Aturan Praktikum:

- Keterlambatan maksimal 10 menit
- Hasil praktikum, selesai atau tidak selesai ditanggung mahasiswa
- Nilai praktikum adalah nilai perorangan, dan digunakan sebagai nilai tambahan jika nilai quiz atau tes kurang mencukupi
- Mahasiswa login ke komputer
- Persiapan dan penjelasan praktikum 10 menit
- Praktikum dilakukan selama 70 menit terdiri dari praktikum sesuai model dan sebuah soal praktikum
- Hasil praktikum di-zip dan dikumpulkan
- Sisa waktu 20 menit digunakan untuk tes kecil hasil praktikum, saat tes tampilan komputer adalah background desktop dan mahasiswa tidak diperkenankan menggunakan komputer dan melihat modul praktikum
- Mahasiswa logout dari komputer
- Praktikum selesai

Java Database Connectivity (JDBC)
(Minggu 5 – Praktikum II)

1. Sekilas JDBC

JDBC API adalah API java untuk mengakses basis data. JDBC API berisi kumpulan kelas yang ditulis dengan bahasa pemrograman Java yang menyediakan API standar untuk mengakses basis data. JDBC memungkinkan *programer* melakukan hal-hal berikut:

1. membuat koneksi dengan sebuah sumber data
2. mengirimkan query dan mengubah isi sumber data
3. memproses hasil query

2. JDBC API vs ODBC

Microsoft membuat ODBC (*Open Database Connectivity*) sebagai antarmuka pemrograman untuk mengakses basis data relasional. ODBC memiliki kemampuan untuk mengakses hampir semua basis data yang ada hampir di semua *platform*. ODBC sudah banyak digunakan ketika JDBC sedang dikembangkan, lalu pertanyaannya, kenapa Java tidak memakai ODBC saja? ODBC dapat digunakan untuk membuat program Java yang mengakses basis data, tapi akan menjadi lebih baik jika penggunaan ODBC juga melalui JDBC yang biasa disebut sebagai JDBC-ODBC *Bridge* (terpaket pada JDK untuk Windows dan Solaris) hal ini dikarenakan beberapa alasan berikut:

1. ODBC kurang sesuai untuk penggunaan secara langsung oleh bahasa pemrograman Java karena ODBC menggunakan bahasa C. Pemanggilan antarmuka C oleh Java dapat menyebabkan masalah keamanan dan kesesuaian pada aplikasi

2. Banyak terdapat redudansi objek, karena dari Java, ODBC akan menyalin objek Java dengan cara C.
3. ODBC lebih kompleks dalam penggunaanya karena fungsi-fungsi sederhana dan kompleks dipaket menjadi satu.
4. JDBC telah dipaket dengan JDK, sedangkan ODBC merupakan *installer* tersendiri

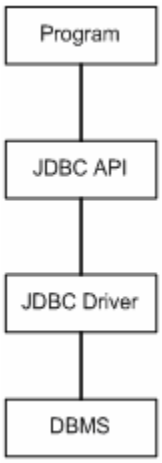
JDBC API merupakan antarmuka Java murni untuk berkerja dengan SQL. Microsoft juga mempublikasikan API selain ODBC seperti OLE DB, ADO (*ActiveX Data Object*), dan ADO.NET yang juga dapat mengeksekusi query SQL.

Hal-hal yang bisa dilakukan dengan menggunakan JDBC API adalah sebagai berikut:

- membuat tabel
- memasukkan nilai *record* ke tabel
- query untuk mengakses tabel
- mendapatkan hasil query
- mengubah tabel

3. JDBC Driver

JDBC *Driver* mengimplementasikan antarmuka yang didefinisikan dalam JDBC API untuk berinteraksi dengan server basis data. JDBC *Driver* digunakan untuk membuka koneksi dan berinteraksi dengan basis data dengan mengirimkan query SQL kemudian menerima hasil query dengan Java. JDBC *Driver* biasanya dapat didapatkan dari *vendor* basis data. Berikut gambat keterhubungan JDBC API dengan JDBC *Driver*:



4. Praktikum

a. Persiapan

- Membuat direktori kerja dengan nama SI319-P5-2-Kelas-NIM misalnya SI319-P5-2-A-23507024
- Di dalam direktori di atas, buat direktori JDBC untuk menyimpan file-file yang akan dibuat

b. MySQL Connector

1. Cari file mysql-connector-java-5.1.2-beta-bin.jar pada komputer yang Anda pakai
2. Masukkan mysql-connector-java-5.1.2-beta-bin.jar pada direktori instal_jdk/jre/lib/ext dan pada jre/lib/ext
   (cara ini digunakan untuk menambah *library* atau pustaka pada Java)

c. Membuat Basis Data Pada MySQL

1. Buka command prompt, masuk ke direktori bin pada MySQL, ketik perintah:

   ```
   mysql –uroot
   ```

   jika perintah tidak dikenali maka start MySQL dengan mengetik perintah:

   ```
   mysqld
   ```

2. pada command prompt (di direktori bin MySQL), kemudian ketik perintah sebelumnya sampai muncul tampilan berikut:

```
C:\xampp\mysql\bin>mysql -uroot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.0.37 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

3. Setiap query SQL yang berhasil dieksekusi , maka akan keluar tulisan seperti contoh berikut:

```
mysql>     INSERT     INTO     mata_kuliah(kode,     nama,     semester)
VALUES('IF3281','Pemrograman Berorientasi Objek', 3);
Query OK, 1 row affected (0.00 sec)
```

Yang penting adalah hasil: `Query OK, 1 row affected`

Ketik query SQL berikut pada command prompt di atas.

```
CREATE DATABASE IF NOT EXISTS Kuliah;
```

```
USE Kuliah;
```

```
CREATE TABLE  mahasiswa(
nim VARCHAR(10) NOT NULL,
nama VARCHAR(255) NOT NULL,
tanggal_lahir  DATE NOT NULL,
alamat VARCHAR(255) NULL,
PRIMARY KEY (nim)
);
```

```
CREATE TABLE  mata_kuliah(
kode VARCHAR(10) NOT NULL,
nama VARCHAR(255) NOT NULL,
semester INTEGER(2) NOT NULL,
PRIMARY KEY (kode)
);
```

```
CREATE TABLE  nilai(
kode VARCHAR(10) NOT NULL,
nim VARCHAR(10) NOT NULL,
nilai DECIMAL(5,2) NULL,
alphabet VARCHAR(2) NULL,
PRIMARY KEY (kode, nim),
FOREIGN KEY (nim) REFERENCES mahasiswa,
FOREIGN KEY (kode) REFERENCES mata_kuliah
);
```

```
INSERT       INTO       mahasiswa(nim,       nama,       tanggal_lahir)
VALUES('13507701','Nana','1960-01-01');
```

```
INSERT    INTO    mahasiswa(nim,    nama,    tanggal_lahir,    alamat)
VALUES('13507702','Rudi','1959-12-31','Bandung');
```

```
INSERT    INTO    mahasiswa(nim,    nama,    tanggal_lahir,    alamat)
VALUES('13507703','Dea','1959-07-07','Bandung');
```

```
INSERT    INTO    mahasiswa(nim,    nama,    tanggal_lahir,    alamat)
VALUES('13507704','Ihsan','1958-11-08','Surabaya');
```

```
INSERT    INTO    mahasiswa(nim,    nama,    tanggal_lahir,    alamat)
VALUES('13507705','Tiara','1959-11-04','Yogyakarta');
```

```
INSERT       INTO       mata_kuliah(kode,       nama,       semester)
VALUES('IF1281','Algoritma dan Pemrograman', 1);
```

```
INSERT INTO mata_kuliah(kode, nama, semester) VALUES('IF2281','Struktur
Data', 2);
```

```
INSERT INTO mata_kuliah(kode, nama, semester) VALUES('IF3281','Pemrograman Berorientasi Objek', 3);
INSERT INTO mata_kuliah(kode, nama, semester) VALUES('IF3111','Basis Data', 3);
INSERT INTO nilai(kode, nim, nilai, alphabet) VALUES('IF1281','13507701', 64.75, 'C');
INSERT INTO nilai(kode, nim, nilai, alphabet) VALUES('IF2281','13507702', 75.11, 'B');
INSERT INTO nilai(kode, nim, nilai, alphabet) VALUES('IF3181','13507703', 84.63, 'A');
INSERT INTO nilai(kode, nim, nilai, alphabet) VALUES('IF3111','13507704', 77.07, 'B');
```

4. Ketik perintah:

   ```
   \q
   ```

   Untuk keluar dari MySQL

d. Penggunaan JDBC pada program

1. Buat file berikut:

Nama file: DB.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

class DB {
 private Statement stmt = null; // koneksi query
 private ResultSet rs = null; // hasil query
 private Connection conn = null; // koneksi MySQL dan basis data

 public DB(String ConAddress) throws Exception, SQLException {
  /**
   * Method DB
   * Konstruktor : melakukan koneksi ke MySQL dan basis data
   * Menerima masukan berupa string alamat koneksi ke MySQL dan basis data
   */
   try {
      // membuat/meregistrasi driver MySQL
      Class.forName("com.mysql.jdbc.Driver").newInstance();
      // membuat koneksi MySQL dan basis data
```

```
      conn = DriverManager.getConnection(ConAddress);
      conn.setTransactionIsolation(conn.TRANSACTION_READ_UNCOMMITTED);
    }
    catch(SQLException es) {
      // mengeluarkan pesan error jika koneksi gagal
      throw es;
    }
}

public void createQuery(String Query)throws Exception, SQLException {
 /**
  * Method createQuery
  * Mengeksekusi query tanpa mengubah isi data
  * Menerima masukan berupa string query
  */
  try {
     stmt = conn.createStatement();
     // eksekusi query
     rs = stmt.executeQuery(Query);
     if (stmt.execute(Query)) {
       // ambil hasil query
       rs = stmt.getResultSet();
     }
  }
  catch(SQLException es) {
     // eksepsi jika query gagal dieksekusi
     throw es;
  }
}

public void createUpdate(String Query)throws Exception, SQLException {
 /**
  * Method createQuery
  * Mengeksekusi query yang mengubah isi data (update, insert, delete)
  * Menerima masukan berupa string query
  */
  try {
    stmt = conn.createStatement();
    // eksekusi query
    int hasil = stmt.executeUpdate(Query);
   }
   catch(SQLException es) {
      // eksepsi jika query gagal dieksekusi
      throw es;
   }
}

public ResultSet getResult()throws Exception {
 /**
  * Method getResult
  * Memberikan hasil query
  */
  ResultSet Temp = null;
  try{
    return rs;
  }
  catch (Exception ex) {
```

```
      // eksepsi jika hasil tidak dapat dikembalikan
      return Temp;
    }
  }

  public void closeResult()throws SQLException, Exception {
   /**
    * Method closeResult
    * Menutup hubungan dari eksekusi query
    */
    if (rs != null) {
      try {
         rs.close();
      }
      catch (SQLException sqlEx) {
         rs = null;
         throw sqlEx;
      }
    }
    if (stmt != null) {
      try {
         stmt.close();
      }
      catch (SQLException sqlEx) {
         stmt = null;
         throw sqlEx;
      }
    }
  }

  public void closeConnection()throws SQLException, Exception {
   /**
    * Method closeConnection
    * Menutup hubungan dengan MySQL dan basis data
    */
    if (conn != null) {
       try {
         conn.close();
       }
       catch(SQLException sqlEx) {
         conn = null;
       }
     }
   }
}

class CobaBDAkses{
  public static void main(String[] args) {
    // isi dengan nim pada kode di bawa ini, misal “Owner: 1206001”
    System.out.println("Owner: [nim]");
    try{
     /* pada kode berikut, user diisi dengan root, password diisi
dengan password mysql jika ada, jika tidak ada bisa dikosongkan */
     DB                  db                    =                    new
DB("jdbc:mysql://localhost:3306/kuliah?user=root&password=");
      db.createQuery("SELECT * FROM MAHASISWA");
      try{
```

```
          int i = 1;
          while(db.getResult().next()){
        // ambil hasil query
        String nim = db.getResult().getString(1);
        String nama = db.getResult().getString(2);
        String tl = db.getResult().getString(3);
        String alamat = db.getResult().getString(4);

            System.out.println("----------------------");
        System.out.println("record ke : " + i);
        System.out.println("nim : " + nim);
        System.out.println("nama : " + nama);
        System.out.println("tanggal lahir : " + tl);
            if(alamat != null){
          System.out.println("alamat : " + alamat);
        }
        else{
          System.out.println("alamat : null");
        }
        System.out.println("----------------------");
        i = i + 1;
          }
        }
        catch(Exception e){
          e.printStackTrace();
        }
        db.closeResult();
        db.closeConnection();
      }
      catch(Exception e){
        e.printStackTrace();
      }
  }
}
```

Pada kode di atas url koneksi jdbc:mysql://localhost:3306/kuliah?user=root&password= merupakan hasil dari jdbc:mysql://<host>:<port>/<nama_basis_data>?user=<user>&password=<password>.

2. Lakukan kompilasi dengan perintah:

   ```
   javac DB.java
   ```

   (
   jika javac tidak dikenali maka set path dengan perintah:

   ```
   path=c:/jdk/bin;%path%
   ```

)

3. lalu eksekusi program dengan perintah:

```
java –cp . CobaBDAkses
```