Program 8

Deploy Java application by connecting to RDS Server in cloud

Setting up Java Environment on EC2 instance

Go to https://www.oracle.com/nz/java/technologies/downloads/

Down load jdk for Linux for X64 in local system

Launch EC2 instance

Create VPC→ Create IGW → Create Subnet → Route table and assign RT to Public VPC → Edit RT and assign 0.0.0.0/0 to access internet & assign created IGW and Save changes → In RT Subnet association → Save Association → Create EC2 instance (after selecting AMI, instance type and key pair) → Edit Network settings and specify the resources you have created and choose Public IP →Enable Auto-assign public IP → Security Group (Type SSH and Source type Anywhare) → Launch Instance → Click on instance ID (to SSH into EC2 instance from internet) →copy public IP → Go to terminal on Desktop→ Change permission chmod –R 400 keypair Name → paste IP address on terminal

On EC2 instance terminal

sudo apt-get update

java –version

#mkdir /usr/lib/jvm

Go to /lib directory

/lib># chmod 777 jvm

Exit from EC2 instance

Form local machine scp the downloaded jdk

scp -i jdk_path on local machine EC2 endpoint:/usr/lib/jvm

Connect to EC2 instance again

Unzip the uploaded jdk file

usr/lib/jdk# tar zxvf file name

#sudo nano /etc/environment

Edit the java HOME

Update the java settings

Reference video to set up java environment: https://www.youtube.com/watch?v=lNJlgZ_aEKM

Create Database in AWS RDS service

Login→ Search RDS →RDS → On left side menu→ Databases → Create Database → Choose Standard Create → In engine Option select MySQL database → Edition → MySQL community →8.0.32 → templates → Free tier → Specify DB instance name→ Specify master User name → password → Storage and Connectivity (leave default) → Public access (Yes) → availability zone (No preference) → Create Database

Getting the details of the database created

Click on the Id of the database created → Note down the End-points and ports

Execute the following code by doing changes in database connectivity

```
import java.sql.*;
public class JdbcCrudDemo {
   // Database connection parameters
   static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database";
   static final String JDBC_USER = "your_username";
   static final String JDBC_PASS = "your_password";

   public static void main(String[] args) {
      try {
         // 1. Load the JDBC driver (optional for newer drivers)
         Class.forName("com.mysql.cj.jdbc.Driver");

         // 2. Establish the connection
         Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASS);

         // 3. CREATE: Insert a new employee
         String insertSql = "INSERT INTO employees (name, email, country, salary) VALUES (?, ?, ?, ?)";
         PreparedStatement insertStmt = conn.prepareStatement(insertSql);
         insertStmt.setString(1, "Alice");
         insertStmt.setString(2, "alice@example.com");
         insertStmt.setString(3, "USA");
         insertStmt.setDouble(4, 50000);
         insertStmt.executeUpdate();
         System.out.println("Inserted new employee.");

         // 4. READ: Retrieve all employees
         String selectSql = "SELECT * FROM employees";
         Statement selectStmt = conn.createStatement();
         ResultSet rs = selectStmt.executeQuery(selectSql);
         System.out.println("Employee List:");
         while (rs.next()) {
            System.out.println(
               rs.getInt("id") + ", " +
               rs.getString("name") + ", " +
               rs.getString("email") + ", " +
               rs.getString("country") + ", " +
               rs.getDouble("salary")
            );
         }
```

```
            // 5. UPDATE: Update employee's salary
            String updateSql = "UPDATE employees SET salary = ? WHERE name = ?";
            PreparedStatement updateStmt = conn.prepareStatement(updateSql);
            updateStmt.setDouble(1, 60000);
            updateStmt.setString(2, "Alice");
            updateStmt.executeUpdate();
            System.out.println("Updated salary for Alice.");

            // 6. DELETE: Delete employee by name
            String deleteSql = "DELETE FROM employees WHERE name = ?";
            PreparedStatement deleteStmt = conn.prepareStatement(deleteSql);
            deleteStmt.setString(1, "Alice");
            deleteStmt.executeUpdate();
            System.out.println("Deleted employee Alice.");

            // 7. Close resources
            deleteStmt.close();
            updateStmt.close();
            rs.close();
            selectStmt.close();
            insertStmt.close();
            conn.close();

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```