LAPORAN PRAKTIKUM
PRAKTIKUM 9:
“PERSISTENT OBJECT”

Disusun Oleh :

Silvira Nabila Anggita Giraldi
24060121120011

PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK
LAB B1

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

A. Menggunakan Persistent Object sebagai Model Basis Data Relasional

1. Interface PersonDAO.java

```
/**
File : PersonDAO.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
Deskripsi : interface untuk person access object
**/
public interface PersonDAO {
    void savePerson(Person p) throws Exception;
}
```

2. Kelas Person.Java

```
/**
File : Person.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
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. Kelas MySQLPersonDAO.java

```
/**
File : MYSQLPersonDAO.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
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","An2v2328.");
        //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. Kelas DAOManager.java

```
/**
File : DAOManager.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
Deskripsi : pengelola DAO dalam program
**/
public class DAOManager{
      private PersonDAO personDAO;

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

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

5. Kelas MainDAO.java

```
/**
File : MainDAO.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
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. Membuat database dengan nama ‘pbo’ dan tabel pada database tersebut dengan : **CREATE TABLE person(id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,name VARCHAR(100))**

```
mysql> prompt Silvira Nabila_24060121120011>
PROMPT set to 'Silvira Nabila_24060121120011> '
Silvira Nabila_24060121120011> create database pbo;
Query OK, 1 row affected (0.02 sec)

Silvira Nabila_24060121120011> use pbo;
Database changed
Silvira Nabila_24060121120011> show tables;
Empty set (0.04 sec)

Silvira Nabila_24060121120011> CREATE TABLE person(
    -> id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,name VARCHAR(100));
Query OK, 0 rows affected (0.12 sec)

Silvira Nabila_24060121120011> select * from person;
Empty set (0.02 sec)
```

Langkah pertama yaitu membuat database dengan nama pbo kemudian menggunakan database pbo tersebut. Setelah itu memanggil perintah CREATE TABLE untuk membuat tabel di dalam database pbo. Tabel yang akan dibuat yaitu tabel person yang terdiri dari 2 column (id dan name). Column id memiliki tipe data INT dan diatur sebagai PRIMARY KEY dengan opsi AUTO_INCREMENT kemudian juga diatur NOT NULL dimana column tersebut harus memiliki nilai. Column name memiliki tipe data VARCHAR(100) yaitu dapat menampung data string dengan panjang maximal 100 karakter.

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

```
D:\SEMESTER 4\PRAKTIKUM\PBO\Praktikum 9>javac *.java

D:\SEMESTER 4\PRAKTIKUM\PBO\Praktikum 9>
```

Hasil compile terlihat bahwa setelah perintah dijalankan tidak terdapat error yang muncul. Maka jika tidak terdapat kesalahan source code tidak ditemukan kesalahan sintaks yang menghambat jalannya program.

8. Jalankan MainDAO dengan perintah

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

```
D:\SEMESTER 4\PRAKTIKUM\PBO\Praktikum 9>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')
```

Untuk menjalankan perintah tersebut, maka file program MainDAO dan mysql.jar harus terdapat pada 1 folder yang sama. Menjalankan program MainDAO dilakukan pemanggilan menggunakan perintah java -classpath .\mysql-connector-j-8.0.33.jar;. MainDAO. Perintah classpath berfungsi untuk menentukan classpath yang diperlukan untuk menghubungkan program dengan MySQL. Setelah menjalankan perintah, maka akan muncul pesan bahwa

MainDAO dapat 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'). Untuk melihat tabel person yang sudah diberikan data, maka dilakukan pemanggilan perintah select * from person. Muncul tabel person yang terdiri dari id 1 dan nama Indra. 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.

```
Silvira Nabila_24060121120011> select * from person;
+----+-------+
| id | name  |
+----+-------+
|  1 | Indra |
+----+-------+
1 row in set (0.00 sec)
```

B. Menggunakan Persistent Object sebagai Objek Terserialisasi

1. SerializePerson.java

```
/**
File : SerializePerson.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
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

```
D:\SEMESTER 4\PRAKTIKUM\PBO\Praktikum 9>javac SerializePerson.java

D:\SEMESTER 4\PRAKTIKUM\PBO\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

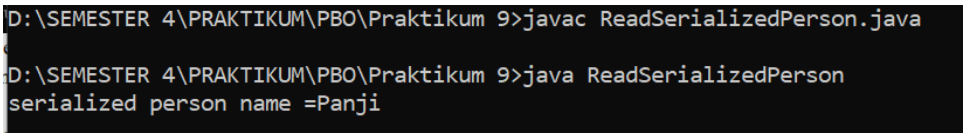
```
/**
File : ReadSerializePerson.java
Pembuat : Silvira Nabila Anggita Giraldi
NIM : 24060121120011
Tanggal : 31 Mei 2023
Lab : B1
Deskripsi : program untuk serialisasi objek Person
**/

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



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