

**LAPORAN PRAKTIKUM**  
**PRAKTIKUM 9:**  
**PERSISTENT OBJECT**



**Disusun Oleh:**

Muhammad Naufal Arkan  
24060121130073

**PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK**  
**LAB B**

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

### 1. Interface PersonDao.java

```
/**
 * File : PersonDAO.java          31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * Deskripsi : Interface untuk person access object
 */

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

### 2. Class Person.java

```
/**
 * File : Person.java            31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * 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. Class MySQLPersonDao.java

```
/**
 * File : MySQLPersonDAO.java    31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * Deskripsi : implemnetasi 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", "");
        // 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. Class DAOManager.java

```

/**
 * File : DAOManager.java          31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * Deskripsi : pengelola DAO dalam program
 */

public class DAOManager {
    private PersonDAO personDAO;

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

```

#### 5. Class MainDAO.java

```

/**
 * File : MainDAO.java            31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * 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'

Create database

pbo utf8mb4\_general\_ci Create

Buat tabel Person dengan query yang ada di modul:

Run SQL query/queries on database pbo:

```
1. CREATE TABLE person(id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,name VARCHAR(100));
```

Clear Format Get auto-saved query

☐ Bind parameters

☐ Show this query here again ☐ Retain query box ☐ Rollback when finished ☒ Enable foreign key checks Go

Tabel Person masih kosong.

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0007 seconds.)

```
SELECT * FROM `person`
```

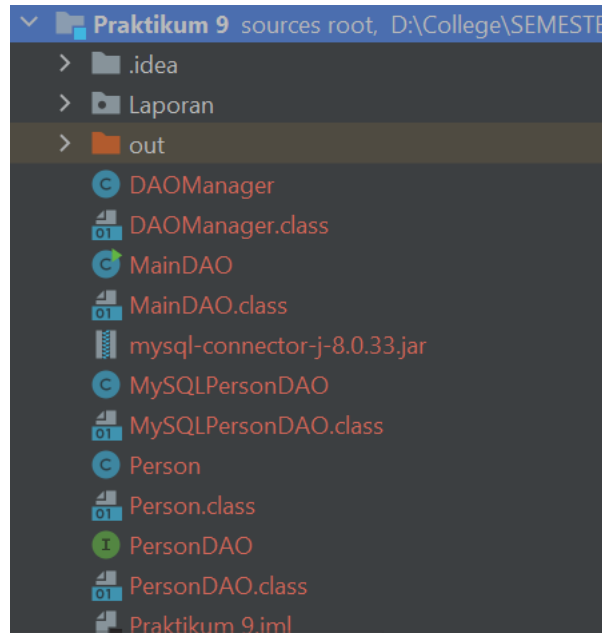
☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

id	name
----	------

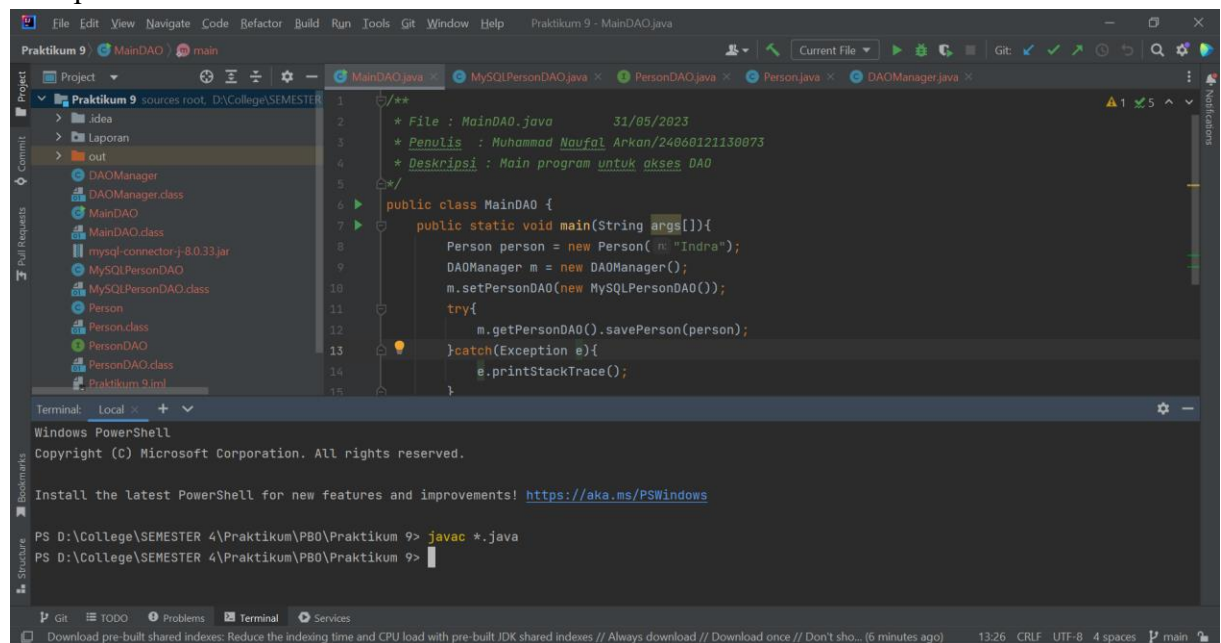
Query results operations

Create view

7. MySQL driver diletakkan satu direktori dengan source code program



8. Kompilasi semua source code



9. Jalankan MainDAO

The screenshot shows an IDE with a project named 'Praktikum 9'. The main editor displays the code for 'MainDAO.java'. The code includes a package declaration, imports for 'Person', 'DAOManager', and 'MySQLPersonDAO', and a 'MainDAO' class with a 'main' method. The 'main' method creates a 'Person' object named 'Indra', a 'DAOManager' object, and a 'MySQLPersonDAO' object. It then calls 'savePerson' on the 'DAOManager' object. The terminal window shows the command 'javac \*.java' and 'java -cp ".\mysql-connector-j-8.0.33.jar;" MainDAO.java'. The output shows the successful insertion of a record into the 'person' table.

```
1 // **
2 * File : MainDAO.java 31/05/2023
3 * Penulis : Muhammad Naufal Arkan/24060121130073
4 * Deskripsi : Main program untuk akses DAO
5 // **
6
7 public class MainDAO {
8     public static void main(String args[]){
9         Person person = new Person("Indra");
10        DAOManager m = new DAOManager();
11        m.setPersonDAO(new MySQLPersonDAO());
12        try{
13            m.getPersonDAO().savePerson(person);
14        }catch(Exception e){
15            e.printStackTrace();
16        }
17    }
18 }
```

Terminal: Local x + v

```
PS D:\College\SEMESTER 4\Praktikum\PBO\Praktikum 9> javac *.java
PS D:\College\SEMESTER 4\Praktikum\PBO\Praktikum 9> java -cp ".\mysql-connector-j-8.0.33.jar;" MainDAO.java
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')
PS D:\College\SEMESTER 4\Praktikum\PBO\Praktikum 9>
```

## 10. Terjadi penambahan record pada tabel

The screenshot shows the phpMyAdmin interface. The left sidebar shows the database structure with 'person' selected. The main area shows the 'person' table with one record. The record has an 'id' of 1 and a 'name' of 'Indra'. The table structure is shown below the record.

id	name
1	Indra

### 1. Class SerializePerson.java

```
/**
 * File : SerializePerson.java          31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * Deskripsi : Program untuk serialisasi objek Person
 */

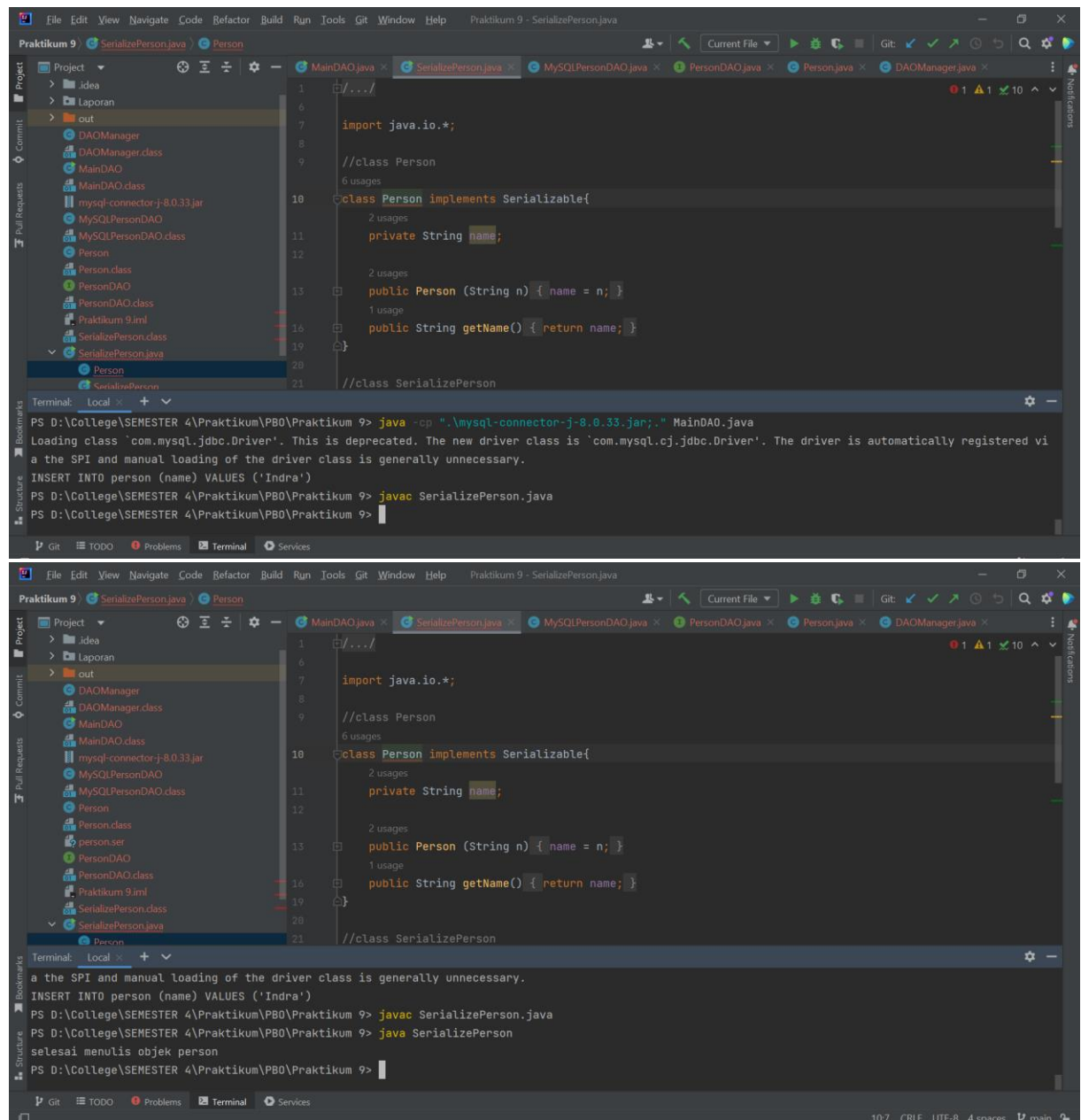
import java.io.*;

//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



### 3. Class ReadSerializedPerson.java

```

/**
 * File : ReadSerializedPerson.java          31/05/2023
 * Penulis : Muhammad Naufal Arkan/24060121130073
 * 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

The screenshot shows an IDE with the following components:

- Project Explorer:** Lists files including DAOManager.class, MainDAO.class, MySQLPersonDAO.class, Person.class, person.ser, PersonDAO.class, and ReadSerializedPerson.class.
- Code Editor:** Displays the source code of `ReadSerializedPerson.java`. The code includes a package comment, imports `java.io.*`, and defines a `main` method that reads a serialized object from `person.ser` and prints its name.
- Terminal:** Shows the execution of the program. The commands `javac ReadSerializedPerson.java` and `java ReadSerializedPerson` are executed, resulting in the output `serialized person name = Panji`.

```

1  /**
2   * File : ReadSerializedPerson.java      31/05/2023
3   * Penulis : Muhammad Naufal Arkan/24060121130073
4   * Deskripsi : Program untuk serialisasi objek Person
5   */
6
7   import java.io.*;
8
9   public class ReadSerializedPerson{
10      public static void main(String[] args) {
11          Person person = null;
12          try{
13              FileInputStream f = new FileInputStream( "name: "person.ser");
14              ObjectInputStream s = new ObjectInputStream(f);
15              person = (Person) s.readObject ();
16              s.close ();
17              System.out.println("serialized person name = "+person.getName());
18          }catch(Exception ioe) {
19              ioe.printStackTrace ();

```

```

Terminal: Local x + v
PS D:\College\SEMESTER 4\Praktikum\PBO\Praktikum 9> javac ReadSerializedPerson.java
PS D:\College\SEMESTER 4\Praktikum\PBO\Praktikum 9> java ReadSerializedPerson
serialized person name = Panji
PS D:\College\SEMESTER 4\Praktikum\PBO\Praktikum 9>

```