

**LAPORAN PRAKTIKUM  
PEMROGRAMAN BERBASIS OBJEK  
LAB B2**



**Disusun oleh :**

**Muhammad Afiat Yulianto (24060121140141)**

**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS DAN MATEMATIKA  
UNIVERSITAS DIPONEGORO  
SEMARANG  
2023**

## A. Menggunakan Persistent Object Sebagai Model Basis Data Relasional

### 1. PersonDAO.java

```
//Nama File: PersonDAO.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//Deskripsi: interface untuk person access object

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

### 2. Person.java

```
//Nama File: Person.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//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

```
//Nama File: MySQLPersonDAO.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//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", "password");

        // 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

```
//Nama File: DAOManager.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//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

```
//Nama File: MainDAO.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//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* “pbo”

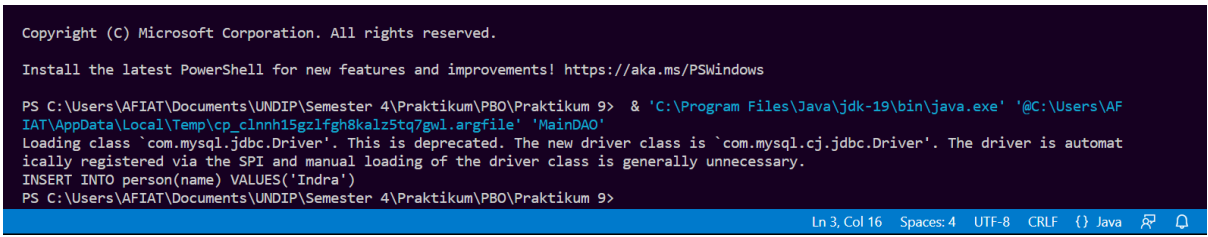
```
CREATE DATABASE pbo
```

## 7. Membuat Tabel “Person”

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

## 8. Compile dan Run Source Code

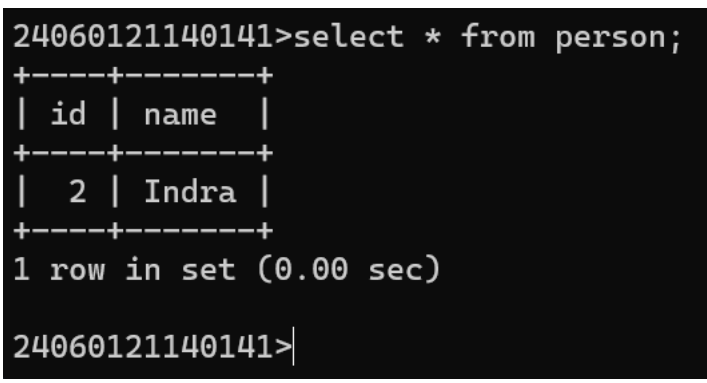
Untuk meng-*compile* semua src sekaligus, gunakan *command* `javac *.java` pada terminal. Kemudian, run MainDAO beserta connector mysql yang telah di download.



```
Copyright (C) Microsoft Corporation. All rights reserved.  
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows  
  
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> & 'C:\Program Files\Java\jdk-19\bin\java.exe' '@C:\Users\AFIAT\AppData\Local\Temp\cp_clnnh15gz1fgh8kalz5tg7gw1.argfile' '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')  
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9>
```

Screenshot di atas merupakan hasil run src dan input person “Indra” ke *database* pbo.

## 9. Check Tabel “Person”



```
24060121140141>select * from person;  
+----+-----+  
| id | name |  
+----+-----+  
|  2 | Indra |  
+----+-----+  
1 row in set (0.00 sec)  
  
24060121140141>|
```

Tampak pada screenshot di atas, tabel person telah diisi dengan nama “Indra”. Id menjadi 2 dikarenakan percobaan pemasukan ke 2.

## B. Menggunakan Persisten Object Sebagai Objek Terealisasi

### 1. SerializePerson.java

```
//Nama File: SerializePerson.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//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 o = new
ObjectOutputStream(f);
            o.writeObject(person);
            System.out.println("Selesai menulis objek
person");
            o.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

### 2. Compile dan Run SerializePerson.java

```
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> javac SerializePerson.java
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> java SerializePerson
Selesai menulis objek person
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> |
```

Screenshot di atas menampilkan proses *compile* dan run `SerializePerson.java`.

Setelah berhasil di run, akan terbuat file `person.ser`.

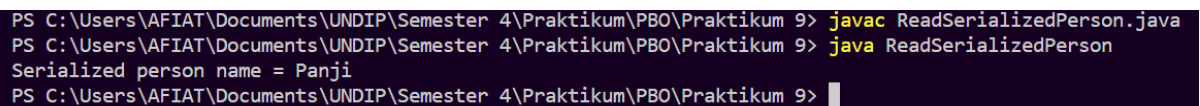
### 3. ReadSerializedPerson.java

```
//Nama File: ReaadSerializedPerson.java 31/05/2023
//Penulis: Muhammad Afiat Yulianto
//NIM: 24060121140141
//LAB: PBO B2
//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 Run ReadSerializedPerson.java



```
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> javac ReadSerializedPerson.java
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> java ReadSerializedPerson
Serialized person name = Panji
PS C:\Users\AFIAT\Documents\UNDIP\Semester 4\Praktikum\PBO\Praktikum 9> |
```

Screenshot di atas menampilkan proses *compile* dan run

ReadSerializedPerson.java. Setelah berhasil di run, Panji akan masuk ke tabel Person.