SESSION 10

JDBC: Java Database Connectivity

Java Database Connectivity (JDBC) is a standard Java API for database-independent connectivity between the Java programming language and a wide range of databases.

Creating JDBC Application:

There are several steps involved in building a JDBC application:

Import the packages:

This requires that you include the packages containing the JDBC classes needed for database programming. Most often, using import java.sql.* will suffice as follows:

```
//STEP 1. Import required packages
import java.sql.*;
```

Register the JDBC driver:

This requires that you initialize a driver so you can open a communications channel with the database. Following is the code snippet to achieve this:

```
//STEP 2: Register JDBC driver
Class.forName("com.mysql.jdbc.Driver");
```

Open a connection:

This requires using the DriverManager.getConnection() method to create a Connection object, which represents a physical connection with the database as follows:

```
//STEP 3: Open a connection
//  Database credentials - replace with the credentials to
your installation
static final String USER = "username";
static final String PASS = "password";
System.out.println("Connecting to database...");
conn = DriverManager.getConnection(DB_URL,USER,PASS);
```

Execute a query:

This requires using an object of type Statement or PreparedStatement for building and submitting an SQL statement to the database as follows:

```
//STEP 4: Execute a query

System.out.println("Creating statement...");
stmt = conn.createStatement();
String sql;
sql = "SELECT fname, minit, lname FROM Employee";
ResultSet rs = stmt.executeQuery(sql);
```

If there is an SQL UPDATE,INSERT or DELETE statement required, then following code snippet would be required:

```
//STEP 4: Execute a query
System.out.println("Creating statement...");
stmt = conn.createStatement();
String sql; sql = "DELETE FROM Employee";
ResultSet rs = stmt.executeUpdate(sql);
```

Extract data from result set:

This step is required in case you are fetching data from the database. You can use the appropriate ResultSet.getXXX() method to retrieve the data from the result set as follows:

```
//STEP 5: Extract data from result set while(rs.next()){
//Retrieve by column name
int ssn  = rs.getInt("ssn");
String fname = rs.getString("fname");
String last = rs.getString("lname");
//Display values
System.out.print("SSN: " + ssn);
System.out.print(", First Name: " + fname);
System.out.println(", Last Name: " + lname); }
```

Clean up the environment:

You should explicitly close all database resources versus relying on the JVM's garbage collection as follows:

```
//STEP 6: Clean-up environment
rs.close();
stmt.close();
conn.close();
```