

# MODUL PRAKTIKUM

## PBO LANJUT

```
    , load from
    requests.get(url)

    checking response.status_code
    if response.status_code != 200:
        print(f"Status: {response.status_code}")
    else:
        print(f"Status: {response.status_code} OK")
        # using BeautifulSoup to parse the response
        soup = BeautifulSoup(response.content, "html.parser")
        # finding Post images in the response
        images = soup.find_all("img")
        # reading images
        for image in images:
            print(image['src'])
```

## KATA PENGANTAR

Dengan mengucap puja dan puji syukur kehadirat Allah SWT, dimana telah memberikan kelimpahan rahmat dan hidayah-Nya, sehingga kami dapat menyusun Modul Pembelajaran Pemrograman Berorientasi Objek Lanjut ini. Harapan kami dengan modul ajar ini dapat memotivasi dan menjadi acuan bagi pembaca yang ingin mempelajari Pemrograman Berorientasi Objek Lanjut.

Bahan ajar ini disusun berdasarkan pengalaman dan berpedoman dari beberapa sumber referensi baik dari dosen, buku, tutorial, website serta karya ilmiah yang mendukung materi pada setiap bab dalam 6 pertemuan. Modul ini berisi teori, uraian contoh studi kasus, penjelasan mendetail sehingga memberikan kemudahan untuk pembaca mempelajarinya. Pada modul Pemrograman Berorientasi Objek Lanjut ini akan membahas tentang Class dan Objek dalam bahasa pemrograman Python, Exception Handling, Private Variabel dan Enkapsulasi, Inheritance, Koneksi PostgreSQL dan Python, PostgreSQL Server dan Tools DBEaver. Besar Harapan kami dengan contoh yang diberikan secara sederhana pembaca akan dapat memahami dan mengerti.

Permintaan maaf kami sampaikan apabila terdapat kesalahan dalam tulisan yang kurang berkenan atas penyampaian materi atau kalimat dalam modul ini, harapan kami kritik dan saran yang membangun dari para pembaca untuk kemajuan modul ini dan memperbaiki kekurangan kedepannya.

Cirebon, Mei 2024

Tim Penyusun

## DAFTAR ISI

KATA PENGANTAR.....	ii
DAFTAR ISI.....	iii
BAB 1 PENDAHULUAN .....	1
A. Latar Belakang .....	1
B. Rumusan Masalah .....	2
C. Tujuan.....	2
BAB 2 CLASS DAN OBJEK.....	3
A. CLASS .....	3
B. OBJEK .....	3
C. Contoh Program .....	4
BAB 3 EXCEPTION HANDLING .....	7
A. Pengertian Exception Handling.....	7
B. Jenis-jenis Exception Handling.....	7
C. Contoh Program .....	8
BAB 4 PRIVATE VARIABLE DAN ENKAPSULASI .....	12
A. Public Access Modifier .....	12
B. Protected Access Modifier .....	13
C. Private Access Modifier .....	13
D. Contoh Program .....	14
BAB 5 INHERITANCE.....	16
A. Konsep Inheritance.....	16
B. Membuat Objek Induk (Parent).....	16
C. Membuat Objek Turunan (Child).....	17
D. Contoh Program .....	18
BAB 6 KONEKSI POSTGRESQL PYTHON .....	19
A. Pengertian.....	19
B. Langkah – langkah Koneksi PostgreSql.....	19
BAB 7 POSTGRESQL SERVER DAN TOOLS DBEAVER .....	21
A. Pengertian.....	21
B. Langkah-langkah.....	21
BAB 8 PENUTUP.....	21
A. Kesimpulan.....	21
B. Saran.....	30
DAFTAR PUSTAKA .....	31
TENTANG PENULIS .....	32

# BAB 1

## PENDAHULUAN

### A. Latar Belakang

Python adalah bahasa pemrograman yang populer dengan dukungan penuh untuk paradigma pemrograman berorientasi objek. Salah satu fitur utamanya adalah kemampuan untuk menggunakan kelas (class) dan objek (object) untuk memodelkan data dan fungsi. Dengan menggunakan konsep class dan objek, pengembang dapat mengorganisir dan mengelola kode mereka dengan lebih terstruktur dan modular.

Selain itu, Python juga memiliki dukungan yang kuat untuk exception handling, yang memungkinkan pengembang untuk menangani kesalahan atau situasi yang tidak terduga saat program dijalankan. Dengan menggunakan blok try-except, pengembang dapat menangkap dan mengelola exception yang terjadi, meningkatkan kehandalan dan ketahanan program.

Python juga mendukung fitur-fitur lanjutan seperti private variabel dan enkapsulasi, serta inheritance. Dengan menggunakan private variabel dan enkapsulasi, pengembang dapat menyembunyikan detail implementasi dari penggunaan luar, sementara inheritance memungkinkan untuk hierarki class yang fleksibel dan penggunaan kembali kode. Hal ini membuat Python menjadi pilihan yang kuat untuk pengembangan berorientasi objek yang kompleks dan skalabel.

Terakhir, untuk mengakses dan mengelola basis data, Python menyediakan koneksi ke PostgreSQL melalui library seperti psycopg2 atau SQLAlchemy. PostgreSQL adalah sistem manajemen basis data relasional open source yang kuat, sedangkan tools seperti DBeaver menyediakan antarmuka grafis untuk mengelola PostgreSQL server dan basis data secara efisien. Dengan demikian, Python memberikan solusi lengkap untuk pengembangan aplikasi yang melibatkan interaksi dengan basis data PostgreSQL.

**B. Rumusan Masalah**

1. Class dan Objek
2. Exception Handling
3. Private Variabel dan Enkapsulasi
4. Inheritance
5. Koneksi PostgreSQL dan Python
6. PostgreSQL Server dan Tools DBEaver

**C. Tujuan**

Dengan terbentuknya modul ini, diharapkan bagi pembaca dapat memahami berbagai macam rumusan masalah yang ada. Serta diharapkan bagi pembaca dapat melakukan praktikum sebagai bentuk latihan dari materi yang telah penulis sampaikan.

## **BAB 2**

### **CLASS DAN OBJEK**

#### **A. CLASS**

Kelas atau class pada python bisa kita katakan sebagai sebuah blueprint (cetakan) dari objek (atau instance) yang ingin kita buat. Kelas adalah cetakannya atau definisinya, sedangkan objek (atau instance) adalah objek nyatanya.

Dari pertemuan kali ini, ada beberapa poin yang bisa kita tarik sebagai kesimpulan:

1. Objek pada python adalah kumpulan dari variabel-variabel (dinamakan atribut) dan kumpulan dari fungsi-fungsi (dinamakan perilaku).
2. Atas definisi itu, maka semua hal di dalam python adalah sebuah Objek.
3. Objek dan Kelas dalam python bermakna sama. Akan tetapi, jika disebutkan dalam konteks terpisah, maka kelas adalah blueprint dan objek adalah variabel nyata.
4. Konstruktor adalah fungsi yang pertama kali dipanggil ketika sebuah objek diinstantiasi.
5. Objek bisa memiliki atribut yang berupa instan dari kelas lainnya.

#### **B. OBJEK**

Atribut adalah semacam identitas atau variabel dari suatu objek, sedangkan perilaku adalah “kemampuan” atau fitur dari objek tersebut. Kita juga bisa definisikan dalam bentuk yang lebih sederhana lagi, yaitu: objek adalah sebuah gabungan dari kumpulan variabel (dinamakan atribut) dan kumpulan fungsi (dinamakan perilaku). Dan atas definisi itu, maka bisa dikatakan bahwa semua hal di dalam python adalah sebuah objek.

Atribut pada OOP merepresentasikan variabel yang dimiliki sebuah objek. Sedangkan perilaku pada OOP merepresentasikan fungsi yang dimiliki sebuah objek.

### C. Contoh Program

## Contoh Class dan Objek Model 1

```
Latihan OOP Versi 1 > Paspor.py > ...
1 #-----Latihan Contoh lain Versi 1 OOP class-----#
2
3 class Paspor:
4     def __init__(self, nama, kwn, ttl, jk, ktr, kode):
5         self.nama = nama
6         self.kewarganegaraan = kwn
7         self.tanggal_lahir = ttl
8         self.jenis_kelamin = jk
9         self.kantor_negara = ktr
10        self.kode_paspor = kode
11
12    def info(self):
13        print(f"Nama : {self.nama}\nKewarganegaraan : {self.kewarganegaraan}
14        \nTempat Tanggal Lahir : {self.tanggal_lahir}\nJenis Kelamin : {self.
15        jenis_kelamin}\nKantor Negara : {self.kantor_negara}\nKode Paspor :
16        {self.kode_paspor}")
17
18 paspor = Paspor("Raka", "Indonesia", "Indonesia, Riau, 14/07/2006", "Laki -
19 Laki", "Kuala Lumpur", "M08362817-KL")
20 paspor.info()
```

## Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_1 & C:/Users/ASUS/AppData/Local/Proxe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Tugas/Pertemuan_1/Latihan OOP Versi 1/Paspor.py"
Nama : Raka
Kewarganegaraan : Indonesia
Tempat Tanggal Lahir : Indoensia, Riau, 14/07/2006
Jenis Kelamin : Laki - Laki
Kantor Negara : Kuala Lumpur
Kode Paspor : M08362817-KL
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_1>
```

## Contoh Class dan Objek Model 2

```
● ● ●  
1 #-----Latihan Contoh Lain Versi 2 OOP class-----#  
2  
3 class LayangLayang:  
4     def __init__(self):  
5         self.diagonal1 = None  
6         self.diagonal2 = None  
7         self.luas = None  
8  
9     def Luas(self, diagonal1, diagonal2):  
10        self.diagonal1 = diagonal1  
11        self.diagonal2 = diagonal2  
12        self.luas = (self.diagonal1 * self.diagonal2) / 2  
13        return self.luas  
14  
15 L = LayangLayang()  
16 diagonal1 = input("masukkan Nilai Diagonal 1 :")  
17 diagonal2 = input("masukkan Nilai Diagonal 2 :")  
18 luas = L.Luas(int(diagonal1), int(diagonal2))  
19  
20 print("Diagonal 1 sebesar :",diagonal1)  
21 print("Diagonal 2 sebesar :",diagonal2)  
22 print("Luas :",luas)
```

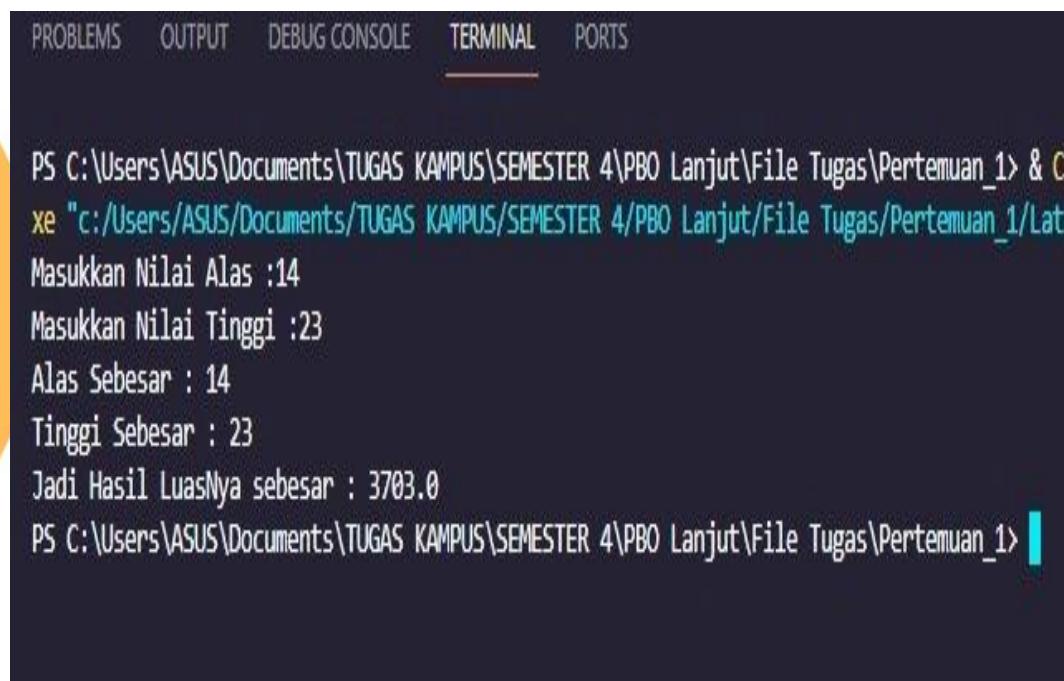
Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_1> & C:/Us  
xe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Tugas/Pertemuan_1/Latihan  
masukkan Nilai Diagonal 1 :56  
masukkan Nilai Diagonal 2 :23  
Diagonal 1 sebesar : 56  
Diagonal 2 sebesar : 23  
Luas : 644.0  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_1>
```

## Contoh Class dan Objek Model 3

```
1 #-----Latihan Contoh Lain Versi 3 OOP class-----#
2
3
4 class PrismaSegitiga:
5     def __init__(self):
6         self._alas = None
7         self._tinggi = None
8
9     @property
10    def alas(self):
11        return self._alas
12
13    @alas.setter
14    def alas(self, value):
15        self._alas = value
16
17    @property
18    def tinggi(self):
19        return self._tinggi
20
21    @tinggi.setter
22    def tinggi(self, value):
23        self._tinggi = value
24
25    def Volume(self):
26        return 0.5 * self._alas * self._tinggi * self._tinggi
27
28 P = PrismaSegitiga()
29 A = input("Masukkan Nilai Alas :")
30 T = input("Masukkan Nilai Tinggi :")
31
32 P.alas = int(A)
33 P.tinggi = int(T)
34
35 V = P.Volume()
36
37 print("Alas Sebesar :", A)
38 print("Tinggi Sebesar :", T)
39 print("Jadi Hasil Luasnya sebesar :", V)
40
```

Output:



The screenshot shows a terminal window with several tabs at the top: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active, indicated by a yellow bar underneath it. The terminal displays the following output:

```
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_1> & C:
xe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Tugas/Pertemuan_1/Lat:
Masukkan Nilai Alas :14
Masukkan Nilai Tinggi :23
Alas Sebesar : 14
Tinggi Sebesar : 23
Jadi Hasil Luasnya sebesar : 3703.0
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_1>
```

## BAB 3

### EXCEPTION HANDLING

#### A. Pengertian Exception Handling

Exception handling adalah suatu mekanisme penanganan flow normal program karena terjadi exception dengan melanjutkan flow ke code block lainnya. Ketika suatu kesalahan atau exception terjadi dalam blok try, program tidak akan langsung dihentikan; sebaliknya, kontrol program dialihkan ke blok except yang sesuai.

#### B. Jenis-jenis Exception Handling

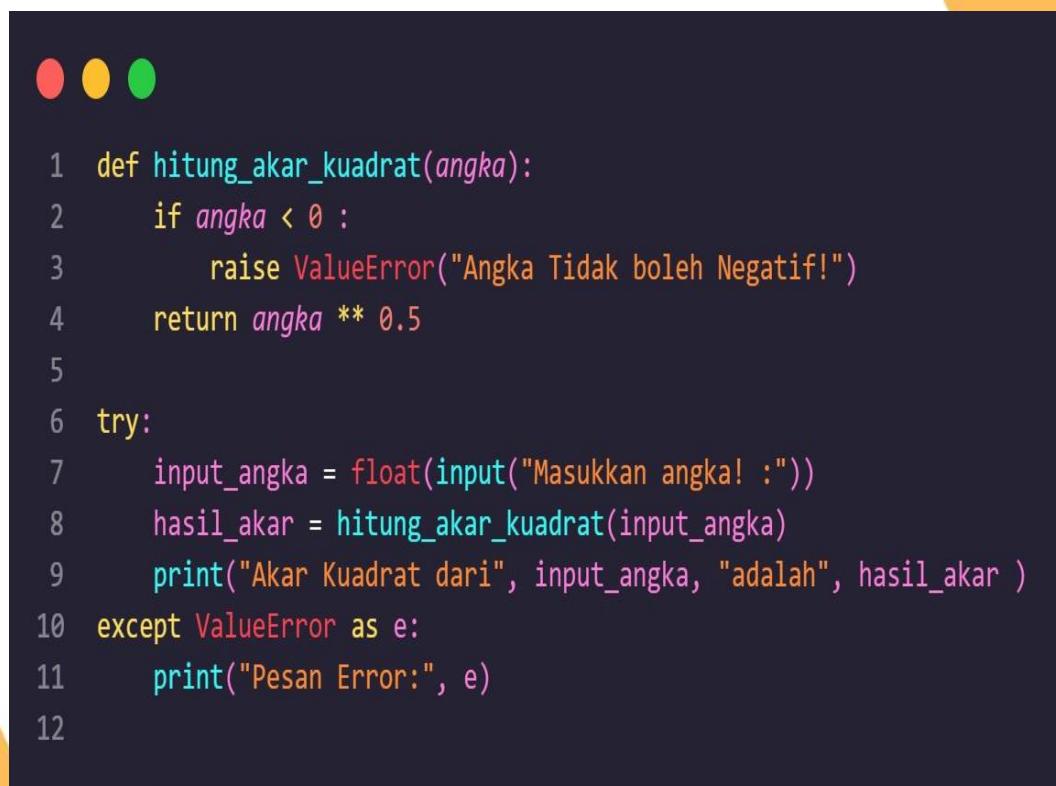
Ada beberapa exception Python bawaan yang dapat dimunculkan ketika terjadi kesalahan selama eksekusi suatu program. Berikut adalah beberapa jenis exception yang paling umum di Python:

1. SyntaxError: Pengecualian ini muncul ketika penerjemah menemukan kesalahan sintaksis dalam kode, seperti kata kunci yang salah eja, titik dua yang hilang, atau tanda kurung yang tidak seimbang.
2. TypeError : Pengecualian ini muncul ketika operasi atau fungsi diterapkan pada objek dengan tipe yang salah, seperti menambahkan string ke integer.
3. NameError : Pengecualian ini muncul ketika nama variabel atau fungsi tidak ditemukan dalam cakupan saat ini.
4. IndexError : Pengecualian ini muncul ketika indeks berada di luar jangkauan untuk daftar, tuple, atau tipe urutan lainnya.
5. KeyError : Pengecualian ini muncul ketika kunci tidak ditemukan dalam kamus.
6. ValueError : Pengecualian ini muncul ketika fungsi atau metode dipanggil dengan argumen atau masukan yang tidak valid, seperti mencoba mengonversi string menjadi bilangan bulat ketika string tidak mewakili bilangan bulat yang valid.
7. AttributeError : Pengecualian ini muncul ketika atribut atau metode tidak ditemukan pada suatu objek, seperti mencoba mengakses atribut yang tidak ada dari instance kelas.

8. IOError : Pengecualian ini muncul ketika operasi I/O, seperti membaca atau menulis file, gagal karena kesalahan input/output.
9. ZeroDivisionError : Pengecualian ini muncul ketika ada upaya untuk membagi angka dengan nol.
10. ImportError : Pengecualian ini muncul ketika pernyataan import gagal menemukan atau memuat modul.

### C. Contoh Program

Contoh Exception Handling 1

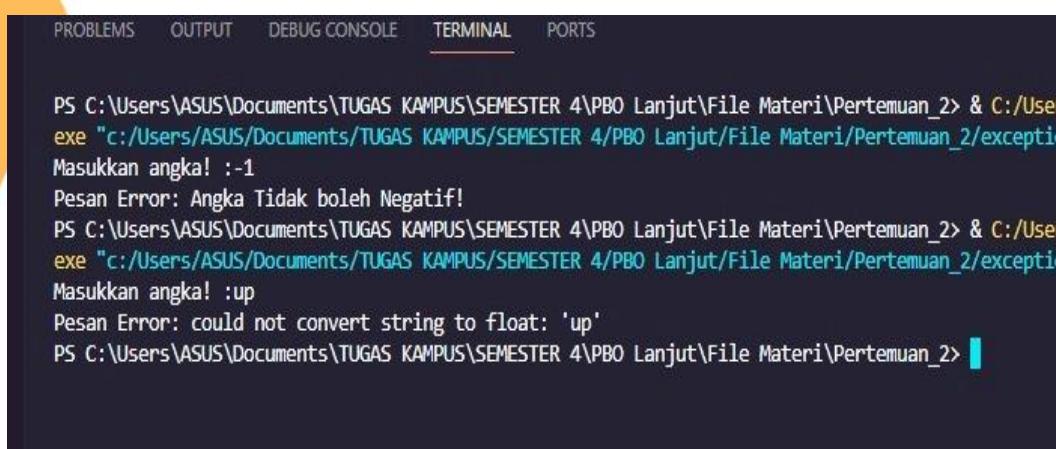


```

1  def hitung_akar_kuadrat(angka):
2      if angka < 0 :
3          raise ValueError("Angka Tidak boleh Negatif!")
4      return angka ** 0.5
5
6 try:
7     input_angka = float(input("Masukkan angka! :"))
8     hasil_akar = hitung_akar_kuadrat(input_angka)
9     print("Akar Kuadrat dari", input_angka, "adalah", hasil_akar )
10 except ValueError as e:
11     print("Peser Error:", e)
12

```

Output:



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/User
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception
Masukkan angka! :-1
Peser Error: Angka Tidak boleh Negatif!
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/User
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception
Masukkan angka! :up
Peser Error: could not convert string to float: 'up'
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2>

```

## Contoh Exception Handling 2

```
● ● ●

1 try:
2     a = [1,2,3]
3     b = int(input("Masukkan Angka indek [0,1,2] ! :"))
4     print (a[b])
5 except IndexError:
6     print("index di luar rentang Array")
7 except Exception as e:
8     print("Terjadi Error :", e)
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception2.py"
Masukkan Angka indek [0,1,2] ! :3
index di luar rentang Array
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2>
```

## Contoh Exception Handling 3

```
● ● ●

1 try:
2     file = open("exception1.py", "r")
3     #Proses Membaca File
4 except FileNotFoundError:
5     print("Maaf, File anda Tidak Ditemukan!")
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception3.py"
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2>
```

#### Contoh Exception Handling 4

```
● ● ●

1 while True:
2     try:
3         input_angka = float(input("Masukkan Angka: "))
4         break #keluar dari loop jika input valid
5     except ValueError:
6         print("Error, Hanya bisa Menginput Angka!")
7 print("angka yang dimasukkan", input_angka)
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut> exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO
Masukkan Angka: r
Error, Hanya bisa Menginput Angka!
Masukkan Angka: P
Error, Hanya bisa Menginput Angka!
Masukkan Angka: 
```

#### Contoh Exception Handling 5

```
● ● ●

1 def ambil_nilai(kamus, kunci):
2     try:
3         nilai = kamus[kunci]
4         print("nilai dari kunci", kunci, "adalah", nilai)
5     except KeyError:
6         print("Error: Kunci", kunci, "Tidak ditemukan dalam Kamus!")
7
8 #contoh pemanggilan fungsi dengan kunci yang tidak ada
9
10 data = {'nama': 'john', 'usia': 30, 'kota': 'Jakarta'}
11
12 kunci = input("Masukkan Kunci [nama, usia, kota]:")
13 ambil_nilai(data, kunci)
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut> File Materi\Pertemuan_2> & C:/User
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception.py
Masukkan Kunci [nama, usia, kota]:ahmad
Error: Kunci ahmad Tidak ditemukan dalam Kamus!
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut> File Materi\Pertemuan_2> 
```

## Contoh Exception Handling 6

```
● ● ●  
1 try:  
2     pilihan = int(input("masukkan angka lebih dari nol:"))  
3     assert pilihan > 0 #cek nilai  
4 except AssertionError:  
5     print("angka tidak boleh nol atau negatif!")
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/Users/  
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception6.py"  
masukkan angka lebih dari nol:0  
Open file in editor (ctrl + click) negatif!  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/Users/  
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception6.py"  
masukkan angka lebih dari nol:-2  
angka tidak boleh nol atau negatif!  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2>
```

## Contoh Exception Handling 7

```
● ● ●  
1 try:  
2     pilihan = int(input("Masukkan angka :"))  
3     assert pilihan > 0 #acuan nilai  
4 except AssertionError:  
5     print("Angka tidak boleh nol atau negatif!")  
6 except ValueError:  
7     print("input harus berupa angka!")
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/Users/  
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception7.py"  
Masukkan angka :-9  
Angka tidak boleh nol atau negatif!  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2> & C:/Users/  
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Pertemuan_2/exception7.py"  
Masukkan angka :P  
input harus berupa angka!  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_2>
```

## BAB 4

### PRIVATE VARIABEL DAN ENKAPSULASI

Access Modifiers (Enkapsulasi) adalah sebuah konsep di dalam pemrograman berorientasi objek di mana kita bisa mengatur hak akses suatu atribut dan fungsi pada sebuah class. Konsep ini juga biasa disebut sebagai enkapsulasi, di mana kita akan mendefinisikan mana atribut atau fungsi yang boleh diakses secara terbuka, mana yang bisa diakses secara terbatas, atau mana yang hanya bisa diakses oleh internal kelas alias privat.

#### A. Public Access Modifier

Variabel atau atribut yang memiliki hak akses publik bisa diakses dari mana saja baik dari luar kelas maupun dari dalam kelas.

```
class Segitiga:  
  
    def __init__(self, alas, tinggi):  
        self.alas = alas  
        self.tinggi = tinggi  
        self.luas = 0.5 * alas * tinggi
```

Pada contoh di atas, kita membuat sebuah kelas dengan nama Segitiga. Kelas tersebut menerima dua buah parameter: yaitu alas dan tinggi.

Kemudian ia juga memiliki 3 buah atribut (alas, tinggi, dan luas) yang mana semuanya memiliki hak akses publik.

## B. Protected Access Modifier

Variabel atau atribut yang memiliki hak akses protected hanya bisa diakses secara terbatas oleh dirinya sendiri (yaitu di dalam internal kelas), dan juga dari kelas turunannya. Mari kita mencoba mempraktikkannya.

Untuk mendefinisikan atribut dengan hak akses protected, kita harus menggunakan prefix underscore ( \_ ) sebelum nama variabel.

```
class Mobil:  
    def __init__(self, merk):  
        self._merk = merk
```

Pada kode program di atas, kita membuat sebuah kelas bernama Mobil. Dan di dalam kelas tersebut, kita mendefinisikan satu buah atribut bernama merk, yang mana hak akses dari atribut tersebut adalah protected karena nama variabelnya diawali oleh underscore () .

## C. Private Access Modifier

Modifier selanjutnya adalah private. Setiap variabel di dalam suatu kelas yang memiliki hak akses private maka ia hanya bisa diakses di dalam kelas tersebut. Tidak bisa diakses dari luar bahkan dari kelas yang mewarisinya.

Untuk membuat sebuah atribut menjadi private, kita harus menambahkan dua buah underscore sebagai prefix nama atribut.

```
class Mobil:  
    def __init__(self, merk):  
        self.__merk = merk
```

Pada kode di atas, kita telah membuat sebuah atribut dengan nama `__merk`. Dan karena nama tersebut diawali dua buah underscore, maka ia tidak bisa diakses kecuali dari dalam kelas `Mobil` saja.

## D. Contoh Program

```
● ● ●  
1  class Persegipanjang:  
2      def __init__(self,panjang,Lebar):  
3          self.__panjang = 0 #private method  
4          self.__lebar = 0  
5          self.__setPanjang(panjang)  
6          self.__setLebar(Lebar)  
7  
8      def getPanjang(self):           #public method  
9          return self.__panjang  
10  
11     def getLebar(self):           #public method  
12         return self.__lebar  
13  
14     def __setPanjang(self,nilai):    #private method  
15         if(nilai<=0):  
16             nilai = 1  
17             self.__panjang = nilai  
18  
19     def __setLebar(self,nilai):      #private method  
20         if(nilai<=0):  
21             nilai=1  
22             self.__lebar = nilai  
  
● ● ●  
1  def Luas(self):  
2      L = self.__panjang * self.__lebar #public methode  
3      return L  
4  
5  def Keliling(self):  
6      K = (2 * self.__panjang)+(2 * self.__lebar) #public method  
7      return K  
8  try:  
9      panjang = int(input("Masukkan Nilai Panjang :"))  
10     lebar = int(input("Masukkan Nilai Lebar :"))  
11  except ValueError:  
12      print("Hanya Bisa memasukkan Angka !")  
13  else:  
14      P = Persegipanjang(panjang,lebar)  
15      L = P.Luas()  
16      K = P.Keliling()  
17      print("panjang : ", P.getPanjang())  
18      print("lebar : ", P.getLebar())  
19      print("Luas persegi panjang : ",L)  
20      print("Keliling persegi panjang: ",K)
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_3> & C:/User
hon39/python.exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Materi/Perten
Masukkan Nilai Panjang :41
Masukkan Nilai Lebar :12
panjang : 41
lebar : 12
Luas persegi panjang : 492
Keliling persegi panjang: 106
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Materi\Pertemuan_3>
```

## BAB 5

# INHERITANCE

### A. Konsep Inheritance

Inheritance, kita bisa membuat satu objek sebagai induk dari objek-objek lainnya. Setiap objek yang menjadi turunan dari objek induk akan memiliki sifat dan perilaku dasar yang sama.

Konsep inheritance adalah konsep dimana sebuah kelas atau objek mewariskan sifat dan perilaku kepada kelas lainnya. Kelas yang menjadi “pemberi waris” dinamakan kelas induk atau kelas basis. Sedangkan kelas yang menjadi “ahli waris” dinamakan sebagai kelas turunan.

### B. Membuat Objek Induk (Parent)

Untuk membuat objek induk pada python, caranya sama dengan cara membuat objek biasa. Karena pada dasarnya, semua objek pada python bisa menjadi objek induk dari objek turunan lainnya.

```
class Orang:  
    def __init__(self, nama, asal):  
        self.nama = nama  
        self.asal = asal  
    def perkenalan(self):  
        print(f'Perkenalkan nama saya {self.nama} dari {self.asal}')
```

Objek di atas sangat sederhana, ia hanya memiliki 2 atribut (nama dan asal) serta memiliki satu buah perilaku yaitu perkenalan().

```
andi = Orang('Andi', 'Surabaya')  
andi.perkenalan()
```

Output:

```
Perkenalkan nama saya Andi dari Surabaya
```

### C. Membuat Objek Turunan (Child)

Langkah selanjutnya adalah membuat objek turunan dari kelas Orang. Sesuai dengan skenario yang telah kita bahas di atas, kita akan membuat dua buah objek baru yaitu objek Pelajar dan Pekerja, yang mana keduanya akan mewarisi objek Orang. Caranya bagaimana? Kita bisa membuat kelas turunan dengan cara mengirimkan kelas induk sebagai parameter saat mendefinisikan kelas.

Perhatikan contoh berikut:

```
class Pelajar (Orang):
    pass

class Pekerja (Orang):
    pass
```

Kita telah membuat dua buah kelas yang keduanya sama-sama memiliki setiap atribut dan fungsi dari kelas Orang. Kita meletakkan perintah pass karena kita hanya ingin melakukan pewarisan apa adanya tanpa menambahkan apa pun lagi. Sehingga kita dapat membuat instance dari kelas pelajar dan pekerjaan, serta memanggil fungsi perkenalan() dengan cara yang benar-benar identik dengan kelas orang:

```
andi = Orang('Andi', 'Surabaya')
andi.perkenalan()

deni = Pelajar('Deni', 'Makassar')
deni.perkenalan()

budi = Pekerja('Budi', 'Pontianak')
budi.perkenalan()
```

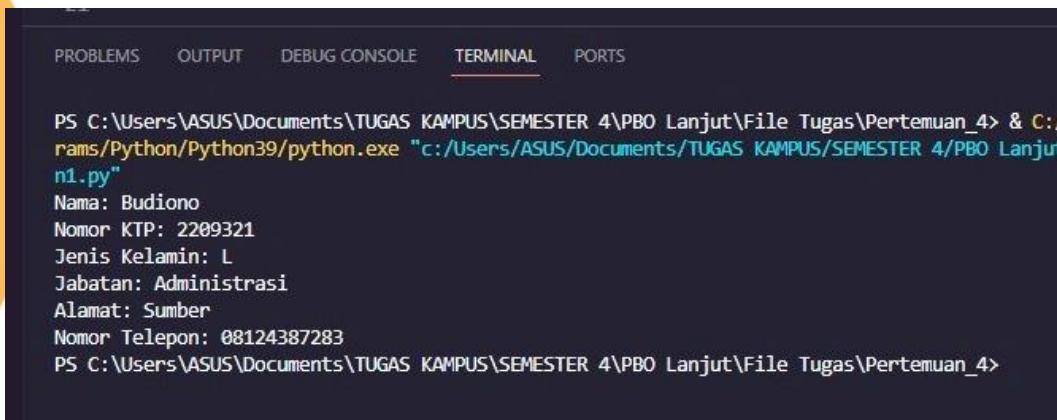
Output:

```
Perkenalkan nama saya Andi dari Surabaya
Perkenalkan nama saya Deni dari Makassar
Perkenalkan nama saya Budi dari Pontianak
```

## D. Contoh Program

```
● ● ●  
1 class Person ():  
2     def __init__ (self, nama, noktp, jk ) :  
3         self.nama = nama  
4         self.noktp= noktp  
5         self.jk = jk  
6  
7 class Dokter (Person):  
8     def __init__ (self, nama, noktp, jk, spesialis, alamat, no_tlp) :  
9         super().__init__(nama, noktp, jk)  
10        self.nama = nama  
11        self.spesialis = spesialis  
12        self.alamat = alamat  
13        self.no_tlp = no_tlp  
14  
15 class Perawat(Person):  
16     def __init__(self, nama, noktp, jk, spesialis, alamat, no_tlp):  
17         super().__init__(nama, noktp, jk)  
18         self.spesialis = spesialis  
19         self.alamat = alamat  
20         self.no_tlp = no_tlp  
● ● ●  
1 class Karyawan(Person):  
2     def __init__(self, nama, noktp, jk, jabatan, alamat, no_tlp):  
3         super().__init__(nama, noktp, jk)  
4         self.jabatan = jabatan  
5         self.alamat = alamat  
6         self.no_tlp = no_tlp  
7  
8  
9 karyawan_1 = Karyawan("Budiono", "2209321", "L", "Administrasi", "Sumber", "08124387283")  
10 print("Nama:", karyawan_1.nama)  
11 print("Nomor KTP:", karyawan_1.noktp)  
12 print("Jenis Kelamin:", karyawan_1.jk)  
13 print("Jabatan:", karyawan_1.jabatan)  
14 print("Alamat:", karyawan_1.alamat)  
15 print("Nomor Telepon:", karyawan_1.no_tlp)
```

Output:



The screenshot shows a terminal window with the following interface elements at the top:

- PROBLEMS
- OUTPUT
- DEBUG CONSOLE
- TERMINAL** (underlined)
- PORTS

The terminal window displays the following text:

```
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_4> & C:/  
rams/Python/Python39/python.exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/  
n1.py"  
Nama: Budiono  
Nomor KTP: 2209321  
Jenis Kelamin: L  
Jabatan: Administrasi  
Alamat: Sumber  
Nomor Telepon: 08124387283  
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_4>
```

## BAB 6

### KONEKSI POSTGRESQL DAN PYTHON

#### A. Pengertian

PostgreSQL adalah sistem database yang didistribusikan secara bebas sesuai dengan perjanjian lisensi BSD. Software ini merupakan salah satu database yang paling banyak digunakan saat ini, selain MySQL dan Oracle.

PostgreSQL menyediakan fitur-fitur yang berguna untuk replikasi database. Fitur-fitur yang disediakan oleh PostgreSQL antara lain DB Mirror, PGPool, Slony, PGCluster, dan lain-lain.

PostgreSQL adalah sebuah sistem basis data yang disebarluaskan bebas menurut Perjanjian lisensi BSD. Database tersebut bersifat multiplatform yang dapat digunakan pada sistem operasi MS Windows, keluarga Linux, keluarga 850, MacOS dan Solaris. Installer PostgreSQL dapat diunduh di link berikut : <https://www.postgresql.org/download/>.

Perancang Basis Data untuk PostgreSQL adalah alat kasus yang mudah dengan antarmuka grafis intuitif yang memungkinkan Anda membangun struktur basis data yang jelas dan efektif secara visual, melihat gambar (diagram) lengkap yang mewakili semua tabel, referensi di antara mereka, tampilan, prosedur tersimpan dan objek lain. Kemudian Anda dapat dengan mudah membuat database fisik di server, memodifikasinya sesuai dengan perubahan yang anda buat pada diagram menggunakan pernyataan mengubah cepat.

#### B. Langkah – langkah Koneksi PostgreSql

Create database, super user dan grant access database ke super user yang telah dibuat

```
SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:
`psql (15.6)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.

postgres=# CREATE DATABASE pegawai;
CREATE DATABASE
postgres=# CREATE USER putra123 WITH SUPERUSER PASSWORD '123';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE pegawai TO putra123;
GRANT
postgres=#

```

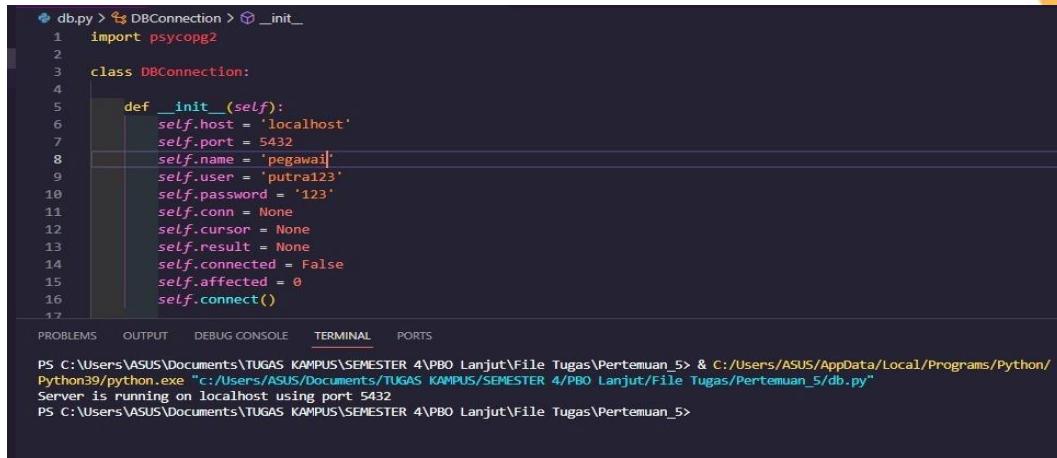
## Create tabel dan menambah data pada tabel

```
SQL Shell (psql)
Username [postgres]: putra123
Password for user putra123:
psql (15.6)
WARNING: Console code page (437) differs from Windows code page (1252)
          8-bit characters might not work correctly. See psql reference
          page "Notes for Windows users" for details.
Type "help" for help.

pegawai=# create table kepegawaian(
pegawai(# id serial primary key not null,
pegawai(# nip varchar(10) unique not null,
pegawai(# nama varchar(50) not null,
pegawai(# jenis_kelamin varchar(1) not null,
pegawai(# posisi varchar(10) not null);
CREATE TABLE
pegawai=# insert into kepegawaian(nip,nama,jenis_kelamin,posisi)values('180915','Budi','L','Marketing');
INSERT 0 1
pegawai=# select * from kepegawaian
pegawai# select * from kepegawaian;
ERROR: syntax error at or near "select"
LINE 2: select * from kepegawaian;
^
pegawai=# select * from kepegawaian;
 id | nip | nama | jenis_kelamin | posisi
----+---+-----+-----+-----
 1 | 180915 | Budi | L | Marketing
(1 row)

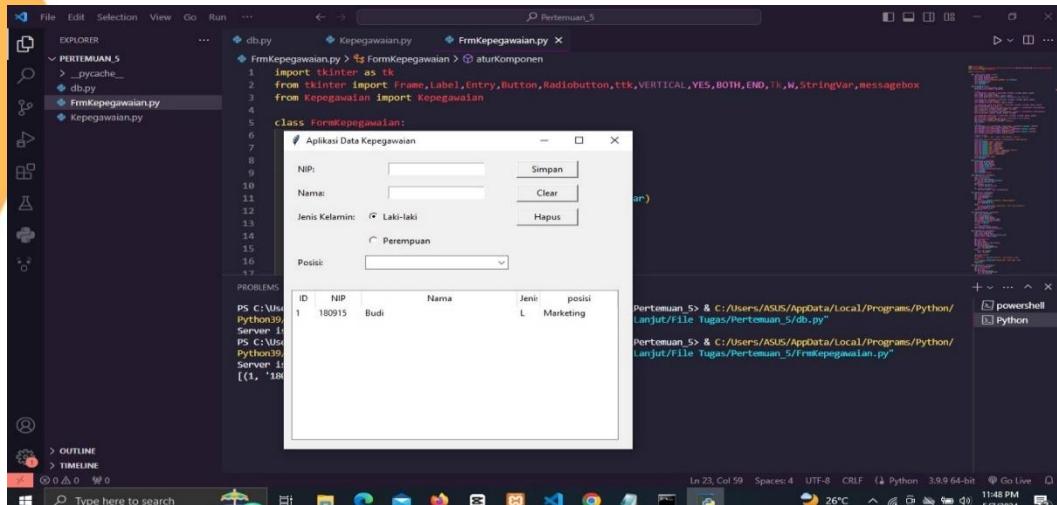
pegawai=#
-
```

Koneksikan dengan baris kode konektor program kita sesuai dengan nama database, port, user, dan password yang sebelumnya telah dibuat.



```
db.py > DBConnection > __init__
1 import psycopg2
2
3 class DBConnection:
4
5     def __init__(self):
6         self.host = 'localhost'
7         self.port = 5432
8         self.name = 'pegawai'
9         self.user = 'putra123'
10        self.password = '123'
11        self.conn = None
12        self.cursor = None
13        self.result = None
14        self.connected = False
15        self.affected = 0
16        self.connect()
17
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_5 & C:/Users/ASUS/AppData/Local/Programs/Python/Python39/python.exe ".\db.py"
Python: warning: This is a pre-3.6 'print' call which formatted its arguments with C-style printf-style conversion specifications.
Server is running on localhost using port 5432
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_5>
```

Lalu coba jalankan dengan program kita yg sebelumnya sudah terkoneksi dengan database yg dibuat di PostgreSQL.



## BAB 7

### POSTGRESQL SERVER DAN TOOLS DBEAVER

#### A. Pengertian

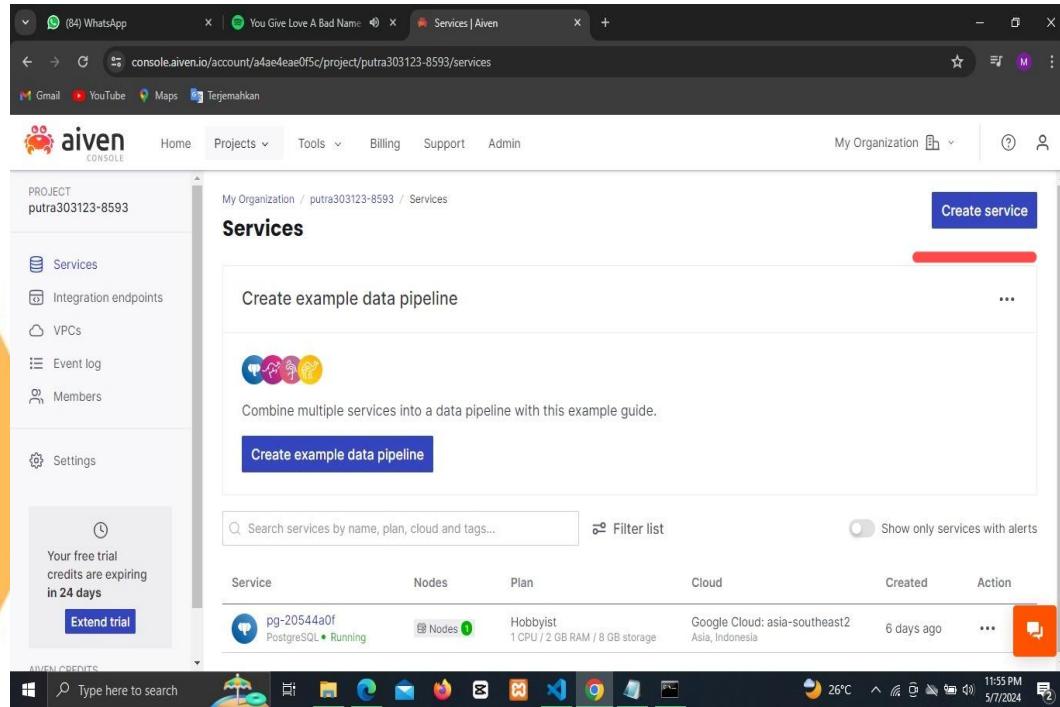
Aiven adalah perusahaan yang menawarkan platform infrastruktur data terbaik kepada dunia. Aiven menawarkan elemen infrastruktur data sebagai layanan. Aiven tidak menjual lisensi, dan tidak memiliki server - Aiven mengelola infrastruktur data masyarakat untuk server tersebut.

DBeaver adalah perangkat lunak yang bertindak sebagai alat basis data universal ditujukan untuk pengembang dan administrator database.

DBeaver adalah aplikasi untuk manajemen database dengan mode GUI (graphical). DBeaver sendiri selain menyediakan versi berbayar, juga menyediakan Community Edition yang open source.

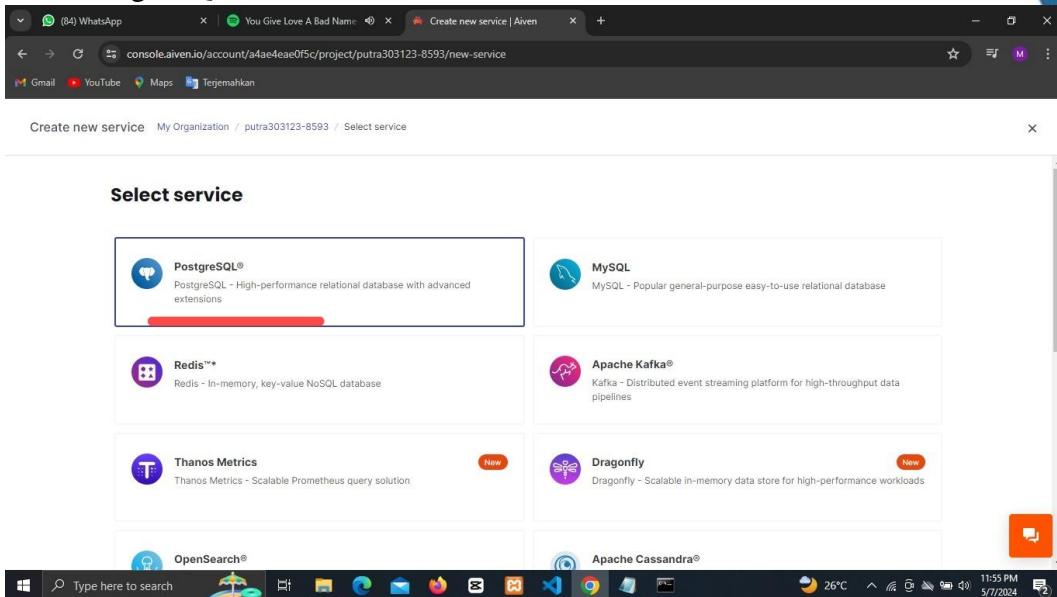
#### B. Langkah-langkah

PostgreSQL server. Create service di web aiven

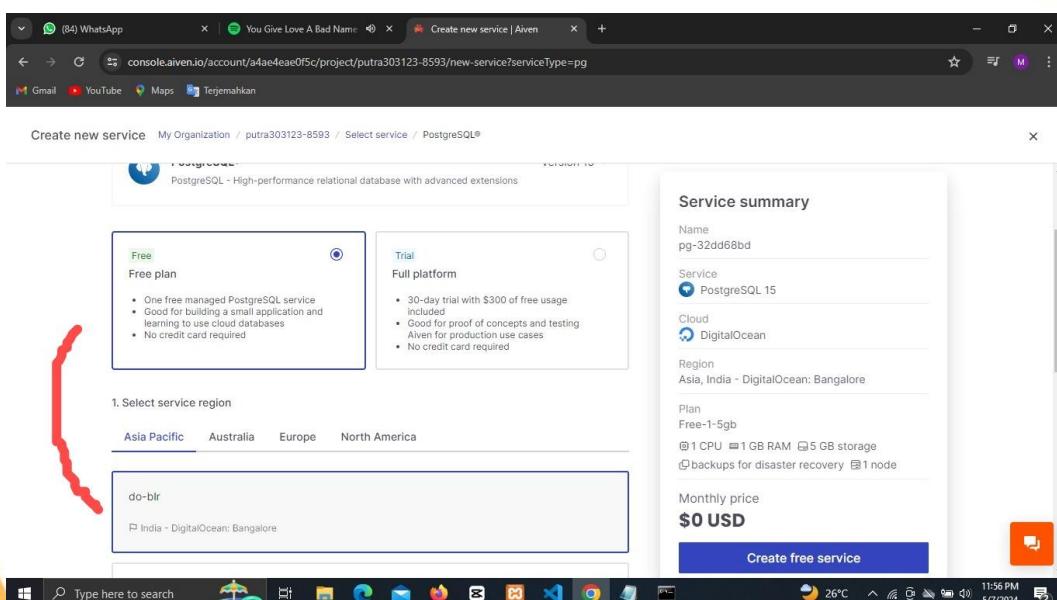


The screenshot shows the Aiven Console interface. The top navigation bar includes links for Home, Projects, Tools, Billing, Support, Admin, and My Organization. On the left, a sidebar lists PROJECT putra303123-8593 with options for Services, Integration endpoints, VPCs, Event log, Members, and Settings. A message indicates a free trial is expiring in 24 days. The main content area is titled 'Services' and features a 'Create example data pipeline' section with a button to 'Create example data pipeline'. Below this is a search bar and a 'Filter list' button. A table lists existing services: pg-20544a0f (PostgreSQL, Running, Hobbyist plan, Google Cloud: asia-southeast2, 6 days ago). The bottom of the screen shows a Windows taskbar with various icons and system status.

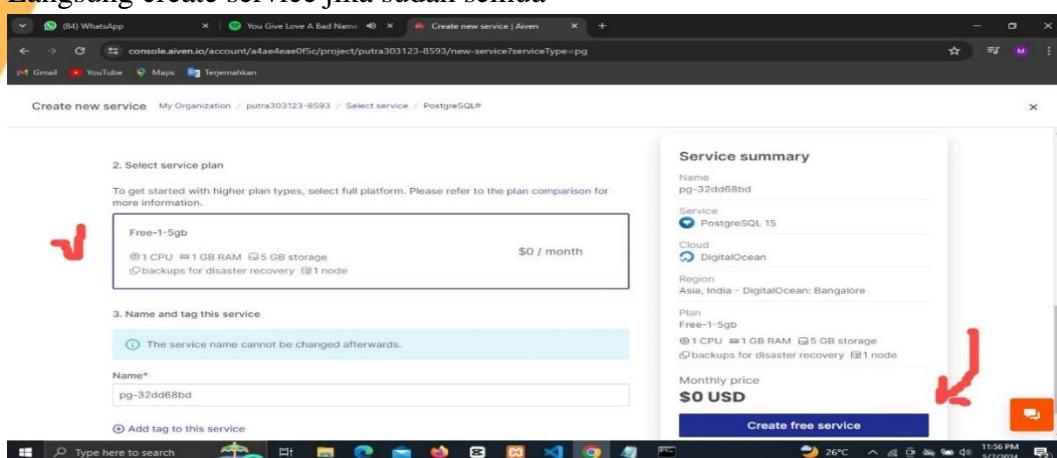
## Pilih PostgreSQL



Pilih paket gratis, lalu pilih region



Langsung create service jika sudah semua



Jika sudah maka service yg dibuat akan muncul di bawah

The screenshot shows the Aiven Console interface. On the left, there's a sidebar with project details and service management options. A prominent message says "Your free trial credits are expiring in 24 days" with a blue "Extend trial" button. Below this, there's a "Create example data pipeline" section with a button. The main area is titled "Services" and lists a single service: "pg-20544a0f" (PostgreSQL). The service details include "Nodes: 1", "Plan: Hobbyist", "Cloud: Google Cloud: asia-southeast2 Asia, Indonesia", and "Created: 6 days ago". A red arrow points from the sidebar message towards this service list.

Klik service yang sudah dibuat, maka isinya akan seperti ini. isinya terkait informasi koneksi yang nantinya akan dihubungkan dengan dbeaver

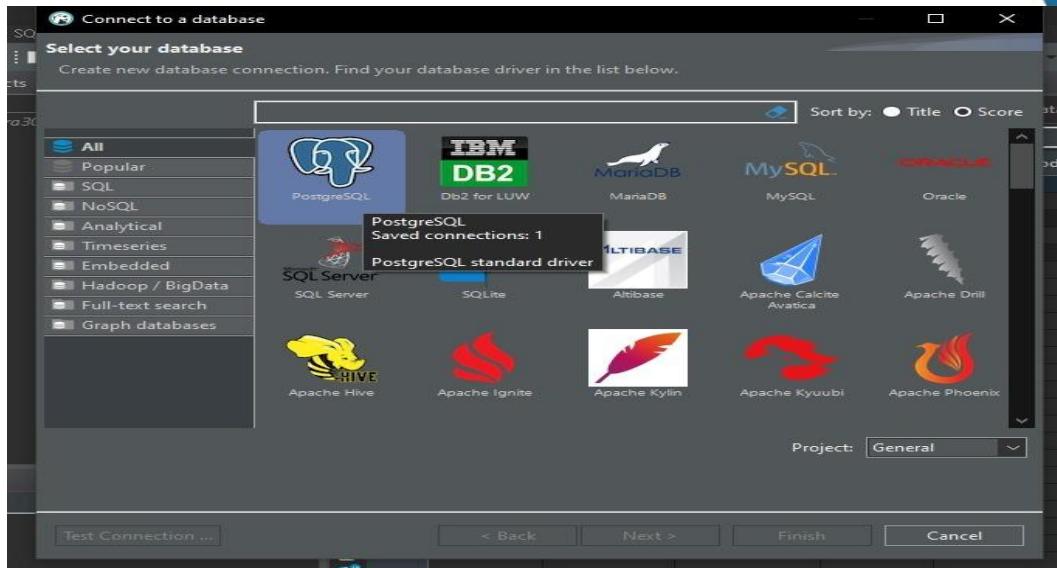
This screenshot shows the "Overview" page for the "pg-20544a0f" service. The sidebar has links like "Back to project", "Overview", "Integrations", "Metrics", "AI insights", "Logs", "Query statistics", "Current queries", "Users", "Databases", "Pools", and "Backups". The main content displays various connection parameters in a table:

Service URI	postgres://CLICK_TO REVEAL_PASSWORD@pg-20544a0f-putra303123-8593.i.aivencloud.com:27691/defaultdb?sslmode=require
Database name	defaultdb
Host	pg-20544a0f-putra303123-8593.i.aivencloud.com
Port	27691
User	avnadmin
Password	*****
SSL mode	require
CA certificate	Show

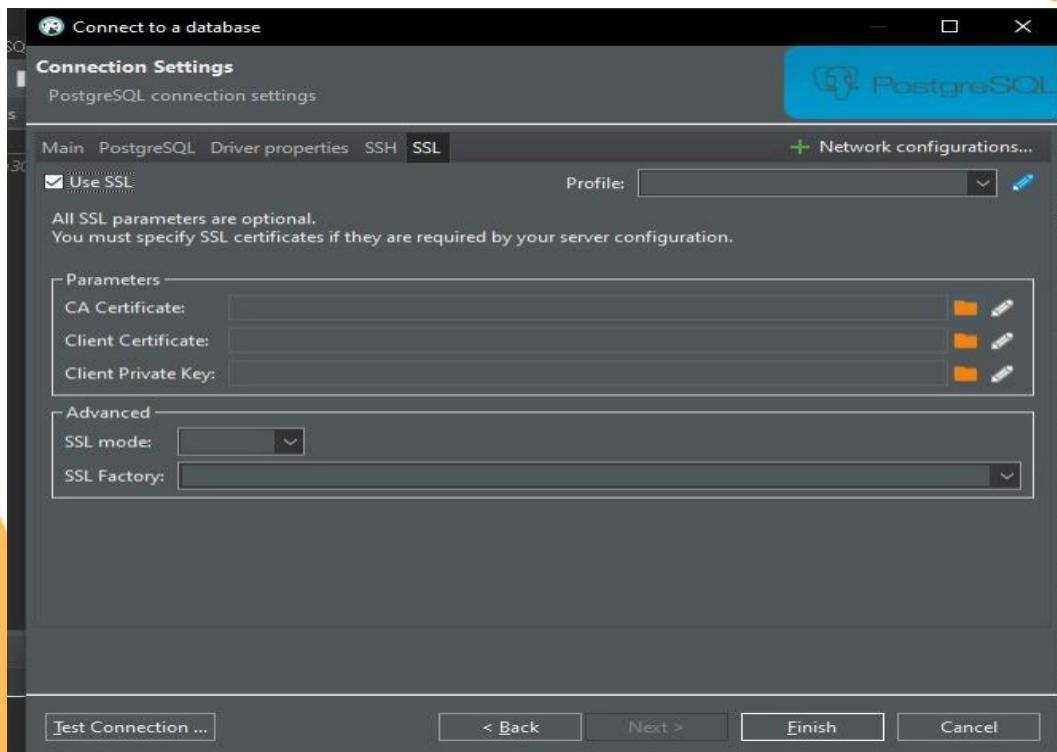
Buka aplikasi Dbeaver, lalu klik di pojok kiri ke database connection

This screenshot shows the DBeaver 24.0.3 application window. The toolbar at the top includes "File", "Edit", "Navigate", "Search", "SQL Editor", "Database", and "Window". Below the toolbar, there are several icons. A red arrow points to the "New Database Connection" button, which is located at the bottom left of the toolbar. The status bar at the bottom right shows the date and time as "12:03 AM 5/8/2024".

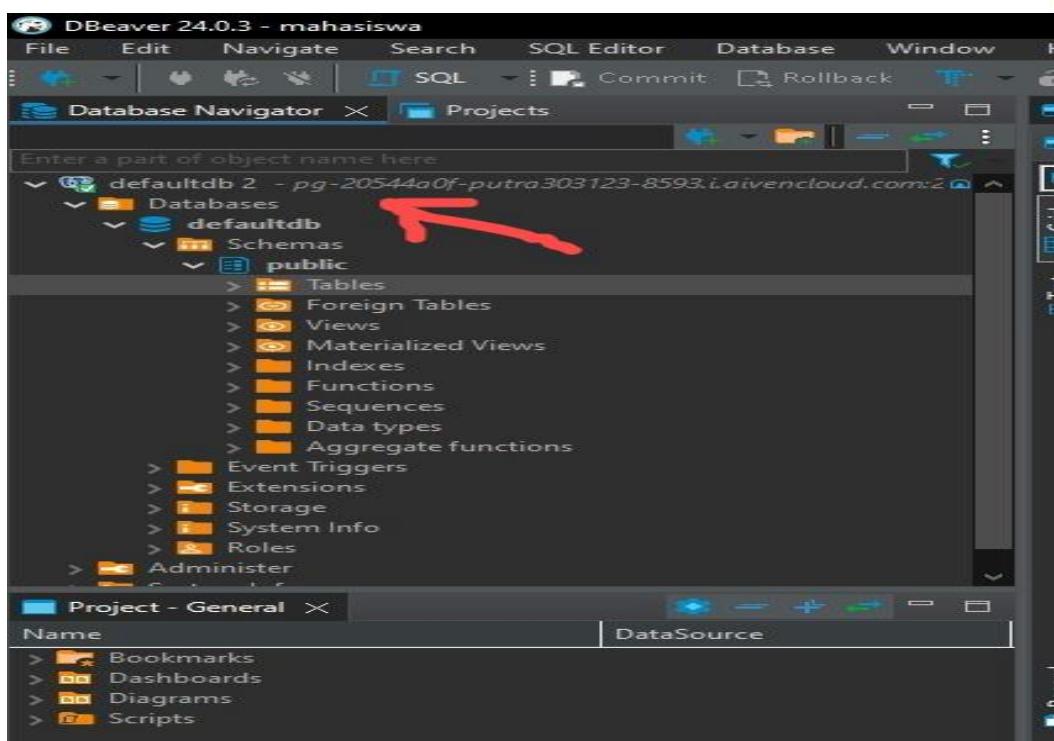
Pilih postgres lalu next



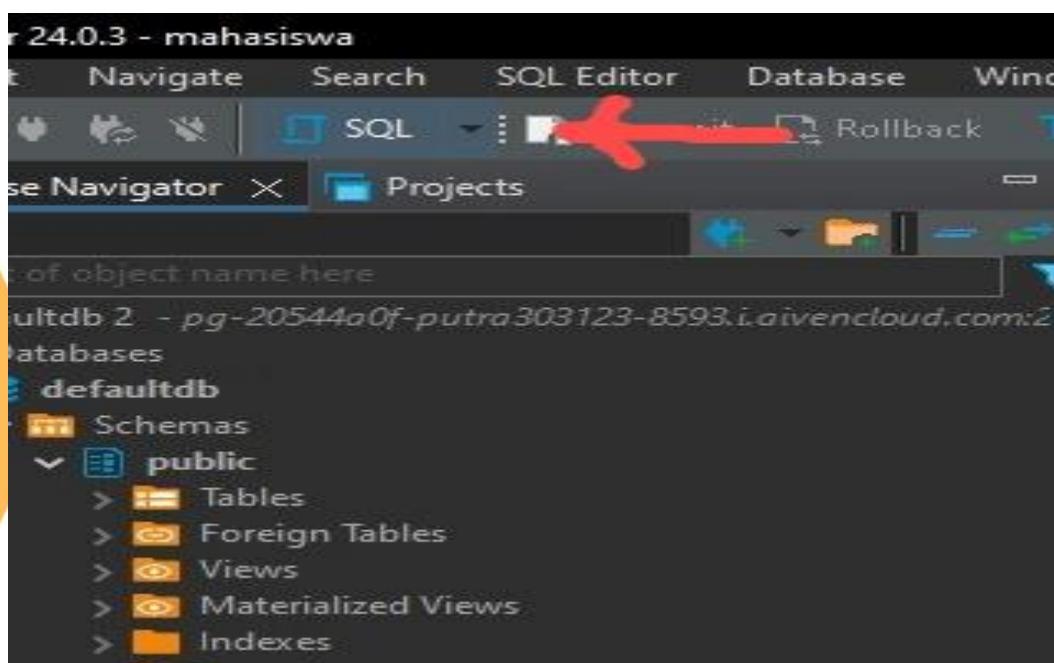
Jika sudah, maka pergi ke menu SSL, lalu masukan file CA certificate yang telah di download di connection infromation service Aiven



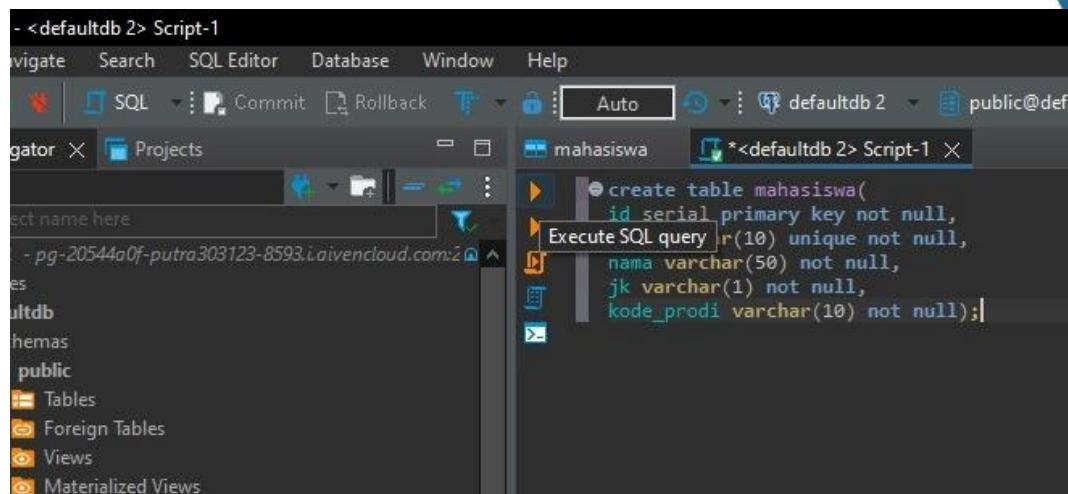
Jika sudah, maka akan muncul databasenya, ikuti saja path file databasenya sampai di file tables



Jika ingin membuat tabel database, pilih sql di menu atas untuk menjalankan script pembuatan sql secara kode

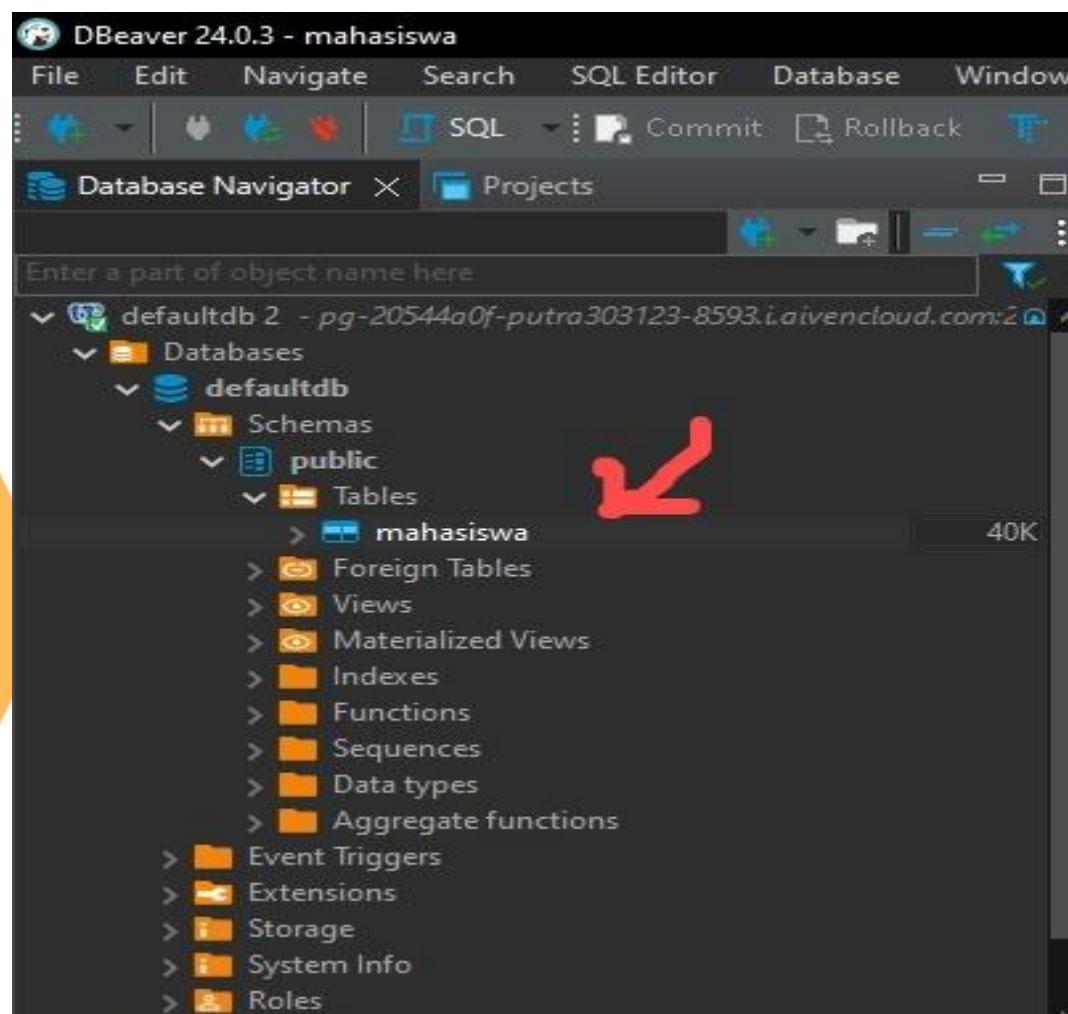


Untuk kode pembuatan tabel bisa dilihat pada gambar, jika script telah selesai, maka klik tombol run atau execute sql query

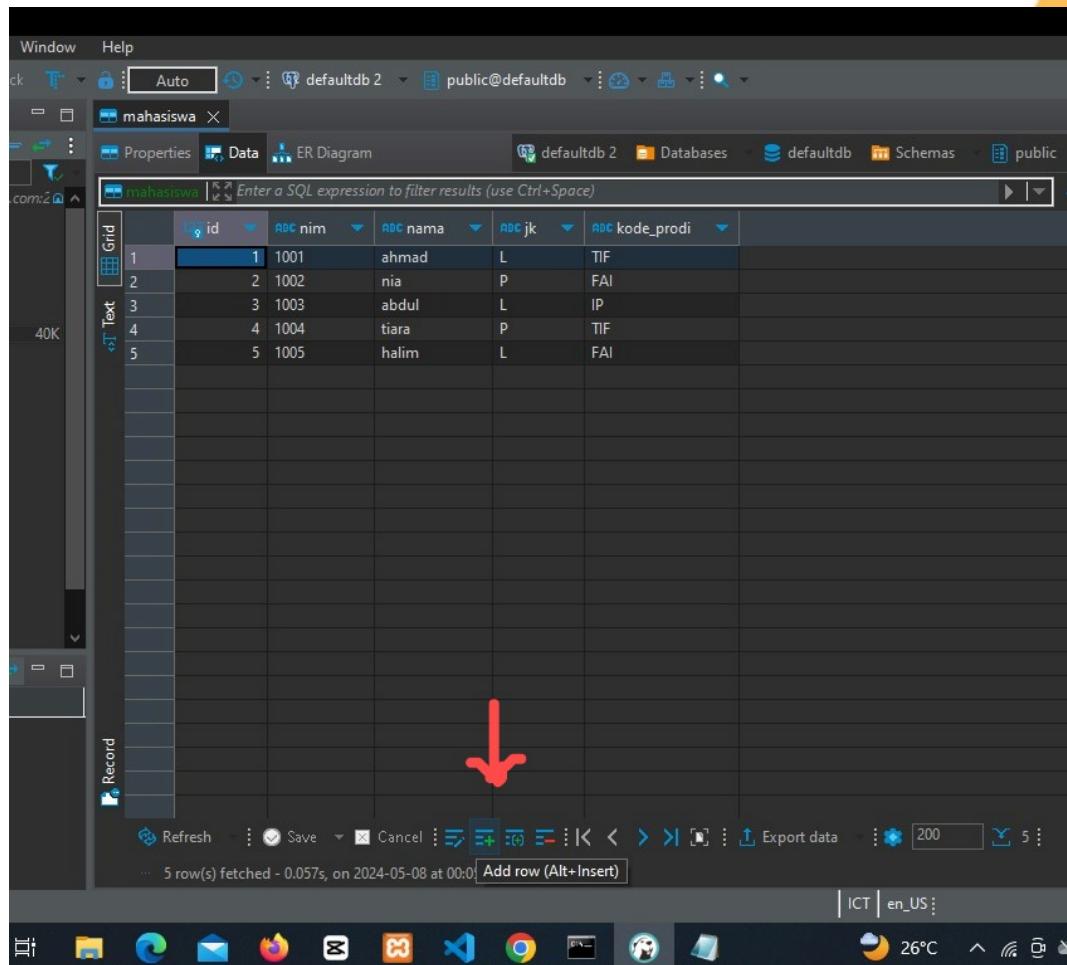


```
- <defaultdb 2> Script-1
File   Search   SQL Editor   Database   Window   Help
SQL   Commit   Rollback   Auto   defaultdb 2   public@def
Database Navigator X   Projects X
mahasiswa * <defaultdb 2> Script-1 X
SELECT name here
-- pg-20544a0f-putra303123-8593.i.aivencloud.com:2
es
ultdb
hemas
public
Tables
Foreign Tables
Views
Materialized Views
CREATE TABLE mahasiswa(
    id SERIAL PRIMARY KEY NOT NULL,
    r(10) UNIQUE NOT NULL,
    nama VARCHAR(50) NOT NULL,
    jk VARCHAR(1) NOT NULL,
    kode_prodi VARCHAR(10) NOT NULL);
```

Jika sudah berhasil membuat tabel dengan script yang dituliskan di SQL, lalu refresh Dbeaver dengan klik tombol refresh di bagian file yang berada di pojok kiri atas. lalu tabel yang baru dibuat akan muncul. jika sudah, klik kanan pada tabel yang sudah dibuat, lalu pilih view table



Jika sudah, lalu klik Data di atas table yang sudah dibuat. lalu tabel akan muncul dan siap di menambahkan data pada tabel. caranya adalah klik tombol add row di bawah untuk menambah baris tabel. lalu anda bisa mengedit dan menambah data pada baris tabel yang telah dibuat. jika sudah menambah maupung edit data, bisa langsung klik save di sebalah kiri tombol add row.



Lalu pada program python kita, seperti bisa untuk koneksi databasenya, sesuaikan host, port, nama database, user, password, dan jangan lupa tambahkan file CA certificate dari service Aiven (ca.pem)

```
db.py > DBConnection > __init__
4
5     def __init__(self):
6         self.host = 'pg-20544a0f-putra303123-8593.i.aivencloud.com'
7         self.port = 27691
8         self.name = 'defaultdb' # nama database
9         self.user = 'avnadmin'
10        self.password = 'AVNS_-rkt5hRKw5c4Vs4npWz'
11        self.ca_file = 'ca.pem' # Path to your ca.pem file
12        self.connection_timeout = 20
13        self.conn = None
14        self.cursor = None
15        self.result = None
16        self.connected = False
17        self.affected = 0
18        self.connect()
19
20    @property
21    def connection_status(self):
22        return self.connected
23
24    def connect(self):
25        try:
26            self.conn = psycopg2.connect(host=self.host,
27                                         port=self.port,
28                                         database=self.name,
29                                         user=self.user,
30                                         password=self.password,
31                                         sslmode='require', # To enforce SSL
32                                         sslrootcert=self.ca_file,
33                                         connect_timeout=self.connection_timeout) # Connection timeout)
34
35            self.cursor = self.conn.cursor()
36            self.connected = True
```

Jika sudah melakukan koneksi antara database kita di Dbeaver dan program python kita, lalu kita jalankan program python kita. maka data yang tambil sudah memuat data dari database dbeaver

```
db.py FrmMahasiswa.py >...  
FrmMahasiswa.py >... Aplikasi Data Mahasiswa  
1 import t Aplikasi Data Mahasiswa  
2 from tki  
3 from Ma  
4  
5 class F  
6  
7 def  
8  
9  
10  
11  
12  
13  
14  
15  
16 def  
17  
PROBLEMS OUTPUT  
PS C:\Users\ASUS\PycharmProjects\PBO_Lanjut\File_Tugas\Pertemuan_6\FrmMa  
hasiswa.py  
Server is running  
[(1, '1001', 'ahmad', 'L', 'TIF'), (2, '1002', 'nia', 'P', 'FAI'), (3, '1003', 'abdul', 'L', 'IP'), (4, '1004', 'tiara', 'P', 'TIF'), (5, '1005', 'halim', 'L', 'FAI')]  
CAL, YES, BOTH, END, Tk, W, StringVar, messagebox
```

## BAB 8

### PENUTUP

#### A. Kesimpulan

Pemahaman konsep-konsep dasar seperti class dan objek, exception handling, private variabel, enkapsulasi, dan inheritance sangat penting dalam pengembangan perangkat lunak berorientasi objek. Dengan menggunakan class dan objek, kode dapat diorganisir dengan lebih terstruktur, meningkatkan keterbacaan dan modularitas. Exception handling memungkinkan program untuk mengatasi situasi yang tidak terduga atau kesalahan saat dijalankan, yang pada gilirannya meningkatkan keandalan dan kestabilan aplikasi.

Selain itu, private variabel dan enkapsulasi membantu dalam menyembunyikan rincian implementasi dari penggunaan luar, memastikan bahwa class hanya dapat diakses melalui metode yang ditentukan, dan mengelola kompleksitas kode dengan lebih baik. Dalam hierarki class, inheritance memungkinkan untuk memperluas fungsionalitas class yang sudah ada dan mengorganisir kode secara hierarkis, mempromosikan kode yang dapat digunakan kembali.

Koneksi antara Python dan PostgreSQL memungkinkan interaksi yang lancar antara kedua platform, memungkinkan pengelolaan data yang efisien dari aplikasi Python ke database PostgreSQL. Tools seperti DBeaver memberikan antarmuka grafis yang intuitif untuk mengelola basis data PostgreSQL, menyederhanakan proses pengelolaan dan administrasi database. Dengan menggabungkan pemahaman tentang konsep-konsep ini, pengembang dapat menciptakan aplikasi yang lebih kuat, terstruktur, dan mudah dielola dengan menggunakan Python dan PostgreSQL.

## **B. Saran**

Setelah memahami konsep-konsep dasar tersebut, diharapkan kepada pembaca untuk mempraktikkannya dalam proyek-proyek kecil. Mulailah dengan membuat program sederhana yang mengimplementasikan class, exception handling, private variabel, dan inheritance lalu lanjut ke PostgreSQL Server dan Tools DBEaver. Ini akan membantu memperdalam pemahaman pembaca dan memberikan pengalaman langsung dalam penggunaan konsep-konsep tersebut.

## DAFTAR PUSTAKA

### Website:

<https://jagongoding.com/python/menengah/oop/kelas-dan-objek/#kelas-pada-python>

[https://linux.die.net/diveintopython/html/getting\\_to\\_know\\_python/everything\\_is\\_an\\_object.html#d0e4665](https://linux.die.net/diveintopython/html/getting_to_know_python/everything_is_an_object.html#d0e4665)

<https://www.programiz.com/python-programming/class>

<https://www.geeksforgeeks.org/python-exception-handling/>

<https://jagongoding.com/python/menengah/oop/access-modifiers/>

<https://jagongoding.com/python/menengah/oop/pewarisan/>

[https://aiven-io.translate.goog/blog/what-is-aiven?\\_x\\_tr\\_sl=en&\\_x\\_tr\\_tl=id&\\_x\\_tr\\_hl=id&\\_x\\_tr\\_pto=tc](https://aiven-io.translate.goog/blog/what-is-aiven?_x_tr_sl=en&_x_tr_tl=id&_x_tr_hl=id&_x_tr_pto=tc)

[https://github.com/Antares023/PBO\\_Lanjut](https://github.com/Antares023/PBO_Lanjut)

<https://github.com/muhhafidaridwan01/PEMROGRAMAN>

<https://github.com/MptrJ1807/PutraRizki-PBO-Lanjut.git>

### Jurnal:

Faisal, M. Reza. Seri Belajar ASP. NET: ASP. NET Core MVC & PostgreSQL dengan Visual Studio Code. M Reza Faisal, 2017.

Nasir, Muhamad, and Novia Natasya. "SISTEM MONITORING AKUARIUM BERBASIS MIKROKONTROLER DAN DJANGO WEB FRAMEWORK." Elektrika Borneo 6.1 (2020): 25-28.

Orru, Matteo, et al. "How do Python programs use inheritance? a replication study." 2015 Asia-Pacific Software Engineering Conference (APSEC). IEEE, 2015.

Rahardja, Untung. "Meningkatkan kualitas sumber daya manusia dengan sistem pengembangan fundamental agile." ADI Bisnis Digital Interdisiplin Jurnal 3.1 (2022): 63-68.

Stiawan, Heru, et al. "Model Visualisasi Informasi Dashboard Pada Pemetaan Tanaman Obat Dan Langka Kabupaten Kediri Menggunakan Microsoft Power Bi." Jurnal Informatika Teknologi dan Sains (Jinteks) 4.4 (2022): 366-371.

## TENTANG PENULIS



NIM : 220511113  
Nama : Muhammad Ilham Ramdhani  
Kelas : TI22C  
Prodi : Teknik Informatika  
Fakultas : Teknik  
Instansi : Universitas Muhammadiyah Cirebon  
GitHub : [https://github.com/Antares023/PBO\\_Lanjut](https://github.com/Antares023/PBO_Lanjut)



NIM : 220511080  
Nama : M Putra Rizky Julianto  
Kelas : TI22C  
Prodi : Teknik Informatika  
Fakultas : Teknik  
Instansi : Universitas Muhammadiyah Cirebon  
GitHub : <https://github.com/MptrJ1807/PutraRizki-PBO-Lanjut.git>



NIM : 220511118  
Nama : Muhammad Hafid Aridwan  
Kelas : TI22C  
Prodi : Teknik Informatika  
Fakultas : Teknik  
Instansi : Universitas Muhammadiyah Cirebon  
GitHub : <https://github.com/muhhafidaridwan01/PEMROGRAMAN>

>>



# PIBO LANJUT

