Dt : 30/1/2025

Program : DBCon14.java

```
package test;
import java.sql.*;
public class DBCon14
{
	public static void main(String[] args)
	{
	  try {
	  Class.forName("oracle.jdbc.driver.OracleDriver");
		Connection con = DriverManager.getConnection

("jdbc:oracle:thin:@localhost:1521:xe","system","tiger");
		PreparedStatement ps1 = con.prepareStatement
				("insert into Product70 values(?,?,?,?)");
		System.out.println("********ParameterMetaData*******");
		ParameterMetaData pmd = ps1.getParameterMetaData();
		System.out.println("Para count :"+pmd.getParameterCount());
		PreparedStatement ps2 = con.prepareStatement
				("select id,mailid,phno from Customer70");
		ResultSet rs = ps2.executeQuery();
		System.out.println("------deatails-----");
		while(rs.next()) {
			System.out.println(rs.getInt(1)+"\t"
					+rs.getString(2)+"\t"
					+rs.getLong(3));
		}//end of loop
		System.out.println("*****ResultSetMetaData******");
		ResultSetMetaData rsmd = rs.getMetaData();
		System.out.println("Col Count : "+rsmd.getColumnCount());
		System.out.println("2nd Col Name :"+rsmd.getColumnName(2));
	  }catch(Exception e) {
	      e.printStackTrace();
	  }
	}
}
```

o/P:

*********ParameterMetaData*******

Para count :4

------deatails-----

1234 alex@gmail.com 9898981234

2312 ram@gmail.com 7676761234

3212 raj@gmail.com 4343431234

5454 d@gmail.com 9898983213

6666 d@gmail.com 7675431234

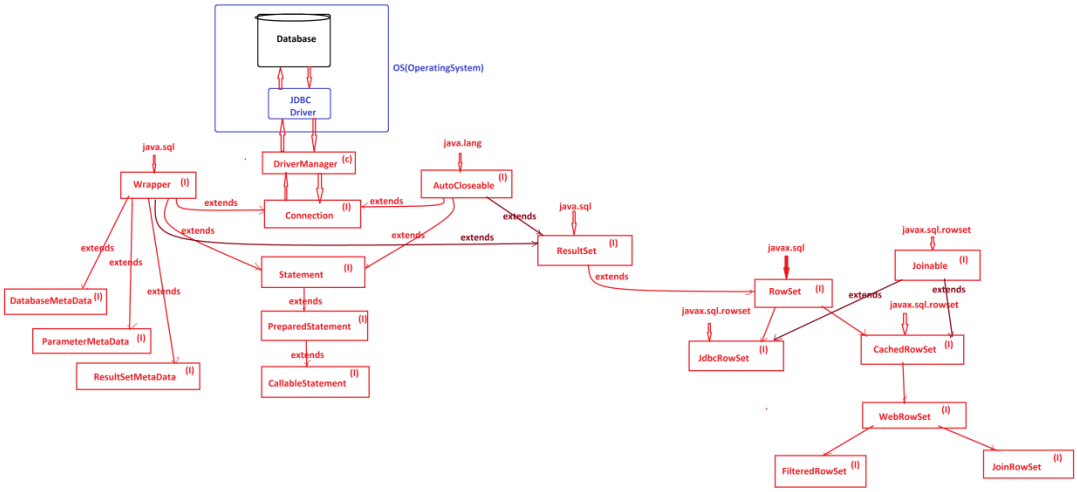
*****ResultSetMetaData******

Col Count : 3

2nd Col Name :MAILID

***imp**

Hierarchy of JDBC API:



***Dt : 31/1/2025**

***faq:**

***define 'Wrapper' in JDBC?**

=>'Wrapper' is an interface from java.sql package and which is used to check the JDBC-Component binded to Database or not.

=>The following are two important methods of 'Wrapper':

```
public abstract <T> T unwrap(java.lang.Class<T>) throws java.sql.SQLException;
public abstract boolean isWrapperFor(java.lang.Class<?>) throws java.sql.SQLException;
```

faq:

define 'AutoCloseable'?

=>'AutoCloseable' is an interface from java.lang package and which supports auto-closing operations.

=>The resource-components which are used in try-with-resource statement must be extended or implemented from java.lang.AutoCloseable interface.

Program : DBCon15.java

```
package test;
import java.sql.*;
import java.util.*;;
public class DBCon15 {
	public static void main(String[] args) {
	Scanner s = new Scanner(System.in);
	 try(s;)//Java9
	 {
		try
		{
			Class.forName("oracle.jdbc.driver.OracleDriver");
```

```
Connection con = DriverManager.getConnection
                ("jdbc:oracle:thin:@localhost:1521:xe","system","tiger");
    try(con;){
        Statement stm = con.createStatement();
        try(stm;){
                System.out.println("Enter the query(Create/Insert/Update/Delete)");
                String qry = s.nextLine();
                int k = stm.executeUpdate(qry);
                System.out.println("Value k : "+k);
                if(k>=0) {
                        System.out.println("query executed successfully....");
                }
        }//end of try
    }//end of try
        }//end of try
        catch(SQLSyntaxErrorException qs) {
                System.out.println(qs.getMessage());
        }
        catch(SQLIntegrityConstraintViolationException se) {
                System.out.println(se.getMessage());
                System.out.println(se.getErrorCode());
        }
        catch(Exception e)
        {
                e.printStackTrace();
```

```
                }
        }//end of try with resource
        }
}
```

o/p:

Enter the query(Create/Insert/Update/Delete)

create table Emp70(id varchar2(10),name varchar2(15),desg varchar2(10),primary key(id))

Value k : 0

query executed successfully....

o/p:

Enter the query(Create/Insert/Update/Delete)

insert into Emp70 values('A121','Alex','SE')

Value k : 1

query executed successfully....

o/p:

Enter the query(Create/Insert/Update/Delete)

update Emp70 set desg='SE' where id='A231'

Value k : 1

query executed successfully....

o/p:

Enter the query(Create/Insert/Update/Delete)

delete from Emp70 where id='A231'

Value k : 1

query executed successfully....

---

faq:

define 'RowSet'?

=>'RowSet' is an interface from javax.sql package and which is extended from 'java.sql.ResultSet'

=>'RowSet' will hold data which is retrived from select-queries.

=>'RowSet-Objects' are automatically Scrollable-Objects.

=>'RowSet' categorized into two types:

1.JdbcRowSet

2.CachedRowSet

1.JdbcRowSet:

=>when we use JdbcRowSet,the connection to database willnot be disconnected after retriving the data.

2.CachedRowSet:

=>when we use CachedRowSet,the connection to database will be disconnected automatically after retrieving the data.

=>In realtim,we use CachedRowSet in the form 'WebRowSet' and which is categorized into two types:

(a)FilteredRowSet

(b)JoinRowSet

(a)FilteredRowSet:

=>FilteredRowSet will hold the data which is retrieved based on condition.

(b)JoinRowSet:

=>JoinRowSet will hold the data which is joined from more than one rowset objects.

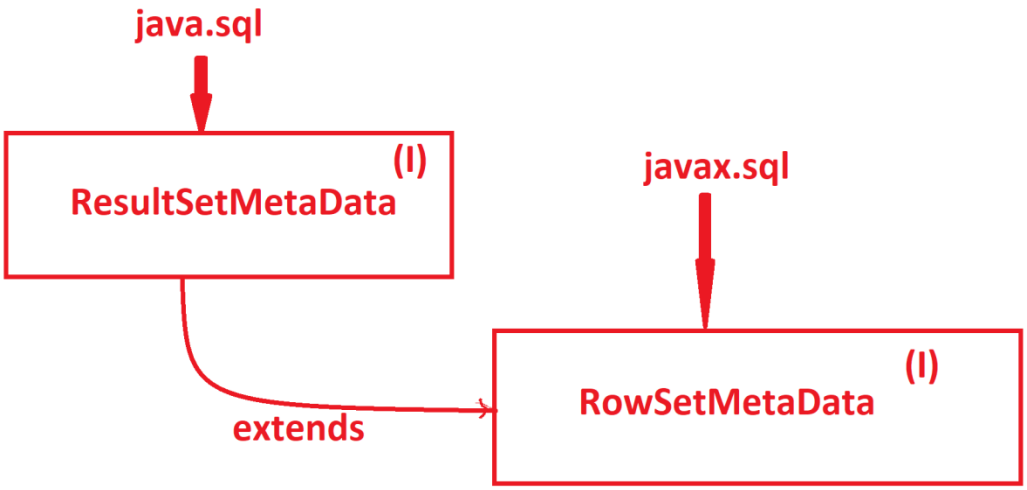
Dt : 1/2/2025

Note:

=>'javax.sql.RowSetMetaData' will hold information about 'RowSet' Objects.

=>'RowSetMetaData' is extended from 'ResultSetMetaData'

Diagram:



**imp*

define JDBC driver?

=>The driver which is used to establish communication b/w java-program and database product is known as JDBC driver(Java DataBase Connectivity driver)

Types of JDBC drivers:

=>JDBC drivers are categorized into four types:

1.JDBC-ODBC bridge driver(Type-1 driver)

2.Native API driver(Type-2 driver)

3.Network protocol driver(Type-3 driver)

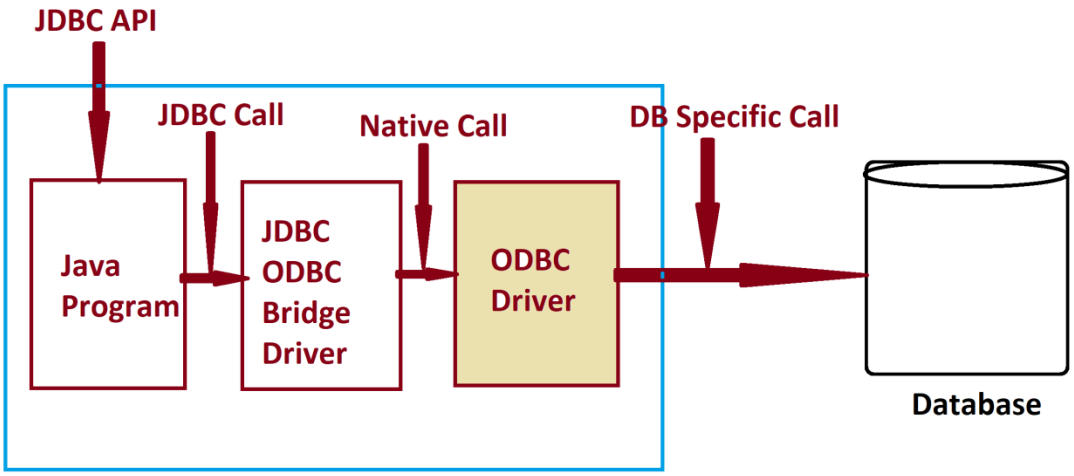
4.Thin driver(Type-4 driver)

1.JDBC-ODBC bridge driver(Type-1 driver):

=>The Type-1 driver will take the support of ODBC-driver to establish connection to Database product.

=>when we use Type-1 driver JDBC-Call is converted into Native call,and the Native Call is converted into DB Specific call for connetion.

Diagram:



DisAdvantage:

=>Type-1 driver internally uses more conversions,and which waste the execution time and degrades the performance of an application.

Note:

=>From Java8 version(2014) onwards Type-1 driver support is not available in Java.

faq:

define ODBC driver?

=>ODBC stands for 'Open DataBase Connectivity',and this driver will support to establish connection to any type of database.

=>This ODBC driver is PlatForm dependent driver,because which internally uses C/C++ codes.

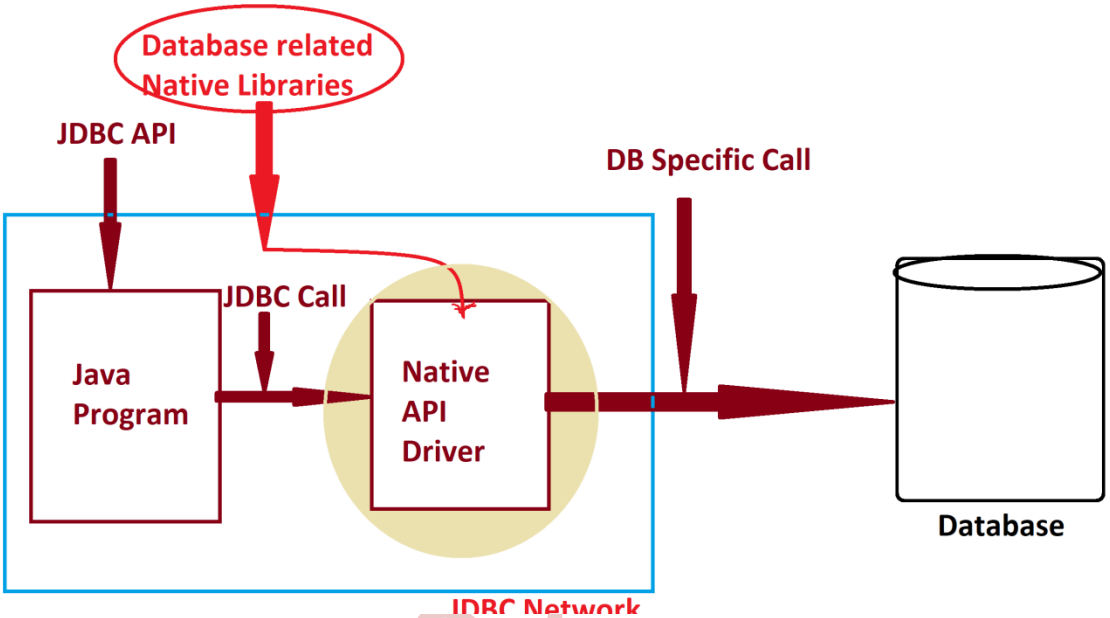
2.Native API driver(Type-2 driver):

=>Type-2 driver will take the support of Database related Native Libraries to establish

Connection to database product.

=>To Use Type-2 driver,the Client Computer must be installed with Database related Native libraries.

Diagram:



DisAdvantage:

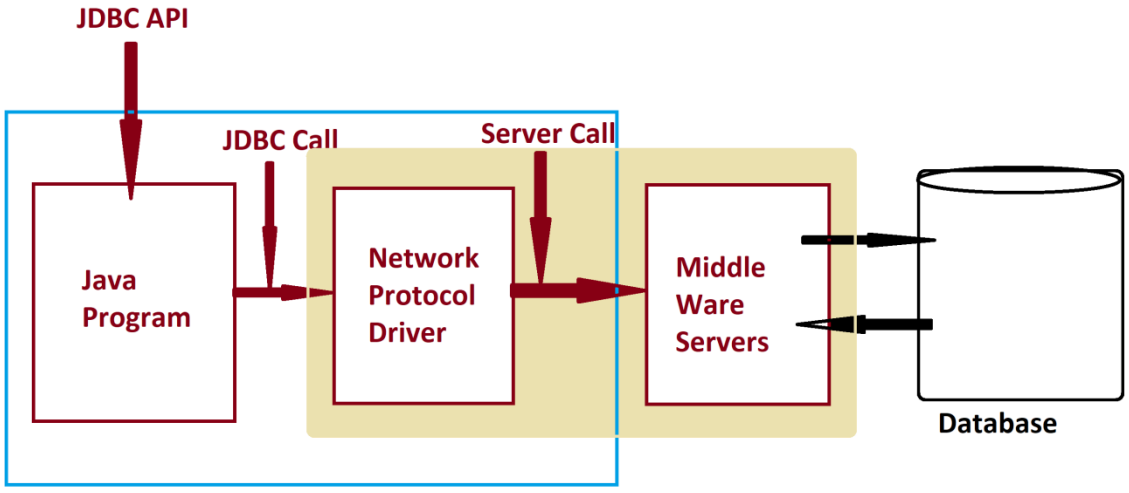
=>when we construct application with Type-2 driver,then the application will become Database dependent and which is not preferable in realtime.

3.Network protocol driver(Type-3 driver):

=>Type-3 driver will take the support of Intermediate MiddleWare server to establish connection to database product.

=>In this process Middleware Servers will hold database related connection code.

Diagram:



DisAdvantage:

=>when we want to use Type-3 driver,we have to make Network settings in ClientComputer and the Network components are involved in execution process and degrades the performance of an application.(Execution time increases)

****imp***

4.Thin driver(Type-4 driver):

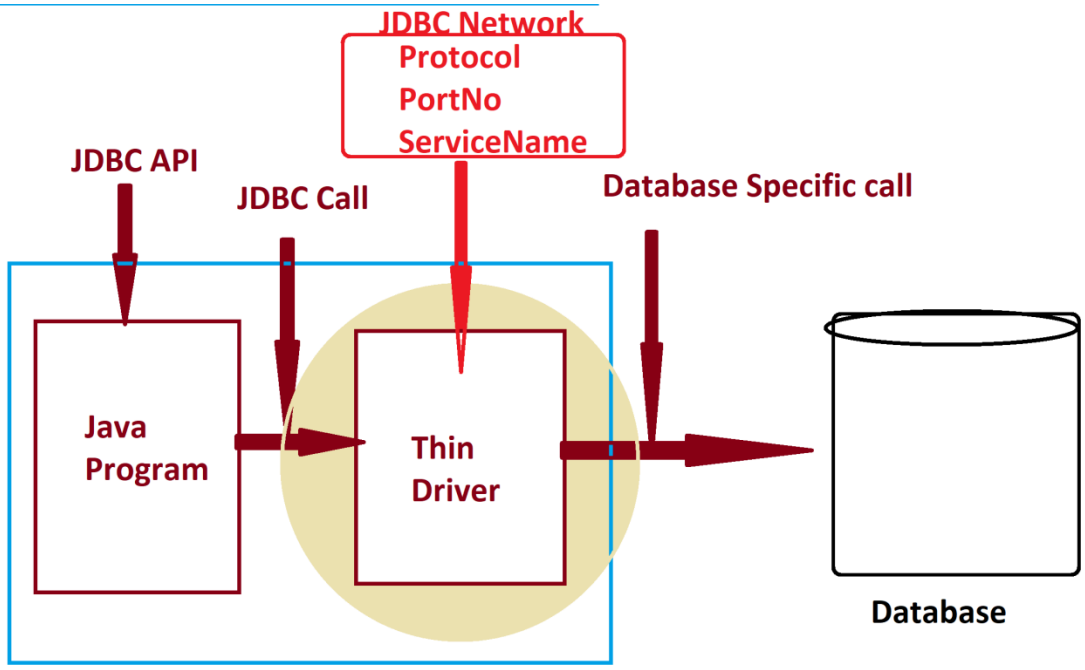
=>Type-4 driver will take the support of Database-Network-protocol to establish connection to to database product.

=>Type-4 driver is pure java-driver.

=>Type-4 driver is PlatForm independent driver.

=>Type-4 driver is high performance driver

Diagram:



==========

Summary of Objects generated from CoreJava:

1.User defined Class Objects

2.String-Objects

3.WrapperClass-Objects

4.Array-Objects

5.Collection<E>-Objects

6.Map<K,V>-Objects

7.Enun<E>-Objects

Summary of Objects generated from JDBC:

1.Connection-Object

2.Statement-Object

3.PreparedStatement-Object

4.CallableStatement-Object

5.ResultSet-Object

(i)Scrollable ResultSet Objects

(ii)NonScrollable ResultSet Objects

6.RowSet-Object

(a)JdbcRowSet Objects

(b)CachedRowSet Objects

=>WebRowSet Objects

(i)FilteredRowSet Objects

(ii)JoinRowSet Objects

7.DatabaseMetaData-Object

8.ParameterMetaData-Object

9.ResultSetMetaData-Object

10.RowSetMetaData-Object
