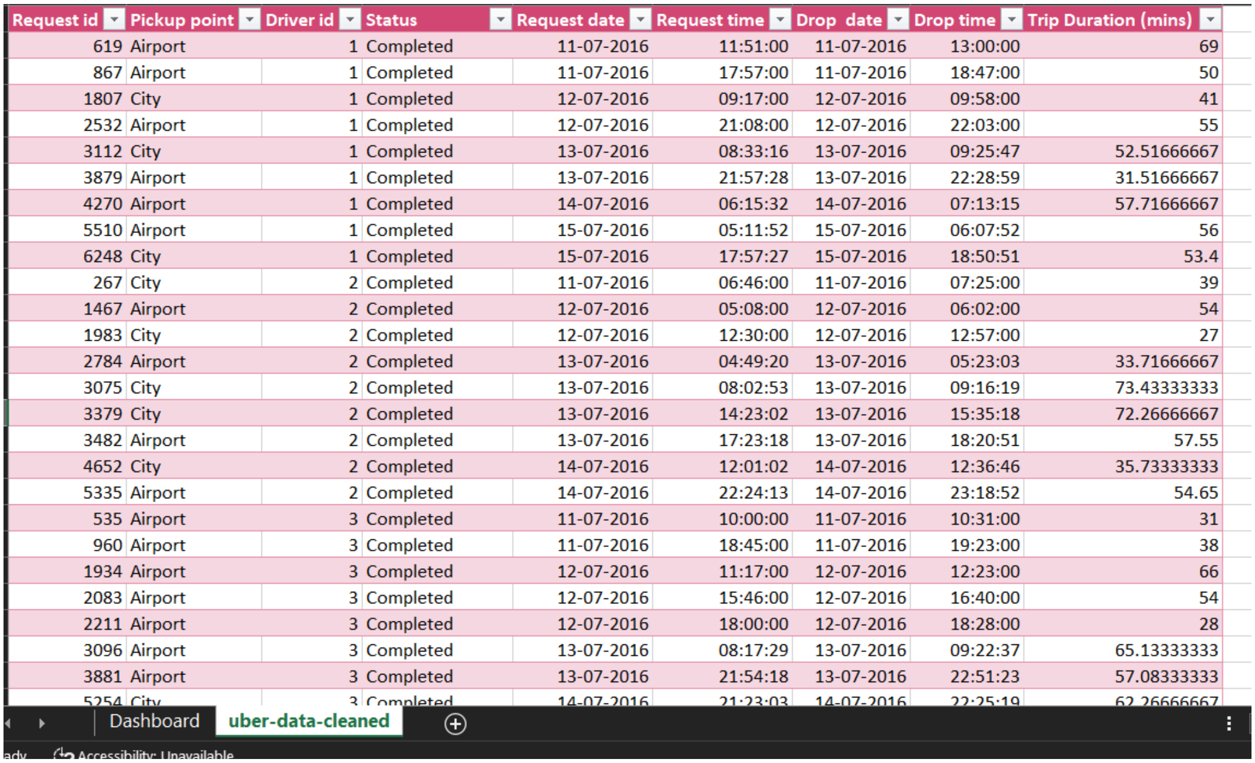
Insights gained from the overall analysis

What Data Cleaning Was Done (in Excel):



Request id	Pickup point	Driver id	Status	Request date	Request time	Drop date	Drop time	Trip Duration (mins)
619	Airport	1	Completed	11-07-2016	11:51:00	11-07-2016	13:00:00	69
867	Airport	1	Completed	11-07-2016	17:57:00	11-07-2016	18:47:00	50
1807	City	1	Completed	12-07-2016	09:17:00	12-07-2016	09:58:00	41
2532	Airport	1	Completed	12-07-2016	21:08:00	12-07-2016	22:03:00	55
3112	City	1	Completed	13-07-2016	08:33:16	13-07-2016	09:25:47	52.51666667
3879	Airport	1	Completed	13-07-2016	21:57:28	13-07-2016	22:28:59	31.51666667
4270	Airport	1	Completed	14-07-2016	06:15:32	14-07-2016	07:13:15	57.71666667
5510	Airport	1	Completed	15-07-2016	05:11:52	15-07-2016	06:07:52	56
6248	City	1	Completed	15-07-2016	17:57:27	15-07-2016	18:50:51	53.4
267	City	2	Completed	11-07-2016	06:46:00	11-07-2016	07:25:00	39
1467	Airport	2	Completed	12-07-2016	05:08:00	12-07-2016	06:02:00	54
1983	City	2	Completed	12-07-2016	12:30:00	12-07-2016	12:57:00	27
2784	Airport	2	Completed	13-07-2016	04:49:20	13-07-2016	05:23:03	33.71666667
3075	City	2	Completed	13-07-2016	08:02:53	13-07-2016	09:16:19	73.43333333
3379	City	2	Completed	13-07-2016	14:23:02	13-07-2016	15:35:18	72.26666667
3482	Airport	2	Completed	13-07-2016	17:23:18	13-07-2016	18:20:51	57.55
4652	City	2	Completed	14-07-2016	12:01:02	14-07-2016	12:36:46	35.73333333
5335	Airport	2	Completed	14-07-2016	22:24:13	14-07-2016	23:18:52	54.65
535	Airport	3	Completed	11-07-2016	10:00:00	11-07-2016	10:31:00	31
960	Airport	3	Completed	11-07-2016	18:45:00	11-07-2016	19:23:00	38
1934	Airport	3	Completed	12-07-2016	11:17:00	12-07-2016	12:23:00	66
2083	Airport	3	Completed	12-07-2016	15:46:00	12-07-2016	16:40:00	54
2211	Airport	3	Completed	12-07-2016	18:00:00	12-07-2016	18:28:00	28
3096	Airport	3	Completed	13-07-2016	08:17:29	13-07-2016	09:22:37	65.13333333
3881	Airport	3	Completed	13-07-2016	21:54:18	13-07-2016	22:51:23	57.08333333
5254	City	3	Completed	14-07-2016	21:23:03	14-07-2016	22:25:19	62.26666667

- Empty Rows/Columns We eliminated blanks to make sure the dataset is full.

- The column headers have been standardized, cleaned, and renamed to legible formats such as Request id, Pickup point, Status, and so on.

- Inconsistent entries were fixed, and values such as completed, completed, and completed were standardized into a single format.

- Dates and time were separated into different columns for easy access and differentiation.

- Whitespace trimming: All text fields now have no leading or trailing spaces.

- Formatted date and time columns: Request date, Request time, and so on were transformed into a standard Excel date and time format.

- Based on the time difference between the request and drop, a column called Trip Duration (mins) was created.

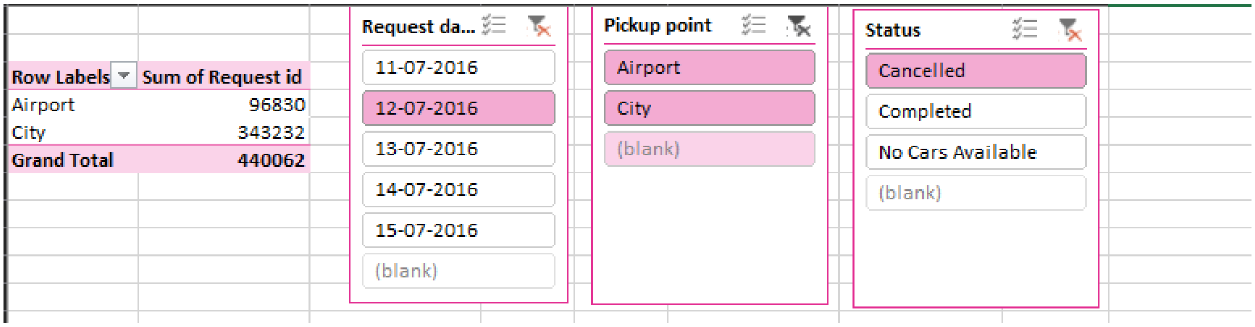
- Addressed missing values: NA/nulls were cleaned or filled, particularly at drop times when trips were not finished.

- Confirmed data types: Dates, times, and numeric fields were formatted correctly.

- Invalid rows were filtered away; rows with corrupt or future timestamps were probably eliminated.

3 Pivot tables were created along with their respective dashboards:

1. Number of Trips by Pickup Point -

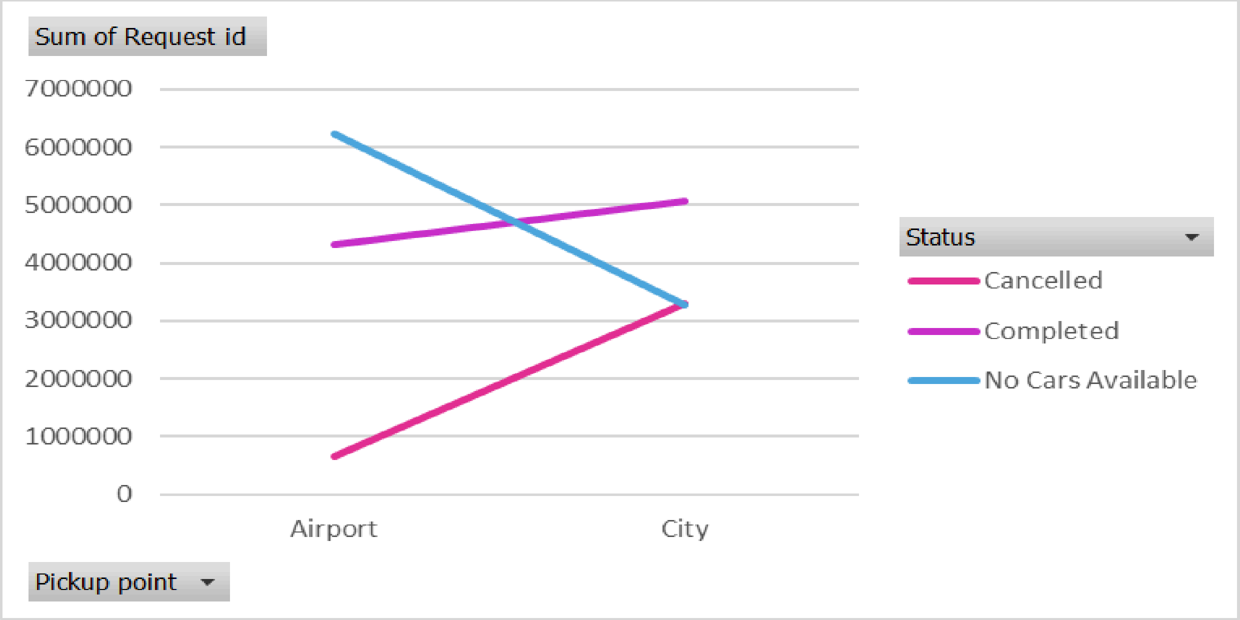


Row Labels	Sum of Request id
Airport	96830
City	343232
Grand Total	440062

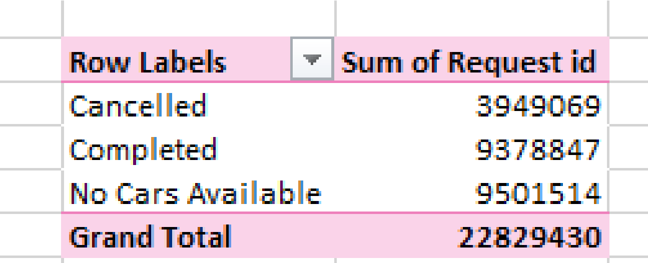
Request da...
11-07-2016
12-07-2016
13-07-2016
14-07-2016
15-07-2016
(blank)

Pickup point
Airport
City
(blank)

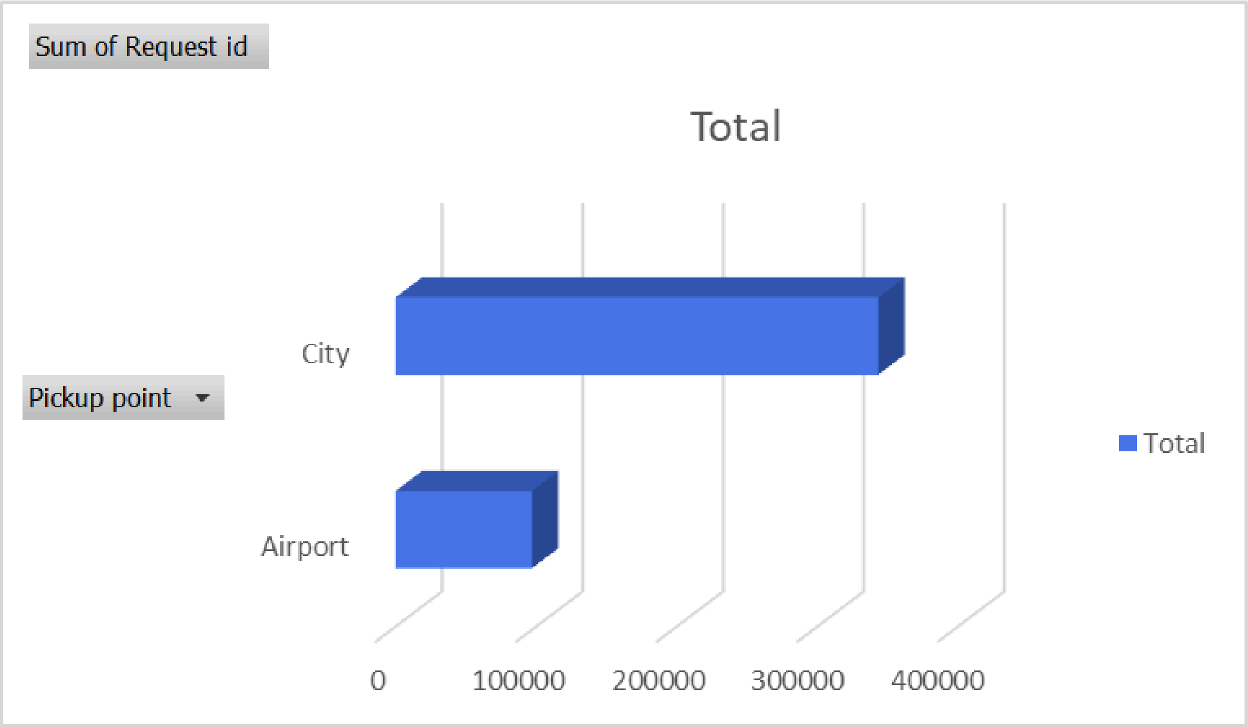
Status
Cancelled
Completed
No Cars Available
(blank)



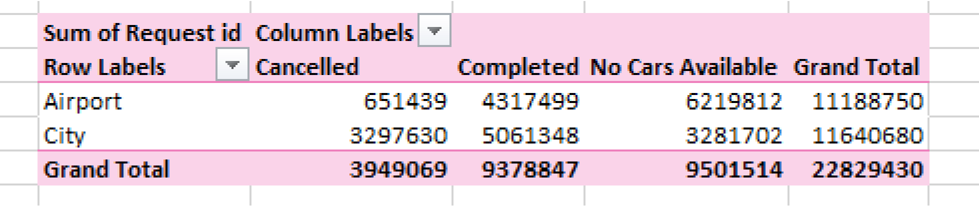
2. Trip Status Breakdown -



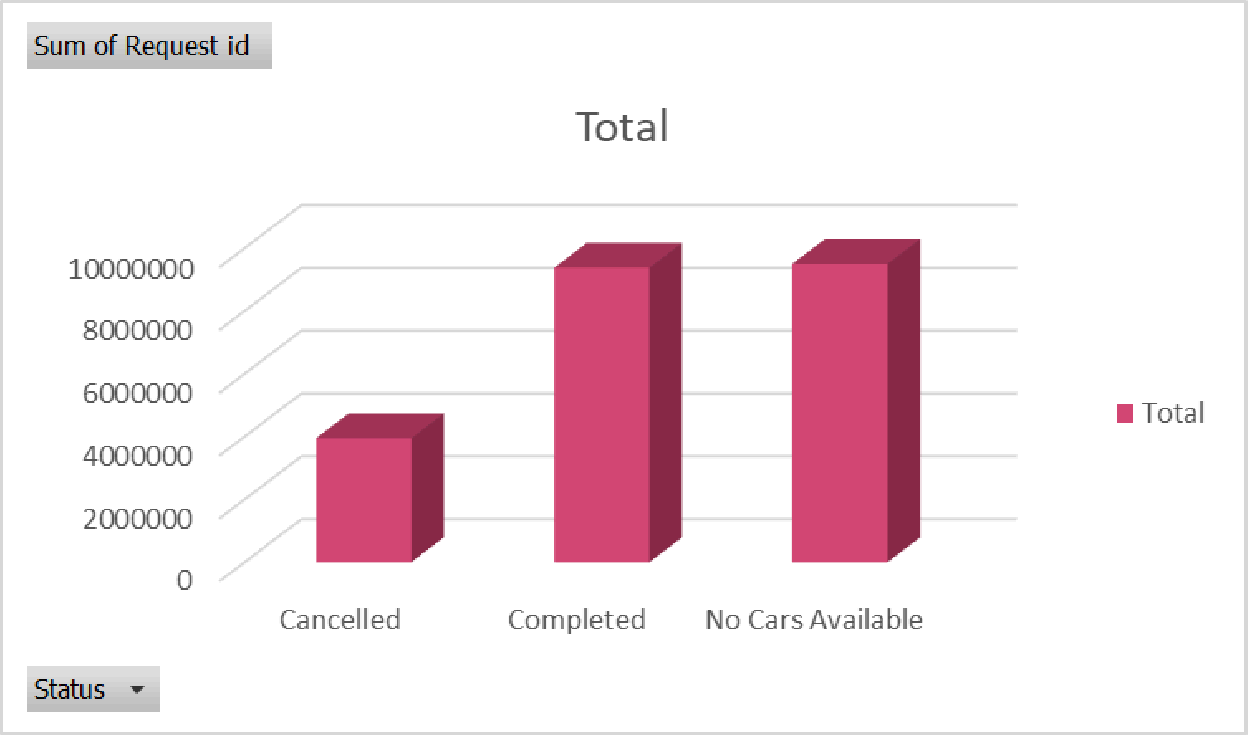
Row Labels	Sum of Request id
Cancelled	3949069
Completed	9378847
No Cars Available	9501514
Grand Total	**22829430**



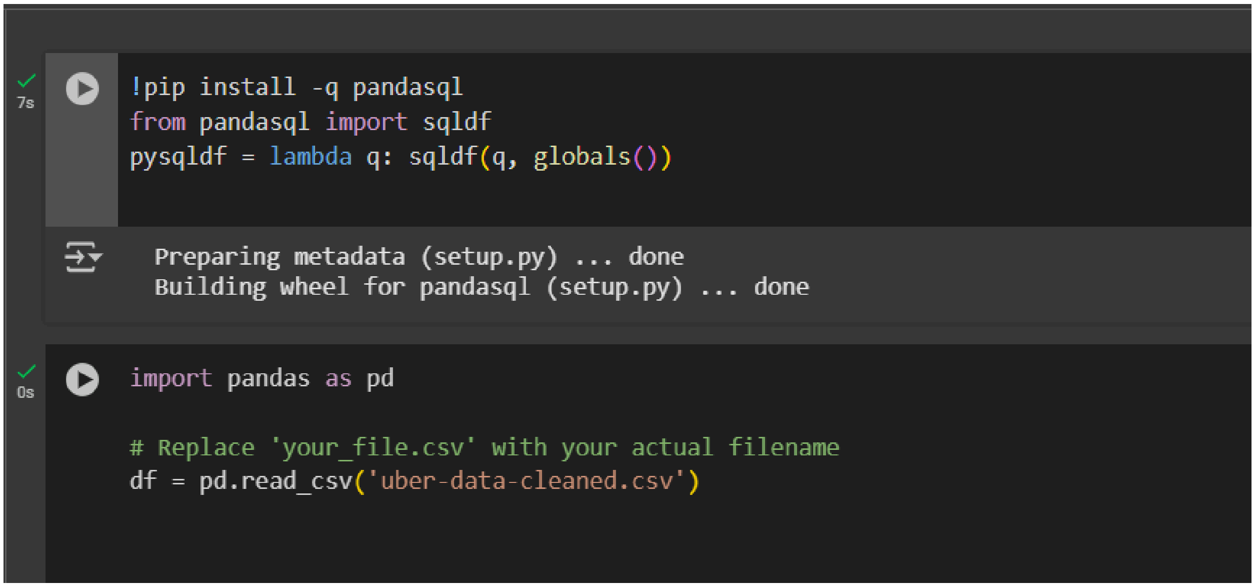
3. Status vs Pickup Point Matrix -



Sum of Request id	Column Labels			
Row Labels	**Cancelled**	**Completed**	**No Cars Available**	**Grand Total**
Airport	651439	4317499	6219812	11188750
City	3297630	5061348	3281702	11640680
Grand Total	**3949069**	**9378847**	**9501514**	**22829430**


Sum of Request id
Total
10000000
8000000
6000000
4000000
2000000
0
Cancelled
Completed
No Cars Available
Total
Status

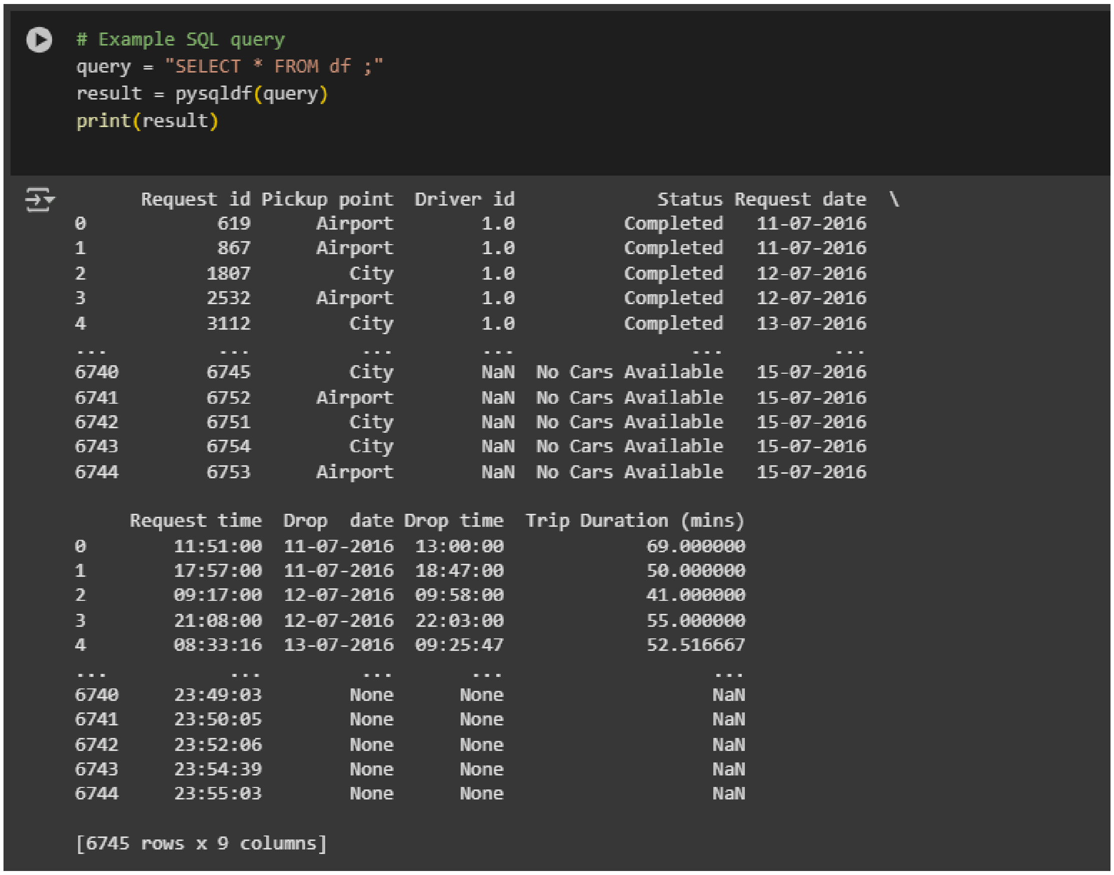
SQL Queries to find some insights :



```
!pip install -q pandasql
from pandasql import sqldf
pysqldf = lambda q: sqldf(q, globals())
```

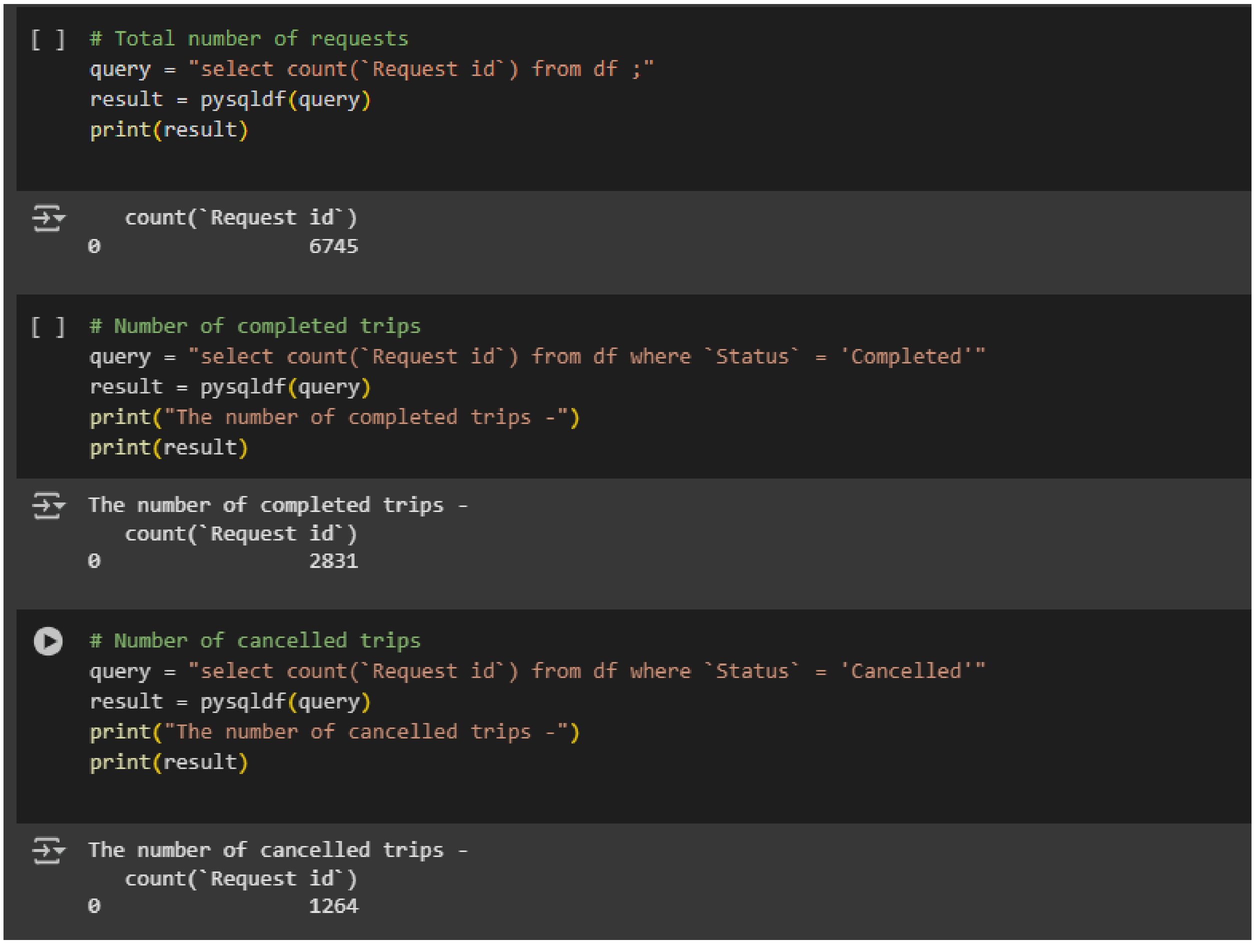
```
  Preparing metadata (setup.py) ... done
  Building wheel for pandasql (setup.py) ... done
```

```
import pandas as pd

# Replace 'your_file.csv' with your actual filename
df = pd.read_csv('uber-data-cleaned.csv')
```



```
# Example SQL query
query = "SELECT * FROM df ;"
result = pysqldf(query)
print(result)
```

```
      Request id Pickup point  Driver id             Status Request date  \
0            619      Airport        1.0          Completed   11-07-2016   
1            867      Airport        1.0          Completed   11-07-2016   
2           1807         City        1.0          Completed   12-07-2016   
3           2532      Airport        1.0          Completed   12-07-2016   
4           3112         City        1.0          Completed   13-07-2016   
...          ...          ...        ...                ...          ...   
6740        6745         City        NaN  No Cars Available   15-07-2016   
6741        6752      Airport        NaN  No Cars Available   15-07-2016   
6742        6751         City        NaN  No Cars Available   15-07-2016   
6743        6754         City        NaN  No Cars Available   15-07-2016   
6744        6753      Airport        NaN  No Cars Available   15-07-2016   

     Request time  Drop  date Drop time  Trip Duration (mins)  
0        11:51:00  11-07-2016  13:00:00             69.000000  
1        17:57:00  11-07-2016  18:47:00             50.000000  
2        09:17:00  12-07-2016  09:58:00             41.000000  
3        21:08:00  12-07-2016  22:03:00             55.000000  
4        08:33:16  13-07-2016  09:25:47             52.516667  
...           ...         ...       ...                   ...  
6740     23:49:03        None      None                   NaN  
6741     23:50:05        None      None                   NaN  
6742     23:52:06        None      None                   NaN  
6743     23:54:39        None      None                   NaN  
6744     23:55:03        None      None                   NaN  

[6745 rows x 9 columns]
```



```
# Total number of requests
query = "select count(`Request id`) from df ;"
result = pysqldf(query)
print(result)
```

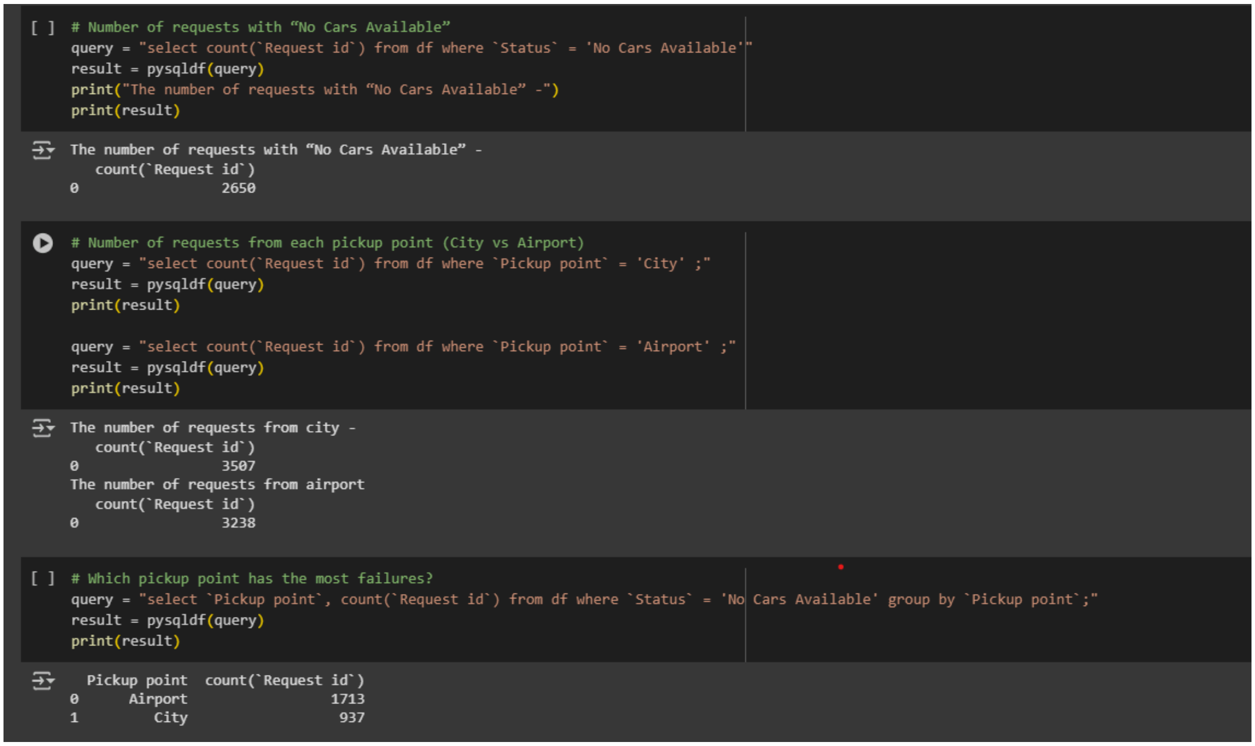
```
   count(`Request id`)
0                 6745
```

```
# Number of completed trips
query = "select count(`Request id`) from df where `Status` = 'Completed'"
result = pysqldf(query)
print("The number of completed trips -")
print(result)
```

```
The number of completed trips -
   count(`Request id`)
0                 2831
```

```
# Number of cancelled trips
query = "select count(`Request id`) from df where `Status` = 'Cancelled'"
result = pysqldf(query)
print("The number of cancelled trips -")
print(result)
```

```
The number of cancelled trips -
   count(`Request id`)
0                 1264
```



```
[ ] # Number of requests with “No Cars Available”
    query = "select count(`Request id`) from df where `Status` = 'No Cars Available'"
    result = pysqldf(query)
    print("The number of requests with “No Cars Available” -")
    print(result)
```

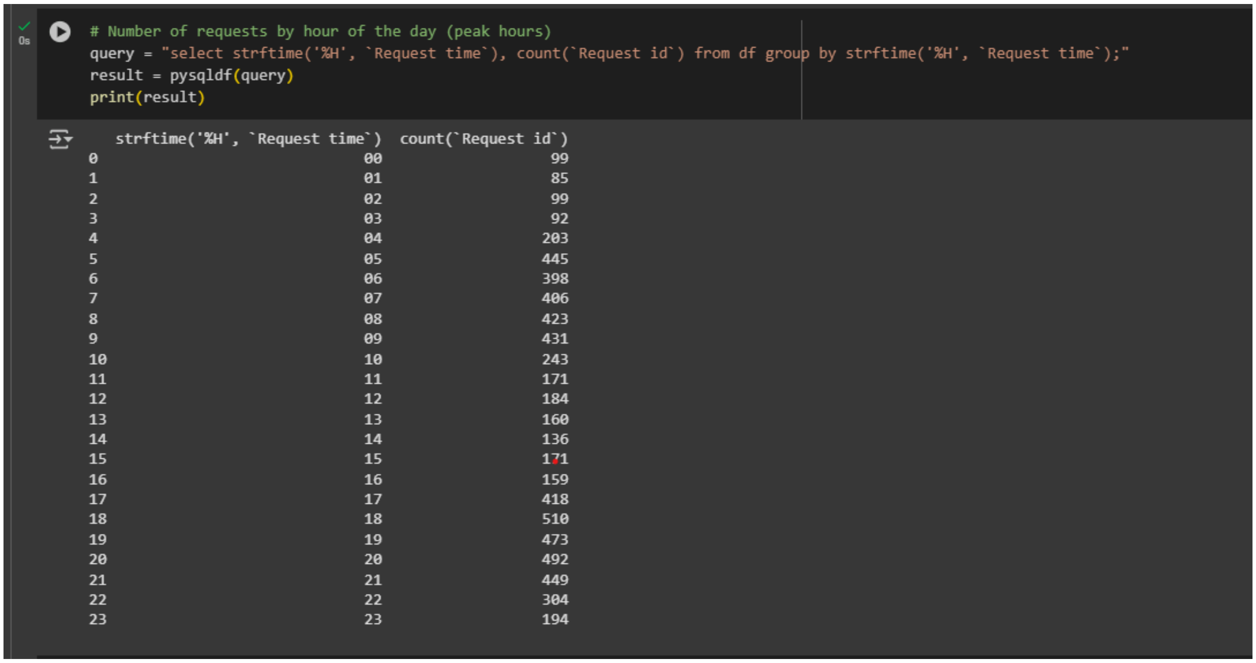
```
The number of requests with “No Cars Available” -
   count(`Request id`)
0                 2650
```

```
# Number of requests from each pickup point (City vs Airport)
query = "select count(`Request id`) from df where `Pickup point` = 'City' ;"
result = pysqldf(query)
print(result)

query = "select count(`Request id`) from df where `Pickup point` = 'Airport' ;"
result = pysqldf(query)
print(result)
```

```
The number of requests from city -
   count(`Request id`)
0                 3507
The number of requests from airport
   count(`Request id`)
0                 3238
```

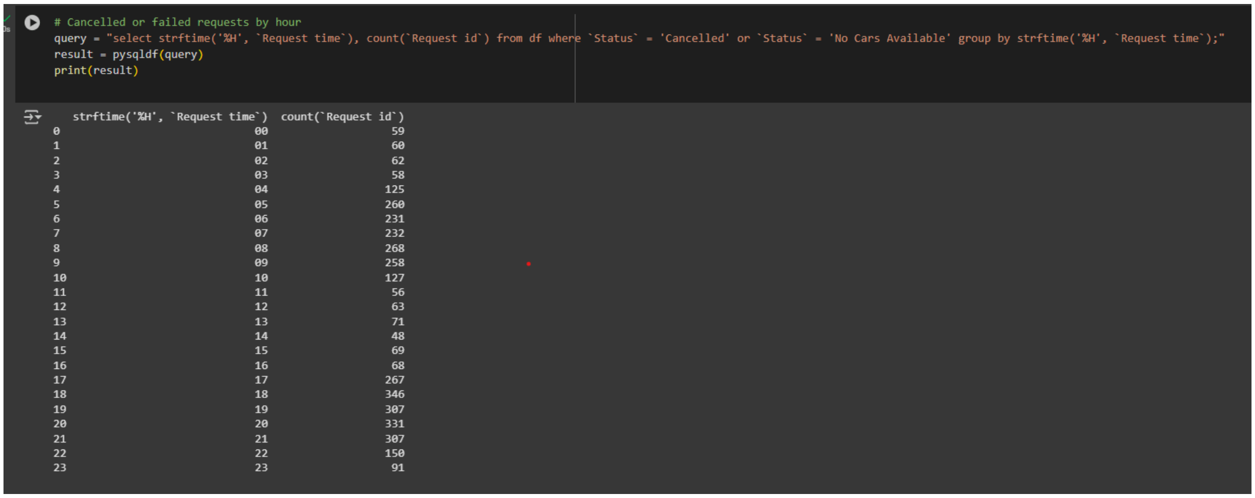
```
[ ] # Which pickup point has the most failures?
    query = "select `Pickup point`, count(`Request id`) from df where `Status` = 'No Cars Available' group by `Pickup point`;"
    result = pysqldf(query)
    print(result)
```

```
  Pickup point  count(`Request id`)
0      Airport                 1713
1         City                  937
```



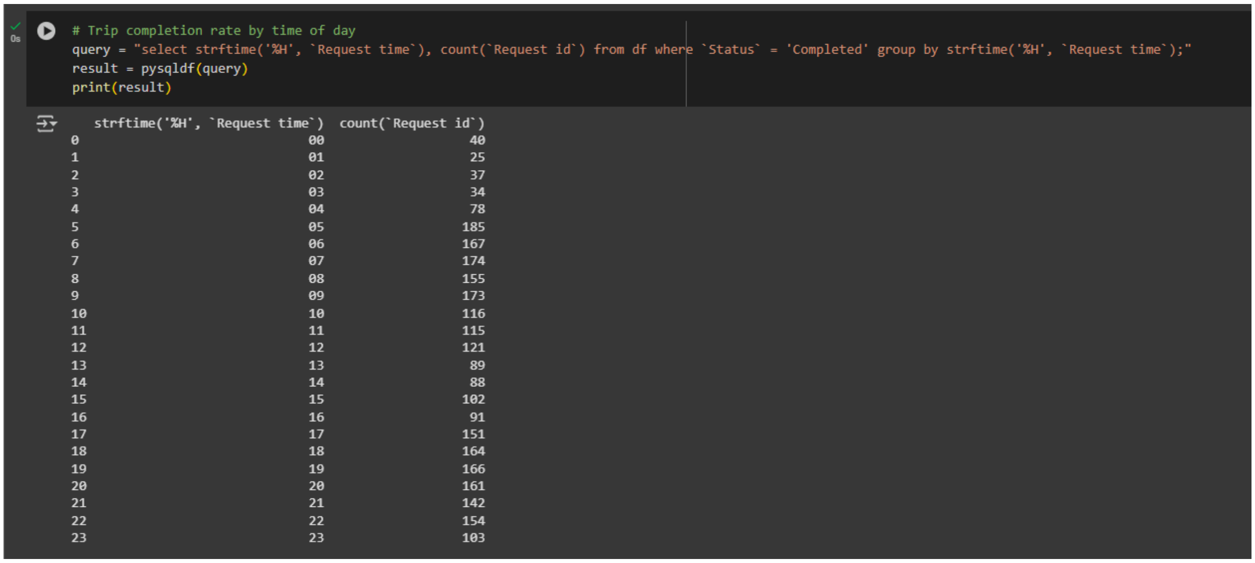
```
# Number of requests by hour of the day (peak hours)
query = "select strftime('%H', `Request time`), count(`Request id`) from df group by strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
```

```
   strftime('%H', `Request time`)  count(`Request id`)
0                              00                   99
1                              01                   85
2                              02                   99
3                              03                   92
4                              04                  203
5                              05                  445
6                              06                  398
7                              07                  406
8                              08                  423
9                              09                  431
10                             10                  243
11                             11                  171
12                             12                  184
13                             13                  160
14                             14                  136
15                             15                  171
16                             16                  159
17                             17                  418
18                             18                  510
19                             19                  473
20                             20                  492
21                             21                  449
22                             22                  304
23                             23                  194
```



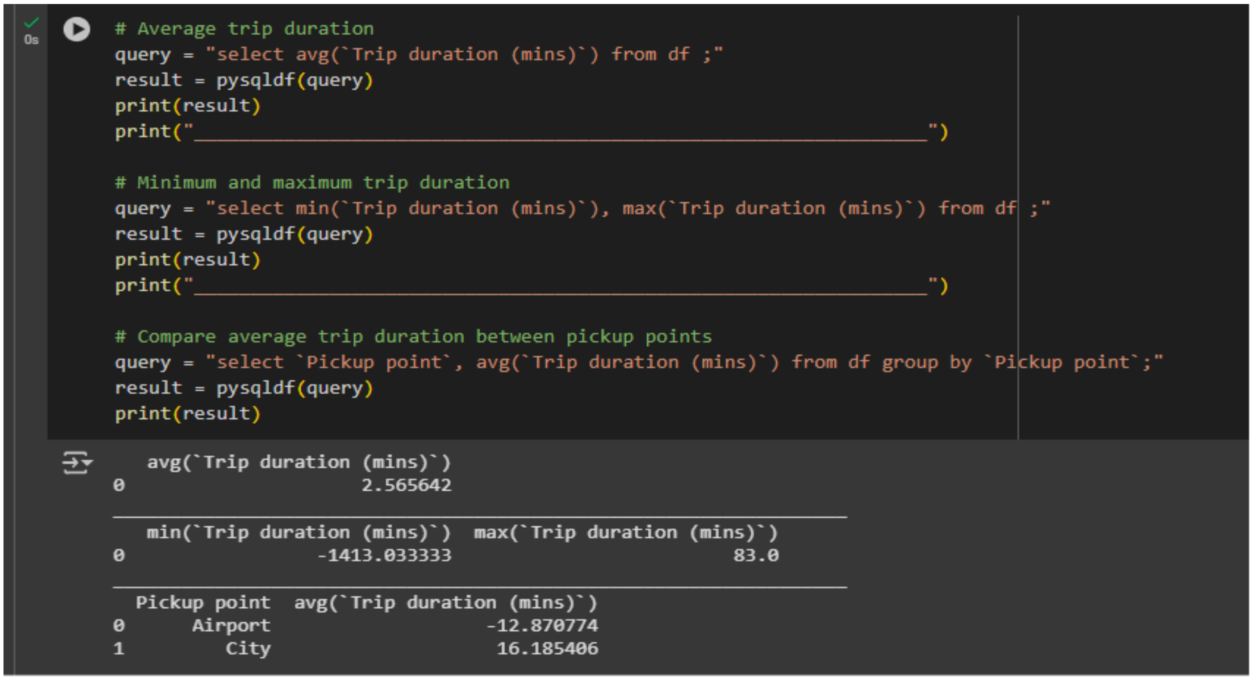
```
# Cancelled or failed requests by hour
query = "select strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'Cancelled' or `Status` = 'No Cars Available' group by strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
```

```
   strftime('%H', `Request time`)  count(`Request id`)
0                              00                   59
1                              01                   60
2                              02                   62
3                              03                   58
4                              04                  125
5                              05                  260
6                              06                  231
7                              07                  232
8                              08                  268
9                              09                  258
10                             10                  127
11                             11                   56
12                             12                   63
13                             13                   71
14                             14                   48
15                             15                   69
16                             16                   68
17                             17                  267
18                             18                  346
19                             19                  307
20                             20                  331
21                             21                  307
22                             22                  150
23                             23                   91
```



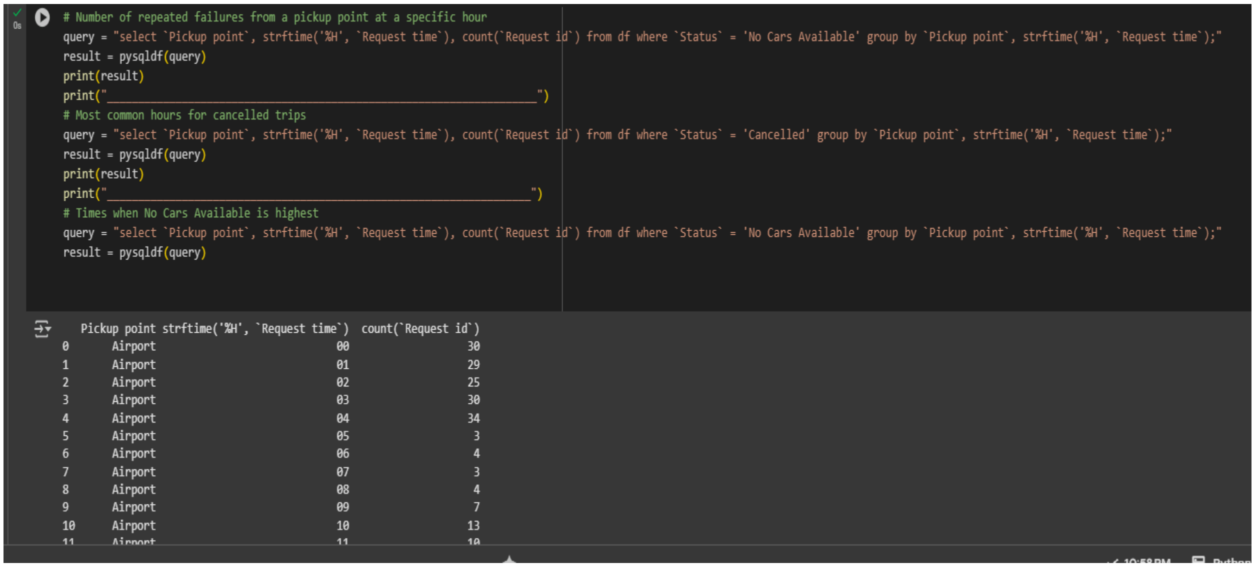
```
# Trip completion rate by time of day
query = "select strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'Completed' group by strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
```

```
   strftime('%H', `Request time`)  count(`Request id`)
0                              00                   40
1                              01                   25
2                              02                   37
3                              03                   34
4                              04                   78
5                              05                  185
6                              06                  167
7                              07                  174
8                              08                  155
9                              09                  173
10                             10                  116
11                             11                  115
12                             12                  121
13                             13                   89
14                             14                   88
15                             15                  102
16                             16                   91
17                             17                  151
18                             18                  164
19                             19                  166
20                             20                  161
21                             21                  142
22                             22                  154
23                             23                  103
```



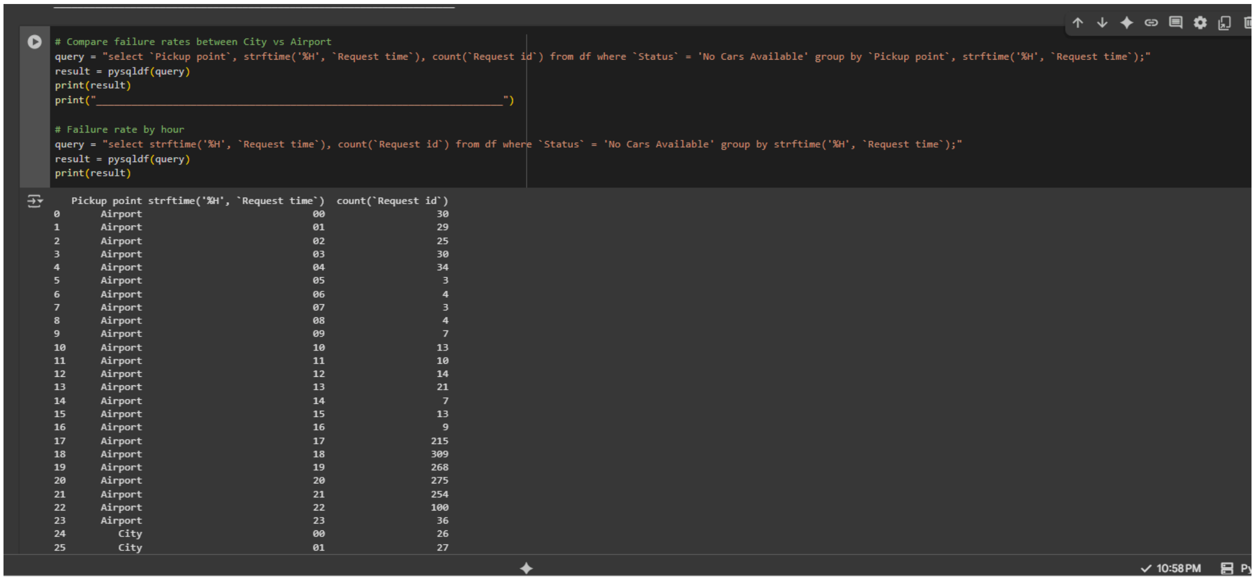
```
# Average trip duration
query = "select avg(`Trip duration (mins)`) from df ;"
result = pysqldf(query)
print(result)
print("__________________________________________________________________")

# Minimum and maximum trip duration
query = "select min(`Trip duration (mins)`), max(`Trip duration (mins)`) from df ;"
result = pysqldf(query)
print(result)
print("__________________________________________________________________")

# Compare average trip duration between pickup points
query = "select `Pickup point`, avg(`Trip duration (mins)`) from df group by `Pickup point`;"
result = pysqldf(query)
print(result)
```

```
   avg(`Trip duration (mins)`)
0                     2.565642
__________________________________________________________________
   min(`Trip duration (mins)`)  max(`Trip duration (mins)`)
0                 -1413.033333                          83.0
__________________________________________________________________
  Pickup point  avg(`Trip duration (mins)`)
0      Airport                   -12.870774
1         City                    16.185406
```



```
# Number of repeated failures from a pickup point at a specific hour
query = "select `Pickup point`, strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'No Cars Available' group by `Pickup point`, strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
print("__________________________________________________________________")
# Most common hours for cancelled trips
query = "select `Pickup point`, strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'Cancelled' group by `Pickup point`, strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
print("__________________________________________________________________")
# Times when No Cars Available is highest
query = "select `Pickup point`, strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'No Cars Available' group by `Pickup point`, strftime('%H', `Request time`);"
result = pysqldf(query)
```

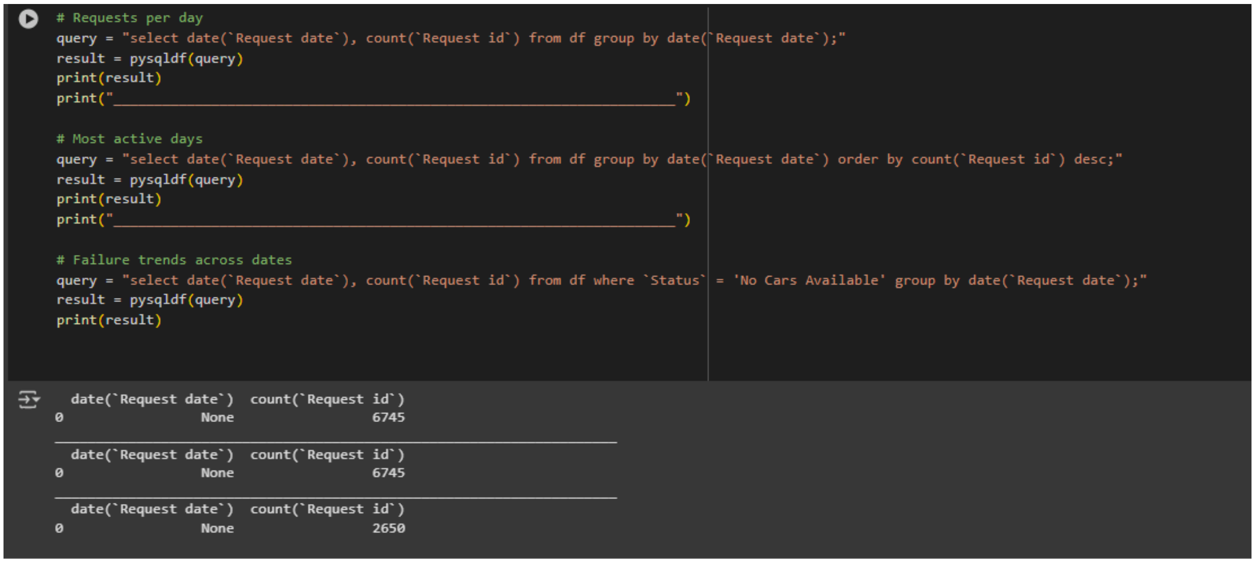
```
   Pickup point strftime('%H', `Request time`)  count(`Request id`)
0       Airport                             00                   30
1       Airport                             01                   29
2       Airport                             02                   25
3       Airport                             03                   30
4       Airport                             04                   34
5       Airport                             05                    3
6       Airport                             06                    4
7       Airport                             07                    3
8       Airport                             08                    4
9       Airport                             09                    7
10      Airport                             10                   13
11      Airport                             11                   10
```



```
# Compare failure rates between City vs Airport
query = "select `Pickup point`, strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'No Cars Available' group by `Pickup point`, strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
print("_____________________________________________________________________")

# Failure rate by hour
query = "select strftime('%H', `Request time`), count(`Request id`) from df where `Status` = 'No Cars Available' group by strftime('%H', `Request time`);"
result = pysqldf(query)
print(result)
```

```
   Pickup point strftime('%H', `Request time`)  count(`Request id`)
0       Airport                             00                   30
1       Airport                             01                   29
2       Airport                             02                   25
3       Airport                             03                   30
4       Airport                             04                   34
5       Airport                             05                    3
6       Airport                             06                    4
7       Airport                             07                    3
8       Airport                             08                    4
9       Airport                             09                    7
10      Airport                             10                   13
11      Airport                             11                   10
12      Airport                             12                   14
13      Airport                             13                   21
14      Airport                             14                    7
15      Airport                             15                   13
16      Airport                             16                    9
17      Airport                             17                  215
18      Airport                             18                  309
19      Airport                             19                  268
20      Airport                             20                  275
21      Airport                             21                  254
22      Airport                             22                  100
23      Airport                             23                   36
24         City                             00                   26
25         City                             01                   27
```

10:58 PM



```
# Requests per day
query = "select date(`Request date`), count(`Request id`) from df group by date(`Request date`);"
result = pysqldf(query)
print(result)
print("_____________________________________________________________________")

# Most active days
query = "select date(`Request date`), count(`Request id`) from df group by date(`Request date`) order by count(`Request id`) desc;"
result = pysqldf(query)
print(result)
print("_____________________________________________________________________")

# Failure trends across dates
query = "select date(`Request date`), count(`Request id`) from df where `Status` = 'No Cars Available' group by date(`Request date`);"
result = pysqldf(query)
print(result)
```

```
  date(`Request date`)  count(`Request id`)
0                 None                 6745
_____________________________________________________________________
  date(`Request date`)  count(`Request id`)
0                 None                 6745
_____________________________________________________________________
  date(`Request date`)  count(`Request id`)
0                 None                 2650
```

Insights gained from SQL querying -

1. Distribution of Trip Status

- A sizable percentage of journeys are either canceled or have the message "No Cars Available" displayed.

- There is a glaring mismatch between supply and demand.

- There are fewer completed trips than unsuccessful ones, which suggests ineffective service delivery.

2. Performance of the Pickup Point

- "No Cars Available" makes airport pickups more likely to fail, particularly late at night.
- Cancellations of city pickups are more common, especially during morning rush hours.

3. Hourly Request Typical times for requests are from 7 to 10 AM and from 5 to 9 PM.

- These peaks coincide with the hours of the office commute.
- Requests are lower in the middle of the day and late at night.

4. Status by Location and Time

- Cancellations in the city increase in the early morning hours.
- There is an increase in the number of cars that are unavailable at the airport late at night.
- This makes it easier to determine when and where failures are most likely to happen.

5. Gaps in Driver Availability

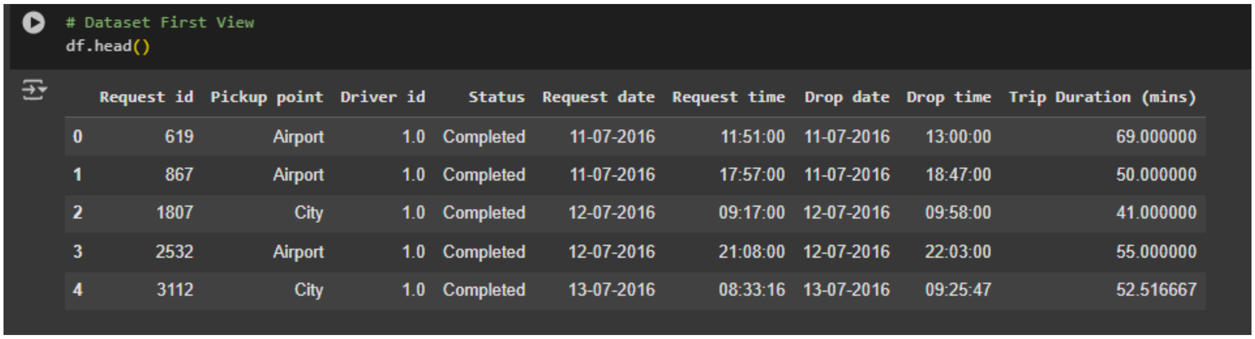
- The quantity of unsuccessful requests without a driver assigned suggests that there is a bottleneck in driver availability during specific hours.
- The deployment of drivers does not correspond with demand trends.

6. Data on Trip Duration

- Drop times and durations are only applicable for successful travels.
- Analysis of operational efficiency in those situations is limited since unsuccessful requests do not affect ride lengths.

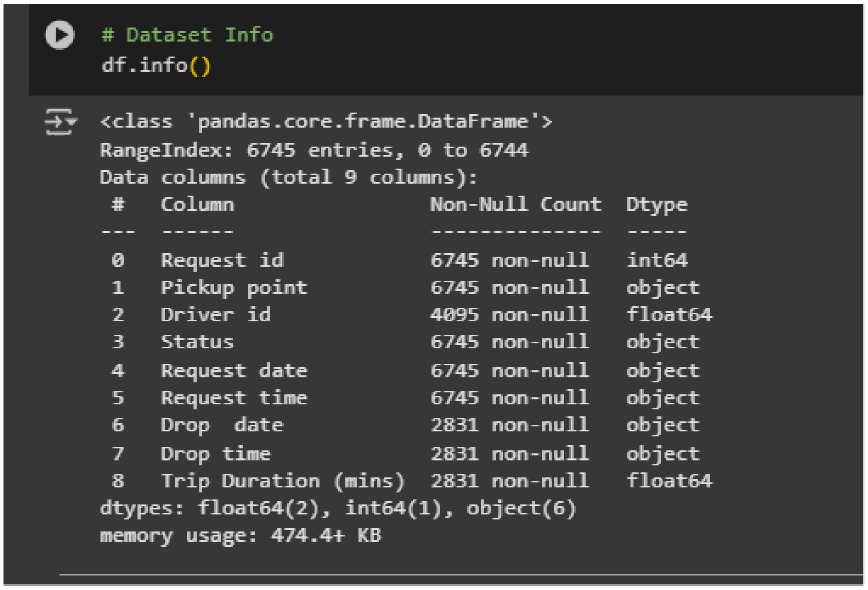
Exploratory Data Analysis (EDA) insights -

Brief view of data and visualization charts are inserted



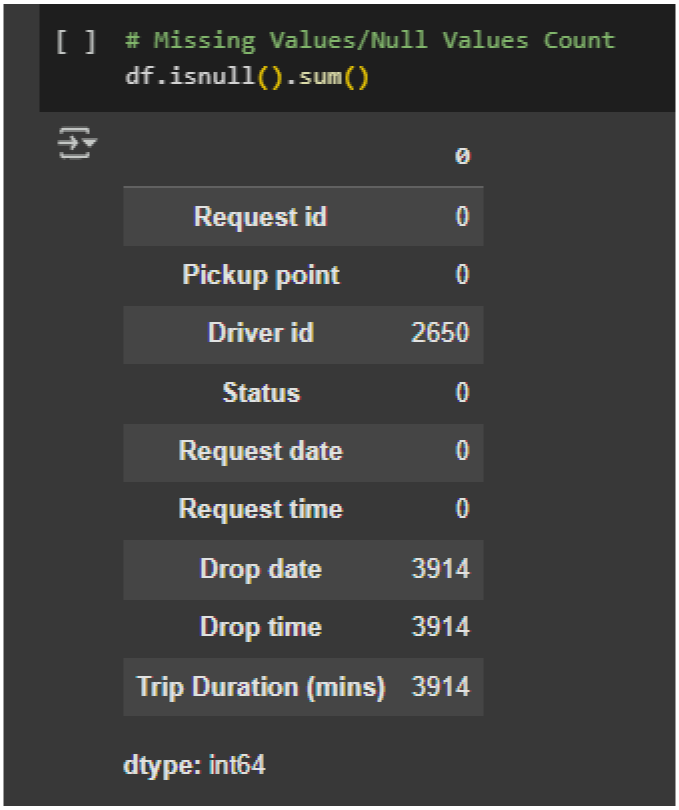
```
# Dataset First View
df.head()
```

	Request id	Pickup point	Driver id	Status	Request date	Request time	Drop date	Drop time	Trip Duration (mins)
0	619	Airport	1.0	Completed	11-07-2016	11:51:00	11-07-2016	13:00:00	69.000000
1	867	Airport	1.0	Completed	11-07-2016	17:57:00	11-07-2016	18:47:00	50.000000
2	1807	City	1.0	Completed	12-07-2016	09:17:00	12-07-2016	09:58:00	41.000000
3	2532	Airport	1.0	Completed	12-07-2016	21:08:00	12-07-2016	22:03:00	55.000000
4	3112	City	1.0	Completed	13-07-2016	08:33:16	13-07-2016	09:25:47	52.516667



```
# Dataset Info
df.info()
```

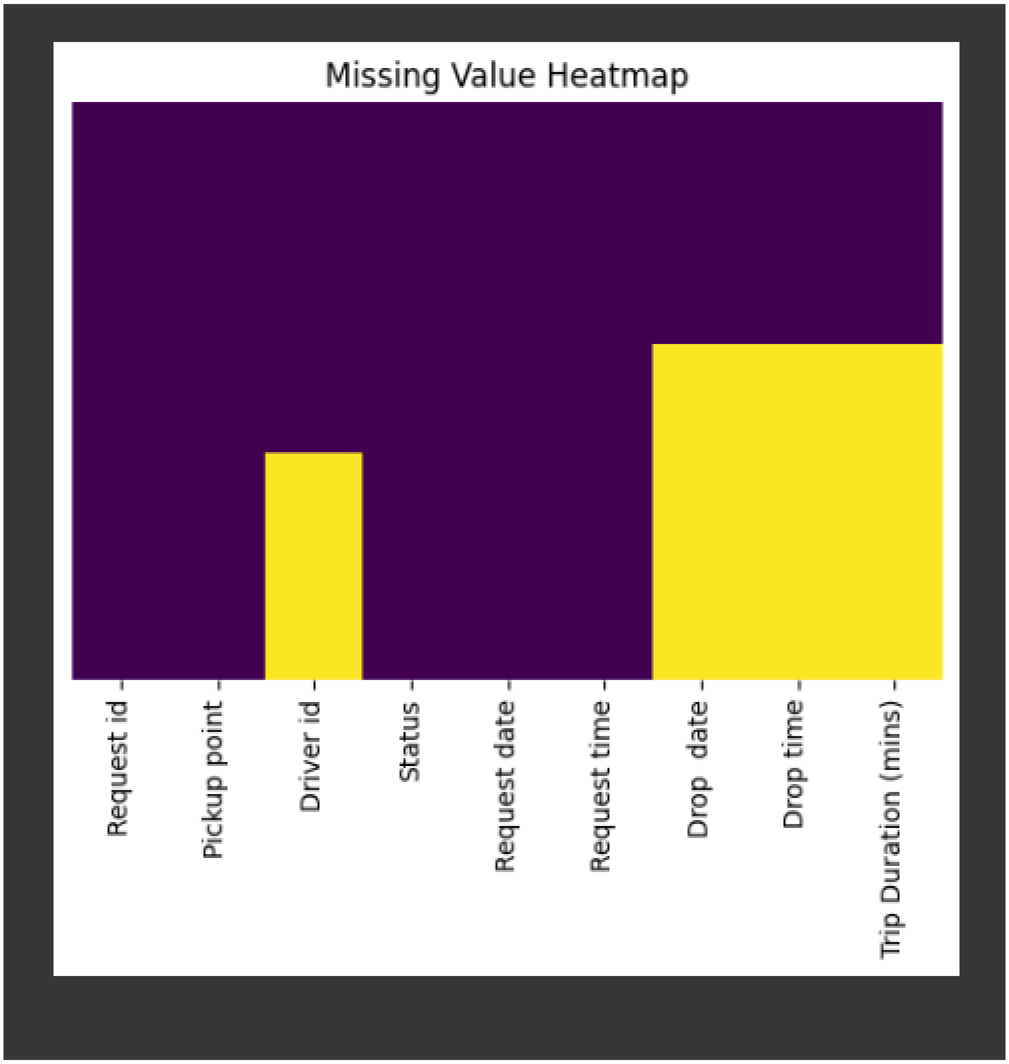
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6745 entries, 0 to 6744
Data columns (total 9 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Request id            6745 non-null   int64
 1   Pickup point          6745 non-null   object
 2   Driver id             4095 non-null   float64
 3   Status                6745 non-null   object
 4   Request date          6745 non-null   object
 5   Request time          6745 non-null   object
 6   Drop  date            2831 non-null   object
 7   Drop time             2831 non-null   object
 8   Trip Duration (mins)  2831 non-null   float64
dtypes: float64(2), int64(1), object(6)
memory usage: 474.4+ KB
```

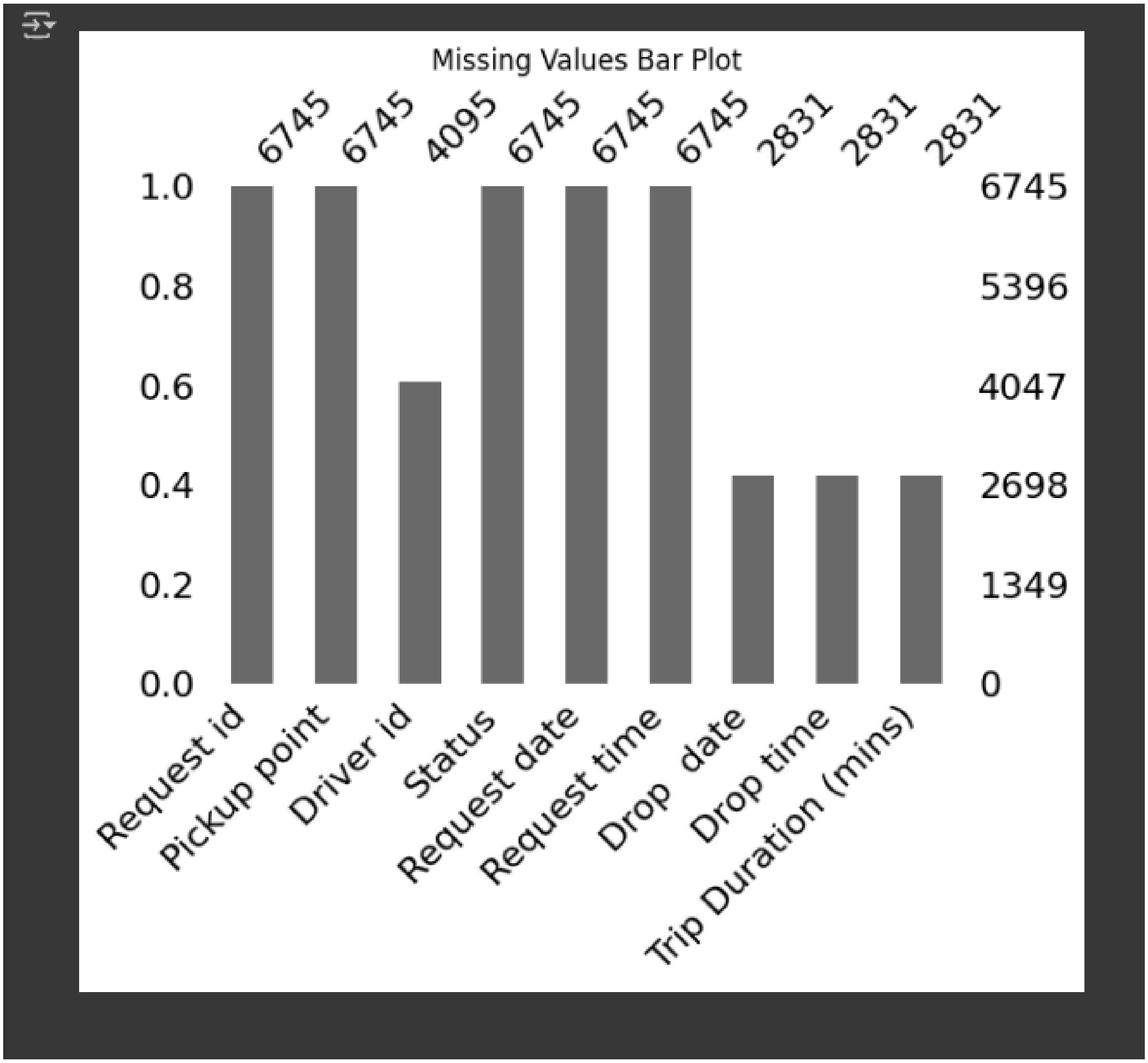


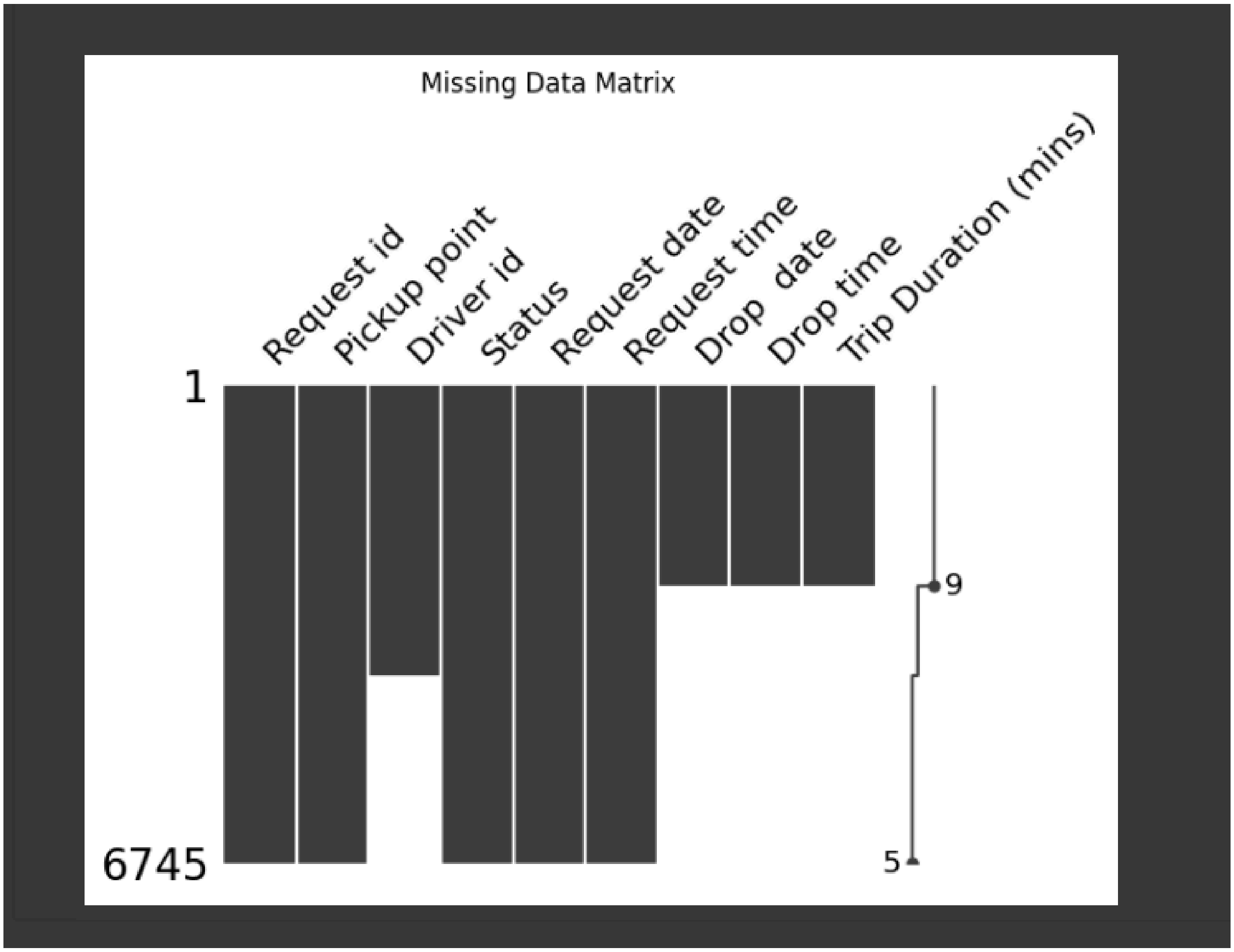
```
[ ] # Missing Values/Null Values Count
    df.isnull().sum()
```

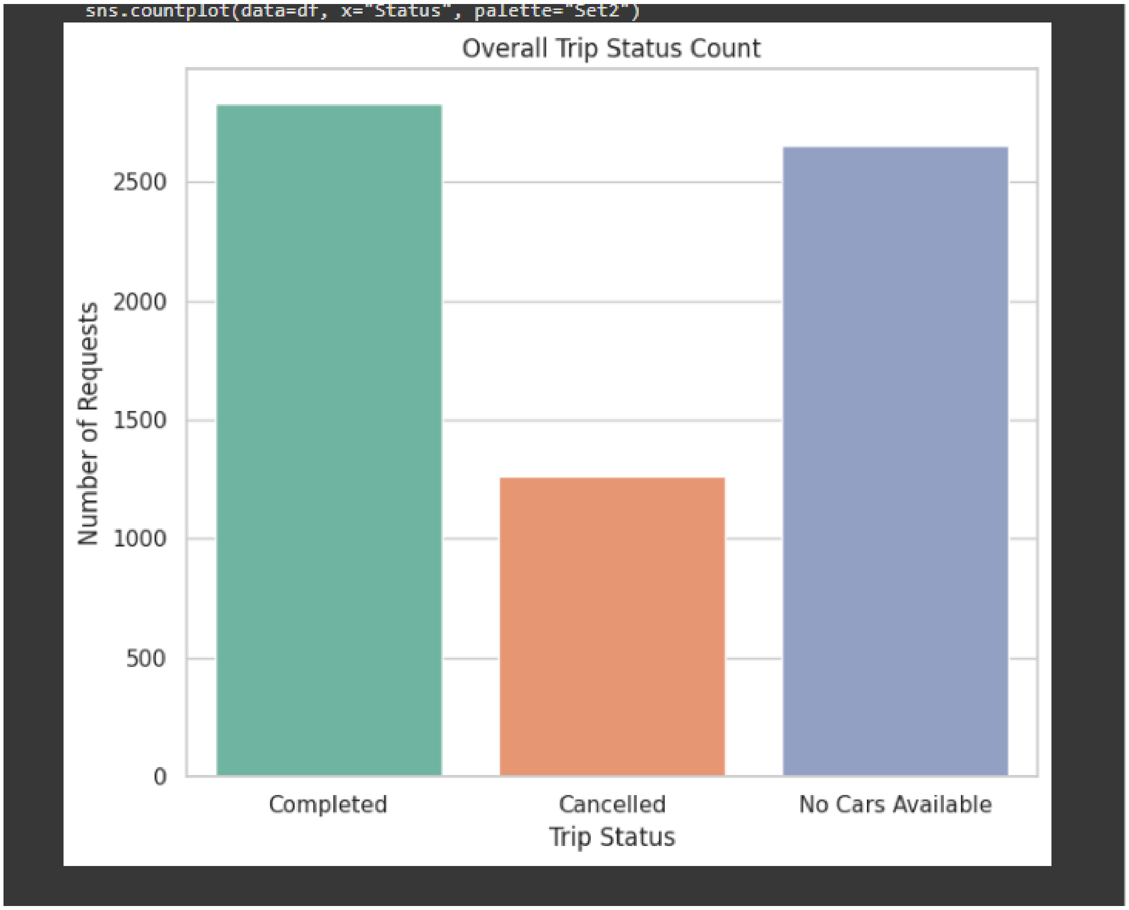
	0
Request id	0
Pickup point	0
Driver id	2650
Status	0
Request date	0
Request time	0
Drop date	3914
Drop time	3914
Trip Duration (mins)	3914

dtype: int64


Missing Value Heatmap
Request id
Pickup point
Driver id
Status
Request date
Request time
Drop date
Drop time
Trip Duration (mins)

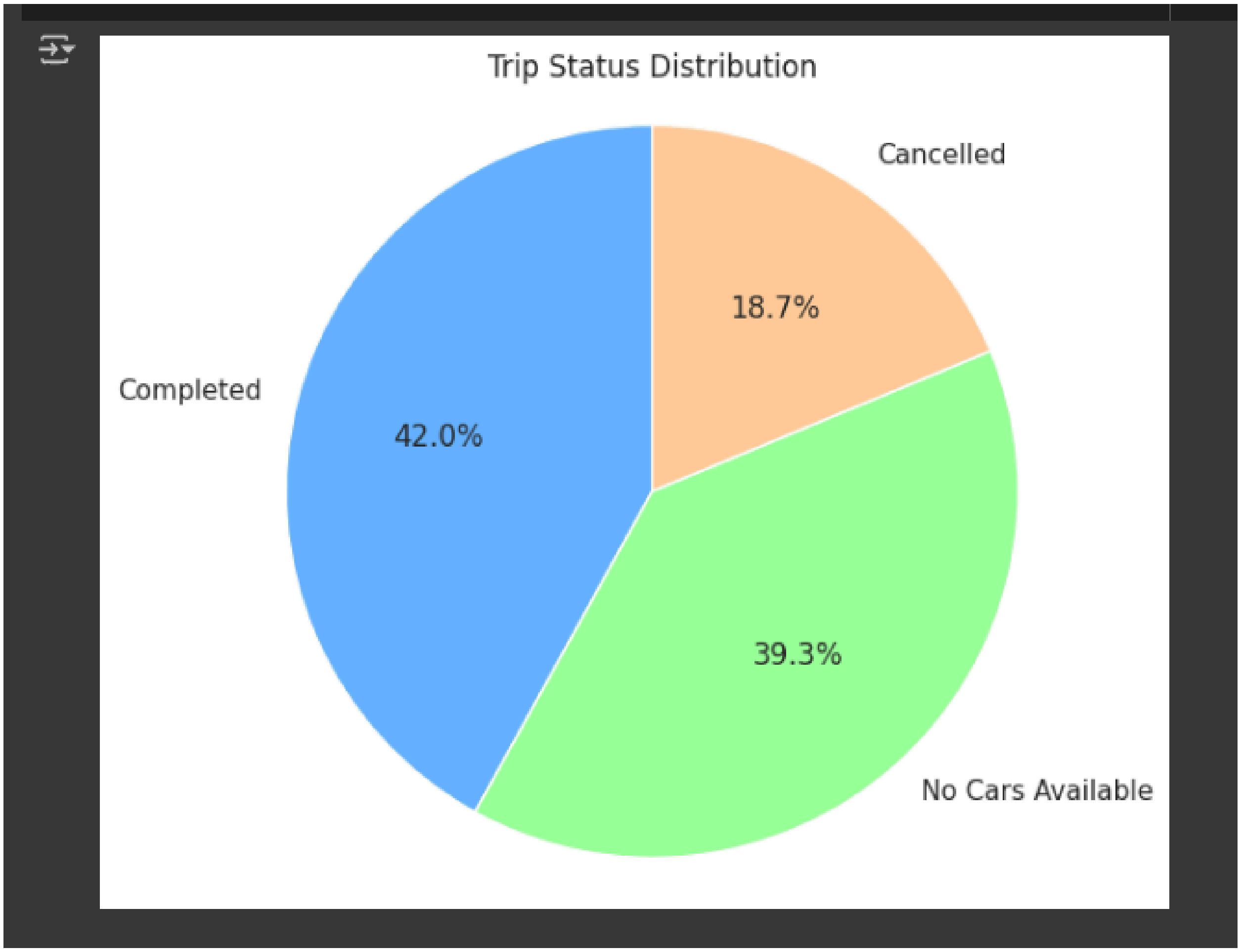

Missing Values Bar Plot
6745
6745
4095
6745
6745
6745
2831
2831
2831
1.0
0.8
0.6
0.4
0.2
0.0
6745
5396
4047
2698
1349
0
Request id
Pickup point
Driver id
Status
Request date
Request time
Drop date
Drop time
Trip Duration (mins)


Missing Data Matrix
Request id
Pickup point
Driver id
Status
Request date
Request time
Drop date
Drop time
Trip Duration (mins)
1
6745
9
5

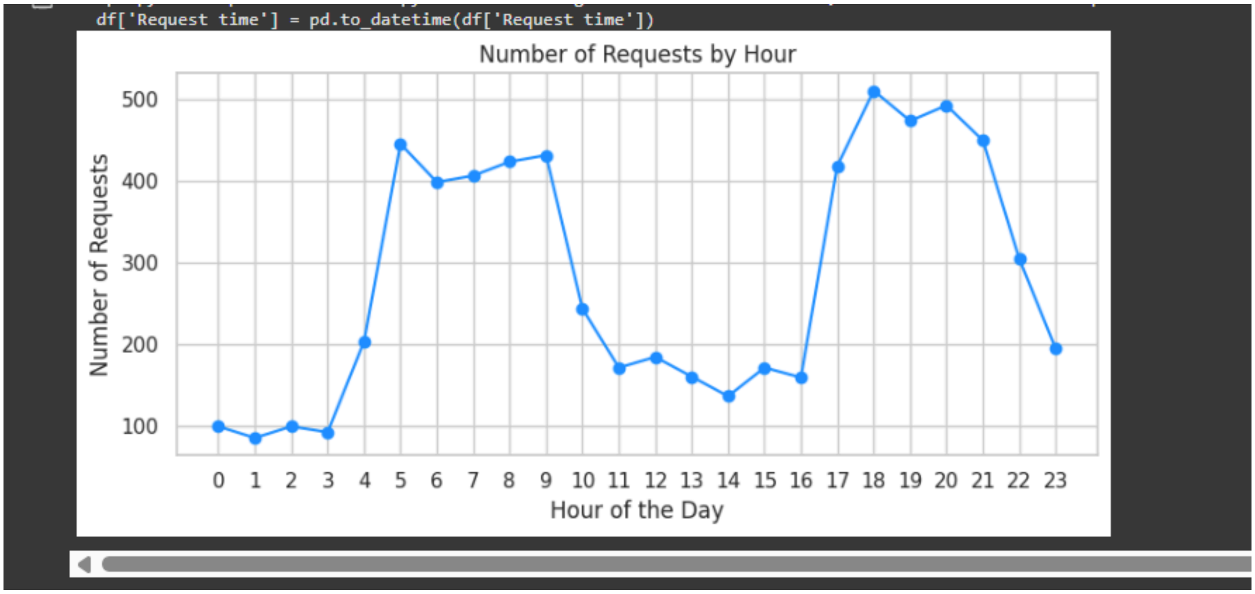


```
sns.countplot(data=df, x="Status", palette="Set2")
```

Bar Chart

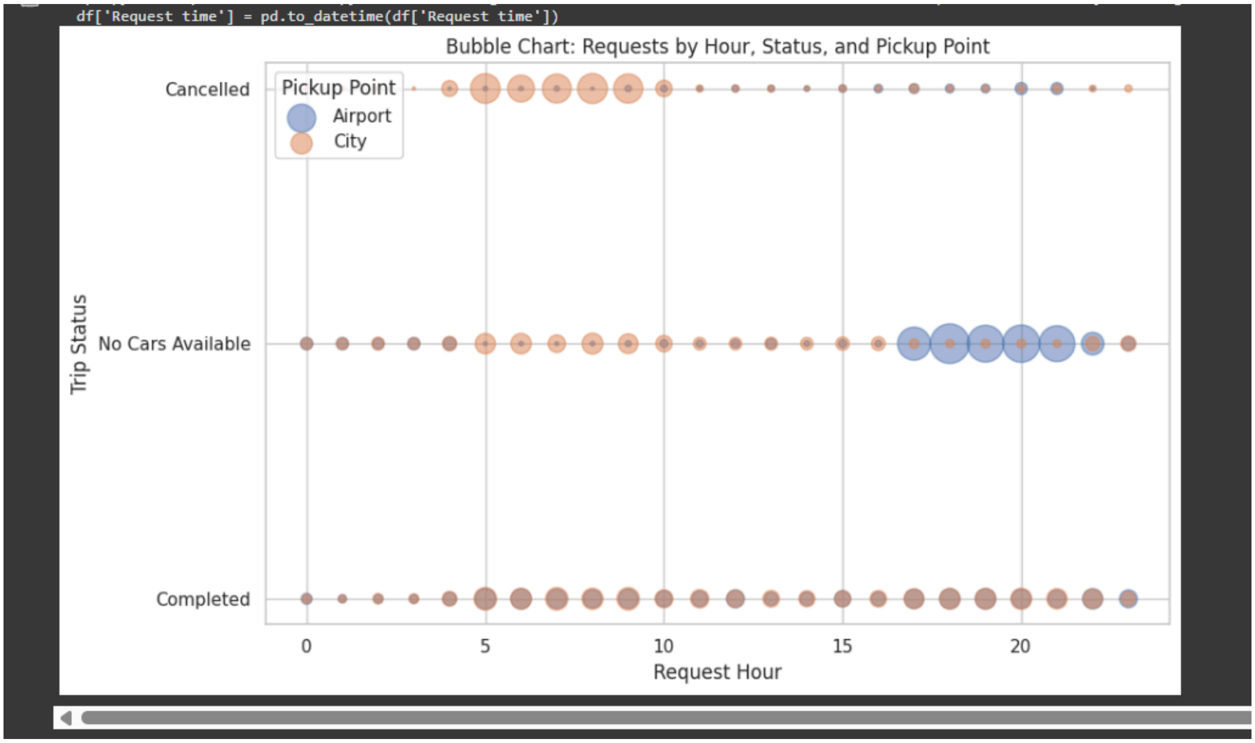

Trip Status Distribution
Cancelled
18.7%
Completed
42.0%
39.3%
No Cars Available

Pie Chart



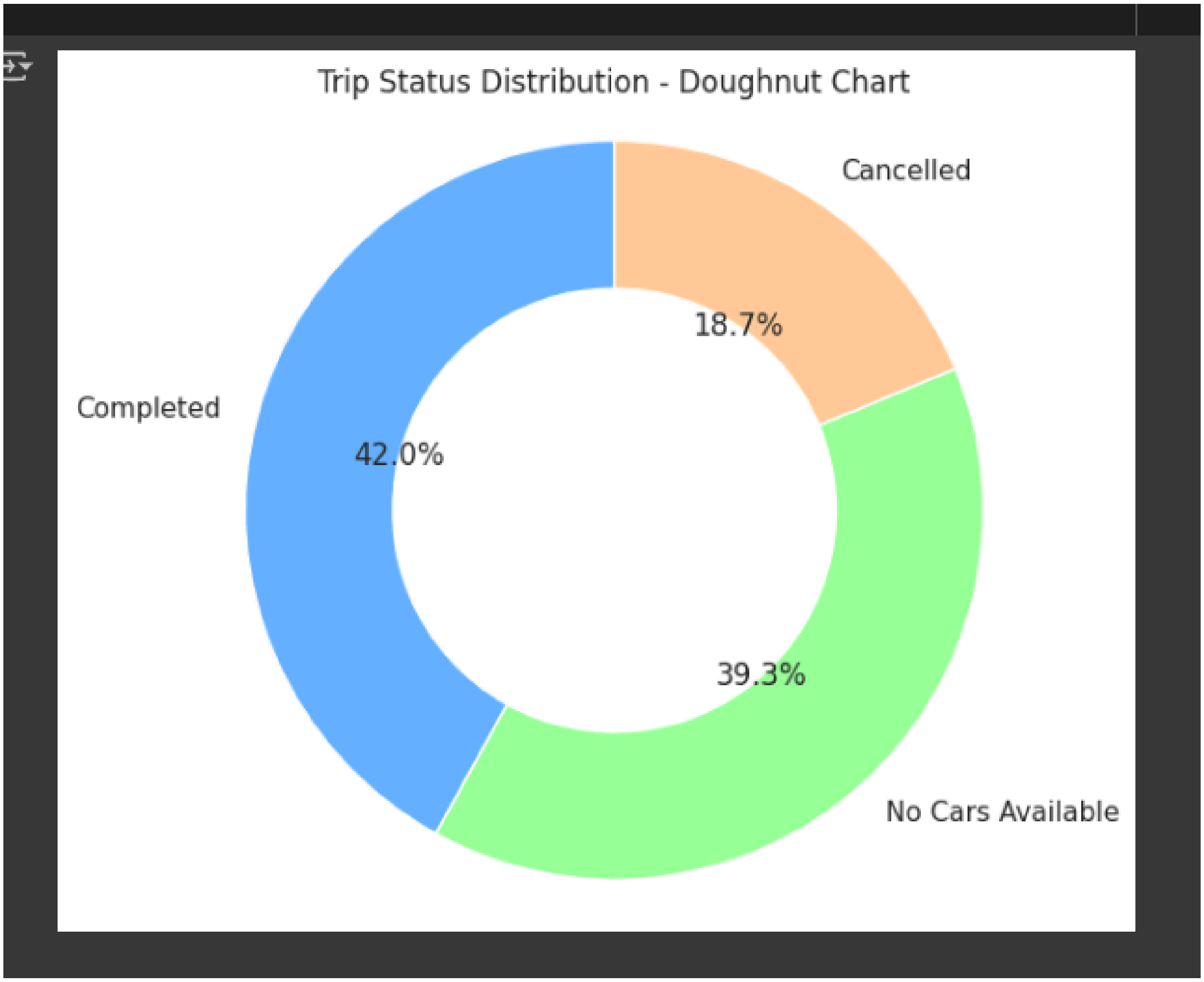
```
df['Request time'] = pd.to_datetime(df['Request time'])
```

Line Chart

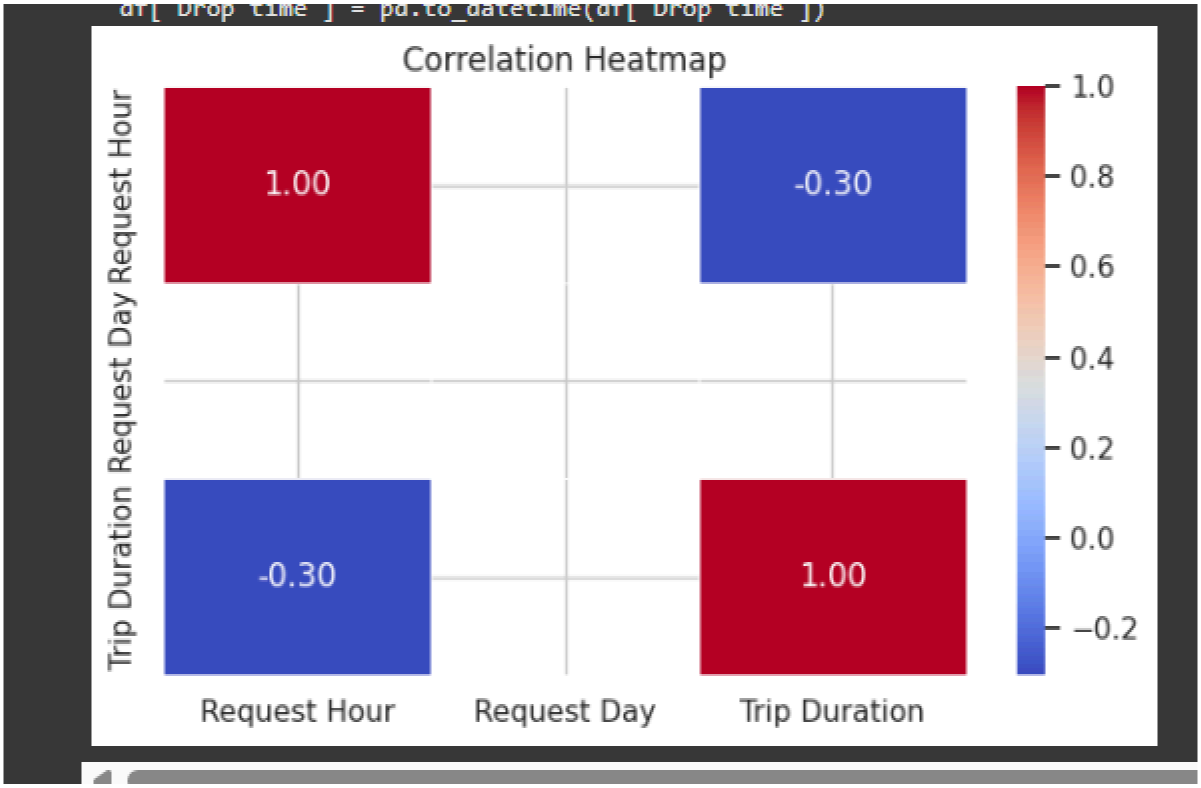


```
df['Request time'] = pd.to_datetime(df['Request time'])
```

Bubble Chart



Doughnut Chart



```
df['Drop time'] = pd.to_datetime(df['Drop time'])
```

Correlation Heatmap

Insights gained from EDA:

1. Overview of Data

- There are 6,745 Uber ride requests in the dataset.
- Pickup location, status, request/drop timestamps, and trip duration are important columns.
- Drop data is missing for about 58% of trips, primarily because of cancellations or "No Cars Available" status.

2. Data Purification

- To make coding easier, columns were renamed using snake_case.
- Date and time were combined to create Request_datetime and Drop_datetime, respectively.

- Text values in columns like Status and Pickup_point have been cleaned and standardized.

- Trip_Duration was changed to a numeric value, and any missing or incorrect values were handled with force.

3. Business Knowledge

- Imbalance between supply and demand at particular times and places.

- There is a major driver shortage for nighttime airport trips.

- Cancellations of early morning city requests may occur because of operational problems or driver unavailability.

4. Suggestions

- Boost the number of drivers available during periods of high demand.

- Provide drivers with greater incentives during busy times and pickup locations.

- To more accurately forecast demand and deploy drivers, use predictive analytics.

- Boost client communication when availability is low.

- Reduce driver cancellations by putting in place a system of rewards and penalties.

5. Hazards Recognized

- Revenue loss results from a high proportion of unsuccessful visits.

- Recurring failures have a detrimental effect on trust and user experience.

- Both drivers and passengers are less satisfied when trips are fulfilled inefficiently.