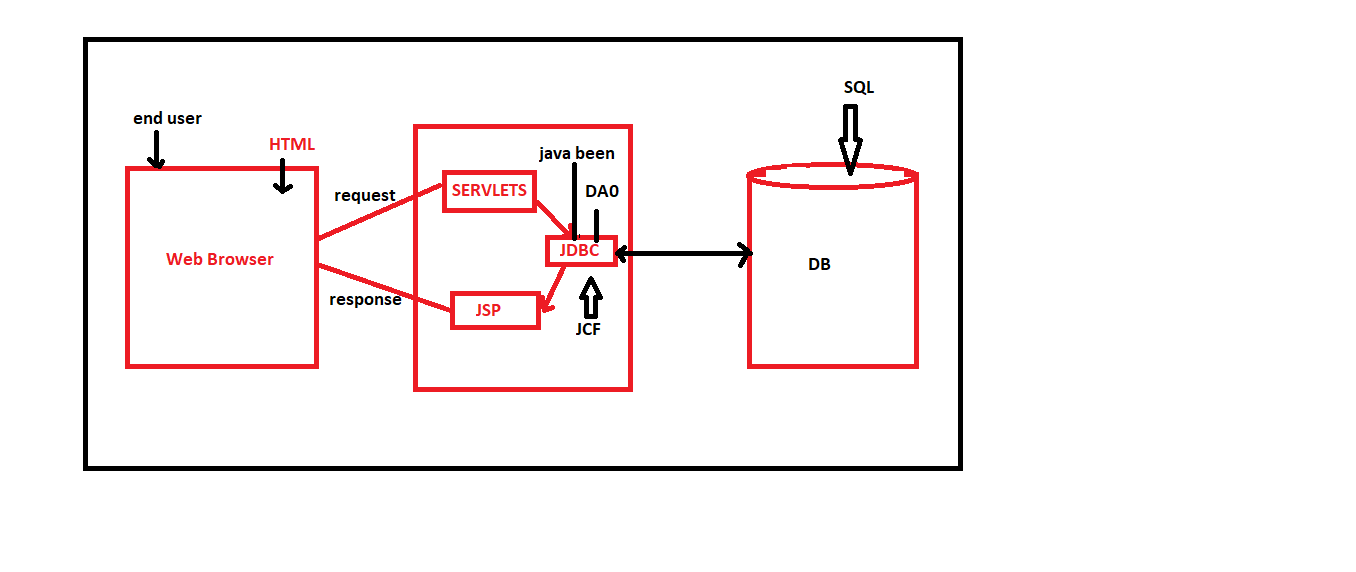
**ADV.JAVA**

* **Application:-** set of programs collected together to perform define action is known as application.
* **Web Application :-** the application which is running in internet environment , or web environment in the process of interfacing with end user through web browser is known as web application.
* The following are three technologies used inconstructing web application.

1. JDBC
2. SERVLETS
3. JSP



* **JDBC:-**  Controlling and manage database product.
* **Servlets:**- Accepting the request coming from web browser
* **JSP:**- Give response to web browser
* **JDBC:-** java database connectivity, which is used to control and manage database product.
* **TYPES OF STORAGE OF JAVA APPLICATION:**-
* **Field Storage**
* **Object Storage**
* **File storage**
* **Database storage**
* **Field Storage:-**  The memory which is generated part of java program to hold single value is known as field storage.

**NOTE:**- This field storage are generated from primitive datatype variable ,(byte, short, int , long, float, double, char and boolean).

EX:- INT K= 10; memory.

10

DataBase

K Variable

* **Object Storage:-** The memory which Is generated part of java program to hold is known as object storage.

**NOTE:**- These object storage or generate when we have Non-primitive datatypes (Class,interface,array ,enum). memory

|  |
| --- |
| -------------------  ------------------- |

EX:- Test t + new Test(); non static member class

Object :- it is a memory create part to Heap area related to a class hiding nonstatic members o the class.

NOTE:- A ccording to java language variable are categorised into two types:-

1. **Primitive datatypes variable**
2. **Non-primitive variables.**

* Primitive datatype variable will hold values
* Non primitive datatypes will hold object refrence.

NOTE:- The field storage and object storage which are generate part of jvm are temporary storages which are destroyed automatically when jvm shutdown. When we want to have permanent storages for java application we use **file storage** and **database storage.**

|  |
| --- |
| *Java program* |

*Java java.io OS (M)*

|  |
| --- |
| *File Storage* |

**Disadavantage of file Storage:-**

* File storage can hold only limited data(small storage).
* The data in file storage can not be oragnised efficiently.This file storage is not preferable for web application.

**DataBase Storage:-** The largest permanent storage which is installed into computer environment is known as database storage (installed from externally)

The data in database storage is organised more efficiently , which is preferable for web application.

|  |  |
| --- | --- |
| |  | | --- | | Java program | |

|  |  |
| --- | --- |
| |  | | --- | | DB storage | |

* When we want to stablish communication B/W java program and database product . the java program must be constructed using clsses aand interfaces available from java.sql package and the java program uses **JDBC driver.**

Java.io

**JDDC DRIVER**

JAVA OS(M)

* **Driver :-** a small s/w program which is used to interact with the resource iof computer system is known as driver.

**EX-**-- > audio driver, video driver….

**NOTE:**- According to computer environment the resources in the computer system are controlled driver programs.

* **API:-** 
  + - java.lang -------- > Language package
    - java.io -----------> Iostream package Core Java
    - java.net -----------> Networkin
    - java.util ------------> JCF(java collection framework)
    - java.sql-------------> DB Connection
    - javax.servlet------> Servlet program
    - javax.servlet.jsp--> JSP Programming

**JDBC Driver:-**  The driver program which is used to connect the database product to java application is **k**nown as JDBC.

NOTE:- This jdbc drivers provide by the vendor company of soft data base product.

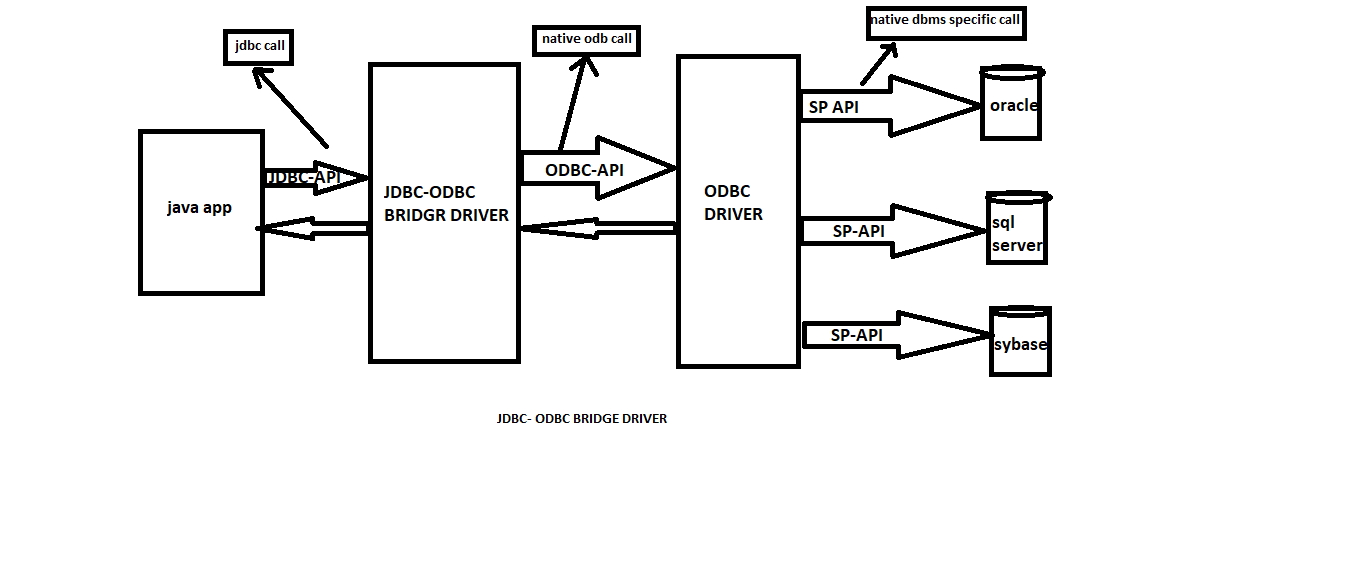
**Types Of JDBC Driver:-**

* + - * JDBC-ODBC bridge driver(Type 1)
      * Native API driver(Type 2)
      * Network protocol Driver(Type 3)
      * Thin Driver(Type 4)
      * JDBC-ODBC bridge driver(Type 1)

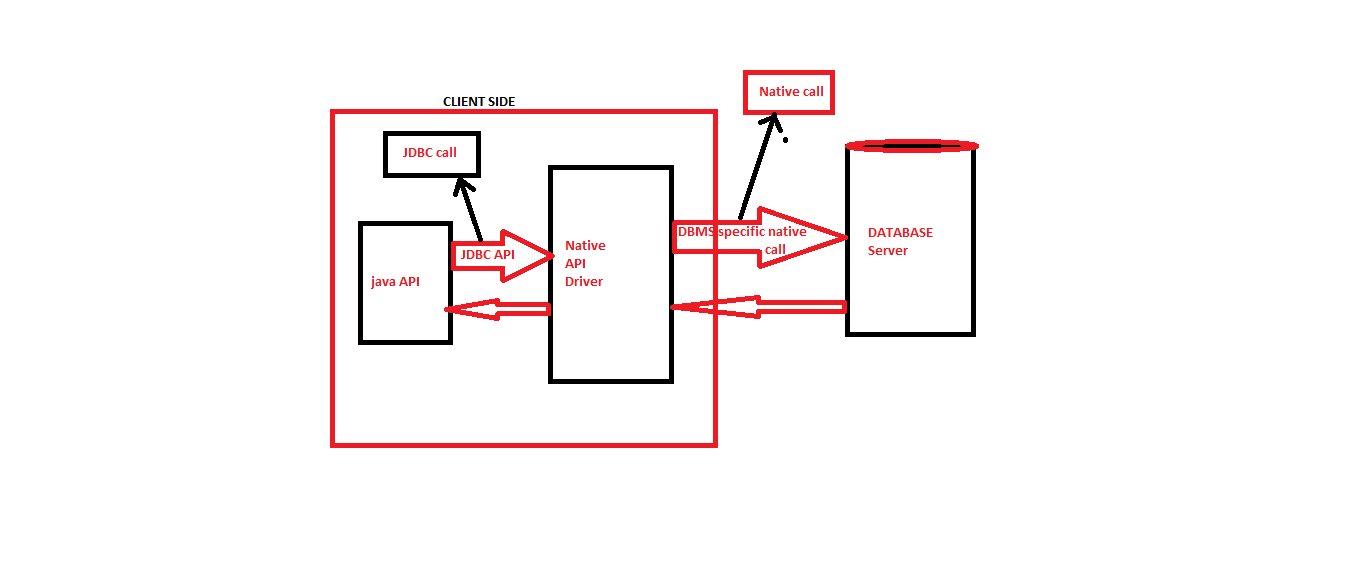
**JDBC-ODBC Bridge Driver (Type 1):-** in jdbc-odbc bridge driver the java calls re converted into odbc calls this odbc calls will active the DB specific to connect to database.

**Disadvadvantage:-** This driver will waste the execution time in converting java calls into odbc calls and degrades the performance of an application.

NOTE:- this type 1 driver is not available for javas onward.



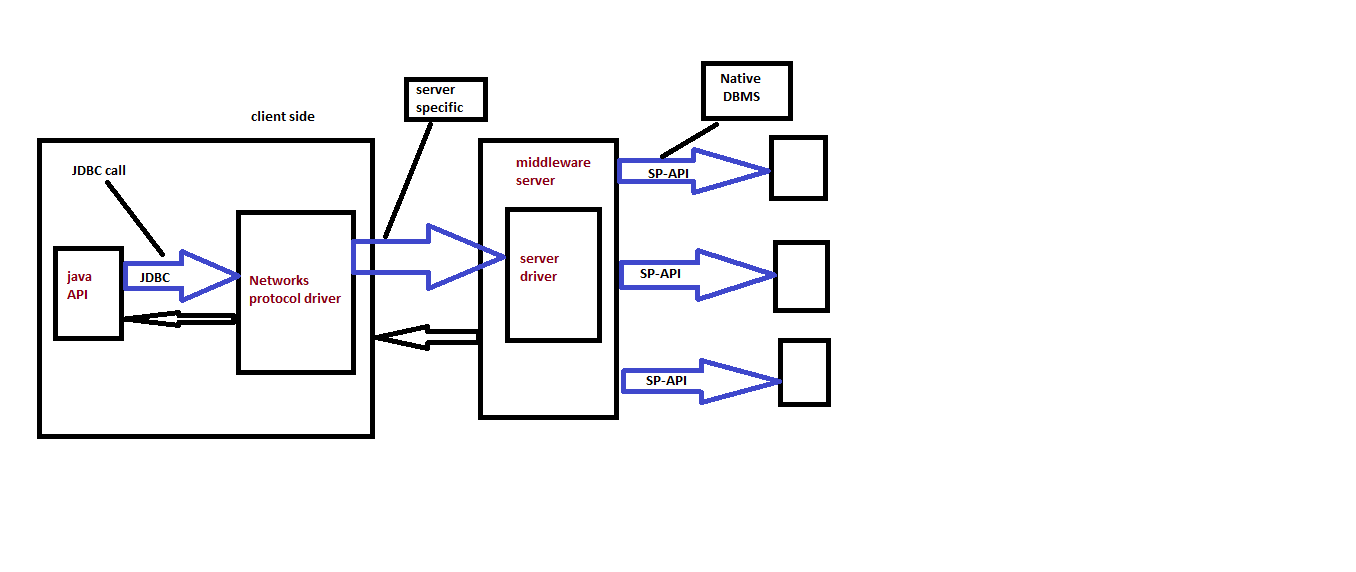
* **ODBC:-** Its stands for open DataBase Connectivity which is constructed using C 0r C++ coding and which is platform dependent.
  + This odbc layer will convert java call into DB specific call.
* **Native API Driver (Type 2):-** in type 2 driver the java program is added with database related native libraries , which communicate with the database product.

****

**Disadvantage:-**

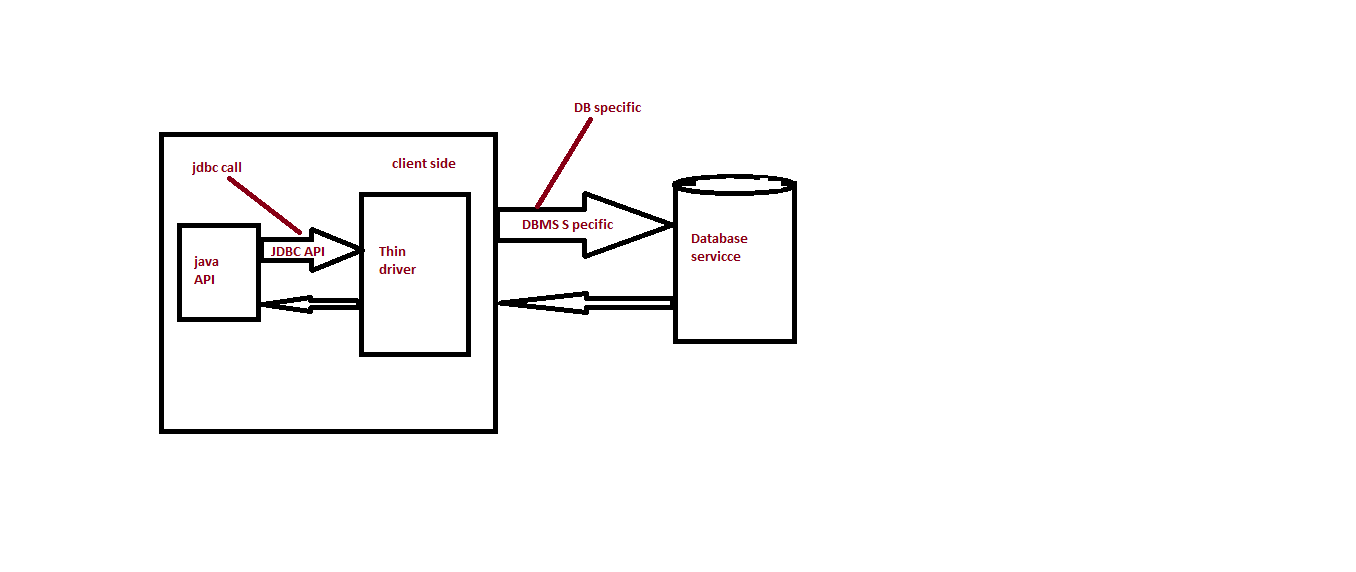
* + - * In type 2 driver we have to install the data base specific native lib in the client computer (which is not prefer)
      * Which also degrades the performance.

**3 . Networking protocol:-** in type 3 we generate middleware server specific call activates DB call.



**Disadvantage:-**  in this driver multiple network calls involve in execution process and degrades the performance of an application.

The middleware server must have database specific code.

**4 Thin Driver:-**  in type 4 driver the java program communicate database product using  driver class directly which generates high performance in application execution.

Installing the database product and making the system ready with DB product.

Step 1 :- install database product (oracle product)

Step 2 :- crate table with name “Employee11” from sql command language.

Create table Employee(id varchar2(10), name varchar2(10), desg varchar2(10), age number(10), bsal number(10));

Step 3 :- insert min. 5 records from sql command language

Insert into employee values(‘a121’,’nit.v’,’sf’,30.12000);

Step 4 :- Copy DB jar file into “ext” folder of jdk.

This DB jar file is available at “lib” of oracle product.

Path:- c:\oracleexe\app\oracle\product\10.2.0\server\jdbc\lib.

This DB jar file must be copied into “ext ”folder of jdk.

Path:- c:\programfiles\java\jdk1.8.0\jre\lib\ext

**The following are five steps to connect database product into java program:-**

* + - * **Loading Driver class**
      * **Craeating Connection**
      * **Preparing Statement**
      * **Executing Query**
      * **Closing the connection**

**1 Loading Driver Class:-**  we use ”forname()” method to load driver class into java program.

* + - * forname() is an static method available from java.lang .class; which is used to load driver class at run time(Execution time)
      * **method signature:-**

public static java.lang.class forname(String) throws java.lang ClassNotFoundException;

syntax:-

Class var =Claas.forName(“DriverClass”);

Note:- when this method is used in user defined class raises the compile time exception as java.lang.class Not Found exception we ignore this exception using “throws” keyword. Added to the method signature.

2. creating connection :-

🡪 we use “getConnection()” method to create DB Connection.

🡪”getConnection ()” is an static method available from java.sql.DriverManager.

* + - * method signature:- public static java.sql.connection.getConnection(String,String,String) throws java.sql.SQLException;

Syntax:- Connection var =DriverManager.getConnection(‘’url”,”Uname”,”Pword”);

Connection con = DriverManager.getConnection(“jdbc:oracle:thin:@localhost:1521:orcl”,”scott”,”tiger”);

Jdbc oracle thin @localhost 1521 orcl

API Product driver location port service

NOTE:- when this method is used in userdefined class raises one compiletime exception

Java.sql.SQLException, which is ignored using “throws” keyword.

**3 PreparingStatemant:-**  The Process of specifying the action to be performed on database product .is known as preparingStatement.

We use CreateStatement() method for preparing statement.

This createstatement() is an non-static method. Avaialable from java.sql.connection interface

**Method signature:-** public abstract java.sql.statement createstatement()throws java.sql,SQLException;

Syntax:- statement stm = con.createstatement();

**4 Executing query:-**

* + - * we use “excuteQuery” method for query execution .
      * this method “excuteQuery” is an non static method available from java.sql.statement interface

**method signature:-**

**public abstact java.sql.ResultSet executeQuery (String) throws java.sql.Exception;**

**Syntax:-** ResultSet rs = stm.excuteQuery(“Query”);

Note:-The Query is executed and the result is based on ResultSet object.

**5 Closing the Connection:-**

* + - * we use close() method to disconnect the DB connection.

**Syntax:-**  con.Close();

EX 1:- WAP to display data from employee11 using javaprogram?

**Step 1 :-** Open Ide – eclipse while opening name the work space click ok.

**Step 2 :-** Craate java project,

Click on file----->new ---->javaproject,name the project and click finish

**Step 3 :-** create package within javaproject (src) right click on java project ----> name the package click finish

**Step 4 :-** create class within the package.

RightClick on package----->new Class ----> name the class and click finish

**Step 5 :-** type the program

**package** test;

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

**public** **class** DBCon1 {

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

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

Connection con = DriverManager.*getConnection*

("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");

Statement stm = con.createStatement();

ResultSet rs = stm.executeQuery("select \* from employee");

**while**(rs.next()) {

System.***out***.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getString(3)+"\t"+rs.getDouble(4));

}

}

}

//retriew the data

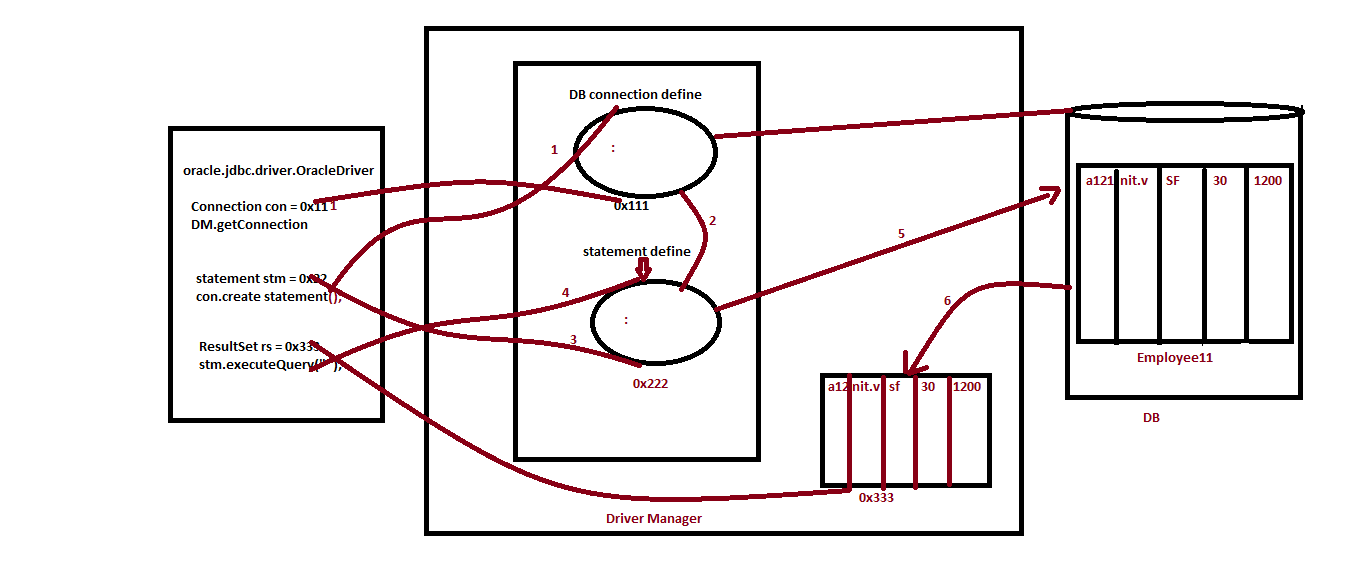
**Step 6 :-**  Execute the program use ctrl+f11

NOTE:- check JRE system library reflected with DB jar file or not if not reflected add external jar file as follows:-

Right click on java project---->Build path---->configure build path --->libraries-->add External jars-->browse the DB jar (odbc14.jar) from “ext ”of jdk.

Check port no. and Service name of DB product check in “tsname.ora” file

* **Execution flow of the program :-**

****

**Rs.next():-**  next() method is from ResultSet which is used to move the pointer or indicator from top the ResultSet data to bootom of ResultSet data

If the record is available then next method will return “true ” if not “false”

NOTE:- we use the following method to retriev data from the col. Of ResultSet.

getString(colno,colName)

getInt(colno,colName)

* **Types of Statement in JDBC :-**
  + - * **Statement**
      * **PreparedStatement**
      * **Callable Statement**
* **Statement:-**  By using statement we can excute normal queries without parameter.

Ex- above program and diagram.

Ex 2 – WAP to create and insert data table using statement?

package test;

import java.sql.\*;

import java.util.\*;

public class DBCon2 {

public static void main(String[] args) throws ClassNotFoundException,SQLException {

Scanner s = new Scanner (System.in);

System.out.println("enter the query create or insert");

String Query = s.nextLine();

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

Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");

Statement stm = con.createStatement();

int k = stm.executeUpdate(Query);

if(k>=0) {

System.out.println("unUpdated database");

}

con.close();

s.close();

}

}

// create table

* executeUpdate:- this executeUpdate() method is used to perfrom operation like create ,insert,upate,and delete on DB table. This method will return int data value

Syntax:- int k= ExecuteUpdate(query);

**PreparedStatement :-**  it is used to execute queries with parameter. We use preparedStatement() method to prepare “preparedstatement” which is available from java.sql.connection(I).

**Method signature:-**

Public abstract java.sql.preparedstatement(java.lang.string);

Syntax:- preparedstatement ps = con.preparedstatement(“query”);

EX:- wap to display the product detail based on product code?

package test;

import java.sql.\*;

import java.util.\*;

public class DBCon5 {

public static void main(String[] args) throws ClassNotFoundException,SQLException{

Scanner s = new Scanner (System.in);

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

Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");

PreparedStatement ps1 = con.prepareStatement("select\* from products11 where pcode = ?");

System.out.println("enter the pcode");

String pcode = s.nextLine();

ps1.setString(1,pcode);

ResultSet rs = ps1.executeQuery();

if (rs.next()) {

PreparedStatement ps2 = con.prepareStatement("delete from products11 where pcode = ?");

ps2.setString(1,pcode);

int k= ps2.executeUpdate();

if (k>=0) {

System.out.println("products detail deleted");

}

else {

System.out.println("invalid pcode");

}

con.close();

s.close();

}

}

}

EX:- WAP to perform user registration process?

**package** advance;

**import** java.util.\*;

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

**public** **class** DBCon6 {

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

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

System.***out***.println("enter the Uname");

String Uname=s.nextLine();

System.***out***.println("enter the pword");

String pword=s.nextLine();

System.***out***.println("enter the fname");

String fname=s.nextLine();

System.***out***.println("enter the lname");

String lname=s.nextLine();

System.***out***.println("enter the address");

String address=s.nextLine();

System.***out***.println("enter the phno");

**long** phno=Long.*parseLong*(s.nextLine());

System.***out***.println("enter the mailid");

String mailid=s.nextLine();

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

Connection con = DriverManager.*getConnection*("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");

PreparedStatement ps = con.prepareStatement("insert into userreg values (?,?,?,?,?,?,?)");

ps.setString(1,Uname);

ps.setString(2,pword);

ps.setString(3,fname);

ps.setString(4,lname);

ps.setString(5,address);

ps.setLong(6,phno);

ps.setString(7,mailid);

**int** k = ps.executeUpdate();

**if**(k>=0) {

System.***out***.println("user reg. successfully");

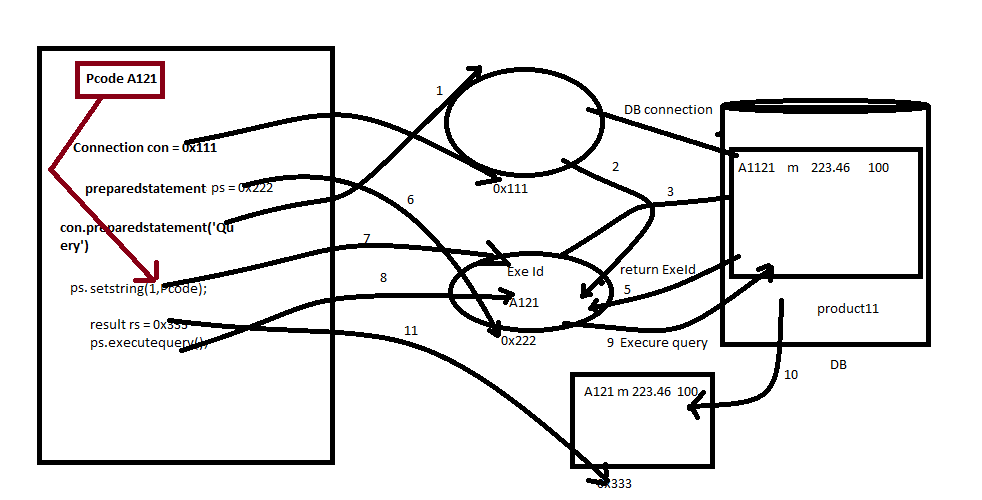
}

//con.close();

//s.close();

}

}



1. executePreparedStatement()method.
2. In the process of method execution PreparedStatement object is created.
3. While object creation query structure is compiled
4. While compilation query executionID is generated.
5. This query ExecutionID is stored in preparedStatement object
6. The refrence of PreparedStatement object is loaded on to the refrence varialble (ps)
7. Set the values within PreparedStatement object.

**NOTE:-** Based on that query requirement we can set multiple values

1. ExecuteQuery (Execute Query () and ExecuteUpdate ())] calling the method.

**NOTE:-**

While query execution exxe ID and values are used

1. Execute Query running query on DB.

**Advantage of PreparedStatement:-**

* When we want to update multiple records to database table the query is compiled once and excute multiple times.
* In which compilation time is saved and generates high performance.

EX:- Wap to perform log in process?

**package** advance;

**import** java.util.\*;

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

**public** **class** DBCon7 {

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

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

System.***out***.println("enter the uName");

String uName=s.nextLine();

System.***out***.println("enter the pword");

String pword = s.nextLine();

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

Connection con = DriverManager.*getConnection*("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");

PreparedStatement ps = con.prepareStatement("select \* from userreg where uname=? and pword = ?");

ps.setString(1, uName);

ps.setString(2, pword);

ResultSet rs = ps.executeQuery();

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

System.***out***.println("Login successful");

}

**else** {

System.***out***.println("failed");

}

}

}// LOGIN

* Using looping structure

EX:- Write a java program to insert multiple data in the table by choice ……

**Note:-**  I the above program the query is compiled once and executed multiple times.

**Question:-**  WAP to update the price of book based on book code?

**NOTE:-**  in the process of constructing the logic we can take any number of preparedstatements.

**Question**:- WAP to delete products11 detail based on pcode?