BIG DATA MANAGEMENT

Assignment III – Amazon RDS Integration using Java

Submitted To:

Dr. Dip Shankar Banerjee
Associate Professor
School of AI and Data Engineering
Indian Institute of Technology Jodhpur – Rajasthan

Submitted By:

Name: Sree Charan
Roll Number: G24AI1095
Email: G24AI1095@iij.ac.in
Course: PGD-Trimester 3
Date: 25/06/2025

Index

1. Introduction

This assignment is aimed at integrating Java applications with cloud-hosted databases using Amazon RDS. By simulating a financial tracking system through two tables — `company` and `stockprice` — the goal was to understand how backend services interact with managed databases in real-time. This includes operations such as schema creation, data population, conditional deletion, and analytical querying.

2. Overview of Amazon RDS

Amazon Relational Database Service (RDS) is a managed cloud database solution that simplifies the setup, use, and scaling of databases. Supporting engines such as MySQL, PostgreSQL, Oracle, and more, RDS allows developers to focus on application logic while offloading the operational burden of backups, patches, and replication to AWS.

3. Features and Advantages of RDS

Key benefits of Amazon RDS include high availability, scalability, and integrated monitoring. It supports automated backups, manual snapshots, and point-in-time recovery. The use of Multi-AZ deployments ensures failover capabilities and minimal downtime. Security is enhanced with options like encryption, VPC integration, and IAM role-based access control.

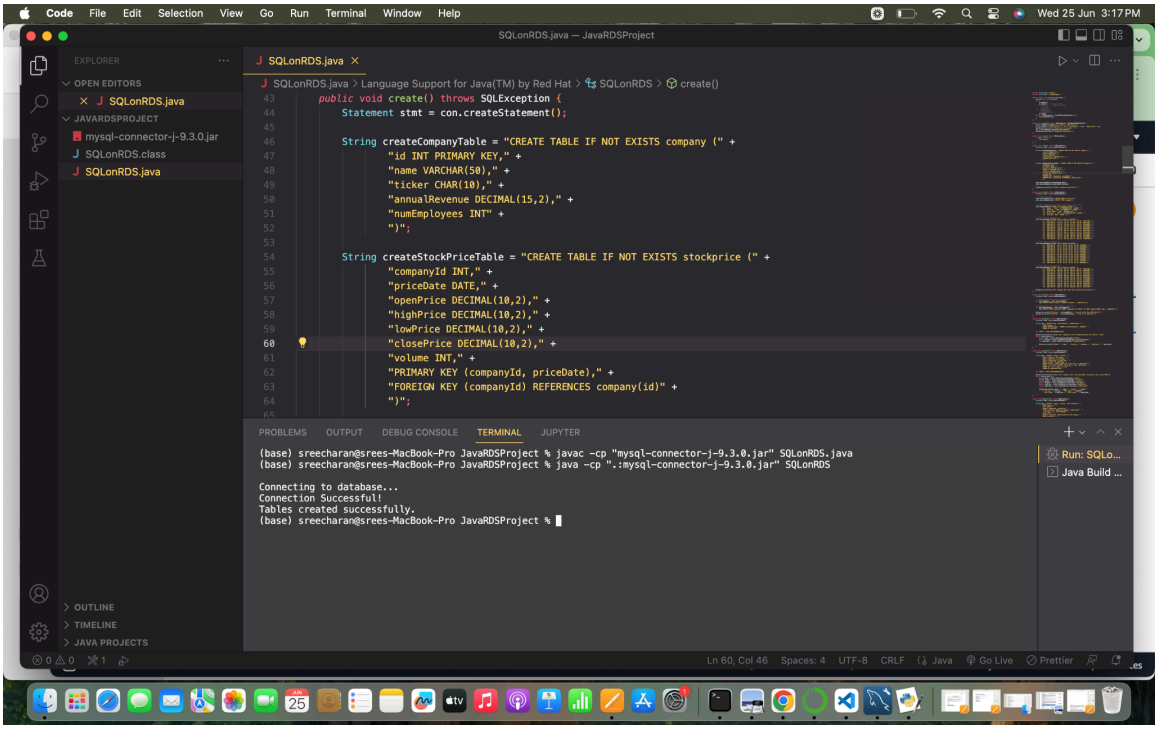
4. Java-RDS Integration Approach

JDBC (Java Database Connectivity) serves as the bridge between the Java application and the RDS-hosted MySQL database. The process begins by loading the MySQL JDBC driver, forming a JDBC URL with RDS endpoint and credentials, and establishing the connection using the DriverManager. Once the connection is successful, SQL queries are executed through Statement objects, and results are fetched using ResultSet.

5. Step-by-Step Implementation

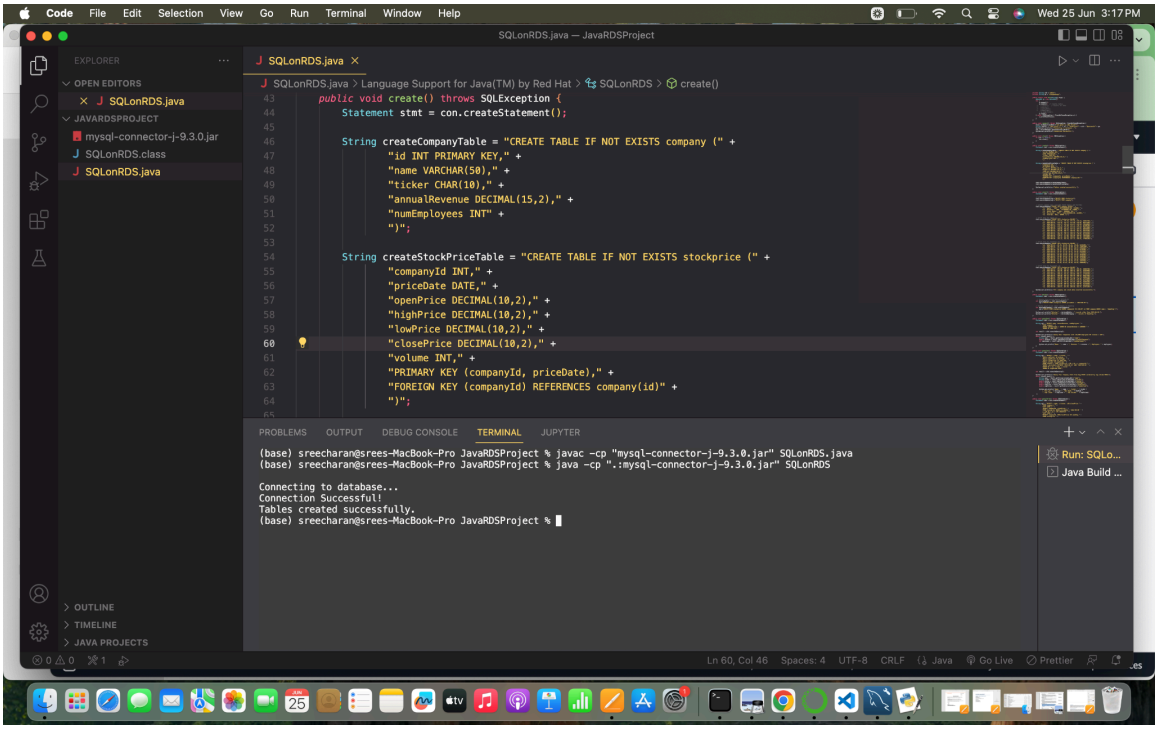
5.1 Connection Setup

The `connect()` method uses JDBC to establish a connection to the RDS instance. Proper exception handling ensures robustness. Connection success is printed on the console.



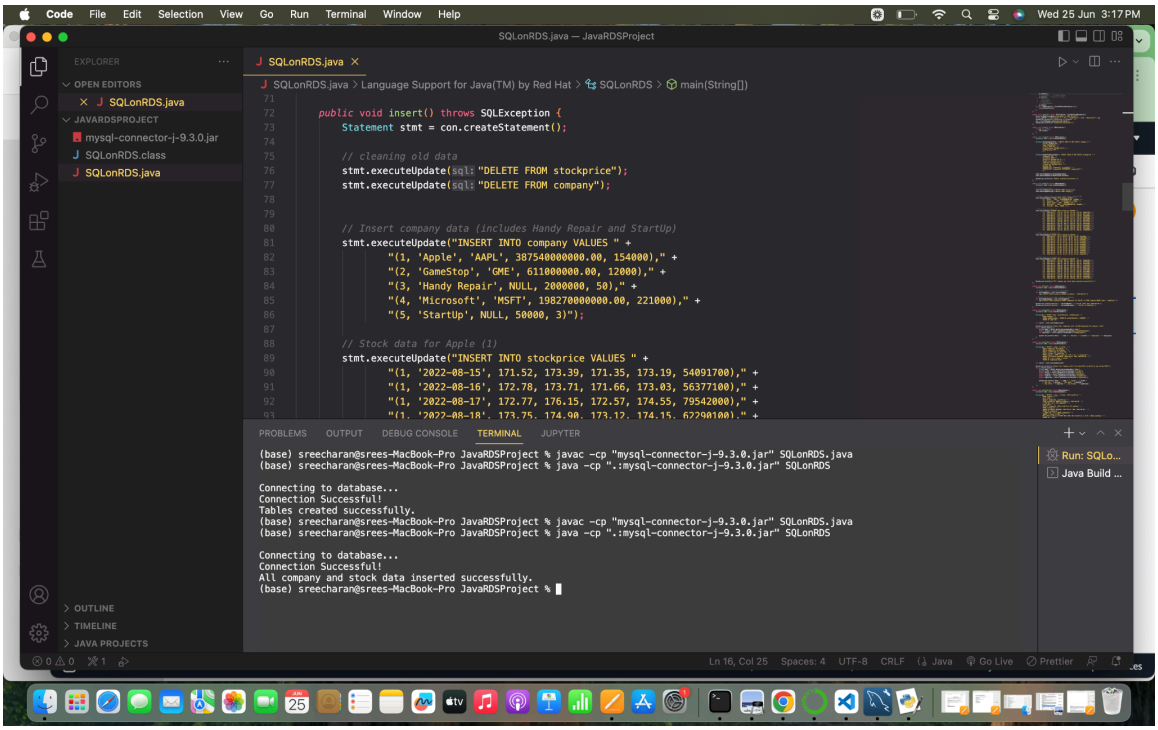
5.2 Creating Tables

The `create()` method defines two tables: `company` and `stockprice`. Constraints like primary keys and foreign keys are used to enforce referential integrity.



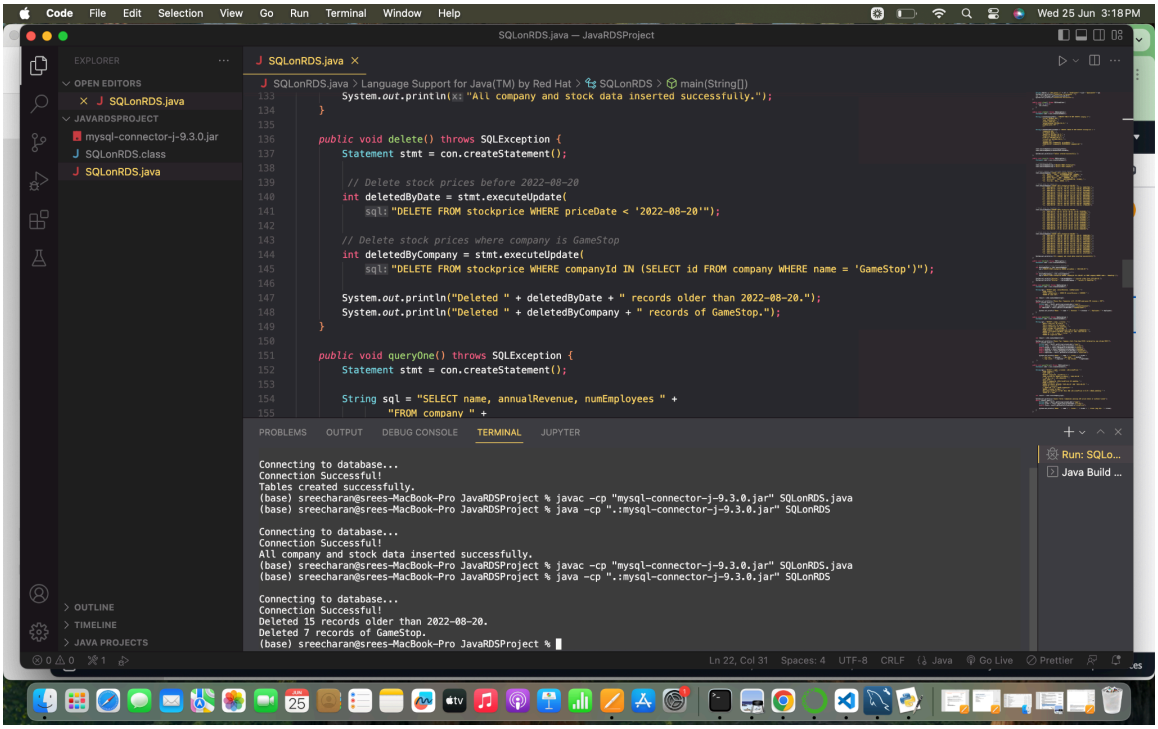
5.4 Data Insertion

Using the `insert()` method, sample data representing real-world stock prices and companies are inserted into the tables. Multiple INSERT statements are bundled and executed in sequence.



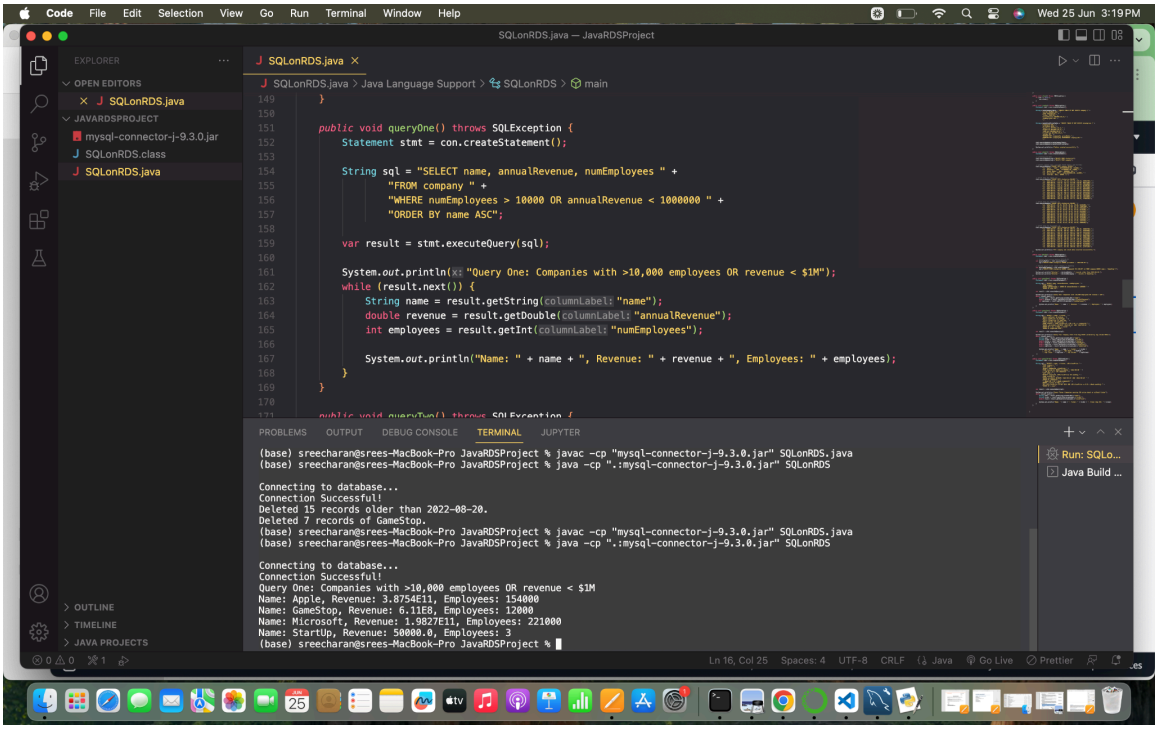
5.5 Deletion Logic

The `delete()` function filters out stock entries dated before 2022-08-20, and all entries related to GameStop. This simulates data cleaning and relevance filtering.



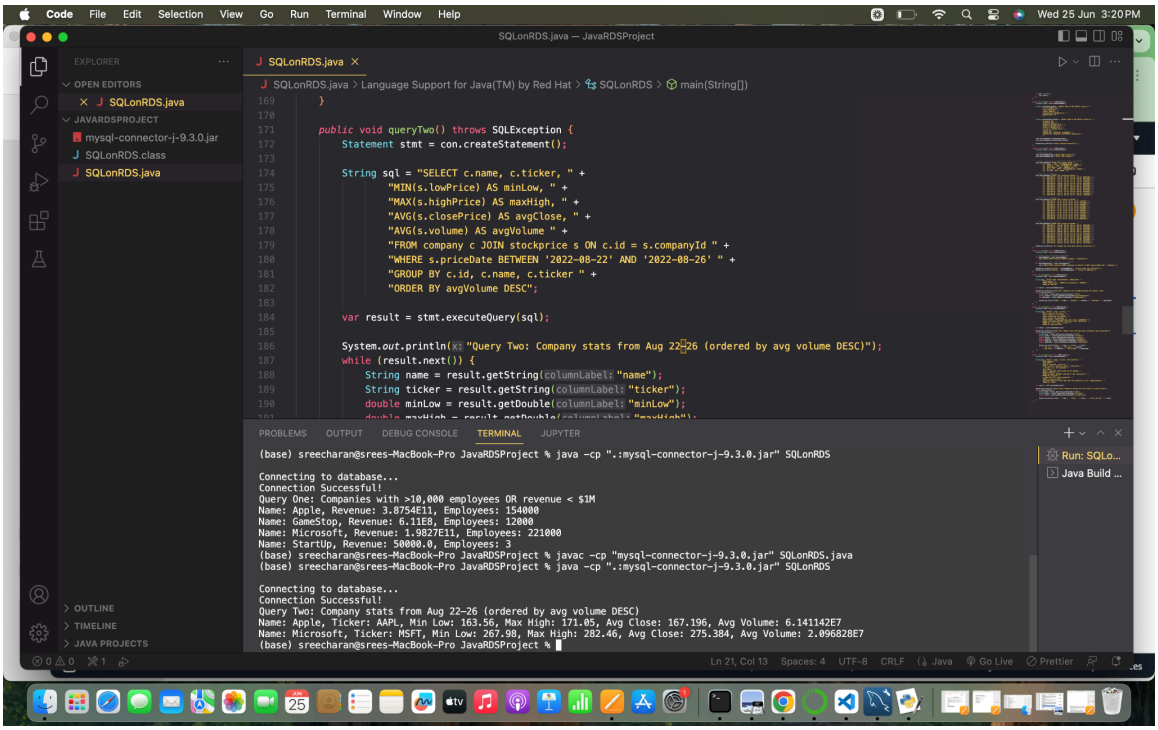
5.6 QueryOne - Conditional Company Info

This query retrieves company records that have more than 10,000 employees or an annual revenue less than 1 million. It demonstrates conditional retrieval.



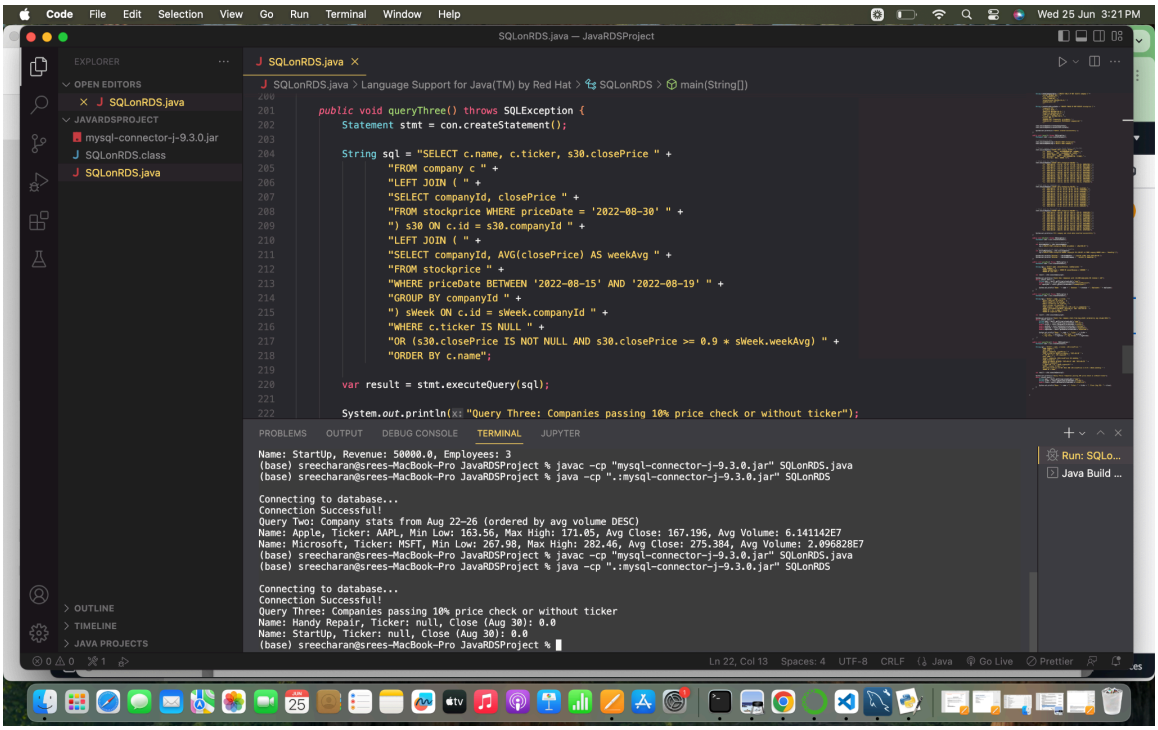
5.7 QueryTwo - Weekly Price Statistics

This method calculates and returns stats like minimum price, maximum price, average closing price, and average volume for each company during Aug 22–26.



5.8 QueryThree - Trend-Based Company Filter

This complex query identifies companies whose closing stock price on Aug 30 is not more than 10% below their average closing price from the previous week. Even companies without a stock ticker are included if they meet conditions.



6. Result Observations and Outputs

All SQL operations — table creation, insertion, deletion, and querying — were successfully executed. The outputs observed in MySQL Workbench matched expected results, confirming the validity of logic and data handling. Complex queries like average calculations and joins returned accurate, timely results based on the inserted sample data.

7. Conclusion

This assignment effectively demonstrates the integration of cloud-hosted relational databases with Java. By simulating a financial database environment and performing meaningful queries on realistic datasets, we gained hands-on experience in backend operations, schema design, data manipulation, and analytics. These foundational skills are essential for real-world applications in enterprise and data-driven domains.

8. GitHub Link

👉 https://github.com/Sreecharan162/Big-Data.git