Pemrograman Jaringan 8

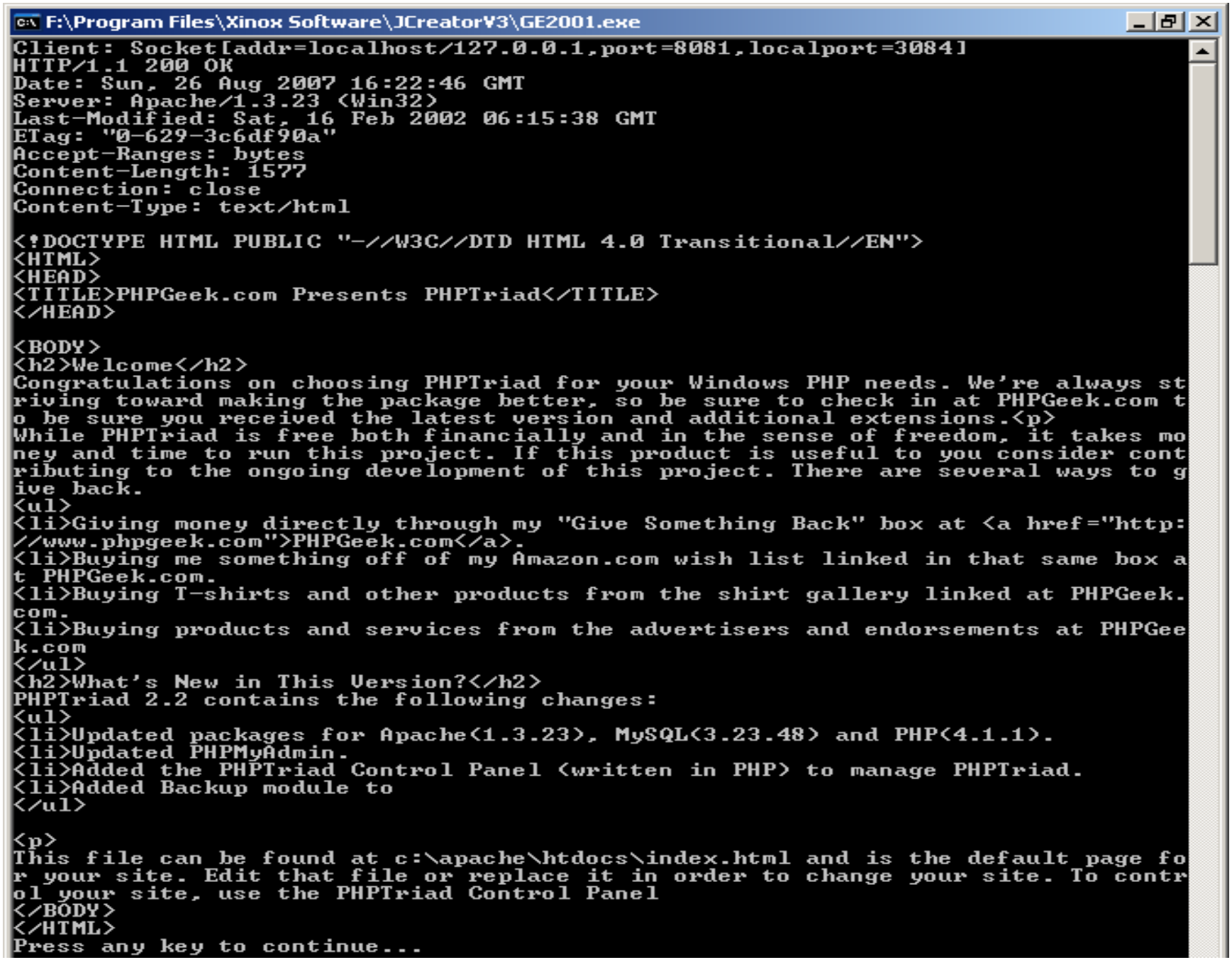
anton@ukdw.ac.id

Outline

- HTTP Client
- Socket Oriented Multithreading
- Sisipan: Manipulasi JAR file
- JDBC

HTTP Socket Client

- HTTP Client yang akan menggunakan perintah HTTP untuk mengambil dokumen yang ada melalui protokol HTTP
- Contoh: ExHTTPClient.java
- Kembangkan untuk method POST


F:\Program Files\Xinox Software\JCreatorV3\GE2001.exe
Client: Socket[addr=localhost/127.0.0.1,port=8081,localport=3084]
HTTP/1.1 200 OK
Date: Sun, 26 Aug 2007 16:22:46 GMT
Server: Apache/1.3.23 (Win32)
Last-Modified: Sat, 16 Feb 2002 06:15:38 GMT
ETag: "0-629-3c6df90a"
Accept-Ranges: bytes
Content-Length: 1577
Connection: close
Content-Type: text/html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>PHPGeek.com Presents PHPTriad</TITLE>
</HEAD>
<BODY>
<h2>Welcome</h2>
Congratulations on choosing PHPTriad for your Windows PHP needs. We're always st
riving toward making the package better, so be sure to check in at PHPGeek.com t
o be sure you received the latest version and additional extensions.<p>
While PHPTriad is free both financially and in the sense of freedom, it takes mo
ney and time to run this project. If this product is useful to you consider cont
ributing to the ongoing development of this project. There are several ways to g
ive back.

Giving money directly through my "Give Something Back" box at <a href="http:
//www.phpgeek.com">PHPGeek.com.
Buying me something off of my Amazon.com wish list linked in that same box a
t PHPGeek.com.
Buying T-shirts and other products from the shirt gallery linked at PHPGeek.
com.
Buying products and services from the advertisers and endorsements at PHPGee
k.com

<h2>What's New in This Version?</h2>
PHPTriad 2.2 contains the following changes:

Updated packages for Apache(1.3.23), MySQL(3.23.48) and PHP(4.1.1).
Updated PHPMyAdmin.
Added the PHPTriad Control Panel (written in PHP) to manage PHPTriad.
Added Backup module to

<p>
This file can be found at c:\apache\htdocs\index.html and is the default page fo
r your site. Edit that file or replace it in order to change your site. To contr
ol your site, use the PHPTriad Control Panel
</BODY>
</HTML>
Press any key to continue...

Socket Oriented Multithread

- Thread dibuat per koneksi,
- Thread dibuat saat client terhubung dan akan dibuang pada saat client memutuskan hubungan.
- Client dapat melakukan permintaan sebanyak mungkin pada Thread tersebut.
- Contoh: InfoServerThread.java
- Pada program utama, akan dilakukan perulangan secara terus menerus untuk memanggil sebuah method accept().
- Ketika ada koneksi client terbentuk, maka Server akan membentuk sebuah Thread dan mengirimkan socket koneksinya sebagai parameter.

Sisipan: JAR File

- File JAR adalah file executable dari class – class Java.
- File JAR berfungsi membungkus satu atau lebih file-file class berserta informasinya menjadi sebuah file archive.
- File JAR dikompresi dengan metode ZIP sehingga kita dapat membukanya dengan program-program kompresi seperti WinZip atau WinRAR.
- Dalam Java juga disediakan program utility tools JAR yang berfungsi untuk memanipulasi file-file JAR.

Keuntungan JAR

- *Security*, karena file JAR dapat ditandatangani secara digital.
  - Mengurangi waktu download file-file class Java. Karena file JAR ukurannya kecil dan biasanya terdiri dari satu buah file saja. Jadi proses download akan lebih mudah dan cepat.
- Memperkecil ukuran file-file java
  - Karena file JAR secara otomatis mengkompres file-file Java, sehingga memperkecil ukuran file.
- *Package Versioning and Information*
  - File JAR dapat diberi informasi tertentu yang unik. Misalnya informasi versi, vendor, main-class dan lain-lain.
- *Portability*
  - Karena file JAR akan dapat diproses oleh JRE yang bersifat multiplatform

Manipulasi JAR

- Membuat JAR
  - Sintaks :
    - jar –cvf <namafileJAR> <namafileclass1> [<namafileclass2>,<namafileclass3>,…]
  - Contoh : jar –cvf coba.jar a.class b.class c.class
  - Contoh : jar –cvf coba2.jar *.class
- Melihat isi JAR
  - Sintaks : jar –tvf <namafileJAR>
  - Contoh : jar –tvf coba.jar
- Mengekstrak JAR
  - Sintaks : jar –xvf <namafileJAR>
  - Contoh : jar –xvf coba.jar

Manifest File

- Di dalam file JAR atau didalam direktori hasil ekstrak dari file JAR maka kita akan menemukan file MANIFEST.MF di dalam direktori META-INF.
- File MANIFEST.MF adalah metafile yang menyediakan berbagai informasi dalam file JAR.
- Secara default isi file MANIFEST.MF adalah:

```
Manifest-Version: 1.0
Created-By: 1.3.0 (Sun Microsystem Inc.)
```

Header Main-Class

- Header Main-Class digunakan agar Java mengetahui file main yang digunakan untuk mengeksekusi program Java.
- Cara membuat header Main-Class
  - Sintaks : Main-Class: <namafilemain>
  - Contoh : Main-Class: kelasMain
- Cara menambah (mengupdate) file MANIFEST.MF
  - Sintaks : jar –ufv <namafileJAR> <namafilemanifet>
  - Contoh : jar –ufv coba.jar mymanifest.mf

JDBC

- Java Database Connectivity?
- Java menyediakan JDBC yang berfungsi untuk berhubungan dengan database.
- Database yang didukung oleh Java cukup banyak, seperti : MySQL, Postgres, Oracle, DB2, Access dan lain-lain.
- JDBC berisi kumpulan kelas-kelas dan interface yang ditulis dengan bahasa Java.

JDBC (2)

- Yang dilakukan JDBC
  - Membangun koneksi ke data source
  - Mengirim statement ke data source
  - Memproses hasil statement tersebut
- Java menyediakan tiga produk JDBC:
  - JDBC driver manager
  - JDBC driver test suite
  - JDBC ODBC bridge

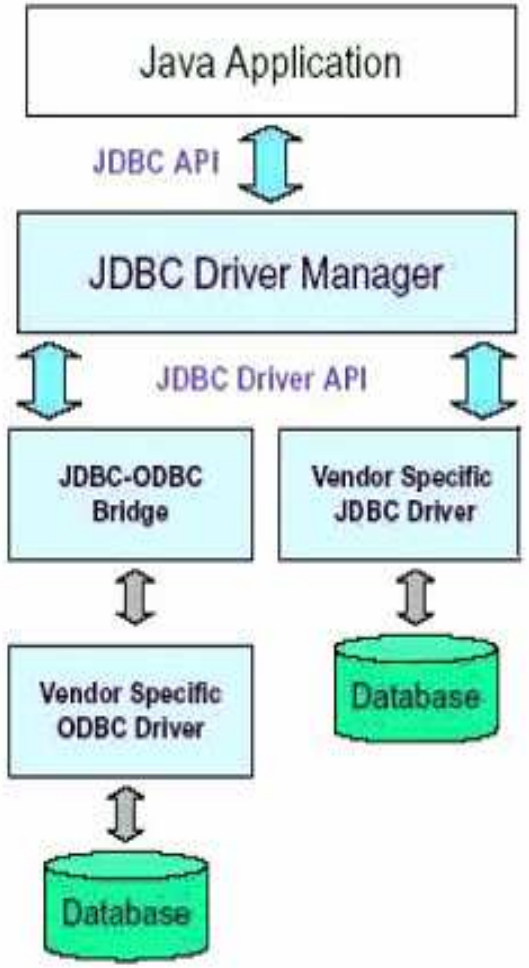
ODBC vs JDBC

- ODBC tidak cocok dipakai langsung dengan Java karena ditulis dengan bahasa C, pemanggilan dari Java ke C memiliki masalah keamanan, implementasi, robustness, dan portabilitas sistem.
- Penerjemahan dari C ke Java tidak akan berhasil baik. Contoh: Java tidak memiliki pointer.
- ODBC sulit dipelajari karena optionnya yang sulit walaupun untuk query yang sederhana.
- Java API diperlukan untuk mempertahankan solusi “murni Java”, agar dapat berjalan di berbagai platform. Karena ODBC harus diinstall dahulu di setiap client dan tidak semua platform.

Keunggulan JDBC

- Mempertahankan data enterprise yang ada
- Menyederhanakan development enterprise
- Tidak memerlukan konfigurasi pada jaringan komputer
- Akses penuh ke meta data
- Koneksi database menggunakan URL dan DataSource (yang meyediakan connection pooling dan distributed transaction)

Arsitektur JDBC



Lapisan Vendor Specific JDBC Driver merupakan driver JDBC yang dikeluarkan oleh para vendor pengembang RDBMS.

Sedangkan JDBC- ODBC Bridge berfungsi sebagai perantara untuk mengakses database melalui ODBC driver.

Baik JDBC driver maupun JDBC-ODBC Bridge diatur dan dapat diakses melalui JDBC Driver Manager.

Aplikasi yang kita kembangkan untuk mengakses database dengan memanfaatkan JDBC akan berinteraksi dengan JDBC Driver Manager.

JDBC API

- Tersedia dalam paket java.sql dan javax.sql.
- DriverManager – memanggil driver JDBC ke memori, dan dapat juga digunakan untuk membuka koneksi ke sumber data.
- Connection – mempresentasikan suatu koneksi dengan suatu data source, juga digunakan untuk membuat objek Statement, PreparedStatement dan CallableStatement.
- Statement – mempresentasikan suatu perintah SQL, dan dapat digunakan untuk menerima objek ResultSet.

JDBC API (2)

- PreparedStatement – merupakan alternatif untuk objek Statement SQL yang telah terkompilasi awal.
- CallableStatement – mempresentasikan suatu stored procedure, dan dapat digunakan untuk menjalankan stored procedures yang terkompilasi dalam suatu RDBMS yang mendukung fasilitas tersebut.
- ResultSet – mempresentasikan sebuah hasil dari database yang dihasilkan dari statemen SQL SELECT.
- SQLException – suatu class exception yang membungkus kesalahan (error) pengaksesan database.

JDBC API (3)

- javax.sql adalah bagian dari J2SE 1.4 dan J2EE 1.3. Paket ini memberikan beberapa tambahan yang telah tersedia pada java.sql :
  - DataSource – Objek ini dapat digunakan untuk penempatan DriverManager untuk lebih efisien dalam melakukan koneksi ke database (yang didefinisikan melalui data source).
  - XADataSource, XAConnection – mendukung transaksi terdisitrbusi.
  - RowSet – merupakan turunan dari ResultSet yang ditambah dukungan untuk resultset yang menampung hasil eksekusi database, walaupun koneksi terputus.

JDBC Data Type

JDBC Type	Java Type
BIT	boolean
TINYINT	byte
SMALLINT	short
INTEGER	int
BIGINT	long
REAL	float
FLOAT DOUBLE	double
BINARY VARBINARY LONGVARBINARY	byte[]
CHAR VARCHAR LONGVARCHAR	String

JDBC Type	Java Type
NUMERIC DECIMAL	BigDecimal
DATE	java.sql.Date
TIME TIMESTAMP	java.sql.Timestamp
CLOB	Clob*
BLOB	Blob*
ARRAY	Array*
DISTINCT	mapping of underlying type
STRUCT	Struct*
REF	Ref*
JAVA_OBJECT	underlying Java class

*SQL3 data type supported in JDBC 2.0

Tipe Driver JDBC

- JDBC Type 1 Driver – yang termasuk tipe ini adalah driver JDBC-ODBC Bridge. Driver ini mendelegasikan tugas pengakses data ke ODBC API. Driver ini memberikan unjuk kerja terlambat dari semua tipe lain. SUN menyediakan penerapan dari driver JDBC/ODBC.
- JDBC Type 2 Driver – driver ini menggunakan API native untuk akses data dan menyediakan class Java wrapper yang memungkinkan untuk dipanggil menggunakan driver JDBC.
- JDBC Type 3 Driver – ditulis 100% dengan Java dan menggunakan protokol jaringan yang vendor independent untuk mengakses independent remote listener.
- JDBC Type 4 Driver – ditulis 100% dengan Java dan merupakan driver yang lebih efisien dibanding tipe lain

Pemrograman JDBC

- Membangun koneksi
  - Memuat driver ODBC

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

    ```
    DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver ());
    ```

- Membangun koneksi URL
  Format: `jdbc:odbc:<nama_db>`
- Contoh lengkap:

```
String url = “jdbc:odbc:Buku”;
String user = “”;
String pass = “”;
Connection con =
DriverManager.getConnection(url,user,pass);
```

Pemrograman JDBC (2)

- Membuat Statement

  Menggunakan Obyek Connection yang sudah kita buat sebelumnya:
  - `Statement stmt = con.createStatement();`
- Menjalankan Statement
- Method **executeUpdate** untuk DDL dan DML insert, update, dan delete.
  - `String query = “delete from tabel where id=1”:`
  - `Statement stmt = con.createStatement();`
  - `int hsl = Stmt.executeUpdate(query);`
- Method **executeQuery** untuk DML select
  - `String query = “select * from tabel”;`
  - `Statement stmt = con.createStatement();`
  - `ResultSet rs = Stmt.executeQuery(query);`

Pemrograman JDBC (3)

- Mengambil hasil Statement dari Query dan Memprosesnya
- DDL dan DML: update, insert, dan delete
  - `int hsl = Stmt.executeUpdate(query);`
  - ```
    if(hsl == 1)
       System.out.println(“Berhasil”);
    else
       System.out.println(“Gagal”);
    ```
- DML: select
  - `ResultSet rs = Stmt.executeQuery(query);`
  - `while(rs.next()){`
    - `int a = rs.getInt(“fieldA”);`
    - `String b = rs.getString(“fieldB”);`
    - `float c = rs.getFloat(“fieldC”);`
  - `}`
    ```

Pemrograman JDBC (4)

- Tutup koneksi yang sudah dibuat.
  - `con.close();`

- Kita dapat membuat class yang berisi semua method yang membantu kita untuk melakukan koneksi dan transaksi ke database!

Penting!

- Harus mengetahui dan memiliki JDBC driver sesuai dengan database yang digunakan.
- Harus mengetahui cara koneksi dengan database.
- Harus mengimport `java.sql.*;`

Contoh: MySQL Create Table

```
import java.sql.*;
public class CreateTableSepatuApp {
    public static void main(String[] args) {
        String createTableSepatu =
            "CREATE TABLE SEPATU "
                + "(NAMA_SEPATU VARCHAR(40),"
                + "SUP_ID INTEGER,"
                + "HARGA REAL,"
                + "PENJUALAN INTEGER,"
                + "TOTAL INTEGER)";
        try {
            Class.forName("org.gjt.mm.mysql.Driver");
            Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/sepatudb?user=root&password=");
            Statement stmt = con.createStatement();
            stmt.executeUpdate(createTableSepatu);
            System.out.println("Tabel SEPATU berhasil dibuat.");
            stmt.close();
            con.close();
        } catch (ClassNotFoundException e) {
            System.out.println("Eksepsi: " + e.getMessage());
        } catch (SQLException e) {
            System.out.println("Eksepsi SQL: " + e.getMessage());
        }
    }
}
```

Membaca Isi Data ODBC

```
public class cobaDB {
    public static void main(String[] args) {
        String selectMhs = "select * from mhs";
        try {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            String url = "jdbc:odbc:mhs";
            String user = "";
            String pass = "";
            Connection con = DriverManager.getConnection(url,user,pass);

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery(selectMhs);
            while(rs.next()){
                String nim = rs.getString("nim");
                String nama = rs.getString("nama");
                            int ipk = rs.getInt("ipk");
                System.out.println(nim + "\t" + nama + "\t" + ipk);
            }
            rs.close();
            stmt.close();
            con.close();
        } catch (ClassNotFoundException e) {
            System.out.println("Eksepsi: " + e.getMessage());
        } catch (SQLException e) {
            System.out.println("Eksepsi SQL: " + e.getMessage());
        }
    }
}
```

Cursor ResultSet

- Method pergerakan kursor yang didukung oleh ResultSet:
  - previous() ke record sebelumnya
  - next() ke record selanjutnya
  - first() ke record pertama
  - last() ke record terakhir
  - absolute() ke nomor baris tertentu
  - relative() ke nomor baris dari baris sekarang
  - beforeFirst() ke nomor baris sebelum pertama
  - afterLast() ke nomor baris setelah terakhir

Cursor ResultSet

- Jika suatu ResultSet dibuat, selalu ResultSet tersebut berada pada poisi record sebelum record pertama (rs.beforeFirst()).
- Sehingga untuk mengambil data yang hanya terdiri dari satu baris, harus terlebih dahulu digunakan method rs.next() sekali.

Cursor ResultSet & Limit

- Method untuk mengambil jumlah baris:
  - getRow() yang mengembalikan nilai integer
- Method untuk membatasi jumlah baris hasil query select:
  - Statement.setFetchSize(number)

setFetchSize()

- setFetchSize() memiliki arah, yaitu:
  - ResultSet.FETCH_FORWARD untuk proses maju
  - ResultSet.FETCH_REVERSE untuk proses berbalik
  - ResultSet.FETCH_UNKNOWN untuk proses yang tidak diketahui
- Contoh:

```
Statement stmt = con.createStatement();
stmt.setFetchDirection(ResultSet.FETCH_FORWARD);
stmt.setFetchSize(30);
ResultSet rs = stmt.executeQuery(…);
```

Kembalian ResultSet

- null
  - Untuk metode getXXX yang mengembalikan obyek
- 0
  - Untuk metode getXXX yang mengembalikan tipe data primitif biasa
- false
  - Untuk metode getXXX yang mengembalikan tipe data boolean.

Exception dalam JDBC

- SQLException: ketika ada masalah pengaksesan data
- SQLWarning: ketika ada peringatan
- DataTruncation: ketika data mungkin terpotong
- BatchUpdateException: ketika tidak semua perintah update berhasil dilakukan.

DEMO JDBC

- Praktikum
- Tugas: terapkan dalam Socket Connection Oriented