

DATABASE AUTOMATION

PROG8850

ASSIGNMENT 2

Database Automation and Scripting

SUBMITTED BY,

MARTIN JOHNY

8945124

Automating Database Schema Changes and CI/CD Pipeline Implementation with AWS RDS

1. Creating the AWS RDS MySQL Database

Login to AWS Management Console:

- Open your browser and go to [AWS Management Console](#).
- Log in with your AWS credentials.

Create an RDS MySQL Database:

- In the AWS console, navigate to **RDS** (search for it in the search bar).
- Click on **Create database**.

Configure the Database:

- **Engine options:** Select **MySQL**.
- **Version:** Choose the MySQL version you want to use.
- **Templates:** Select **Free tier**
- **Settings:**
 - Set **DB instance identifier**
 - Set **Master username**
 - Set **Master password**

Choose Database Instance Class:

- Choose the instance type based on your needs. You can select the free-tier instance for testing.

Set Database Connectivity:

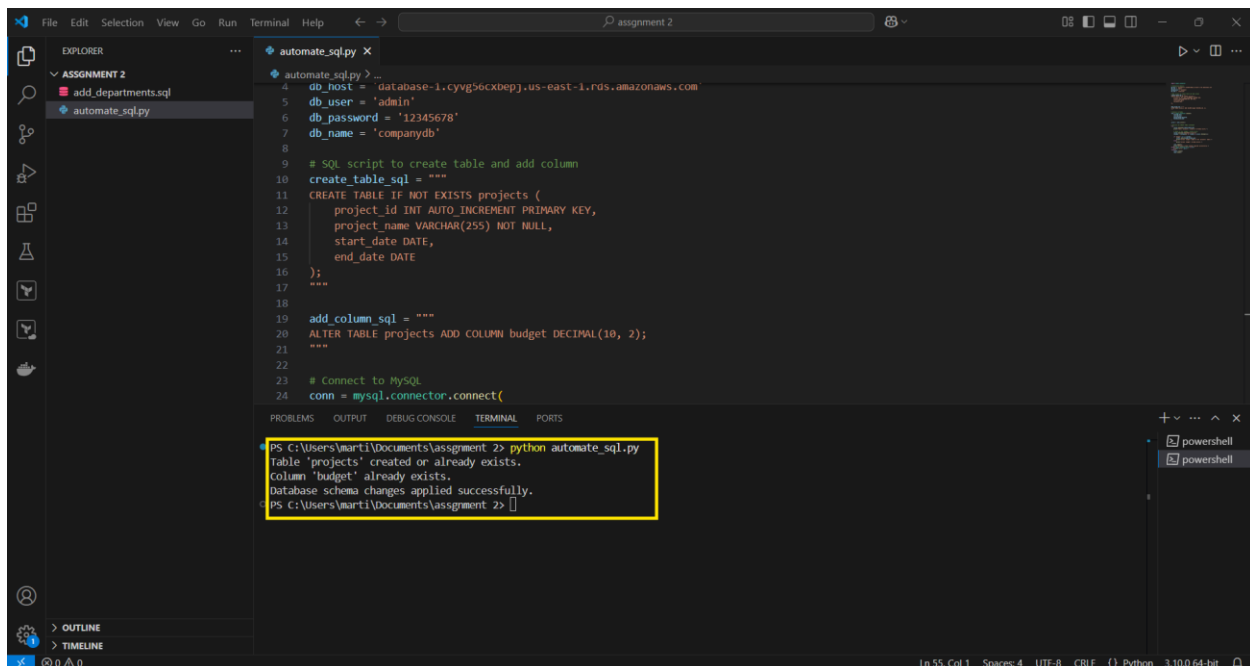
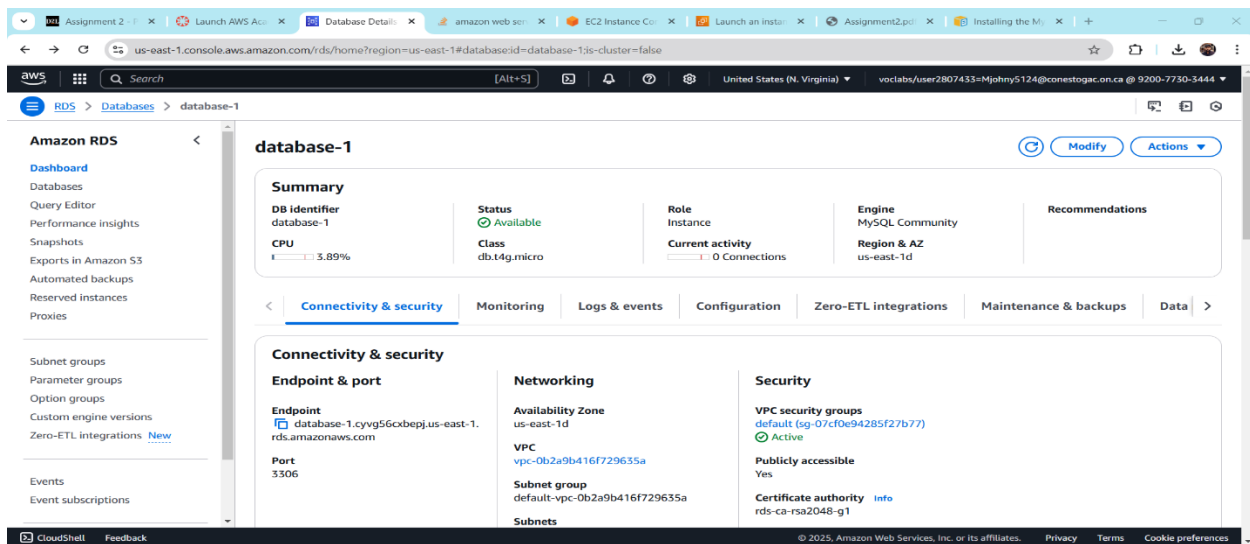
- Choose the **VPC** (Virtual Private Cloud).
- Ensure that **Public accessibility** is set to **Yes** if you need external access (e.g., for connection from GitHub Actions).
- In **VPC security group**, create a new security group to allow inbound traffic on port 3306 (MySQL default port).

Create the Database Instance:

- After reviewing your configuration, click **Create database**. AWS will take a few minutes to provision the RDS MySQL instance.

Create the Database companydb:

- Once your RDS instance is available, connect to it using MySQL Workbench or a similar client.
- Run the following SQL command to create the companydb database.



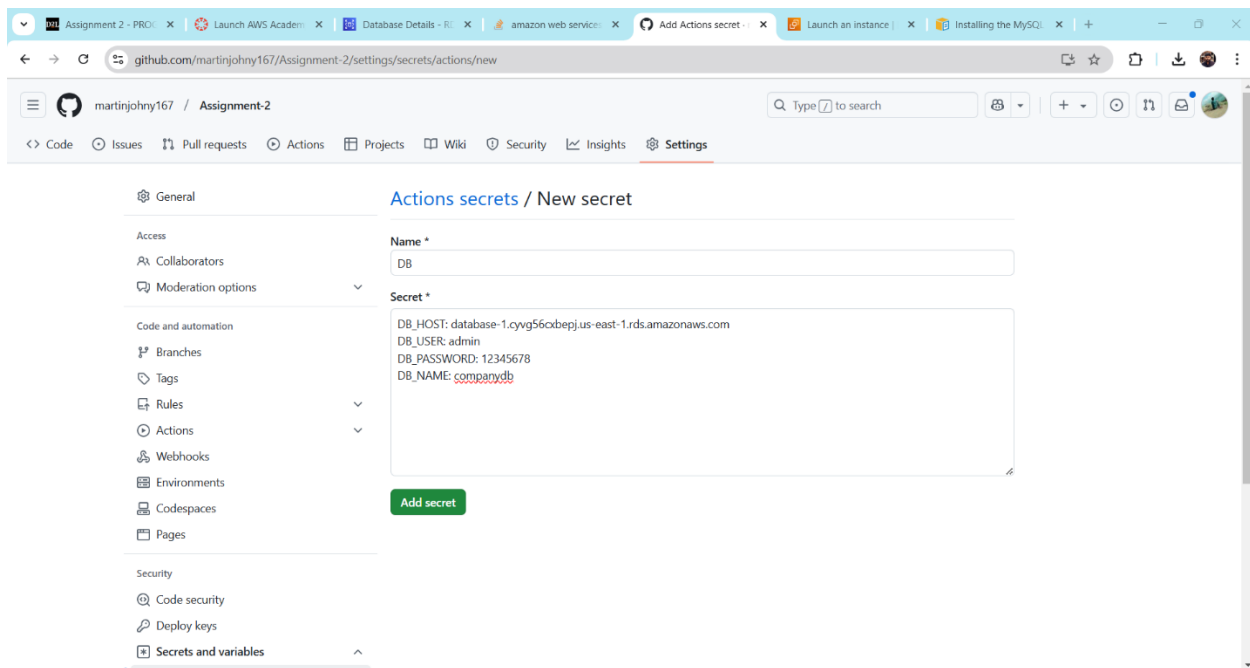
2. Setting Up GitHub Actions

1. Create a New GitHub Repository:

- Log in to [GitHub](#) and create a new repository for your project.

2. Add Secrets for Database Connection:

- Go to the **Settings** of your GitHub repository.
- In the left sidebar, click **Secrets and variables > Actions**.
- Add the following secrets:
 - **DB_HOST**: The endpoint of your AWS RDS MySQL instance
 - **DB_USER**: The admin username for MySQL
 - **DB_PASSWORD**: The password for your MySQL instance
 - **DB_NAME**: The name of the database



3. Add the GitHub Actions Workflow File:

- In your repository, create the directory `.github/workflows/` if it doesn't already exist.
- Add the necessary YAML file for your CI/CD pipeline.

3. Testing the Workflow and Results

Push Changes to GitHub:

- Push the SQL script (e.g., add_departments.sql) to the GitHub repository. This will trigger the GitHub Actions workflow.

Monitor the Workflow:

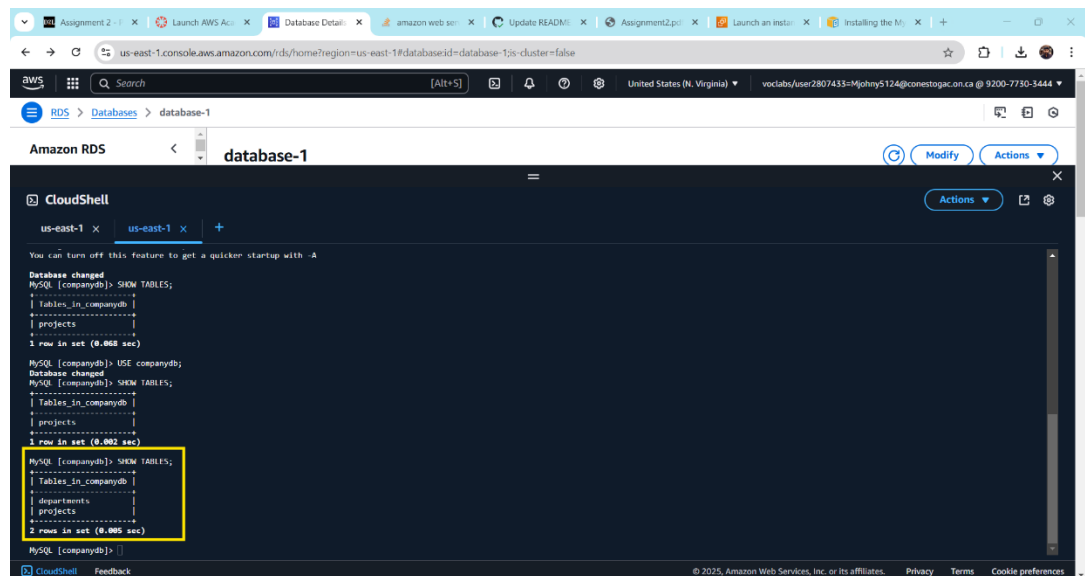
- Go to the **Actions** tab in your GitHub repository.
- You should see the workflow listed under **Workflow runs**. Click on the latest run to see the details.

Verify Workflow Success:

- The logs should show the execution of the Python script and indicate that the SQL script was executed successfully.

Verify the Changes in AWS RDS MySQL:

- Log into your RDS MySQL instance.
- Run SQL queries to verify that the changes (e.g., table creation) have been applied successfully.



```
us-east-1 x us-east-1 x +
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [companydb]> SHOW TABLES;
+-----+
| Tables_in_companydb |
+-----+
| projects             |
+-----+
1 row in set (0.068 sec)

MySQL [companydb]> USE companydb;
Database changed
MySQL [companydb]> SHOW TABLES;
+-----+
| Tables_in_companydb |
+-----+
| departments         |
| projects             |
+-----+
1 row in set (0.002 sec)

MySQL [companydb]> SHOW TABLES;
+-----+
| Tables_in_companydb |
+-----+
| departments         |
| projects             |
+-----+
2 rows in set (0.005 sec)

MySQL [companydb]> |
```

