Ex 4	CRUD Operations using JDBC in MySQL

Aim

To Connect to a relational database MySQL and perform Create, Read, Update, and Delete operations using JDBC.

Definitions

MySQL

MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter My, and "SQL", the acronym for Structured Query Language.

CRUD

CRUD is an acronym for **Create, Read, Update, and Delete**, which are the four fundamental operations for data manipulation in databases and applications. These operations allow users or systems to create new data, view existing data, modify existing data, and remove data

JDBC

JDBC, or Java Database Connectivity, is a Java API that lets Java applications connect to and interact with databases like MySQL, Oracle, and PostgreSQL. It provides a standard way to execute SQL queries, send updates, and process results, allowing developers to build database-driven applications that can work with different relational databases.

Procedure

Open MySQL Workbench → Right Click on Localhost → Select Open Connection →Type the following codes:

```
CREATE DATABASE testdb;


USE testdb;


CREATE TABLE users (
   id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(100),
   email VARCHAR(100)
);
select * from users;
```

Open NetBeans IDE.

To create a Project go to File Menu → choose New Project → choose Java from Categories →choose Java Application from Projects →click next →specify the project name as JDBC →click Finish.

Download MySQL Connector from https://dev.mysql.com/downloads/connector/j/

To add JDBC Library, Right click on Libraries folder → Choose Add JAR/Folder → Browse and add **mysql-connector-j-9.4.0.jar**

Right click on source packages folder → choose New → select Java Class → specify the class name as DBConnection →click Finish.

Type the following codes in DBConnection.java:

DBConnection.java

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;


public class DBConnection {

   private static final String URL = "jdbc:mysql://localhost:3306/testdb"; // or "jdbc:postgresql://localhost:5432/testdb"

   private static final String USER = "root"; // change if needed

   private static final String PASSWORD = "root";


   public static Connection getConnection() {

      Connection conn = null;

      try {

         conn = DriverManager.getConnection(URL, USER, PASSWORD);

         System.out.println("Database connected successfully!");

      } catch (SQLException e) {

         System.out.println("Connection failed: " + e.getMessage());

      }

      return conn;

   }
}
```

Right click on source packages folder → choose New → select Java Class → specify the class name as CRUD →click Finish.

Type the following codes in CRUD.java:

CRUD.java

```
import java.sql.*;

public class CRUD {

    // CREATE
    public void addUser(String name, String email) {
        String sql = "INSERT INTO users (name, email) VALUES (?, ?)";
        try (Connection conn = DBConnection.getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, name);
            stmt.setString(2, email);
            stmt.executeUpdate();
            System.out.println("User added successfully!");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // READ
    public void getUsers() {
        String sql = "SELECT * FROM users";
        try (Connection conn = DBConnection.getConnection();
             Statement stmt = conn.createStatement();
```

```
         ResultSet rs = stmt.executeQuery(sql)) {

        while (rs.next()) {
            System.out.println(rs.getInt("id") + " | " +
                               rs.getString("name") + " | " +
                               rs.getString("email"));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// UPDATE
public void updateUser(int id, String name, String email) {
    String sql = "UPDATE users SET name=?, email=? WHERE id=?";
    try (Connection conn = DBConnection.getConnection();
         PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setString(1, name);
        stmt.setString(2, email);
        stmt.setInt(3, id);
        stmt.executeUpdate();
        System.out.println("User updated successfully!");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// DELETE
public void deleteUser(int id) {
    String sql = "DELETE FROM users WHERE id=?";
```

```
        try (Connection conn = DBConnection.getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setInt(1, id);
            stmt.executeUpdate();
            System.out.println("User deleted successfully!");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Right click on source packages folder → choose New → select Java Class → specify the class name as Main →click Finish.

Type the following codes in Main.java:

Main.java

```
public class Main {
    public static void main(String[] args) {
        CRUD dao = new CRUD();

        // CREATE
        dao.addUser("John Doe", "john@example.com");
        dao.addUser("Peter Pan", "peter@example.com");

        // READ
        dao.getUsers();

        // UPDATE
        dao.updateUser(1, "John Smith", "johnsmith@example.com");
        dao.updateUser(3, "Peter Mathew", "mathew@example.com");
        // DELETE
        dao.deleteUser(1);
    }
}
```

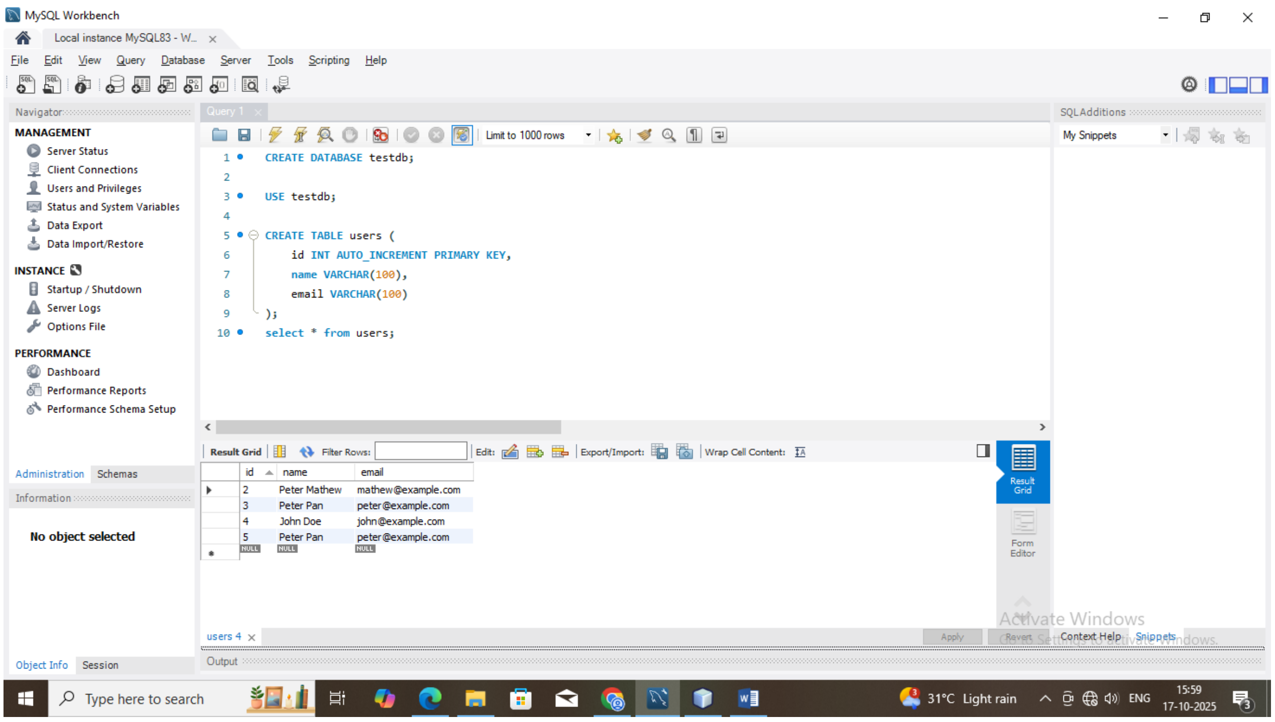
Right click on Main.java file → choose Run File. You can see the following result in the output window.

Output

```
run:
Database connected successfully!
User added successfully!
Database connected successfully!
User added successfully!
Database connected successfully!
2 | Peter Mathew | mathew@example.com
3 | Peter Mathew | mathew@example.com
4 | John Doe | john@example.com
5 | Peter Pan | peter@example.com
6 | John Doe | john@example.com
7 | Peter Pan | peter@example.com
8 | John Doe | john@example.com
9 | Peter Pan | peter@example.com
Database connected successfully!
User updated successfully!
Database connected successfully!
User updated successfully!
Database connected successfully!
User deleted successfully!
BUILD SUCCESSFUL (total time: 0 seconds)
```

To view the table, go to MySQL workbench → Run the below query,

select * from users;



```
CREATE DATABASE testdb;

USE testdb;

CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100)
);
select * from users;
```

id	name	email
2	Peter Mathew	mathew@example.com
3	Peter Pan	peter@example.com
4	John Doe	john@example.com
5	Peter Pan	peter@example.com
NULL	NULL	NULL

Result

Thus, a java application to connect to a relational database MySQL has been successfully executed and Create, Read, Update, and Delete operations have been performed using JDBC.