LAPORAN PRAKTIKUM PEMOGRAMAN BERIORIENTASI OBJEK

PRAKTIKUM 9 – *PERSISTENT OBJECT*

Disusun untuk Memenuhi Tugas Individu Praktikum 9

Pada Mata Kuliah Pemograman Beriorientasi Objek Semester 4

Disusun oleh:

Nama : Yesy Margharetta Munthe

NIM : 24060121120031

Lab : B2

PROGRAM STUDI INFORMATIKA

DEPARTEMEN ILMU KOMPUTER/INFORMATIKA

FAKULTAS SAINS DAN MATEMATIKA

UNIVERSITAS DIPONEGORO

SEMARANG

2023

PERSISTENT OBJECT

3.1 Menggunakan Persistent Object sebagai model basis data relasional

1. Interface PersonDAO.java

```
/**
* File: PersonDAO.java
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* Deskripsi: interface untuk person access object
*/

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

2. Kelas Person.java

```
/**
* File: Person.java
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* 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
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* 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", "pangemor123");
		//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
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* 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
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* 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();
                }
        }
}
```

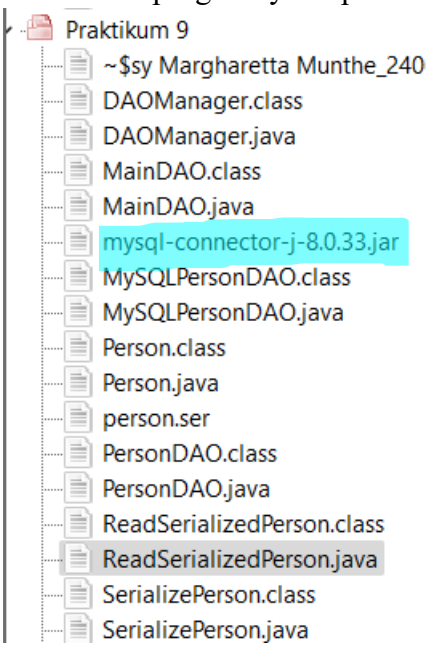
6. Buatlah database dengan nama ‘pbo’ dan tabel pada database

```
mysql> prompt Yesy_24060121120031>
PROMPT set to 'Yesy_24060121120031>'
Yesy_24060121120031> create database pbo;
Query OK, 1 row affected (0.04 sec)

Yesy_24060121120031> use pbo;
Database changed
Yesy_24060121120031> show tables;
Empty set (0.08 sec)

Yesy_24060121120031> CREATE TABLE person(
    -> id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    -> name VARCHAR(100));
Query OK, 0 rows affected (0.06 sec)

Yesy_24060121120031> SELECT *  FROM person;
Empty set (0.01 sec)
```

Database dibuat dengan menggunakan SQL CLI dengan perintah seperti SS diatas. Untuk membuat *database* pbo digunakan create database pbo, kemudian untuk mengaktifkan *database* pbo digunakan perintah *use* pbo. Karena belum ada tabel yang dibuat maka ketika menjalankan perintah *show tables* maka akan mengembalikan *empty set*. Untuk membuat tabel dalam database maka gunakan perintah *CREATE*

TABLE seperti pada SS diatas. Setelah perintah berhasil dijalankan maka gunakan *SELECT * FROM person* untuk mengecek isi tabel. Karena tabel belum diisi maka akan mengembalikan *empty set.*

7. File berekstensi *.jar (mysql-connector-java-[versi].jar) telah didownload dan dibuat satu direktori dengan source code program yaitu pada file Praktikum 9.



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

```
C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>javac *.java

C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>
```

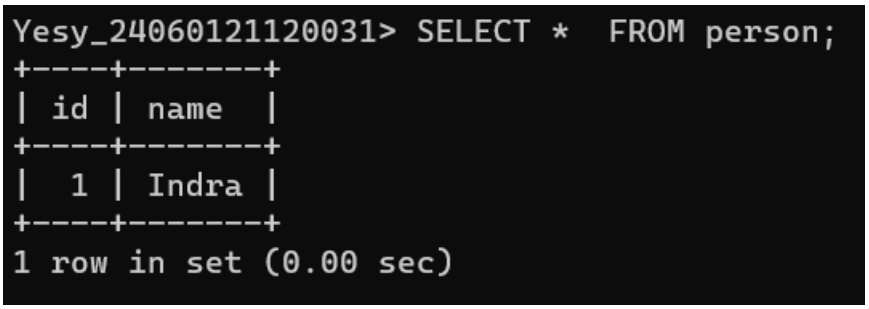
Setelah membuat database dan tabel, semua *source code* pada modul praktikum yang telah ditulis di notepad++ dapat dijalankan dengan perintah javac *.java. SS menunjukkan bahwa perintah telah berhasil dijalankan dan tidak ditemukan error.

9. Jalankan MainDAO dengan perintah:

```
java -classpath .\mysql-connector-java-[versi].jar;. MainDAO
```

```
C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>java -classpath .\mysql-connect
or-j-8.0.33.jar;. MainDAO
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysq
l.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')
```

Perintah diatas dibuat untuk menghubungkan program dengan SQL CLI. Untuk menghubungkan file java dengan sql maka dibutuhkan file mysql-conecctor-java-j-8.0.33.jar yang disimpan dalam folder yang sama. Setelah menjalankan MainDAO dengan perintah seperti pada SS diatas maka person bernama Indra yang ditambahkan di MainDAO akan langsung dimasukkan ke *database* pbo yang telah dibuat di SQL CLI. Dengan demikian maka akan ada penambahan *record* pada tabel. Untuk melihat penambahan *record* tersebut dapat menggunakan kembali perintah *SELECT * FROM person* maka akan menampilkan tabel dengan *person* bernama Indra dan id=1 seperti pada SS dibawah:



3.2 Menggunakan *Persistent Object* sebagai Objek Terserialisasi

1. SerializePerson.java

```
/**
* File: SerializePerson.java
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* 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 diatas

```
C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>javac SerializePerson.java

C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>java SerializePerson
selesai menulis objek person
```

Kelas erializePerson.java berhasil di compile dan dijalankan seperti pada SS diatas dan mengembalikan hasil ‘selesai menulis objek person’ sebagai output yang sudah ditetapkan di *source code*. Kelas ini dibuat untuk menyimpan objek dalam file yang bernama “person.ser”.

3. ReadSerializePerson.java

```
/**
* File: ReadSerializePerson.java
* Penulis: Yesy Margharetta Munthe
* NIM: 24060121120031
* 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 diatas

```
C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>javac ReadSerializedPerson.java


C:\Users\ACER\Documents\Semester 4\PBO\Praktikum\Praktikum 9>java ReadSerializedPerson
serialized person name = Panji
```

Kelas ini dibuat untuk membaca objek yang telah terealisasi. Setelah kelas ReadSerializePerson.java berhasil di compile dan dijalankan seperti pada SS diatas maka mengembalikan hasil ‘serialized person name = Panji’ sebagai objek yang sudah direalisasikan di *source code*.