

DATA INGESTION

TASK-2

1. We are going to create a DataFrame from a [parquet file](#) on our [datasets](#).

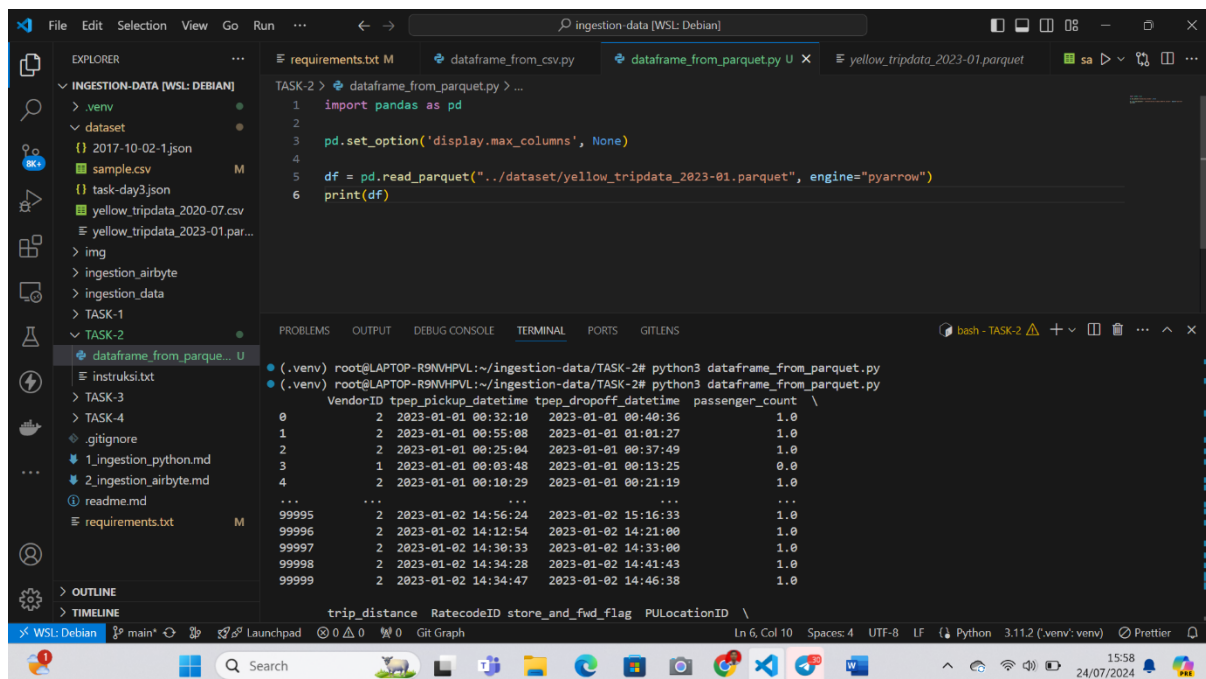
Jawaban :

```
import pandas as pd

pd.set_option('display.max_columns', None)

df = pd.read_parquet("../dataset/yellow_tripdata_2023-01.parquet",
engine="pyarrow")
print(df)
```

Screenshot:



The screenshot shows a VS Code editor window with a file explorer on the left and a terminal at the bottom. The file explorer shows a project named 'INGESTION-DATA [WSL: DEBIAN]' with a folder 'dataset' containing 'yellow_tripdata_2023-01.parquet'. The terminal shows the execution of a Python script 'dataframe_from_parquet.py' which reads the parquet file using the 'pyarrow' engine and prints the first 5 rows of the resulting DataFrame.

```
TASK-2 > dataframe_from_parquet.py > ...
1 import pandas as pd
2
3 pd.set_option('display.max_columns', None)
4
5 df = pd.read_parquet("../dataset/yellow_tripdata_2023-01.parquet", engine="pyarrow")
6 print(df)
```

```
bash - TASK-2
(.venv) root@LAPTOP-R9MVHPVL:~/ingestion-data/TASK-2# python3 dataframe_from_parquet.py
(.venv) root@LAPTOP-R9MVHPVL:~/ingestion-data/TASK-2# python3 dataframe_from_parquet.py
VendorID tpep_pickup_datetime tpep_dropoff_datetime passenger_count \
0 2 2023-01-01 00:32:10 2023-01-01 00:40:36 1.0
1 2 2023-01-01 00:35:08 2023-01-01 01:01:27 1.0
2 2 2023-01-01 00:25:04 2023-01-01 00:37:49 1.0
3 1 2023-01-01 00:03:48 2023-01-01 00:13:25 0.0
4 2 2023-01-01 00:10:29 2023-01-01 00:21:19 1.0
... ..
99995 2 2023-01-02 14:56:24 2023-01-02 15:16:33 1.0
99996 2 2023-01-02 14:12:54 2023-01-02 14:21:00 1.0
99997 2 2023-01-02 14:30:33 2023-01-02 14:33:00 1.0
99998 2 2023-01-02 14:34:28 2023-01-02 14:41:43 1.0
99999 2 2023-01-02 14:34:47 2023-01-02 14:46:38 1.0
```

2. Load the parquet file to a DataFrame with `fastparquet` library.

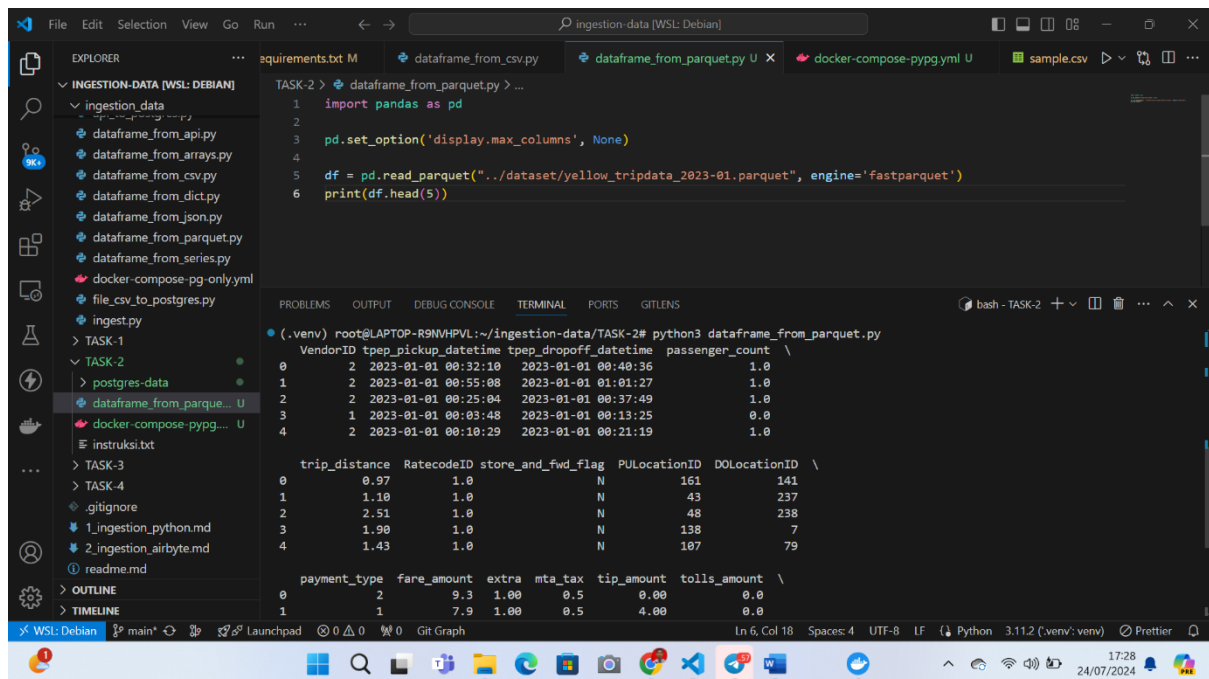
Jawaban :

```
import pandas as pd

pd.set_option('display.max_columns', None)

df = pd.read_parquet("../dataset/yellow_tripdata_2023-01.parquet",
engine='fastparquet')
print(df.head(5))
```

Screenshot :



3. Clean the Yellow Trip dataset.

Jawaban :

```
import pandas as pd

pd.set_option('display.max_columns', None)

df = pd.read_parquet("../dataset/yellow_tripdata_2023-01.parquet",
engine='fastparquet')
#print(df.head(5))

df.columns = [col.lower().replace(' ', '_') for col in df.columns]

df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'])
df['tpep_dropoff_datetime'] = pd.to_datetime(df['tpep_dropoff_datetime'])
df['passenger_count'] = df['passenger_count'].astype(int)
df['trip_distance'] = df['trip_distance'].astype(float)

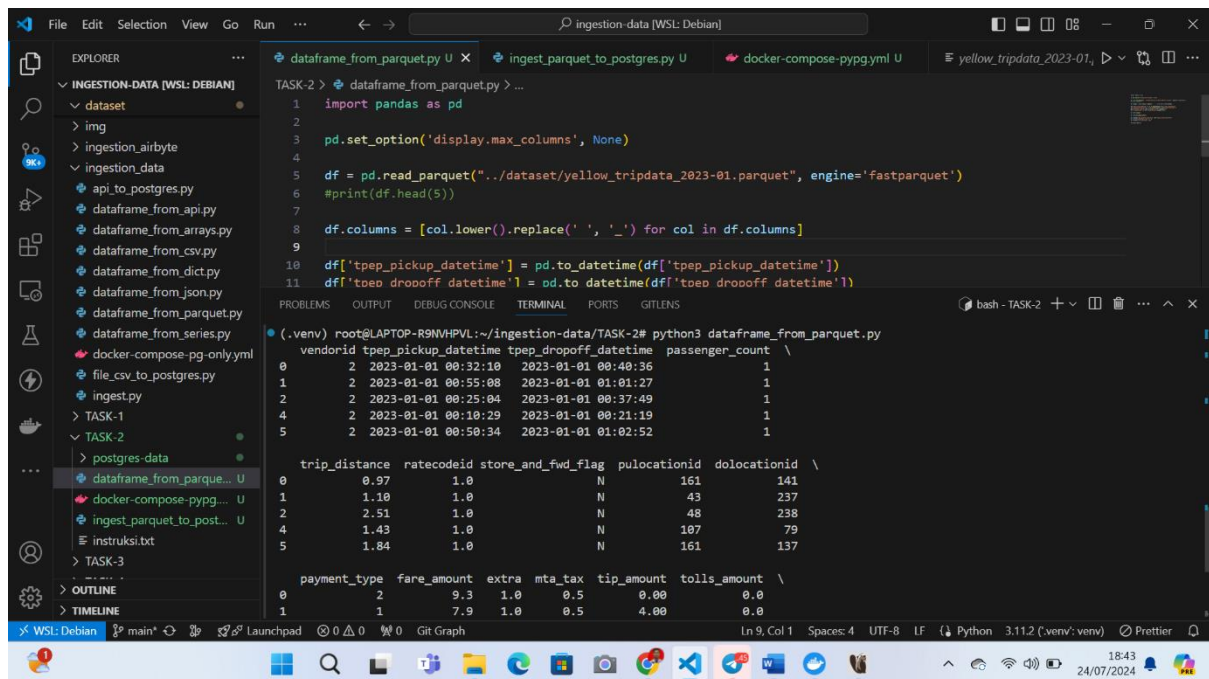
df = df.dropna()

df = df.drop_duplicates()

df = df[df['tpep_dropoff_datetime'] > df['tpep_pickup_datetime']]
df = df[df['passenger_count'] > 0]
df = df[df['trip_distance'] > 0]

print(df.head(5))
```

Screenshot :



4. Define the data type schema when using `to_sql` method.

Jawaban :

```
def to_postgres(self, db_name: str, data: pd.DataFrame):
    from sqlalchemy.types import BigInteger, String, JSON, DateTime,
    Boolean, Float, Integer
    from sqlalchemy.exc import SQLAlchemyError

    self.db_name = db_name
    self.__create_connection()

    try:
        df_schema = {
            "VendorID": BigInteger,
            "tpep_pickup_datetime": DateTime,
            "tpep_dropoff_datetime": DateTime,
            "passenger_count": BigInteger,
            "trip_distance": Float,
            "RatecodeID": Float,
            "store_and_fwd_flag": Boolean,
            "PULocationID": Integer,
            "DOLocationID": Integer,
            "payment_type": Integer,
            "fare_amount": Float,
            "extra": Float,
            "mta_tax": Float,
            "tip_amount": Float,
            "tolls_amount": Float,
```

```

        "improvement_surcharge": Float,
        "total_amount": Float,
        "congestion_surcharge": Float,
        "airport_fee": Float
    }

    data.to_sql(name=self.db_name, con=self.engine,
if_exists="replace", index=False, schema="public", dtype=df_schema,
method=None, chunksize=5000)
    except SQLAlchemyError as err:
        print("error >> ", err.__cause__)

```

5. Ingest the Yellow Trip dataset to PostgreSQL

Jawaban :

```

import pandas as pd

class Extraction():
    def __init__(self) -> None:
        self.path: str
        self.url: str
        self.dataframe = pd.DataFrame()

    def local_file(self, path: str):
        self.path = path
        self.extension = self.__ext_checker()
        self.__read_parquet()
        self.investigate_schema()
        self.cast_data()

        return self.dataframe

    def __ext_checker(self) -> str:
        return self.path.split(".")[1]

    def __read_parquet(self) -> pd.DataFrame:
        """
        problem: DtypeWarning: Columns (6) have mixed types.Specify dtype
option on import or set low_memory=False.
        to solve specify schema

        """
        self.dataframe = pd.read_parquet(self.path)

    def investigate_schema(self):
        pd.set_option('display.max_columns', None)

```

```

        print(self.dataframe["store_and_fwd_flag"])

    def cast_data(self):
        # file parquet cast data handler
        self.dataframe["passenger_count"] =
self.dataframe["passenger_count"].astype("Int8")

        self.dataframe["store_and_fwd_flag"] =
self.dataframe["store_and_fwd_flag"].replace(["N", "Y"], [False, True])
        self.dataframe["store_and_fwd_flag"] =
self.dataframe["store_and_fwd_flag"].astype("boolean")

        self.dataframe["tpep_pickup_datetime"] =
pd.to_datetime(self.dataframe["tpep_pickup_datetime"])
        self.dataframe["tpep_dropoff_datetime"] =
pd.to_datetime(self.dataframe["tpep_dropoff_datetime"])

class Load():
    # https://www.geeksforgeeks.org/how-to-insert-a-pandas-dataframe-to-an-
existing-postgresql-table/
    def __init__(self) -> None:
        self.df = pd.DataFrame
        self.db_name = ""
        self.engine = None
        self.connection = None

    def __create_connection(self):
        from sqlalchemy import create_engine

        user = "postgres"
        password = "admin"
        host = "localhost"
        database = "mydb"
        port = 5432
        conn_string =
f"postgresql://{user}:{password}@{host}:{port}/{database}"

        self.engine = create_engine(conn_string)

    def to_postgres(self, db_name: str, data: pd.DataFrame):
        from sqlalchemy.types import BigInteger, String, JSON, DateTime,
Boolean, Float, Integer
        from sqlalchemy.exc import SQLAlchemyError

        self.db_name = db_name
        self.__create_connection()

```

```

try:
    df_schema = {
        "VendorID": BigInteger,
        "tpep_pickup_datetime": DateTime,
        "tpep_dropoff_datetime": DateTime,
        "passenger_count": BigInteger,
        "trip_distance": Float,
        "RatecodeID": Float,
        "store_and_fwd_flag": Boolean,
        "PULocationID": Integer,
        "DOLocationID": Integer,
        "payment_type": Integer,
        "fare_amount": Float,
        "extra": Float,
        "mta_tax": Float,
        "tip_amount": Float,
        "tolls_amount": Float,
        "improvement_surcharge": Float,
        "total_amount": Float,
        "congestion_surcharge": Float,
        "airport_fee": Float
    }

    data.to_sql(name=self.db_name, con=self.engine,
if_exists="replace", index=False, schema="public", dtype=df_schema,
method=None, chunksize=5000)
    except SQLAlchemyError as err:
        print("error >> ", err.__cause__)

def main():

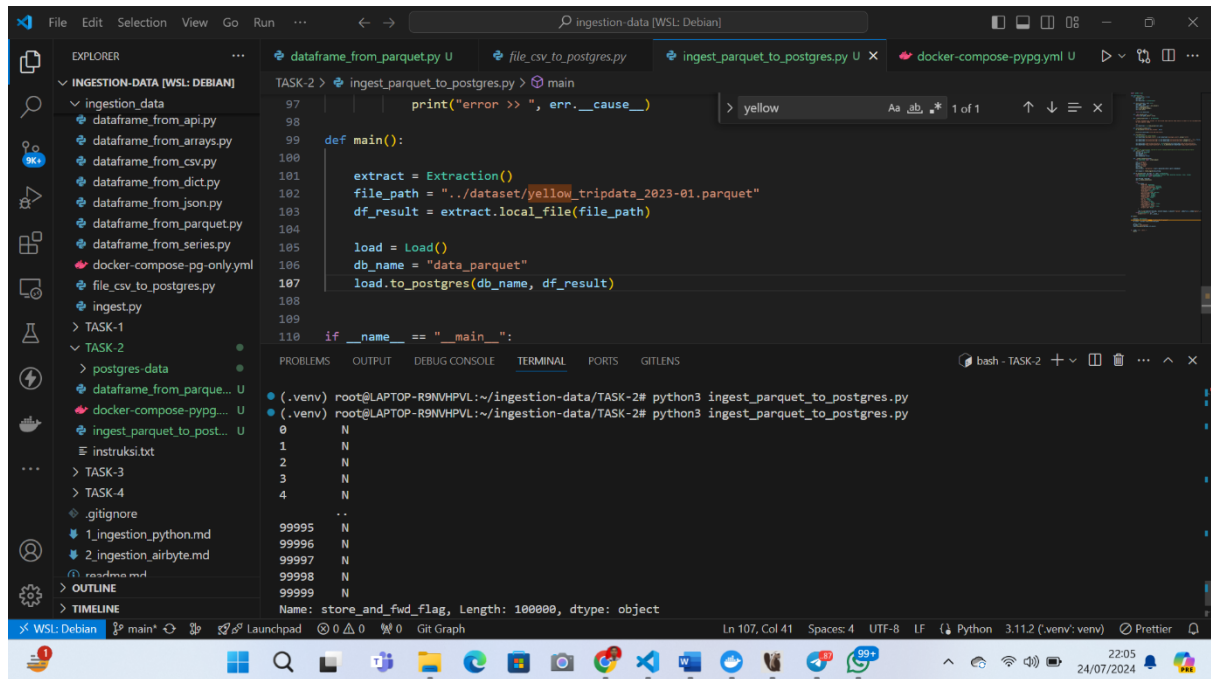
    extract = Extraction()
    file_path = "../dataset/yellow_tripdata_2023-01.parquet"
    df_result = extract.local_file(file_path)

    load = Load()
    db_name = "data_parquet"
    load.to_postgres(db_name, df_result)

if __name__ == "__main__":
    main()

```

Screenshot :



6. Count how many rows are ingested.

Jawabannya : 100.000

