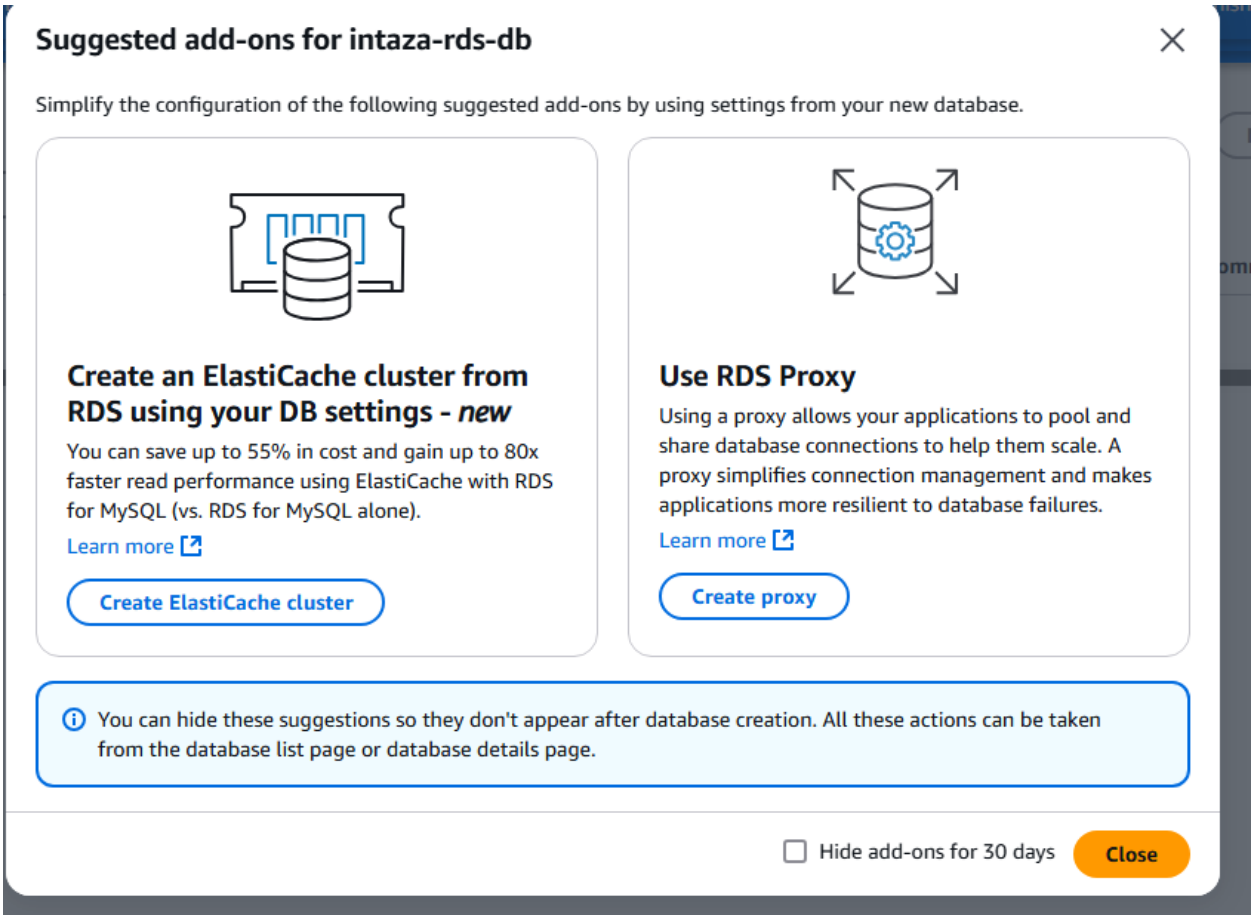




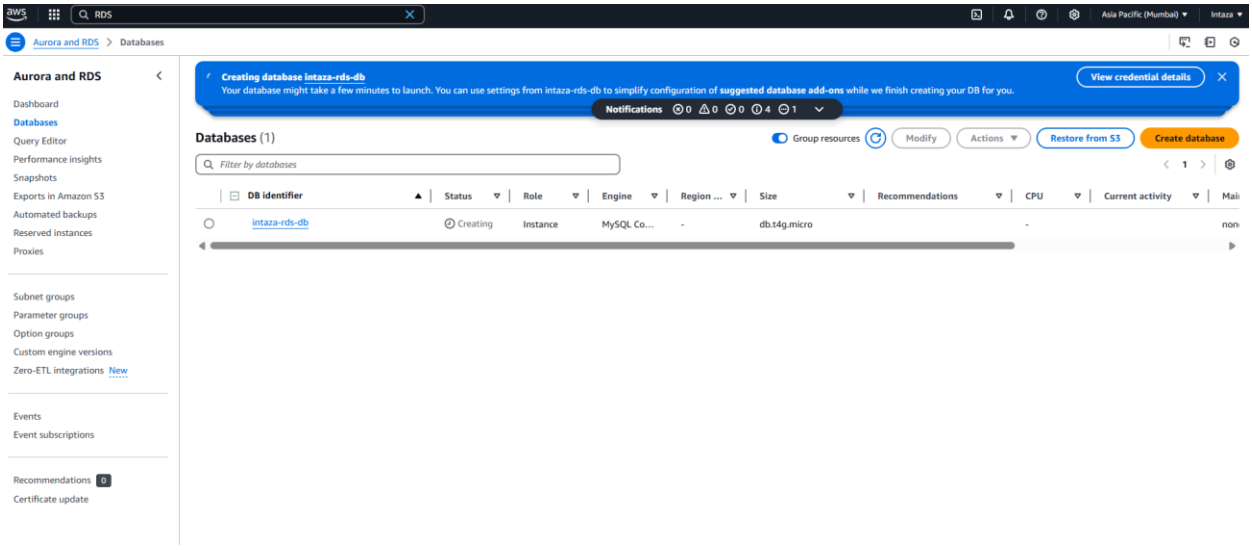
- **Step 7 : Create the Database(**Wait 5–10 minutes for the instance status to turn to **Available**.**)**

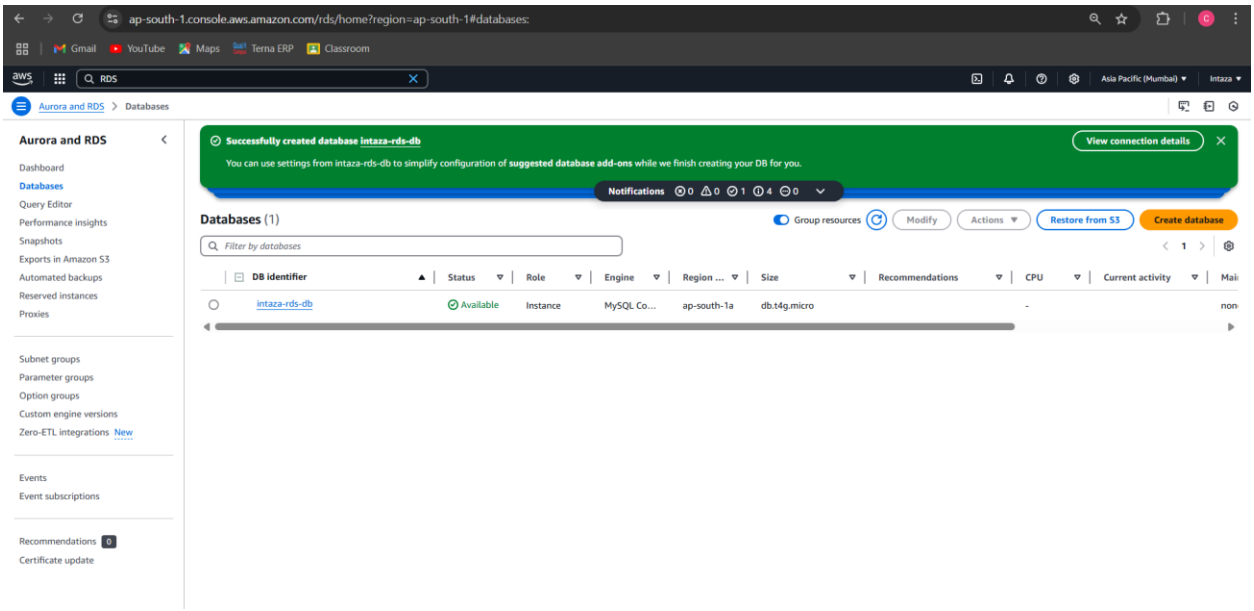


Click on close:

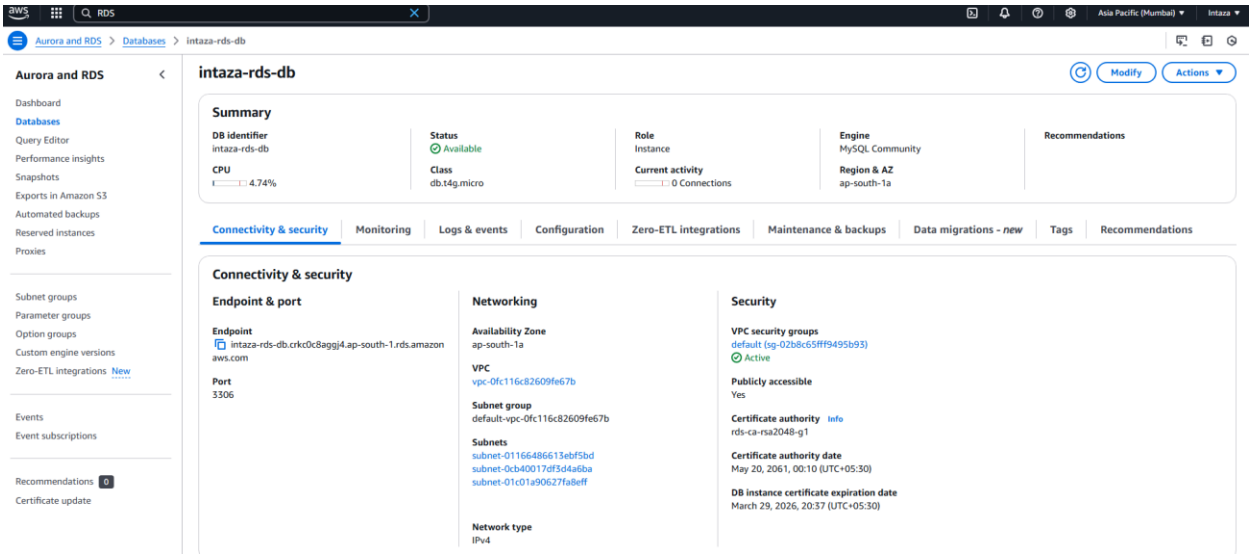


- instance status to turn to **Available**.

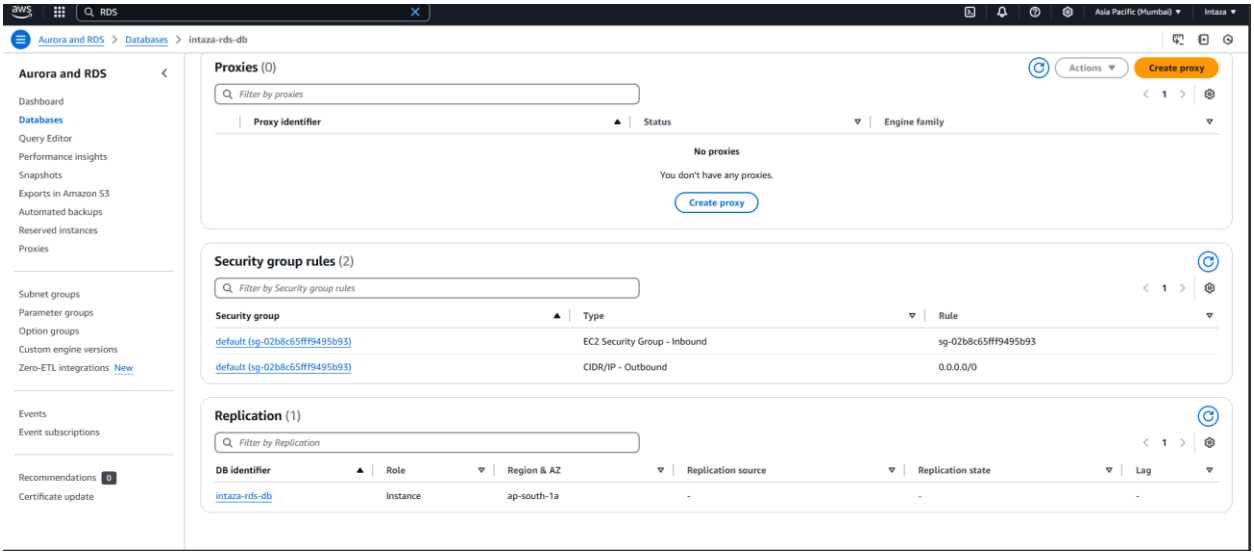




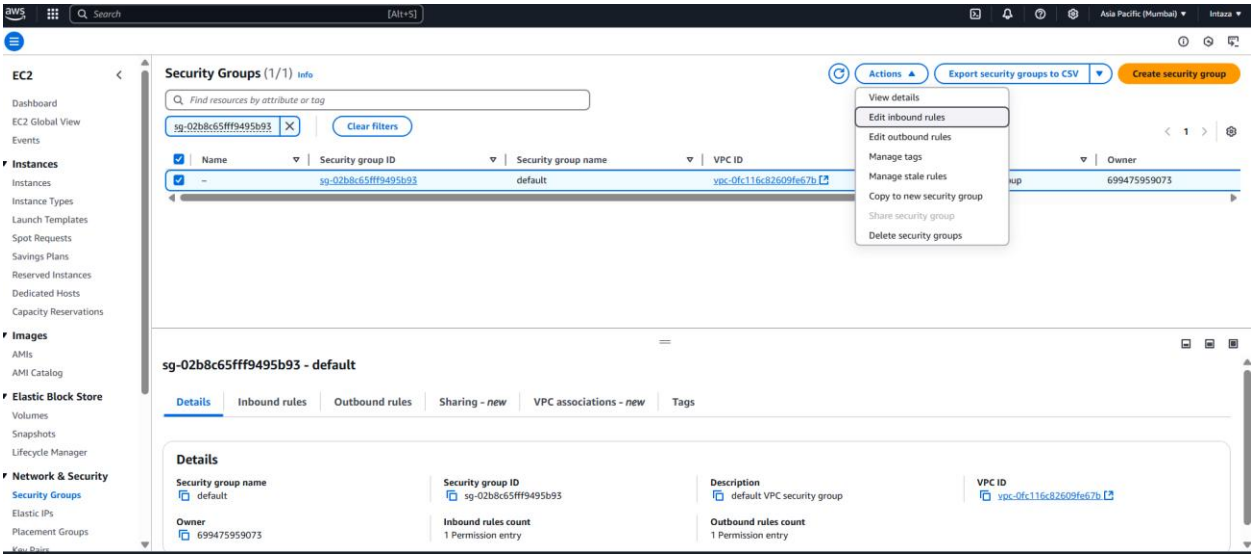
Step 8 : Copy Endpoint :



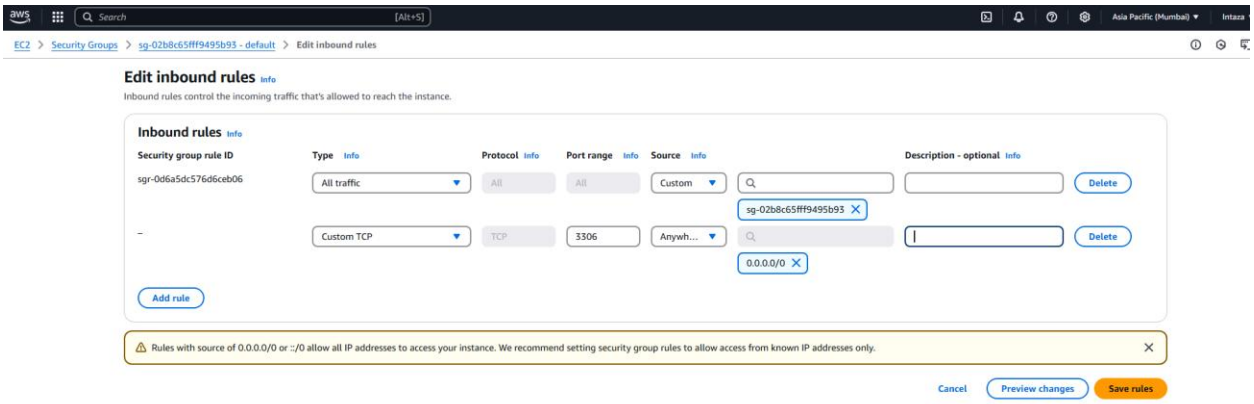
Step 9: Scroll Down and Click on EC2 security group-inbound from Security group rule.



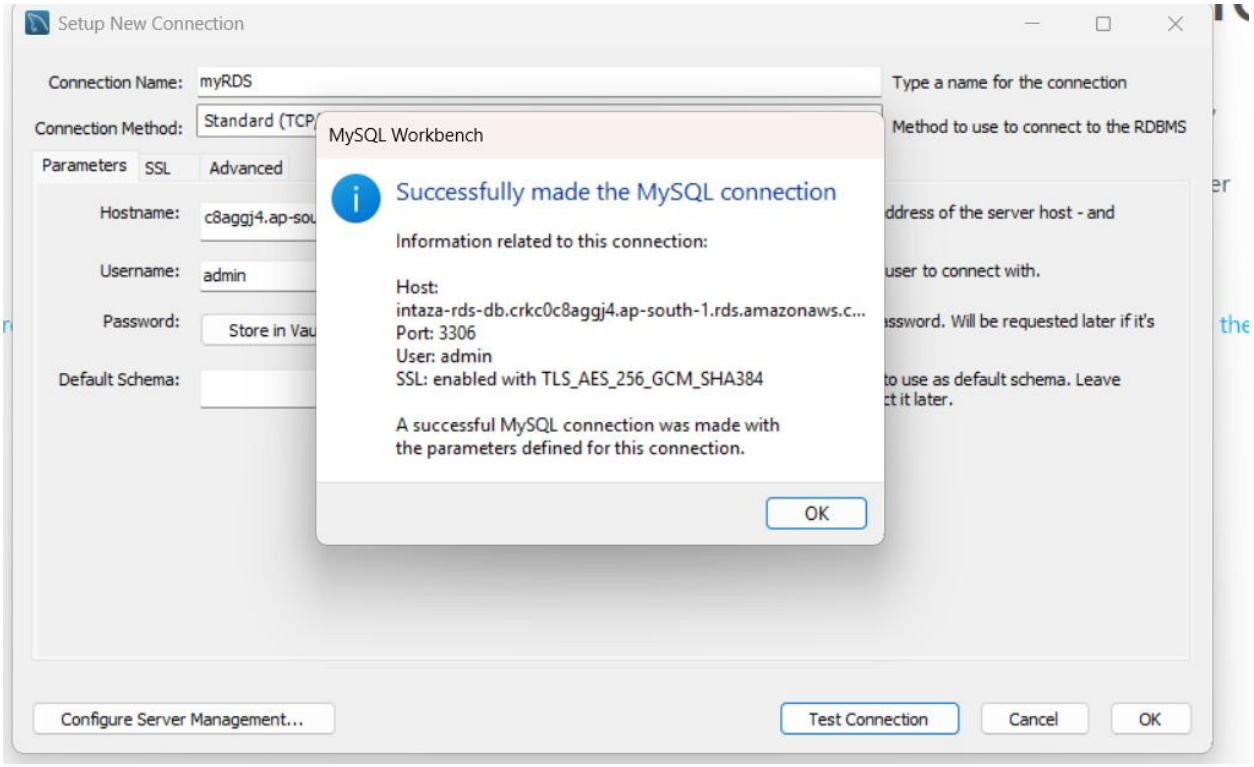
Step 10:Tick Check box and action>Edit inbound rule

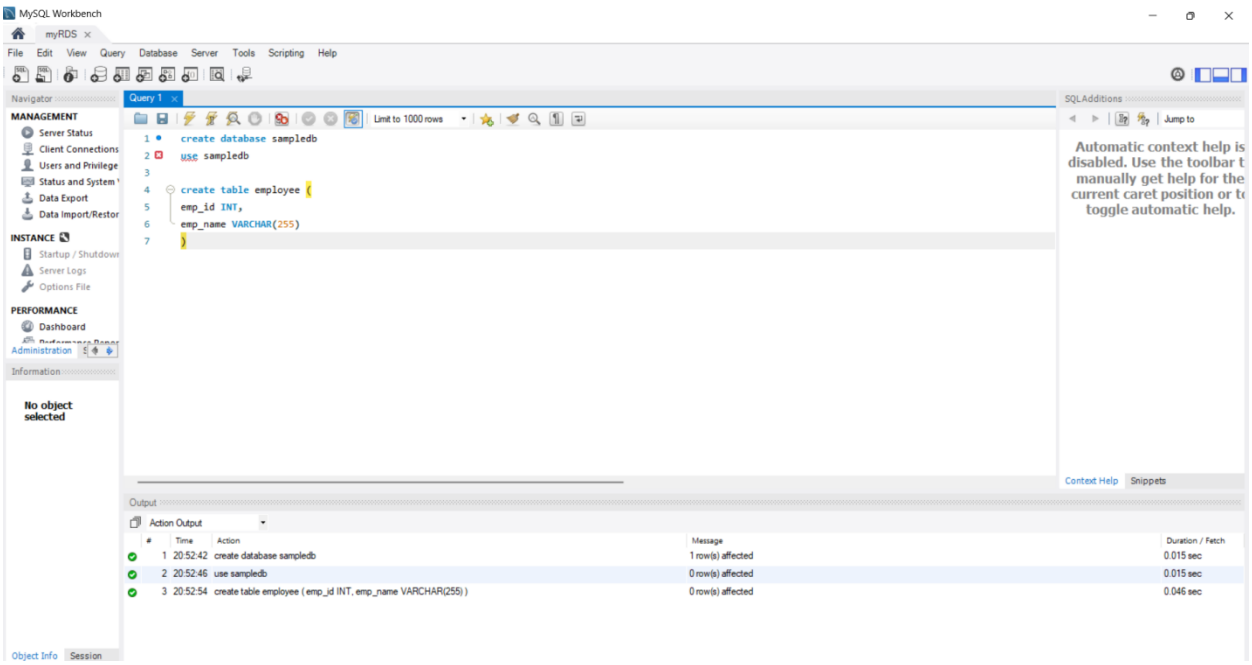


Step 11: Type = Custom TCP and source= Anywere IPv4 , save rule



Step 12: Connect from Local Machine(Mysql Workbench) and run some queries



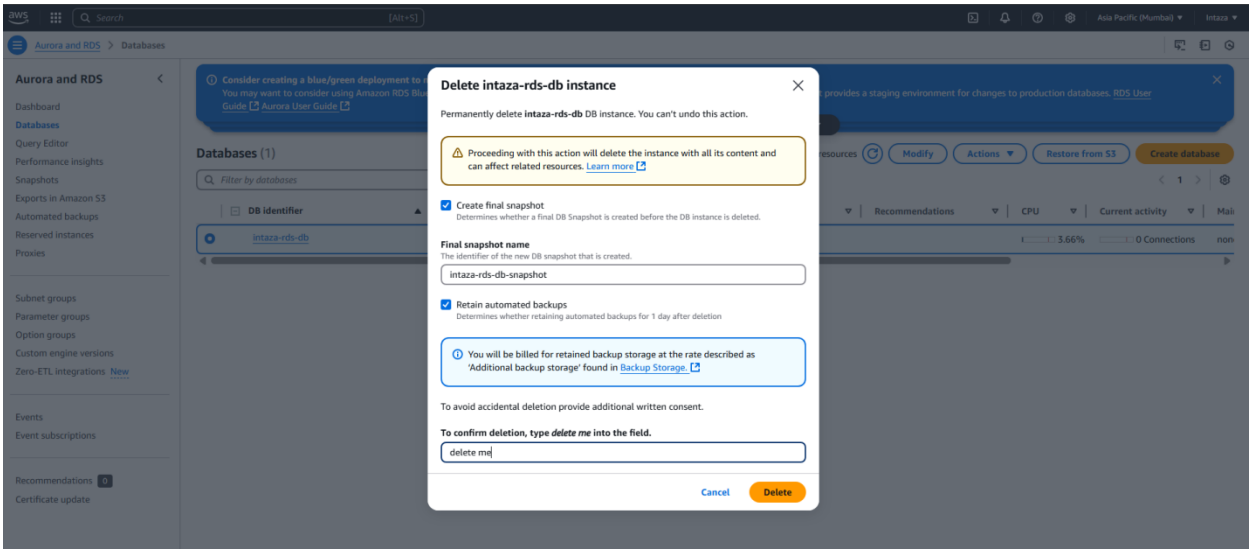


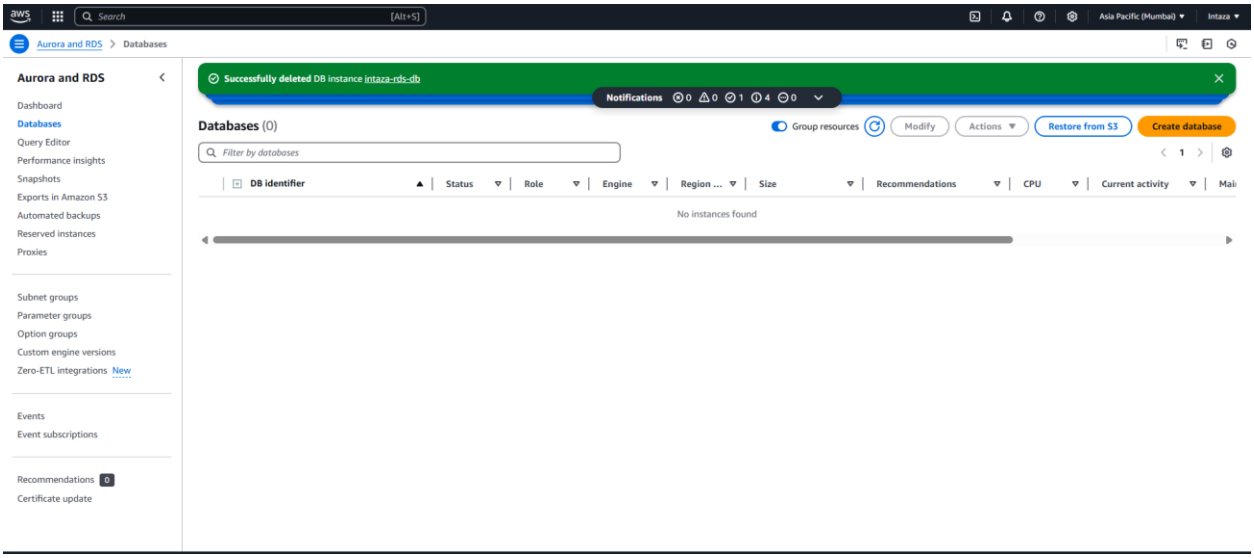
Step 13: Delete the RDS Database Instance

Go to AWS Console → RDS → Databases

Select your database instance (`ma231t.ap-south-1.rds.amazonaws.com`)

Click "Actions" → "Delete"





Step 14: Reset the Default Security Group

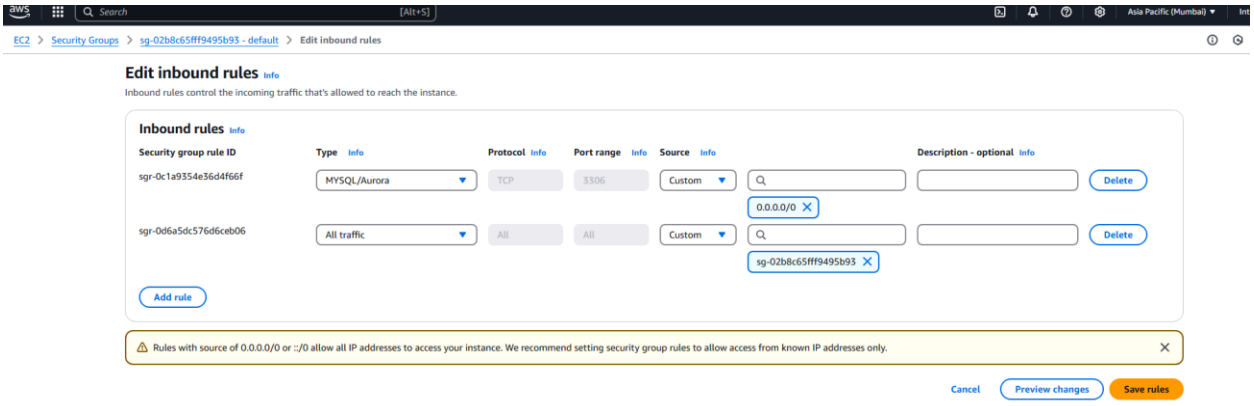
Go to AWS Console → EC2 → Security Groups

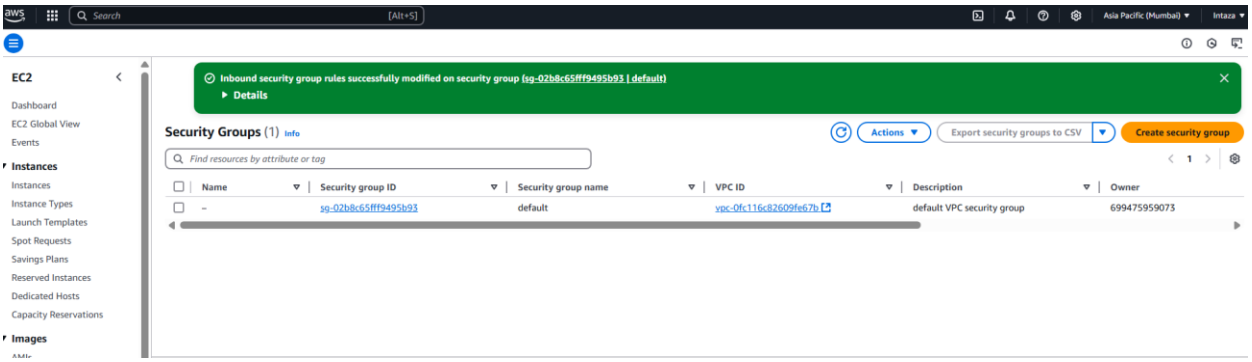
Find and select `sg-067fae1521f348144` (or any other related security groups)

Click "Actions" → "Click "Edit inbound rules"

Remove any custom inbound rules (if you mistakenly allowed external access).

Click "Save rules"





B.3 Observations and learning:

The experiment involved setting up a database using Amazon RDS, exploring its features, and understanding its management interfaces.We were able to configure and manage a relational database instance using AWS Management Console, AWS CLI, and API. Security features such as access control through IAM and database backups were observed as essential components of RDS.The ability to scale resources independently (CPU, memory, storage, and IOPS) was demonstrated. Automated database management, including backups, software patching, and failure recovery, was tested.Various database engines like MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server were available for deployment.

B.4 Conclusion:

The experiment successfully demonstrated the ease of deploying and managing a relational database using Amazon RDS. The automated features of RDS reduce the administrative overhead while providing scalability, security, and reliability. By utilizing AWS Management Console, CLI, and SDKs, students gained practical insights into database operations in a cloud environment, reinforcing the importance of cloud-based database solutions in modern applications.

B.5 Question of Curiosity

Q1: What types of RDS databases are currently available?
Ans:

Different types of database engines are currently available in RDS:

1. Amazon Aurora
2. MySQL
3. MariaDB
4. Microsoft SQL Server
5. Oracle
6. PostgreSQL

Q2: Where are RDS backups stored?

Ans:

Backups are stored in Amazon S3. For more information about backup storage costs, see Amazon RDS pricing. If you chose to retain automated backups when you delete a DB instance, the automated backups are saved for the full retention period.

Q3: Is RDS A PaaS?

Amazon RDS is SaaS, not PaaS. It's a fully managed service; it just turns out that users are often software developers. It's not a platform with an integrated stack where one can develop and/or deploy applications, so not PaaS. Google photos is SaaS for people who take photos. You can access Google Photos via an API, so having an end-user facing app is not the deterministic criterion IMO.