
Bab 9

Mengakses Database Dasar

POKOK BAHASAN:

- JDBC
- JDBC API
- Langkah-langkah menggunakan JDBC
- Langkah-langkah membuat data source

TUJUAN BELAJAR:

Setelah mempelajari bab ini, mahasiswa diharapkan mampu:

1. Memahami langkah – langkah dasar menggunakan JDBC.
2. Mengetahui cara membuat datasource.
3. Mengetahui cara membuat tabel.
4. Mengetahui cara memasukan data ke tabel.

JSP sebagai teknologi untuk pembuatan aplikasi web memiliki kemampuan menangani database dengan menggunakan JDBC sebagai perantara antara program JSP dengan database server. Oleh karena itu untuk memahami bagaimana membuat program JSP yang dapat mengakses database harus mengerti teknologi JDBC terlebih dahulu.

9.1 JDBC

JDBC adalah Application Programming Interface (API) yang dirancang untuk mengakses database universal berdasarkan SQL. JDBC terdiri atsa JDBC 1.0 API yang

memberikan fungsi-fungsi dasar untuk akses data. JDBC 2.0 API memberikan tambahan ke fungsi-fungsi dasar dengan kelebihan-kelebihan lain yang lebih mutakhir.

9.2 JDBC API

JDBC adalah suatu nama trade mark, bukan sebuah singkatan. Tapi JDBC sering dikira singkatan dari Java Database Connectivity. JDBC API terdiri dari sejumlah class dan interface yang ditulis dalam bahasa Java yang menyediakan API standar sebagai alat bantu bagi pembuat program dan memberikan kemungkinan untuk menulis aplikasi database dengan menggunakan semua Java API.

JDBC API memudahkan untuk mengirim statement SQL ke sistem database relasional dan mendukung bermacam-macam bahasa SQL. Keunggulan JDBC API adalah sebuah aplikasi dapat mengakses sembarang sumber data dan dapat berjalan pada sembarang platform yang mempunyai Java Virtual Machine(JVM). Sehingga kita tidak perlu menulis satu program untuk mengakses database Sybase, Oracle atau Access dan lain-lain. Kita cukup menulis satu program yang menggunakan JDBC API, dan program dapat mengirimkan statement SQL atau statement lain ke sumber data tertentu. Dengan aplikasi yang ditulis dalam bahasa Java seseorang tidak perlu khawatir untuk menulis aplikasi yang berbeda-beda agar dapat berjalan pada platform yang berbeda-beda. Teknologi JDBC mampu untuk melakukan tiga hal berikut:

1. Membangun sebuah koneksi ke sumber data (data source).
2. Mengirim statement ke sumber data.
3. Memproses hasil dari statement tersebut

9.3 Langkah-langkah menggunakan JDBC

1. Load driver

Untuk mengaktifkan hubungan antar aplikasi dan database, maka sebuah Connection harus dibentuk dengan menggunakan JDBC Driver. Connection dibentuk melalui satu class `java.sql.DriverManager` dan dua interface, yaitu `java.sql.Driver` dan `java.sql.Connection`. Class untuk JDBC diakses melalui `java.sql.*`. Driver adalah

software yang menangani komunikasi ke database server. Berikut ini adalah jika yang kita gunakan adalah JDBC-ODBC driver.

```
try {
    Class.forName("sun.jdbc.odbc.JdbcDriver");
}
catch (ClassNotFoundException ex) {
    System.err.println("Driver Error");
    ex.printStackTrace();
    System.exit(1);
}
```

Penggunaan Class akan throw `ClassNotFoundException`. Dokumentasi driver akan memberikan nama class yang digunakan.

Contoh :

Oracle :

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

Sybase:

```
Class.forName("com.sybase.jdbc.SybDriver");
```

2. Mendefinisikan koneksi URL

Menspesifikasikan lokasi database server. Untuk mendefinisikan URL bisa menggunakan dokumentasi driver. Untuk penggunaan JDBC di applet maka database server harus berada pada node yang sama dengan letak applet dan menggunakan proxy server yang me "reroute" request database ke actual server. Berikut ini contoh mendefinisikan url:

Untuk database Oracle dan Sybase:

```
String host = "dbhost.yourcompany.com";
String dbName = "someName";
int port = 1234;
String oracleURL = "jdbc:oracle:thin:@" + host + ":"
    + port + ":" + dbName;
```

```
String sybaseURL = "jdbc:sybase:Tds:" + host + ":"  
                + port + ":" + "?SERVICENAME=" + dbName;
```

Untuk database access:

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

3. Membuat koneksi

Membuat koneksi bisa dilakukan dengan cara memanggil method `getConnection()` dari `DriverManager` dengan melewati URL sebagai argumen. Method `getConnection()` akan melempar `SQLException`. Contoh:

```
String username = "jay_debese";  
String password = "secret";  
Connection con = DriverManager.getConnection(oracleURL,  
                                             username, password);
```

Akses ke Driver dan `DriverManager` dapat menyebabkan Exception yang harus dikendalikan oleh program. Misal:

```
try{  
    ... ..  
}  
catch(ClassNotFoundException ex) {  
    System.err.println("Driver Error");  
    ex.printStackTrace();  
    System.exit(1);  
}  
catch(SQLException ex) {  
    System.err.println("Tidak Berhasil Koneksi dengan  
Northwind");  
    System.exit(1);  
}
```

4. Membuat obyek statement

Obyek `Statement` digunakan untuk mengirim query dan perintah ke database. Obyek `statement` dibuat dengan cara bekerjasama dengan class `Connection`. Untuk membuat obyek `Statement` maka kita harus memanggil method `createStatement()` dari `Connection`.

Contoh:

```
Statement statement = connection.createStatement();
```

5. Mengeksekusi query

Untuk mengeksekusi query kita bisa memanfaatkan objek `Statement` untuk memproses hasil query. Caranya adalah dengan memanggil method `executeQuery()` dari objek `Statement`. Method `executeQuery()` akan mengembalikan nilai yang bertipe `ResultSet`.

Contoh:

```
String sql="select col1, col2, col3 from sometable";  
ResultSet rs=statement.executeQuery(sql);
```

Untuk memodifikasi database, gunakan statement `executeUpdate(sql);` yang mendukung string sql `UPDATE, INSERT INTO, DELETE`.

6. Memproses result

Untuk memproses result kita bisa menggunakan method `next()` pada objek `ResultSet` untuk mendapatkan result per satu baris. Selama data masih ada method `next()` akan mengembalikan nilai `true` dan jika sudah tidak ada akan mengembalikan nilai `false`.

Contoh:

```
String nrp;  
String nama;  
while (rs.next()){  
    nrp=rs.getString(1);
```

```
        nama=rs.getString(2);
        System.out.println("NRP : " +nrp);
        System.out.println("NAMA : " +nama);
        System.out.println("-----");
    }
```

Kolom pertama mempunyai index 1 bukan 0. Objek `ResultSet` otomatis akan ditutup bila ada objek `ResultSet` baru. `ResultSet` memberikan bermacam-macam method `getXxx()` dengan parameter index kolom atau nama kolom dan mengembalikan data. Beberapa method yang ada pada `ResultSet` adalah sebagai berikut:

- `findColumn()` untuk mendapatkan index (integer value) berdasarkan nama kolom.
- `getMetaData()` untuk meretrieve informasi mengenai `ResultSet`, returns object `ResultSetMetaData`.
- `wasNull()` untuk mengetahui apakah `getXxx()` menghasilkan SQL null.

7. Menutup koneksi

Karena membuka koneksi adalah mahal, maka penundaan langkah terakhir ini hanya jika masih ada operasi database yang dilakukan. Deklarasi untuk menutup koneksi harus didefinisikan secara eksplisit dengan cara sebagai berikut:

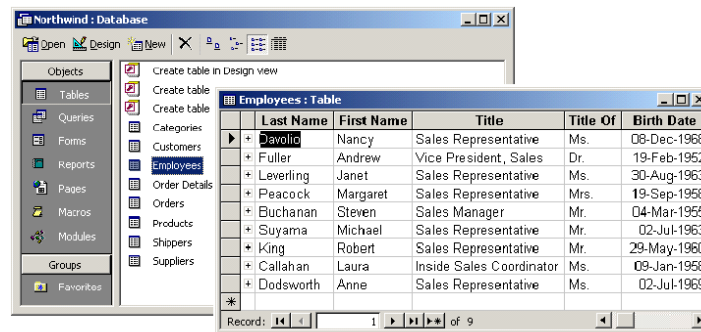
```
connection.close();
```

Membuat Data Source

Berikut ini adalah langkah-langkah untuk mengakses tabel yang tersimpan pada database Acces dengan menggunakan ODBC. Misalnya akan dibuat data source untuk database Northwind dengan karakteristik sebagaimana ditunjukkan pada Gambar 9.1.

9.4 Langkah-langkah membuat data source

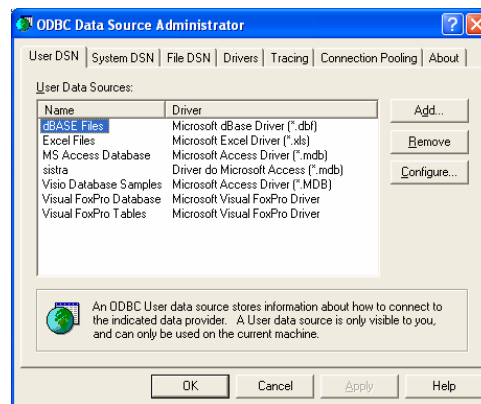
Northwind sample database



- Northwind.mdb located in C:\Program Files\Microsoft Office\Office\Samples
- <http://office.microsoft.com/downloads/2000/Nwind2k.aspx>

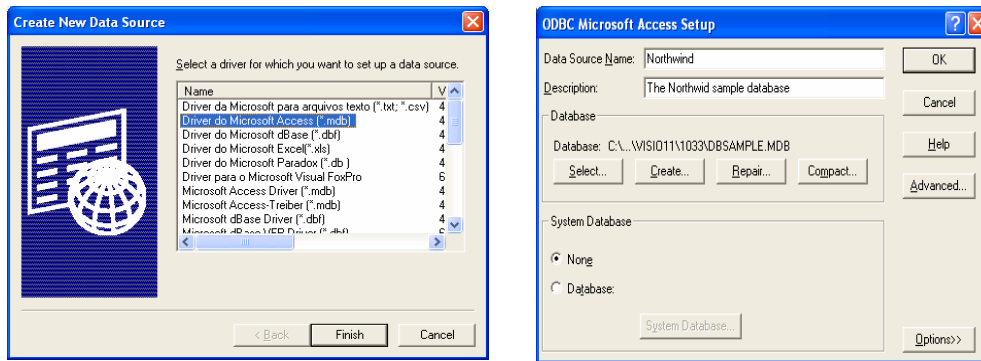
Gambar 9.1 Database Northwind

Klik Start, Settings, Control Panel, Administrative Tools, Data Sources(ODBC), System DSN, dan pilih Add



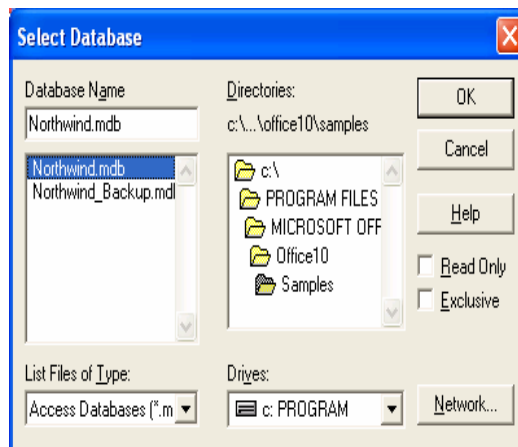
Gambar 9.2 ODBC Source Administrator

Memilih driver Microsoft Access, Finish, ketikkan nama Data Source Name dan tekan Select untuk memilih nama dan lokasi database sebagaimana ditunjukkan Gambar 9.3.

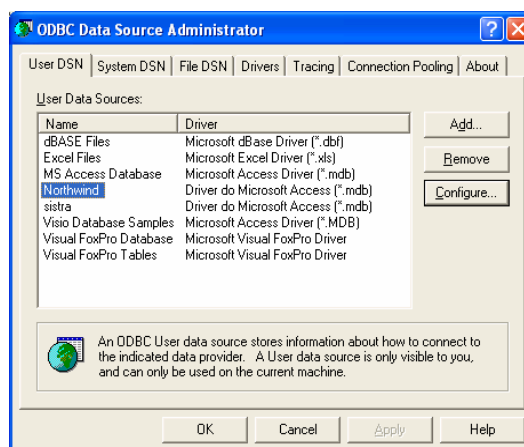


Gambar 9.3 Memilih driver

Navigasi pada directory Samples ms office seperti ditunjukkan pada Gambar 9.4, pilih Northwind.mdb, tekan OK. Hasil pembuatan data source seperti pada Gambar 9.5.



Gambar 9.4 Tampilan navigasi database Northwind



Gambar 9.5 Hasil data source database Northwind

Untuk data source database yang dibuat dengan Access yang bernama Northwind ini maka gunakan `sun.jdbc.JdbcOdbcDriver` sebagai nama class dari JDBC driver

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

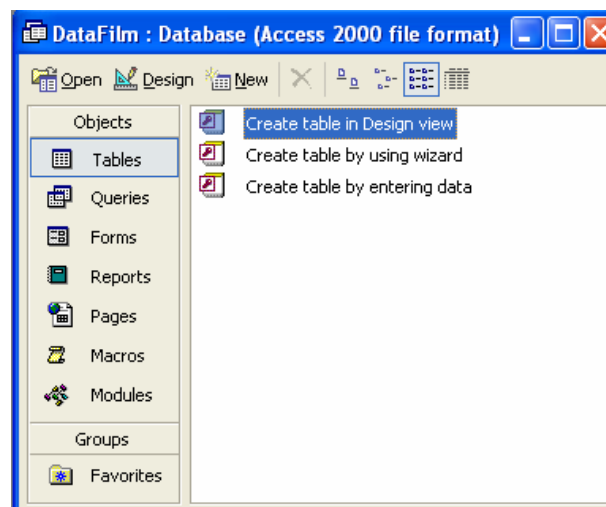
Gunakan “`jdbc:odbc:Northwind`” sebagai alamat database, dan gunakan empty string pada username dan password.

```
Connection
```

```
con=DriverManager.getConnection(jdbc:odbc:Northwind,"","");
```

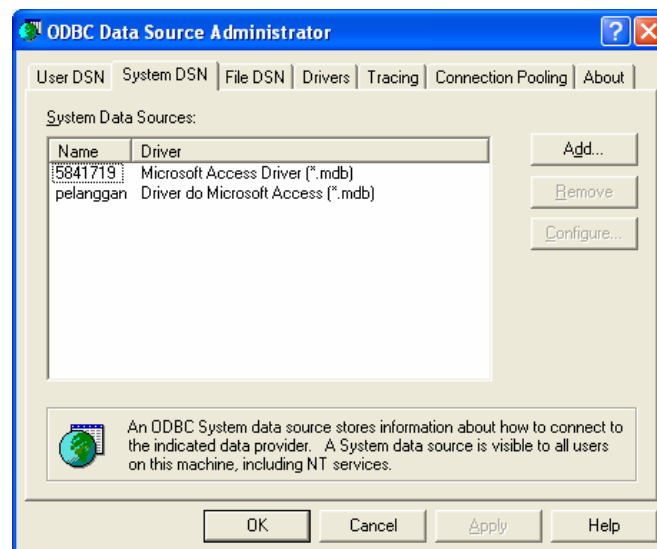
9.5 Percobaan

1. Buat database dengan nama DataFilm. Simpan dalam folder `d:\data\`. Gambar 9.6 adalah tampilan database DataFilm ketika pertama kali dibuat.



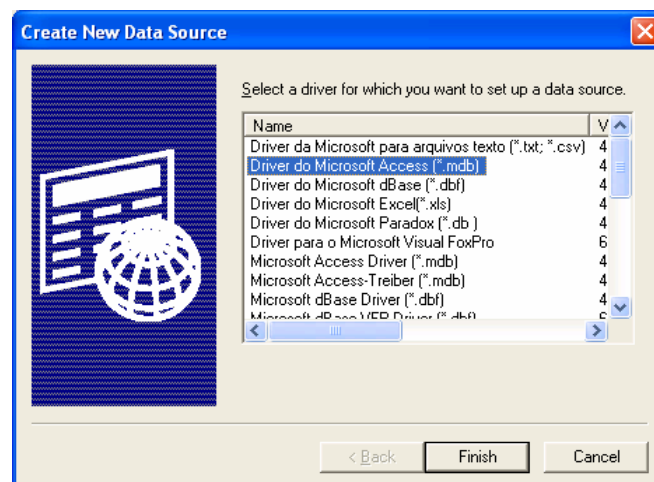
Gambar 9.6 Database DataFilm

2. Membuat data source untuk DataFilm. Berikut ini adalah langkah-langkah untuk mengakses tabel yang tersimpan pada database Acces dengan menggunakan ODBC.
 - a. Click Start, Settings, Control Panel, Administrative Tools, Data Sources(ODBC), System DSN, dan pilih Add seperti pada Gambar 9.7.



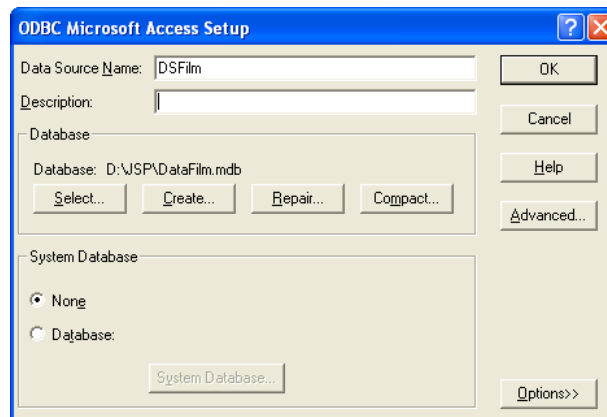
Gambar 9.7 ODBC Data Source Administrator

- b. Memilih driver Microsoft Access, pilih **Driver do Microsoft Access**.
Selanjutnya tekan Finish sebagaimana ditunjukkan pada Gambar 9.8.

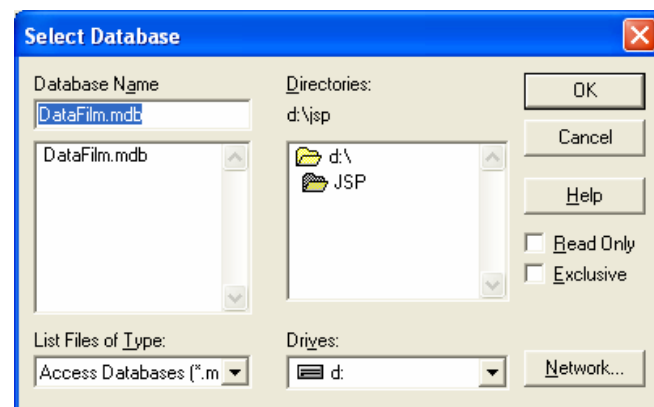


Gambar 9.8 Pilihan driver untuk data source

- c. Pada bagian **Data Source Name** ketikkan "DSFilm" sebagai nama data source tersebut dan tekan Select untuk memilih nama dan lokasi database sebagaimana ditunjukkan pada Gambar 9.9 dan 9.10.

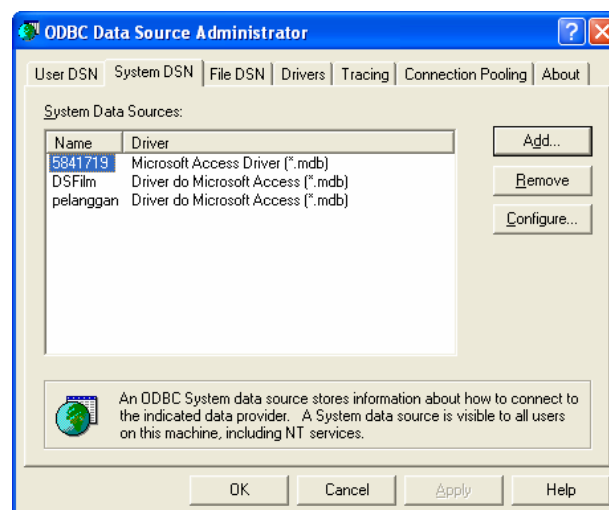


Gambar 9.9 Data source dengan nama DSFilm untuk database DataFilm



Gambar 9.10 Navigasi untuk memilih database DataFilm

- d. Bila sudah selesai maka akan keluar tampilan sebagaimana ditunjukkan pada Gambar 9.11.

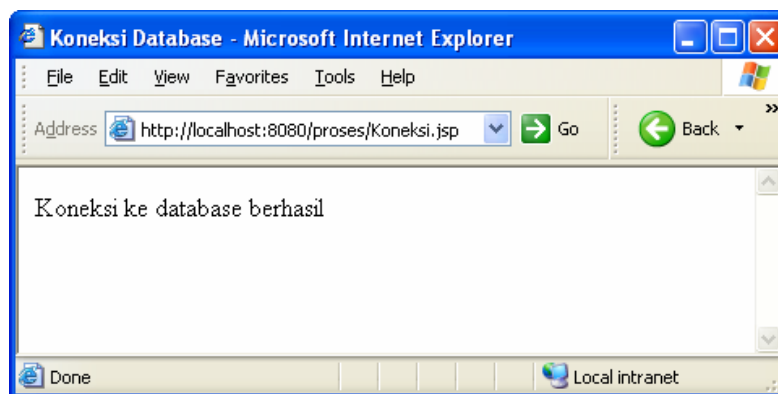


Gambar 9.11 Data source DSFilm

3. Membuat program JSP untuk membuat koneksi dengan database DataFilm seperti pada Listing 9.1. Jika koneksi berhasil maka akan tampil Gambar 9.12.

```
<%@ page import="java.sql.*" %>
<%
    Connection con=null;
    String dbname="jdbc:odbc:DSFilm";
    String status="";
    try {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con=DriverManager.getConnection(dbname,"","");
        if (con==null)
            status = "gagal";
        else
            status = "berhasil";
    }catch(ClassNotFoundException ex) {
        status = "Driver Error";
    }catch(SQLException ex) {
        status = "gagal";
    }
    con.close();
%>
<HTML>
<HEAD>
    <TITLE>Koneksi Database</TITLE>
</HEAD>
<BODY>
    Koneksi ke database <%=status%>
</BODY>
</HTML>
```

Listing 9.1 Koneksi.jsp



Gambar 9.12 Tampilan Koneksi.jsp bila berhasil koneksi

4. Buat sebuah program yang membuat tabel bernama FILM dengan truktur tabel sebagai berikut (seperti pada Listing 9.2):

| Nama kolom | Tipe data |
|------------|-------------|
| ID | VARCHAR(10) |
| JUDUL | VARCHAR(50) |
| JUMLAH | INTEGER |

Berikut ini adalah query untuk membuat tabel FILM:

```
"CREATE TABLE FILM(  
    ID VARCHAR(10),  
    JUDUL VARCHAR(10),  
    JUMLAH INTEGER  
)"
```

Karena query diatas melakukan update terhadap database maka gunakan method `executeUpdate()`. Jangan lupa menentukan database yang digunakan untuk menyimpan tabel buku terlebih dahulu! Bila berhasil membuat tabel maka muncul tampilan sebagaimana pada Gambar 9.13

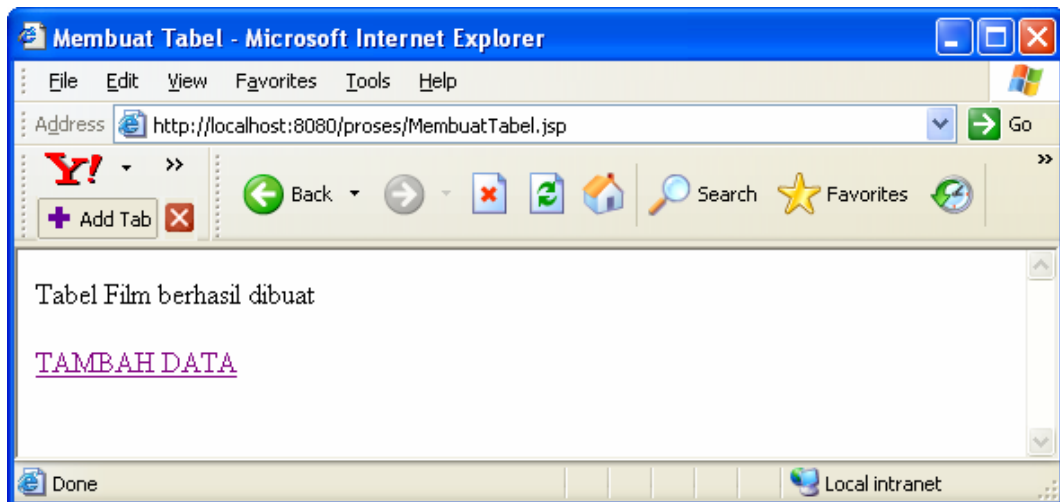
```
<%@ page import="java.sql.*" %>  
<%  
    Connection con=null;  
    String dbname="jdbc:odbc:DSFilm";  
    String status="";  
    try {  
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");  
        con=DriverManager.getConnection(dbname,"","");  
        if (con==null)  
            status = "gagal";  
        else  
            status = "berhasil";  
    }catch(ClassNotFoundException ex) {  
        status = "Driver Error";  
    }catch(SQLException ex) {  
        status = "gagal";  
    }  
  
    Statement st = con.createStatement();  
    String kueri = "CREATE TABLE FILM(ID VARCHAR(10), JUDUL  
VARCHAR(50), JUMLAH INTEGER)";  
    int hasil = st.executeUpdate(kueri);  
    st.close();  
    con.close();  
>%>
```

```

<HTML>
<HEAD>
  <TITLE>Membuat Tabel</TITLE>
</HEAD>
<BODY>
<%
  if (hasil == -1)
    out.println("Tabel Film berhasil dibuat");
  else
    out.println("Tabel Film gagal dibuat");
%>
<br>
<br>
<a href="FormMasukanData.htm">TAMBAH DATA</a>
</BODY>
</HTML>

```

Listing 9.2 MembuatTabel.jsp



Gambar 9.13 Tampilan MembuatTabel.jsp

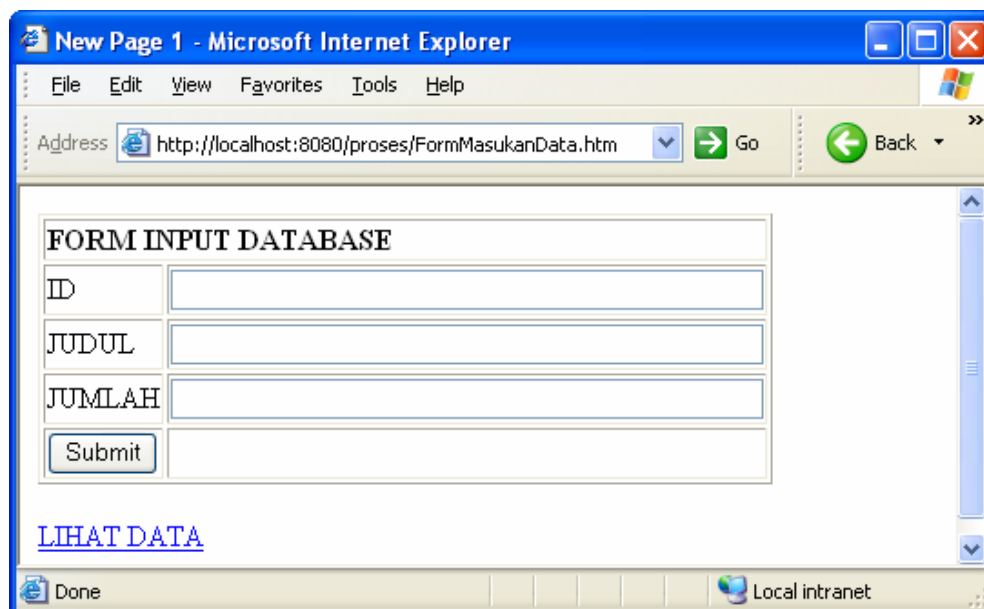
5. Memasukkan data ke tabel film sebagai berikut:

| ID | JUDUL | JUMLAH |
|-------|-----------------------|--------|
| 10001 | Tom and Jerry | 4 |
| 10002 | SpongeBob SquarePants | 3 |
| 10003 | Teletubbies | 2 |
| 20001 | The Davinci Code | 2 |
| 20002 | Casino Royale | 3 |

Untuk memasukkan data, desainlah form masukan data seperti pada Gambar 9.14.

Listing 9.3 adalah kode untuk Gambar 9.14. Listing 9.4 adalah kode yang

menangani hasil submit masukan data. Bila berhasil memasukkan data maka akan keluar tampilan seperti pada Gambar 9.15. Gambar 9.16 adalah isi tabel FILM setelah data berhasil dimasukkan.



Gambar 9.14 Tampilan FormMasukkanData.html

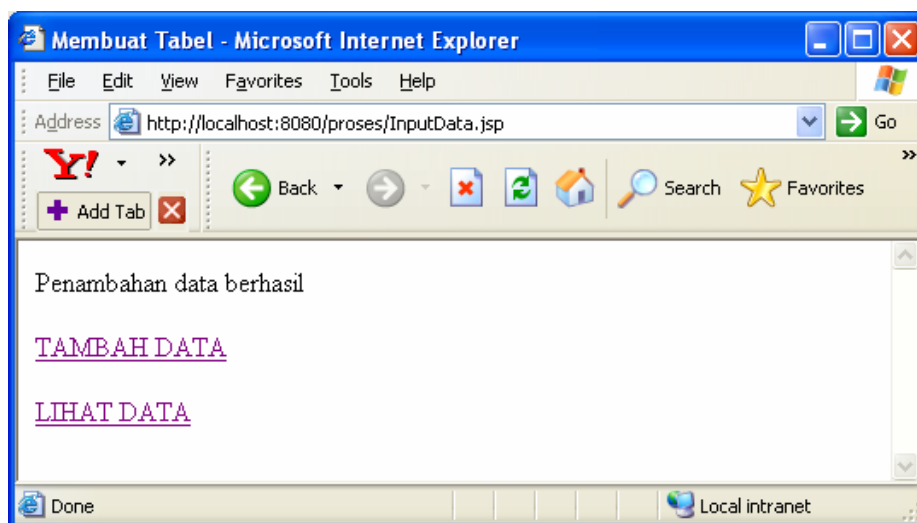
```
<html>
<head>
<title>New Page 1</title>
</head>
<body>
<form name="form1" method="post" action="InputData.jsp">
  <table width="56%" border="1">
    <tr>
      <td colspan="2"><strong>FORM INPUT
DATABASE</strong></td>
    </tr>
    <tr>
      <td width="22%">ID</td>
      <td width="78%"><input name="tf_id" type="text "
id="tf_id" size="50"></td>
    </tr>
    <tr>
      <td>JUDUL</td>
      <td><input name="tf_judul" type="text " id="tf_judul "
size="50"></td>
    </tr>
    <tr>
      <td>JUMLAH</td>
      <td><input name="tf_jumlah" type="text " id="tf_jumlah"
size="50"></td>
  </table>
</form>
<a href="#">LIHAT DATA</a>
</body>
</html>
```

```

    </tr>
    <tr>
      <td><input          type="submit"          name="Submit"
value="Submit"></td>
      <td>&nbsp;</td>
    </tr>
  </table>
  <p><a href="LihatData.jsp">LIHAT DATA</a></p>
</form>
</body>
</html>

```

Listing 9.3 FormMasukanData.htm



Gambar 9.15 Tampilan InputData.jsp jika berhasil memasukkan data

```

<%@ page import="java.sql.*" %>
<%
  /**
   * Mengambil parameter dari halaman FormMasukanData.html
   */
  String id = request.getParameter("tf_id");
  String judul = request.getParameter("tf_judul");
  String jumlah = request.getParameter("tf_jumlah");

  /**
   * Menyiapkan variabel untuk mengakses Database
   */
  Connection con=null;
  String dbname="jdbc:odbc:DSFilm";
  String status="";
  Statement st=null;

  /**
   * Melakukan koneksi ke database
   */
  try {

```



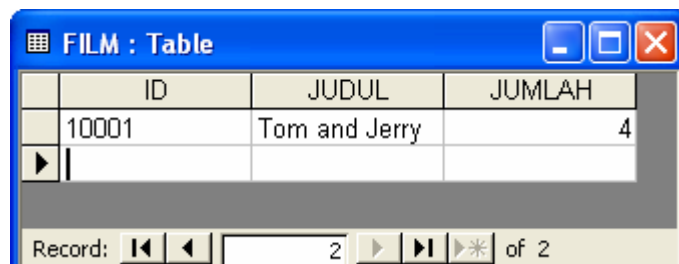
```

        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con=DriverManager.getConnection(dbname,"","");
        if (con==null)
            status = "gagal";
        else
            status = "berhasil";
    }catch(ClassNotFoundException ex) {
        status = "Driver Error";
    }catch(SQLException ex) {
        status = "gagal";
    }

    /**
     * Menyiapkan kueri
     */
    String kueri = "INSERT INTO FILM(ID, JUDUL, JUMLAH)
VALUES ('"+id+"','"+judul+"','"+jumlah+"")";
    st = con.createStatement();
    int isiTabel = st.executeUpdate(kueri);
    st.close();
    con.close();
%>
<HTML>
<HEAD>
    <TITLE>Membuat Tabel</TITLE>
</HEAD>
<BODY>
<%
    if (isiTabel == 1)
        out.println("Penambahan data berhasil");
    else
        out.println("Penambahan data gagal");
%>
<br>
<br>
<a href="FormMasukanData.htm"> TAMBAH DATA </a>
<br>
<br>
<a href="LihatData.jsp">LIHAT DATA </a>
</BODY>
</HTML>

```

Lising 9.4 InputData.jsp



| ID | JUDUL | JUMLAH |
|-------|---------------|--------|
| 10001 | Tom and Jerry | 4 |

Record: 2 of 2

Gambar 9.16 Isi tabel FILM setelah 1 record data dimasukkan

Masukkan semua data yang ada di tabel dengan menggunakan form masukan data yang telah dibuat. Untuk kode melihat data (`LihatData.jsp`), mengedit data, dan menghapus data akan dibahas pada pertemuan selanjutnya.

9.6 Soal Latihan

1. Apa yang dimaksud dengan JDBC?
2. Apa yang dimaksud dengan driver?
3. Sebutkan dan jelaskan langkah-langkah dasar penggunaan JDBC?
4. Apa hubungan antara JSP dengan JDBC?
5. Apa hubungan antara SQL dengan JDBC?
6. Bagaimana bahasa SQL untuk proses pembuatan tabel dan memasukkan data ke tabel?
7. Buat sebuah program yang membuat tabel bernama BUKU dengan truktur tabel sebagai berikut:

| Nama kolom | Tipe data |
|-------------|-----------|
| JUDUL_BUKU | VARCHAR |
| ID_PENERBIT | INTEGER |
| HARGA | REAL |
| PENJUALAN | INTEGER |

Berikut ini adalah query untuk membuat tabel BUKU:

```
CREATE TABLE BUKU(  
    JUDUL_BUKU VARCHAR(40),  
    ID_PENERBIT INTEGER,  
    HARGA REAL,  
    PENJUALAN INTEGER)
```

Karena query diatas melakukan update terhadap database maka gunakan method `executeUpdate()`. Jangan lupa membuat atau menentukan database yang

digunakan untuk menyimpan tabel buku serta membuat data source database tersebut terlebih dahulu!

8. Buat program yang digunakan untuk memasukkan data tanpa menggunakan form dengan data sebagai berikut:

| NAMA_BUKU | ID_PENERBIT | HARGA | PENJUALAN |
|----------------|-------------|-------|-----------|
| PemrogramanWeb | 111 | 27000 | 14 |

Gunakan template kueri sebagai berikut:

```
INSERT INTO BUKU(NAMA_BUKU, ID_PENERBIT, HARGA,  
PENJUALAN) VALUES('nama_buku', 'penerbit', 'harga', penjualan);
```

9. Setelah membuat tabel pada nomor 1 diatas. Buatlah program yang digunakan untuk mengisi tabel BUKU tersebut dengan menggunakan form masukan data dengan data masukan sebagai berikut:

| NAMA_BUKU | ID_PENERBIT | HARGA | PENJUALAN |
|------------------------|-------------|-------|-----------|
| E-Learning | 101 | 25000 | 10 |
| Jaringan Tanpa Hardisk | 203 | 23000 | 8 |
| Oracle9i | 311 | 54000 | 20 |
| Pemrograman Java | 101 | 49000 | 35 |