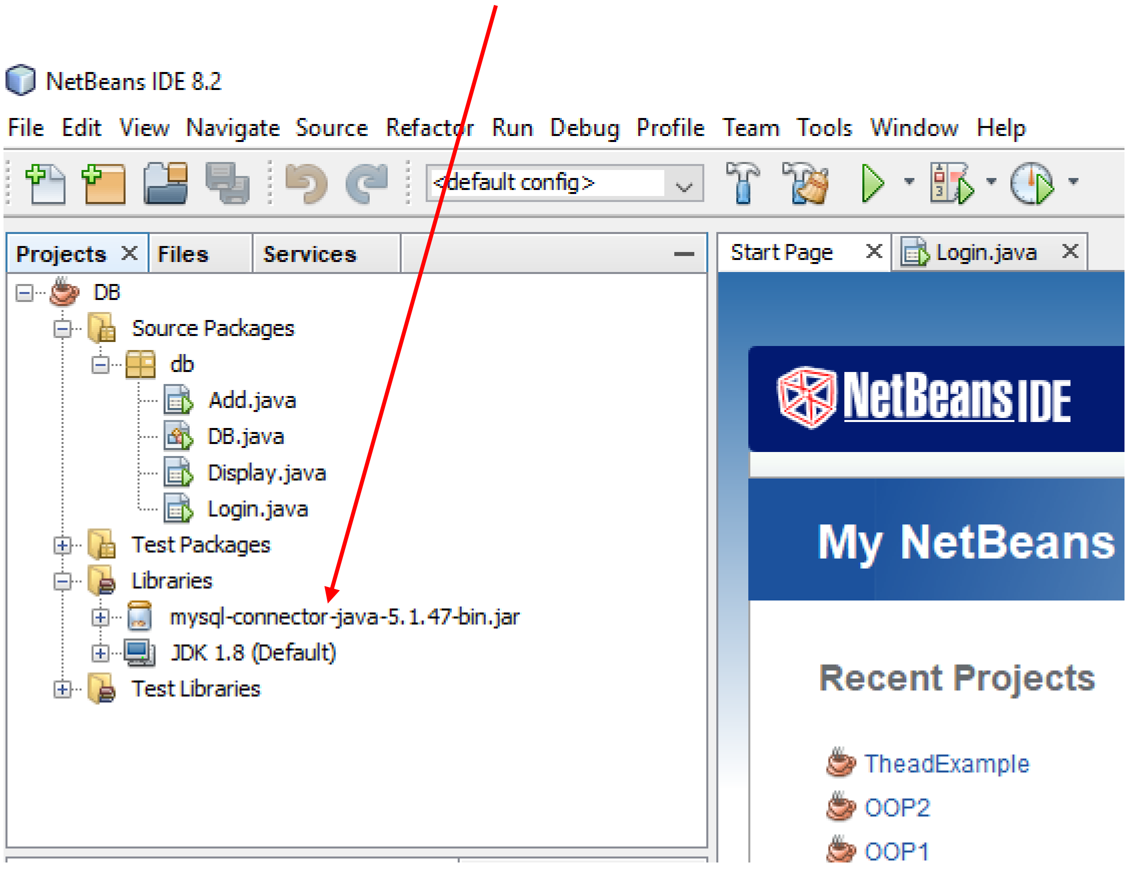
Java Database Connectivity (JDBC) with Swings

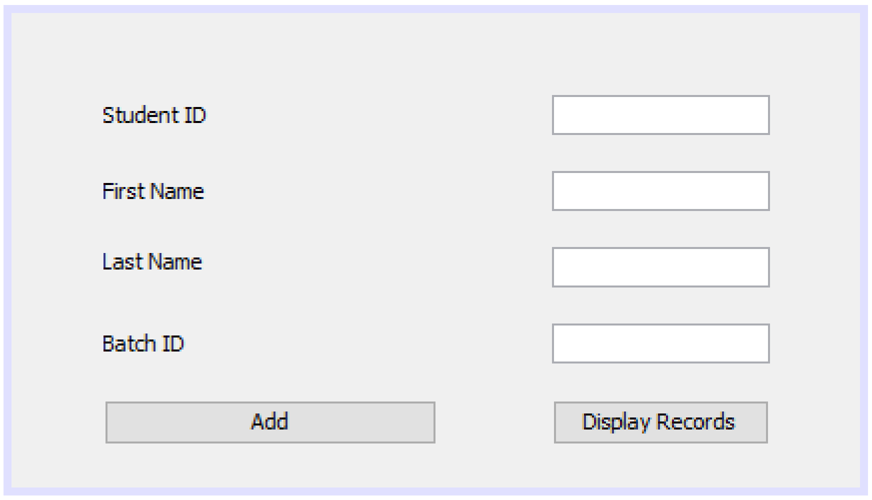
Step 1

Adding records into Database Table

1. Add mySql connector into the libraries.



2. Design the following UI.



3. Import the following two packages.

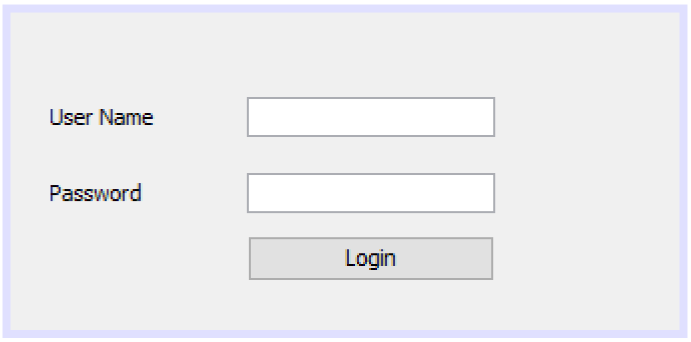
```
import java.sql.*;

import javax.swing.*;
```

4. Add the following to the command button.

```
// TODO add your handling code here:
        String id,fn,ln,bi;
        id=jTextField1.getText();
        fn=jTextField2.getText();
        ln=jTextField3.getText();
        bi=jTextField4.getText();
        int iid=Integer.parseInt(id);
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sample","root","");
            Statement stmt=con.createStatement();
            String sql;
            sql="INSERT INTO student VALUES("+iid+",'"+fn+"','"+ln+"','"+bi+"')";
            stmt.executeLargeUpdate(sql);
            JOptionPane.showMessageDialog(this, " record is inserted ");
            con.close();
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
```

Step 2

Login Screen



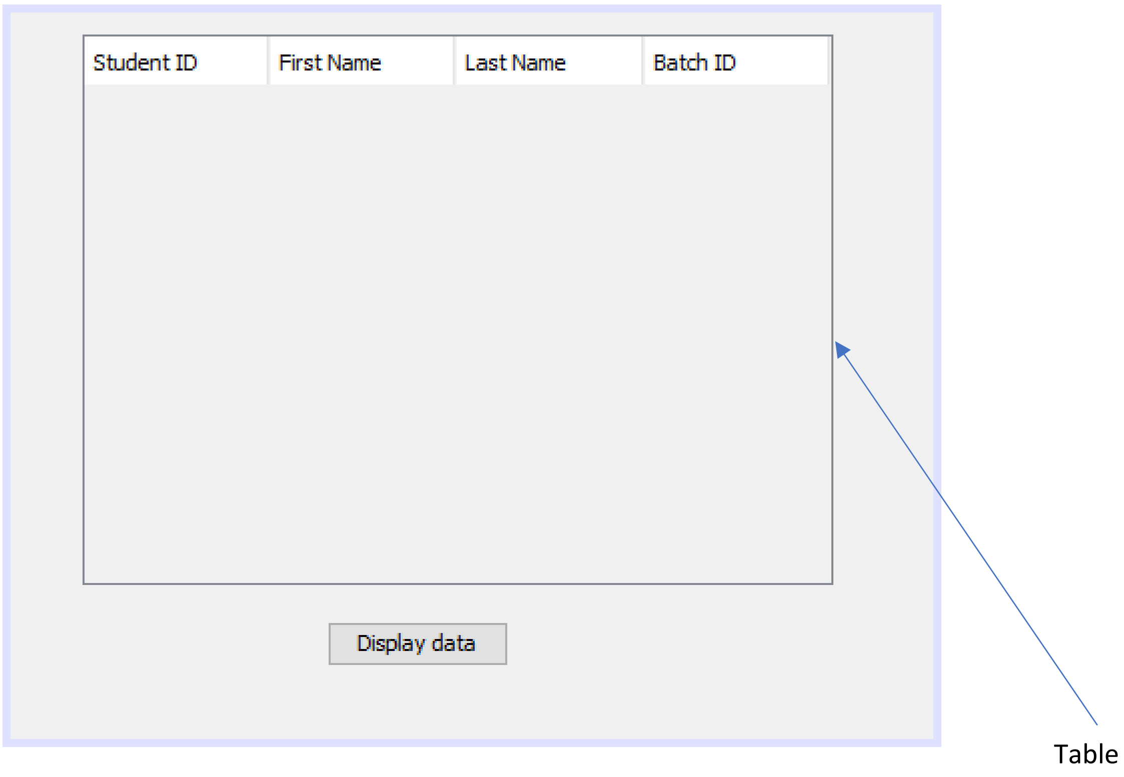
```
// TODO add your handling code here:
        // TODO add your handling code here:
       String id,pwd;
       id=jTextField1.getText();
       pwd=jPasswordField1.getText();
           try
       {
          Class.forName("com.mysql.jdbc.Driver");
          Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sample","root","");
          Statement stmt=con.createStatement();
          String sql;
          sql="Select * from login where User_Id='"+id+"' AND Password='"+pwd+"'";
          ResultSet rs=stmt.executeQuery(sql);
          if(rs.next())
          {
             dispose();
             Add a=new Add();
             a.show();
          }
          else
          {
             JOptionPane.showMessageDialog(this, "Invalid User Name or Password");
```

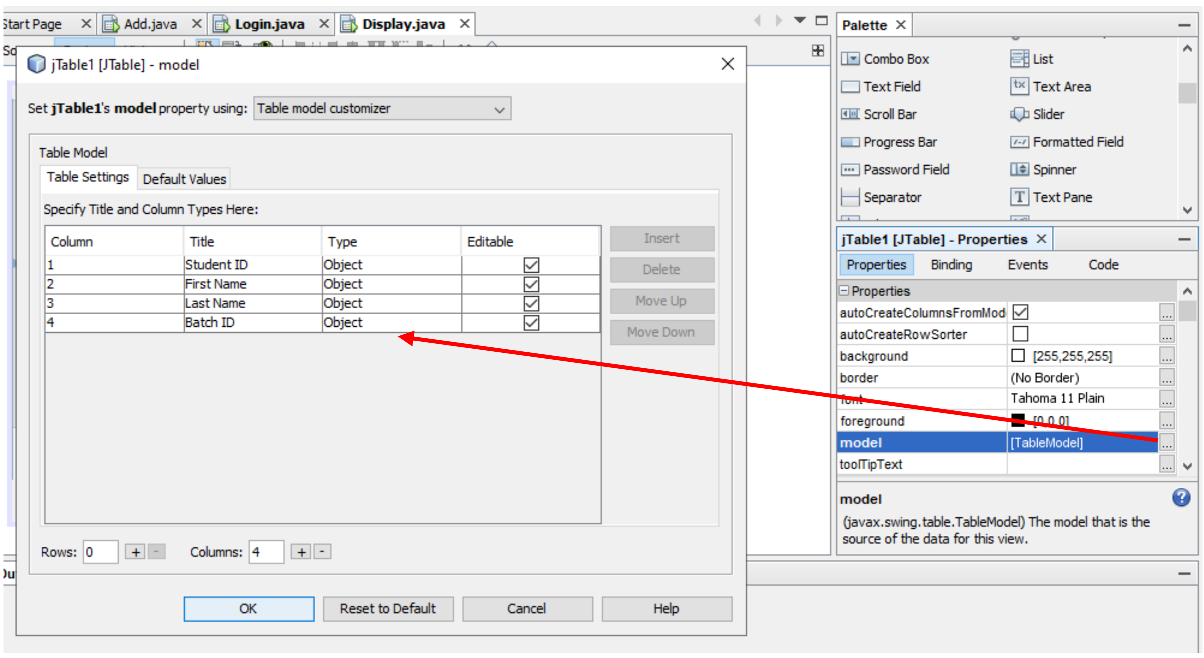
```
            jTextField1.setText(" ");

            jPasswordField1.setText(" ");

        }

        con.close();

    }

    catch(Exception e)

    {

        System.out.println(e.getMessage());

    }
```

Step 3

Displaying data in a Table

1. Add ‘Table’ and change the model properties (Refer to the model property in the below)





jTable1 [JTable] - model

Set **jTable1**'s **model** property using: Table model customizer

Table Model

Table Settings | Default Values

Specify Title and Column Types Here:

Column	Title	Type	Editable
1	Student ID	Object	✓
2	First Name	Object	✓
3	Last Name	Object	✓
4	Batch ID	Object	✓

Rows: 0 Columns: 4

jTable1 [JTable] - Properties

Properties | Binding | Events | Code

Properties	
autoCreateColumnsFromMod	✓
autoCreateRowSorter	☐
background	[255,255,255]
border	(No Border)
font	Tahoma 11 Plain
foreground	[0,0,0]
model	[TableModel]
toolTipText	

model

(javax.swing.table.TableModel) The model that is the source of the data for this view.

Step 4

Add the following code under display button

```
    try
    {
       Class.forName("com.mysql.jdbc.Driver");
       Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sample","root","");
       Statement stmt=con.createStatement();
       String sql;
       sql="SELECT * FROM student";
       ResultSet rs=stmt.executeQuery(sql);
       while(rs.next())
       {
          String id=String.valueOf(rs.getInt(1));
          String fn=rs.getString(2);
          String ln=rs.getString(3);
          String bi=rs.getString(4);

          String tbData[]={id,fn,ln,bi};
```

```
        DefaultTableModel tblModel=(DefaultTableModel)jTable1.getModel();
        tblModel.addRow(tbData)
      }
      con.close();
```

Step 5

Add the following code to run the Login form as the first screen to display during the run time.

Here Login is the name of the Login form.

```
package db;
public class DB
{
   public static void main(String[] args)
   {
      Login l=new Login();
      l.show();
   }

}
```

Some additional note

PreparedStatement interface

The PreparedStatement interface is a subinterface of Statement. It is used to execute parameterized query.

Let's see the example of parameterized query:

```
1. String sql="insert into emp values(?,?,?)";
```

As you can see, we are passing parameter (?) for the values. Its value will be set by calling the setter methods of PreparedStatement.

Why use PreparedStatement?

Improves performance: The performance of the application will be faster if you use PreparedStatement interface because query is compiled only once.

How to get the instance of PreparedStatement?

The prepareStatement() method of Connection interface is used to return the object of PreparedStatement. Syntax:

```
public PreparedStatement prepareStatement(String query)throws SQLException{}
```

Example of PreparedStatement interface that inserts the record

```
import java.sql.*;
class InsertPrepared{
public static void main(String args[]){
try{
Class.forName("oracle.jdbc.driver.OracleDriver");

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

PreparedStatement stmt=con.prepareStatement("insert into Emp values(?,?)");

stmt.setInt(1,101);//1 specifies the first parameter in the query
stmt.setString(2,"Ratan");

int i=stmt.executeUpdate();
System.out.println(i+" records inserted");

con.close();

}catch(Exception e){ System.out.println(e);}

}
}
```

Example of PreparedStatement interface that updates the record

```
PreparedStatement stmt=con.prepareStatement("update emp set name=? where id=?");
stmt.setString(1,"Sonoo");//1 specifies the first parameter in the query i.e. name
stmt.setInt(2,101);

int i=stmt.executeUpdate();
System.out.println(i+" records updated");
```

Example of PreparedStatement interface that deletes the record

```
PreparedStatement stmt=con.prepareStatement("delete from emp where id=?");
stmt.setInt(1,101);

int i=stmt.executeUpdate();
System.out.println(i+" records deleted");
```

Example of PreparedStatement interface that retrieve the records of a table

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

Ref: https://www.javatpoint.com/PreparedStatement-interface