PRAKTIKUM 9
PEMROGRAMAN BERORIENTASI OBJEK
"PERSISTENT OBJECT"

Disusun Oleh :

Derva Anargya Ghaly

24060121140149

Praktikum PBO Lab B

PROGRAM STUDI S-1 INFORMATIKA

DEPARTEMEN ILMU KOMPUTER / INFORMATIKA

FAKULTAS SAINS DAN MATEMATIKA

UNIVERSITAS DIPONEGORO

SEMARANG

2023

A. Menggunakan Persistent Object sebagai Model Basis Data Relasional

1. PersonDAO.java

```java
/**
 * PersonDAO.java 31/05/2023
 * Nama          : Derva Anargya Ghaly
 * NIM           : 24060121140149
 * Deskripsi     : interface untuk person access object
 */

public interface PersonDAO {
    public void savePerson(Person p) throws Exception;
}
```

2. Person.java

```java
/**
 * Person.java 31/05/2023
 * Nama          : Derva Anargya Ghaly
 * NIM           : 24060121140149
 * Deskripsi     : database model person
 */

public class Person {
    private int id; private String name;

        public Person(String n){ name = n;
    }
        public Person(int i, String n){ id = i;
        name = n;
    }
        public int getId(){
        return id;
    }
        public String getName(){ return name;
    }
}
```

3. MySQLPersonDAO.java

```java
/**
 * MySQLPersonDAO.java 31/05/2023
 * Nama         : Derva Anargya Ghaly
 * NIM          : 24060121140149
 * Deskripsi    : implementasi PersonDAO untuk mysql
 */

import java.sql.*;

public class MySQLPersonDAO implements PersonDAO{
    public void savePerson(Person person) throws Exception {
        String name = person.getName();

        //membuat koneksi, nama db, user, password menyesuaikan
        Class.forName(className:"com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection(url:"jdbc:mysql://localhost/pbo",user:"root",password:"Na mamu30");

        //kerjakan mysql query
        String query = "INSERT INTO person(name) VALUES('"+name+"')";
        System.out.println(query);
        Statement s = con.createStatement(); s.executeUpdate(query);
        // tutup koneksi database con.close();
    }
}
```

4. DAOManager.java

```java
/**
 * DAOManager.java 31/05/2023
 * Nama          : Derva Anargya Ghaly
 * NIM           : 24060121140149
 * Deskripsi     : pengelola DAO
 */

public class DAOManager{
    private PersonDAO personDAO;
        public void setPersonDAO(PersonDAO person) {
            personDAO = person;
    }
        public PersonDAO getPersonDAO() {
            return personDAO;
    }
}
```

5. **MainDAO.java**

```java
/**
 * MainDAO.java 31/05/2023
 * Nama          : Derva Anargya Ghaly
 * NIM           : 24060121140149
 * Deskripsi     : main program DAO
 */

public class MainDAO {
    Run | Debug
    public static void main(String args[]) {
        Person person = new Person(n:"Indra");
        DAOManager m = new DAOManager();
        m.setPersonDAO(new MySQLPersonDAO());
            try {
                m.getPersonDAO().savePerson(person);
            }
            catch(Exception e) {
                e.printStackTrace();
            }
    }
}
```

6. **Buat database dengan nama ‘pbo’ dan tabel pada database tersebut**

```
mysql> prompt Derva_24060121140149 >
PROMPT set to 'Derva_24060121140149 >'
Derva_24060121140149 > Create database praktikum_pbo;
Query OK, 1 row affected (0.00 sec)

Derva_24060121140149 >use praktikum_pbo;
Database changed
Derva_24060121140149 >show tables
    -> ;
Empty set (0.01 sec)

Derva_24060121140149 >CREATE TABLE person(
    -> id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    -> name VARCHAR(100));
Query OK, 0 rows affected (0.01 sec)

Derva_24060121140149 >SELECT * FROM person;
Empty set (0.01 sec)

Derva_24060121140149 >
```

Untuk membuat database dengan nama praktikum_pbo dan tabel pada database tersebut pertama harus membuat database baru. Setelah database praktikum_pbo berhasil dibuat, kemudian memanggil database tersebut sebagai database aktif.

Selanjutnya, memanggil perintah CREATE TABLE untuk membuat tabel di dalam database. Dalam permasalahan ini, tabel yang akan dibuat bernama person dan dua kolom, yaitu id dan name. Kolom id memiliki tipe data INT dan diatur sebagai primary key dengan opsi AUTO_INCREMENT, yang berarti nilai id akan dihasilkan secara otomatis. Kolom id juga diatur sebagai NOT NULL, berarti kolom tersebutharus memiliki nilai. Pada kolom name memiliki tipe data VARCHAR(100) yang artinya kolom name dapat menampung data string dengan panjang maks 100 karakter.

7. **Kompilasi semua *source code* dengan perintah:** `javac *.java`

```
PS C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum 9\DAO> javac *.java
PS C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum 9\DAO>
```

Pada hasil compile di atas, terlihat bahwa setelah perintah dijalankan, tidak ada pesan error yang muncul. Dengan tidak adanya pesan error saat menjalankan perintah, maka dapat dipastikan source code yang dibuat berhasil. Hal ini berarti tidak terdapat kesalahan sintaks atau masalah lainnya dalam source code yang dapat menghambat jalannya program.

8. **Jalankan MainDAO dengan perintah:**

```
java -classpath .\mysql-connector-j-8.0.33.jar;. MainDAO
```

```
C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum9>java -classpath .\mysql-connector-j-8.0.33.jar;. MainDAO
Loading class 'com.mysql.jdbc.Driver'. This is deprecated. The new driver class is 'com.mysql.cj.jdbc.Driver'. The driver is automati
cally registered via the SPI and manual loading of the driver class is generally unnecessary.
INSERT INTO person(name)VALUES('Indra')
```

Ketika ingin menjalankan perintah di atas, maka file program MainDAO dan mysql.jar harus ada pada satu folder yang sama agar dapat melakukan operasi sesuai yang diperlukan terhadap data di database.

Dalam menjalankan program MainDAO maka dilakukan pemanggilan menggunakan perintah java -classpath .\mysql-connector-j-8.0.33.jar;. MainDAO. Pada perintah -classpath digunakan untuk menentukan classpath yang berisi filemysql-connector-j-8.0.33.jar yang diperlukan untuk menghubungkan program dengan MySQL. Tanda titik koma (;) digunakan sebagai pemisah jika ada lebih dari satu file yang perlu ditambahkan ke classpath.

Setelah menjalankan perintah tersebut, akan muncul pesan yang menunjukkan bahwa program MainDAO berhasil dijalankan dan perintah untuk memasukkan data ke dalam tabel person telah berhasil dilakukan. Hal ini dapat dilihat dari pesan INSERT INTO person(name) VALUES('Indra').

Selanjutnya, dilakukan pemanggilan untuk memverifikasi data telah ditambahkan ke dalam tabel sesuai dengan perintah yang dijalankan sebelumnya. Pemanggilan perintah select * from person ketika telah menambahkan tabel sebelumnya, data pada tabel person masih kosong. Namun, setelah menjalankan perintah java seperti yang dijelaskan sebelumnya, kemudian menjalankan perintah select * from person, maka data dengan id 1 bernama Indra berhasil tercantum dalam tabel person. Hal ini menunjukkan bahwa program dan SQL CLI telah terhubung, sebagaimana ditandai dengan penambahan data ke dalam tabel person melalui perintah java yang telah dijalankan.

```
Derva_24060121140149 >SELECT * FROM person;
+----+-------+
| id | name  |
+----+-------+
|  1 | Indra |
+----+-------+
```

B. Menggunakan Persistent Object sebagai Objek Terserialisasi

1. SerializePerson.java

```java
/**
 * MySQLPersonDAO.java 31/05/2023
 * Nama          : Derva Anargya Ghaly
 * NIM           : 24060121140149
 * Deskripsi     : implementasi PersonDAO untuk mysql
 * Deskripsi     : Program untuk serialisasi objek Person
**/

import java.io.*;
//class Person
class Person implements Serializable{
    private String name;
    public Person(String n){
        name = n;
    }

    public String getName(){
        return name;
    }
}

//class SerializePerson
public class SerializePerson{
    Run | Debug
    public static void main(String[] args){
        Person person = new Person(n:"Panji");
        try {
            FileOutputStream f= new FileOutputStream(name:"person.ser");
            ObjectOutputStream s = new ObjectOutputStream(f);
            s.writeObject(person);
            System.out.println(x:"selesai menulis objek person");
            s.close();
        }
            catch(IOException e) {
                e.printStackTrace();
        }
    }
}
```

2. **Compile dan jalankan program di atas**

```
PS C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum 9\Serialize> java SerializePerson
selesai menulis objek person
PS C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum 9\Serialize>
```

Dengan menjalankan perintah di atas, program SerializePerson berhasil dijalankan tanpa ada pesan error dan terdapat keluaran yang dihasilkan oleh program tersebut, yaitu berupa pesan selesai menulis objek person sesuai dengan serialisasi yang telah diatur sebelumnya.

3. **ReadSerializedPerson.java**

```java
/**
 * MySQLPersonDAO.java 31/05/2023
 * Nama          : Derva Anargya Ghaly
 * NIM           : 24060121140149
 * Deskripsi     : Program untuk serialisasi objek Person
**/

import java.io.*;
public class ReadSerializedPerson{
    Run | Debug
    public static void main(String[] args){
        Person person = null;
            try {
                FileInputStream f = new FileInputStream(name:"person.ser");
                ObjectInputStream s = new ObjectInputStream(f);
                person = (Person)s.readObject();
                s.close();
                System.out.println("serialized person name = "+person.getName());
            }

            catch(Exception ioe) {
                    ioe.printStackTrace();
            }
    }
}
```

4. **Compile dan jalankan program di atas**

```
PS C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum 9\Serialize> java ReadSerializedPerson
serialized person name = Panji
PS C:\Users\ASUS TUF\Documents\COOLYEAH\SEMESTER 4\PRAKTIKUM PBO\Praktikum 9\Serialize>
```

Dengan menjalankan perintah di atas, maka akan mengkompilasi dan menjalankan program ReadSerializedPerson untuk membaca objek yang telah di-serialize sebelumnya. Output dari program di atas menampilkan informasi objek yang telah di-serialize, yaitu serialized person name = Panji.