



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

EXPERIMENT- 09

Student Name: Yeshika

UID: 23BDA70132

Branch: BE-CSE-BDA

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.

The screenshot shows the AWS RDS Databases page. The sidebar on the left has options like Aurora and RDS, Dashboard, Databases (which is selected), Query editor, Performance insights, and Snapshots. The main area is titled 'Databases (1)'. It lists one database entry: 'ruchi-db'. The table columns are DB identifier, Status, Role, Engine, Region ..., and Size. The database details are: Status: Config..., Instance: PostgresSQL, Region: eu-north-1a, and Size: db.t4g.micro. There are buttons for Group resources, Modify, Actions, and Create database.



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

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.

The screenshot shows the AWS RDS Security Groups page for a specific database instance. The title bar says "sg-0570f959421927738 - default". Below it, there's a table titled "Inbound rules (2)". The table has columns for Name, Security group rule ID, IP version, Type, Protocol, Port range, and Source. The first two rows show rules for PostgreSQL traffic on port 5432 from the source "106.206.235.43" (which corresponds to the IP of the user's local machine). The third row shows a general rule for all traffic on all ports.

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-0d39d1bf593210da4	IPv4	PostgreSQL	TCP	5432	106.206.235.43
-	sgr-0ee4f18536cb88772	-	All traffic	All	All	sg-0570f95942

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: postege.
4. Under the Connection tab, fill in the following details:
 - Host name/address: ruchi-db.xxxxxxx.rds.amazonaws.com
 - Port: 5432
 - Username: postege
 - Check Save password.
5. Click Save to connect your RDS PostgreSQL database.

The screenshot shows the pgAdmin 4 interface with a connection tree. At the top, it says "Default Workspace". Below that, under "Servers", there are two entries: "PostgreSQL 17" and "ruchi-db". The "ruchi-db" entry is expanded, showing its sub-nodes: "Databases", "Login/Group Roles", and "Tablespaces".

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.