


Nama: Jonathan Bob Dylan Mongisidi NIM: 064102400013	 Praktikum Statistika	MODUL 2 Nama Dosen: Dr. Dedy Sugiarto, S.Si, M.Kom
Hari/Tanggal: Senin, 10 Februari 2024		Nama Asisten Laboratorium: 1. Michael Briant (064002300004) 2. Monica Sicilia Simanjuntak (065002300030)

Tipe Data, Filter Data & Koneksi ke Database

1. Teori Singkat

Python memiliki beberapa tipe data dasar, di antaranya:

- Integer (int): Representasi bilangan bulat, misalnya: 5, -3, 100.
- Float (float): Representasi bilangan pecahan, misalnya: 3.14, 2.718.
- String (str): Urutan karakter, misalnya: 'hello', "world".
- Boolean (bool): Representasi nilai kebenaran, yaitu True atau False.
- List (list): Kumpulan elemen yang terurut dan dapat diubah, misalnya: [1, 2, 3, 4], ['apple', 'banana', 'cherry'].
- Tuple (tuple): Kumpulan elemen yang terurut dan tidak dapat diubah, misalnya: (1, 2, 3), ('red', 'green', 'blue').
- Dictionary (dict): Kumpulan pasangan kunci-nilai yang tidak terurut, misalnya: {'name': 'John', 'age': 30}.
- Set (set): Kumpulan elemen yang unik dan tidak terurut, misalnya: {1, 2, 3, 4}.

Filter Data dalam Python:

Untuk melakukan filter data dalam Python, Anda dapat menggunakan berbagai cara, tergantung pada struktur data yang Anda gunakan. Dalam konteks DataFrame, seperti yang digunakan dalam Pandas, Anda dapat menggunakan metode `query()` atau pengindeksan boolean.

2. Alat dan Bahan

Hardware : Laptop/PC

Software : R Studio

3. Elemen Kompetensi

Terdapat beberapa tipe data di Jupyter antara lain vektor, matriks dan data frame. Cantumkan setiap output yang dihasilkan dari console Jupyter, ke kolom yang sudah disediakan.

a. Latihan pertama – Vektor

1. Tuliskan Perintah berikut ini di jupyter notebook

```
a = [1, 2, -5, 0.3, 6, -2, 4] # numeric vector
b = ["one", "two", "three"] # character vector
c = [True, True, True, False, True] # logical vector
print(a)
print(b)
print(c)
```

Output:



The screenshot shows a Jupyter Notebook interface. The code cell contains the following R code:

```
a = [1, 2, -5, 0.3, 6, -2, 4]
b = ["one", "two", "three"]
c = [True, True, True, False, True]
print(a)
print(b)
print(c)
```

The output cell displays the results of the code execution:

```
[1, 2, -5, 0.3, 6, -2, 4]
['one', 'two', 'three']
[True, True, True, False, True]
```

b. Latihan Kedua – Matriks

1. Seluruh kolom dalam sebuah matriks harus memiliki tipe yang sama (numerik semua, karakter semua, dll) dan memiliki panjang yang sama.

**gunakan nama variable dengan nama anda masing-masing*

```
#MATRIKS
import numpy as np
cells = [3, 15, -27, 38]
r_nama = ["R1", "R2"]
c_nama = ["C1", "C2"]
nama_matrix = np.matrix(cells).reshape(2, 2)
print(nama_matrix)
```

Output:

```
import numpy as np
cells = [3,15,-27,38]
r_tarum = ["R1","R2"]
c_tarum = ["C1","C2"]
tarum_matrix = np.matrix(cells).reshape(2,2)
print(tarum_matrix)
```

```
[[ 3 15]
 [-27 38]]
```

c. Latihan Ketiga – Data Frame

1. Mengubah data input menjadi data frame

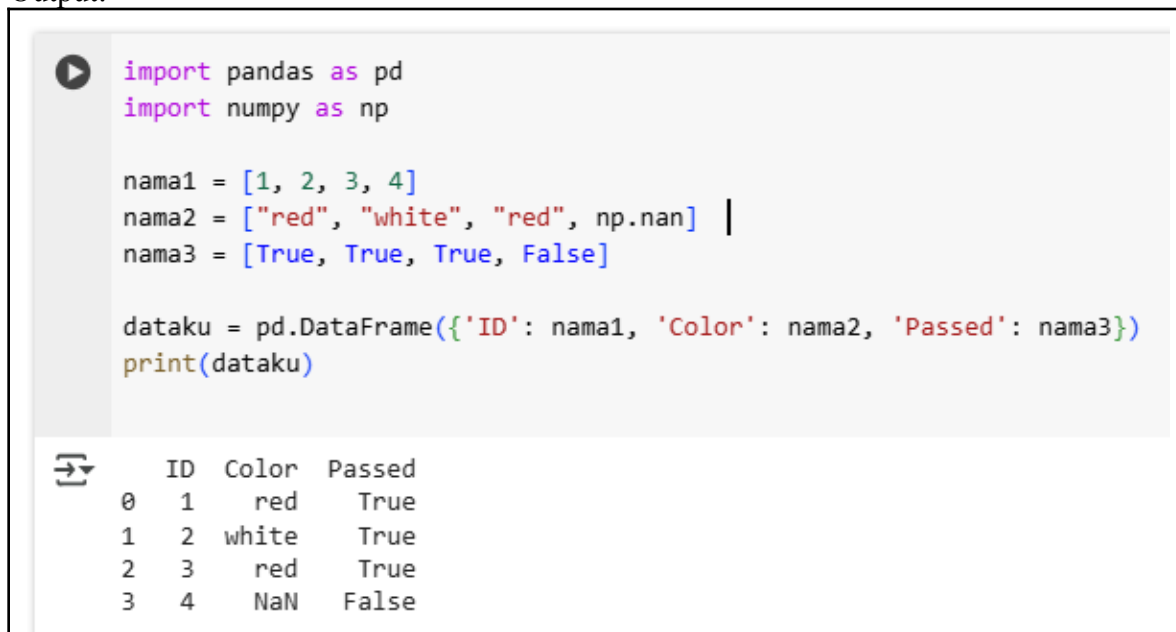
*gunakan nama variable dengan nama anda masing-masing

```
import pandas as pd
import numpy as np

nama1 = [1, 2, 3, 4]
nama2 = ["red", "white", "red", np.nan] # Menggunakan np.nan untuk
    merepresentasikan NA
nama3 = [True, True, True, False]

dataku = pd.DataFrame({'ID': nama1, 'Color': nama2, 'Passed': nama3})
print(dataku)
```

Output:



```
import pandas as pd
import numpy as np

nama1 = [1, 2, 3, 4]
nama2 = ["red", "white", "red", np.nan] |
nama3 = [True, True, True, False]

dataku = pd.DataFrame({'ID': nama1, 'Color': nama2, 'Passed': nama3})
print(dataku)
```

	ID	Color	Passed
0	1	red	True
1	2	white	True
2	3	red	True
3	4	NaN	False

2. Selanjutnya ketikkan perintah dibawah ini

```
import pandas as pd

data_nama = pd.DataFrame({'id': list('abcdefghij'), 'x': list(range(1, 11)), 'y': list(range(11, 21))})
print(data_nama)
```

[https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.h](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html)
tml

Output:



The image shows a Jupyter Notebook cell with the following code and output:

```
import pandas as pd

data_nama = pd.DataFrame({'id': list('abcdefghij'), 'x': list(range(1, 11)), 'y': list(range(11, 21))})
print(data_nama)
```

The output is a DataFrame with 10 rows and 4 columns (index, id, x, y):

	id	x	y
0	a	1	11
1	b	2	12
2	c	3	13
3	d	4	14
4	e	5	15
5	f	6	16
6	g	7	17
7	h	8	18
8	i	9	19
9	j	10	20

<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.htm>

d. Latihan Keempat – Koneksi ke Database

1. Buat sebuah nama database terlebih dahulu dengan nama houseprices di phpmyadmin, Lalu klik menu import

Start Apache* & MySQL, Buka browser, ketik <http://localhost/phpmyadmin/>

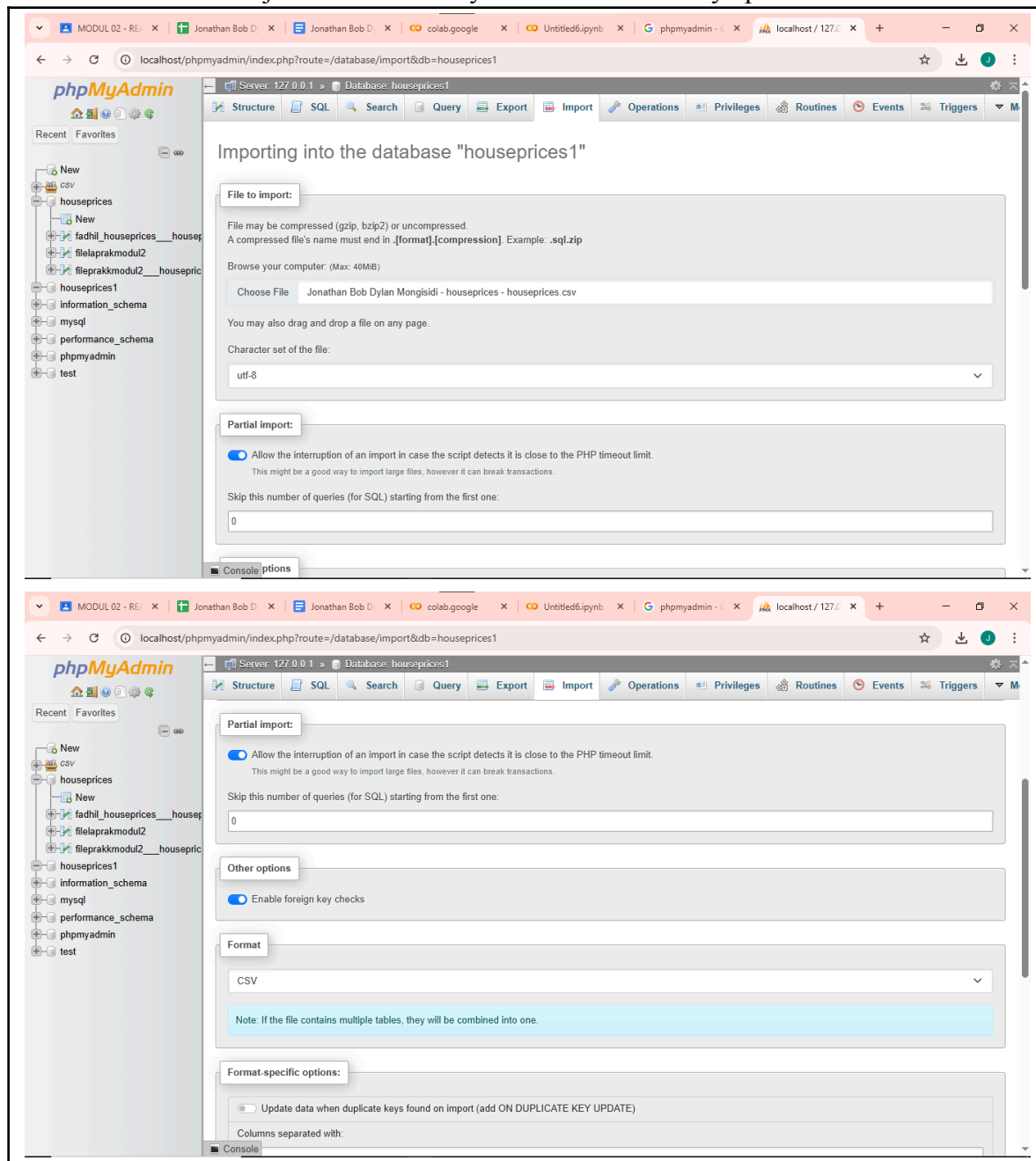
The image shows two screenshots. The top screenshot is the XAMPP Control Panel v3.3.0. It displays the status of various services: Apache (running), MySQL (stopped), FileZilla (stopped), Mercury (stopped), and Tomcat (stopped). The MySQL 'Stop' button is highlighted with a red box. Below the service list is a log window showing the initialization and status changes of the services.

The bottom screenshot is the phpMyAdmin interface. The 'Databases' tab is selected. On the left sidebar, a tree view shows the database structure, including 'houseprices' and 'mysql'. In the main area, the 'Create database' form is filled with 'houseprices1' as the name and 'utf8mb4_general_ci' as the collation. Below this, a table lists existing databases and their collations:

Database	Collation	Action
<input type="checkbox"/> csv_db 7	utf8_general_ci	Check privileges
<input type="checkbox"/> csv_db 8	utf8_general_ci	Check privileges
<input type="checkbox"/> csv_db 9	utf8_general_ci	Check privileges
<input type="checkbox"/> csv_db 10	utf8_general_ci	Check privileges
<input type="checkbox"/> houseprices	utf8mb4_general_ci	Check privileges
<input type="checkbox"/> information_schema	utf8_general_ci	Check privileges
<input type="checkbox"/> mysql	utf8mb4_general_ci	Check privileges
<input type="checkbox"/> performance_schema	utf8_general_ci	Check privileges
<input type="checkbox"/> phpmyadmin	utf8_bin	Check privileges
<input type="checkbox"/> test	latin1_swedish_ci	Check privileges

Total: 10

2. Pilih file yang ingin di import ke database (untuk file nama_excel.csv), Ceklis the first line of the file contains the table column name untuk membuat baris pertama pada file excel tersebut menjadi nama atributnya atau nama kolomnya pada database.



The image displays two screenshots of the phpMyAdmin web interface, showing the process of importing a CSV file into a MySQL database.

Top Screenshot: Import Options

The browser address bar shows the URL: `localhost/phpmyadmin/index.php?route=/database/import&db=houseprices1`. The interface shows the "Import" tab selected. The "Format-specific options" section is visible, with the following settings:

- ☐ Update data when duplicate keys found on import (add ON DUPLICATE KEY UPDATE)
- Columns separated with: `,`
- Columns enclosed with: `"`
- Columns escaped with: `"`
- Lines terminated with: `auto`
- Name of the new table (optional):
- Import these many number of rows (optional):
- ☒ The first line of the file contains the table column names (if this is unchecked, the first line will become part of the data)
- ☐ Do not abort on INSERT error

Bottom Screenshot: Import Results

The browser address bar shows the URL: `localhost/phpmyadmin/index.php?route=/import`. The interface shows the "Import" tab selected. The "Import has been successfully finished, 2 queries executed." message is displayed. The following structures have either been created or altered:

- houseprices1 (Options)
- jonathan_bob_dylan_mongisidi__houseprices__houseprices (Structure) (Options)

(Jonathan Bob Dylan Mongisidi - houseprices - houseprices.csv)

The MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)

```
CREATE TABLE IF NOT EXISTS `houseprices1`.`jonathan_bob_dylan_mongisidi__houseprices__houseprices` (
  `Price` int(6), `Sqft` int(4), `Bedrooms` int(1), `Bathrooms` int(1), `Offers` int(1), `Brick` varchar(3), `Neighborhood` varchar(5)
) DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;
```

128 rows inserted. (Query took 0.0004 seconds.)

```
INSERT INTO `houseprices1`.`jonathan_bob_dylan_mongisidi__houseprices__houseprices` (
  `Price`, `Sqft`, `Bedrooms`, `Bathrooms`, `Offers`, `Brick`, `Neighborhood`
) VALUES (
  114300, 1790, 2, 2, 2, 'No', 'East'), (114200, 2030, 4, 2, 3, 'No', 'East'), (114800, 1740, 3, 2, 1, 'No', 'East'), (94700, 1980, 3, 2, 3, 'No', 'East'), (119800, 2130, 3, 3, 3, 'No', 'East'), (114600, 1780, 3, 2, 2, 'No', 'North'), (151600, 1830, 3, 3, 'Yes', 'West'), (150700, 2160, 4, 2, 2, 'No', 'West'), (119200, 2110, 4, 2, 3, 'No', 'East'), (104000, 1730, 3, 3, 3, 'No', 'East'), (132500, 2030, 3, 2, 3, 'Yes', 'East'), (123000, 1870, 2, 2, 2, 'Yes', 'East'), (102600, 1910, 3, 2, 4, 'No', 'North'), (126300, 2150, 3, 3, 5, 'Yes', 'North'), (176800, 2590, 4, 3, 4, 'No', 'West'), (145800, 1780, 4, 2, 1, 'No', 'West'), (147100, 2190, 3, 3, 4, 'Yes', 'East'), (83600, 1990, 3,
```


3. Klik go, Rename nama tabel sesuai dengan nama anda

The screenshot shows the phpMyAdmin interface with the 'houseprices1' database selected. The table 'jonathan_houseprices' is highlighted in the left sidebar. The main panel shows the 'Operations' tab for this table. The 'Alter table order by' section is expanded, showing 'Price' as the sort order, 'Ascending', and '(singly)'. The 'Move table to (database.table)' section shows the table being moved to 'houseprices1' in the 'jonathan_bob_dylan' database. The 'Table options' section is expanded, showing the 'Rename table to' field set to 'jonathan_houseprices', with 'Adjust privileges' checked. The 'Storage engine' is set to 'InnoDB'.

Below the operations panel, a message states: "Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available." A green bar indicates "Showing rows 0 - 24 (128 total, Query took 0.0004 seconds)". The SQL query shown is "SELECT * FROM 'jonathan_houseprices'". The table data is displayed in a grid with columns: Price, SqFt, Bedrooms, Bathrooms, Offers, Brick, and Neighborhood.

Price	SqFt	Bedrooms	Bathrooms	Offers	Brick	Neighborhood
114300	1790	2	2	2	No	East
114200	2030	4	2	3	No	East
114800	1740	3	2	1	No	East
94700	1980	3	2	3	No	East
119800	2130	3	3	3	No	East
114600	1780	3	2	2	No	North
151600	1830	3	3	3	Yes	West
150700	2160	4	2	2	No	West
119200	2110	4	2	3	No	East
104000	1730	3	3	3	No	East
132500	2030	3	2	3	Yes	East
123000	1870	2	2	2	Yes	East
102600	1910	3	2	4	No	North
126300	2150	3	3	5	Yes	North
159000	1590	4	3	4	No	West

4. Kembali ke jupyter notebook, lalu instal dahulu library yang dibutuhkan pada python. Jika belum tersedia, maka lakukan instruksi:

pip install mysql-connector-python

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\User\Documents> pip install mysql-connector-python
Requirement already satisfied: mysql-connector-python in c:\users\user\anaconda3\lib\site-packages (9.2.0)
PS C:\Users\User\Documents> █
```

5. Lalu jalankan perintah dibawah ini

```
In [ ]: import mysql.connector

# Membuat koneksi ke MySQL
connection = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="houseprices"
)

# Membuat objek cursor untuk mengeksekusi kueri
cursor = connection.cursor()

try:
    # Mengeksekusi kueri SQL
    my_query = "SELECT * FROM nama;"
    cursor.execute(my_query)

    # Mengambil semua hasil kueri
    result = cursor.fetchall()

    # Menampilkan hasil kueri
    print("\nHasil Kueri:")
    for row in result:
        print(row)

finally:
    # Menutup kurson dan koneksi
    cursor.close()
    connection.close()
```

Output:

```
[10]: import mysql.connector

connection = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="houseprices1"
)

cursor = connection.cursor()

try:
    my_query= "SELECT * FROM jonathan_houseprices;"
    cursor.execute(my_query)

    result = cursor.fetchall()

    print("\nHasil Kueri")
    for row in result:
        print(row)

finally:
    cursor.close()
    connection.close()
```

```

Hasil Kueri
(114300, 1790, 2, 2, 'No', 'East')
(114200, 2030, 4, 2, 3, 'No', 'East')
(114800, 1740, 3, 2, 1, 'No', 'East')
(94700, 1980, 3, 2, 3, 'No', 'East')
(119800, 2130, 3, 3, 3, 'No', 'East')
(114600, 1780, 3, 2, 2, 'No', 'North')
(151600, 1830, 3, 3, 3, 'Yes', 'West')
(150700, 2160, 4, 2, 2, 'No', 'West')
(119200, 2110, 4, 2, 3, 'No', 'East')
(104000, 1730, 3, 3, 3, 'No', 'East')
(132500, 2030, 3, 2, 3, 'Yes', 'East')
(123000, 1870, 2, 2, 2, 'Yes', 'East')
(102600, 1910, 3, 2, 4, 'No', 'North')
(126300, 2150, 3, 3, 5, 'Yes', 'North')
(176800, 2590, 4, 3, 4, 'No', 'West')
(145800, 1780, 4, 2, 1, 'No', 'West')

```

6. Jalankan perintah dibawah ini:

***Perintah ini akan menampilkan 86 baris data hasil filter.**

```

In [ ]: import pandas as pd
# Mengonversi hasil kueri ke DataFrame Pandas
df = pd.DataFrame(result, columns=[desc[0] for desc in cursor.description])

# Filter data berdasarkan kolom 'Brick' yang bernilai 'No'
df_filtered = df[df['Brick'] == 'No']

# Menampilkan hasil filter
print("\nHasil Filter:")
print(df_filtered)

```

Output:

```

[12]: import pandas as pd

df = pd.DataFrame(result, columns=[desc[0] for desc in cursor.description])

df_filtered = df[df['Brick'] == 'No']

print("\nHasil Filter:")
print(df_filtered)

```

```

Hasil Filter:
   Price  SqFt  Bedrooms  Bathrooms  Offers  Brick  Neighborhood
0  114300  1790         2          2        2    No          East
1  114200  2030         4          2        3    No          East
2  114800  1740         3          2        1    No          East
3   94700  1980         3          2        3    No          East
4  119800  2130         3          3        3    No          East
..    ...    ...    ...    ...    ...    ...    ...
120 110400  1930         2          3        3    No          North
121 105600  1930         3          3        3    No          East
125 113500  2070         2          2        2    No          North
126 149900  2020         3          3        1    No          West
127 124600  2250         3          3        4    No          North

```

[86 rows x 7 columns]

7. Jalankan perintah dibawah ini:

***Perintah ini akan menampilkan 105 baris data hasil filter.**

```
In [ ]: import pandas as pd
# Mengonversi hasil kueri ke DataFrame Pandas
df = pd.DataFrame(result, columns=[desc[0] for desc in cursor.description])

# Filter data berdasarkan kondisi yang kompleks
df_filtered = df[(df['Brick'] == 'No') | (df['Neighborhood'] == 'East')]

# Menampilkan hasil filter
print(df_filtered)
```

Output:

```
[16]: import pandas as pd

df = pd.DataFrame(result, columns=[desc[0] for desc in cursor.description])

df_filtered = df[(df['Brick'] == 'No') | (df['Neighborhood'] == 'east')]

print(df_filtered)
```

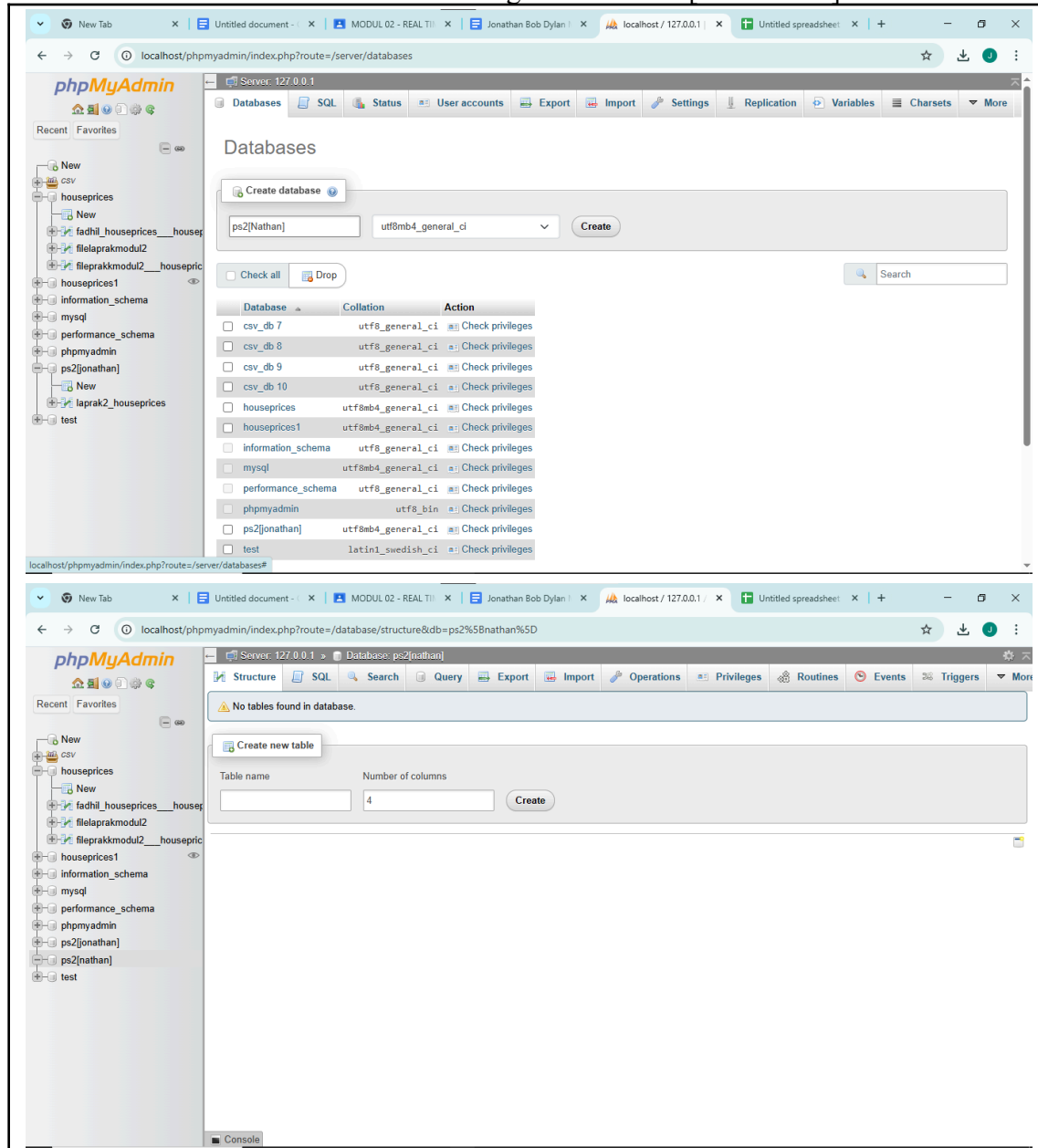
	Price	SqFt	Bedrooms	Bathrooms	Offers	Brick	Neighborhood
0	114300	1790	2	2	2	No	East
1	114200	2030	4	2	3	No	East
2	114800	1740	3	2	1	No	East
3	94700	1980	3	2	3	No	East
4	119800	2130	3	3	3	No	East
..
120	110400	1930	2	3	3	No	North
121	105600	1930	3	3	3	No	East
125	113500	2070	2	2	2	No	North
126	149900	2020	3	3	1	No	West
127	124600	2250	3	3	4	No	North

[86 rows x 7 columns]

e. Latihan Keenam – Tugas

Buat sebuah database serta tabel di dalamnya (bisa gunakan data teman dipraktikum ke 1). Lakukan koneksi python ke database serta berikan beberapa filter data sesuai yang anda inginkan. Tampilkan data tersebut

1. Buatlah terlebih dahulu Database baru dengan format PS2[NamaAnda]



2. Import file berformat csv yang telah anda buat sebelumnya di Praktikum 1, dimana file tersebut berisi 6 kolom dan 20 baris data.

Showing rows 0 - 19 (20 total, Query took 0.0004 seconds.)

```
SELECT * FROM `data_mahasiswa`
```

Extra options

Nama	Gender	Angkatan	Tinggi Badan	Waktu Pelajaran	Wilayah Tinggal
name1	P	2024	165	15	Bekasi
name2	L	2024	170	25	Tomang
name3	L	2024	175	20	Grogol
name4	P	2024	160	30	Taman Anggrek
name5	L	2024	160	35	Tangerang
name6	P	2024	165	45	Tangerang
name7	L	2024	180	40	Taman Anggrek
name8	P	2024	160	60	Bekasi
name9	P	2024	175	65	Bekasi
name10	P	2024	170	50	Bogor
name11	L	2024	160	55	Bogor
name12	L	2024	180	50	Bogor
name13	L	2024	175	45	Taman Anggrek
name14	P	2024	170	70	Grogol
Console		2024	165	45	Grogol

3. Koneksikan Python ke Database tersebut sebagaimana yang telah anda lakukan pada Elemen Kompetensi 1 dimodul kedua ini dengan menyesuaikan kembali nama Database baru yang sudah dibuat.

```
print("\nHasil Kueri:")
for row in result:
    print(row)

finally:
    cursor.close()
    connection.close()
```

Hasil Kueri:

```
(('name1', 'P', 2024, 165, 15, 'Bekasi'))
(('name2', 'L', 2024, 170, 25, 'Tomang'))
(('name3', 'L', 2024, 175, 20, 'Grogol'))
(('name4', 'P', 2024, 160, 30, 'Taman Anggrek'))
(('name5', 'L', 2024, 160, 35, 'Tangerang'))
(('name6', 'P', 2024, 165, 45, 'Tangerang'))
(('name7', 'L', 2024, 180, 40, 'Taman Anggrek'))
(('name8', 'P', 2024, 160, 60, 'Bekasi'))
(('name9', 'P', 2024, 175, 65, 'Bekasi'))
(('name10', 'P', 2024, 170, 50, 'Bogor'))
(('name11', 'L', 2024, 160, 55, 'Bogor'))
(('name12', 'L', 2024, 180, 50, 'Bogor'))
(('name13', 'L', 2024, 175, 45, 'Taman Anggrek'))
(('name14', 'P', 2024, 170, 70, 'Grogol'))
(('name15', 'P', 2024, 165, 45, 'Grogol'))
(('name16', 'P', 2024, 160, 15, 'Daan Mogot'))
(('name17', 'L', 2024, 170, 25, 'Manado'))
(('name18', 'L', 2024, 165, 20, 'Makassar'))
(('name19', 'L', 2024, 170, 70, 'Palu'))
(('name20', 'P', 2024, 175, 65, 'Ternate'))
```

4. Lakukan filter data terhadap Kolom Gender, untuk melihat berapa baris data Pria/Wanita (Pilih salah 1).

The screenshot shows a JupyterLab window with a code cell containing a list of student data and a filtered DataFrame. The data is as follows:

Nama	Gender	Angkatan	Tinggi Badan	Waktu Pelajaran Wilayah	Tinggal
name2	L	2024	170	25	Tomang
name3	L	2024	175	20	Grogol
name5	L	2024	160	35	Tangerang
name7	L	2024	180	40	Taman Anggrek
name11	L	2024	160	55	Bogor
name12	L	2024	180	50	Bogor
name13	L	2024	175	45	Taman Anggrek
name15	L	2024	165	25	Manado
name17	L	2024	170	20	Makassar
name18	L	2024	165	20	Makassar
name19	L	2024	170	70	Palu
name20	P	2024	175	65	Ternate

The code cell shows the following Python code:

```
[3]: import pandas as pd

df = pd.DataFrame(result, columns=[desc[0] for desc in cursor.description])
df_filtered = df[df['Gender'] == 'L']
print(df_filtered)
```

5. Lampirkan Screenshot

☞ Kode koneksi Jupyter ke Database

☞ Kode serta hasil filter di Jupyter

The screenshot shows a JupyterLab window with a code cell containing database connection code and filtered data. The code is as follows:

```
[1]: import mysql.connector
connection=mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="ps2[Nathan]"
)

cursor=connection.cursor()

try:
    my_query="SELECT * FROM data_mahasiswa"
    cursor.execute(my_query)

    result=cursor.fetchall()

    print("\nHasil Kueri:")
    for row in result:
        print(row)

finally:
    cursor.close()
    connection.close()
```

The output of the code is:

```
Hasil Kueri:
('name1', 'P', 2024, 165, 15, 'Bekasi')
('name2', 'L', 2024, 170, 25, 'Tomang')
('name3', 'L', 2024, 175, 20, 'Grogol')
('name4', 'P', 2024, 160, 30, 'Taman Anggrek')
('name5', 'L', 2024, 160, 35, 'Tangerang')
('name6', 'P', 2024, 165, 45, 'Tangerang')
('name7', 'L', 2024, 180, 40, 'Taman Anggrek')
('name8', 'P', 2024, 160, 60, 'Bekasi')
('name9', 'P', 2024, 175, 65, 'Bekasi')
('name10', 'P', 2024, 170, 50, 'Bogor')
('name11', 'L', 2024, 160, 55, 'Bogor')
('name12', 'L', 2024, 180, 50, 'Bogor')
('name13', 'L', 2024, 175, 45, 'Taman Anggrek')
('name14', 'P', 2024, 170, 70, 'Grogol')
('name15', 'P', 2024, 165, 45, 'Grogol')
('name16', 'P', 2024, 160, 15, 'Daan Hogot')
('name17', 'L', 2024, 170, 25, 'Manado')
('name18', 'L', 2024, 165, 20, 'Makassar')
('name19', 'L', 2024, 170, 70, 'Palu')
('name20', 'P', 2024, 175, 65, 'Ternate')
```

The code cell shows the following Python code:

```
[3]: import pandas as pd

df = pd.DataFrame(result, columns=[desc[0] for desc in cursor.description])
df_filtered = df[df['Gender'] == 'L']
print(df_filtered)
```

Jonathan Bob Dylan Mongisidi - 064102400013

4. File Praktikum

Github Repository:

--

5. Kesimpulan

Kesimpulan yang bisa saya ambil dari praktikum probabilitas dan statistika kali ini adalah dimana saya bisa mengerti tipe data, filter data, dan koneksi ke database dan juga mendapatkan source code baru

6. Cek List (✓)

No	Elemen Kompetensi	Penyelesaian	
		Selesai	Tidak Selesai
1.	Latihan Pertama	✓	
2.	Latihan Kedua	✓	
3.	Latihan Ketiga	✓	
4.	Latihan Keempat	✓	
5.	Latihan Kelima	✓	
6.	Latihan Keenam	✓	

7. Formulir Umpan Balik

No	Elemen Kompetensi	Waktu Pengerjaan	Kriteria
1.	Latihan Pertama	30 Menit	Menarik
2.	Latihan Kedua	30 Menit	Menarik
3.	Latihan Ketiga	30 Menit	Menarik
4.	Latihan Keempat	30 Menit	Menarik
5.	Latihan Kelima	30 Menit	Menarik
6.	Latihan Keenam	30 Menit	Menarik

Keterangan:

1. Menarik
2. Baik
3. Cukup
4. Kurang

Jonathan Bob Dylan Mongisidi - 064102400013