**JDBC**

**Q-1: What is the use of Driver?**

**A:** JDBC Driver is a software component that enables java application to interact with the database

the JDBC API defines the Java interfaces and classes that programmers use to connect to databases and send queries.

A JDBC driver implements these interfaces and classes for a particular DBMS vendor.

**java application ----------------------> jdbc driver**

**(uses jdbc API) |**

**---------------------------------------------------**

**| |**

**MySql libraries) (Oracle libraries)**

**MySql Oracle**

**Q-2: Who Provide the Driver?**

**A:** Database vendor provide the driver

**Q-3: How many type of Driver.**

**A:** there are 4 type of jdbc driver

type-1 -> JDBC ODBC Bridge Driver

type-2 -> Native-API Driver

type-3 -> Net Protocol Driver

type-4 -> Native-Protocol- pure Java Driver

**Q-4 : Why we use Type -4 Driver**

**A:** This kind of driver is extremely flexible, you don't need to install special software on the client or server it can directly interact with the server

**................**

**: jdbc program : ................**

**:..............: : oracle :**

**| ^ :..............:**

**| | | ^**

**| | | |**

**| | | |**

**................ ................**

**: type - 4 :<------------------------------:server side DB:**

**: driver : : libraries :**

**:..............:------------------------------>:..............:**

**But in previous Driver Some Problem was there that’s why we use Type-4 Driver**

**Type – 1 Driver->**

...........

**: :**

**........... ........... ..........<-------: oracle :**

**: :------->: :-------->: :------->:.........:**

**: jdbc : : jdbc : : odbc : DSN**

**: program : : program : : driver : ..........**

**: :<------ : :<--------: :-------->: :**

**:.........: :.........: :........:<--------: mysql :**

**DSN :........:**

**Disadvantage->**

its implementation is possible in window os only because ODBC drivers available only with windows

**Type-2 Driver->**

**................**

**: jdbc program :**

**:..............:**

**| ^**

**| |**

**................ ................**

**: type-2 driver: : oracle :**

**:..............: :..............:**

**| ^ | ^**

**| | | |**

**................ ................**

**:client side DB:<------------------------------:server side DB:**

**: libraries : : libraries :**

**:..............:------------------------------>:..............:**

**Disadvantage->**

(1)in this both client and server m/c will have the database library

(2)when database is migrated then you will get much maintenance because

you need to re-install client sid libraries in all client m/c

**Type-3 Driver->**

**................ ................**

**: jdbc program : IDS server : oracle :**

**:..............: Internert Database :..............:**

**| ^ Access Server | ^**

**| | | |**

**................ ................ ................**

**: :----------->:client side DB:--------->:server side DB:**

**: type-3 driver: : libraries : : libraries :**

**:..............:<-----------:..............:<---------:..............:**

**Disadvantage->**

(1)you need to purchase extra s/w called IDS server

(2)because of having middleware server between your program and database

server,performance will be reduced.

**Q-5: What is the Step to write JDBC Code**

A: There are the following Steps to

step 1 - Load the Driver class

step 2 - create the connection

step 3 - create the statement

step 4 - execute query

try

{

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

String url= "jdbc:oracle:thin:@localhost:1521:XE";

**2-**Connection con = DriverManager.getConnection(url,"system","system");

**3-** Statement st=con.createStatement();

String query="insert into emp4 values(3,'b',8302457983,5000)";

**4-**int x=st.executeUpdate(query);

}

catch(Exception e)

{

e.printStackTrace();

}

**Q-6 : How execute() work**.

**A:** it is used to perfoem any type of operation on table such as insert,update,delete and select

**Syntex:**

**public boolean execute(String sql)->**

when you want to submit "insert or update or delete select" sql statements

then use execute() method which returns the boolean value

**1->**if return value is **true** which means that **"select"** sql statement is submitted

and ResultSet object is created

\* use the following method of Statement to get the ResultSet object

public ResultSet getResultSet()

**2->**if returned value is **false** which means that **"insert,update or delete"** sql

statement is submitted and int number is available which represents no. of

record inserted,updated or deleted

\* use the following method of Statement to get the int number available

public int getUpdateCount()

**Q-7 : What is the Diff b/w RowSet and ResultSet.**

**A:**

|  |  |
| --- | --- |
| **ResultSet** | **RowSet** |
| ResultSet object is used to store records returned by SELECT sql Statement | RowSet object is also used to store records returned by SELECT sql Statement |
| By Default , ResultSet object are forward-only  And read - only | By Default , RowSet object are Scrollable And updatable |
| ResultSet objects are connection oriented i.e  You can access the ResultSet data as long as connection is available. Once Connection is closed , ResultSet also will be closed automatically | RowSets are connection less scrollable and updatable |
| ResultSet objects are not eligible for Serialization | RowSet objects are eligible for Serialization |

**Type of ResultSet:**

Depending on the ResultSet cursor movement , you can divide the ResultSet into 2 types

1. Forward-Only ResultSet
2. Scrollable ResultSet

**1-Forward-Only ResultSet->**

When ResultSet is forward – only then:

1. Pointer can be moved in forward direction only
2. Pointer can be moved in sequential order only

By default , ResultSets are forward only.

**2-Scrollable ResultSet->**

When ResultSet is Scrollable then:

1. Pointer can be moved in both forward and reverse direction
2. Pointer can be moved in random order only

**Type of RowSet:**

1. Connected RowSet
2. Disconnected RowSet

**1-Connected RowSet->**

1. Connected RowSets are like ResultSets i.e Connected RowSets needs the connection
2. You can not serialize Connected RowSet.
3. JdbcRowSet is Connected RowSet

**Disconnected RowSet->**

1. Disconnected RowSets are not like ResultSets i.eDisconnected RowSets do not needs the connection
2. You can serialize Disconnected RowSet.
3. cachedRowSet is Disconnected RowSet