LAPORAN PRAKTIKUM
PRAKTIKUM 9:
“PERSISTENT OBJECT”

Disusun Oleh :

Dafa Kurnia Dinata
24060121120003

PRAKTIKUM JARINGAN KOMPUTER
LAB B2

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

PERSISTENT OBJECT

A. Menggunakan *Persistent Object* sebagai model basis data relasional

a. PersonDAO.java

```
	/**
	 * Nama : Dafa Kurnia Dinata
	 * NIM : 24060121120003
	 * File : PersonDAO.java
	 * Deskripsi : interface untuk person access object
	*/
	public interface PersonDAO{
		public void savePerson(Person p) throws Exception;
}
```

b. Person.java

```
	/**
	 * Nama : Dafa Kurnia Dinata
	 * NIM : 24060121120003
	 * File : Person.java
	 * 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;
		}
}
```

c. MySQLPersonDAO.java

```
import java.sql.*;
/**
 * Nama          : Dafa Kurnia Dinata
 * NIM           : 24060121120003
 * FIle          : MySQLPersonDAO.java
 * Deskripsi     : implentasi PersonDAO untuk MySQL
*/


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(

"jbdc:mysql://localhost/pbo","root","");
        // 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();
    }
}
```

d. DAOManager.java

```
/**
 * Nama          : Dafa Kurnia Dinata
 * NIM           : 24060121120003
 * FIle          : DAOManager.java
 * Deskripsi     : pengelola DAO dalam program
 */
public class DAOManager{
    private PersonDAO personDAO;

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

e. MainDAO.java

```
/**
 * Nama          : Dafa Kurnia Dinata
 * NIM           : 24060121120003
 * FIle          : MainDAO.java
 * Deskripsi     : file main
 */
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();
        }
    }
}
```

Buat Database dengan nama “pbo” dan tabel pada database tersebut

```
mysql> prompt Dafa_24060121120003>
PROMPT set to 'Dafa_24060121120003> '
Dafa_24060121120003> create database PBO;
Query OK, 1 row affected (0.78 sec)

Dafa_24060121120003> use PBO;
Database changed
Dafa_24060121120003> show tables;
Empty set (0.43 sec)

Dafa_24060121120003> CREATE TABLE person(
    -> id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    -> name VARCHAR(100));
Query OK, 0 rows affected (0.27 sec)

Dafa_24060121120003> select * from person;
Empty set (0.05 sec)
```

Untuk membuat database menggunakan SQL CLI dengan perintah seperti yang terlihat pada gambar di atas, langkah pertama adalah membuat database pbo seperti yang ditunjukkan pada screenshot pertama. Setelah itu, gunakan perintah "use pbo" untuk mengaktifkan database pbo. Saat perintah "show tables" dieksekusi, tabel masih kosong karena belum ada tabel yang dibuat. Selanjutnya, buat tabel sesuai dengan perintah modul, yaitu membuat tabel "person" dengan atribut atau kolom "id" dan "name" serta tipe data yang sesuai dengan screenshot kedua. Pada screenshot ketiga, karena belum ada data yang dimasukkan, saat perintah "select * from person" dieksekusi, hasilnya masih kosong karena belum ada data yang ada dalam tabel.

Kompilasi semua source code menggunakan perintah; `javac *.java`

```
D:\UNDIP\Semester 4\PBO\Praktikum 9>javac *.java

D:\UNDIP\Semester 4\PBO\Praktikum 9>
```

Setelah database dan tabel berhasil dibuat, langkah selanjutnya adalah menjalankan semua source code yang telah dibuat menggunakan perintah seperti yang terlihat pada gambar di atas. Setelah menjalankannya tanpa adanya masalah, semua source code berhasil dikompilasi.

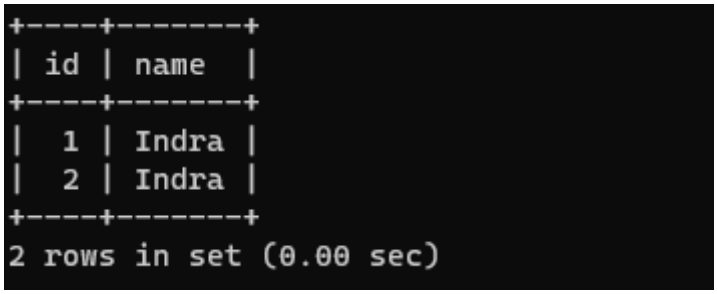
Jalankan MainDAO dengan perintah

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

```
D:\UNDIP\Semester 4\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 automatically registered via the SPI and manual loading of the driver class is
generally unnecessary.
INSERT INTO person(name) VALUES('Indra')
```

Setelah menjalankan perintah yang telah disebutkan di atas untuk menghubungkan program dan SQL CLI, semua file source code dan file mysql.jar ditempatkan dalam satu folder. Setelah menjalankan perintah tersebut, sebuah pesan akan muncul seperti yang ditunjukkan pada gambar di atas. Di pesan tersebut, terdapat perintah "INSERT INTO person(name) VALUES('Indra')" yang menandakan bahwa MainDAO telah berhasil dieksekusi dan perintah untuk memasukkan data ke dalam tabel "person" juga

berhasil dilakukan. Data tabel sudah otomatis terisi saat menjalankan perintah select * from person; pada mysql



```
+----+-------+
| id | name  |
+----+-------+
|  1 | Indra |
|  2 | Indra |
+----+-------+
2 rows in set (0.00 sec)
```

Menggunakan persistent object sebagai objek terserialisasi

- **SerializePerson.java**

```
/*
 * Nama         : Dafa Kurnia Dinata
 * NIM          : 24060121120003
 * FIle         : SerializePerson.java
 * Deskripsi    : program untuk serialisasi prorgram
*/
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();
                }
            }
}
```

Compile dan jalankan program menggunakan code dibawah ini:

```
javac SerializePerson.java
java SerializePerson
```

Program berhasil di compile dan dijalankan dan menghasilkan “selesai menulis objek person” seperti pada modul.

```
D:\UNDIP\Semester 4\PBO\Praktikum 9>javac SerializePerson.java

D:\UNDIP\Semester 4\PBO\Praktikum 9>java SerializePerson
selesai menulis objek person
```

ReadSerializedPerson.java

```
/**
 * Nama          : Dafa Kurnia Dinata
 * NIM           : 24060121120003
 * File          : ReadSerializedPerson.java
 * 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();
            }
      }
}
```

Jika compile berhasil, akan muncul tulisan “serialized person name = Panji”.

```
D:\UNDIP\Semester 4\PBO\Praktikum 9>javac ReadSerializedPerson.java

D:\UNDIP\Semester 4\PBO\Praktikum 9>java ReadSerializedPerson
serialized person name = Panji
```