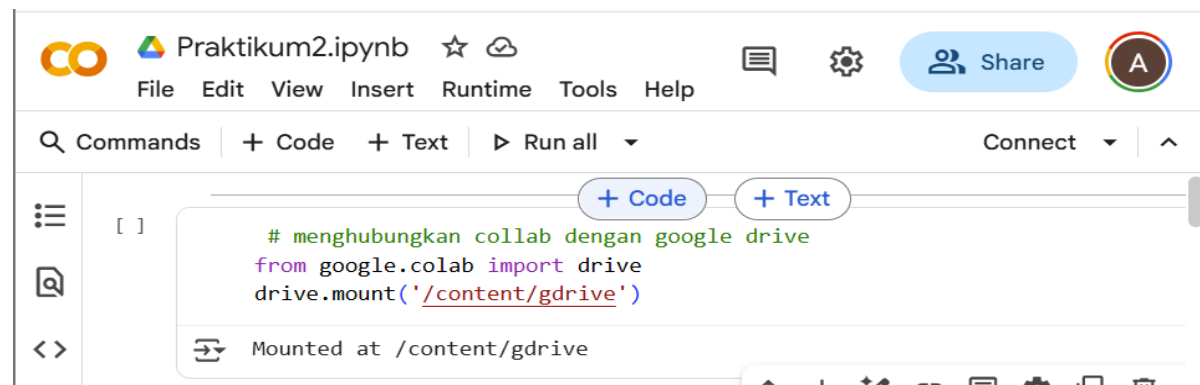


Nama : Alia Maisyarah

Class : 3TI04

Nim : 0110224095

Kode & Penjelasan



The screenshot shows the Google Colab interface for a notebook named 'Praktikum2.ipynb'. The top menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu, there are buttons for '+ Code', '+ Text', and 'Run all'. The code editor contains the following Python code:

```
# menghubungkan collab dengan google drive
from google.colab import drive
drive.mount('/content/gdrive')
```

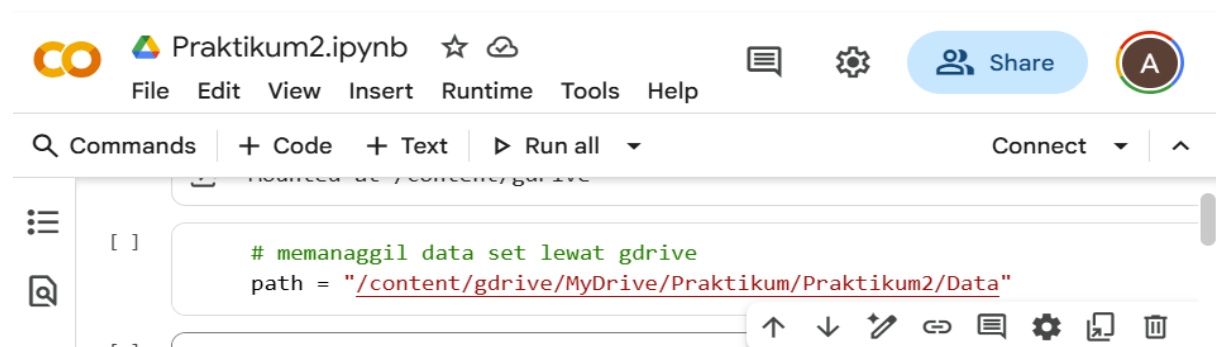
Below the code editor, a status bar indicates 'Mounted at /content/gdrive'.

Kode ini digunakan untuk menghubungkan Google Colab dengan Google Drive agar file dataset bisa diakses langsung.

Kesimpulan:

Drive sudah terhubung, sehingga file data dapat digunakan dalam proses analisis.

Kode & Penjelasan



The screenshot shows the Google Colab interface for the same notebook. The code editor contains the following Python code:

```
# memanaggil data set lewat gdrive
path = "/content/gdrive/MyDrive/Praktikum/Praktikum2/Data"
```

The status bar below the code editor shows the path '/content/gdrive/MyDrive/Praktikum/Praktikum2/Data'.

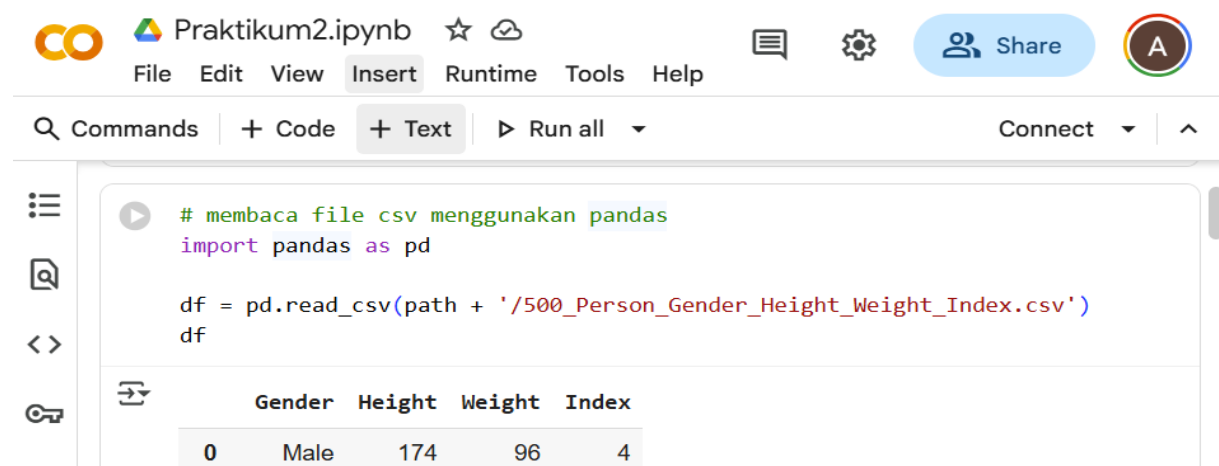
Kode ini digunakan untuk menyimpan **alamat folder tempat dataset berada** di Google Drive.

Variabel path akan dipakai nanti saat membaca file CSV.

Kesimpulan:

Dengan membuat variabel path, kita bisa memanggil file dataset dari folder Drive tanpa menulis alamat panjang berulang kali.

Kode & Penjelasan



The screenshot shows a Jupyter Notebook interface with the following components:

- Top Bar:** Includes the Colab logo, the filename "Praktikum2.ipynb", and icons for star, cloud, chat, settings, share, and a user profile.
- Menu Bar:** Contains "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help".
- Toolbar:** Includes "Commands", "+ Code", "+ Text", "Run all", "Connect", and a caret icon.
- Code Cell:** Contains the following Python code:

```
# membaca file csv menggunakan pandas
import pandas as pd

df = pd.read_csv(path + '/500_Person_Gender_Height_Weight_Index.csv')
df
```
- Output:** A table with 5 columns: Gender, Height, Weight, and Index. The first row of data is shown below the header.

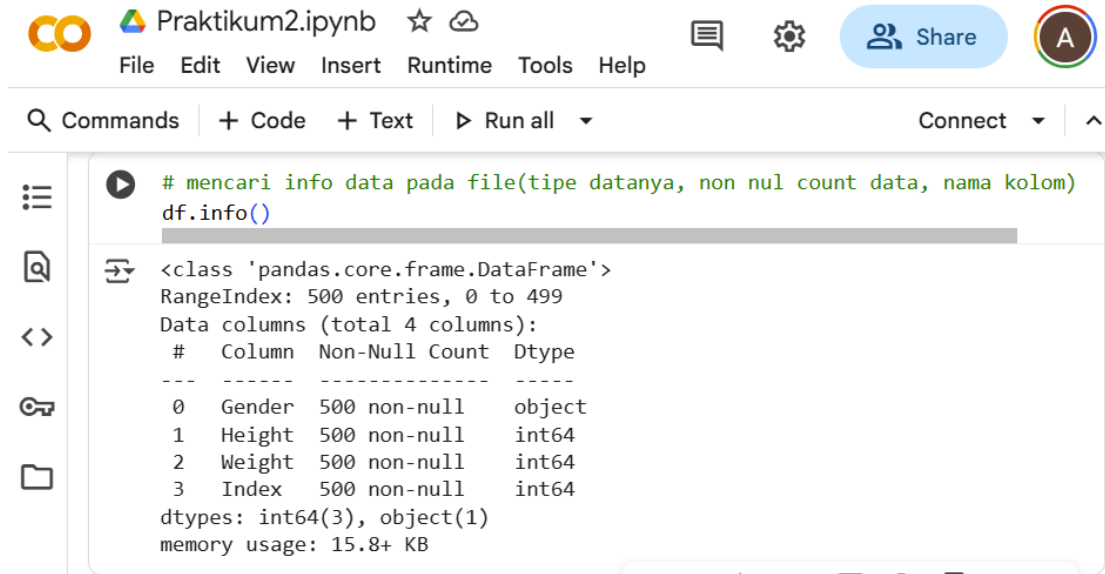
	Gender	Height	Weight	Index
0	Male	174	96	4

Kode ini menggunakan library pandas untuk membaca file CSV dari folder yang sudah ditentukan.
Hasilnya disimpan dalam variabel df, lalu ditampilkan untuk melihat isi data.

Kesimpulan:

Dataset berhasil dibaca dan ditampilkan dalam bentuk tabel, berisi kolom seperti *Gender*, *Height*, *Weight*, dan *Index*.

Kode & Penjelasan



```
# mencari info data pada file(tipe datanya, non nul count data, nama kolom)
df.info()

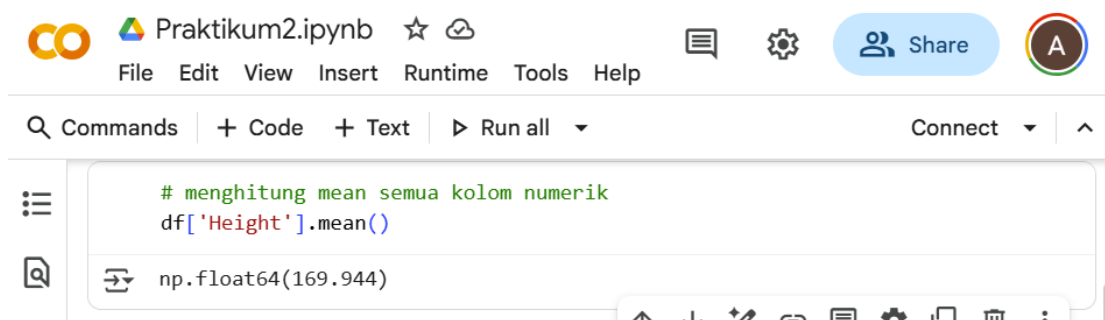
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  -
 0   Gender  500 non-null    object  
 1   Height  500 non-null    int64   
 2   Weight  500 non-null    int64   
 3   Index   500 non-null    int64   
dtypes: int64(3), object(1)
memory usage: 15.8+ KB
```

Kode ini menampilkan **informasi umum tentang dataset**, seperti jumlah baris, nama kolom, tipe data, dan jumlah data yang tidak kosong (*non-null*).

Kesimpulan:

Dari output terlihat bahwa dataset memiliki 500 baris dan 4 kolom, semuanya terisi penuh tanpa data kosong.

Kode & Penjelasan



```
# menghitung mean semua kolom numerik
df['Height'].mean()

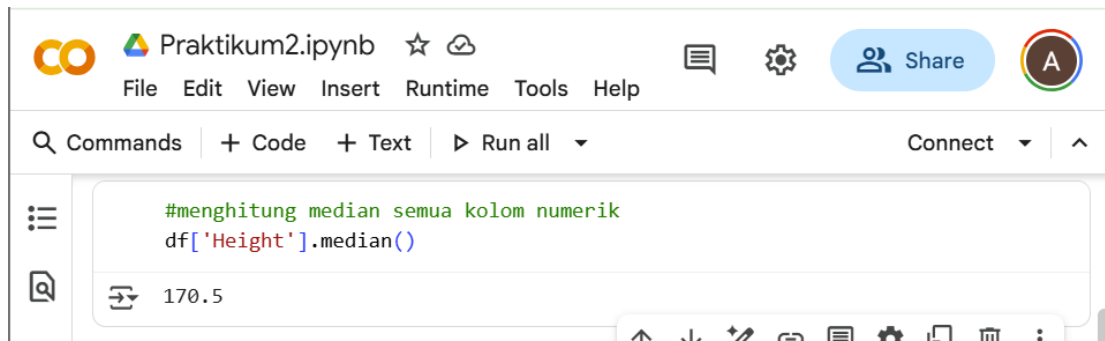
np.float64(169.944)
```

Kode ini menghitung **nilai rata-rata (mean)** dari kolom Height.

Kesimpulan:

Output menunjukkan rata-rata tinggi badan dari seluruh data dalam dataset.

Kode & Penjelasan



The screenshot shows a Jupyter Notebook interface with a file named 'Praktikum2.ipynb'. The code cell contains the following Python code:

```
#menghitung median semua kolom numerik  
df['Height'].median()
```

The output of the code is displayed below the code cell:

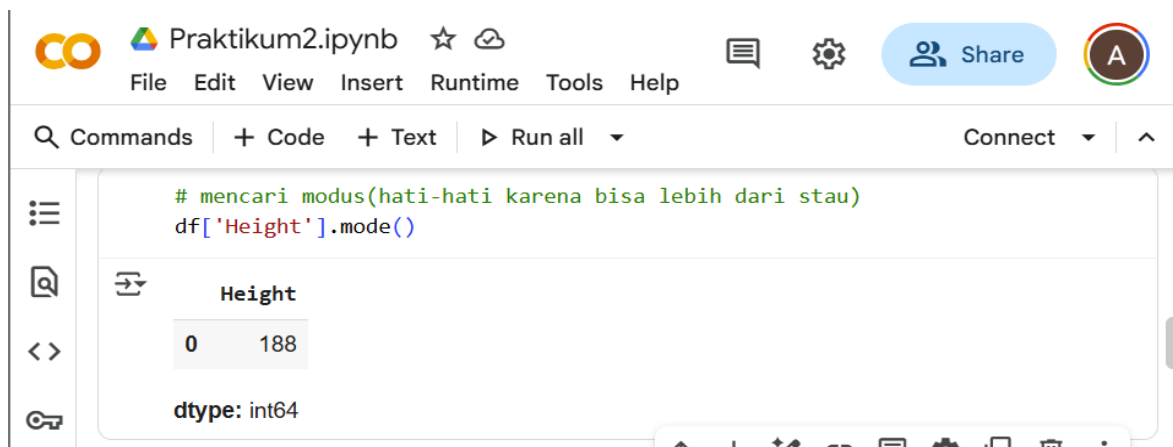
```
170.5
```

Kode ini digunakan untuk menghitung **nilai tengah (median)** dari kolom Height.

Kesimpulan:

Output menunjukkan nilai tinggi badan yang berada di posisi tengah setelah semua data diurutkan.

Kode & Penjelasan



The screenshot shows a Jupyter Notebook interface with a file named 'Praktikum2.ipynb'. The code cell contains the following Python code:

```
# mencari modus(hati-hati karena bisa lebih dari stau)  
df['Height'].mode()
```

The output of the code is displayed below the code cell:

```
Height  
0    188  
  
dtype: int64
```

Kode ini digunakan untuk mencari **modus**, yaitu nilai yang **paling sering muncul** pada kolom Height.

Kesimpulan:

Output menunjukkan satu atau lebih nilai tinggi badan yang paling sering muncul dalam dataset.

Kode & Penjelasan

The screenshot shows a Jupyter Notebook titled 'Praktikum2.ipynb'. The code cell contains the following Python code:

```
# menghitung variansi & standard deviasi
df.var(numeric_only=True)
```

The output of the code is displayed as a table with the following values:

	0
Height	268.149162
Weight	1048.633267
Index	1.836168

Below the table, it indicates 'dtype: float64'.

Kode ini menghitung **variansi** dari setiap kolom numerik, yaitu ukuran seberapa jauh data menyebar dari rata-ratanya.

Kesimpulan:

Output menunjukkan tingkat penyebaran nilai pada tiap kolom numerik dalam dataset.

Kode & Penjelasan

The screenshot shows a Jupyter Notebook titled 'Praktikum2.ipynb'. The code cell contains the following Python code:

```
# menghitung standar deviasi
df.std(numeric_only=True)
```

The output of the code is displayed as a table with the following values:

	0
Height	16.375261
Weight	32.382607
Index	1.355053

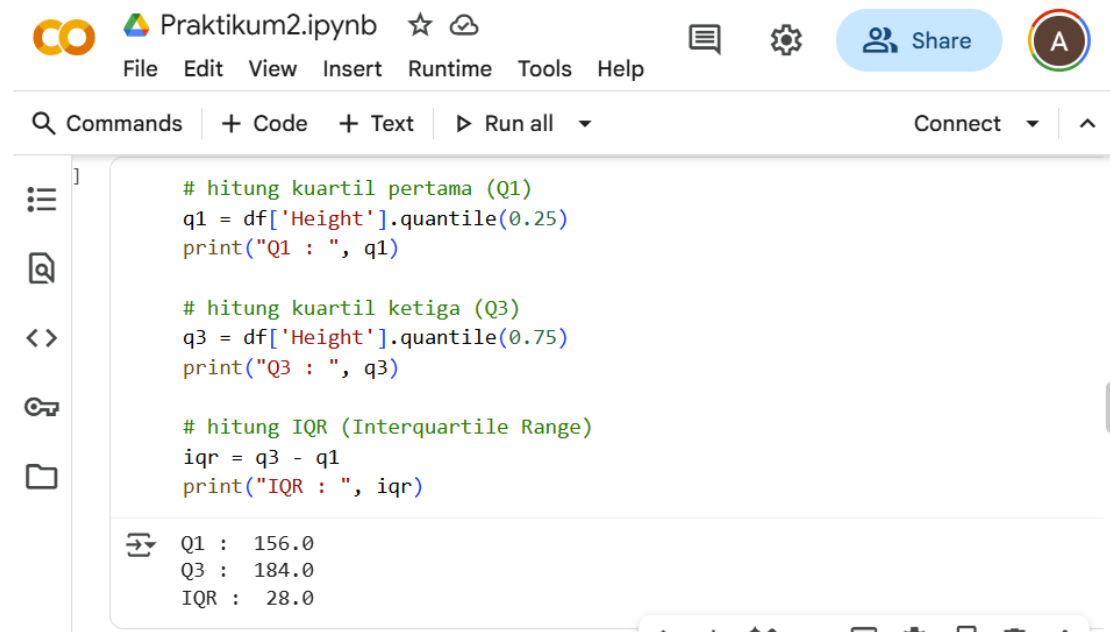
Below the table, it indicates 'dtype: float64'.

Kode ini menghitung **standar deviasi** dari setiap kolom numerik, yaitu ukuran seberapa jauh nilai data bervariasi dari rata-ratanya.

Kesimpulan:

Output menunjukkan besar penyimpangan data terhadap nilai rata-rata pada setiap kolom numerik.

Kode & Penjelasan



The screenshot shows a Jupyter Notebook interface with the file 'Praktikum2.ipynb'. The code in the cell calculates the first quartile (Q1), third quartile (Q3), and the Interquartile Range (IQR) for the 'Height' column of a DataFrame named 'df'. The output of the code is displayed below the cell.

```
# hitung kuartil pertama (Q1)
q1 = df['Height'].quantile(0.25)
print("Q1 : ", q1)

# hitung kuartil ketiga (Q3)
q3 = df['Height'].quantile(0.75)
print("Q3 : ", q3)

# hitung IQR (Interquartile Range)
iqr = q3 - q1
print("IQR : ", iqr)
```

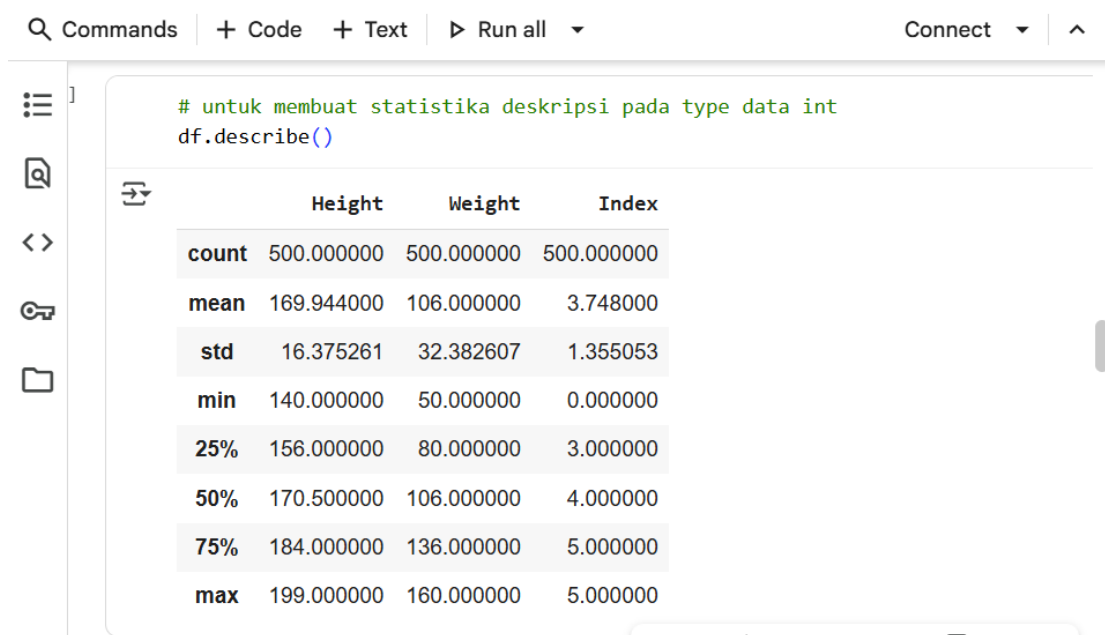
Q1 : 156.0
Q3 : 184.0
IQR : 28.0

Kode ini menghitung **kuartil pertama (Q1)**, **kuartil ketiga (Q3)**, dan **IQR (Interquartile Range)** dari kolom Height. IQR menunjukkan jarak antara Q1 dan Q3 atau sebaran data di tengah.

Kesimpulan:

Output menampilkan nilai Q1, Q3, dan IQR yang menggambarkan rentang penyebaran data tinggi badan di bagian tengah dataset.

Kode & Penjelasan



```
# untuk membuat statistika deskripsi pada type data int
df.describe()
```

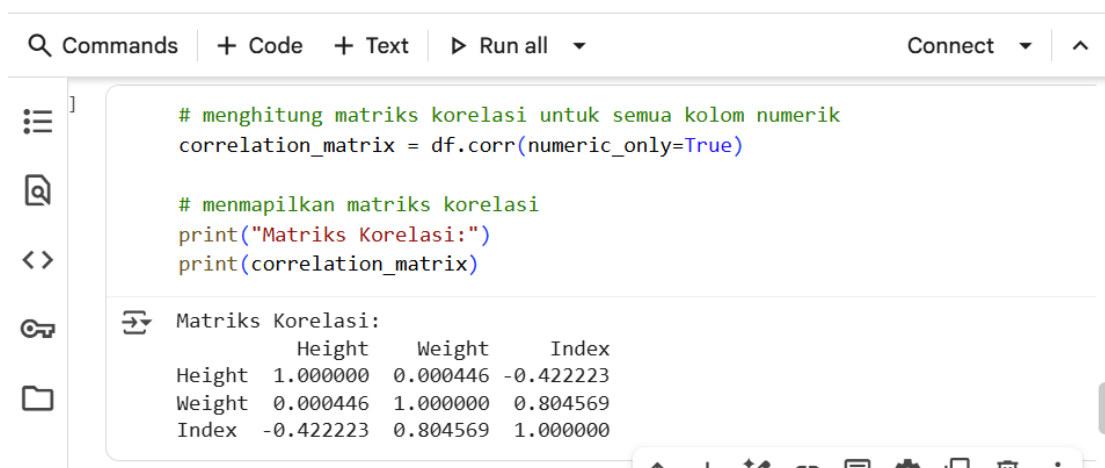
	Height	Weight	Index
count	500.000000	500.000000	500.000000
mean	169.944000	106.000000	3.748000
std	16.375261	32.382607	1.355053
min	140.000000	50.000000	0.000000
25%	156.000000	80.000000	3.000000
50%	170.500000	106.000000	4.000000
75%	184.000000	136.000000	5.000000
max	199.000000	160.000000	5.000000

Kode ini menampilkan **statistik deskriptif otomatis** untuk kolom bertipe numerik, seperti jumlah data, rata-rata, standar deviasi, nilai minimum, maksimum, dan kuartil.

Kesimpulan:

Output memberikan gambaran umum tentang distribusi dan karakteristik data numerik dalam dataset.

Kode & Penjelasan



```
# menghitung matriks korelasi untuk semua kolom numerik
correlation_matrix = df.corr(numeric_only=True)

# menampilkan matriks korelasi
print("Matriks Korelasi:")
print(correlation_matrix)
```

Matriks Korelasi:

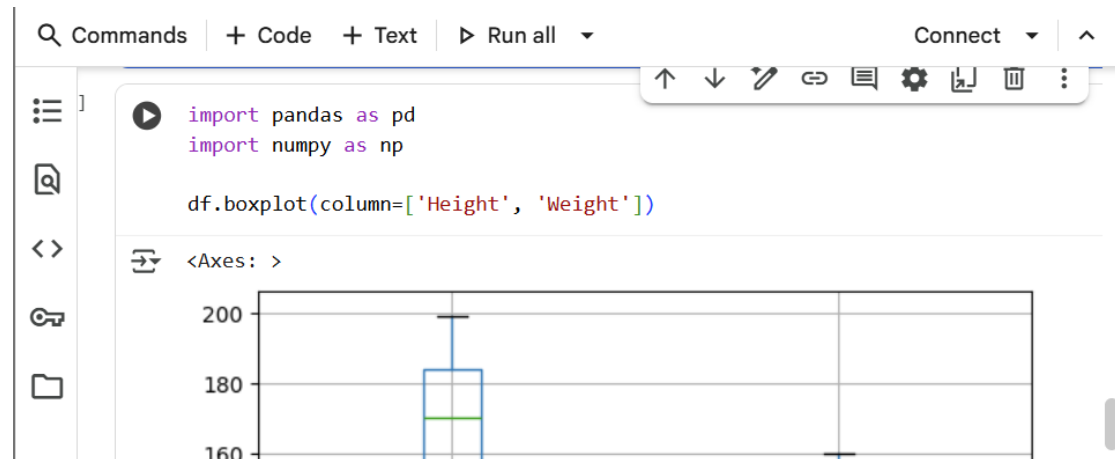
	Height	Weight	Index
Height	1.000000	0.000446	-0.422223
Weight	0.000446	1.000000	0.804569
Index	-0.422223	0.804569	1.000000

Kode ini menghitung dan menampilkan **matriks korelasi** antar kolom numerik untuk melihat hubungan antar variabel.

Kesimpulan:

Output menunjukkan seberapa kuat hubungan antar kolom — nilai positif berarti hubungan searah, sedangkan negatif berarti berlawanan arah.

Kode & Penjelasan

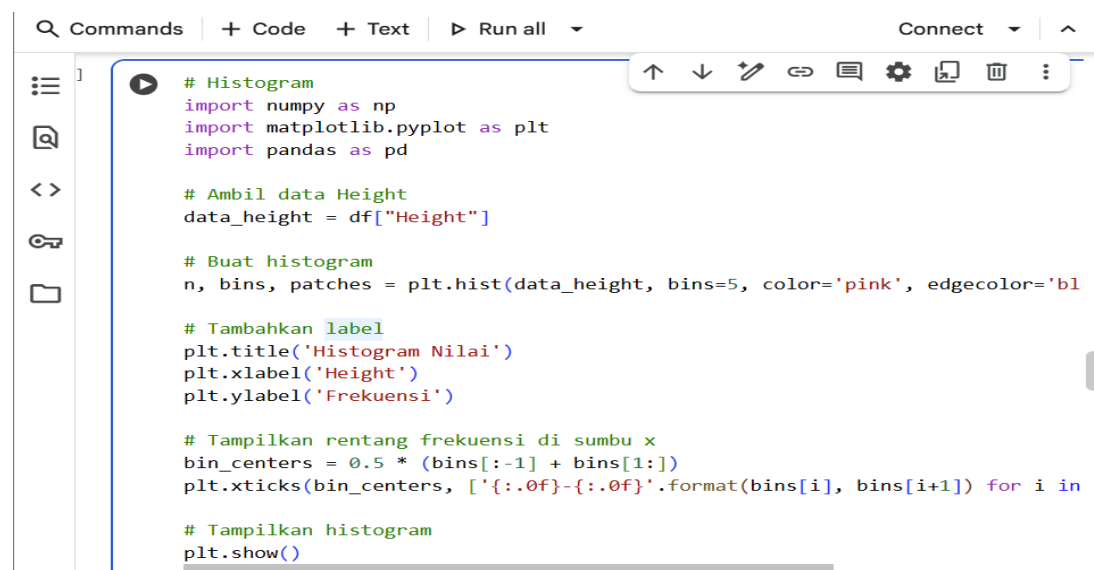


Kode ini membuat **boxplot** untuk kolom Height dan Weight guna melihat persebaran data serta mendeteksi nilai ekstrem (*outlier*).

Kesimpulan:

Output menampilkan diagram boxplot yang memperlihatkan median, kuartil, dan kemungkinan adanya *outlier* pada data tinggi dan berat badan.

Kode & Penjelasan



Kode ini membuat **histogram** dari kolom Height untuk menampilkan sebaran frekuensi data tinggi badan ke dalam 5 kelompok (bins).

Kesimpulan:

Output berupa grafik batang yang menunjukkan rentang nilai tinggi badan dan berapa banyak data yang masuk ke tiap rentang.

Kode & Penjelasan

```
import pandas as pd
import matplotlib.pyplot as plt

# Buat DataFrame contoh
data = {
    'Nilai1': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'Nilai2': [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
}

df2 = pd.DataFrame(data)

#Buat scatter plot
plt.scatter(df2['Nilai1'], df2['Nilai2'], color='blue', marker='o')

# Tambahkan label
plt.title('Scatter Plot Korelasi Positif')
plt.xlabel('Nilai1')
plt.ylabel('Nilai2')

#Tambahkan grid
plt.grid(True)

#Tampilkan plot
plt.show()
```

Kode ini membuat **scatter plot** (diagram sebar) untuk menunjukkan hubungan antara dua variabel dengan titik-titik data.

Kesimpulan:

Output menampilkan titik-titik yang membentuk pola naik, menandakan **korelasi positif** — ketika satu nilai naik, nilai lainnya juga ikut naik.

Kode & Penjelasan

```
import pandas as pd
import matplotlib.pyplot as plt

# Buat DataFrame contoh
data = {
    'Nilai1': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'Nilai2': [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
}

df3 = pd.DataFrame(data)

#Buat scatter plot
plt.scatter(df3['Nilai1'], df3['Nilai2'], color='red', marker='x')

# Tambahkan label
plt.title('Scatter Plot Korelasi Negatif')
plt.xlabel('Nilai1')
plt.ylabel('Nilai2')
|
#Tambahkan grid
plt.grid(True)

#Tampilkan plot
plt.show()
```

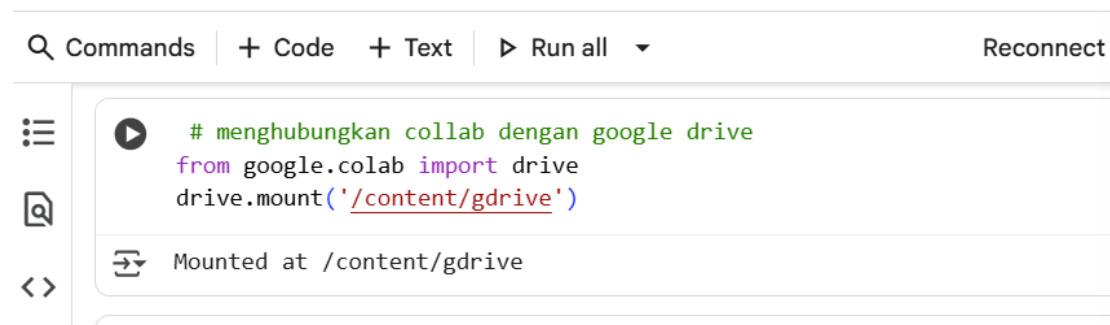
Kode ini membuat **scatter plot** untuk menunjukkan hubungan antara dua variabel (Nilai1 dan Nilai2) dengan tanda “x” berwarna merah.

Kesimpulan:

Titik-titik pada grafik membentuk pola menurun, menunjukkan **korelasi negatif** — ketika satu nilai naik, nilai lainnya menurun.

TUGAS PRAKTIKUM2

Kode & Penjelasan



```
Q Commands | + Code | + Text | ▶ Run all | Reconnect
```

```
# menghubungkan collab dengan google drive
from google.colab import drive
drive.mount('/content/gdrive')
```

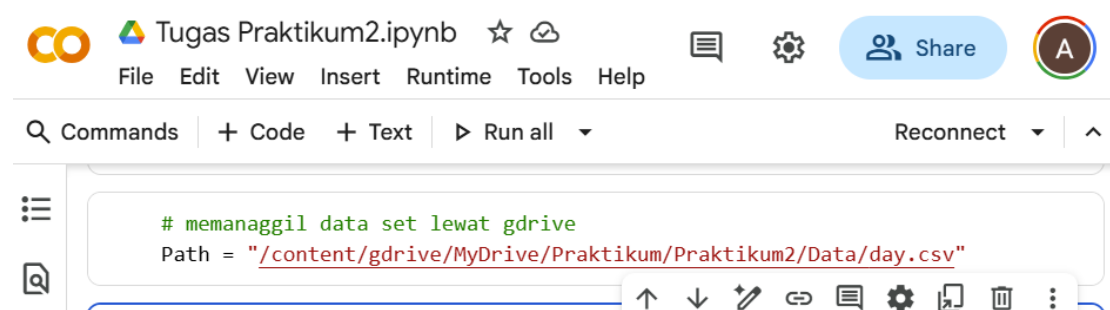
Mounted at /content/gdrive

Kode ini digunakan untuk **menghubungkan Google Colab dengan Google Drive** agar file di Drive bisa diakses dari Colab.

Kesimpulan:

Output Mounted at /content/gdrive menandakan Drive berhasil terhubung dan siap digunakan untuk membaca dataset.

Kode & Penjelasan



```
CO Tugas Praktikum2.ipynb | ☆ | ☁ | File | Edit | View | Insert | Runtime | Tools | Help | Share | A
```

```
Q Commands | + Code | + Text | ▶ Run all | Reconnect | ^
```

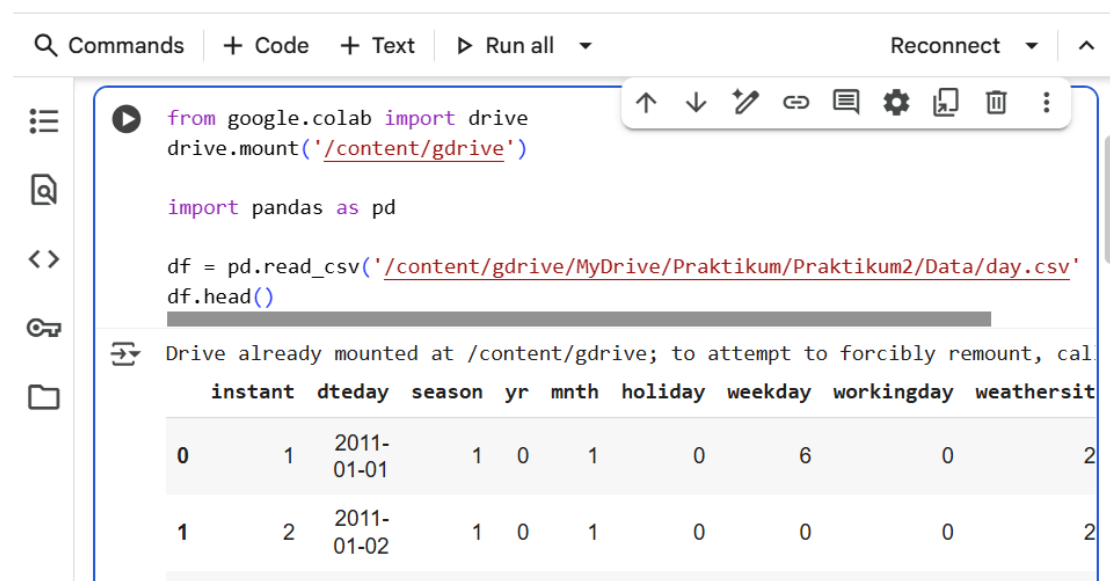
```
# memanaggil data set lewat gdrive
Path = "/content/gdrive/MyDrive/Praktikum/Praktikum2/Data/day.csv"
```

Kode ini menyimpan **lokasi lengkap file dataset** day.csv yang ada di Google Drive ke dalam variabel Path.

Kesimpulan:

Dengan membuat variabel Path, file day.csv bisa dipanggil dengan mudah saat proses pembacaan data di langkah berikutnya.

Kode & Penjelasan



```
from google.colab import drive
drive.mount('/content/gdrive')

import pandas as pd

df = pd.read_csv('/content/gdrive/MyDrive/Praktikum/Praktikum2/Data/day.csv')
df.head()
```

Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount('/content/gdrive', force_remount=True).

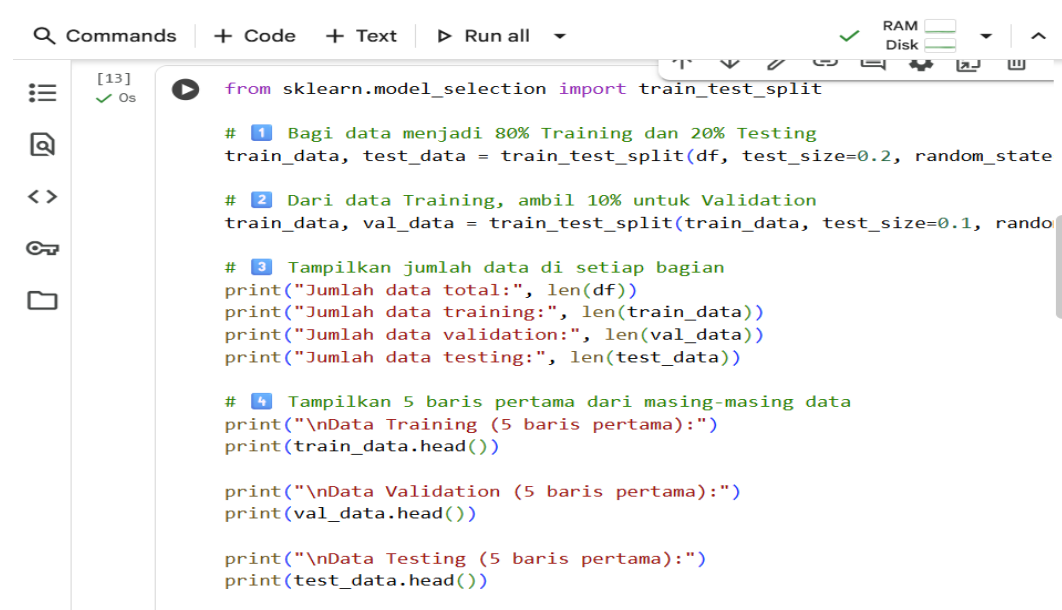
	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit
0	1	2011-01-01	1	0	1	0	6	0	2
1	2	2011-01-02	1	0	1	0	0	0	2

Kode ini menghubungkan Colab dengan Google Drive, lalu membaca dataset day.csv menggunakan pandas dan menampilkan 5 baris pertama data dengan .head().

Kesimpulan:

Dataset day.csv berhasil dibaca dan ditampilkan, menandakan data siap diproses untuk pembagian menjadi training, validation, dan testing.

Kode & Penjelasan



```
[13] ✓ 0s
from sklearn.model_selection import train_test_split

# 1 Bagi data menjadi 80% Training dan 20% Testing
train_data, test_data = train_test_split(df, test_size=0.2, random_state=42)

# 2 Dari data Training, ambil 10% untuk Validation
train_data, val_data = train_test_split(train_data, test_size=0.1, random_state=42)

# 3 Tampilkan jumlah data di setiap bagian
print("Jumlah data total:", len(df))
print("Jumlah data training:", len(train_data))
print("Jumlah data validation:", len(val_data))
print("Jumlah data testing:", len(test_data))

# 4 Tampilkan 5 baris pertama dari masing-masing data
print("\nData Training (5 baris pertama):")
print(train_data.head())

print("\nData Validation (5 baris pertama):")
print(val_data.head())

print("\nData Testing (5 baris pertama):")
print(test_data.head())
```

Kode ini menggunakan train_test_split untuk membagi dataset day.csv menjadi tiga bagian:

Training data (80%) untuk melatih model,

Validation data (10% dari training) untuk mengevaluasi model,

Testing data (20%) untuk menguji hasil akhir.

Lalu menampilkan jumlah dan contoh isi tiap bagian.

Kesimpulan:

Dataset berhasil dibagi dengan proporsi yang tepat. Setiap bagian sudah berisi data berbeda dan siap digunakan untuk proses pelatihan, validasi, dan pengujian model machine learning.