

MODUL 5

DATABASE STORAGE 2

A. TUJUAN

1. Mahasiswa dapat melakukan penyimpanan database dengan Java batch
2. Mahasiswa dapat melakukan perubahan database dengan Java
3. Mahasiswa dapat melakukan penghapusan database dengan Java

B. ALOKASI WAKTU 1 x 50 menit

C. DASAR TEORI

1. Prepared Statement

Prepared statement mewakili statment SQL yang dapat dikompilasi ulang sehingga dapat dieksekusi berkali-kali. Statement ini menerima kueri SQL dengan parameter. Di dalam statement ini dapat memiliki simbol "?" yang nantinya digunakan sebagai placeholder parameter. Kita dapat meneruskan parameter secara dinamis dengan menggunakan metode PREPARED STATEMENT pada saat run time.

Setelah objek PreparedStatement dibuat, ada tiga cara untuk mengeksekusinya:

- **execute():** Method ini mengembalikan nilai boolean dan mengeksekusi statement SQL statis yang ada di objek prepared statemtn.
- **executeQuery():** Method ini mengembalikan ResultSet dari prepared statemtnt saat ini.
- **executeUpdate():** Method ini mengembalikan jumlah baris yang dipengaruhi oleh pernyataan seperti INSERT, DELETE, dan lainnya yang ada dalam Prepared Statement saat ini.

Contoh :

```
INSERT INTO member VALUES ("Nurcahya",32);
```

Untuk dapat menjalankannya, kita buat sebuah query dengan placeholder seperti ketentuan di atas.

```
String query = "INSERT INTO member(nama, umur)VALUES(?, ?)";
Statement ps = con.prepareStatement(query);
ps.setString(1,"Nurcahya");
ps.setInt(2,32);
ResultSet result = ps.executeQuery();
```

2. Batch Processing

Batch processing merujuk pada pemrosesan sejumlah pekerjaan secara otomatis dalam satu waktu, tanpa interaksi langsung dari pengguna. Batch processing sering digunakan untuk mengelola, mengolah, atau mengubah data dalam jumlah besar. Fungsi batch dalam Java dapat digunakan untuk mengotomatisasi tugas-tugas ini.

Pada konsep pengolahan database, sejumlah data yang besar dapat dieksekusi dalam bentuk batch. Konsep ini sangat mungkin dilakukan dengan menggunakan method berikut:

- **addBatch():** Method ini menambahkan nilai parameter ke batch secara internal. Kita dapat menambahkan kumpulan nilai lain, satu per satu, dimasukkan ke dalam statement SQL. Setelah batch lengkap dikirim ke database, setiap set parameter dimasukkan ke dalam SQL dan dieksekusi secara terpisah.
- **executeBatch():** Method ini menjalankan semua proses batch yang sudah dipersiapkan. Statement SQL yang sudah berisi kumpulan parameter dikirim ke database sekaligus. Array dalam bentuk int[] yang dikembalikan oleh method executeBatch() adalah array int yang menunjukkan berapa banyak record yang berhasil diproses oleh setiap pernyataan SQL dalam batch. Berikut adalah contoh penggunaan batch.

```
PreparedStatement ps = con.prepareStatement(sql);
ps.setString(1, "Nurchaya");
ps.setInt(2, "32");
ps.addBatch();
ps.setString(1, "Pradana");
ps.setInt(2, "30");
ps.addBatch();
int[] jml = ps.executeBatch();
```

3. Update Data

Update data adalah proses mengganti nilai yang ada di dalam sebuah record atau lebih dari tabel database. Pada dasarnya proses mengubah data dapat bersifat parsial atau total. Parsial berarti hanya field tertentu saja, seperti field nama dari tabel mahasiswa atau field nilai pada tabel mata kuliah. Update yang baik adalah perubahan data yang merujuk pada record yang spesifik dengan menggunakan parameter tertentu menggunakan WHERE. Parameter yang ditunjuk dapat berupa nilai unik atau primary key sehingga tidak mengubah record lain yang tersimpan dalam tabel. Untuk mengubah record dari database, kita perlu menggunakan pernyataan SQL UPDATE dan menjalankannya menggunakan objek koneksi. Contoh:

```
try {
    String sql = "UPDATE table_name SET column1 = ?, column2 = ?,
    column3 = ? WHERE id = ?";
    PreparedStatement statement = con.prepareStatement(sql);
```

```

        statement.setString(1, "new_value1");
        statement.setString(2, "new_value2");
        statement.setInt(3, 456);
        statement.setInt(4, 1);
        statement.executeUpdate();
        System.out.println("Record updated.");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

4. Hapus Data

Serupa dengan ubah data, proses hapus data membutuhkan parameter berupa field tertentu yang unik menggunakan WHERE, biasanya dapat berupa primary key yang ditunjuk. Hal ini untuk mengantisipasi terjadinya penghapusan secara masal akibat tidak adanya field dan nilai tertentu yang ditunjuk. Untuk menghapus record dari database, kita perlu menggunakan pernyataan SQL DELETE dan menjalankannya menggunakan objek koneksi. Contoh:

```


try {
    String sql = "DELETE FROM table_name WHERE id = ?";
    PreparedStatement statement = con.prepareStatement(sql);
    statement.setInt(1, 1);
    statement.executeUpdate();
    System.out.println("Record deleted.");
} catch (SQLException e) {
    e.printStackTrace();
}

```

D. PRAKTIKUM

1. Persiapan Database

- Kita masih menggunakan database pertemuan lalu yakni **DatabaseDemo**.

Fields	Indexes	Foreign Keys	Checks	Triggers	Options	Comment	SQL Preview		
Name				Type	Length	Decimals	Not null	Virtual	Key
id				int	11		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
vendor				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>	
tipe				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>	
mesin				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>	
maxSpeed				double			<input type="checkbox"/>	<input type="checkbox"/>	

2. Menulis Kode Untuk Update dan Hapus

- Import semua library yang dibutuhkan

```
package com.mycompany.databasedemo;

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

- Di dalam fungsi main(), definisikan config

```
String url = "jdbc:mysql://localhost:3306/databasedemo";
String username = "root";
String password = "";
```

- Buat perintah koneksi di dalam blok try catch

```
try {
    Class.forName(className: "com.mysql.cj.jdbc.Driver");
    Connection koneksi = DriverManager.getConnection(url, user: username, password);
    System.out.println(x: "Koneksi berhasil");
} catch (ClassNotFoundException ex) {
    System.out.println(x: ex.getMessage());
}
```

- Modifikasi blok try catch dengan menambahkan statement dan mencetak query.
Jika ingin mengubah data, gunakan prosedur ini

```
//query mengubah data dengan ID = 2
String query = "UPDATE mobil SET vendor = ?, tipe = ?, mesin = ?, maxSpeed = ? WHERE id = ?"
PreparedStatement st = koneksi.prepareStatement(sql: query);
st.setString(parameterIndex: 1, x: "Bugatti");
st.setString(parameterIndex: 2, x: "veyron");
st.setString(parameterIndex: 3, x: "8000");
st.setInt(parameterIndex: 4, x: 254);
st.setInt(parameterIndex: 5, x: 2);
st.executeUpdate();
System.out.println(x: "Data berhasil diubah");
```

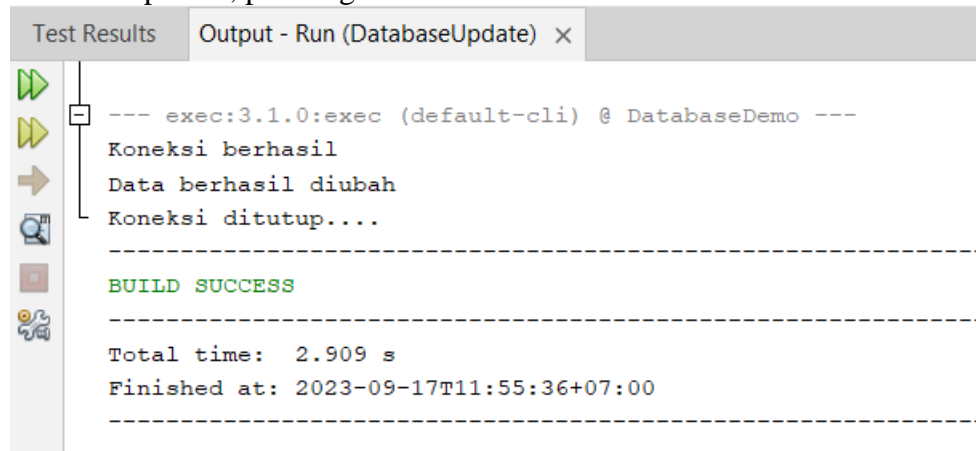
- Jika ingin menghapus, gunakan prosedur ini

```
//query menghapus data dengan ID=3
String query = "DELETE FROM mobil WHERE id = ?";
PreparedStatement st = koneksi.prepareStatement(sql: query);
st.setInt(parameterIndex: 1, x: 3);
st.executeUpdate();
System.out.println(x: "Data berhasil dihapus");
```

- Tutup koneksi

```
st.close();
koneksi.close();
System.out.println(x: "Koneksi ditutup....");
```

- Jalankan aplikasi, pada log akan muncul hasil berikut:



```
Test Results  Output - Run (DatabaseUpdate) x
--- exec:3.1.0:exec (default-cli) @ DatabaseDemo ---
Koneksi berhasil
Data berhasil diubah
Koneksi ditutup....
-----
BUILD SUCCESS
-----
Total time:  2.909 s
Finished at: 2023-09-17T11:55:36+07:00
-----
```

- Buka database kalian, perhatikan perubahan yang terjadi pada mobil

E. LATIHAN

1. Buatlah sebuah program yang dapat melakukan penyimpanan dengan memanfaatkan proses batch. Program tersebut bebas sehingga kalian boleh mengekspresikan imajinasi kalian sendiri. Namun demikian, program harus tetap menerapkan desain Singleton, sehingga class koneksi harus dipisah dari class main. Simpan dengan nama **Latihan1.zip!**
2. Modifikasilah program kalian sebelumnya sehingga dapat melakukan CRUD data anggota perpustakaan. Operasi yang perlu ditambah adalah update data anggota dan menghapus data anggota. Jangan lupa bahwa dalam melakukan update dan penghapusan data perlu diberi exception jika data yang ingin kalian ubah tersebut tidak terdapat dalam database. Simpan dengan nama **Latihan2.zip!**

[SELAMAT DATANG DI PERPUS PUSAT UNS]

1. Pendaftaran anggota
2. Tampilkan semua data anggota
3. Ubah data anggota
4. Hapus data anggota
5. Keluar

=====

Masukkan perintah : 1

Masukkan Nama : Nurcahya Pradana

Masukkan NIM : K353535

Masukkan Prodi : PTIK

Data berhasil dibuat

=====

Masukkan perintah : 2

[DATA ANGGOTA]

1 - Budi - K353531 - PTIK

2 - Randy - M0509051 - Informatika

3 - Nurcahya - K353535 - PTIK

=====

Masukkan perintah : 3

Masukkan no ID : 2

[DATA ANGGOTA LAMA]

Randy - M0509051 - Informatika

Masukkan Nama baru : Randy Orton

Masukkan NIM baru : M0509051

Masukkan Prodi baru : Sains Data

Data berhasil diperbarui

```
[DATA ANGGOTA BARU]
Randy Orton - M0509051 - Sains Data
=====
Masukkan perintah : 4
Masukkan no ID : 1
[DATA ANGGOTA]
Budi - K353531 - PTIK
Apakah data ini yang ingin Anda hapus (y/n)? :y
Data berhasil dihapus!
=====
Masukkan perintah : 5
Sampai jumpa!
```