PART 1

DATA INGESTION

STUDENT : WARTADI
MENTOR : BILAL BENEFIT

TASK

alterra

TASK

1. We have already learned how to create DataFrame from files here. Now, we are going to create a DataFrame from a larger csv file on our datasets.
2. Rename all the columns with snake_case format.
3. Select only 10 top of highest number of `passenger_count`, show only columns `vendor_id, passenger_count, trip_distance, payment_type, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge, total_amount, congestion_surcharge` from the DataFrame.
4. [Extra] Cast the data type to the appropriate value.

Setup Environment & Install Requirements.txt

Hal yang pertama dilakukan untuk mengerjakan tugas adalah clone repository yang sudah disediakan

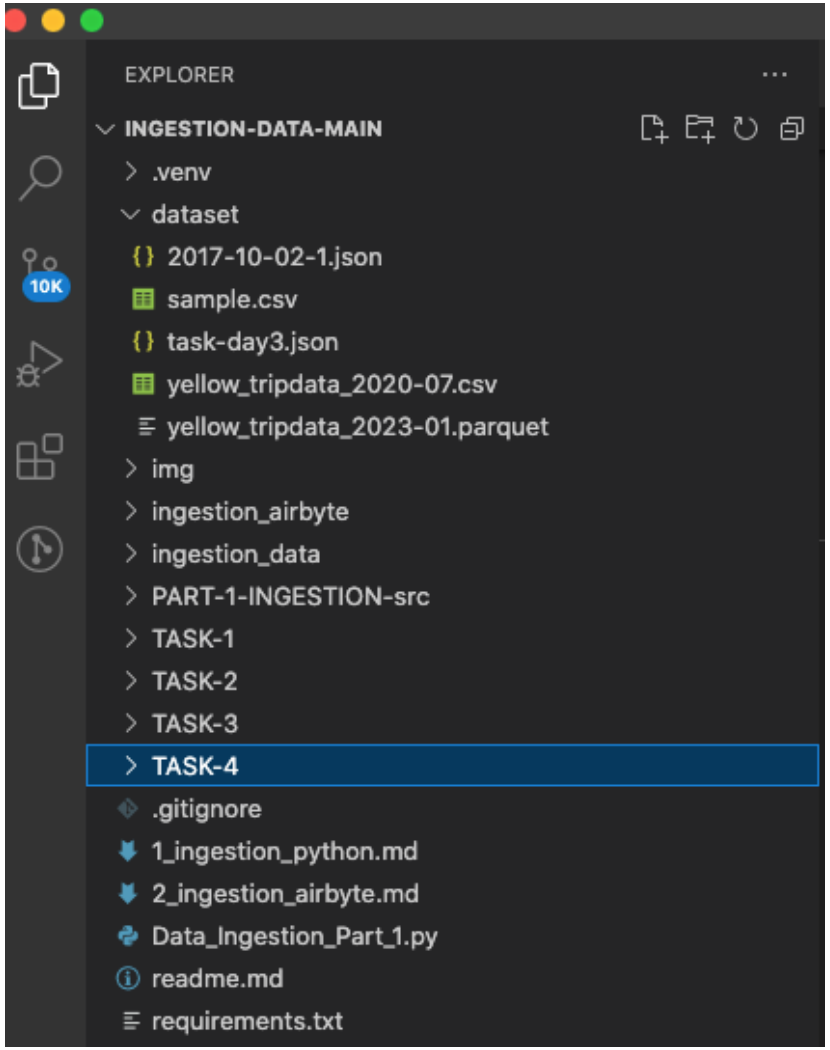
git clone https://github.com/Immersive-DataEngineer-Resource/ingestion-data.git

Bisa dilihat pada gambar disamping jika virtual environtment sudah terinstall maka setelah itu bisa menggunakan perintah

source .venv/bin/activate

```
(.venv) wartadi@Wartadis-MacBook-Pro ingestion-data-main %
```

pip install –r requirements.
Daftar requirements.txt mencakup paket-paket yang diperlukan untuk



TASK 1

Input Program

```python
def main():
    # 1. Memuat dataset dengan low_memory=False untuk menangani peringatan tipe campuran
    df = pd.read_csv(os.path.join(os.path.dirname(__file__), "../dataset/yellow_tripdata_2020-07.csv"), low_memory=Fals
```

Dataset dimuat dari file yellow_tripdata_2020-07.csv.Parameter low_memory=False digunakan untuk menangani peringatan tipe data campuran yang mungkin muncul saat membaca file besar.

Output

```
(.venv) wartadi@Wartadis-MacBook-Pro PART-1-INGESTION-src % python tugas_part1_ingestion.py
1. DataFrame setelah memuat dataset:
+-------+----------+----------------------+-----------------------+-----------------+---------------+--------------+--------------------+-------------------------+
| Index | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance | RatecodeID | store_and_fwd_flag |      Other columns      |
+-------+----------+----------------------+-----------------------+-----------------+---------------+--------------+--------------------+-------------------------+
|   0   |   1.0    | 2020-07-01 00:25:32  |  2020-07-01 00:33:39  |       1.0       |      1.5      |     1.0      |         N          | Other columns (11 more) |
|   1   |   1.0    | 2020-07-01 00:03:19  |  2020-07-01 00:25:43  |       1.0       |      9.5      |     1.0      |         N          | Other columns (11 more) |
|   2   |   2.0    | 2020-07-01 00:15:11  |  2020-07-01 00:29:24  |       1.0       |     5.85      |     1.0      |         N          | Other columns (11 more) |
|   3   |   2.0    | 2020-07-01 00:30:49  |  2020-07-01 00:38:26  |       1.0       |      1.9      |     1.0      |         N          | Other columns (11 more) |
|   4   |   2.0    | 2020-07-01 00:31:26  |  2020-07-01 00:38:02  |       1.0       |     1.25      |     1.0      |         N          | Other columns (11 more) |
+-------+----------+----------------------+-----------------------+-----------------+---------------+--------------+--------------------+-------------------------+

Showing 7 columns out of 18 total columns.
Hidden columns (11): PULocationID, DOLocationID, payment_type, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge, total_amount, congestion_s
urcharge
```

TASK 2

Input Program

```
# 2. Mengganti nama kolom menjadi snake_case
df.columns = df.columns.str.lower().str.replace(' ', '_')

# Menampilkan nama-nama kolom setelah diganti menjadi snake_case
print("2. Nama kolom setelah diganti menjadi snake_case:")
print(df.columns.tolist())
print("\n")
```

Nama kolom diubah menjadi huruf kecil dan format snake_case (mengganti spasi dengan underscore _).

Output Program

```
2. Nama kolom setelah diganti menjadi snake_case:
['vendorid', 'tpep_pickup_datetime', 'tpep_dropoff_datetime', 'passenger_count', 'trip_distance', 'ratecodeid', 'store_and_fwd_flag', 'pulocationid', 'dolocationid', '
payment_type', 'fare_amount', 'extra', 'mta_tax', 'tip_amount', 'tolls_amount', 'improvement_surcharge', 'total_amount', 'congestion_surcharge']
```

TASK 3

Input Program

```
# 3. Memilih kolom-kolom yang relevan dan 10 baris teratas dengan jumlah penumpang tertinggi
selected_columns = [
    'vendorid', 'passenger_count', 'trip_distance', 'payment_type',
    'fare_amount', 'extra', 'mta_tax', 'tip_amount', 'tolls_amount',
    'improvement_surcharge', 'total_amount', 'congestion_surcharge'
]
df_selected = df[selected_columns]

# Memilih 10 baris teratas dengan jumlah penumpang tertinggi
top_10_passenger_count = df_selected.nlargest(10, 'passenger_count')

print("3. 10 baris teratas dengan jumlah penumpang tertinggi:")
print_table_with_summary(top_10_passenger_count)
print("\n")
```

- Beberapa kolom yang relevan dipilih dari DataFrame asli.
- 10 baris dengan jumlah penumpang tertinggi dipilih menggunakan nlargest.

Output Program

```
3. 10 baris teratas dengan jumlah penumpang tertinggi:
```

Index	vendorid	passenger_count	trip_distance	payment_type	fare_amount	extra	mta_tax	tip_amount	tolls_amount	Other columns
214141	2.0	9.0	0.0	1.0	9.8	0.0	0.0	1.96	0.0	Other columns (3 more)
79823	2.0	8.0	0.0	1.0	8.5	0.0	0.0	1.0	0.0	Other columns (3 more)
737023	2.0	8.0	0.0	1.0	8.0	0.0	0.0	0.0	0.0	Other columns (3 more)
164792	2.0	7.0	0.0	1.0	7.0	0.0	0.5	1.0	0.0	Other columns (3 more)
385688	2.0	7.0	0.0	2.0	7.3	0.0	0.5	0.0	0.0	Other columns (3 more)
385689	2.0	7.0	0.0	1.0	7.3	0.0	0.5	3.18	0.0	Other columns (3 more)
690829	2.0	7.0	0.0	1.0	7.0	0.0	0.5	1.0	0.0	Other columns (3 more)
732901	2.0	7.0	10.84	1.0	70.0	0.0	0.0	0.0	0.0	Other columns (3 more)
65	2.0	6.0	0.73	2.0	4.0	0.5	0.5	0.0	0.0	Other columns (3 more)
144	2.0	6.0	1.09	1.0	5.0	0.5	0.5	2.64	0.0	Other columns (3 more)

```
Showing 9 columns out of 12 total columns.
Hidden columns (3): improvement_surcharge, total_amount, congestion_surcharge
```

Task 4

Input Program

```
# 4. Mengubah tipe data dan menampilkan DataFrame setelah casting
print("4. Nilai NaN sebelum mengubah tipe data:")
print(df_selected.isna().sum())
print("\n")

# Menggunakan .loc untuk mengisi nilai NaN dan mengubah tipe
df_selected.loc[:, 'vendorid'] = df_selected['vendorid'].fillna(0).astype(int)
df_selected.loc[:, 'passenger_count'] = df_selected['passenger_count'].fillna(0).astype(int)

float_columns = [
    'trip_distance', 'fare_amount', 'extra', 'mta_tax', 'tip_amount',
    'tolls_amount', 'improvement_surcharge', 'total_amount', 'congestion_surcharge'
]
for col in float_columns:
    df_selected.loc[:, col] = df_selected[col].fillna(0.0).astype(float)

# Menampilkan DataFrame setelah mengubah tipe data
print("4. DataFrame setelah mengubah tipe data:")
print_table_with_summary(df_selected.head())
print("\n")
```

Nilai NaN di kolom vendorid dan passenger_count diisi dengan 0 dan diubah menjadi tipe data int.Kolom-kolom yang seharusnya bertipe float juga diisi dengan 0.0 dan diubah menjadi tipe float.

Task 4

Output Program

```
4. Nilai NaN sebelum mengubah tipe data:
vendorid                62847
passenger_count         62847
trip_distance               0
payment_type            62847
fare_amount                 0
extra                       0
mta_tax                     0
tip_amount                  0
tolls_amount                0
improvement_surcharge       0
total_amount                0
congestion_surcharge        0
dtype: int64


4. DataFrame setelah mengubah tipe data:
```

Index	vendorid	passenger_count	trip_distance	payment_type	fare_amount	extra	mta_tax	tip_amount	tolls_amount	Other columns
0	1.0	1.0	1.5	2.0	8.0	0.5	0.5	0.0	0.0	Other columns (3 more)
1	1.0	1.0	9.5	1.0	26.5	0.5	0.5	0.0	0.0	Other columns (3 more)
2	2.0	1.0	5.85	2.0	18.5	0.5	0.5	0.0	0.0	Other columns (3 more)
3	2.0	1.0	1.9	1.0	8.0	0.5	0.5	2.36	0.0	Other columns (3 more)
4	2.0	1.0	1.25	2.0	6.5	0.5	0.5	0.0	0.0	Other columns (3 more)

```
Showing 9 columns out of 12 total columns.
Hidden columns (3): improvement_surcharge, total_amount, congestion_surcharge
```

alterra

THANK YOU ☺