DATA ANALITIK DASAR

BAB 1 : Data Loading

1.1 Tujuan

Mahasiswa akan **memahami konsep, menerapkan teknik, dan memiliki dasar yang kuat** untuk melakukan Data Loading dalam berbagai proyek analisis data.

1.2 Ulasan Materi

A. Data Loading

1. Apa itu Data Loading?

Data Loading adalah proses **membawa data dari sumber eksternal** ke dalam sistem penyimpanan data atau platform analisis.

2. Mengapa Data Loading Penting?

- Data Loading menyediakan **bahan baku** untuk analisis data.
- Data Loading memungkinkan **penggabungan data** dari berbagai sumber untuk analisis yang lebih komprehensif.
- Data Loading memastikan **data yang dianalisis selalu terbaru** dan relevan.
- Data Loading **memfasilitasi pengambilan keputusan** yang didasarkan pada data yang akurat dan terkini.

3. Teknik Data Loading

1. Memuat Data dari CSV Menggunakan URL:

- Cara yang Anda gunakan dalam kode contoh, `pd.read_csv(url)`, efektif untuk memuat data dari file CSV yang disimpan di URL publik. Pastikan URL tersebut dapat diakses publik.

```
import pandas as pd

url =
"https://raw.githubusercontent.com/noora20FH/skripsi_noora2023/ma
ster/data_toko.csv"  # Replace with your actual URL

# Read the CSV directly from the URL
def data_load():
  data_toko = pd.read_csv(url)
  return data_toko
```

2. Memuat Data dari Dictionary:

- Anda dapat membuat dictionary Python yang berisi data Anda, lalu mengonversinya menjadi DataFrame menggunakan `pd.DataFrame(data)`. Ini berguna untuk data sederhana yang tidak terlalu besar.

```
import pandas as pd

# Membuat dictionary dengan data
data = {
    'Nama': ['Alice', 'Bob', 'Charlie'],
    'Usia': [25, 30, 35],
    'Kota': ['Jakarta', 'Bandung', 'Surabaya']
}

# Mengonversi dictionary menjadi DataFrame
df = pd.DataFrame(data)

# Menampilkan DataFrame
print(df)
```

3. Memuat Data dari File Lokal:

- Jika data Anda tersimpan dalam file CSV di komputer Anda, gunakan `pd.read_csv('filename.csv')`. Ganti 'filename.csv' dengan nama file aktual Anda.

```
import pandas as pd

# Memuat data dari file CSV
df = pd.read_csv('data.csv')

# Menampilkan DataFrame
print(df)
```

B. Menampilkan Shape dan Size Dataset

1. df.shape:

Metode ini mengembalikan tupel yang berisi jumlah baris (elemen pertama) dan jumlah kolom (elemen kedua) di DataFrame. DataFrame ini memiliki 50 baris (pengamatan) dan 11 kolom (fitur).

```
df.shape
```

Output:

```
(50, 4)
```

2. **len(df):**

   Fungsi ini mengembalikan jumlah baris dalam DataFrame. Ini cara yang lebih ringkas untuk menghitung jumlah baris. DataFrame ini memiliki 50 baris (pengamatan).

```
len(df)
```

Output:

```
50
```

3. **df.columns:**

   Metode ini mengembalikan daftar nama kolom di DataFrame. Ini memungkinkan Anda mengakses dan merujuk ke kolom tertentu dengan mudah.

```
df.columns
```

Output:

```
Index(['Duration', 'Pulse', 'Maxpulse', 'Calories'], dtype='object')
```

4. **df.size:**

   Metode ini mengembalikan jumlah total elemen (titik data) di DataFrame, termasuk semua baris dan kolom. DataFrame ini memiliki total 550 titik data (50 baris * 11 kolom)..

```
df.size
```

Output:

```
200
```

C. Menampilkan Dataset

1. **df.head():**
   Metode ini menampilkan beberapa baris pertama (default: 5) dari DataFrame.

```
df.head()
```

Output:

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0

2. **df.tail():**
   Metode ini menampilkan beberapa baris terakhir (default: 5) dari DataFrame. Ini memberikan pandangan sekilas pada catatan terbawah.

```
df.tail()
```

Output:

	Duration	Pulse	Maxpulse	Calories
164	60	105	140	290.8
165	60	110	145	300.0
166	60	115	145	310.2
167	75	120	150	320.4
168	75	125	150	330.4

3. **df.sample():**
   Metode ini secara acak memilih sejumlah baris tertentu (default: 1) dari DataFrame. Ini memberikan contoh data tanpa menampilkan semua baris.

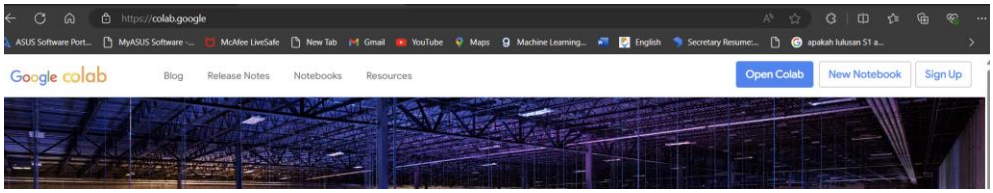
```
df.sample()
```

Output:

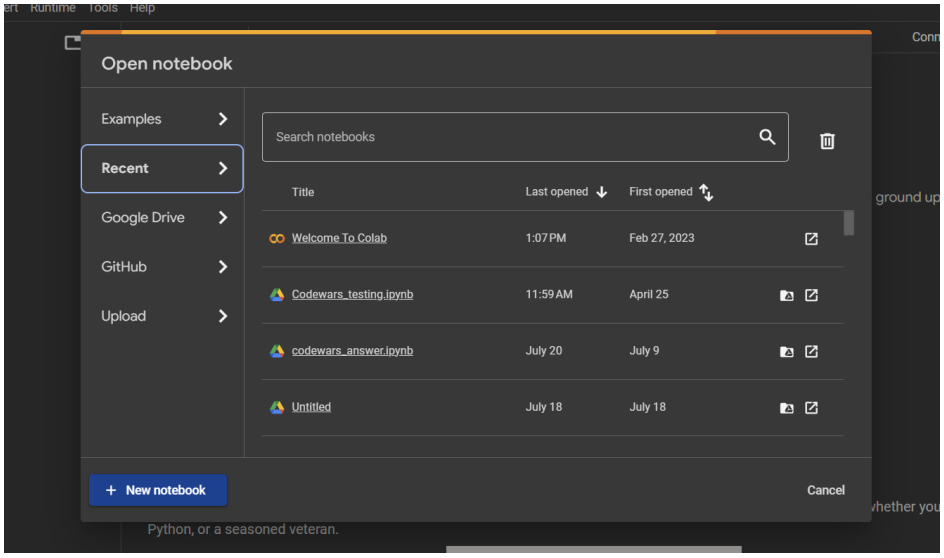
	Duration	Pulse	Maxpulse	Calories
153	30	150	167	275.8

1.3 Langkah Persiapan

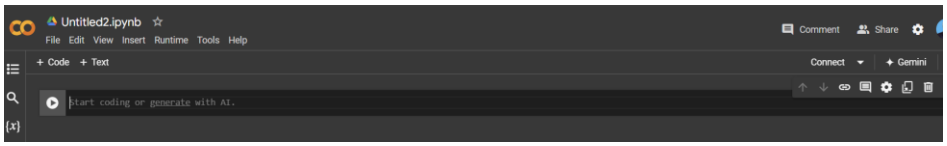
1. Unduh file testing dengan format Python (.py) pada link Google drive berikut ini, https://drive.google.com/drive/folders/1vELIyfN3MQDoTSnug9ChqyUuz61Q0X35?usp=sharing
2. Membuka Google Colab
    a. Buka Google Colaboratory dengan link berikut https://colab.research.google.com/ .
    b. Klik Open Colab di pojok kanan atas



    c. Anda bisa login menggunakan akun Google.
    d. Klik New Notebook pada pojok kiri bawah, untuk membuka halaman baru google colab.



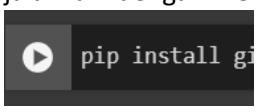
    e. Tampilan Google Colab.



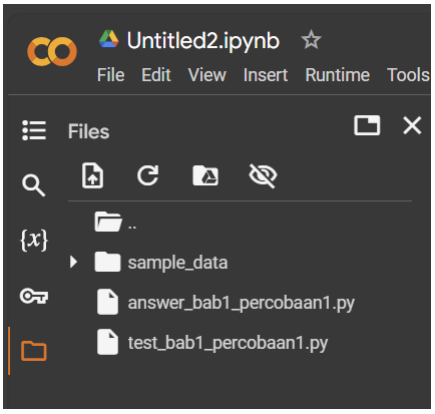
3. Menjalankan install codewars_test pada cell google colab
codewars_test adalah framework yang digunakan untuk menjalankan validasi kode jawaban.
    a. Jalankan command berikut:

```
pip install git+https://github.com/codewars/python-test-framework.git#egg=codewars_test
```

    b. jalankan dengan menekan tombol play



4. Menjalankan test Validasi untuk hasil kode jawaban
    a. Upload file testing file untuk validasi jawaban dan file berisi kode hasil dari jawaban praktikum pada modul. Klik pada ikon files.



    b. Jalankan file testing pada cell google colab untuk memvalidasi kode hasil jawaban praktikum.
    c. Ketikkan command berikut, lalu jalankan cell:

```
!python /content/test_bab1_percobaan1.py
```

Pada nama file testing test_bab1_percobaan 1.py, sesuaikan dengan nama file test yang akan dijalankan sesuai dengan modul yang telah dikerjakan.

    d. Berikut adalah output hasil dari validasi
PASSED menandakan case yang dikerjakan berhasil, sedangkan FAILED menandakan case yang dikerjakan gagal dan perlu perbaikan.

```
<DESCRIBE::>BAB 1 | Percobaan 1
======================================================================

<IT::>1. Test Memuat Data

<PASSED::>Test Passed

<PASSED::>Test Passed

<COMPLETEDIN::>50.77

<IT::>2. Test Print Nilai Fungsi data_load()
======================================================================

<FAILED::>====> Error :Tidak Menampilkan nilai fungsi data_load() menggunakan print(): False should equal True

<COMPLETEDIN::>0.03

<COMPLETEDIN::>50.88
```

1.4 Contoh Studi Kasus

Memuat Data dari CSV File Lokal

Scenario: Dataset diambil dari direktori github. Muat dataset dan baca data dari CSV format menjadi Pandas DataFrame objek.

Objective:

- Gunakan library Pandas untuk memuat dataset
- Memuat data csv dari file lokal

Langkah - Langkah:

1. **Import Pandas libraries:**

```
import pandas as pd
```

2. **Load Data:**

2.1 Membuat fungsi data_load(). Dalam fungsi data_load(), tentukan variabel df untuk menyimpan pd.read_csv(nama file.csv) untuk membaca data dari file lokal CSV.

```
def data_load():
  df = pd.read_csv('data.csv')
  return df
```

2.2 Panggil fungsi **data_load()** untuk menjalankan program

```
data_load()
```

Tampilan Keseluruhan kode:

```
import pandas as pd

def data_load():
  df = pd.read_csv('data.csv')
  return df

data_load()
```

1.5 Praktikum

Memuat Data dari CSV menggunakan URL

Skenario: Dataset diambil dari direktori github. Muat dataset dan baca data dari CSV format menjadi Pandas DataFrame objek.

Objektif:

- Gunakan library Pandas untuk memuat dataset
- Memuat data csv dari url

Langkah-Langkah:

1. **Import Library:**
   - **import pandas as pd**.
2. **Load Data:**
   - 2.1 Definisikan variabel **url** ,
   - 2.2 Gunakan link di bawah ini sebagai nilai dari **url** variabel https://raw.githubusercontent.com/noora20FH/skripsi_noora2023/main/purchases.csv
   - 2.3 Buatlah fungsi **data_load().**
   - 2.4 Definisikan variabel **df** di dalam fungsi **data_load().** Gunakan **pd.read_csv(url)** membaca data CSV.
   - 2.5 print fungsi **load_data()** untuk menggunakan **print().**