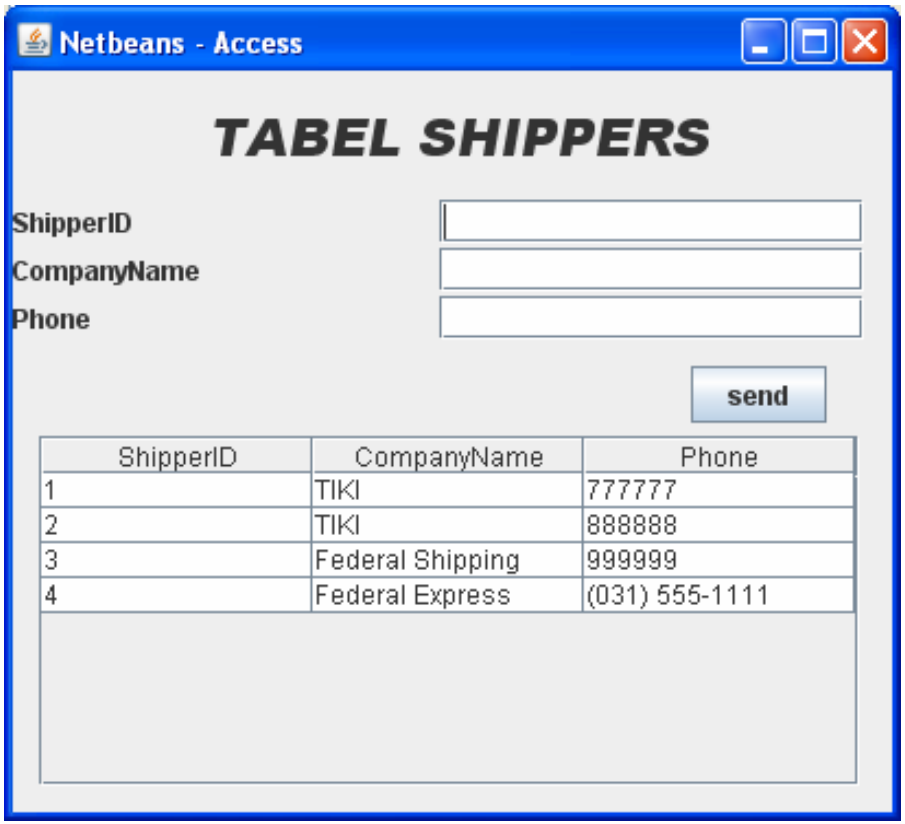
Membuat Aplikasi Database dengan Netbeans

Yuliana Setiowati
Politeknik Elektronika Negeri Surabaya

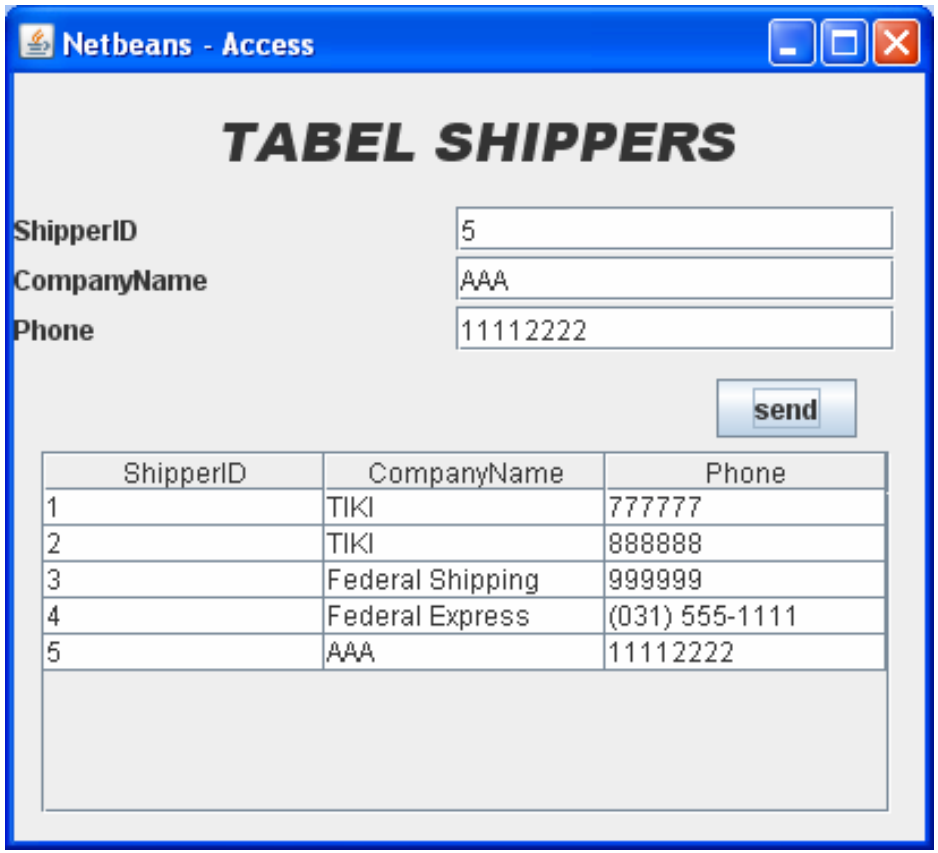
Membuat Aplikasi Database dengan Netbeans

- Pada pertemuan ini akan dibahas tentang aplikasi database sederhana menggunakan database Access.



Membuat Aplikasi Database dengan Netbeans

- Menambahkan data



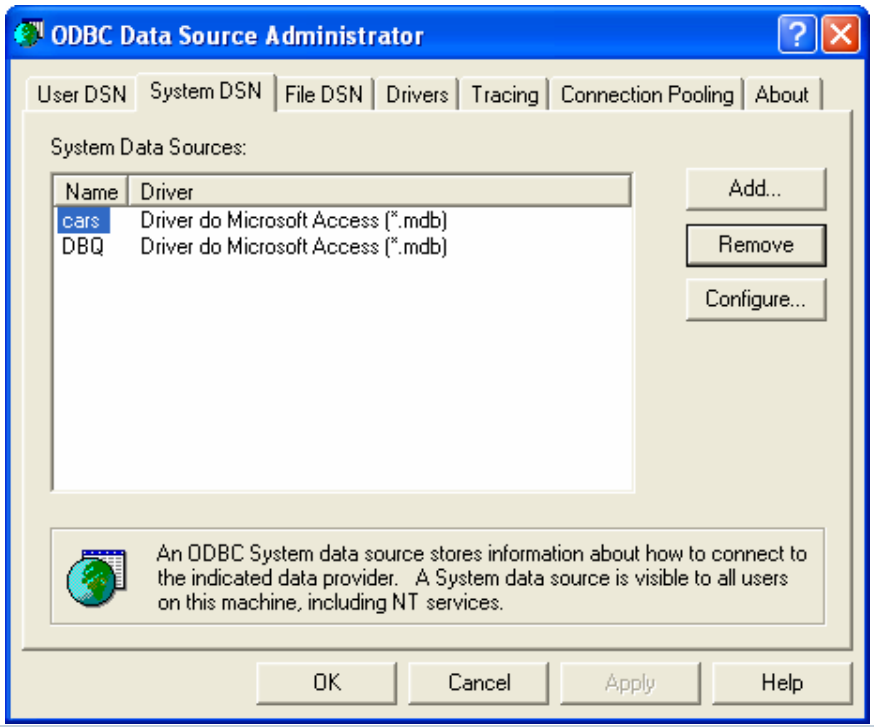
Database yang Digunakan

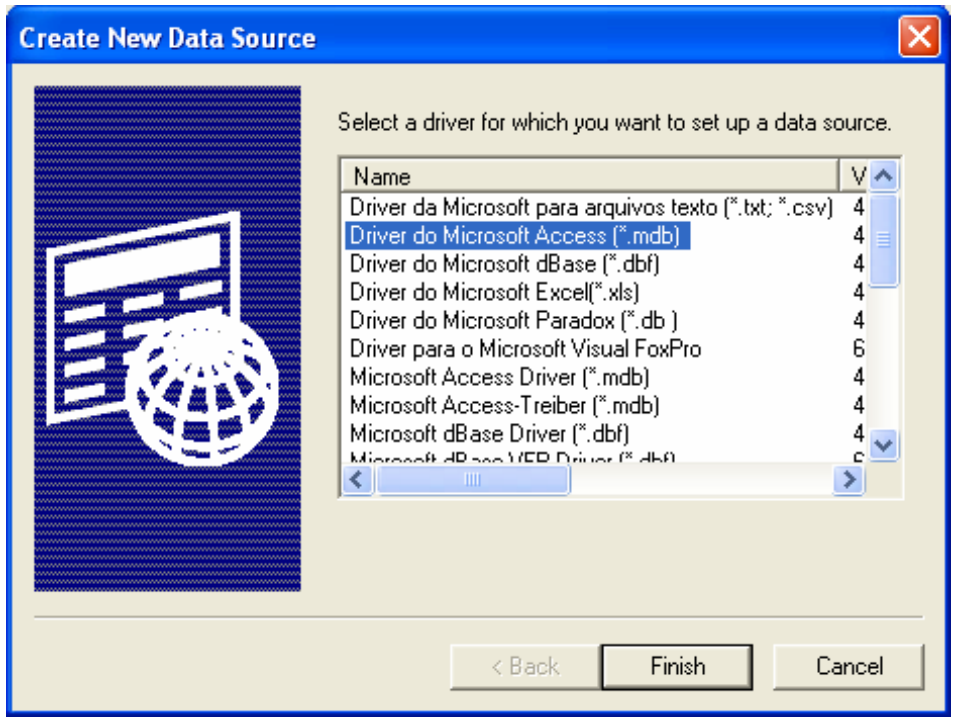
- Database : Northwind
- Tabel : Shippers

Field	Tipe Data	Keterangan
ShipperID	Int	Menyatakan id
CompanyName	Text	Nama perusahaan
Phone	Text	No telp

Menggunakan Microsoft Access via ODBC

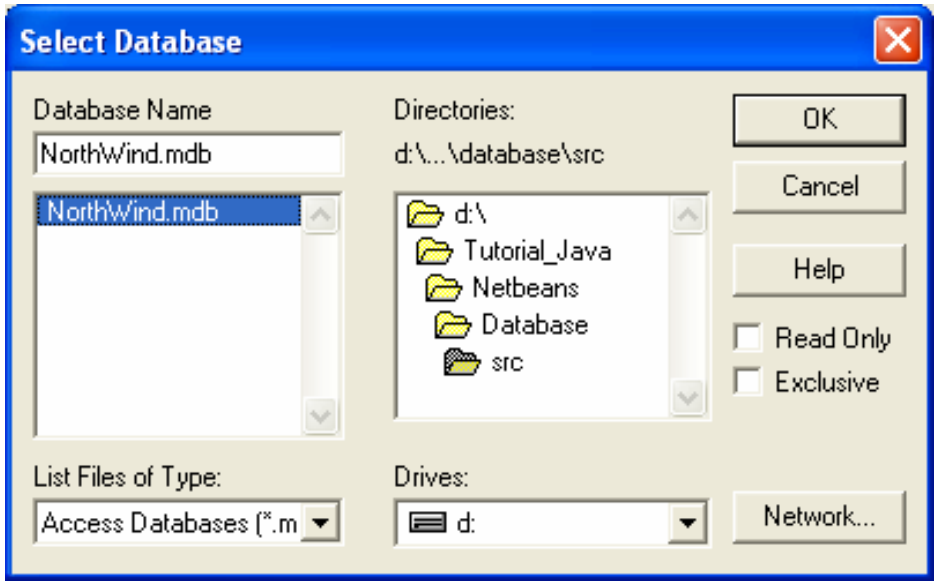
- **Click Start → Control Panel → Administrative Tools → Data Sources(ODBC) → System DSN, dan pilih Add**
- **Pada form "Create New Data Source" pilih Driver do Microsoft Access (*.mdb)**

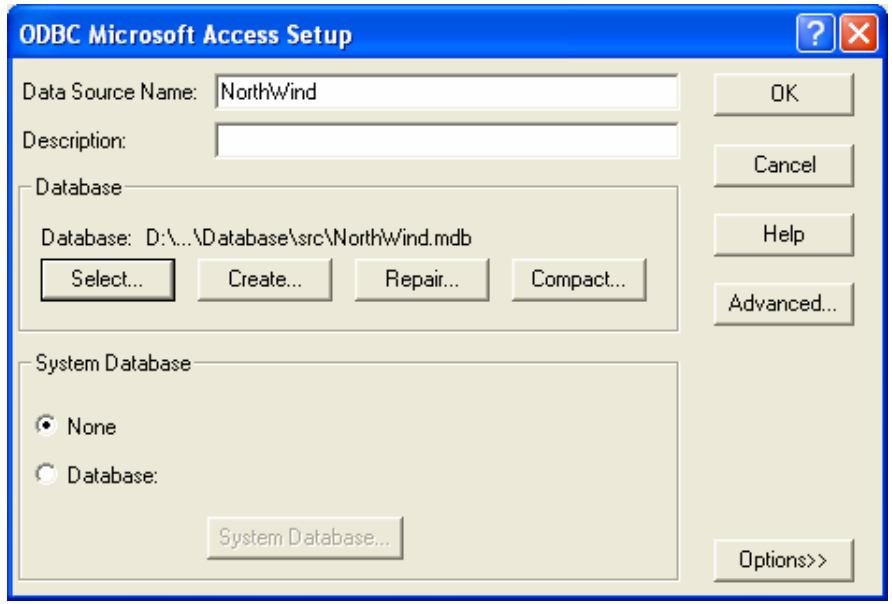




Menggunakan Microsoft Access via ODBC

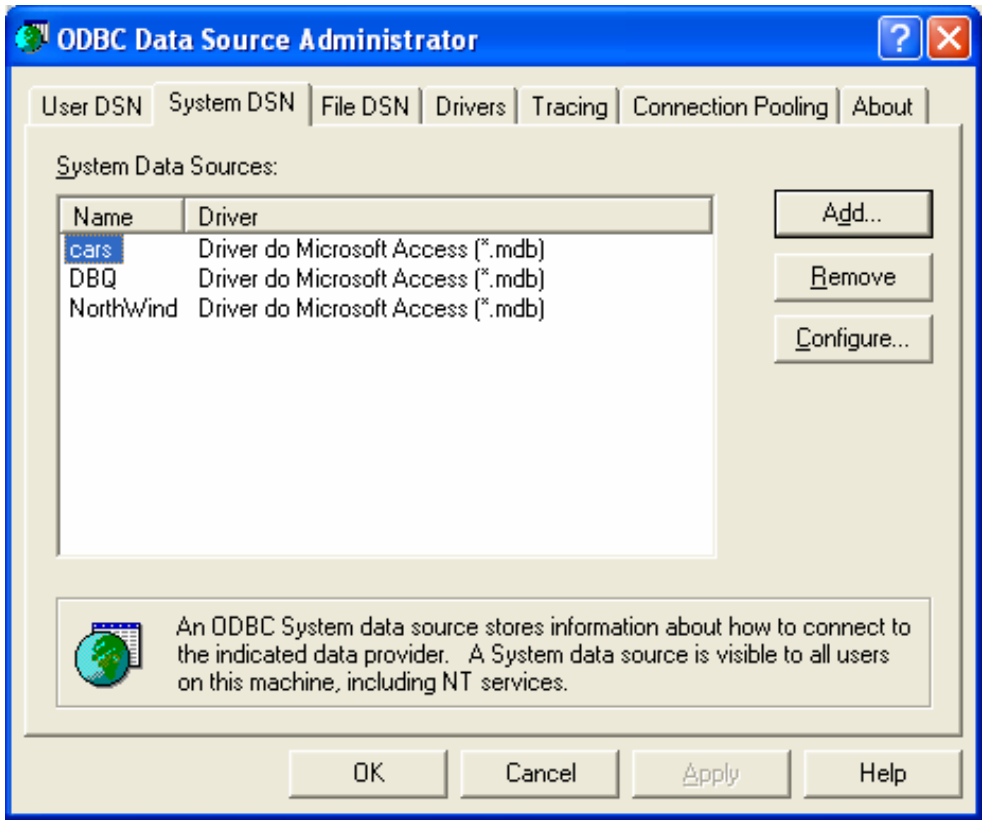
- Pada form ODBC Microsoft Access Setup tentukan nama data source dan letak database.





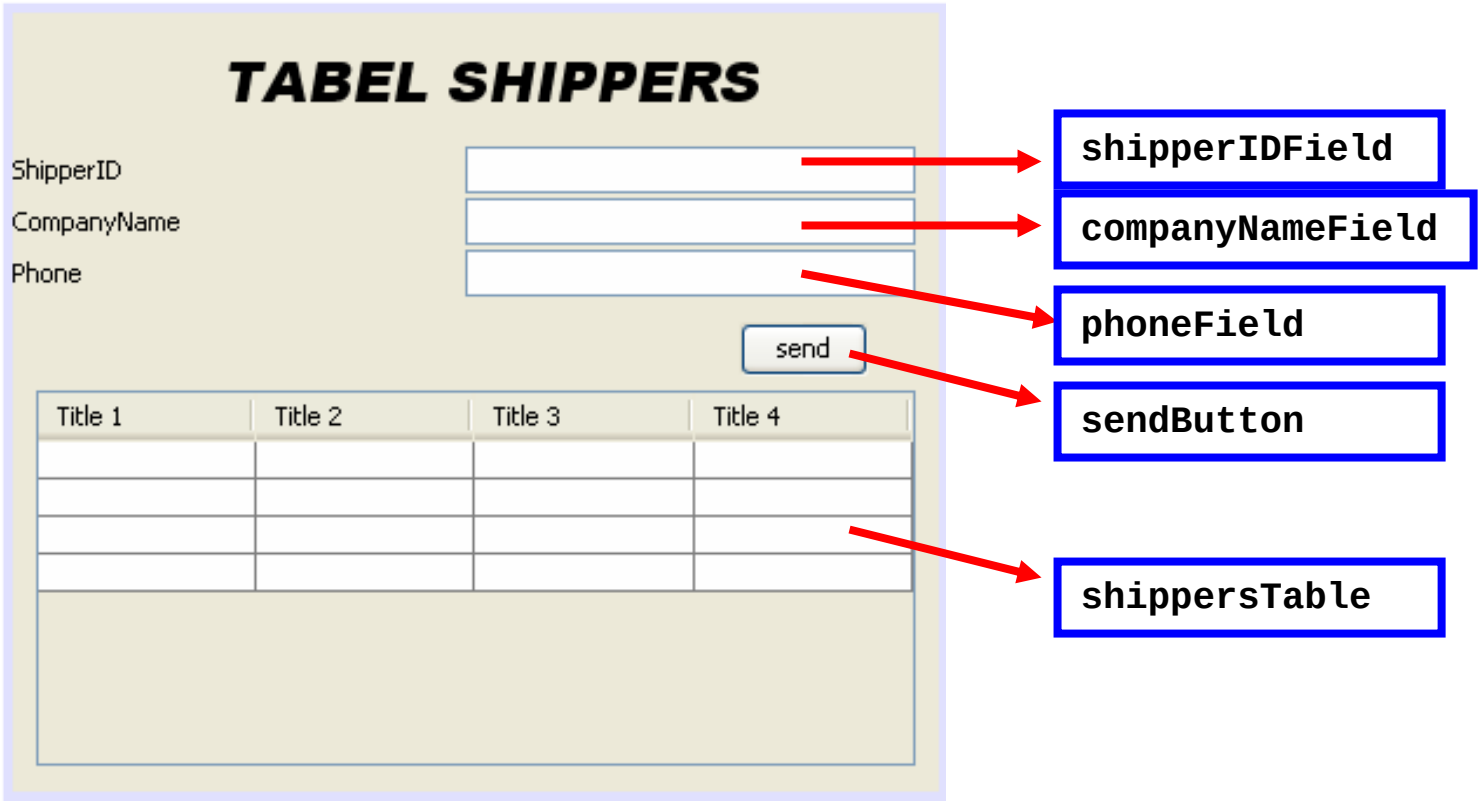
Menggunakan Microsoft Access via ODBC

- Setelah proses selesai maka akan terdapat Northwind pada System DSN.



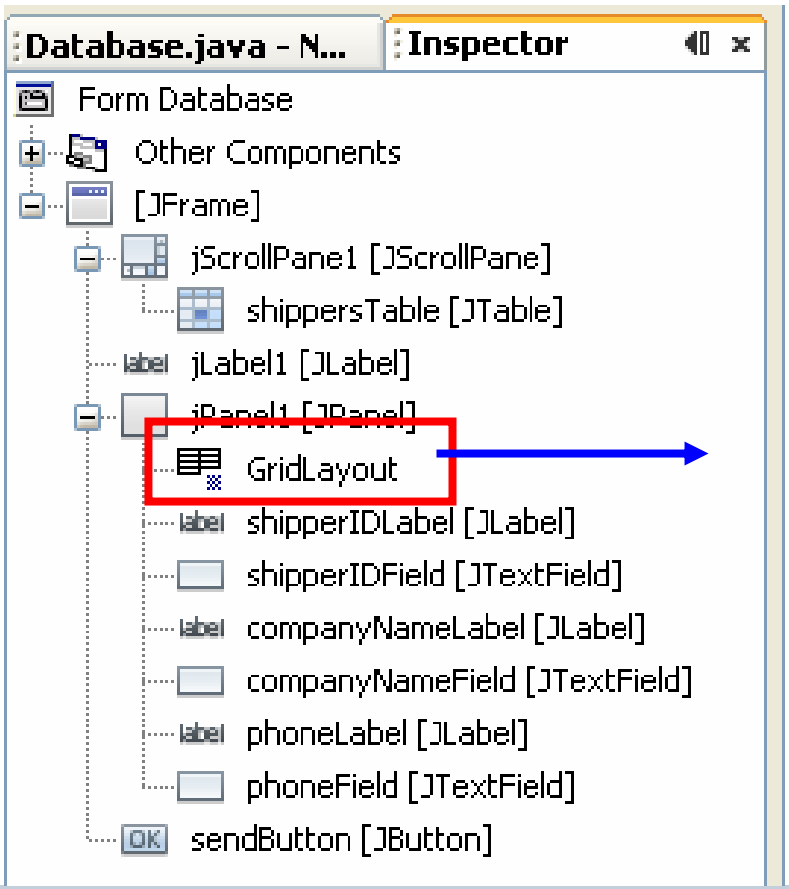
Perancangan Aplikasi Database

- Buatlah form seperti dibawah ini dengan nama Database.java



Perancangan Aplikasi Database

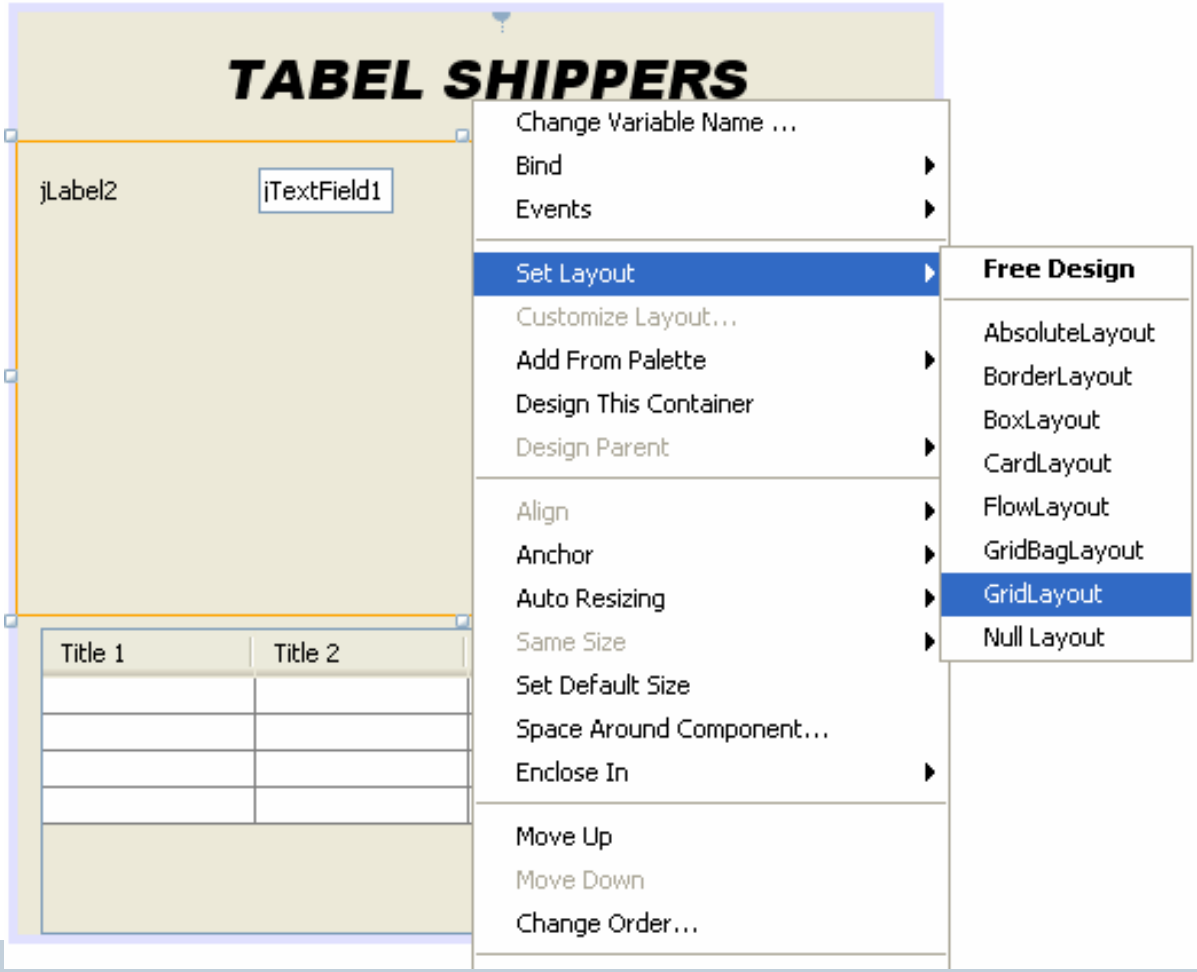
- Pada jendela inspector.



Menggunakan GridLayout pada object JPanel

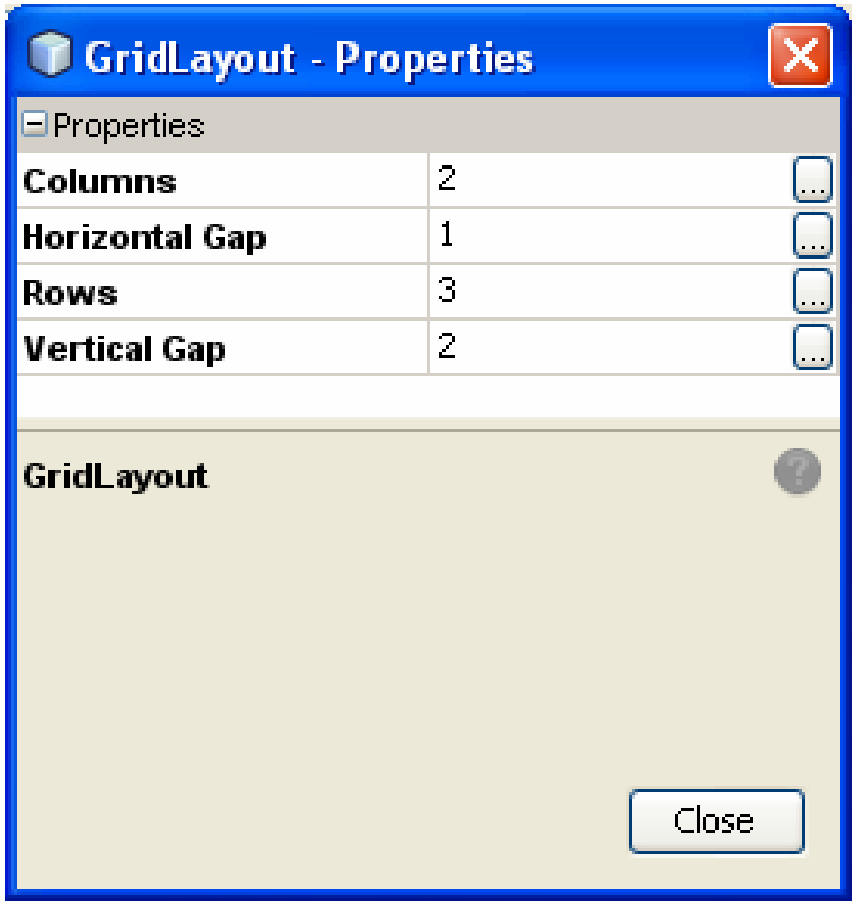
Perancangan Aplikasi Database

- Cara menggunakan GridLayout pada object JPanel klik kanan → SetLayout → GridLayout



Perancangan Aplikasi Database

- Pada jendela inspector pilih GridLayout klik kanan → properties



GridLayout

- GridLayout adalah cara mengatur komponen/object dalam aturan baris dan kolom.
- Tiap cell mempunyai ukuran yang sama.
- Cara peletakan komponen : dimulai dari ujung kiri atas, yaitu baris teratas grid sebelah kiri dilanjutkan ke kanan. Bila baris teratas sudah penuh, penambahkan komponen akan diletakkan pada baris selanjutnya dimulai dari sebelah kiri.

Perancangan Aplikasi Database

- Untuk menampilkan data pada object JTable gunakan fungsi:

```
public void setModel(TableModel dataModel)
```

- Parameter dari fungsi adalah object dari class TableModel.

Membuat class TableModel 1

```
import java.sql.ResultSet;
import java.util.ArrayList;
import javax.swing.table.AbstractTableModel;

public class ShippersTableModel extends AbstractTableModel {
   private int colnum=3;
   private String[] colNames={
        "ShipperID","CompanyName","Phone");
   private  ArrayList<String[]> ResultSets;

   public ShippersTableModel(){}
   //memecah object ResultSet, data disimpan dalam ArrayList
   public ShippersTableModel(ResultSet rs) {
      ResultSets=new ArrayList<String[]>();
      try{
        while(rs.next()){
             String[] row={
                rs.getString("ShipperID"),rs.getString("CompanyName"), rs.getString("Phone")};
            ResultSets.add(row);
         }
      }
      catch(Exception e){
          System.out.println("Exception in CarTableModel");
      }
    }
```

Membuat class TableModel 1

```
    //mendapatkan data pada field/kolom dan baris yang diinginkan
    public Object getValueAt(int rowindex, int columnindex) {
        String[] row=ResultSets.get(rowindex);
        return row[columnindex];
     }
     //mendapatkan jumlah data
     public int getRowCount() {
         return ResultSets.size();
     }
     //untuk mendapatkan jumlah field/kolom
     public int getColumnCount() {
         return colnum;
     }
     //untuk mendapatkan nama kolom pada indeks tertentu
     public String getColumnName(int param) {
        return colNames[param];
     }
}
```

Membuat class TableModel 1

- Class ShippersTableModel merupakan class TableModel.
- Class ini untuk memecah object Result Set menjadi data-data yang tersimpan dalam ArrayList.
- Isi ArrayList berupa Array String

Membuat class TableModel 2

```
import java.sql.ResultSet;
import java.util.Vector;
import javax.swing.table.AbstractTableModel;

public class ShippersTableModel2 extends AbstractTableModel{
   private Vector v = new Vector() ;
   private String[] columnNames={
        "ShipperID","CompanyName","Phone");
   private Object[][] data = new Object[50][3];
   private int brs ;
   public ShippersTableModel2(){}
   //memecah object ResultSet, data disimpan dalam Array Dimensi 2
   public ShippersTableModel2(ResultSet rs) {
      brs =0;
      try{
        while(rs.next()){
             data[brs][0] = rs.getString("ShipperID") ;
             data[brs][1] = rs.getString("CompanyName") ;
             data[brs][2] = rs.getString("Phone") ;
             brs++ ;
         }
      }
      catch(Exception e){
          System.out.println("Exception in CarTableModel");
      }
    }
```

Membuat class TableModel 2

```
    //mendapatkan data pada field/kolom dan baris yang diinginkan
    public Object getValueAt(int rowindex, int columnindex) {
         return data[rowindex][columnindex];
    }
    //mendapatkan jumlah data
    public int getRowCount() {
        return brs ;
    }
    //untuk mendapatkan jumlah field/kolom
    public int getColumnCount() {
        return columnNames.length ;
    }
     //untuk mendapatkan nama kolom pada indeks tertentu
    public String getColumnName(int param) {
        return columnNames[param];
    }
}
```

Membuat class TableModel 2

- Cara lain ditunjukkan pada Class ShippersTableModel2
- Class ini untuk memecah object Result Set menjadi data-data yang tersimpan dalam Array Dimensi 2.
- Tipe Array ini adalah class Object.

Langkah – langkah JDBC

- Membangun sebuah koneksi ke sumber data (data source).
- Mengirim statement ke sumber data
- Memproses object ResultSet.
- Menutup koneksi

Membangun sebuah koneksi ke sumber data (data source)

- Load Driver

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

- Menentukan letak database

```
String dbname = "jdbc:odbc:NorthWind";
```

- Membuat koneksi database dengan aplikasi

```
con = DriverManager.getConnection(dbname,"","");
```

Membangun sebuah koneksi ke sumber data (data source)

```
public void Koneksi(){
     String dbname = "jdbc:odbc:NorthWind";
     try{
          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          con = DriverManager.getConnection(dbname,"","");
     }
     catch(ClassNotFoundException ex) {
      System.err.println("Driver Error");
      ex.printStackTrace();
      System.exit(1);
    }
     catch(SQLException e){
          System.out.println("Tidak berhasil koneksi");
     }
}
```

Mengirim statement ke sumber data

- Membuat Obyek Statement
  - Object `Statement` digunakan untuk mengirim query dan perintah ke database.
  - Object `Statement` dibuat dengan cara bekerjasama dengan class `Connection`.
  - con adalah object Connection dan st adalah object Statement

```
st=con.createStatement();
```

- Mengeksekusi Query
  - Memanfaatkan object `Statement` untuk memproses query.
  - Cara: memanggil method `executeQuery()` dari object `Statement`. → memberikan return value bertipe `ResulSet`
  - st adalah object Statement dan rs adalah object ResultSet

```
rs=st.executeQuery("SELECT * FROM Shippers");
```

Mengirim statement ke sumber data

```
public ResultSet getResultFromShippers(){
     ResultSet rs=null;
          try{
               st=con.createStatement();
               rs=st.executeQuery("SELECT * FROM Shippers");
          }
          catch(SQLException ex){
               ex.printStackTrace();
          }
          return(rs);
}
```

Memproses object ResultSet

- Dilakukan pada constructor class ShippersTableModel

```
//memecah object ResultSet, data disimpan dalam ArrayList
public ShippersTableModel(ResultSet rs) {
   ResultSets=new ArrayList<String[]>();
   try{
     while(rs.next()){
          String[] row={
             rs.getString("ShipperID"),rs.getString("CompanyName"), rs.getString("Phone")};
         ResultSets.add(row);
      }
   }
   catch(Exception e){
       System.out.println("Exception in CarTableModel");
   }
 }
```

Memproses object ResultSet

- Dilakukan pada constructor class ShippersTableModel2

```
//memecah object ResultSet, data disimpan dalam Array Dimensi 2
public ShippersTableModel2(ResultSet rs) {
   brs =0;
   try{
     while(rs.next()){
          data[brs][0] = rs.getString("ShipperID") ;
          data[brs][1] = rs.getString("CompanyName") ;
          data[brs][2] = rs.getString("Phone") ;
          brs++ ;
      }
   }
   catch(Exception e){
       System.out.println("Exception in CarTableModel");
   }
 }
```

Menutup Koneksi

- Harus didefinisikan secara eksplisit.

```
connection.close();
```

```
public void dbClose(){
 try {
    con.close();
 }
 catch(SQLException sqlex){
    System.err.println("Error :Koneksi Database tidak Bisa diputus");
 }
```

Proses untuk Menampilkan Data pada object JTable

```
public class Database extends javax.swing.JFrame {
    private Connection con ;
    private Statement st;

    /** Creates new form Database */
    public Database(){
        super("Netbeans - Access");
        initComponents();
        Koneksi();
        ResultSet rs = getResultFromShippers();
        shippersTable.setModel(new ShippersTableModel(rs));
    }
```

- atau

```
shippersTable.setModel(new ShippersTableModel2(rs));
```

Mengambil data dari TextField

- Mendapatkan shipperID, CompanyName dan Phone dari TextField

```
String id=shipperIDField.getText();
String name=companyNameField.getText();
String ph=phoneField.getText();
```

Menambahkan data

- Membuat object Statement
- Menjalankan query untuk menambahkan data dengan menjalankan fungsi executeUpdate(String).
- Fungsi ini mengembalikan nilai berupa int (menyatakan jumlah baris yang berhasil ditambahkan, jika tidak berhasil menambahkan data akan mengembalikan nilai 0)

```
st = con.createStatement();
insertStr="insert into Shippers (ShipperID, CompanyName, Phone) values("
        +quotate(id)+","
        +quotate(name)+","
        +quotate(ph)
        +")";
int done=st.executeUpdate(insertStr);
```

```
public String quotate(String content){
    return "'"+content+"'";
 }
```

Menampilkan data pada object JTable

- Setelah melakukan penambahan data, maka data pada database akan bertambah.
- Ingat !!! Jangan lupa untuk menampilkan kembali database pada object JTable

```
//setelah data masuk ke database
//jangan lupa untuk menampilkan kembali ke JTable
ResultSet rs = getResultFromShippers();
shippersTable.setModel(new ShippersTableModel(rs));
```

Menambahkan data

```
private void sendButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String id=shipperIDField.getText();
    String name=companyNameField.getText();
    String ph=phoneField.getText();
    String insertStr="";

    try{
        st = con.createStatement();
        insertStr="insert into Shippers (ShipperID, CompanyName, Phone) values("
                +quotate(id)+","
                +quotate(name)+","
                +quotate(ph)
                +")";
        int done=st.executeUpdate(insertStr);
        //setelah data masuk ke database
        //jangan lupa untuk menampilkan kembali ke JTable
        ResultSet rs = getResultFromShippers();
        shippersTable.setModel(new ShippersTableModel(rs));
    } catch(Exception e){
        //commentLabel.setText("Error occurred in inserting data");
        e.printStackTrace();
    }
}
```

Download

- Silakan download aplikasi database dengan netbeans yang terdapat dalam ppt

  http://lecturer.eepis-its.edu/~yuliana/Prog%20Lanjut/JDBC/Database.rar
- Materi dalam bentuk doc (dalam dir yang sama)

  Tutorial Connecting Access-Netbeans.pdf