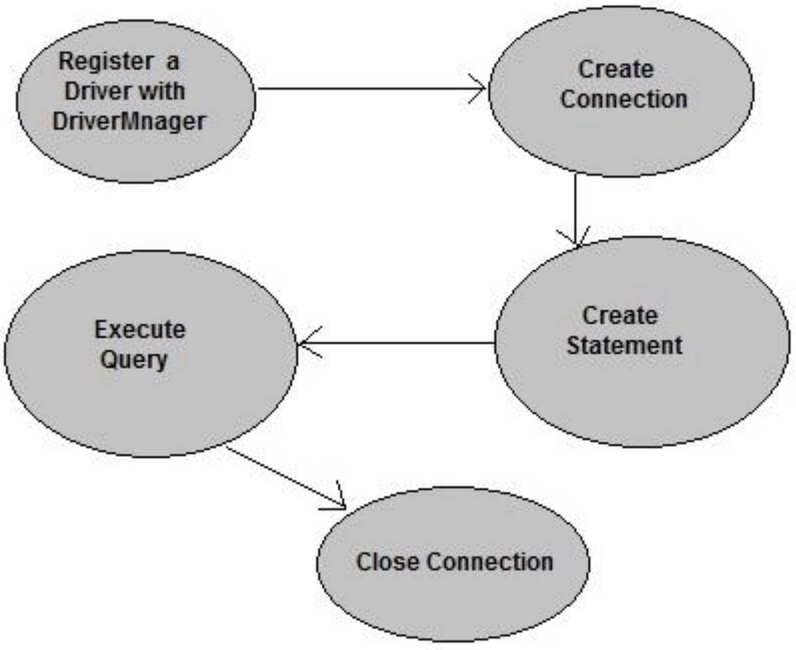
Steps to connect a Java Application to Database

The following 5 steps are the basic steps involve in connecting a Java application with Database using JDBC.

1. Register the Driver
2. Create a Connection
3. Create SQL Statement
4. Execute SQL Statement
5. Closing the connection



Register the Driver

forName(): The forName() method of Class class is used to register the driver class. This method is used to dynamically load the driver class.

Syntax:

```
Class.forName("Database drivername");
```

Example

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

In the above example Class is a class of java.lang package.

Create a Connection

getConnection(): The getConnection() method of DriverManager class is used to establish connection with the database.

Example

```
Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","password");
```

Connection: Connection is an interface present in java.sql package. A Connection is a session in a specific database design engine. Here Connection is established at runtime.

con: It is a reference of Connection Interface.

Explanation

In above example we are using Oracle10g as the database. So we need to know following information's for the oracle database:

1. **Driver class:** The driver class for the oracle database is **oracle.jdbc.driver.OracleDriver**.
2. **Connection URL:** The connection URL for the oracle10G database is **jdbc:oracle:thin:@localhost:1521:xe** where jdbc is the API, oracle is the database, thin is the driver, localhost is the server name on which oracle is running, we may also use IP address, 1521 is the port number and XE is the Oracle service name. You may get all these information's from the **tnsnames.ora file.**
3. **Username:** The default username for the oracle database is system.
4. **Password:** Password is given by the user at the time of installing the oracle database.

Create SQL Statement

createStatement(): The createStatement() method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database.

Example

```
Statement stmt=con.createStatement();
```

Statement: It is predefined interface present in java.sql package. This interface used to execute the SQL statement by which we can implement DML, DDL operation i.e., (SELECT, CREATE, INSERT, UPDATE, DELETE, etc.)

con: Here con is the object of Connection interface

Execute SQL Statement

executeQuery(): The executeQuery() method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table.

Example

```
ResultSet rs=stmt.executeQuery("select * from emp");
while(rs.next())
{
System.out.println(rs.getInt(1)+"   "+rs.getString(2));
}
```

Close the connection object

By closing connection object statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection.

Syntax

```
con.close();
```

Loading jar file:

To connect your java application with Oracle, you will also need to load ojdbc14.jar file.

Copy the jar file into C:\Program Files\Java\jre7\lib\ext folder.

Download ojdbc14.jar file

Example

Let's see programs for performing simple operations like create, insert and select on database tables.

Create a table in Oracle Database

```
Create table emp
(empno number(10),
name varchar(20),
age number(3));
```

Insert some record into the table

```
insert into emp values(101,'Adam',40);
insert into emp values(102,'Scott',35);
insert into emp values(103,'Allen',42);
```

Accessing record from Student table in Java application

```
importjava.sql.*;
class Test
{
    public static void main(String args[])
    {
        try
        {
            //Loading driver
            Class.forName("oracle.jdbc.driver.OracleDriver");

            //creating connection
            Connection con =DriverManager.getConnection
                ("jdbc:oracle:thin:@localhost:1521:XE","username","password");

            Statement s=con.createStatement();        //creating statement
                                            //executing statement
            ResultSetrs=s.executeQuery("select * from Student");
            while(rs.next())
            {
                System.out.println(rs.getInt(1)+" "+rs.getString(2));
            }

            con.close();        //closing connection
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

Output

```
101 Adam
102 Scott
103 Allen
```

Inserting record into a table using java application

```
importjava.sql.*;
classTest
{
    publicstaticvoidmain(String []args)
    {
    try{
        //Loading driver...
        Class.forName("oracle.jdbc.driver.OracleDriver");

        //creating connection...
        Connection con =DriverManager.getConnection
            ("jdbc:oracle:thin:@localhost:1521:XE","username","password");

        PreparedStatementpst=con.prepareStatement("insert into Student values(?,?)");

        pst.setInt(1,104);
        pst.setString(2,"Alex");
        pst.executeUpdate();

        con.close();   //closing connection
    }
    catch(Exception e)
    {
        e.printStacktrace();
    }
   }
}
```