**// Java program to execute a query using PreparedStatement**

**import java.sql.\*;**

**public class JDBCTest {**

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

**// 1- Register Driver Class**

**Class.forName("org.apache.derby.jdbc.ClientDriver");**

**// 2- Connection to your database**

**Connection con = DriverManager.getConnection();**

**// 3- Query statement which needs parameters (Could by procedure)**

**String query**

**= "Select \* from students where age> ? and name = &";**

**// 4- Prepare your Statement**

**PreparedStatement myStmt = con.prepareStatement(query);**

**// 5- Set Parameters**

**myStmt.setInt(1, 20);**

**myStmt.setStrin(2, 'Prateek');**

**// 6- Execute SQL query**

**ResultSet myRs = myStmt.executeQuery();**

**System.out.println('Age Name');**

**// Display function to show the Resultset**

**while (myRs.next()) { // still there are rows to get from the dataset**

**String Name = rs.getString("name");**

**int age = rs.getInt("age");**

**System.out.println(Name + " " + age);**

**}**

**// 7- Close the connection**

**con.close();**

**}**

**}**

**// Call a PL/SQL stored procedure or stored Function**

**import java.sql.\*;**

**import java.io.\*;**

**public class PLSQLExample** **{**

**public static void main (String args [])** **throws SQLException, IOException**

**{**

**// 1- Load the driver**

**DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());**

**// Connect to the database**

**// You can put a database name after the @ sign in the connection URL.**

**Connection conn =**

**DriverManager.getConnection ("jdbc:oracle:DBNAME:@", "USER", "PASSWORD");**

**// Create a statement**

**Statement stmt = conn.createStatement ();**

**// Create the stored function (Create DDL at runtime)**

**stmt.execute ("create or replace function RAISESAL (name CHAR, raise NUMBER)**

**return NUMBER is begin return raise + 100000; end;");**

**// Close the statement**

**stmt.close();**

**// Prepare to call the stored procedure RAISESAL.**

**CallableStatement cstmt = conn.prepareCall ("{? = call RAISESAL (?, ?)}");**

**// Declare that the first ? is a return value of type Int**

**cstmt.registerOutParameter (1, Types.INTEGER);**

**// We want to raise LESLIE's salary by 20,000**

**cstmt.setString (2, "LESLIE"); // The name argument is the second ?**

**cstmt.setInt (3, 20000); // The raise argument is the third ?**

**// Do the raise**

**cstmt.execute ();**

**// Get the new salary back**

**int new\_salary = cstmt.getInt (1);** **// retrieve value manipulated by statement.**

**System.out.println ("The new salary is: " + new\_salary);**

**// Close the statement**

**cstmt.close();**

**// Close the connection**

**conn.close();**

**}**

**}**

**// Call Oracle API from Java Code**

*import java.sql.CallableStatement;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.Types;*

*public class OrderInformation {*

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

*//call API for ORDER\_ID 101  
System.out.println(getOrderStatus(“jdbc:oracle:thin:@localhost:1521:DBNAME”,”USERNAME”,”PASSWORD”,101));}*

*public static String getOrderStatus(String DBURL, String User, String Password,int orderId) throws Exception  
{  
String message;  
try {  
Connection con = DriverManager.getConnection(DBURL, User, Password);  
System.out.println(“Connected to database”);*

*String command =* ***“{call SYSTEM.PRC\_GET\_ORDER\_STATUS(?,?)}”;***

*CallableStatement cstmt =  con.prepareCall(command);*

*cstmt.setInt(1, orderId);  
cstmt.registerOutParameter(2, Types.VARCHAR);*

*cstmt.execute();  
message=cstmt.getString(2);  
cstmt.close();*

*}  
catch(Exception ex) {  
message= ex.getMessage(); }  
return message;  
} }*