LAPORAN PRAKTIKUM
PRAKTIKUM 9:
PERSISTENT OBJECT

Disusun Oleh :

Nama: Novi Dwi Fitriani
NIM : 24060121120027

PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK
LAB B2

DEPARTEMEN ILMU KOMPUTER / INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO
SEMARANG
2023

PRAKTIKUM 9: Persistent Object

A. Menggunakan Persistent Object sebagai Model Basis Data Relasional

1. PersonDAO.java

```
/**
* File           : PersonDAO.java
* Nama           : Novi Dwi Fitriani
* NIM            : 24060121120027
* Tanggal        : 31 Mei 2023
* Deskripsi      : interface untuk person access object
*/

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

2. Person.java

```
/**
* File      : Person.java
* Nama      : Novi Dwi Fitriani
* NIM       : 24060121120027
* Tanggal   : 31 Mei 2023
* Deskripsi : Person database model
*/

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

```
/**
* File      : MySQLPersonDAO.java
* Nama      : Novi Dwi Fitriani
* NIM       : 24060121120027
* Tanggal   : 31 Mei 2023
* 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("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/pbo","root","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

```
/**
* File      : DAOManager.java
* Nama      : Novi Dwi Fitriani
* NIM       : 24060121120027
* Tanggal   : 31 Mei 2023
* Deskripsi : pengelola DAO dalam program
*/

public class DAOManager{
      private PersonDAO personDAO;

      public void setPersonDAO(PersonDAO person){
            personDAO = person;
      }

      public PersonDAO getPersonDAO(){
            return personDAO;
      }
}
```

5. MainDAO.java

```
/**
* File      : MainDAO.java
* Nama      : Novi Dwi Fitriani
* NIM       : 24060121120027
* Tanggal   : 31 Mei 2023
* Deskripsi : Main program untuk akses DAO
*/

public class MainDAO{
      public static void main(String args[]){
            Person person = new Person("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 Novi_24060121120027>
PROMPT set to 'Novi_24060121120027>'
Novi_24060121120027> create database pbo;
Query OK, 1 row affected (0.02 sec)

Novi_24060121120027>use pbo;
Database changed
Novi_24060121120027>show tables;
Empty set (0.05 sec)

Novi_24060121120027>CREATE TABLE person(
    -> id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    -> name VARCHAR(100));
Query OK, 0 rows affected (0.54 sec)

Novi_24060121120027>select * from person;
Empty set (0.10 sec)
```

Untuk membuat database dengan nama pbo dan tabel pada database tersebut pertama harus membuat database baru dengan perintah create database pbo;. Setelah database pbo berhasil dibuat, kemudian memanggil perintah use pbo; untuk memilih database tersebut sebagai database aktif.

Selanjutnya, memanggil perintah CREATE TABLE untuk membuat tabel di dalam database pbo. 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, yang berarti kolom tersebut harus memiliki nilai (tidak boleh kosong). Pada kolom name memiliki tipe data VARCHAR(100) yang artinya kolom name dapat menampung data string dengan panjang maksimal 100 karakter.

Setelah tabel berhasil dibuat, maka database dapat dimanfaatkan untuk menyimpan dan mengelola data terkait dalam database pbo.

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

```
C:\Users\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\Praktikum 9\DAO>javac *.java

C:\Users\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\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\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\Praktikum 9\DAO>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 file mysql-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 (empty set). 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.

```
Novi_24060121120027>select * from person;
+----+-------+
| id | name  |
+----+-------+
|  1 | Indra |
+----+-------+
1 row in set (0.11 sec)
```

B. Menggunakan Persistent Object sebagai Objek Terserialisasi

1. SerializePerson.java

```
/**
File        : SerializePerson.java
Nama        : Novi Dwi Fitriani
NIM         : 24060121120027
Tanggal     : 31 Mei 2023
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{
      public static void main(String[] args){
            Person person = new Person("Panji");
            try{
                  FileOutputStream f= new
FileOutputStream("person.ser");
                  ObjectOutputStream s = new ObjectOutputStream(f);
                  s.writeObject(person);
                  System.out.println("selesai menulis objek
person");
                  s.close();
            }catch(IOException e){
                  e.printStackTrace();
            }
      }
}
```

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

```
C:\Users\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\Praktikum 9>javac SerializePerson.java

C:\Users\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\Praktikum 9>java SerializePerson
selesai menulis objek person
```

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**

```
/**
File        : ReadSerializedPerson.java
Nama        : Novi Dwi Fitriani
NIM         : 24060121120027
Tanggal     : 31 Mei 2023
Deskripsi   : Program untuk serialisasi objek Person
**/

import java.io.*;

public class ReadSerializedPerson{
      public static void main(String[] args){
            Person person = null;
            try{
                  FileInputStream f = new
FileInputStream("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**

```
C:\Users\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\Praktikum 9>javac ReadSerializedPerson.java

C:\Users\Novi Dwi\Documents\Kuliah\SMT 4\PBO\PRAKTIKUM\Praktikum 9>java ReadSerializedPerson
serialized person name = Panji
```

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.