EXPERIMENT- 09

Student Name: Advitiya Sharma **UID: 23BIT70015**

Branch: BE-CSE-IOT **Section/Group: 23AIT_KRG2-A**

Semester: 05 **Date of Performance: 31/10/25**

Subject Name: ADBMS **Subject Code: 23CSP-333**

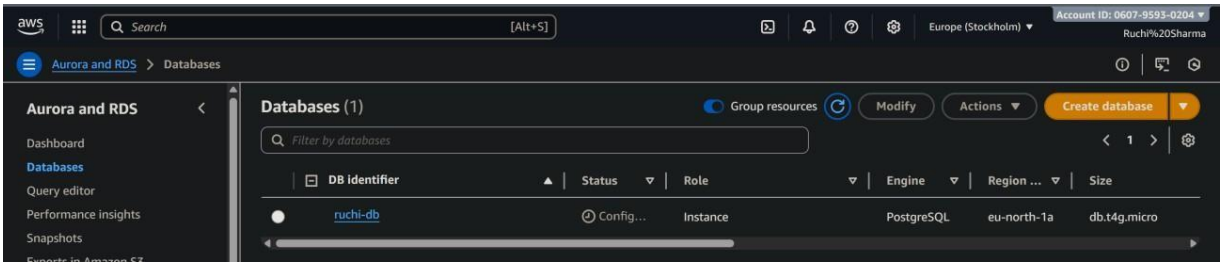
1. Aim:
To create and connect a PostgreSQL database instance on **Amazon RDS (Relational Database Service)**

2. Objective:
1. To understand the steps involved in launching a database instance using Amazon RDS.
2. To configure a database for public access and connect it with a local client (pgAdmin).
3. To perform basic SQL operations (CREATE, INSERT, SELECT).

3. Tools / Software
1. Amazon Web Services (AWS)
2. PostgreSQL
3. pgAdmin 4
4. RDS (Relational Database Service)

4. Program:

Step 1: Create and Configure Database Instance
1. Login to AWS Console → RDS → Create database, select Standard create and PostgreSQL under the Free Tier template.
2. Set DB identifier: ruchi-db, Username: postgre, choose db.t3.micro, 20 GB gp2 storage, and enable Public access.
3. Click Create database and wait until the status shows Available in the RDS dashboard.

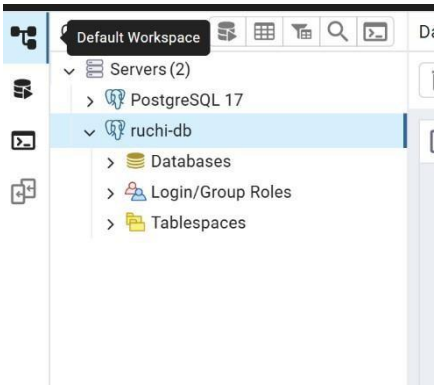


Step 2: Configure Security Group (Allow Local Access Only)

1. In AWS Console → go to RDS → Databases → click your DB (ruchi-db).
2. Open the Connectivity & Security tab.
3. Under VPC security groups, click the linked group name (it opens EC2 security groups).
4. Click Edit inbound rules → Add rule
   - Type: PostgreSQL
   - Protocol: TCP
   - Port: 5432
   - Source: My IP
5. Click Save rules.

Step 3: Connect Database Using pgAdmin

1. Open pgAdmin 4 on your local system.
2. Right-click Servers → Create → Server.
3. Under the General tab, enter the name: postgre.
4. Under the Connection tab, fill in the following details:
   - Host name/address: ruchi-db.xxxxxxx.rds.amazonaws.com
   - Port: 5432
   - Username: postgre
   - Check Save password.
5. Click Save to connect your RDS PostgreSQL database.



5. Learning Outcomes:

1. Understand the procedure to provision and configure a PostgreSQL instance using AWS RDS.
2. Configure security groups and network access controls for secure database connectivity.
3. Establish a remote database connection using pgAdmin and verify successful access.