

Lab Exercise Manual for JDBC – CRUD Operations

Hands-on 1: Creating a Table in MySQL Database Using JDBC

Step 1: Open Eclipse IDE

Step 2: Create New Project

File-> New Project ->Dynamic Web Project Name (as CreateDemo) -> Save

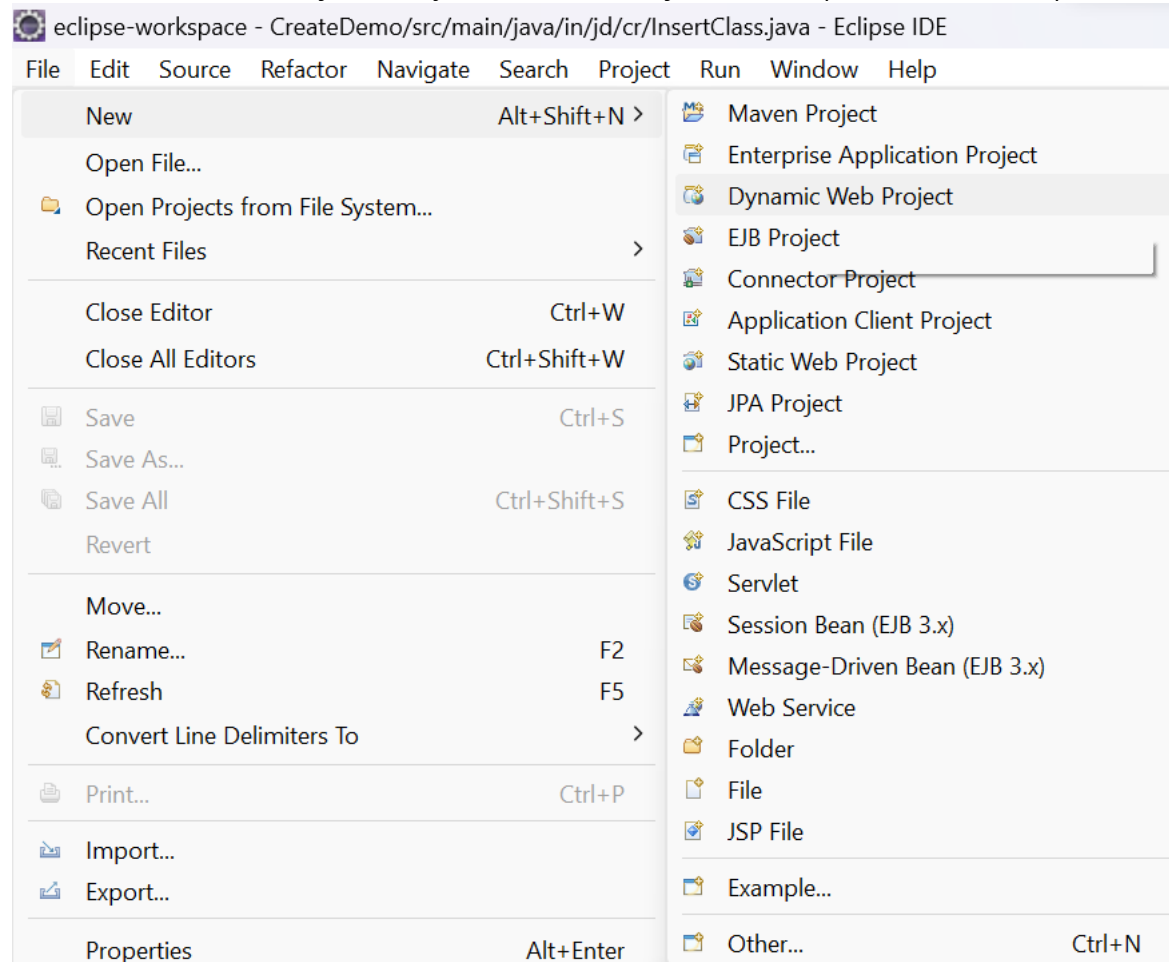


Image: Steps to Create Dynamic Web Project

Step 3: Click on CreateDemo folder -> Java Resources -> src/main/java -> new -> class -> Enter (Class Name and Package Name ex. **InsertClass** and **in.jd.cr**)

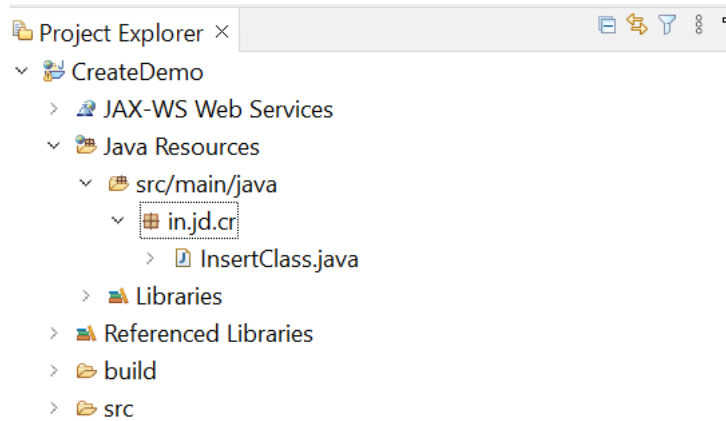


Image: Folder Hierarchy for CreateDemo Project

Step 4: Write Down the Code

Here, we are creating table i.e. **jdbc_emp4** in a database i.e. **jdbc_db**.

InsertClass.java

```
package in.jd.cr;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class InsertClass {

    public static void main(String[] args) throws Exception{
        //loading driver class (Load and Register Driver) - 1
        Class.forName("com.mysql.cj.jdbc.Driver");

        // Connection Create - 2
        Connection con =DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc_db",
"db_user_name", "db_password");
        //System.out.println("Done");

        //create a statement - 3
        PreparedStatement ps = con.prepareStatement("create table jdbc_emp4(id int, name
varchar(30))");

        //execute statement - 4
        ps.executeUpdate();
        System.out.println("Query Execute Succesfully");

        //Close connection - 5
        con.close();
    }
}
```

Step 5: Insert the mysql-connector jar file
click on Project Folder (CreateDemo) -> Build Path -> Configure builds path -> Libraries -> classpath -> add external jars ->(mysql-connector jar file) -> Apply and Close.

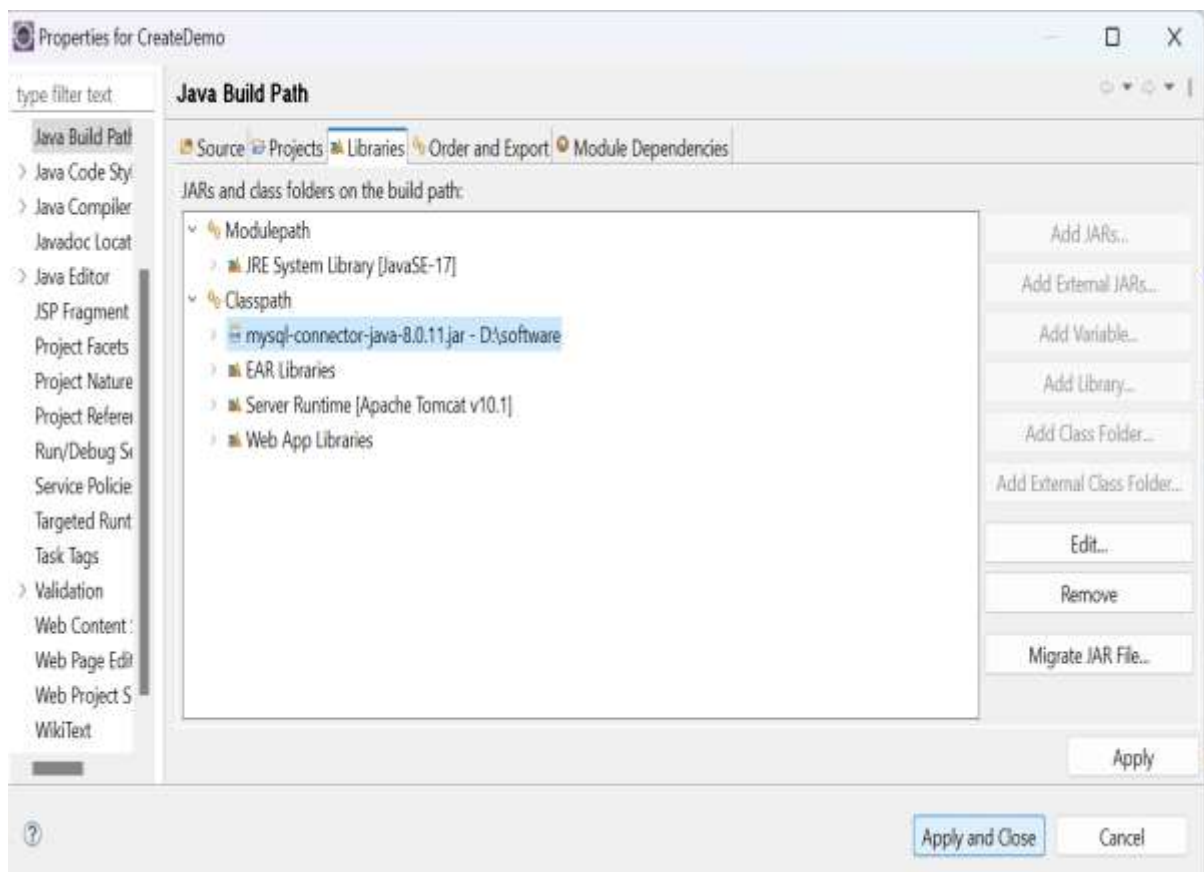


Image: mysql-connector-java.jar insertion

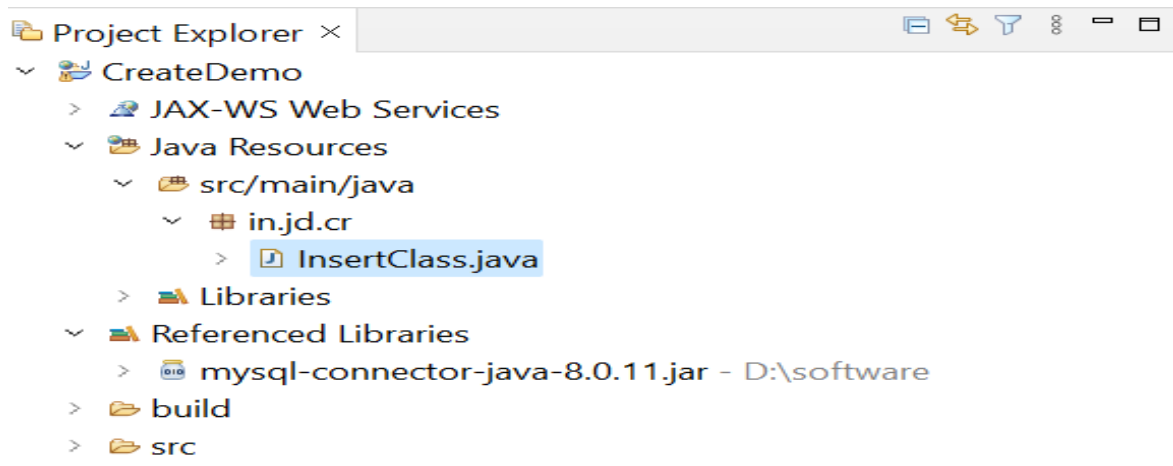


Image: Folder Hierarchy for CreateDemo Project after inserting jar file.

Step 6: Create a Database in MySQL

- I. Open MySQL command line client. - > Enter Database password
- II. Execute the following command to create a database.
`CREATE DATABASE database_name;`
 Ex: **CREATE DATABASE jdbc_db;**
 (In our Example we database_name is **jdbc_db**)

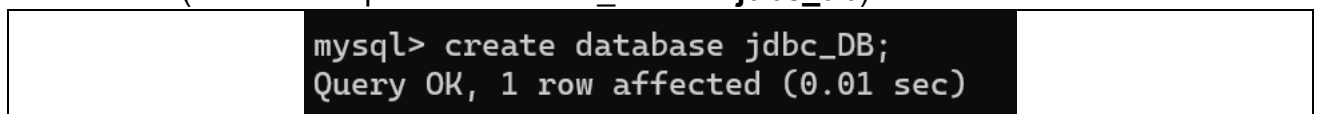


Image: Screenshot of MySQL command line clients.

Step 7: Run the InsertDemo.java file

Select Class File (InsertDemo.java) -> Run as -> Java Application.

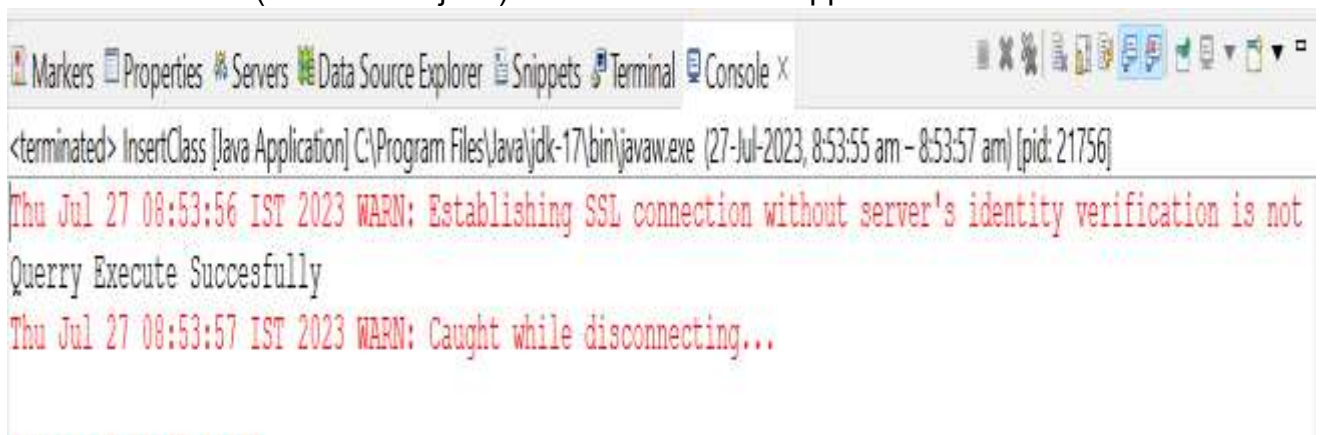


Image: Screenshot of Eclipse IDE Console

Steps to Find Out Whether Table Created or Not in MySQL Database.

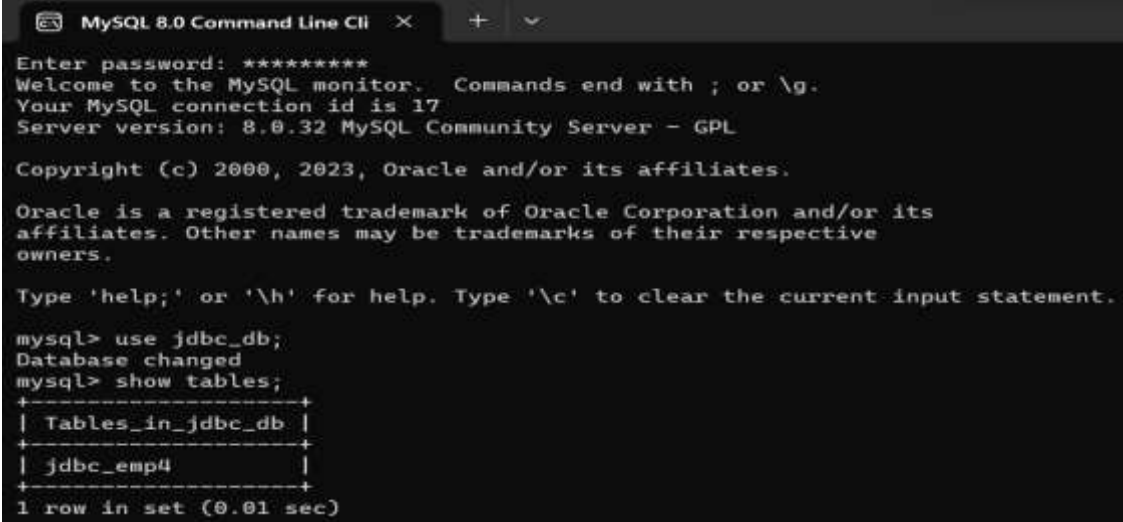
Step 1: Open MySQL command line client. - > Enter Database password

Step 2: Execute the command to select the database

➤ `USE database_name;`

Ex. `USE jdbc_db;`

➤ `SHOW TABLES;`



```
MySQL 8.0 Command Line Cli  X  +  v
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.32 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use jdbc_db;
Database changed
mysql> show tables;
+-----+
| Tables_in_jdbc_db |
+-----+
| jdbc_emp4         |
+-----+
1 row in set (0.01 sec)
```

Image: Screenshot of MySQL command line clients.

Hands-on 2: Insert Data in MySQL using JDBC

Step 1: Open Eclipse IDE

Step 2: Create New Project

File-> New Project ->Dynamic Web Project Name (as InsertDataDemo)-> Save

Step 3: Click on InsertDataDemo folder -> Java Resources -> src/main/java -> new -> class -> Enter (Class Name and Package Name ex. **InsertDataClass** and **in.jd.insert**)

Step 4: Write Down the Code

Here, we are inserting data in **jdbc_emp4** table that is available in **jdbc_db** Database.

InsertDataClass.java

```
package in.jd.insert;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class InsertDataClass {

    public static void main(String[] args) throws Exception{

        //loading driver class (Load and Register Driver) - 1
        Class.forName("com.mysql.cj.jdbc.Driver");

        // Connection Create - 2
        Connection con =DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc_db", "db_username",
        "db_password");

        //create a statement - 3
        PreparedStatement ps = con.prepareStatement("insert into jdbc_emp4(id,name)values(01, 'S Y
        Narayana')");

        //execute statement - 4
        int i = ps.executeUpdate();

        if(i>0)
        {
            System.out.println("Data Inserted");
        }
        else
        {
            System.out.println("Error: Something went wrong");
        }
    }
}
```

```
//Close connection - 5
con.close();
}
}
```

Step 5: Insert the mysql-connector jar file

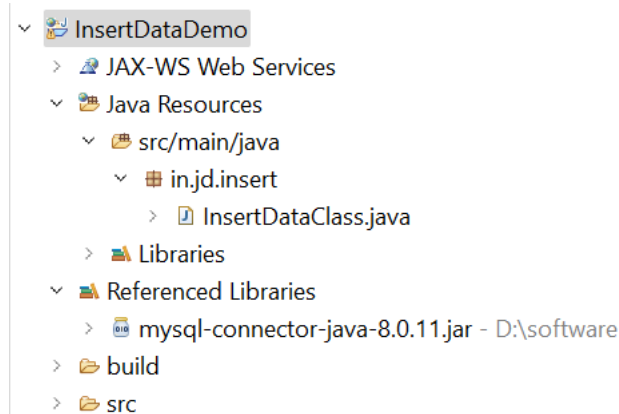


Image: Folder Hierarchy for InsertDataDemo Project after inserting jar file.

Step 6: Run the InsertDataClass.java file

Output:



Image: Screenshot of Eclipse IDE Console

Steps to Find Out Whether Data inserted or not in MySQL Database.

Step 1: Open MySQL command line client. - > Enter Database password

Step 2: Execute the command to select the database

- `USE database_name;`
- Ex. `USE jdbc_db;`
- `SELECT * FROM table_name;`
- Ex. `SELECT * FROM jdbc_emp4;`

```
MySQL 8.0 Command Line Cli  X  +  v

Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 25
Server version: 8.0.32 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use jdbc_db;
Database changed
mysql> select * from jdbc_emp4;
+-----+-----+
| id   | name       |
+-----+-----+
|    1 | S Y Narayana |
+-----+-----+
1 row in set (0.01 sec)
```

Image: Screenshot of MySQL command line clients.

Hands-on 3: Update and Delete Data in MySQL using JDBC

Step 1: Open Eclipse IDE

Step 2: Create a new Project

File-> New Project ->Dynamic Web Project Name (as UpdateDeleteDataDemo)-> Save

Step 3:

Create a Class for Update Data

Click on UpdateDeleteDataDemo Folder -> Java Resources -> src/main/java -> new -> class -> Enter (Class Name and Package Name ex. UpdateDataClass and in.jd.update)

Create a Class for Delete Data

Click on UpdateDeleteDataDemo Folder -> Java Resources -> src/main/java -> new -> class -> Enter (Class Name and Package Name ex. DeleteDataClass and in.jd.delete)

Step 4: Write Down the Code

Code 1) Here, we are updating data in **jdbc_emp4** table that is available in **jdbc_db** Database.

UpdateDataClass.java

Code 2) Here, we are deleting data in **jdbc_emp4** table that is available in **jdbc_db** Database.

DeleteDataClass.java

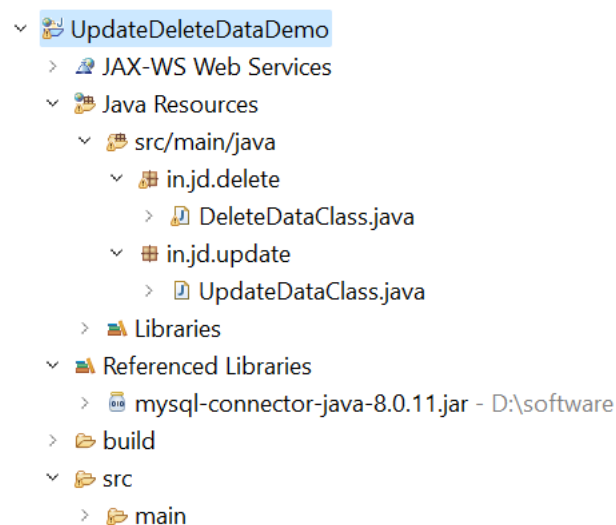


Image: Folder Hierarchy for UpdateDeleteDataDemo Project

Step 5: Insert the mysql-connector jar file

UpdateDataClass.java

```
package in.jd.update;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class UpdateDataClass {

    public static void main(String[] args) throws Exception {
        int id1=1;
        String name1="S Y Mehata";

        Class.forName("com.mysql.cj.jdbc.Driver");

        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc_emp4","db_username","db_password");

        PreparedStatement ps =con.prepareStatement("update jdbc_emp4 set name=? where id=?");
        ps.setString(1, name1);
        ps.setInt(2, id1);

        int count = ps.executeUpdate();
        if (count > 0)
        {
            System.out.println("Update Successfully");
        }else {
            System.out.println("Error: Something Went Wrong");
        }

        con.close();
        System.out.println("Connection Close");
    }
}
```

Step 6.1: Run the UpdateDataClass.java file

Output on Eclipse Console:

The screenshot shows the Eclipse IDE's console window. At the top, there are tabs for 'Markers', 'Properties', 'Servers', 'Data Source Explorer', 'Snippets', 'Terminal', and 'Console'. The console output shows a Java application 'UpdateDataClass' running. It starts with a warning about establishing an SSL connection without identity verification. Then, it shows an 'Update Successfully' message. This is followed by another warning about catching an exception while disconnecting. Below this, an 'EXCEPTION STACK TRACE:' is displayed. The exception is 'javax.net.ssl.SSLException' with the message 'closing inbound before receiving peer's close_notify'. The stack trace lists several frames, including 'SSLSocketImpl.shutdownInput', 'NativeProtocol.quit', 'NativeSession.quit', 'ConnectionImpl.realClose', 'ConnectionImpl.close', and 'UpdateDataClass.main'. The console ends with the text 'Connection Close'.

```
<terminated> UpdateDataClass [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (27-Jul-2023, 11:35:29 am - 11:35:31 am) [pid: 26956]
Thu Jul 27 11:35:30 IST 2023 WARN: Establishing SSL connection without server's identity verification is not recommended
Update Successfully
Thu Jul 27 11:35:31 IST 2023 WARN: Caught while disconnecting...

EXCEPTION STACK TRACE:

** BEGIN NESTED EXCEPTION **

javax.net.ssl.SSLException
MESSAGE: closing inbound before receiving peer's close_notify

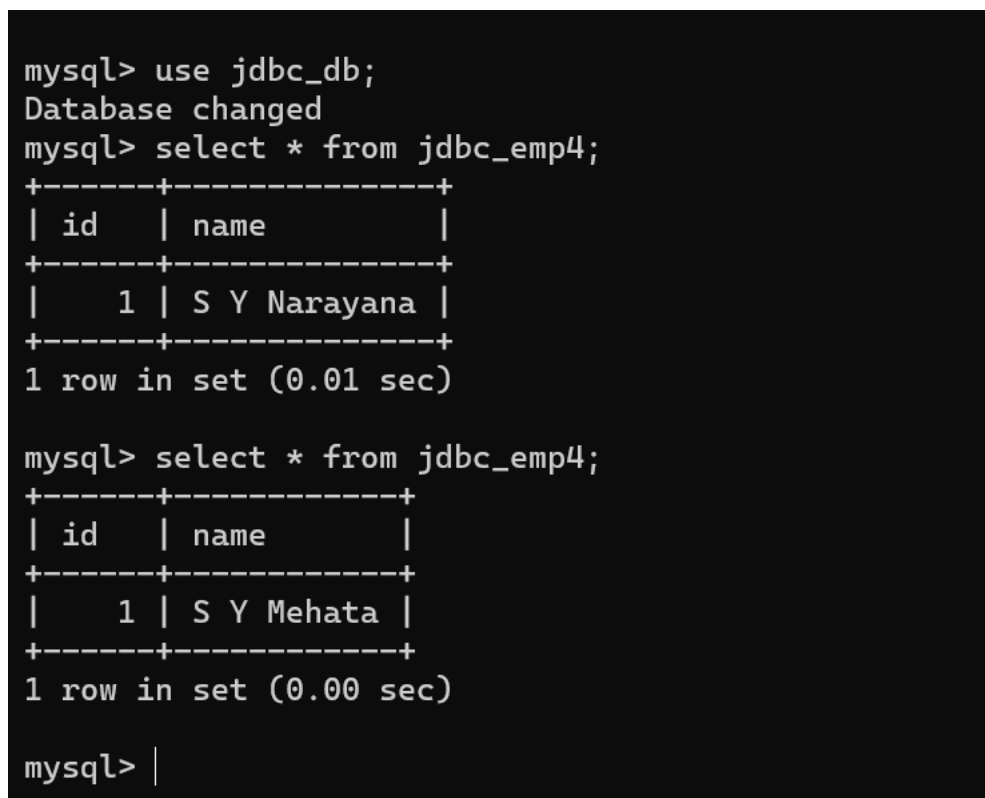
STACKTRACE:
javax.net.ssl.SSLException: closing inbound before receiving peer's close_notify
    at java.base/sun.security.ssl.SSLSocketImpl.shutdownInput(SSLSocketImpl.java:843)
    at java.base/sun.security.ssl.SSLSocketImpl.shutdownInput(SSLSocketImpl.java:821)
    at com.mysql.cj.protocol.a.NativeProtocol.quit(NativeProtocol.java:1294)
    at com.mysql.cj.NativeSession.quit(NativeSession.java:182)
    at com.mysql.cj.jdbc.ConnectionImpl.realClose(ConnectionImpl.java:1911)
    at com.mysql.cj.jdbc.ConnectionImpl.close(ConnectionImpl.java:730)
    at in.jd.update.UpdateDataClass.main(UpdateDataClass.java:32)

** END NESTED EXCEPTION **

Connection Close
```

Image: Screenshot of Eclipse IDE Console

MySQL Command Line Clients Result

The screenshot shows a MySQL command-line interface. The user enters 'mysql> use jdbc_db;' and the prompt changes to 'Database changed'. Then, the user enters 'mysql> select * from jdbc_emp4;'. The result is displayed as a table with two columns: 'id' and 'name'. The first row shows '1' for id and 'S Y Narayana' for name. Below the table, it says '1 row in set (0.01 sec)'. The user then enters 'mysql> select * from jdbc_emp4;' again. The result is a table with 'id' and 'name' columns. The first row shows '1' for id and 'S Y Mehata' for name. Below the table, it says '1 row in set (0.00 sec)'. Finally, the user enters 'mysql> |' and the prompt is ready for the next command.

```
mysql> use jdbc_db;
Database changed
mysql> select * from jdbc_emp4;
+-----+-----+
| id    | name          |
+-----+-----+
|      1 | S Y Narayana  |
+-----+-----+
1 row in set (0.01 sec)

mysql> select * from jdbc_emp4;
+-----+-----+
| id    | name          |
+-----+-----+
|      1 | S Y Mehata    |
+-----+-----+
1 row in set (0.00 sec)

mysql> |
```

Image: Screenshot of MySQL command line clients

DeleteDataClass.java

```
package in.jd.delete;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class DeleteDataClass {

    public static void main(String[] args) throws Exception {
        //load Driver
        Class.forName("com.mysql.cj.jdbc.Driver");

        //connection
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc_db","db_username","db_pass
word");

        //create statement

        PreparedStatement ps = con.prepareStatement("delete from jdbc_emp4 where
id=1");

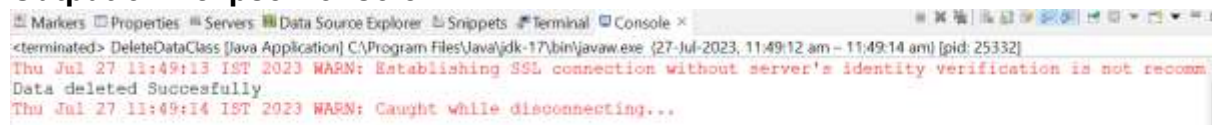
        //execute statement
        int count =ps.executeUpdate();

        if(count>0) {System.out.println("Data deleted Successfully");}
        else {System.out.println("Error: Something went Wrong");}

        //close connection
        con.close();
    }
}
```

Step 6.2: Run the DeleteDataClass.java file

Output on Eclipse Console:



```
<terminated> DeleteDataClass [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (27-Jul-2023, 11:49:12 am - 11:49:14 am) [pid: 25332]
Thu Jul 27 11:49:13 IST 2023 WARN: Establishing SSL connection without server's identity verification is not recom
Data deleted Successfully
Thu Jul 27 11:49:14 IST 2023 WARN: Caught while disconnecting...
```

Image: Screenshot of Eclipse IDE Console

MySQL Command Line Clients Result:

```
mysql> select * from jdbc_emp4;
+-----+-----+
| id    | name      |
+-----+-----+
|      1 | S Y Mehata |
+-----+-----+
1 row in set (0.00 sec)

mysql> select * from jdbc_emp4;
Empty set (0.00 sec)
```

Image: Screenshot of MySQL command line clients

Hands-on 4: Data Retrieve from MySQL using JDBC

Step 1: Open Eclipse IDE

Step 2: Create New Project

File-> New Project ->Dynamic Web Project Name (as RetriveDataDemo)-> Save

Step 3:

Create a Class

Click on RetriveDataDemo Folder -> Java Resources -> src/main/java -> new -> class -> Enter (Class Name and Package Name ex. SelectDataClass and in.jd.select)

Step 4: Write Down the Code

Here, we retrieve data from the **jdbc_emp4** table that is available in **jdbc_db** Database.

Step 5: Insert the mysql-connector jar file.

Step 6: Run the SelectDataClass.java file

Note: Before Executing the SelectDataClass.java file we need to insert a record in the **jdbc_emp4** table.

As per follow,

```
MySQL 8.0 Command Line Cli  X  +  v

mysql> select * from jdbc_emp4;
Empty set (0.00 sec)

mysql> insert into jdbc_emp4(id, name) values (1, "R M Mehata");
Query OK, 1 row affected (0.01 sec)

mysql> insert into jdbc_emp4(id, name) values (2, "O N Nabbi");
Query OK, 1 row affected (0.01 sec)

mysql> insert into jdbc_emp4(id, name) values (3, "T N Saxenna");
Query OK, 1 row affected (0.01 sec)

mysql> insert into jdbc_emp4(id, name) values (4, "R D Varma");
Query OK, 1 row affected (0.01 sec)

mysql> insert into jdbc_emp4(id, name) values (5, "S K Ramarao");
Query OK, 1 row affected (0.01 sec)

mysql> select * from jdbc_emp4;
+-----+-----+
| id    | name          |
+-----+-----+
| 1     | R M Mehata    |
| 2     | O N Nabbi     |
| 3     | T N Saxenna   |
| 4     | R D Varma     |
| 5     | S K Ramarao   |
+-----+-----+
5 rows in set (0.00 sec)

mysql> |
```

Image: Screenshot of MySQL command line clients

SelectDataClass.java

```
package in.jd.select;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class SelectDataClass {

    public static void main(String[] args) throws Exception{
        //load Driver
        Class.forName("com.mysql.cj.jdbc.Driver");
```

```

//create Connection
Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc_db","db_username","db_password");

//Create Statement
PreparedStatement ps= con.prepareStatement("select * from jdbc_emp4");

//execute statement-Query
ResultSet rs=ps.executeQuery();

while(rs.next()) {
    int id1=rs.getInt("id");
    String name1=rs.getString("name");
    System.out.println("Id of emp = "+id1+" Name of Emp = "+name1);
}
}
}

```

Output on Eclipse Console:



```

<terminated> SelectDataClass [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (27-Jul-2023, 12:07:30 pm - 12:07:31 pm) [pid: 33476]
Thu Jul 27 12:07:30 IST 2023 WARN: Establishing SSL connection without server's identity verification is not recommended.
Id of emp = 1 Name of Emp = R M Mehata
Id of emp = 2 Name of Emp = O N Nabbi
Id of emp = 3 Name of Emp = T N Saxenna
Id of emp = 4 Name of Emp = R D Varma
Id of emp = 5 Name of Emp = S K Ramarao

```

Image: Screenshot of Eclipse IDE Console