PART 2

DATA INGESTION

STUDENT : WARTADI
MENTOR : BILAL BENEFIT

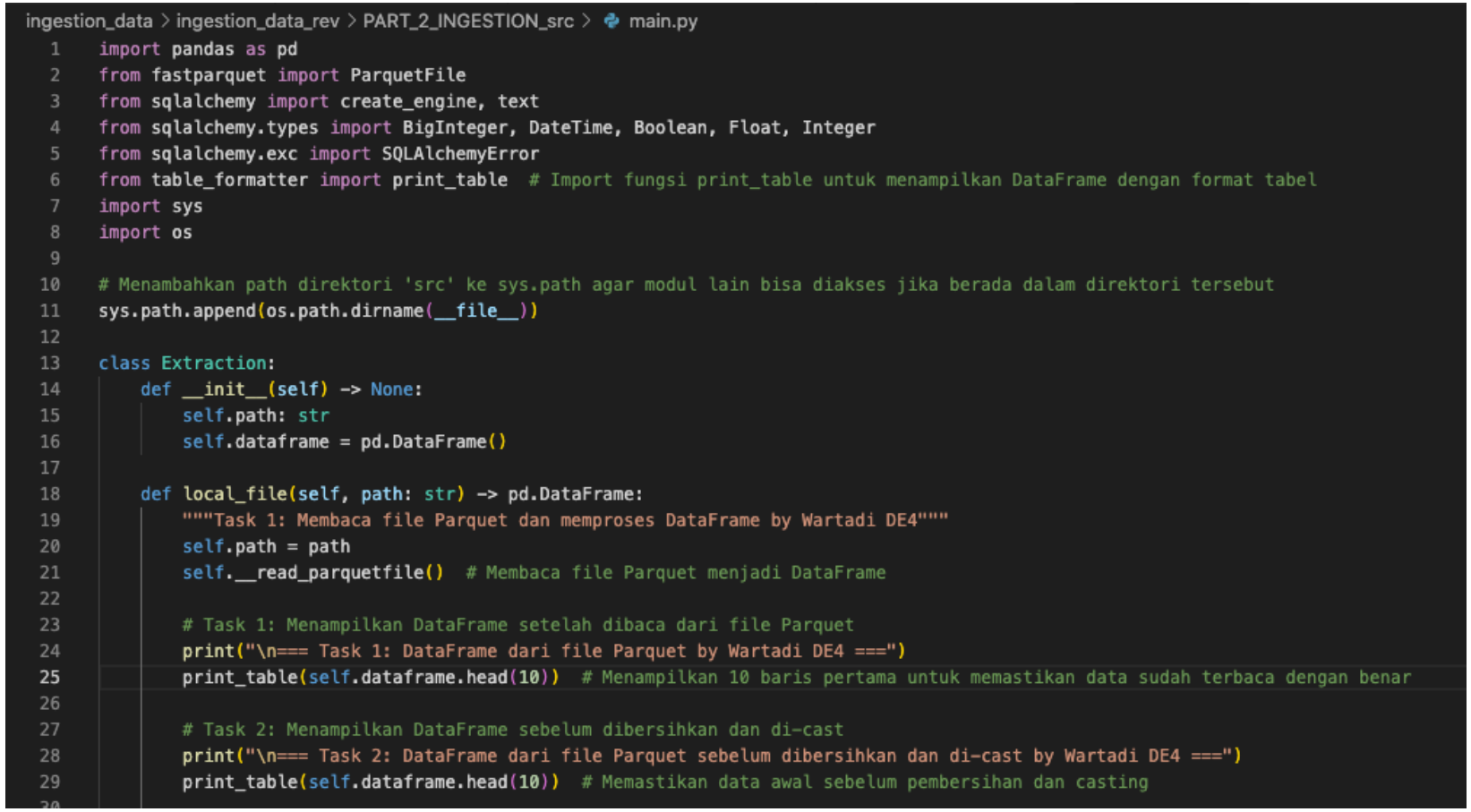
TASK

alterra

4. Task

1. We are going to create a DataFrame from a parquet file on our datasets.
2. Load the parquet file to a DataFrame with `fastparquet` library.
3. Clean the Yellow Trip dataset.
4. Define the data type schema when using `to_sql` method.
5. Ingest the Yellow Trip dataset to PostgreSQL
6. Count how many rows are ingested.

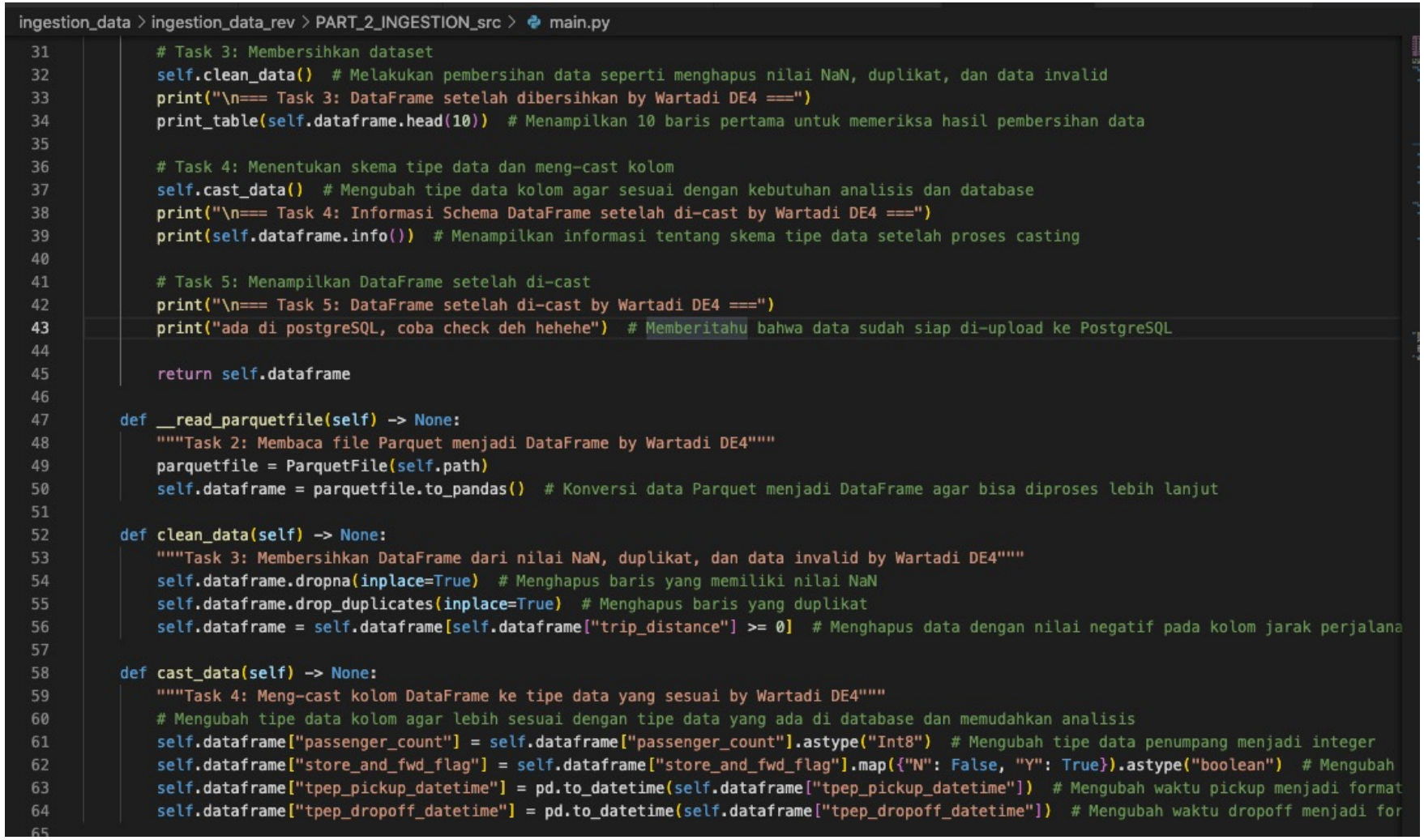
Input Program & Deskripsi



ingestion_data > ingestion_data_rev > PART_2_INGESTION_src > main.py

```python
import pandas as pd
from fastparquet import ParquetFile
from sqlalchemy import create_engine, text
from sqlalchemy.types import BigInteger, DateTime, Boolean, Float, Integer
from sqlalchemy.exc import SQLAlchemyError
from table_formatter import print_table  # Import fungsi print_table untuk menampilkan DataFrame dengan format tabel
import sys
import os

# Menambahkan path direktori 'src' ke sys.path agar modul lain bisa diakses jika berada dalam direktori tersebut
sys.path.append(os.path.dirname(__file__))

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

    def local_file(self, path: str) -> pd.DataFrame:
        """Task 1: Membaca file Parquet dan memproses DataFrame by Wartadi DE4"""
        self.path = path
        self.__read_parquetfile()  # Membaca file Parquet menjadi DataFrame

        # Task 1: Menampilkan DataFrame setelah dibaca dari file Parquet
        print("\n=== Task 1: DataFrame dari file Parquet by Wartadi DE4 ===")
        print_table(self.dataframe.head(10))  # Menampilkan 10 baris pertama untuk memastikan data sudah terbaca dengan benar

        # Task 2: Menampilkan DataFrame sebelum dibersihkan dan di-cast
        print("\n=== Task 2: DataFrame dari file Parquet sebelum dibersihkan dan di-cast by Wartadi DE4 ===")
        print_table(self.dataframe.head(10))  # Memastikan data awal sebelum pembersihan dan casting
```

Input Program & Deskripsi



ingestion_data > ingestion_data_rev > PART_2_INGESTION_src > main.py

```
31          # Task 3: Membersihkan dataset
32          self.clean_data()  # Melakukan pembersihan data seperti menghapus nilai NaN, duplikat, dan data invalid
33          print("\n=== Task 3: DataFrame setelah dibersihkan by Wartadi DE4 ===")
34          print_table(self.dataframe.head(10))  # Menampilkan 10 baris pertama untuk memeriksa hasil pembersihan data
35
36          # Task 4: Menentukan skema tipe data dan meng-cast kolom
37          self.cast_data()  # Mengubah tipe data kolom agar sesuai dengan kebutuhan analisis dan database
38          print("\n=== Task 4: Informasi Schema DataFrame setelah di-cast by Wartadi DE4 ===")
39          print(self.dataframe.info())  # Menampilkan informasi tentang skema tipe data setelah proses casting
40
41          # Task 5: Menampilkan DataFrame setelah di-cast
42          print("\n=== Task 5: DataFrame setelah di-cast by Wartadi DE4 ===")
43          print("ada di postgreSQL, coba check deh hehehe")  # Memberitahu bahwa data sudah siap di-upload ke PostgreSQL
44
45          return self.dataframe
46
47      def __read_parquetfile(self) -> None:
48          """Task 2: Membaca file Parquet menjadi DataFrame by Wartadi DE4"""
49          parquetfile = ParquetFile(self.path)
50          self.dataframe = parquetfile.to_pandas()  # Konversi data Parquet menjadi DataFrame agar bisa diproses lebih lanjut
51
52      def clean_data(self) -> None:
53          """Task 3: Membersihkan DataFrame dari nilai NaN, duplikat, dan data invalid by Wartadi DE4"""
54          self.dataframe.dropna(inplace=True)  # Menghapus baris yang memiliki nilai NaN
55          self.dataframe.drop_duplicates(inplace=True)  # Menghapus baris yang duplikat
56          self.dataframe = self.dataframe[self.dataframe["trip_distance"] >= 0]  # Menghapus data dengan nilai negatif pada kolom jarak perjalana
57
58      def cast_data(self) -> None:
59          """Task 4: Meng-cast kolom DataFrame ke tipe data yang sesuai by Wartadi DE4"""
60          # Mengubah tipe data kolom agar lebih sesuai dengan tipe data yang ada di database dan memudahkan analisis
61          self.dataframe["passenger_count"] = self.dataframe["passenger_count"].astype("Int8")  # Mengubah tipe data penumpang menjadi integer
62          self.dataframe["store_and_fwd_flag"] = self.dataframe["store_and_fwd_flag"].map({"N": False, "Y": True}).astype("boolean")  # Mengubah
63          self.dataframe["tpep_pickup_datetime"] = pd.to_datetime(self.dataframe["tpep_pickup_datetime"])  # Mengubah waktu pickup menjadi format
64          self.dataframe["tpep_dropoff_datetime"] = pd.to_datetime(self.dataframe["tpep_dropoff_datetime"])  # Mengubah waktu dropoff menjadi for
65
```

Input Program & Deskripsi

```python
class Load:
    def __init__(self) -> None:
        self.engine = None

    def __create_connection(self) -> None:
        """Task 5: Membuat koneksi ke database PostgreSQL by Wartadi DE4"""
        user = "postgres"
        password = "admin"
        host = "localhost"
        database = "mydb"
        port = 5434
        conn_string = f"postgresql://{user}:{password}@{host}:{port}/{database}"
        self.engine = create_engine(conn_string)  # Membuat engine untuk koneksi ke PostgreSQL
```

Input Program & Deskripsi

```python
def to_postgres(self, db_name: str, data: pd.DataFrame) -> None:
    """Task 6: Mengimpor DataFrame ke dalam database PostgreSQL by Wartadi DE4"""
    self.__create_connection()  # Membuat koneksi ke database
    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
    }
    try:
        # Mengimpor DataFrame ke PostgreSQL dengan skema tipe data yang telah ditentukan
        data.to_sql(name=db_name, con=self.engine, if_exists="replace", index=False, schema="public", dtype=df_schema, method=None, chun
```

Input Program & Deskripsi

```python
        }
        try:
            # Mengimpor DataFrame ke PostgreSQL dengan skema tipe data yang telah ditentukan
            data.to_sql(name=db_name, con=self.engine, if_exists="replace", index=False, schema="public", dtype=df_schema, method=None, chunksi

            # Task 6: Menghitung dan menampilkan jumlah baris yang di-ingest
            with self.engine.connect() as connection:
                result = connection.execute(text(f"SELECT COUNT(*) FROM public.{db_name}"))
                row_count = result.scalar()  # Menghitung jumlah baris yang berhasil di-ingest
                print(f"\n=== Task 6: Jumlah baris yang di-ingest by Wartadi DE4 ===")
                print(f"Jumlah baris yang di-ingest: {row_count}")
        except SQLAlchemyError as err:
            print(f"error >> {err}")  # Menangani error saat mengimpor data ke PostgreSQL

def main():
    extract = Extraction()
    file_path = "/Users/wartadi/Desktop/alta/ingestion-data-main/dataset/yellow_tripdata_2023-01.parquet"
    df_result = extract.local_file(file_path)  # Melakukan ekstraksi data dari file Parquet

    load = Load()
    db_name = "data_parquet_by_wartadi"
    load.to_postgres(db_name, df_result)  # Mengimpor data yang sudah diproses ke PostgreSQL

if __name__ == "__main__":
    main()
```

Input Program & Deskripsi

Kode yang Anda berikan sudah sesuai dengan instruksi untuk melakukan proses ekstraksi, transformasi, dan load (ETL) pada dataset "Yellow Trip". Berikut adalah rangkuman dari setiap langkah yang sesuai dengan deskripsi Anda:

1. **Membaca File Parquet ke DataFrame**:
   File Parquet dibaca menggunakan pustaka fastparquet, dan hasilnya disimpan dalam DataFrame.
2. **Membersihkan Dataset "Yellow Trip"**:
   Dataset dibersihkan dari nilai NaN, duplikat, dan data dengan jarak perjalanan negatif.
3. **Menentukan Skema Tipe Data saat Menggunakan Metode to_sql**:
   Skema tipe data didefinisikan menggunakan tipe data SQLAlchemy agar sesuai dengan struktur tabel di PostgreSQL.
4. **Mengimpor Dataset "Yellow Trip" ke PostgreSQL**:
   Data diimpor ke dalam database PostgreSQL menggunakan metode to_sql dengan skema tipe data yang telah ditentukan.
5. **Menghitung Berapa Banyak Baris yang Diingest**:
   Setelah data diimpor, jumlah baris yang berhasil diingest dihitung dan ditampilkan.
   Kodenya sudah cukup komprehensif dan telah mengikuti alur ETL yang diharapkan.

TASK 1

Output

Untuk output nomor 1 harusnya tidak ada tapi karena saya ingin menamplkan data yang sudah di load

```
(.venv) wartadi@Wartadis-MacBook-Pro PART_2_INGESTION_src % python main.py

=== Task 1: DataFrame dari file Parquet by Wartadi DE4 ===
```

Row Index	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	...
0	2	2023-01-01 00:32:10	2023-01-01 00:40:36	1.0	0.97	Other columns (14 more)
1	2	2023-01-01 00:55:08	2023-01-01 01:01:27	1.0	1.1	Other columns (14 more)
2	2	2023-01-01 00:25:04	2023-01-01 00:37:49	1.0	2.51	Other columns (14 more)
3	1	2023-01-01 00:03:48	2023-01-01 00:13:25	0.0	1.9	Other columns (14 more)
4	2	2023-01-01 00:10:29	2023-01-01 00:21:19	1.0	1.43	Other columns (14 more)
5	2	2023-01-01 00:50:34	2023-01-01 01:02:52	1.0	1.84	Other columns (14 more)
6	2	2023-01-01 00:09:22	2023-01-01 00:19:49	1.0	1.66	Other columns (14 more)
7	2	2023-01-01 00:27:12	2023-01-01 00:49:56	1.0	11.7	Other columns (14 more)
8	2	2023-01-01 00:21:44	2023-01-01 00:36:40	1.0	2.95	Other columns (14 more)
9	2	2023-01-01 00:39:42	2023-01-01 00:50:36	1.0	3.01	Other columns (14 more)

TASK 2

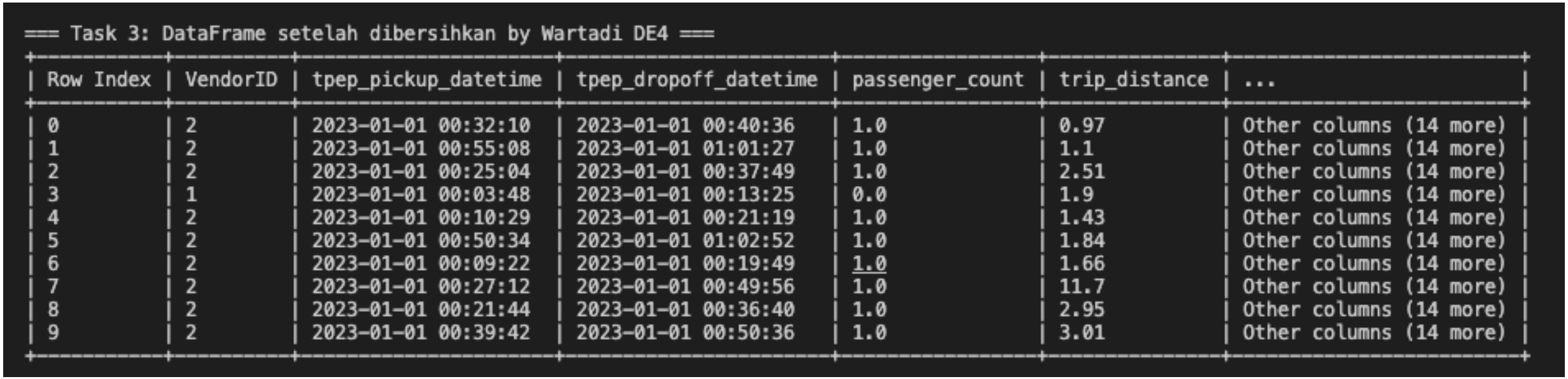
Output

Hasilnya menunjukkan sama dengan nomor 1 karena ini adalah hasil load data parquet yang masih belum bersihkan

```
=== Task 2: DataFrame dari file Parquet sebelum dibersihkan dan di-cast by Wartadi DE4 ===
+-----------+----------+-----------------------+------------------------+-----------------+---------------+-------------------------+
| Row Index | VendorID | tpep_pickup_datetime  | tpep_dropoff_datetime  | passenger_count | trip_distance | ...                     |
+-----------+----------+-----------------------+------------------------+-----------------+---------------+-------------------------+
| 0         | 2        | 2023-01-01 00:32:10   | 2023-01-01 00:40:36    | 1.0             | 0.97          | Other columns (14 more) |
| 1         | 2        | 2023-01-01 00:55:08   | 2023-01-01 01:01:27    | 1.0             | 1.1           | Other columns (14 more) |
| 2         | 2        | 2023-01-01 00:25:04   | 2023-01-01 00:37:49    | 1.0             | 2.51          | Other columns (14 more) |
| 3         | 1        | 2023-01-01 00:03:48   | 2023-01-01 00:13:25    | 0.0             | 1.9           | Other columns (14 more) |
| 4         | 2        | 2023-01-01 00:10:29   | 2023-01-01 00:21:19    | 1.0             | 1.43          | Other columns (14 more) |
| 5         | 2        | 2023-01-01 00:50:34   | 2023-01-01 01:02:52    | 1.0             | 1.84          | Other columns (14 more) |
| 6         | 2        | 2023-01-01 00:09:22   | 2023-01-01 00:19:49    | 1.0             | 1.66          | Other columns (14 more) |
| 7         | 2        | 2023-01-01 00:27:12   | 2023-01-01 00:49:56    | 1.0             | 11.7          | Other columns (14 more) |
| 8         | 2        | 2023-01-01 00:21:44   | 2023-01-01 00:36:40    | 1.0             | 2.95          | Other columns (14 more) |
| 9         | 2        | 2023-01-01 00:39:42   | 2023-01-01 00:50:36    | 1.0             | 3.01          | Other columns (14 more) |
+-----------+----------+-----------------------+------------------------+-----------------+---------------+-------------------------+
```

TASK 3

Ouput Program



```
=== Task 3: DataFrame setelah dibersihkan by Wartadi DE4 ===
+-----------+----------+-----------------------+------------------------+-----------------+---------------+--------------------------+
| Row Index | VendorID | tpep_pickup_datetime  | tpep_dropoff_datetime  | passenger_count | trip_distance | ...                      |
+-----------+----------+-----------------------+------------------------+-----------------+---------------+--------------------------+
| 0         | 2        | 2023-01-01 00:32:10   | 2023-01-01 00:40:36    | 1.0             | 0.97          | Other columns (14 more)  |
| 1         | 2        | 2023-01-01 00:55:08   | 2023-01-01 01:01:27    | 1.0             | 1.1           | Other columns (14 more)  |
| 2         | 2        | 2023-01-01 00:25:04   | 2023-01-01 00:37:49    | 1.0             | 2.51          | Other columns (14 more)  |
| 3         | 1        | 2023-01-01 00:03:48   | 2023-01-01 00:13:25    | 0.0             | 1.9           | Other columns (14 more)  |
| 4         | 2        | 2023-01-01 00:10:29   | 2023-01-01 00:21:19    | 1.0             | 1.43          | Other columns (14 more)  |
| 5         | 2        | 2023-01-01 00:50:34   | 2023-01-01 01:02:52    | 1.0             | 1.84          | Other columns (14 more)  |
| 6         | 2        | 2023-01-01 00:09:22   | 2023-01-01 00:19:49    | 1.0             | 1.66          | Other columns (14 more)  |
| 7         | 2        | 2023-01-01 00:27:12   | 2023-01-01 00:49:56    | 1.0             | 11.7          | Other columns (14 more)  |
| 8         | 2        | 2023-01-01 00:21:44   | 2023-01-01 00:36:40    | 1.0             | 2.95          | Other columns (14 more)  |
| 9         | 2        | 2023-01-01 00:39:42   | 2023-01-01 00:50:36    | 1.0             | 3.01          | Other columns (14 more)  |
+-----------+----------+-----------------------+------------------------+-----------------+---------------+--------------------------+
```

Task 4

Output Program

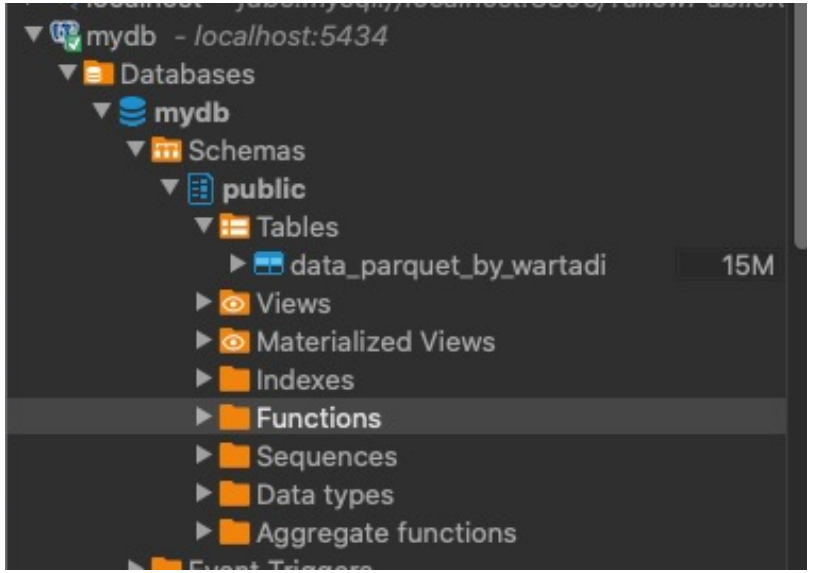
```
=== Task 4: Informasi Schema DataFrame setelah di-cast by Wartadi DE4 ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 19 columns):
 #   Column                 Non-Null Count   Dtype
---  ------                 --------------   -----
 0   VendorID               100000 non-null  int64
 1   tpep_pickup_datetime   100000 non-null  datetime64[ns]
 2   tpep_dropoff_datetime  100000 non-null  datetime64[ns]
 3   passenger_count        100000 non-null  Int8
 4   trip_distance          100000 non-null  float64
 5   RatecodeID             100000 non-null  float64
 6   store_and_fwd_flag     100000 non-null  boolean
 7   PULocationID           100000 non-null  int64
 8   DOLocationID           100000 non-null  int64
 9   payment_type           100000 non-null  int64
 10  fare_amount            100000 non-null  float64
 11  extra                  100000 non-null  float64
 12  mta_tax                100000 non-null  float64
 13  tip_amount             100000 non-null  float64
 14  tolls_amount           100000 non-null  float64
 15  improvement_surcharge  100000 non-null  float64
 16  total_amount           100000 non-null  float64
 17  congestion_surcharge   100000 non-null  float64
 18  airport_fee            100000 non-null  float64
dtypes: Int8(1), boolean(1), datetime64[ns](2), float64(11), int64(4)
memory usage: 13.4 MB
None
```

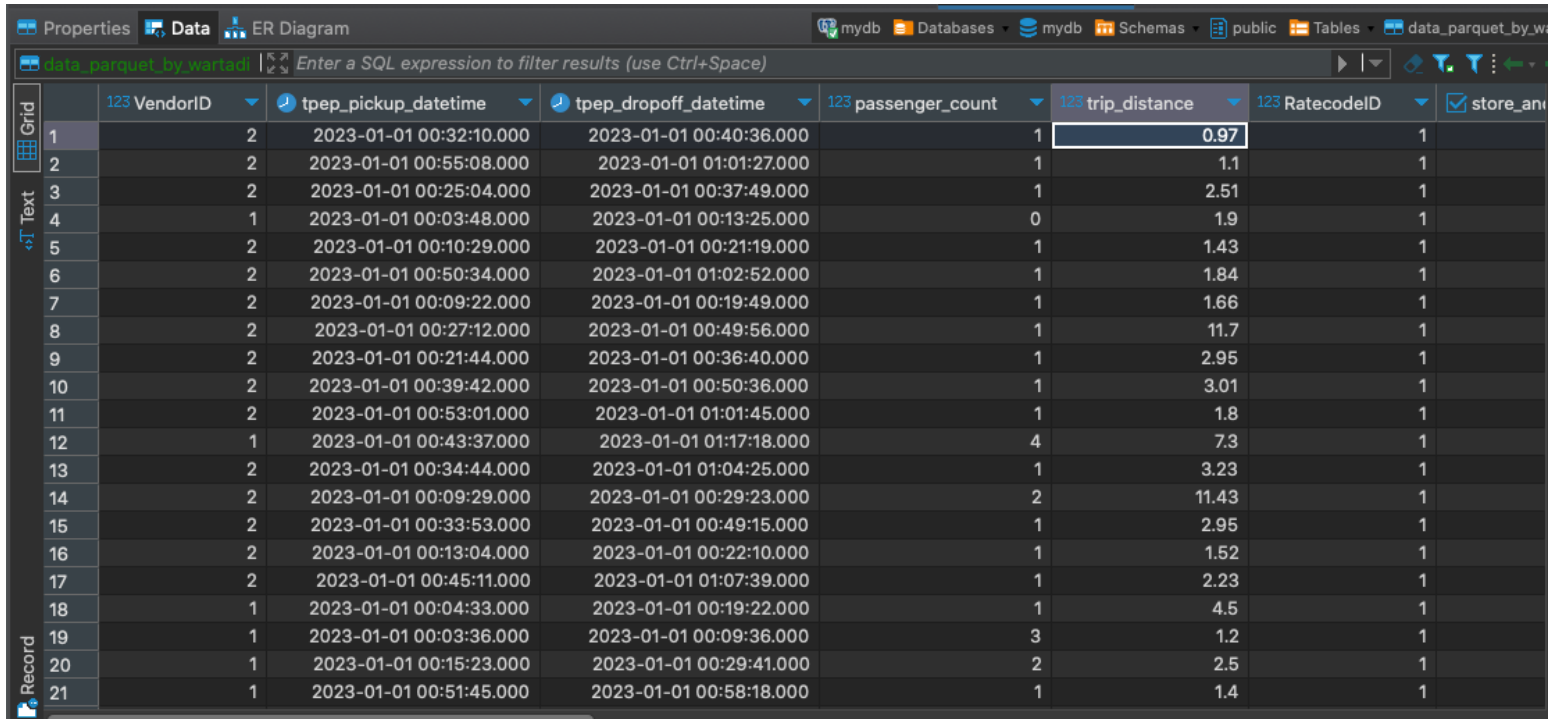
Task 5

Output Program

Sebelumnya kita setting dulu untuk collection database PosgreSQL sesuai dengan Docker Dompose yang sudah dibuat. Dimana dalam hal ini Database dibuat dengan nama **mydb** dengan host :**localhost** dan port **5434**

```
=== Task 5: DataFrame setelah di-cast by Wartadi DE4 ===
ada di postgreSQL, coba check deh hehehe
```

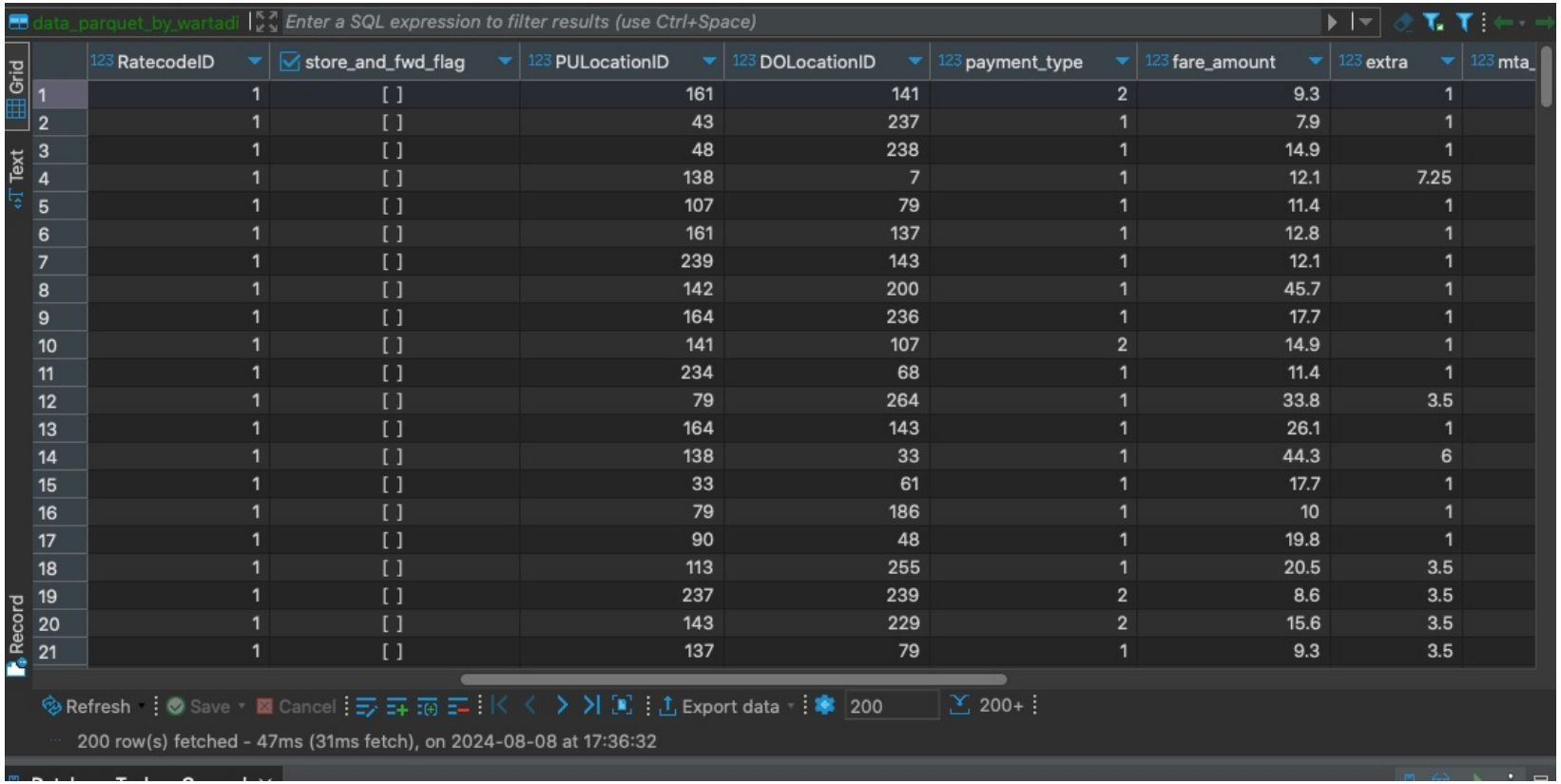




	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_an
1	2	2023-01-01 00:32:10.000	2023-01-01 00:40:36.000	1	0.97	1	
2	2	2023-01-01 00:55:08.000	2023-01-01 01:01:27.000	1	1.1	1	
3	2	2023-01-01 00:25:04.000	2023-01-01 00:37:49.000	1	2.51	1	
4	1	2023-01-01 00:03:48.000	2023-01-01 00:13:25.000	0	1.9	1	
5	2	2023-01-01 00:10:29.000	2023-01-01 00:21:19.000	1	1.43	1	
6	2	2023-01-01 00:50:34.000	2023-01-01 01:02:52.000	1	1.84	1	
7	2	2023-01-01 00:09:22.000	2023-01-01 00:19:49.000	1	1.66	1	
8	2	2023-01-01 00:27:12.000	2023-01-01 00:49:56.000	1	11.7	1	
9	2	2023-01-01 00:21:44.000	2023-01-01 00:36:40.000	1	2.95	1	
10	2	2023-01-01 00:39:42.000	2023-01-01 00:50:36.000	1	3.01	1	
11	2	2023-01-01 00:53:01.000	2023-01-01 01:01:45.000	1	1.8	1	
12	1	2023-01-01 00:43:37.000	2023-01-01 01:17:18.000	4	7.3	1	
13	2	2023-01-01 00:34:44.000	2023-01-01 01:04:25.000	1	3.23	1	
14	2	2023-01-01 00:09:29.000	2023-01-01 00:29:23.000	2	11.43	1	
15	2	2023-01-01 00:33:53.000	2023-01-01 00:49:15.000	1	2.95	1	
16	2	2023-01-01 00:13:04.000	2023-01-01 00:22:10.000	1	1.52	1	
17	2	2023-01-01 00:45:11.000	2023-01-01 01:07:39.000	1	2.23	1	
18	1	2023-01-01 00:04:33.000	2023-01-01 00:19:22.000	1	4.5	1	
19	1	2023-01-01 00:03:36.000	2023-01-01 00:09:36.000	3	1.2	1	
20	1	2023-01-01 00:15:23.000	2023-01-01 00:29:41.000	2	2.5	1	
21	1	2023-01-01 00:51:45.000	2023-01-01 00:58:18.000	1	1.4	1	

Task 5

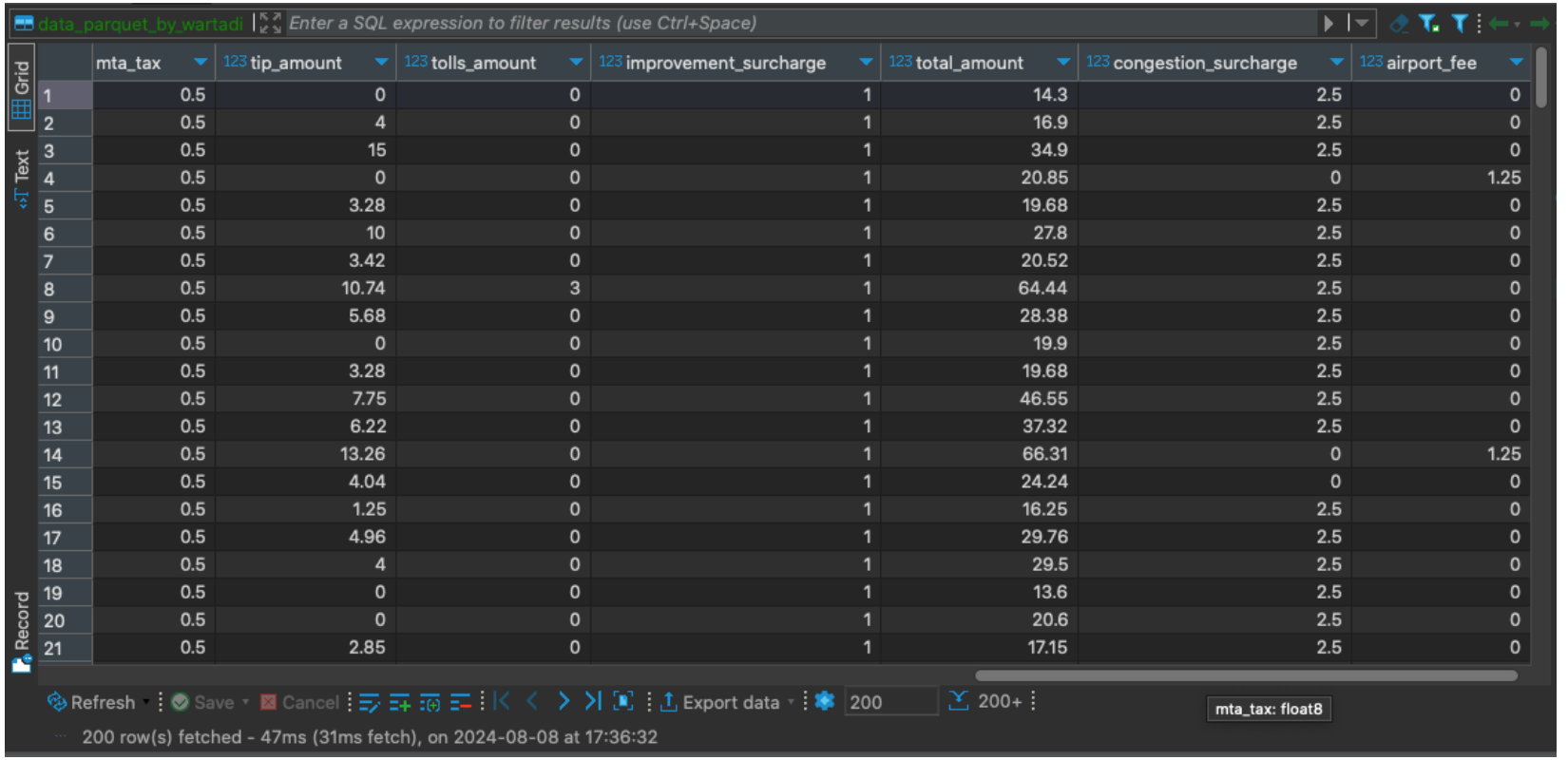
Output Program



	RatecodeID	store_and_fwd_flag	PULocationID	DOLocationID	payment_type	fare_amount	extra	mta_
1	1	[ ]	161	141	2	9.3	1	
2	1	[ ]	43	237	1	7.9	1	
3	1	[ ]	48	238	1	14.9	1	
4	1	[ ]	138	7	1	12.1	7.25	
5	1	[ ]	107	79	1	11.4	1	
6	1	[ ]	161	137	1	12.8	1	
7	1	[ ]	239	143	1	12.1	1	
8	1	[ ]	142	200	1	45.7	1	
9	1	[ ]	164	236	1	17.7	1	
10	1	[ ]	141	107	2	14.9	1	
11	1	[ ]	234	68	1	11.4	1	
12	1	[ ]	79	264	1	33.8	3.5	
13	1	[ ]	164	143	1	26.1	1	
14	1	[ ]	138	33	1	44.3	6	
15	1	[ ]	33	61	1	17.7	1	
16	1	[ ]	79	186	1	10	1	
17	1	[ ]	90	48	1	19.8	1	
18	1	[ ]	113	255	1	20.5	3.5	
19	1	[ ]	237	239	2	8.6	3.5	
20	1	[ ]	143	229	2	15.6	3.5	
21	1	[ ]	137	79	1	9.3	3.5	

Task 5

Output Program



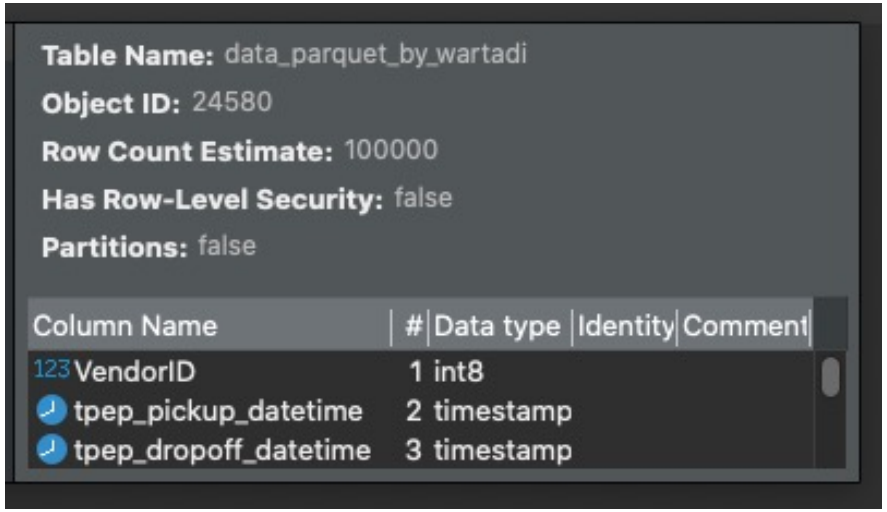
	mta_tax	tip_amount	tolls_amount	improvement_surcharge	total_amount	congestion_surcharge	airport_fee
1	0.5	0	0	1	14.3	2.5	0
2	0.5	4	0	1	16.9	2.5	0
3	0.5	15	0	1	34.9	2.5	0
4	0.5	0	0	1	20.85	0	1.25
5	0.5	3.28	0	1	19.68	2.5	0
6	0.5	10	0	1	27.8	2.5	0
7	0.5	3.42	0	1	20.52	2.5	0
8	0.5	10.74	3	1	64.44	2.5	0
9	0.5	5.68	0	1	28.38	2.5	0
10	0.5	0	0	1	19.9	2.5	0
11	0.5	3.28	0	1	19.68	2.5	0
12	0.5	7.75	0	1	46.55	2.5	0
13	0.5	6.22	0	1	37.32	2.5	0
14	0.5	13.26	0	1	66.31	0	1.25
15	0.5	4.04	0	1	24.24	0	0
16	0.5	1.25	0	1	16.25	2.5	0
17	0.5	4.96	0	1	29.76	2.5	0
18	0.5	4	0	1	29.5	2.5	0
19	0.5	0	0	1	13.6	2.5	0
20	0.5	0	0	1	20.6	2.5	0
21	0.5	2.85	0	1	17.15	2.5	0

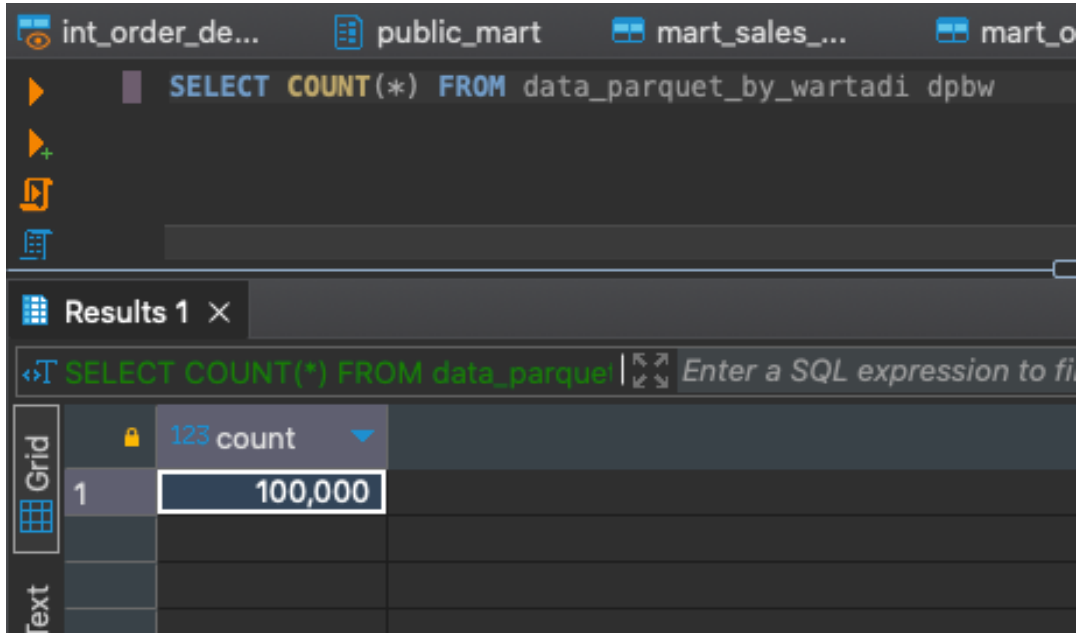
Task 6

Output Program

Bisa kita lihat bahwa hasil menunjukkan baik di terminal dan di porsgres sudah sesuai bahwa jumlah baris yang berhasidi ingest adalah 100000

```
=== Task 6: Jumlah baris yang di-ingest by Wartadi DE4 ===
Jumlah baris yang di-ingest: 100000
```





alterra

THANK YOU ☺