

## **Database Management Systems Lab Experiment No: - 12**

### **Aim: - Study of Database Connectivity using JDBC**

JDBC stands for **Java Database Connectivity**, which is a standard Java API for database-independent connectivity between the Java programming language and a wide range of databases.

The JDBC library includes APIs for each of the tasks mentioned below that are commonly associated with database usage.

- Making a connection to a database.
- Creating SQL or MySQL statements.
- Executing SQL or MySQL queries in the database.
- Viewing & Modifying the resulting records.

### **JDBC Architecture**

The JDBC API supports both two-tier and three-tier processing models for database access but in general, JDBC Architecture consists of two layers –

- **JDBC API:** This provides the application-to-JDBC Manager connection.
- **JDBC Driver API:** This supports the JDBC Manager-to-Driver Connection.

The JDBC API uses a driver manager and database-specific drivers to provide transparent connectivity to heterogeneous databases.

The JDBC driver manager ensures that the correct driver is used to access each data source. The driver manager is capable of supporting multiple concurrent drivers connected to multiple heterogeneous databases.

Following is the architectural diagram, which shows the location of the driver manager with respect to the JDBC drivers and the Java application –

### **Common JDBC Components**

The JDBC API provides the following interfaces and classes –

- **DriverManager:** This class manages a list of database drivers. Matches connection requests from the java application with the proper database driver using communication sub protocol. The first driver that recognizes a certain subprotocol under JDBC will be used to establish a database Connection.

## Database Management Systems Lab Experiment No: - 12

### Aim: - Study of Database Connectivity using JDBC

- **Driver:** This interface handles the communications with the database server. You will interact directly with Driver objects very rarely. Instead, you use DriverManager objects, which manages objects of this type. It also abstracts the details associated with working with Driver objects.
- **Connection:** This interface with all methods for contacting a database. The connection object represents communication context, i.e., all communication with database is through connection object only.
- **Statement:** You use objects created from this interface to submit the SQL statements to the database. Some derived interfaces accept parameters in addition to executing stored procedures.
- **ResultSet:** These objects hold data retrieved from a database after you execute an SQL query using Statement objects. It acts as an iterator to allow you to move through its data.
- **SQLException:** This class handles any errors that occur in a database application.

#### Steps for creating Database Connectivity Applications:

There are mainly six steps:

Step1: Import the packages Required for Database Programming.

Step2: Register the JDBC Driver to JDBC Driver manager

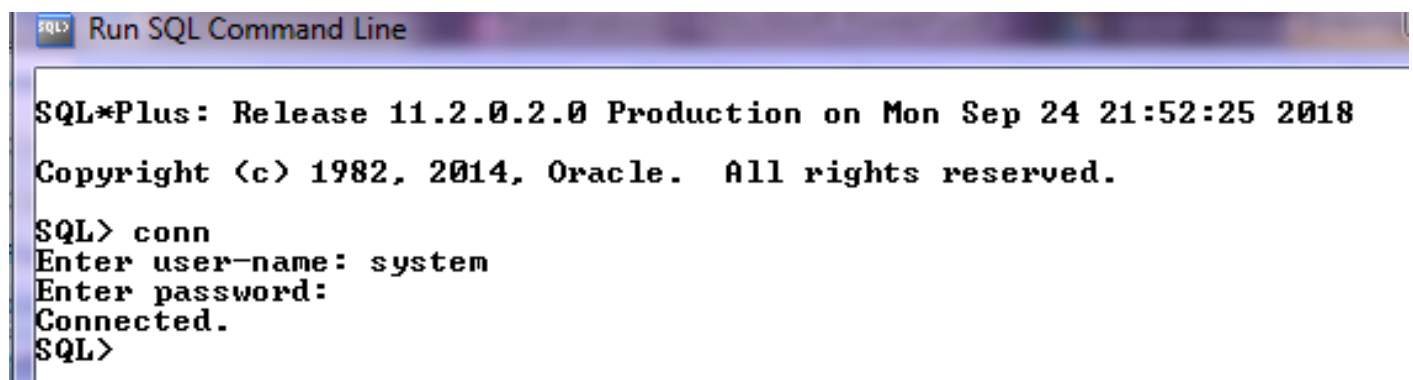
Step3: Open a connection

Step4: Execute a Query

Step5: Extract Data from Result set

Step6: Close Connection.

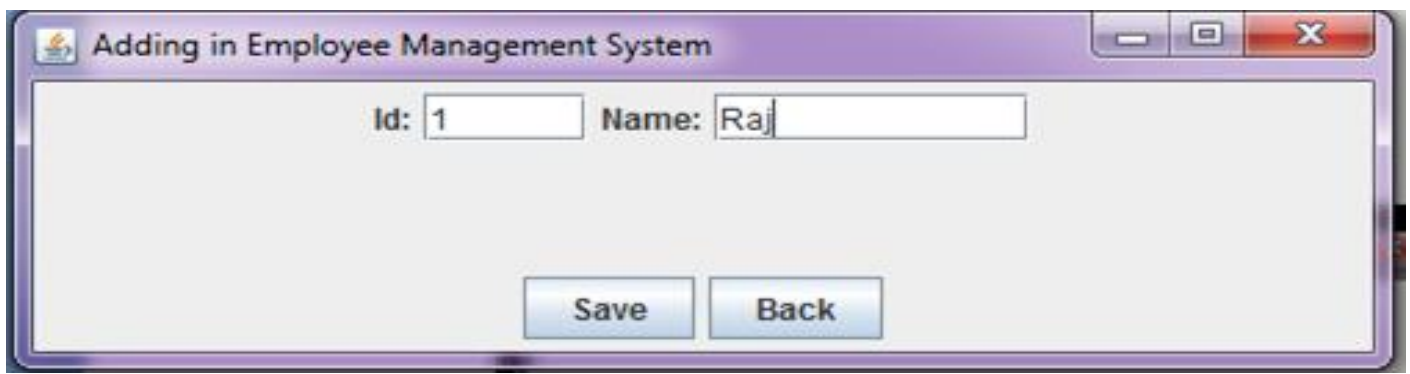
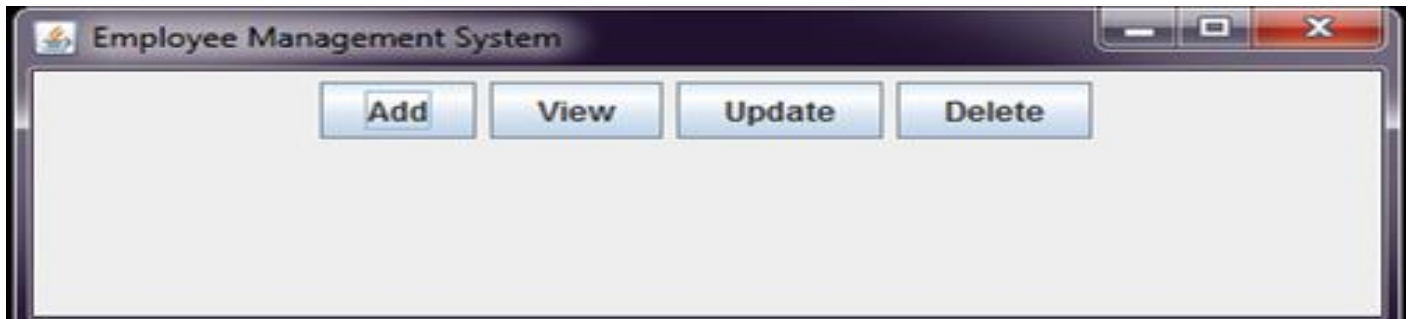
#### Output:-



```
SQL> Run SQL Command Line
SQL*Plus: Release 11.2.0.2.0 Production on Mon Sep 24 21:52:25 2018
Copyright (c) 1982, 2014, Oracle. All rights reserved.
SQL> conn
Enter user-name: system
Enter password:
Connected.
SQL>
```

Database Management Systems Lab Experiment No: - 12  
Aim: - Study of Database Connectivity using JDBC

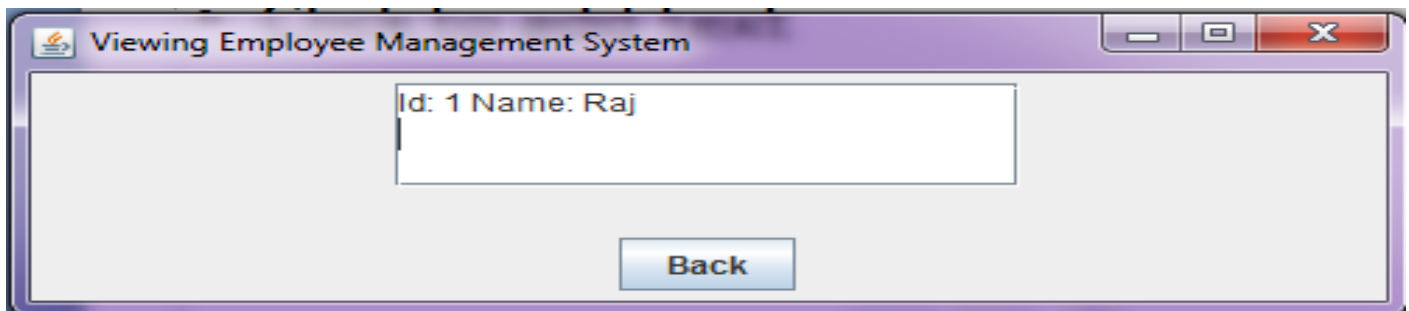
```
SQL> create table employee(eid integer primary key,ename varchar2(200));  
Table created.  
SQL>
```



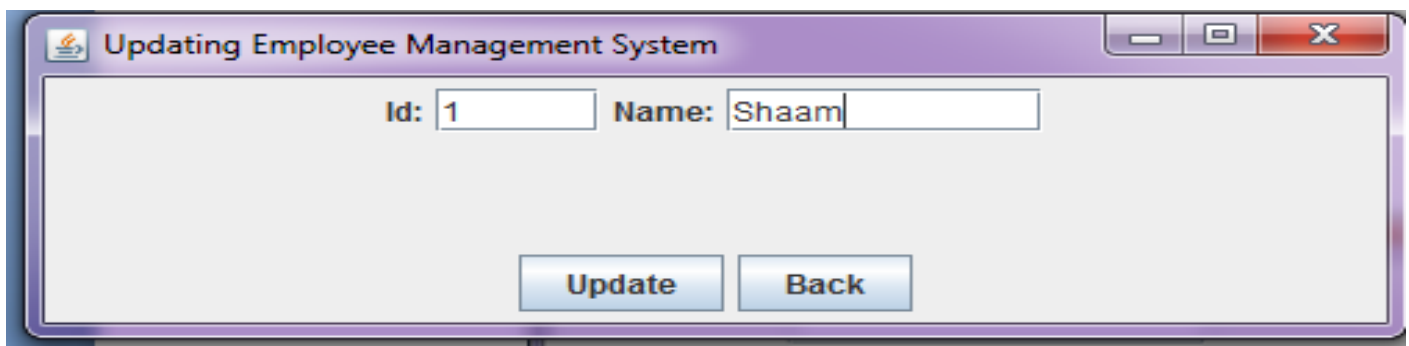
```
SQL> select * from employee;
```

EID	ENAME
1	Raj

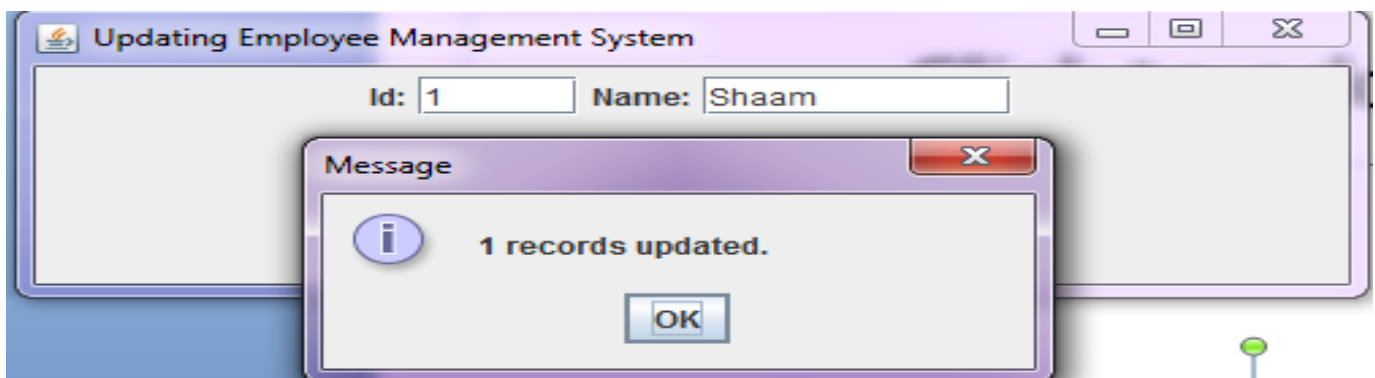
Database Management Systems Lab Experiment No: - 12  
Aim: - Study of Database Connectivity using JDBC



The screenshot shows a window titled "Viewing Employee Management System". Inside, there is a text box displaying "Id: 1 Name: Raj". Below the text box is a button labeled "Back".



The screenshot shows a window titled "Updating Employee Management System". It contains two input fields: "Id: 1" and "Name: Shaam". Below these fields are two buttons: "Update" and "Back".



The screenshot shows the "Updating Employee Management System" window with "Id: 1" and "Name: Shaam". A "Message" dialog box is overlaid on top, displaying an information icon, the text "1 records updated.", and an "OK" button.

```
SQL> select * from employee;
```

```
      EID  
-----  
ENAME
```

```
Raj      1
```

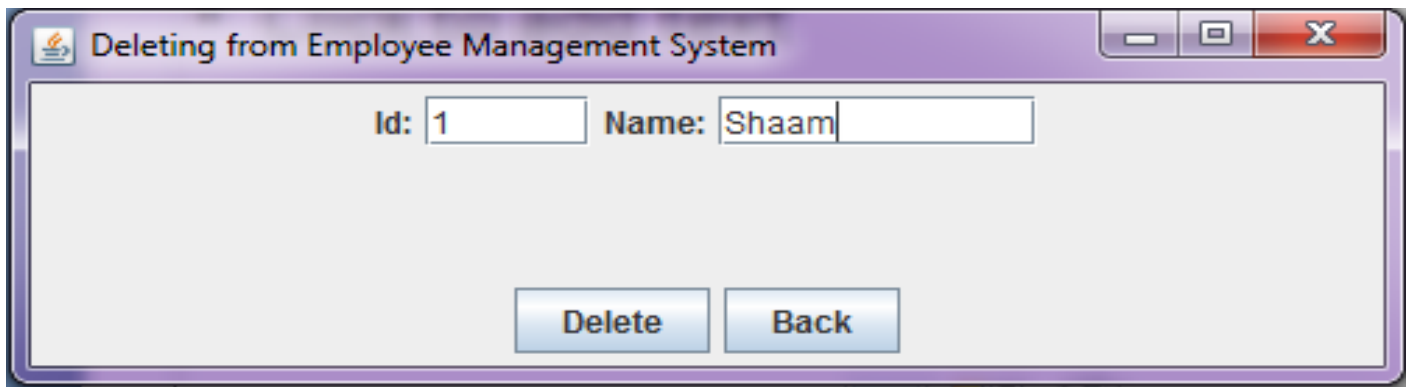
```
SQL> select * from employee;
```

```
      EID  
-----  
ENAME
```

```
Shaam    1
```

```
SQL>
```

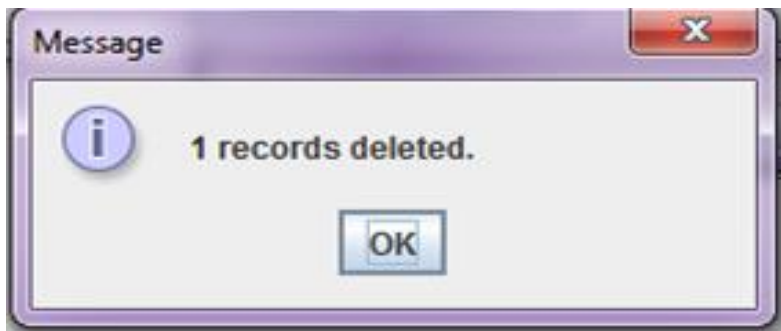
**Database Management Systems Lab Experiment No: - 12**  
**Aim: - Study of Database Connectivity using JDBC**



Deleting from Employee Management System

Id: 1 Name: Shaam

Delete Back



```
SQL> select * from employee;
```

```
      EID
-----
ENAME
```

```
      1
Raj
```

```
SQL> select * from employee;
```

```
      EID
-----
ENAME
```

```
      1
Shaam
```

```
SQL> select * from employee;
```

```
no rows selected
```

```
SQL>
```

Conclusion:- LO mapped LO5.