Data Integration and Analysis with DBT, Python, and SQL Server

Objective

The objective of this project is to load a Parquet file into a SQL Server database, clean and transform the data using DBT models, and analyze it using Power BI.

Pre-Requisites

1. **Environment Setup**
    - A SQL Server database installed and running locally.
    - Python installed on your system.
    - DBT installed for managing SQL transformations.
    - Visual Studio Code installed for editing and running scripts.
2. **Visual Studio Code Extensions**
    - Python
    - Code Runner

Steps to Recreate

1. Environment Setup

a. Create the Project Directory

1. Open the terminal in Visual Studio Code.

Navigate to your desired project path:
bash
Copy code (Use the path where you want to create the project)
```
cd Z:\Archives\dbt_Project
```

2.

Create a Python virtual environment:
bash

Copy code

```
python -m venv dbt_venv
```

3.

Activate the virtual environment:
bash
Copy code

```
.\dbt\Scripts\Activate.ps1
```

4.

Install the required DBT package for SQL Server:
bash
Copy code

```
pip install dbt-sqlserver
```

5.

Create the DBT profile folder:
bash
Copy code

```
mkdir $home\.dbt
```

6.

b. Configure DBT Profile

1. Navigate to `C:\Users\<YourUsername>\.dbt`.

Create or edit a `profiles.yml` file with the following configuration:
yaml
Copy code

```
dbt_proj:
  outputs:
    dev:
      type: sqlserver
      driver: "ODBC Driver 17 for SQL Server"
      server: "SQLEXPRESS01"
      port: 1433
      user: "dbt"
      password: "admin"
      database: "dbt_project"
      schema: "dbo"
      threads: 1
      encrypt: false
      trust_cert: true
  target: dev
```

2.

c. Test DBT Connection

Activate the virtual environment again:
bash
Copy code

```
Z:\Archives\dbt_Project\dbt_venv\Scripts\Activate.ps1
```

-

Run the following command to debug the DBT connection:
bash
Copy code

```
dbt debug
```

-

---

2. Load Parquet File into SQL Server

a. Install Required Python Libraries

bash
Copy code

```
pip install pandas pyodbc fastparquet
```

b. Create the SQL Table

Use the provided SQL script (`Create table sql`) to create the table `FHV_Trip_Records` in your SQL Server database.

c. Load Data Using Python

Use the following Python script (`Insert_data_into_SQL.py`) to load data into the SQL table:

python
Copy code

```
import pandas as pd
import pyodbc

# Connection details
server = 'Wa\MSSQLSERVER011'
database = 'dbt_project'
username = 'dbt'
password = 'admin'

# Parquet file path
parquet_file_path = 'Z:/Archives/dbt_Project/fhvhv_tripdata_2024-01
(1).parquet'

# Read the Parquet file
df = pd.read_parquet(parquet_file_path, engine='fastparquet')

# Filter out rows with invalid datetime values
datetime_columns = ['request_datetime', 'on_scene_datetime',
'pickup_datetime', 'dropoff_datetime']
for col in datetime_columns:
    df[col] = pd.to_datetime(df[col], errors='coerce')

# Connect to SQL Server
conn = pyodbc.connect(f"DRIVER={{ODBC Driver 17 for SQL
Server}};SERVER={server};DATABASE={database};UID={username};PWD={passw
ord}")
cursor = conn.cursor()

# Insert data into SQL Server
sql_insert = """
INSERT INTO FHV_Trip_Records (hvfhs_license_num, dispatching_base_num,
originating_base_num, request_datetime, on_scene_datetime,
pickup_datetime, dropoff_datetime, PULocationID, DOLocationID,
trip_miles, trip_time, base_passenger_fare, tolls, bcf, sales_tax,
congestion_surcharge, airport_fee, tips, driver_pay,
shared_request_flag, shared_match_flag, access_a_ride_flag,
wav_request_flag, wav_match_flag)
```

```
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?)
"""

batch_size = 1000
for i in range(0, len(df), batch_size):
    batch = df.iloc[i:i + batch_size].to_records(index=False)
    cursor.executemany(sql_insert, batch)
    conn.commit()

cursor.close()
conn.close()
print("Data successfully loaded into SQL Server.")
```

---

3. Create DBT Models

a. **`base_trip_data.sql`**

This model cleans the loaded data:

sql
Copy code
```
WITH cleaned_data AS (
SELECT
    newid() as FHV_Trip_ID,
    hvfhs_license_num,
    dispatching_base_num,
    originating_base_num,
    CONVERT(datetime, request_datetime) AS request_datetime,
    CONVERT(datetime, on_scene_datetime) AS on_scene_datetime,
    CONVERT(datetime, pickup_datetime) AS pickup_datetime,
    CONVERT(datetime, dropoff_datetime) AS dropoff_datetime,
    PULocationID,
    DOLocationID,
    trip_miles,
    trip_time,
    base_passenger_fare,
    tolls,
    bcf,
    sales_tax,
    congestion_surcharge,
    airport_fee,
    tips,
    driver_pay
FROM [dbo].[FHV_Trip_Records]
WHERE
    trip_miles IS NOT NULL AND trip_time IS NOT NULL
    AND trip_miles > 0 AND trip_time > 0
)

SELECT * FROM cleaned_data
```

b. `avg_duration_distance.sql`

This model calculates average trip durations and distances by time of day:

sql
Copy code

```
WITH cte AS (
SELECT
    CASE
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 0 AND 5 THEN
'Late Night'
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 6 AND 11 THEN
'Morning'
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 12 AND 17 THEN
'Afternoon'
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 18 AND 23 THEN
'Evening'
    END AS time_of_day,
    AVG(trip_time / 60.0) AS avg_duration_minutes,
    AVG(trip_miles) AS avg_distance_miles
FROM base_trip_data
GROUP BY
    CASE
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 0 AND 5 THEN
'Late Night'
```

```
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 6 AND 11 THEN
'Morning'
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 12 AND 17 THEN
'Afternoon'
        WHEN DATEPART(HOUR, pickup_datetime) BETWEEN 18 AND 23 THEN
'Evening'
    END
)

SELECT * FROM cte
```

---

4. Load Data into Power BI

- Open Power BI.
- Connect to the SQL Server database.
- Load the tables and create visuals for:
    1. Daily trip trends.
    2. Average duration and distance by time of day.
    3. Top 5 pickup locations.