**Step 3 🡪 Create a Statement:**

* In order to send the SQL commands to database from our java program, we need Statement object. This can be done in following three ways:

|  |  |  |  |
| --- | --- | --- | --- |
| Sl.No | Method | Return Type Object |  |
| 1 | createStatement() | Statement | Used to execute sql queries |
| 2 | PreparedStatement() | PreparedStatement |
| 3 | prepareCall() | CallableStatement | Used to execute the procedure or function |

What is the difference between Statement and Prepared Statement?

|  |  |
| --- | --- |
| Statement | PreparedStatement |
| Used for executing simple SQL statements like CRUD (create, retrieve, update and delete | Used for executing dynamic and pre-compiled SQL statements |
| Statement interface cannot accept parameters at run time. | PreparedStatement interface accepts input parameters at runtime. |
| Statement is slower as compared to Prepared Statement | PreparedStatement is faster because it is used for executing pre-compiled SQL statement |
| Statement is suitable for executing DDL commands – CREATE, drop, alter and truncate | PreparedStatement is suitable for executing DML commands – SELECT, INSERT, UPDATE and DELETE |
| It is a base Interface. | It is Extends statement interface. |
| Statement can not use for reading binary data. | PreparedStatement used for reading binary data. |
| Statement is static | PreparedStatement is dynamic. |
| Statement is usually parsed and executed each time. | PreparedStatement is parsed once and executed with different parameters repeatedly. |
| Statement verifies metadata against database every time. | PreparedStatement verifies metadata against the database only once. |
| Statement st = con.createStatement();  .  .  .  st.executeUpdate(SQLQuery) | PreparedStatement pst =  con.preparedStatement(SQLQuery);  .  .  .  pst.executeUpdate() |

Write a JDBC program to insert employee information in employee table using PreparedStatement

Using Statement

Statement stmt = con.createStatement();

Using PreparedStement

Before creating the object of Prepared statement we must define the template of the sql query we want to execute

Step 1:

**final** String SQLQUERY = "insert into employee"

+ "(empno,ename,job,mgr,sal,comm,deptno) "

+ "values(?,?,?,?,?,?,?)";

Step 2:

Create the object of Prepared statement by passing the SQL query template as an argument.

PreparedStatement pstmt = con.prepareStatement(SQLQUERY);

Step 3: inject the values to the prepared statement object using setX methods.

pstmt.setInt(1, empno);

pstmt.setString(2, ename);

pstmt.setString(3,desig);

pstmt.setInt(4, mgrid);

pstmt.setDouble(5, salary);

pstmt.setDouble(6, comm);

pstmt.setInt(7, deptid);

Step 4: call the respective execute method

|  |  |  |  |
| --- | --- | --- | --- |
| Prepared Statement | execute() | Boolean | used to write any SQL Query (select or non – select) |
| executeUpdate() | int | used to write execute DML queries(insert or delete or update) |
| executeQuery() | ResultSet | used to write select statement |

Using statement:

stmt.executeUpdate(SQLQUERY);

Using Prepared Statement

**int** result = pstmt.executeUpdate();

Program

**package** com.sssit.jdbcapp3;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.SQLException;

**import** java.sql.Statement;

**import** java.util.Scanner;

**public** **class** InsertUsingPreparedStatement {

**public** **static** **final** String ***URL*** = "jdbc:oracle:thin:@localhost:1521:orcl";

**public** **static** **final** String ***UNAME*** = "advdec23";

**public** **static** **final** String ***PASS*** = "advdec23";

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

**try** {

Class.*forName*("oracle.jdbc.driver.OracleDriver");

System.***out***.println("Class Loaded successfully....");

Connection con = DriverManager.*getConnection*(***URL***, ***UNAME***, ***PASS***);

System.***out***.println("Connection Estd....");

//Statement stmt = con.createStatement();

**final** String SQLQUERY = "insert into employee"

+ "(empno,ename,job,mgr,sal,comm,deptno) "

+ "values(?,?,?,?,?,?,?)";

PreparedStatement pstmt = con.prepareStatement(SQLQUERY);

Scanner ip = **new** Scanner(System.***in***);

System.***out***.println("Enter Employee number:");

**int** empno = ip.nextInt();

System.***out***.println("Enter Employee Name:");

String ename = ip.next();

System.***out***.println("Enter the Designation:");

String desig = ip.next();

System.***out***.println("Enter Manager id:");

**int** mgrid = ip.nextInt();

System.***out***.println("Enter Salary:");

**double** salary = ip.nextDouble();

System.***out***.println("Enter Commission:");

**double** comm = ip.nextDouble();

System.***out***.println("Enter department number:");

**int** deptid = ip.nextInt();

/\*

\* final String SQLQUERY = "insert into employee" +

\* "(empno,ename,job,mgr,sal,comm,deptno) " + "values(" + empno + ",'" + ename +

\* "','" + desig + "'," + mgrid + "," + salary + "," + comm + "," + deptid +

\* ")";

\*/

pstmt.setInt(1, empno);

pstmt.setString(2, ename);

pstmt.setString(3,desig);

pstmt.setInt(4, mgrid);

pstmt.setDouble(5, salary);

pstmt.setDouble(6, comm);

pstmt.setInt(7, deptid);

System.***out***.println(SQLQUERY);

//int result = stmt.executeUpdate(SQLQUERY);

**int** result = pstmt.executeUpdate();

**if**(result>0)

System.***out***.println("Inserted successfully.....");

**else**

System.***out***.println("Insertion Failed.....");

pstmt.close();

con.close();

} **catch** (ClassNotFoundException e) {

System.***out***.println("Failed to Load Driver class...");

} **catch** (SQLException e) {

System.***out***.println("Failed in executing SQL operation");

}

}

}

Write a JDBC Program to update the employee name by id.

**package** com.sssit.jdbcapp3;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.sql.Statement;

**import** java.util.Scanner;

**import** com.sssit.jdbcapp2.EmployeeNotFoundException;

**public** **class** UpdateNameByIdUsingPS {

**public** **static** **final** String ***URL*** = "jdbc:oracle:thin:@localhost:1521:orcl";

**public** **static** **final** String ***UNAME*** = "advdec23";

**public** **static** **final** String ***PASS*** = "advdec23";

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

**try** {

Class.*forName*("oracle.jdbc.driver.OracleDriver");

System.***out***.println("Class Loaded successfully....");

Connection con = DriverManager.*getConnection*(***URL***, ***UNAME***, ***PASS***);

System.***out***.println("Connection Estd....");

//Statement stmt = con.createStatement();

**final** String SELECTQUERY = "select ename from employee "

+ "where empno=?";

**final** String UPDATEQUERY = "update employee set "

+ "ename=? where empno=?";

PreparedStatement selectPStmt = con.prepareStatement(SELECTQUERY);

PreparedStatement updatePStmt = con.prepareStatement(UPDATEQUERY);

Scanner ip = **new** Scanner(System.***in***);

System.***out***.println("Enter Employee number:");

**int** empno = ip.nextInt();

selectPStmt.setInt(1, empno);

ResultSet rs = selectPStmt.executeQuery();

//ResultSet rs = stmt.executeQuery(SELECTQUERY);

**if**(rs.next()) {

System.***out***.println("Existed Name is:" + rs.getString(1));

}

**else**

{

**throw** **new** EmployeeNotFoundException(empno + " Not found in DB");

}

System.***out***.println("Do you want to change the name...");

**char** ch = ip.next().charAt(0);

**if**(ch=='y' || ch=='Y') {

System.***out***.println("Enter New Employee Name:");

String ename = ip.next();

updatePStmt.setString(1,ename);

updatePStmt.setInt(2, empno);

**int** result = updatePStmt.executeUpdate();

//int result = stmt.executeUpdate(SQLQUERY);

**if**(result>0)

System.***out***.println("Updated successfully.....");

**else**

System.***out***.println("Updation Failed.....");

}

rs.close();

selectPStmt.close();

updatePStmt.close();

con.close();

} **catch** (ClassNotFoundException e) {

System.***out***.println("Failed to Load Driver class...");

} **catch** (SQLException e) {

System.***out***.println("Failed in executing SQL operation");

} **catch** (EmployeeNotFoundException e) {

System.***out***.println(e.getMessage());

}

}

}