LAPORAN PRAKTIKUM
PRAKTIKUM 9:
"PERSISTENT OBJECT"

Disusun Oleh :

Sana Saffanah
24060121140143

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
 //Penulis      : Sana Saffanah
 //NIM          : 24060121140143
 //Tanggal      : 31/05/23
 //Deskripsi: Interface untuk person access object

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

2. Person.java

```
//File          : Person.java
 //Penulis      : Sana Saffanah
 //NIM          : 24060121140143
 //Tanggal      : 31/05/23
 //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
 //Penulis      : Sana Saffanah
 //NIM          : 24060121140143
 //Tanggal      : 31/05/23
 //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","Manunggal.4");
        //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
 //Penulis      : Sana Saffanah
 //NIM          : 24060121140143
 //Tanggal      : 31/05/23
 //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
 //Penulis      : Sana Saffanah
 //NIM          : 24060121140143
 //Tanggal      : 31/05/23
 //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 Sana Saffanah_24060121140143>
PROMPT set to 'Sana Saffanah_24060121140143>'
Sana Saffanah_24060121140143> create database pbo;
Query OK, 1 row affected (0.03 sec)
```

```
Sana Saffanah_24060121140143> use pbo;
Database changed
Sana Saffanah_24060121140143>show tables;
Empty set (0.04 sec)

Sana Saffanah_24060121140143> CREATE TABLE person(
    -> id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    -> name VARCHAR(100));
Query OK, 0 rows affected (0.04 sec)

Sana Saffanah_24060121140143> select * from person;
Empty set (0.01 sec)
```

Pada pembuatan database akan menggunakan SQL Command Line dengan memasukkan prompt nama_nim yaitu Sana Saffanah_24060121140143. Kemudian membuat database yang diberi nama ‘pbo’ dengan perintah ‘create database pbo’. Setelah database pbo terbuat, kita menggunakan perintah ‘use pbo’ untuk menunjukkan bahwa database tersebut yang kita gunakan. Kemudian menggunakan perintah ‘show tables’ untuk melihat tabel dalam database, tetapi karena belum membuat tabel maka akan memunculkan hasil empty set yang berarti belum ada tabel apapun. Sehingga kita akan membuat tabel dalam database pbo menggunakan perintah ‘CREATE TABLE’. Dalam tabel yang dibuat akan berisi 2 kolom yaitu id dengan tipe data INT, kemudian diatur sebagai PRIMARRY KEY, NOT NULLS dan menggunakan AUTO_INCREMENT yang bertujuan agar nilai id akan dihasilkan secara otomatis. Kemudian untuk kolom kedua berisi name yang memiliki tipe data VARCHAR(100). Kemudian menggunakan perintah

select * from person dan akan memunculkan hasil empty set yang berarti tabel tersebut masih kosong belum diisi data apapun.

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

```
C:\Users\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9\DAO>javac *.java

C:\Users\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9\DAO>
```

Kita dapat melakukan kompilasi semua source code dengan menggunakan perintah javac *.java. Kemudian, jika tidak ada pesan error pada saat menjalankan perintah, maka artinya source code yang kita jalankan berhasil dan dapat di compile.

8. **Jalankan MainDAO dengan perintah:**

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

```
C:\Users\Asus\Downloads\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 automatically registered via the SPI and manual
 loading of the driver class is generally unnecessary.
INSERT INTO person(name) VALUES('Indra')

C:\Users\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9\DAO>
```

Untuk menjalankan perintah di atas, yang perlu dilakukan pertama adalah memastikan bahwa file MainDAO.java dan mysql.jar ada dalam 1 folder yang sama. Kemudian jalankan perintah menggunakan **`java -classpath .\mysql-connector-j-8.0.33.jar;. MainDAO.`** Jika berhasil tanpa error, maka akan muncul hasil seperti di atas, yaitu INSERT INTO person(name) VALUES(‘Indra’) yang artinya adalah MainDAO dan perintah INSERT data ke tabel berhasil dijalankan.

```
Sana Saffanah_24060121140143> select * from person;
+----+-------+
| id | name  |
+----+-------+
|  1 | Indra |
+----+-------+
1 row in set (0.01 sec)
```

Kemudian untuk kita dapat menggunakan SQL Command Line untuk melihat hasil dari perintah yang berhasil dijalankan sebelumnya. Kemudian yang dilakukan adalah menggunakan perintah select * from untuk melihat isi data pada tabel. Isi data pada tabel sebelumnya adalah empty set yang berarti kosong, tetapi setelah kita

menjalankan MainDAO dengan perintah **`java -classpath .\mysql-connector-j-8.0.33.jar;. MainDAO`** maka isi data pada tabel telah terupdate dan terisi. Dimana kolom id berisi 1 dan kolom name berisi Indra.Sehingga dapat disimpulkan bahwa program dan SQL Command Line telah terhubung.

B. Menggunakan Persistent Object sebagai Objek Terserialisasi

1. SerializePerson.java

```
//File          : SerializePerson.java
 //Penulis     : Sana Saffanah
 //NIM         : 24060121140143
 //Tanggal     : 31/05/23
 //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\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9>javac SerializePerson.java

C:\Users\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9>java SerializePerson
selesai menulis objek person
```

Pada program SerializePerson berhasil di compile dan dijalankan seperti yang sudah dilakukan tanpa adanya error dan memunculkan hasil yaitu ‘selesai menulis objek person’ sesuai pada perintah yang telah dibuat.

3. ReadSeializedPerson.java

```
//File          : ReadSerializedPerson.java
 //Penulis      : Sana Saffanah
 //NIM          : 24060121140143
 //Tanggal      : 31/05/23
 //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\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9>javac ReadSerializedPerson.java

C:\Users\Asus\Downloads\PBO\PRAKTIKUM\Praktikum 9>java ReadSerializedPerson
serialized person name = Panji
```

Pada program ReadSerializedPerson berhasil di compile dan dijalankan seperti yang sudah dilakukan tanpa adanya error dan memunculkan hasil yaitu ‘serialized person name = Panji’. Program ini dijalankan bertujuan untuk membaca objek yang telah di serialize sebelumnya.