PRAKTIKUM 8. MENGAKSES DATABASE MYSQL

Tujuan: Mahasiswa dapat membuat program Java untuk mengakses database MySQL dan melakukan berbagai operasi menggunakan Structure Query Language (SQL).

Koneksi ke Database MySQL dengan Java

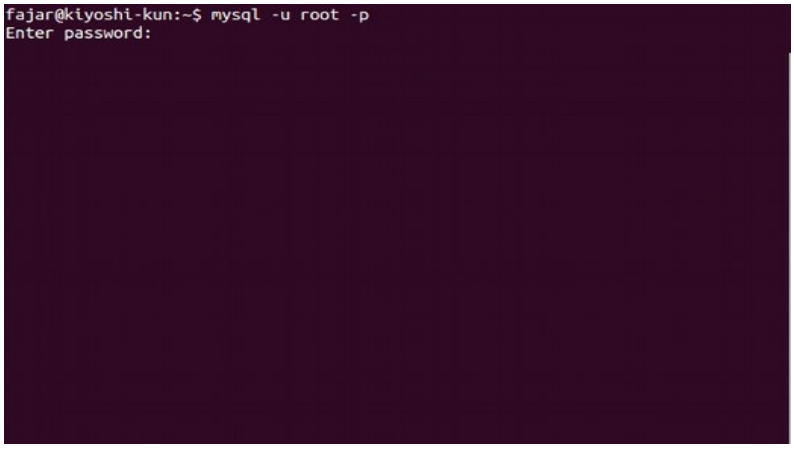
Agar manajemen data lebih mudah dan dapat diandalkan, maka pemakaian aplikasi database merupakan salah satu solusinya. Menggunakan aplikasi database membuat pemrogram Java lebih leluasa berkonsentrasi pada lapisan logik aplikasi. Antarmuka untuk mengakses berbagai aplikasi database dari Java yaitu melalui *Java Database Connectivity (JDBC)*. Melalui JDBC pemrogram dapat melakukan berbagai operasi menggunakan *Structure Query Language* (SQL) seperti membuat sebuah koneksi ke database, melakukan query, update data, dan menerima hasilnya untuk selanjutnya diolah dalam program Java. Untuk menggunakan JDBC, pemrogram membutuhkan implementasi spesifik (driver) dari setiap database yang akan digunakan. Pada tulisan ini, akan disajikan langkah-langkah untuk mengakses database MySQL melalui Java.

1. Menginstal dan mengkonfigurasi MySQL

Langkah pertama adalah menginstal dan mengkonfigurasi database MySQL di sistem anda. Untuk menginstal dan menggunakan MySQL silahkan merujuk ke
https://dev.mysql.com/doc/refman/5.0/en/tutorial.html

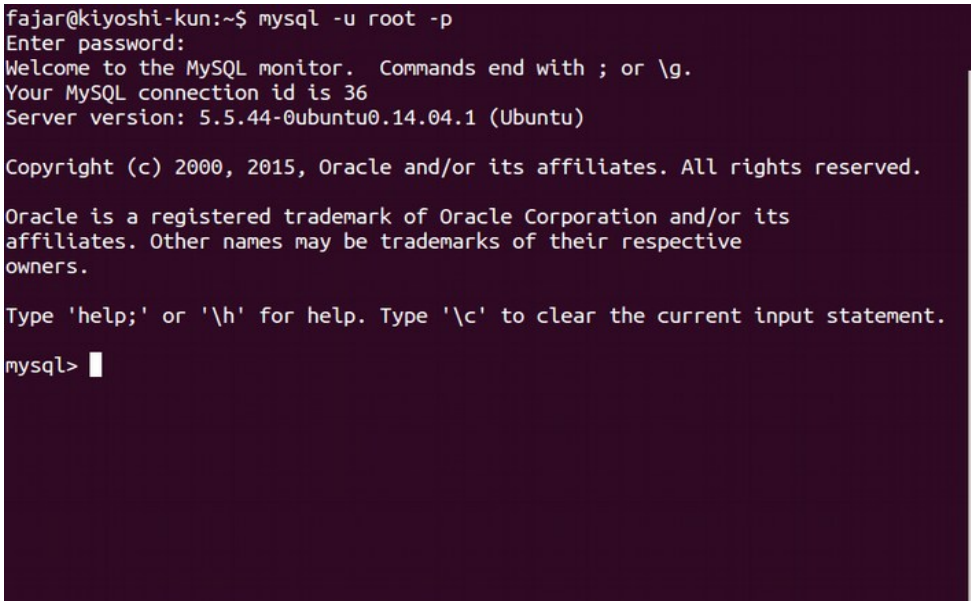
2. Uji coba koneksi ke MySQL Server

Setelah MySQL terinstal dan terkonfigurasi (termasuk pembuatan akun), pastikan sever MySQL disistem anda telah dijalankan. Selanjutnya, lakukan uji coba pengaksesan server MySQL melalui command prompt berikut : mysql -u root -p



Gambar 1. Melakukan koneksi ke server MySQL

Masukkan password anda, dan jika koneksi ke server MySQL sukses akan ditampikan notifikasi seperti pada Gambar 2.



Gambar 2. Prompt MySQL Client, setelah koneksi ke server MySQL

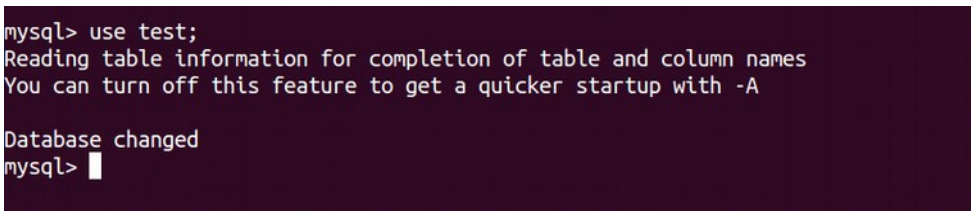
3. Aktifkan database test dengan perintah:

use test;

tetapi jika database test belum ada, maka buatlah terlebih dahulu dengan perintah:

create database test;

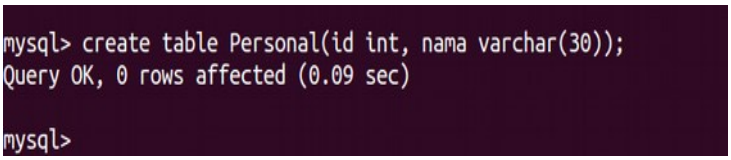
setelah selesai dibuat aktifkan dengan perintah use test;



Gambar 3. Mengaktifkan database test

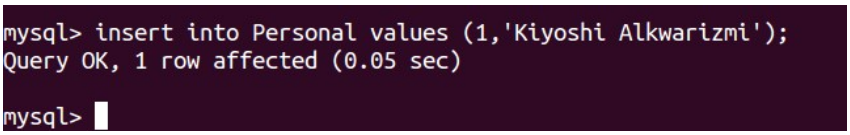
Selanjutnya, sebagai contoh mari kita membuat tabel dalam database test dengan nama **Personal** yang berisi field (kolom) **id** bertipe integer, dan **nama** bertipe string (dapat menggunakan tipe **variable char**).

create table tes(**id** int, **nama** varchar(30));



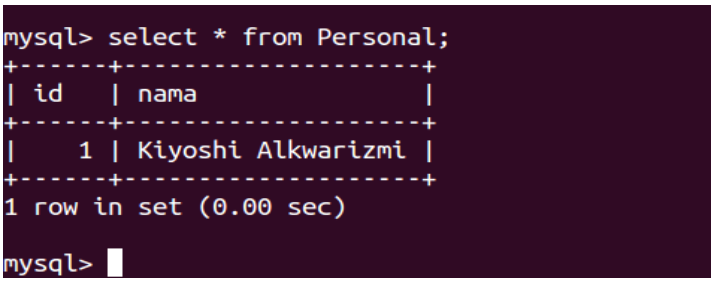
Gambar 4. Membuat tabel Personal

Lakukan pengisian data ke tabel Personal menggunakan perintah **insert** seperti gambar 5 berikut :



Gambar 5. Mengisi satu data (record) ke tabel Personal

Tampilkan data dari tabel Personal menggunakan perintah select seperti gambar 6 berikut:



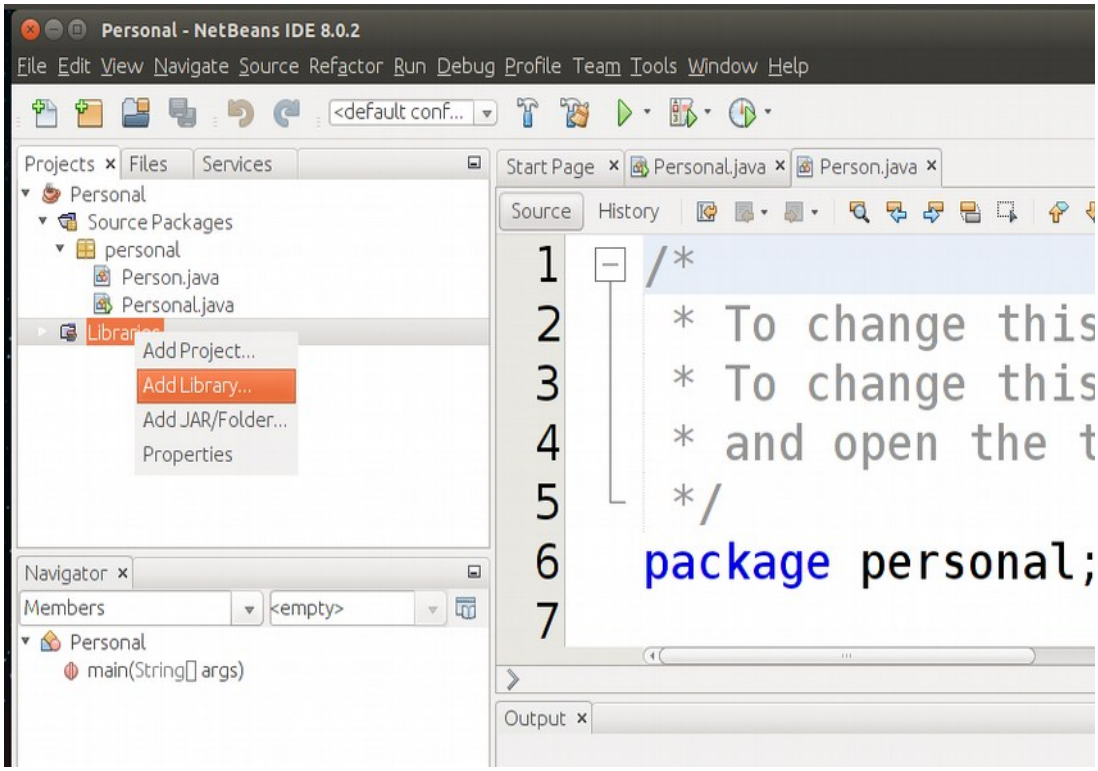
Gambar 6. Menampilkan seluruh data tabel Personal

Lebih jauh tentang perintah-perintah SQL silahkan merujuk ke tutorial MySQL, salah satunya di URL berikut: https://dev.mysql.com/doc/refman/5.0/en/tutorial.html

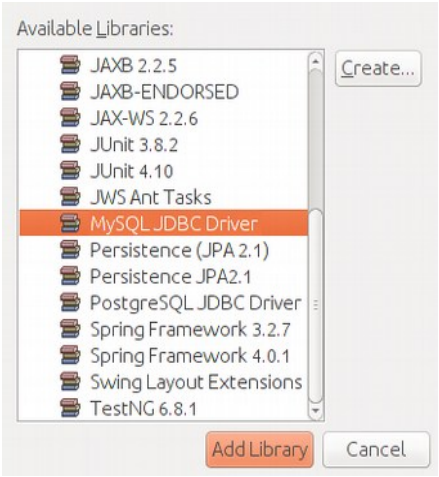
Untuk mempermudah instalasi, konfigurasi dan manajemen MySQL, dapat digunakan berbagai tools bantu pengembangan seperti XAMPP.

4. Mempersiapkan koneksi Aplikasi Java ke server MySQL

Selanjutnya, menyiapkan driver JDBC MySQL sehingga program Java kita dapat berkomunikasi dengan server MySQL yang telah terinstal dan diuji pada langkah sebelumnya. Driver tersebut dapat ditambahkan melalui fasilitas **Add Library** yang disediakan Netbeans. Gambar 7 dan Gambar 8 memperlihatkan proses penambahan library (driver MySQL) ke proyek kita.

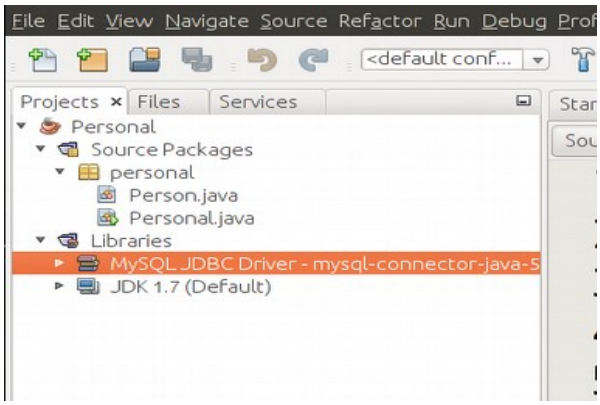


Gambar 7. Menambahkan library ke proyek



Gambar 8. Menambahkan library MySQL JDBC driver ke proyek

Pastikan driver JDBC MySQL telah ditambahkan sebagai library di proyek kita, seperti terlihat pada Gambar 9.



Gambar 9. Driver MySQL Connector sudah ditambahkan ke proyek Personal

5. Menyusun kode Java untuk berkomunikasi dengan server MySQL

Selanjutnya, menyiapkan kode program Java untuk berkomunikasi dengan server MySQL yang dimulai dengan **melakukan koneksi**, **mempersiapkan operasi atau query**, **menjalankan operasi atau query**, **menerima hasilnya** untuk selanjutnya **melakukan pemrosesan data lebih lanjut**. Contoh berikut memperlihatkan satu metode dalam class OperasiMySQL yang digunakan untuk menulis data ke database MySQL. Metode ini membutuhkan parameter berupa objek Person. (Modifikasilah sesuai dengan aplikasi yang anda kembangkan).

```
public class OperasiMySQL{
        Connection conn = null;
        String url = "jdbc:mysql://localhost/";
        String dbName = "test";                 //sesuaikan dengan database yg anda gunakan
        String driver = "com.mysql.jdbc.Driver";
        String userName = "root";               //sesuaikan dengan user MySQL anda
        String password = "mypassword";         //sesuaikan dengan password anda
        private Statement statement = null;
        private PreparedStatement preparedStatement = null;
        private ResultSet resultSet = null;

        public void insertData(Person p){
                try {
                        Class.forName(driver).newInstance();
                        conn = DriverManager.getConnection(url+dbName,userName,password);
                        statement = conn.createStatement();
```

```
                    preparedStatement = conn.prepareStatement("insert into Personal values (?,?)");
                    preparedStatement.setInt(1, p.getId());
                    preparedStatement.setString(2, p.getNama());
                    preparedStatement.executeUpdate();
                    conn.close();
        } catch (Exception e) {
            System.out.println("NO CONNECTION =(");
        }

    }
}
```

Berikut contoh potongan kode pemakaian class OperasiMySQL dan metode insertData() dengan skenario Ketika tombol Save (jButton1) pada form diklik maka data dibaca dan objek person diisi dengan data dari form tersebut, selanjutnya operasi insertData() dilakukan untuk menyimpan data ke tabel di MySQL.

```
public class FormPerson extends javax.swing.JFrame {
        ……................
        private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
                Person p=new Person();
                p.setId(Integer.parseInt(jTextField1.getText()));   //konversi data
                p.setNama(jTextField2.getText());
                new DataLayer().insertdata(p);
        }
        ……..................
    }
```

Silahkan tambahkan dengan metode-metode lain sesuai kebutuhan proyek anda seperti updateData(), deleteData(), readData(), dll.

Tugas Praktikum:

Buatlah aplikasi penginputan data yang menyimpan data inputan dari form ke database MySQL.