**1. Introduction**

This project aims to build a data warehouse for analyzing taxi service data. The primary goal is to understand taxi service usage patterns, including high-demand zones, peak times throughout the year and other factors affecting the total revenue like tip amount. This analysis will enable taxi drivers to optimize their services, improve customer satisfaction, and increase profitability. The document gives an idea of the dataset we worked with and a guideline to set up the project in your local environment.

**2. Data Describe your data, make sense of the data**

| **Field Name** | **Description** |
| --- | --- |
| VendorID | A code indicating the TPEP provider that provided the record. 1= Creative Mobile Technologies, LLC; 2= VeriFone Inc. |
| vendor\_name | The categorical name of the vendor |
| Pickup\_datetime | The date and time when the meter was engaged. |
| Dropoff\_datetime | The date and time when the meter was disengaged. |
| passenger\_count | The number of passengers in the vehicle2. This is a driver-entered value. |
| trip\_distance | The elapsed trip distance in miles reported by the taximeter. |
| PULocationID | TLC Taxi Zone in which the taximeter was engaged |
| DOLocationID | TLC Taxi Zone in which the taximeter was disengaged |
| RateCodeID | The final rate code in effect at the end of the trip. 1= Standard rate 2=JFK 3=Newark 4=Nassau or Westchester 5=Negotiated fare 6=Group ride |
| rate\_code\_name | Categorical name of the RateCodeID |
| store\_and\_fwd\_flag | This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server5. Y= store and forward trip N= not a store and forward trip |
| payment\_type | A numeric code signifying how the passenger paid for the trip. 1= Credit card 2= Cash 3= No charge 4= Dispute 5= Unknown 6= Voided trip |
| payment\_name | Categorical name of the payment type |
| fare\_amount | The time-and-distance fare calculated by the meter. |
| extra | Miscellaneous extras and surcharges7. Currently, this only includes the $0.50 and $1 rush hour and overnight charges. |
| mta\_tax | $0.50 MTA tax that is automatically triggered based on the metered rate in use. |
| improvement\_surcharge | $0.30 improvement surcharge assessed trips at the flag drop10. The improvement surcharge began being levied in 2015. |
| tip\_amount | Tip amount – This field is automatically populated for credit card tips12. Cash tips are not included. |
| tolls\_amount | Total amount of all tolls paid in trip. |
| total\_amount | The total amount charged to passengers14. Does not include cash tips. |
| congestion\_Surcharge | Total amount collected in trip for NYS congestion surcharge. |
| airport\_fee | $1.25 for pick up only at LaGuardia and John F. Kennedy Airports |
| trip\_time | total trip duration |
| trip\_time\_minutes | total duration of the trip in minutes |
| avg\_mph | average speed of the trip |
| fare\_per\_minute | per minute charges for the trip |
| tip\_per\_minute | per minute tip for the trip |
| tip\_percentage | fraction corresponding to the tip amount with respect to the total fare |
| dropoff\_zone | zone corresponding to the dropoff location |
| dropoff\_borough | borough corresponding to the dropoff location |
| pickup\_zone | zone corresponding to the pickup location |
| pickup\_borough | borough corresponding to the pickup location |
| dropoff\_time | dropoff time of the trip in hh:mm:ss military format |
| dropoff\_hour | dropoff hour corresponding to dropofftime, 24hr format; military time |
| dropoff\_month | corresponsing month of the dropoff time 1=January 2=Februray 3=March 4=April 5=May 6=June 7=July 8=August 9=September 10=October 11=November 12=December |
| dropoff\_year | corresponding year of the dropoff time in yyyy format |
| pickup\_time | pickup time of the trip in hh:mm:ss military format |
| pickup\_hour | pickup hour corresponding to pickup time, 24hr format; military time |
| pickup\_month | corresponsing month of the pickup time 1=January 2=Februray 3=March 4=April 5=May 6=June 7=July 8=August 9=September 10=October 11=November 12=December |
| pickup\_year | corresponding year of the pickup time in yyyy format |
| day\_of\_week | 1= Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday, 7 = Sunday |
| is\_holiday | Boolean, if the day fell on a federal holiday per the USFederalHolidayCalendar Pandas library |

**3. ETL: Yellow\_Taxi\_ETL.ipynb**

**3.1 ETL: Extract and Transform**

**Cleaning:** Data cleaning was performed to eliminate records with negative values for trip distance, trip time, and fare amount, along with those not matching the data dictionary definitions for payment type, vendor id, and rate code. Records outside the years 2020-2023, containing NaN values, or having a '0' trip distance were also removed.





**Transformation:** Feature engineering added new measures: trip time in minutes, fare per minute, tip per minute, tip percentage, and average mph, calculated using the cleaned data. Categorical dimensions were enriched by merging with the taxi\_zones.csv file downloaded from the below link:

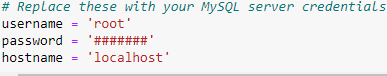
<https://data.cityofnewyork.us/Transportation/NYC-Taxi-Zones/d3c5-ddgc>

for location attributes, and using pandas for day of week and holiday flags. Additionally, columns with "name" provide descriptive values for corresponding numeric codes in the dataset, utilizing Python packages and custom calculations. The process resulted in a dataset with a total of 42 features, optimized for analysis.



**3.2 ETL: how to load data to DW by year (2020, 2021, 2022, ...)**

To load data in the data warehouse, the user will need to specify the below variables :



The python script will use SQLAlchemy to connect to the database ‘bi\_and\_dw’. If the database does not exist, it will create the database and then establish a connection to the server using the credentials specified.



Once the connection has been established, we use the below code to load data into the ‘source\_table’ in batches of 1000.



**4. Star schema**

**4.1 How to build your star schema, logic and rationale**

The star schema we came up with is designed to optimize data storage and query performance for a taxi trip dataset, facilitating efficient analysis and business intelligence (BI) queries. The rationale behind the specific dimensions and how they contribute to benefiting taxi drivers through observations and analysis are as follows:

**Fact Table**

The Fact Table stores quantitative data (measures) about each trip, such as passenger\_count, trip\_distance, fare\_amount, and derived metrics like fare\_per\_minute and avg\_mph. It serves as the central table in the star schema, linked to various dimensions through foreign keys. This structure supports fast aggregation and complex calculations across multiple dimensions, enabling detailed analysis of trip profitability, efficiency, and customer behavior.

**Dimension Tables**

**a. Vendor Dimension:**

Purpose: Contains details about the taxi vendors. It allows analysis by vendor to assess vendor-specific performance, compliance, and service quality.

Benefits: By analyzing vendor-specific data, taxi drivers can understand the competitive landscape, align themselves with higher-performing or more popular vendors, and leverage vendor-specific advantages.

**.b. Datetime Dimension:**

Purpose: Provides detailed time-related attributes for each trip, including pickup and dropoff times down to the hour, and categorizations like day\_of\_week and is\_holiday.

Benefits: Enables time-based analysis to identify peak demand periods, assess the impact of holidays on fares, and optimize shift times for drivers. This helps drivers plan their schedules to maximize earnings and efficiency.

**c. Location Dimension:**

Purpose: Captures geographical details about trip pickup and dropoff points, including zones and boroughs. This dimension is vital for spatial analysis.

Benefits: Helps identify high-demand areas, understand geographic patterns in trip distributions, and optimize routes. Drivers can use this information to position themselves in areas with higher trip requests or higher fare rates, improving their earnings.

**d. Rate Dimension:**

Purpose: Contains information about the rate codes applied to trips, indicating special pricing conditions (e.g., airport trips, peak hours).

Benefits: Enables analysis of trip earnings in relation to different rate codes. Drivers can identify the most lucrative types of trips and adjust their availability or target specific trip types accordingly.

**e. Payment Type Dimension:**

Purpose: Details the methods of payment used for trips, allowing analysis of payment preferences and the impact on tips.

Benefits: Understanding payment trends can help drivers anticipate customer preferences and prepare for transactions accordingly. Analysis of tip amounts by payment type can also inform strategies to encourage tipping.

**4.2 Star schema description: FACT table(s) and DIM tables**

**fact\_table**

| **Column name** | **Data type** |
| --- | --- |
| airport\_fee | double |
| avg\_mph | double |
| congestion\_surcharge | double |
| date\_id | int |
| extra | double |
| fare\_amount | double |
| fare\_per\_minute | double |
| improvement\_surcharge | double |
| location\_id | int |
| mta\_tax | double |
| passenger\_count | int |
| payment\_id | int |
| rate\_id | int |
| store\_and\_fwd\_flag | varchar |
| tip\_amount | double |
| tip\_per\_minute | double |
| tolls\_amount | double |
| total\_amount | double |
| trip\_distance | double |
| trip\_id | int |
| trip\_time\_minutes | int |
| vendor\_id | int |

**Dimension Tables**

**a. datetime\_dimension**

| **Column name** | **Data type** |
| --- | --- |
| date\_id | int |
| day\_of\_week | varchar |
| dropoff\_datetime | datetime |
| dropoff\_hour | int |
| dropoff\_month | int |
| dropoff\_time | time |
| dropoff\_year | year |
| is\_holiday | tinyint |
| pickup\_datetime | datetime |
| pickup\_hour | int |
| pickup\_month | int |
| pickup\_time | time |
| pickup\_year | year |

**b. location\_dimension**

| **Column name** | **Data type** |
| --- | --- |
| do\_location\_id | int |
| dropoff\_borough | varchar |
| dropoff\_zone | varchar |
| location\_id | int |
| pickup\_borough | varchar |
| pickup\_zone | varchar |
| pu\_location\_id | int |

**c. paymenttype\_dimension**

| **Column name** | **Data type** |
| --- | --- |
| payment\_id | int |
| payment\_type | bigint |
| payment\_type\_name | varchar |

**d. vendor\_dimension**

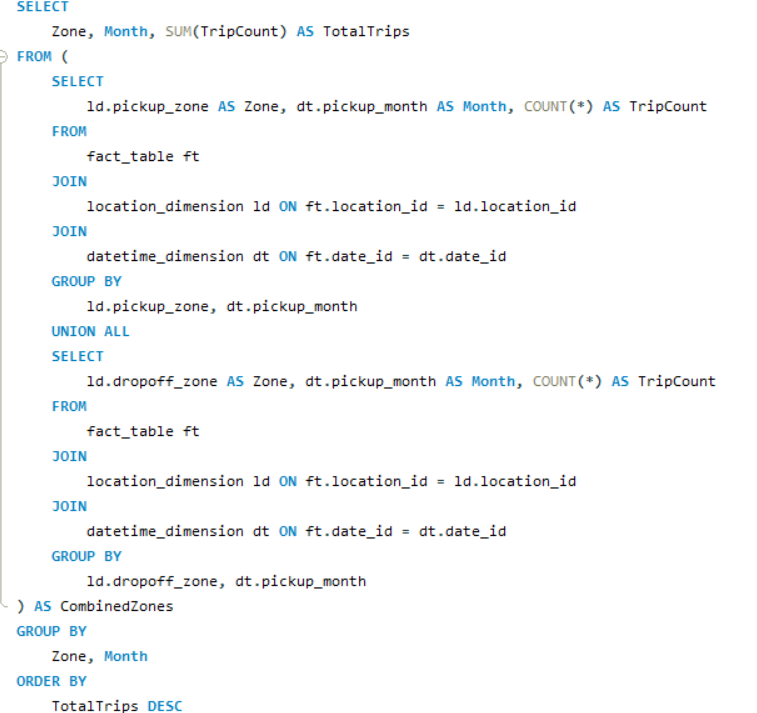
| **Column name** | **Data type** |
| --- | --- |
| vendor\_id | int |
| vendorID | bigint |
| vendor\_name | varchar |

**e. rate\_dimension**

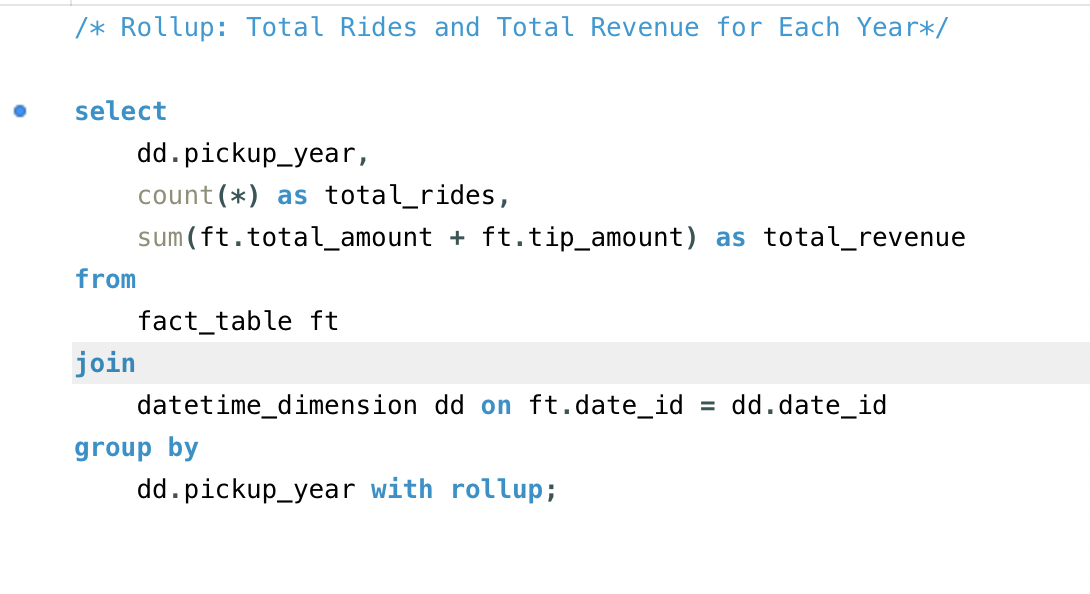
| **Column name** | **Data type** |
| --- | --- |
| rate\_id | int |
| RatecodeID | double |
| rate\_code\_name | varchar |

**5. BI queries and presentations: list of queries in English and SQL**

**a. Zones of highest demands and month of the year:** This SQL query is designed to aggregate and analyze taxi trip data, specifically focusing on the volume of trips by pickup and dropoff zones across different months.

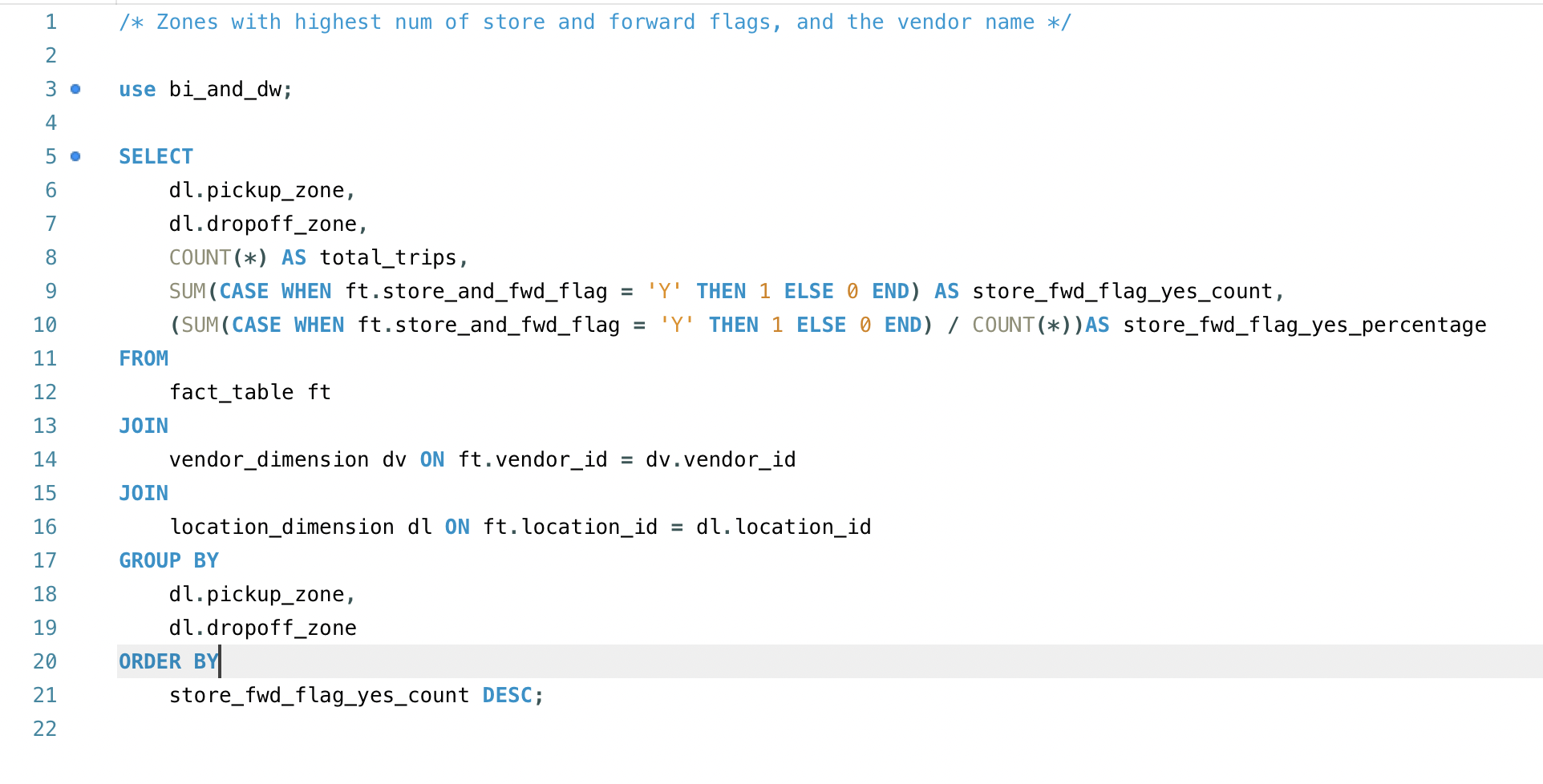


**b. Rollup Operation- Total Rides and Total Revenue by Year:** Provide summary output of total rides and revenue generated for the four years of analysis

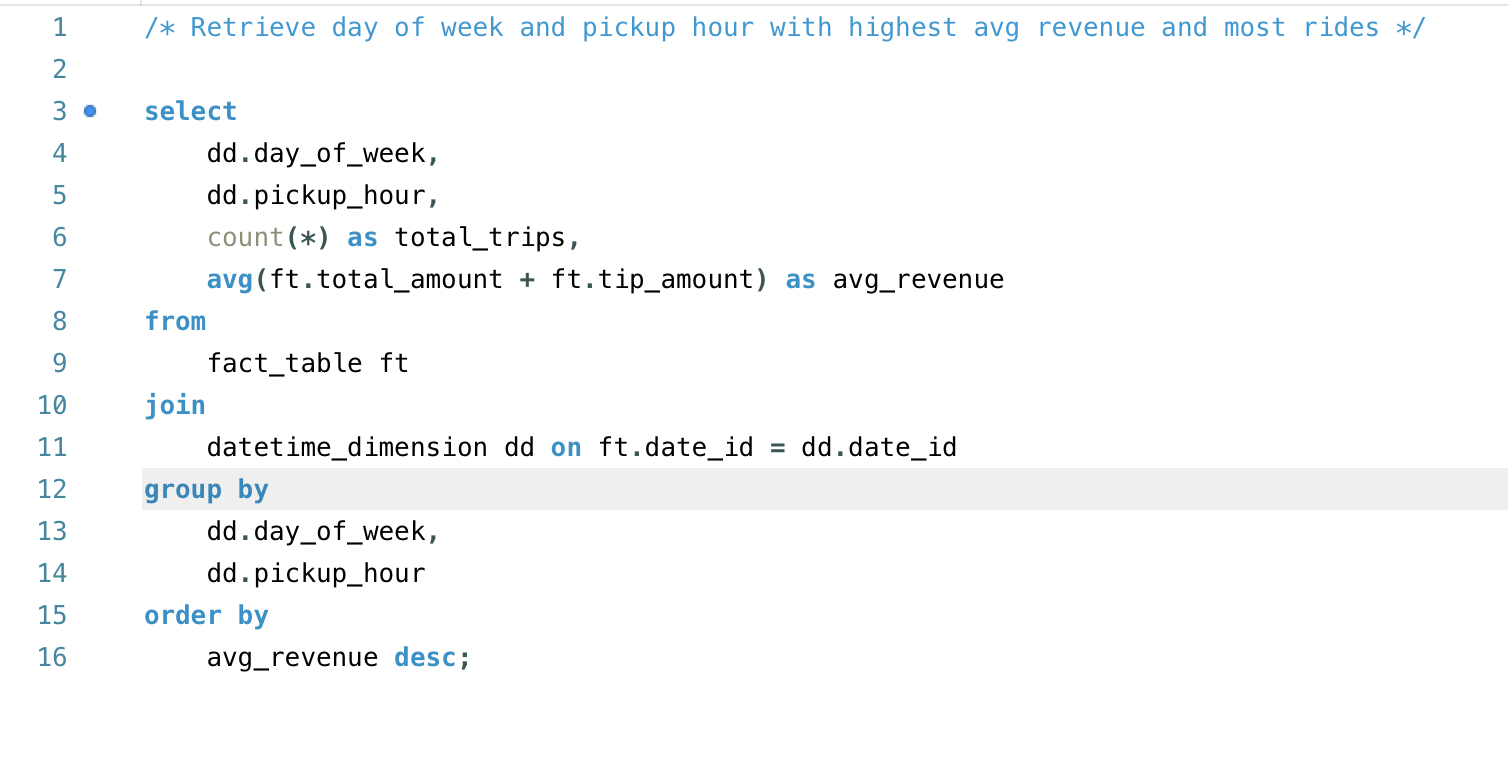


| **pickup\_year** | **total\_rides** | **total\_revenue** |
| --- | --- | --- |
| **2020** | 2344433 | 47249857.55970810 |
| **2021** | 2890725 | 62097075.20932570 |
| **2022** | 3745050 | 90563045.65791920 |
| **2023** | 3290168 | 106660227.7698370 |
| **NULL** | 12270376 | 306570206.20409800 |

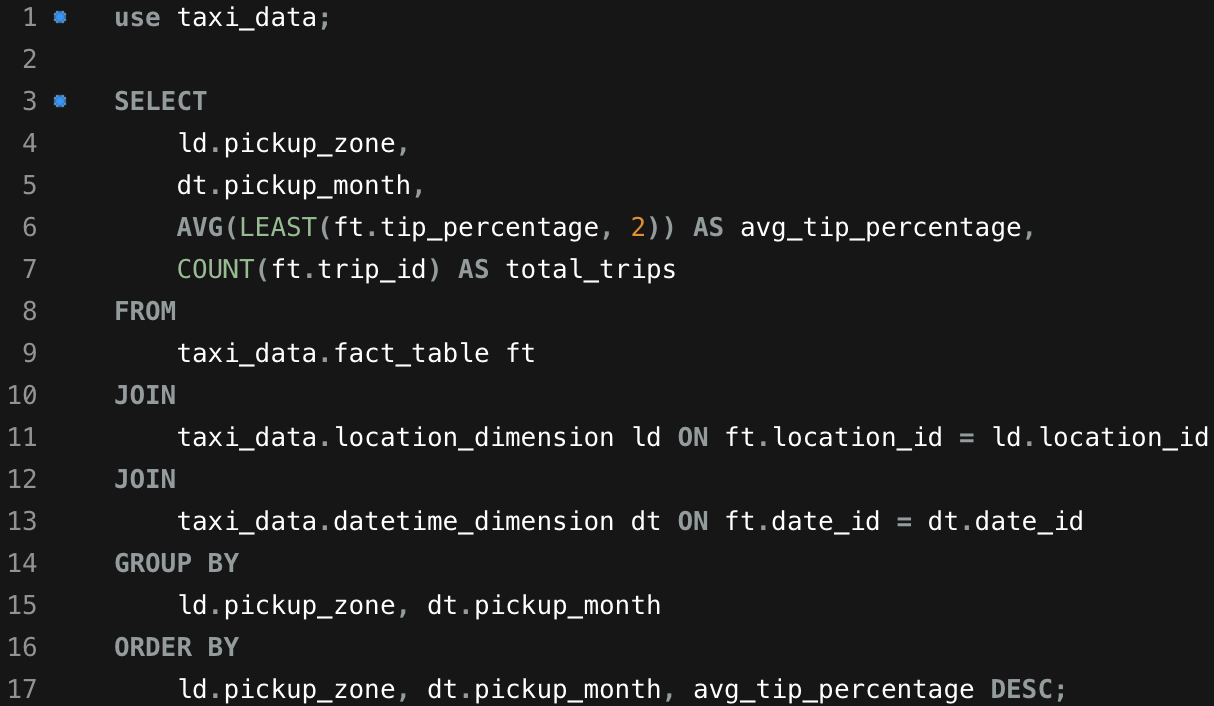
C. **Store and Forward Analysis:** Identify both the pickup and dropoff zones that showed the highest average percentage of cases where the ride had a store and forward flag



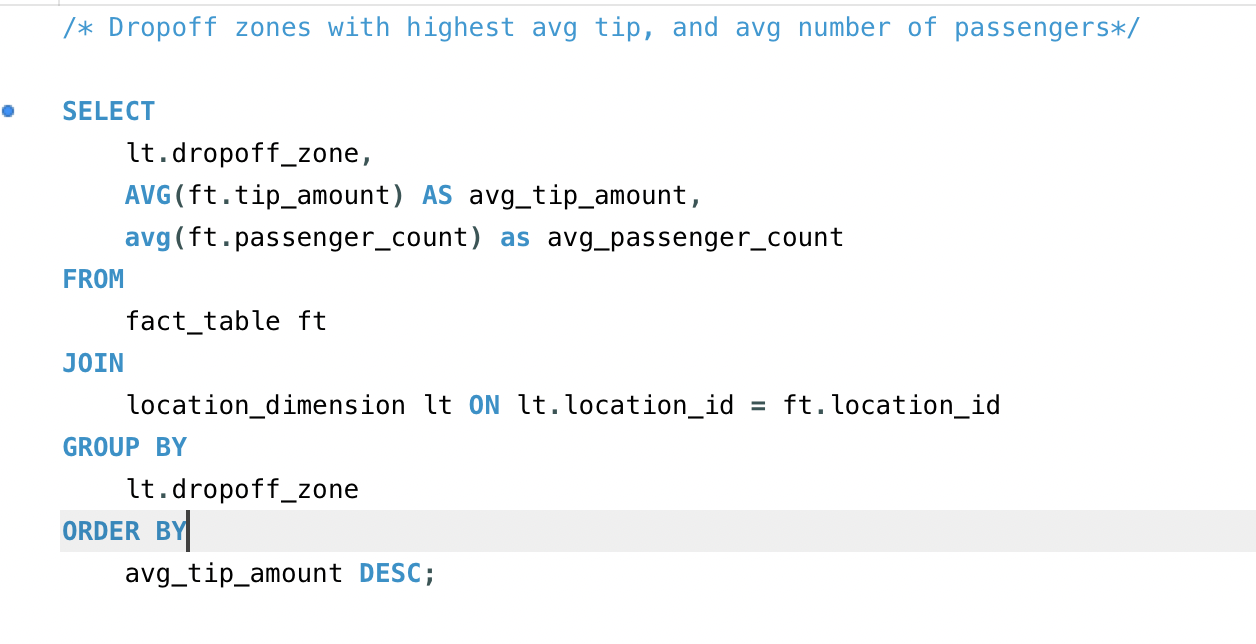
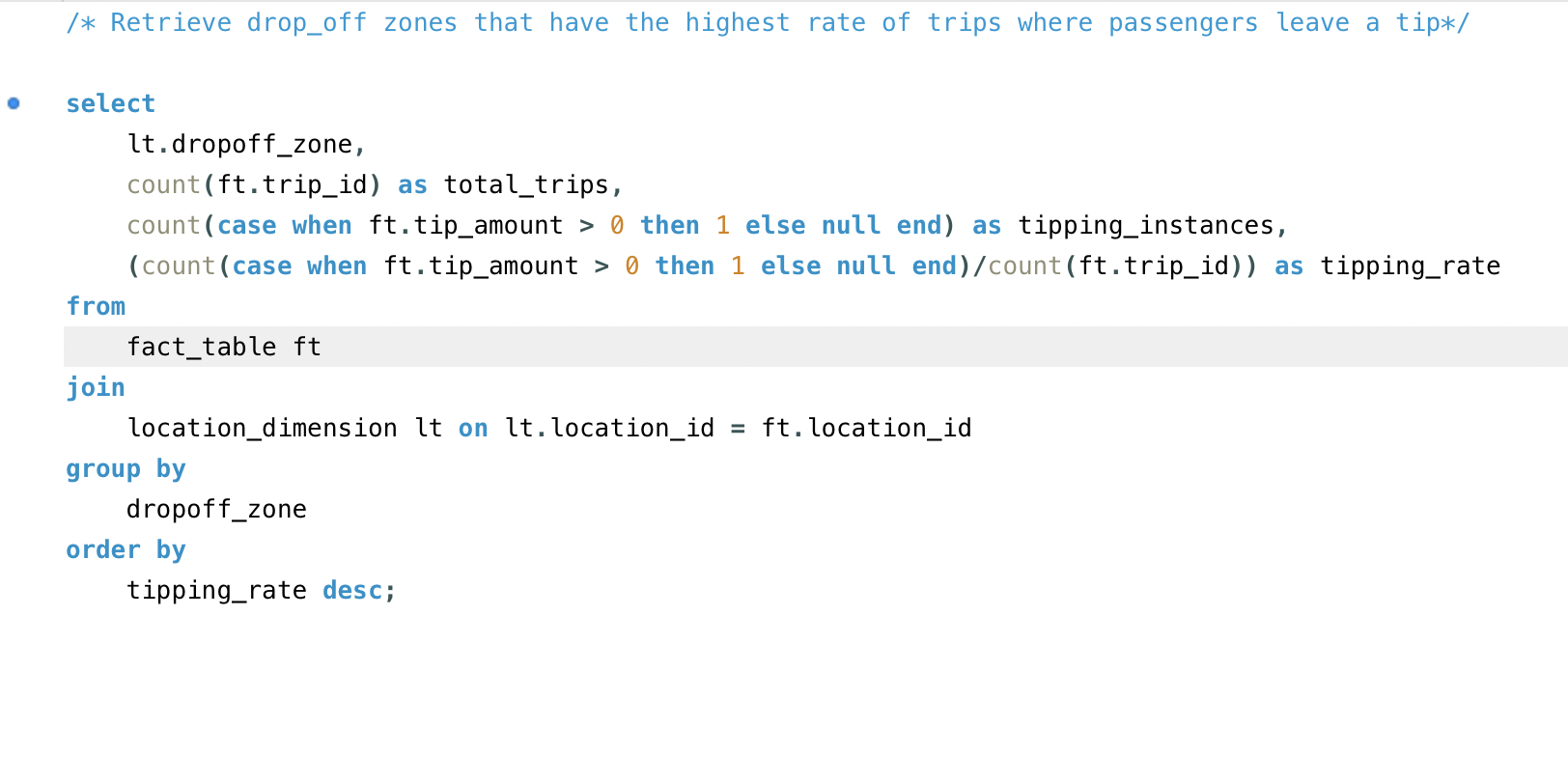
D. **Day and Average Hour Revenue and Tips:** Identify which hours of each day of the week have the highest demand and highest average revenue



E. **Average tip percentage by month:** Identify zones on the basis of average tip percentage and total trips for each month. Average excludes outliers greater than 200% (2)



F. **Tipping Rate vs Tip Amount:** Retrieve dropoff zones that show both highest tipping rate, and highest average tip; Needed to be mindful of outliers, as some routes had single trips that had large tip amounts.



**6. Process to setup the project :**

1. Unzip the CabCapitalConsulting folder
2. Open the Yellow\_Taxi\_ETL.ipynb file in your Jupyter Notebook
3. Replace the following fields with respect to your local environment :
   1. save\_dir : replace the path with your local folder to download the taxi data
   2. Replace the values for the following variables with respect to the local MySQL instance :

username = 'root'

password = ‘######’

hostname = 'localhost'

1. Run the Yellow\_Taxi\_ETL.ipynb script
2. Open your MySQL instance and it should have created a schema : ‘bi\_and\_dw’ and a table ‘source\_table’. The ‘source\_table’ should have all the records imported from the dataframe(approx ~ 12.2 million)
3. Open the ‘Create\_Dimension\_Tables.sql’ file and run it on MySQL. It should create the following dimension tables :
   1. vendor\_dimenison
   2. datetime\_dimenison
   3. location\_dimenison
   4. rate\_dimenison
   5. paymenttype\_dimenison
4. Open the ‘Create\_Fact\_Table.sql’ file and run it. It should create the table ‘fact\_table’
5. Open the ‘Insert\_Data.sql’ file and run it. It should insert the data in the dimension and fact tables. This will take a while to run(apprx~30 mins)
6. Run the bi queries files one by one in the OLAP queries folder to see the result.