Universitas Indraprasta PGRI

Koneksi Java Database (Part 1)

Pemrograman 3

JDBC

- JDBC adalah Application Programming Interface (API) yang menyediakan fungsi-fungsi dasar untuk akses data. JDBC API terdiri atas sejumlah class dan interface yang dapat digunakan untuk menulis aplikasi database dengan menggunakan Java. Class dan Interface JDBC terdapat pada package java.sql. JDBC memungkinkan kita untuk membuat aplikasi Java yang melakukan tiga hal: koneksi ke sumber data, mengirimkan query dan statement ke database, menerima dan mengolah resultset yang diperoleh dari database.

Database Driver

- JDBC memerlukan database driver untuk melakukan koneksi ke suatu sumber data. Database driver ini bersifat spesifik untuk setiap jenis sumber data. Untuk aplikasi Java, driver database disebut dengan JDBC Driver. JDBC Driver adalah software library yang diperlukan agar program JDBC dapat berkomunikasi dengan database tertentu.

Membuat Koneksi

Melakukan koneksi ke database melibatkan dua langkah: Meload driver dan membuat koneksi itu sendiri. Cara meload driver sangat mudah, pertama letakkan file jar database driver ke dalam classpath. Kemudian load driver dengan menambahkan kode berikut ini:

Class.forName(“org.sqlite.JDBC”);

Nama class database driver untuk setiap DBMS berbeda, class tersebut bisa ditemukan dalam dokumentasi driver database yang digunakan. Dalam contoh ini, nama class database driver dari sqlite3 yaitu org.sqlite.JDBC.

Memanggil method Class.forName secara otomatis membuat instance dari database driver, class DriverManager secara otomatis juga dipanggil untuk mengelola class database driver ini. Jadi tidak perlu menggunakan statement new untuk membuat instance dari class database driver tersebut.

Langkah berikutnya adalah membuat koneksi ke database menggunakan database driver yang sudah diload sebelumnya. Class DriverManager bekerja sama dengan interface Driver untuk mengelola driver-driver yang diload oleh aplikasi. Method DriverManager.getConnection bertugas untuk membuat koneksi:

Connection conn = DriverManager.getConnection("JDBC:sqlite:latihan.db");

Setelah sukses melakukan koneksi ke database, dapat dilakukan pengambilan data dari database menggunakan perintah query ataupun melakukan perubahan terhadap database.

Mengambil dan Memanipulasi Data dari Database

Proses pengambilan data dari database memerlukan suatu class untuk menampung data yang berhasil diambil, class tersebut harus mengimplement interface ResultSet. Instance dari object bertipe ResultSet diperlukan untuk menampung hasil kembalian data dari database. Sebelum bisa memperoleh instance dari ResultSet, harus dibuat instance dari class Statement. Class Statement mempunyai method executeQuery yang digunakan untuk menjalankan perintah query dalam database kemudian mengembalikan data hasil eksekusi query ke dalam object ResultSet.

Statement stat=conn.createStatement();

Berikut ini adalah contoh kode untuk membuat instance class Statement, kemudian menjalankan query untuk mengambil data dari database yang hasilnya dipegang oleh ResultSet :

ResulSet rs = statement.executeQuery(“select * from Customers”);

ResultSet akan meletakkan kursornya (posisi pembacaan baris) di sebuah posisi sebelum baris pertama. Untuk menggerakkan kursor maju, mundur, ke suatu posisi relatif atau ke suatu posisi absolute tertentu

Contoh Program

```
import java.sql.*;
class ContohKoneksiJava{
    public static void main (String []args) throws Exception{
        Class.forName("org.sqlite.JDBC");
        Connection con = DriverManager.getConnection  ("jdbc:sqlite:coba.db");
        Statement st=con.createStatement();

        st.executeUpdate("drop table if exists mhs;");
        st.executeUpdate ("create table mhs (npm integer ,nama varchar(20));");
        st.executeUpdate("insert into mhs values (567,'dian');");
        st.executeUpdate("insert into mhs values (678,'aryo');");
        st.executeUpdate("insert into mhs values (789,'linda');");
        st.executeUpdate("insert into mhs values (890,'heni');");

        st.executeUpdate("update mhs set nama='haya' where  npm=235;");
        st.executeUpdate("update mhs set nama='ario' where nama='aryo';");

        st.executeUpdate("delete from mhs where nama='linda';");

        ResultSet rs=st.executeQuery("select * from mhs;");
        System.out.println ("NPM \t Nama");
        while (rs.next()){
            System.out.print(rs.getInt("npm"));
            System.out.println("\t "+rs.getString("nama"));
        }
    }
}
```

Contoh Program 2

- Berikut tabel barang yang ada pada database kampus

kode	namabar	harga
B11	Buku	5000
B12	Pensil	3500
B13	Kemeja	50000
B14	Payung	45000

```
import java.sql.*;
public class DBTest1 {
    public static void main(String[] args){
        try {
            Class.forName("org.sqlite.JDBC");
            Connection koneksi = DriverManager.getConnection("jdbc:sqlite:D:/Data/kampus.db");
            Statement stat = koneksi.createStatement();
            ResultSet set = stat.executeQuery("select * from barang");
            while(set.next()){
                String a = set.getString("kode");
                    System.out.print(a);
                String b=set.getString("namabar");
                    System.out.print("\t"+b);
                int c=set.getInt("harga");
                    System.out.print("\t"+c);
                System.out.println();
            }
         }
         catch (Exception e)
         {   System.err.println("Error : "+e.getMessage()); }
     }
}
```

- Output Program :