# # JDBC = JAVA DATABASE CONNECTIVITY

** JDBC :- jdbc is a api is used to connect java application with the data base( with-ought jdbc is not connect to each other)*

*NOTE:- jdbc is located inside jdk (not download manually)*

**** What is jdbc ....?***

*jdbc(java database connectivity) is an api which allows java application to communicate with database.*

*Or*

*jdbc is an api which is use to connect java application with the database*

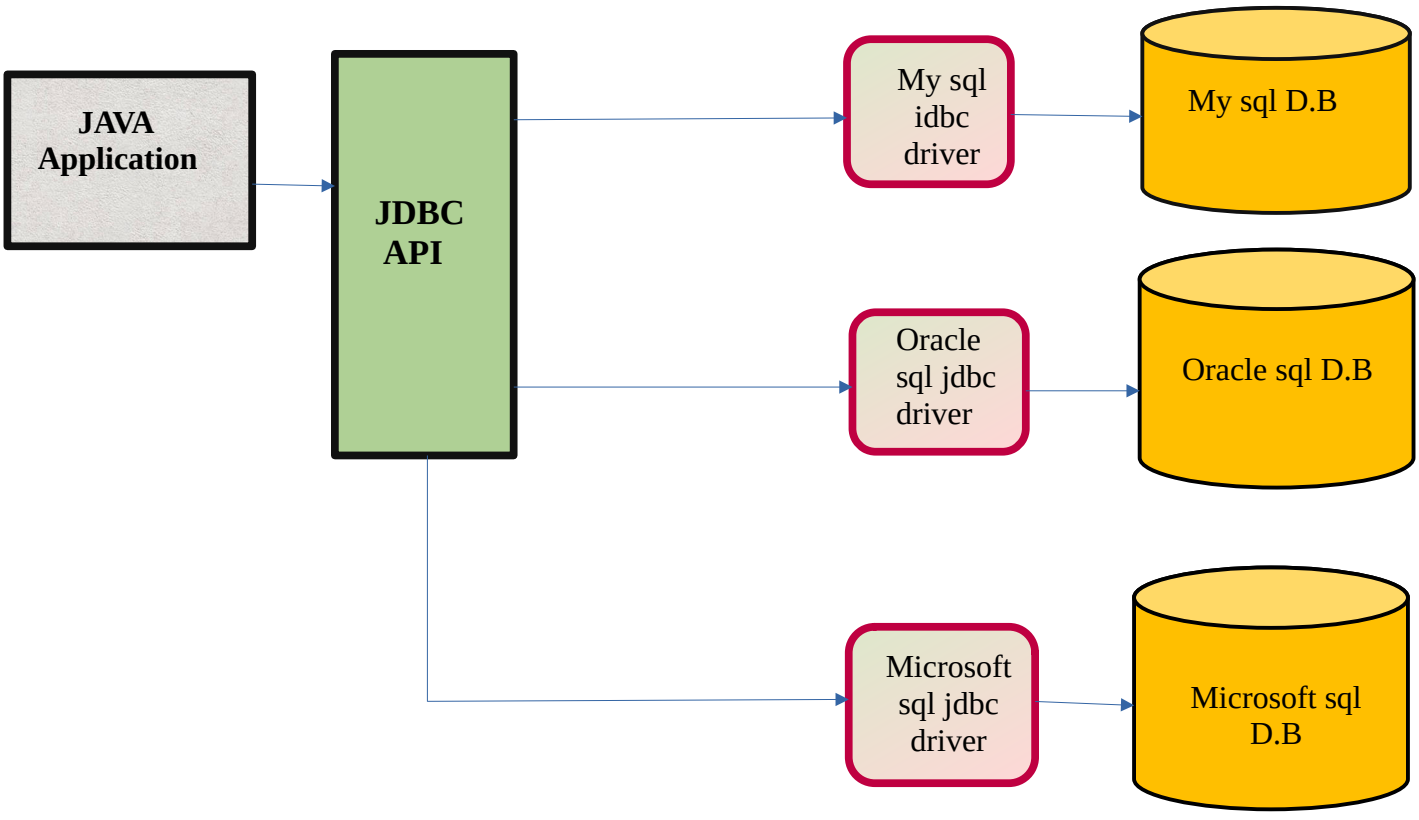
*only one way- jdbc 1.data= some information*

*jdbc is a mediator*

** API :- API is a mediator which allows two application communicate with each other*

** API stands for Application programming interface. Api is a software mediator which allows two applications to communicate with each other*

## # Architecture of jdbc



- jdbc is built in java
- jdbc is located in jdk but jdbc driver is not built in java

## Steps of follow to connect java application with database in JDBC

- load and register the driver
- establish the connection
- create the statement
- execute the statement
- process the resutltset
- close the connection

### A) load and register the driver :-

➔ **Class.forName**("com.mysql.cj.jdbc.Driver");

(a)**Load the Driver:-** the process of adding jdbc driver in our java application is called as loading the driver.

Steps:-

1. right click on project name
2. select build path
3. select configure build path
4. click on libraries
5. there are two option in libraries ( a. class path, b. module path)
6. click on class path
7. click on add external jars
8. select jar file( which we have download)
9. click on apply
10. click on apply and close

### (b) Register the Driver:-

We Register the Driver in multiple Ways Class.forName is one among them

why register to driver :-
When ever we load the jdbc driver our jdbc api doesn't recognized it.

(c) Class.forName("com.mysql.cj.jdbc.Driver");

- ✗ it is static method (in class)
- ✗ use to register Driver pass URL of our JDBC Driver.
- ✗ This method of class is used to registered driver class. This method is used to dynamic load the driver class
- ✗ it is a public and static method and it is accept only one String parameter

- **Interfaces of Jdbc (inside jdbc is present java.sql.package)**

We can not create the object of interface that time helper class is present that is DRIVER MANAGER

(d) DRIVER MANAGER:-

- ✗ insert in java.sql.package present.
- ✗ It is helper class.
- ✗ And get the connection

(e) getConnection("jdbc:mysql://localhost:3306" "root" ,"root");

- ✗ establish the connection between the DATABASE.
- ✗ It is private static method
- ✗ it is a method we gives as to object of connection.
- ✗ Static method and returntype is a connection & datatype is a String

## B) establish the connection :-

1. connection:- (to establish the connection with the help of **Driver Manager.getconnectionmethod**)
   jdbc is establish the connection with the help of DriverManager
   - we need a object but
   - we can not create a object for connection because it is a interface
   - the driver manager is provide the object
   - inside a driver manager class is present in get-connection method we have provide URL username and password.

- Connection con= DriverManager.getConnection(“URL” “username””password”)
- URL =”jdbc:mysql://Localhost:3306”,
- Username = “root”
- Password = “root”
- Connection con= DriverManager.getConnection(”jdbc:mysql://Localhost:3306”, “root” “root”)
- Return type of Get connection method is = **Connection**
- when ever the URL username and password is connect then gives the object
- when ever the URL username and password anything is wrong then gives the exception

**PROGRAMM OF REGISTER AND CONNECTION**

```
package jdbcdemo;
import java.sql.connection;
import java.sql.DriverManager;
import java.sql.sql.Exception;

public class jdbcBasic
        {
            Public Static void main(String []args)
    {
try{
 Class.forName(“com.mysql.cj.jdbc.Driver”);

 system.out.println(“registered”);

 Connection con=
DriverManager.getConnection(”jdbc:mysql://Localhost:3306”
,“root”“root”);

 system.out.println(“connected”);
```

*}*

***Catch(SQL Exception)***

***{ e.printstacktrace}***

*}*

C) create the statement :-

- Create statement is a no argument method or non parameterized method

2. statement:- (it is use to execute a querry with the help of **st.createStatement method**)
   - to create the statement we have a important method create statement
   - create statement is a non-static method
   - which is present inside connection interface
   - to call create statement we need the reference because it is a non-static method
   - call that method because to create the statement
   - the connection reference variable is **con**
   - returen type of create statement method is **Statement**
   - **Statement st= con.createstatement();**
   - return type of execute querry method is resultset and it is present inside statement
3. prepared statement:- (it is use to execute a querry with the help of **prepareStatement**)
4. resultset:- (is used to **fetch** the data )

5. Resultset Meta Data:- (it is use to specify the detail of table)
6. Driver Manager:- (it is a helper class)

➔ **Execute Statement Method** :- (3 methods)

**1. execute(String sql);**
return type of execute method is boolean

**2.execute Update(String sql));**
return type of execute update method is int

**3.executeQuerry(String sql);**
return type of execute querry method is result set

## Program of Create the database

```
package jdbcdemo;
import java.sql.connection;
import java.sql.DriverManager;
import java.sql.sql.Exception;

public class jdbcBasic
        {
            Public Static void main(String []args)
    {
try{
 Class.forName("com.mysql.cj.jdbc.Driver");
 system.out.println("registered");
Connection con=
DriverManager.getConnection("jdbc:mysql://Localhost:3306"
,"root""root");
 system.out.println("connected");
```

```
Statement st = con.CreateStatement();
st.execute("Create database java2");
System.out.println("database created");

    }
Catch(SQL Exception)
 { e.printstacktrace}

                    }
```

**output :-** Database created

## Program of Create the Table

```
package jdbcdemo;
import java.sql.connection;
import java.sql.DriverManager;
import java.sql.sql.Exception;

public class jdbcBasic
          {
                Public Static void main(String []args)
     {
 try{
  Class.forName("com.mysql.cj.jdbc.Driver");

  system.out.println("registered");
```

```
Connection con=
DriverManager.getConnection("jdbc:mysql://Localhost:3306/
java2", "root""root");
  system.out.println("connected");


Statement st = con.CreateStatement();
st.execute("Create table student (id int, name varchar(20),
age int)");
System.out.println("Table created");
    }
Catch(SQL Exception)
 { e.printstacktrace}
                    }
```

output :- Table created

**insert Value :-**

```
st.execute("insert into student value(1,' sneha', 24)");

or
st.execute("insert into student(id,name,age)values(1,'sneha',24)");
```

**Delete value :-**

**st.execute("delete from student"); = whole table value is deleted**
**st.execute("delete from student where(id=1)"); = specific row is deleted**

**Update value:-**

**st.execute("update student set name='radha' where (id=1)");**

➔ **Prepare Statement Method** :- (3 methods)

**1. execute(String sql);**

- return type of execute method is **boolean**
- We use execute() method whenever we don't won't how many rows are getting affected with our querry (not Written any Value)

**ex:- st.execute();**

**2.execute Update(String sql));**

- return type of execute update method is **int**
- We use execute update() method we know how many rows are getting affected with our querry
- **ex:- st.executeupdate ();** // it denoted no.of row getting affected

**3.executeQuerry(String sql); // String sql is a variable name**

- return type of execute querry method is **resultset.**
- Fetch the data from Database
- The Resultset object holds the data are use execute querry whenever we want fetch the data
-

**4.setInt();**
**5.setString();**
**6. setBoolean();**
**7.setFloat();**
**8.setDouble();**
**9.setLong();**

- RESULTSET:- it is a interface , it is used to store the result or data after the execution of querry.

- it is an interface it is used to store data which is fetch the data from the database
- Resultset is an interface is present inside ***java.sql.package.***
- it store the data inside the resultset
- return type of execute querry method is resultset
- Result set is a table of data which represent a data from the database this is generated by executing the querry statement to the database.

ResultSet rs = st.executeQuerry("Select * from Student");

**Some Methods of resultset**

- **next():-** moves the curser from next row to the current possion
- return type of next method is boolean
- if data Is present that row then it return true
- data is absent then it returns false

- **getInt(); :-** getInt (int columnindex) is used to fetch the int type of data from the resultset

- **getString();** fetch the data of String type from resultset

- **getBoolean();** (int columnindex) fetch the data of boolean type from resultset

- **getFloat();** (int columnindex) fetch the data of Float type from resultset

- **getDouble();** (int columnindex) fetch the data of Double type from resultset

- **getLong();**(int columnindex) fetch the data of Long type from resultset

**Q. What is use of get method ?**
When I have use execute & executeUpdate method

**STATEMENT (Parent/Super)** ⟶ **PreparedStatement(child/sub)**

- **PREPARED STATEMENT :-** (it is an non static method)

- ✗ it is a interface and it is a sub interface of sub-interface of statement.
- ✗ Prepared statement is always faster than statement.
- ✗ Statement is parent interface.
- ✗ prepare-statement is a child interface.
- ✗ It is a non-static method it is present in connection interface.
- ✗ Prepared statement is parameterized method
- ✗ in that method we can not write any querry inside execute method
- ✗ it will take dynamic input from username
- ✗ Connection interface calling for connection reffrance (con.preparedStatement);

whenever we create prepared statement we need to pass a querry

**ex. con.prepareStatement(String sql);**

- **Difference between Statement and Prepared Statement**

| Statement | preparedStatement |
|---|---|
| ✔ Statement st=con.createstatement(); | ✔ Preparedstatememt ps=con.preparedstaement(Querry) |
| ✔ statement is not associated with any querry | ✔ prepared statement associated with one perticular querry. |
| ✔ st.execute(insert querry) | ✔ ps.execute(); |
| ✔ st.execute(update querry) | ✔ ps.execute(); |
| ✔ st.execute (delete querry) | ✔ ps.execute(); |
| ✔ at the same time we execute different querryes | ✔ We execute same querry multiple time |
| ✔ statement is slower than prepared statement | ✔ Prepared statement is faster than statement |
| ✔ 1000* 4 =4000 sec | ✔ 1000 * 3=3000 sec |
| ✔ each querry compiled | ✔ it is not compiled again and again querry compiled only one time |

| | |
|---|---|
| ✔ Whenever no need to pass a querry then programmer Choose the Statement<br>EX:-<br>Con.CreateStatement()<br>↓<br>not pass any querry<br><br>✔ In Statement exach querry is compile and executed<br><br>✔ We Use Statement Whenever we want to execute ***different*** types of querry at the same time | ✔ Whenever we create a Praparestatement definitely we need to pass a Querry<br>EX :-<br>Con.PreparedStatement(String Sql);<br>↓<br>Pass a querry<br><br>✔ In preparedStatement Each querry is directly executed.<br><br>✔ We Use Statement Whenever we want to execute ***Same*** types of querry at the same time |

****** To Execute Prepared Statement We have Three methods**

**1) Execute():-**

- return type of execute method is **boolean.**
- The return true when programmer writes select querry
- We use execute() method whenever we don't won't how many rows are getting affected with our querry (not Written any Value)

**2)Execute Update method:-**

- ✗ specifies the no of rows getting affected.
- ✗ Return type of **execute update method is int.**
- ✗ int represent the no of rows getting affected.
- ✗ We use execute update() method we know how many rows are getting affected with our querry

**3) ExecuteQuerry():-**

- ✗
- ✗ return type of execute querry method is **resultset.**
- ✗ Fetch the data from Database
- ✗ The Resultset object holds the data are use execute querry whenever we want fetch the data

**4) SetInt(int, P , intvalue) :-** (p= parametor)

- Use the set value of Integer
- It just set a value not change Database

Ex:-

***ps.setInt(1, 70);*** *// 1 indicates a index of placeholder.*

*2 Formal Argument*

**5) SetString(int, P , String value) :- (p= parametor)**

- Use the set value of String

Ex:-

***ps.SetString(2 , '500, mesh');*** *2 formal Argument*

**6) SetDouble(int , double);**

**7)SetDouble(int , long value);**

- it is use to set value of long

Ex:-

***Ps.SetLong(4 , 6014);***

- **BATCH-PROCESSING / Batch Holder / batchUpdate :-**

**(Group of Multiple Querry)**

- it is a process of grouping SQL statement/ SQL Query and sending the one request to the database after execution one response will be submitting
- **Batch:-** it is a container to store the some queries.
- we use batch-processing when we want to huge no of querries.
- We can sent multiple querries only one request
- hold the querry /Reduce the type of execution.

St.execute(Querry) Execute the Querry in the batch

batch

- **Return type of execute batch is int[]**
- Batch processing reduces network traffic and time complexity.
- Batch processing improves the efficiency and performance of our applications.
- In batch we can write /add any type of Querry except select Querry.

******** ***Prepared Statement with Batch Processing is faster than that Prapared Statement Withought Batch processing .*********

- **1.Statement Without Batch Processing :-**

***Statement st=con.CreateStatement();***

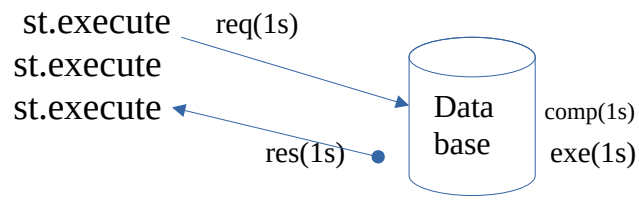
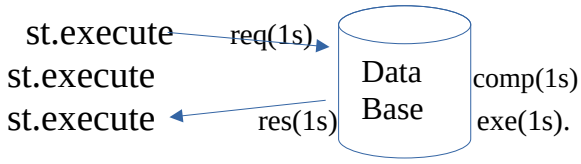
***St.execute(Querry)***
***st.execute(Querry)***
.
.
.
.
***st.exercute(Querry)***

- **1.Statement With Batch Processing :-** *(Add method is faster)*

***Statement st=con.CreateStatement();***

***St.addBatch(Querry);***
***st.addBatch(Querry);***
.
.
.
.
***st.addBatch(Querry);***
***st.addBatch(Querry);***

one method is use to Batch processing , it is “add” method add querry in the batch is called add method.

| Statement With-out Batch processing | Statement with Batch Processing |
| --- | --- |
|  <br>1000 * 4=4000 |  <br>1+ 1000*1 +1000*1+1=2002 |