Uber Exploratory Data Anlysis Project

Project Name - Uber EDA Project

Project Type - EDA

Contribution - Individual

Name – Gautam Bollu

1. Project Summary:-

This project focuses on analyzing Uber ride request data to uncover operational bottlenecks and improve service efficiency. The dataset contains detailed information on pickup points, driver availability, trip statuses, and timestamps of ride requests and completions. Data was first cleaned in Excel to handle missing values and inconsistent timestamps. Using SQL, we extracted key insights such as hourly request volumes, trip status trends, driver utilization, and fulfillment rates. In Python, we conducted deeper exploratory analysis, calculated trip durations, detected outliers, and visualized patterns using matplotlib and seaborn. The goal was to identify peak demand periods, cancellation causes, and areas with frequent unavailability, supporting data-driven decisions for better fleet and service management.

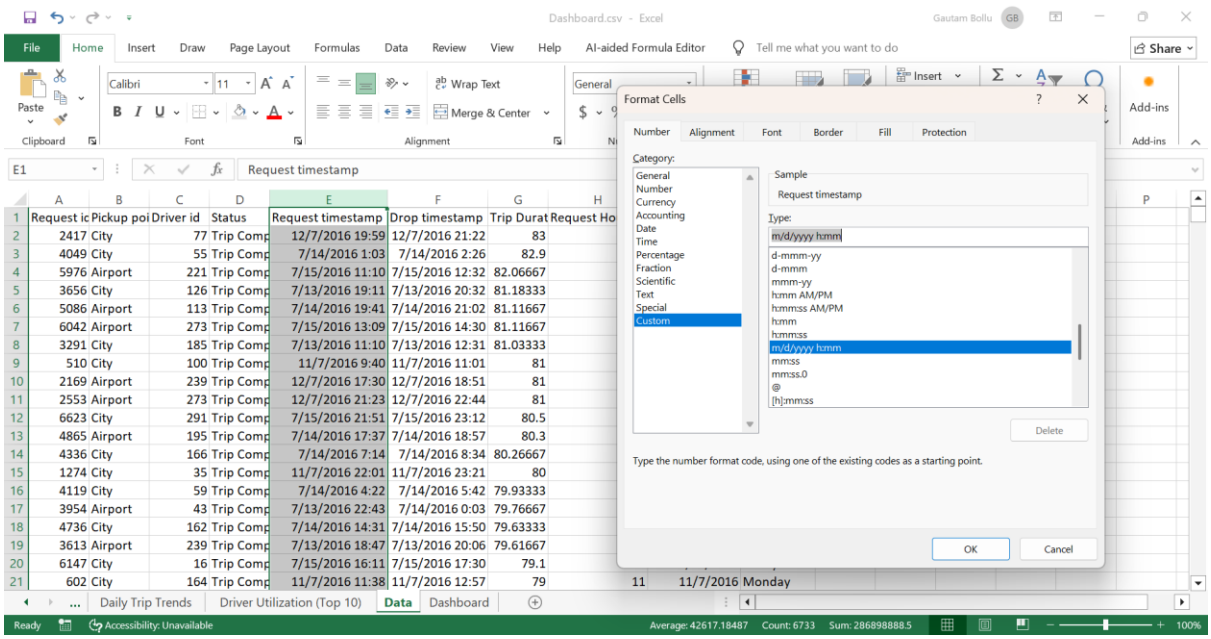
2. Tools & Technologies Used

- Excel: Initial data cleaning, splitting datetime, and dashboarding
- PostgreSQL: SQL queries for summary statistics and trend analysis
- Python (Pandas, Seaborn, Matplotlib): Exploratory Data Analysis (EDA), feature engineering, and outlier detection
- Excel :- Dashboard

Excel –

Data Cleaning

- Removed blank rows and handled NA values
- Converted text timestamps to proper datetime format
- Split timestamps into date, time, hour, and day_name
- Checking the duplicates
- Adjusted timestamp in proper format like m/d/yyyy h:mm



PostgreSQL

Why PostgreSQL Was Used

PostgreSQL was chosen as the database engine for this project to manage, query, and analyze structured Uber ride data efficiently. It is a powerful, open-source relational database system that supports complex queries, filtering, aggregation, and date-time operations—making it ideal for analytical workloads. Its ability to handle large volumes of data and extract meaningful insights through SQL makes it essential for structured analysis in any data-driven project.

How PostgreSQL Was Used in the Project

In this project, the cleaned Uber ride data was first imported into PostgreSQL. A relational table named uber was created to store the data with appropriate data types such as TIMESTAMP for request and drop times. Once the data was imported, SQL queries were used to summarize and slice the dataset to answer business questions such as when and why cancellations occur, what hours see peak demand, and how often trips are successfully fulfilled.

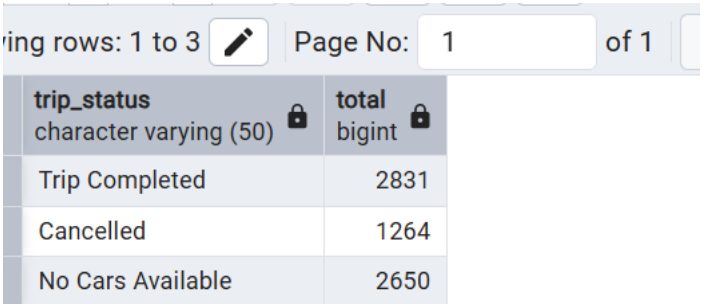
Using PostgreSQL, we were able to group data by pickup points, hours, days, and trip statuses, which helped identify trends and operational gaps. We leveraged powerful functions such as EXTRACT, FILTER, TO_CHAR, and aggregation functions like COUNT and ROUND to generate analytical summaries. This SQL-driven analysis formed the foundation for subsequent Python-based EDA and dashboarding efforts.

Key SQL Insights

1. Trip Status Distribution

SELECT trip_status, COUNT(*) AS total FROM uber GROUP BY trip_status;

- Use: Understands overall service efficiency by showing how many trips were completed, cancelled, or faced no car availability.



ing rows: 1 to 3 | Page No: 1 of 1

trip_status character varying (50)	total bigint
Trip Completed	2831
Cancelled	1264
No Cars Available	2650

2. Hourly Request Volume

SELECT EXTRACT(HOUR FROM request_timestamp) AS hour, COUNT(*) AS total_requests FROM uber GROUP BY hour ORDER BY total_requests DESC;

- Use: Identifies peak hours of demand, useful for resource planning and driver allocation.

	hour numeric	total_requests bigint
1	18	510
2	20	492
3	19	473
4	21	449
5	5	445
6	9	431
7	8	423
8	17	418
9	7	406
10	6	398

3. Daily Trip Trends

SELECT DATE(request_timestamp) AS trip_date, COUNT(*) AS total_requests FROM uber GROUP BY trip_date ORDER BY trip_date;

- Use: Reveals daily usage trends, helping in time-based performance analysis.

	trip_date date	total_requests bigint
1	2016-07-11	1367
2	2016-07-12	1307
3	2016-07-13	1337
4	2016-07-14	1353
5	2016-07-15	1381

4. Pickup Point vs Status Comparison

   SELECT pickup_point, trip_status, COUNT(*) FROM uber GROUP BY pickup_point, trip_status ORDER BY pickup_point;

   - Use*:* Compares customer experience from City vs Airport; shows which pickup points have more issues.

	pickup_point character varying (50)	trip_status character varying (50)	count bigint
1	Airport	No Cars Available	1713
2	Airport	Trip Completed	1327
3	Airport	Cancelled	198
4	City	Trip Completed	1504
5	City	Cancelled	1066
6	City	No Cars Available	937

5. Trip Fulfillment Rate

   SELECT COUNT(*) FILTER (WHERE trip_status = 'Trip Completed') * 100.0 / COUNT(*) AS fulfillment_rate FROM uber;

   - Use*:* Measures success rate of Uber's service—critical for business performance KPIs.

	fulfillment_rate numeric
1	41.9718309859154930

6. No Car Availability by Hour

   SELECT EXTRACT(HOUR FROM request_timestamp) AS hour, COUNT(*) AS no_car_count FROM uber WHERE trip_status = 'No Cars Available' GROUP BY hour ORDER BY hour DESC;

   - Use*:* Identifies which hours experience the most car shortages—helps in fleet management.

	hour numeric	no_car_count bigint
1	0	56
2	2	57
3	9	83
4	3	56
5	11	41
6	4	74
7	10	65
8	12	44

7. Day-of-Week Demand Pattern

   SELECT DATE(request_timestamp) AS request_date, TO_CHAR(request_timestamp, 'Day') AS day_name, COUNT(*) AS total_requests FROM uber GROUP BY request_date, day_name ORDER BY request_date;

   - Use*:* Shows user behavior by day; helps in analyzing demand spikes on weekdays or weekends.

	request_date date	day_name text	total_requests bigint
1	2016-07-11	Monday	1367
2	2016-07-12	Tuesday	1307
3	2016-07-13	Wednesday	1337
4	2016-07-14	Thursday	1353
5	2016-07-15	Friday	1381

Python Analysis

Why Python Is Used in the Project

Python is used in data analytics projects because of its simplicity, powerful libraries. It offers a flexible environment for working with data through libraries like Pandas for data manipulation, Matplotlib and Seaborn for visualization, and NumPy for numerical analysis. These tools make it easier to clean, analyze, and visualize large datasets efficiently and interactively.

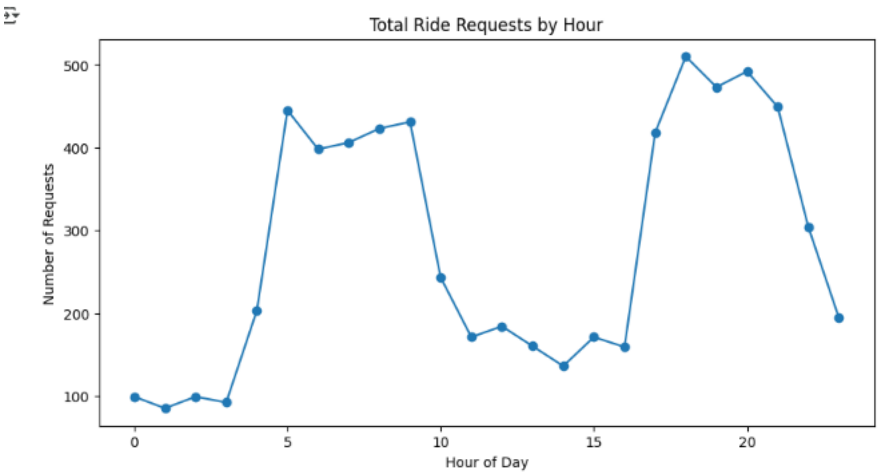
How Python Was Used in the Uber Project

In this Uber driver analysis project, Python was used to load and clean the dataset, especially for fixing errors in timestamps and calculating accurate trip durations. It handled missing values, detected and removed outliers, and added new time-based features like hour, weekday, and trip duration. Once the data was cleaned, Python was used to create 15 in-depth visualizations that revealed patterns in trip demand, supply gaps, cancellations, and driver behavior. This helped in drawing actionable insights from raw data using only a few lines of code with clear visual storytelling.

Insights from Python Visualizations

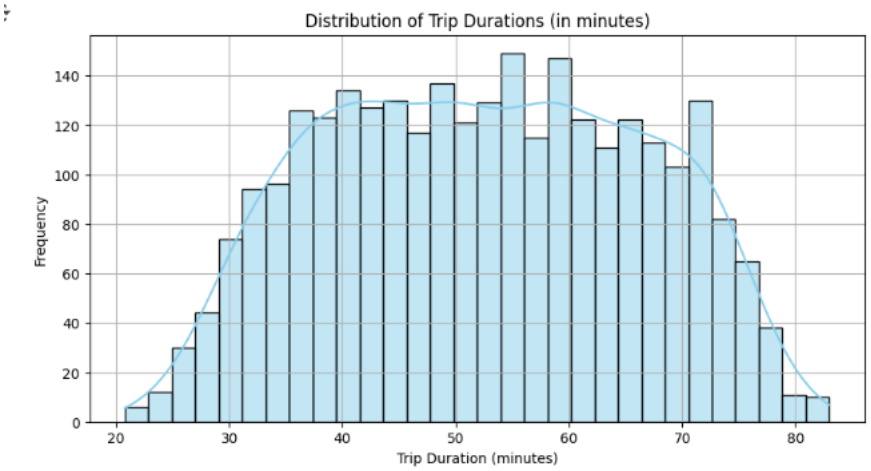
1. Hourly Completed Trips

This shows how many trips were successfully completed at each hour of the day. It helps identify when the supply of drivers is sufficient, and peak successful service hours. Planners can use this to schedule shifts and optimize availability.



2. Trip Duration Distribution

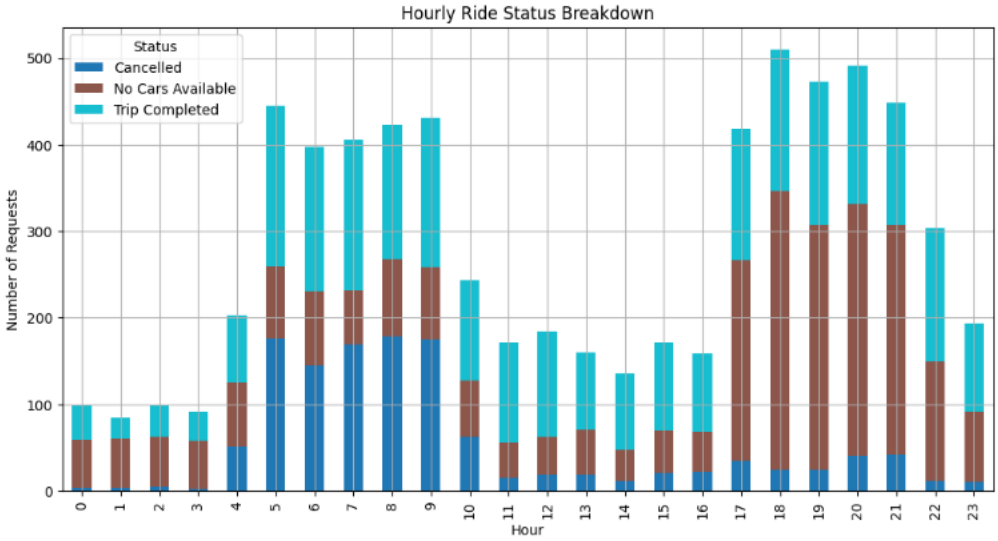
Insight:
Shows how long trips usually last. This helps understand what a "normal" trip looks like and detect outliers. It also provides insight into the average efficiency of rides and helps in pricing or route optimization.



---

3. Request Status Share by Hour

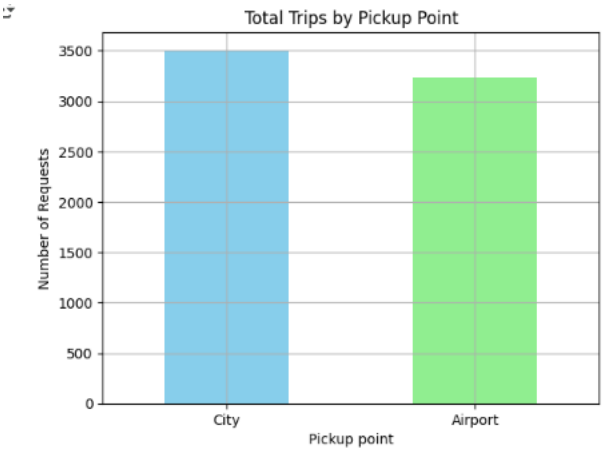
Insight:
This shows the distribution of Completed, Cancelled, and No Cars Available statuses per hour. It gives a full picture of hourly demand vs supply and service reliability, revealing when most problems occur.



---

4. Trips by Pickup Point

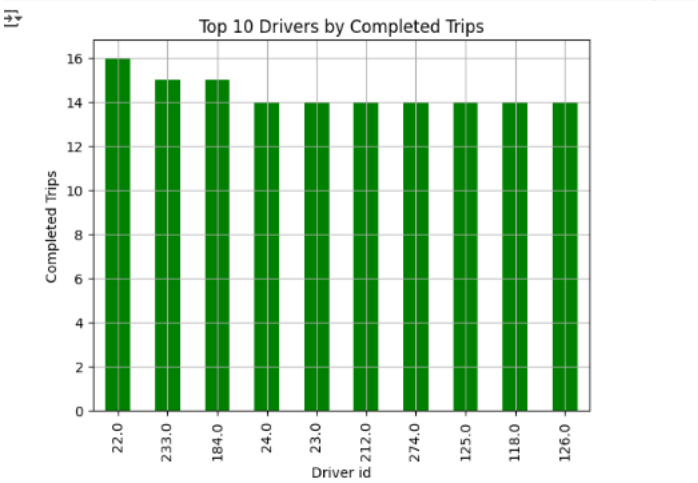
Type: Bar Chart
Insight:
Compares total trip requests from City vs Airport. Helps identify geographic demand centers, aiding in fleet distribution and driver placement strategies.



5. Top 10 Drivers by Completed Trips

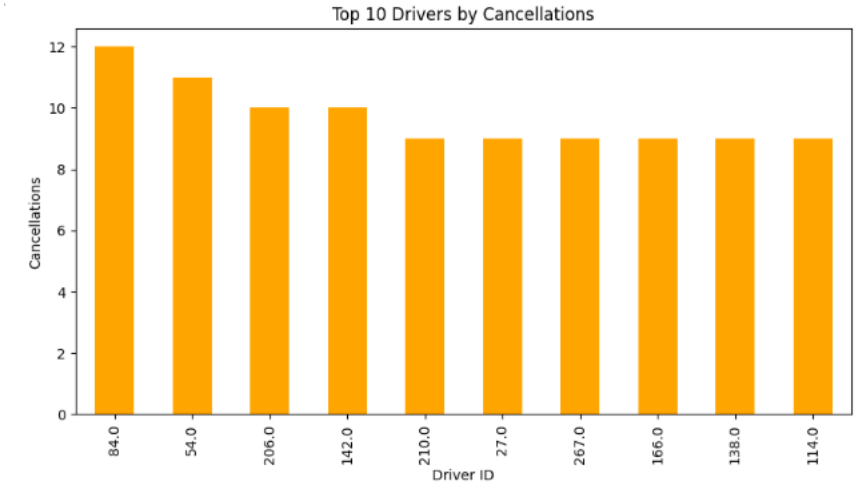
Type: Bar Chart
Insight:
Ranks the most active or efficient drivers based on completed rides. Useful for rewards, incentives,

and recognition, or identifying high-performing drivers.



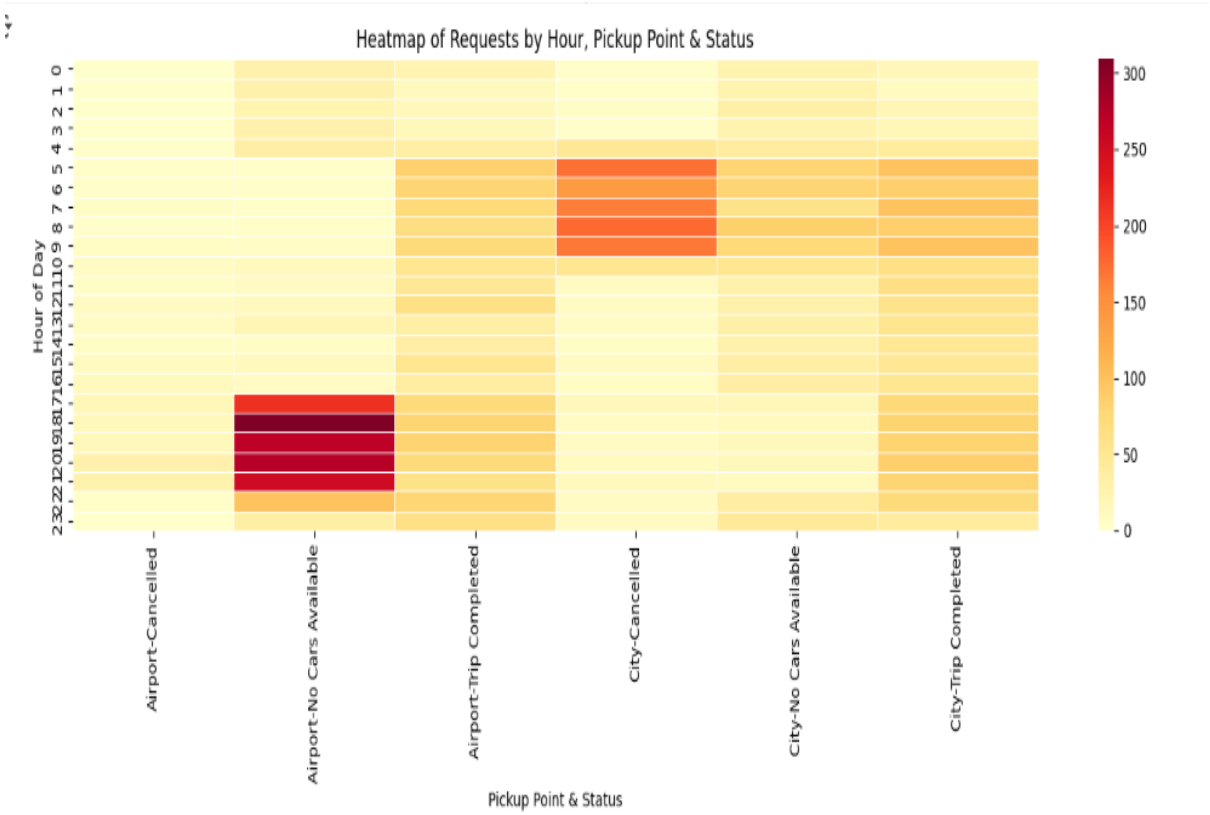
6. Top 10 Drivers by Cancellations

Type: Bar Chart
Insight:
Shows which drivers cancel the most trips. Helps identify drivers who may need retraining or monitoring due to potentially harming customer experience.



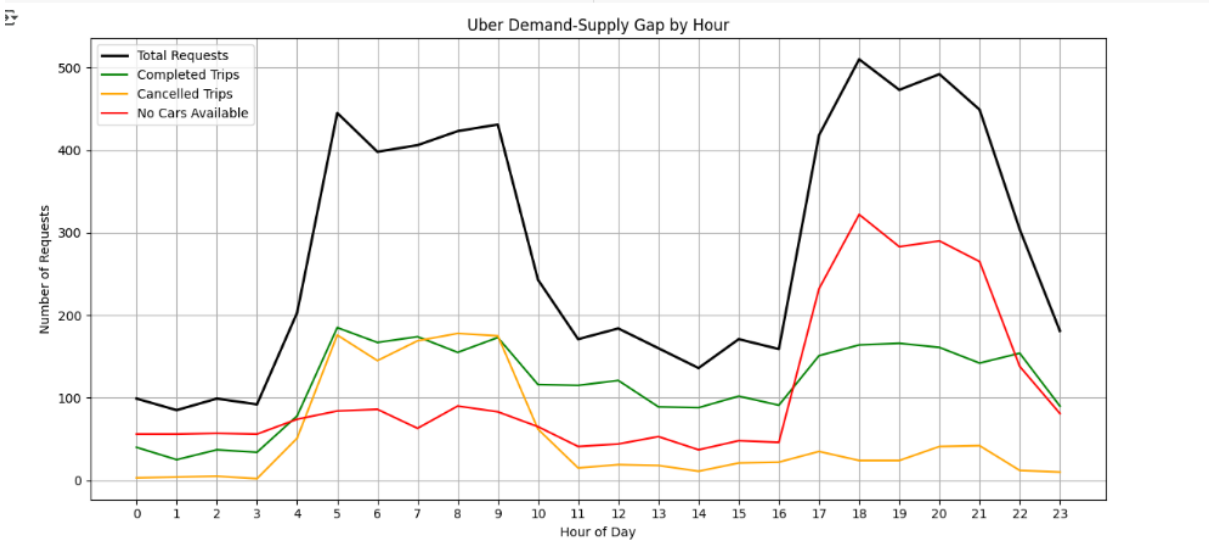
7. Heatmap: Hour vs Pickup Point vs Status

Type: Heatmap
Insight:
This powerful chart shows the intersection of time of day, location (City/Airport), and ride status (Completed, Cancelled, No Cars). It's great for spotting problem periods or service gaps visually across multiple dimensions.



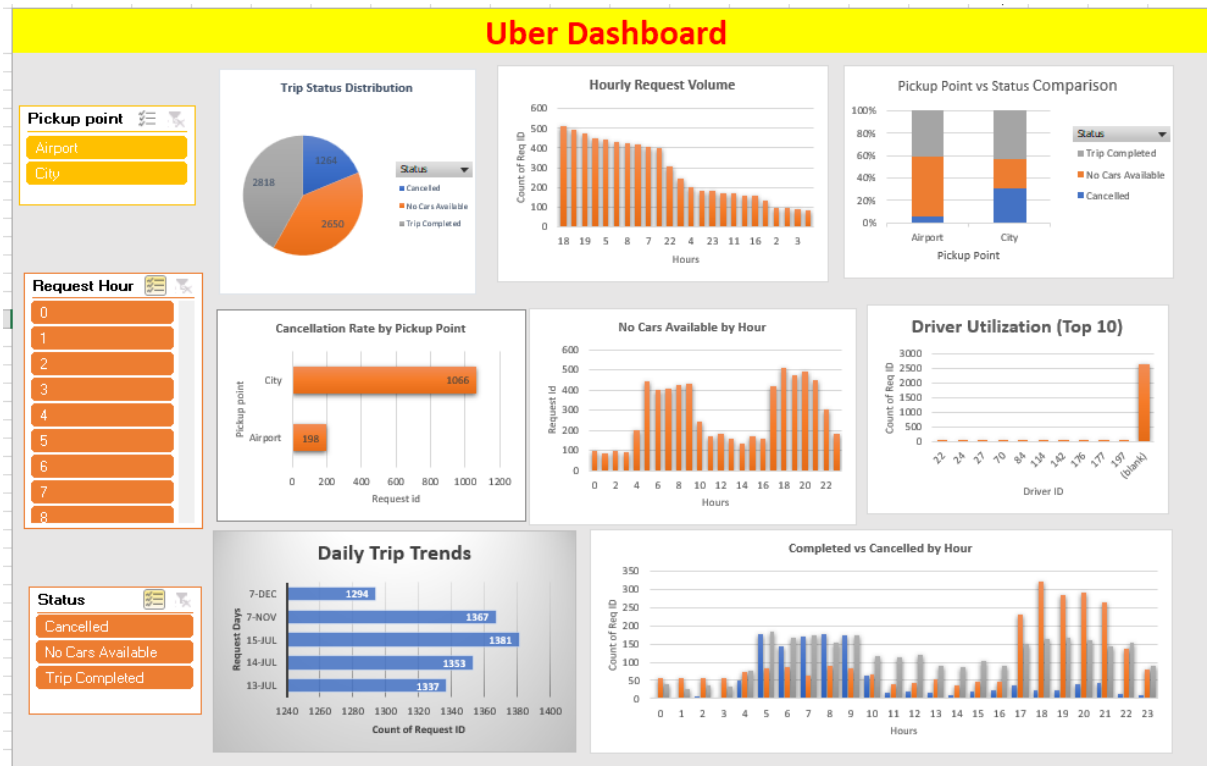
8) Uber Demand-Supply Gap by Hour

This visualization helps uncover imbalances between demand and supply by hour. It allows stakeholders to visually compare ride requests vs fulfilled rides, while also tracking service issues like cancellations and lack of available cars.



Python Colab Link :-
https://colab.research.google.com/drive/11mAJRdDtq0FHlTL2zPgBxPG2SqDo2tbD?usp=sharing

Excel



Explanation:-

Slicers (Left Panel)

- **Pickup Point, Request Hour, Status**: These slicers allow dynamic filtering of all charts based on user selections. For instance, you can instantly view only City requests or only Cancelled trips during certain hours.

1. Trip Status Distribution (Pie Chart)

- Shows the proportion of trips that were:
    - **Trip Completed** (Blue)
    - **Cancelled** (Gray)
    - **No Cars Available** (Orange)
- Insight: A significant portion of trips are either cancelled or not serviced, revealing areas for operational improvement.

2. Hourly Request Volume (Bar Chart)

- Displays total trip requests for each hour.
- Peak hours are around **17:00 to 20:00**, indicating high demand in the evening.
- Suggests a need to align driver availability with these busy times.

3. Pickup Point vs Status Comparison (100% Stacked Bar)

- Compares the trip status distribution between **Airport** and **City**.
- The City has a more balanced mix, while the Airport sees more unfulfilled requests.
- Insight: Airport demand is likely underserved.

4. Cancellation Rate by Pickup Point (Horizontal Bar)

- Clearly shows that the **City** has a higher number of cancellations.
- Could be due to traffic, wait time, or app-related issues in urban areas.
- Calls for deeper investigation into city-specific factors.

5. No Cars Available by Hour (Bar Chart)

- Highlights the number of unfulfilled requests due to **no cars being available** across hours.
- A second peak appears in the evening.
- Suggests the need for better driver deployment or incentives during high-demand hours.

6. Driver Utilization (Top 10 Drivers – Bar Chart)

- Shows the top 10 drivers based on completed trips.
- A single driver (ID hidden) dominates the list, suggesting possible overuse or high efficiency.
- Helps in identifying top performers or dependency on few drivers.

7. Daily Trip Trends (Bar Chart)

- Displays total trips by day.
- Highlights variation in demand across specific dates.
- Useful for analyzing patterns over time or assessing the impact of promotions/holidays.

8. Completed vs Cancelled by Hour (Clustered Column Chart)

- Compares the number of **Completed** vs **Cancelled** trips at every hour.
- Cancellations peak around **9–11 AM and 6–9 PM**, showing potential stress on the system.
- Helps target operational fixes during critical periods.

Conclusion

This Uber Data Analysis project successfully identified key operational challenges faced by the ride-hailing service using real-world data. By combining Excel for cleaning, PostgreSQL for structured querying, and Python for exploratory data analysis, we uncovered critical insights into trip statuses, hourly demand patterns, driver availability, and trip fulfillment rates. The analysis revealed that nearly 58% of ride requests either got cancelled or couldn't be served due to a lack of available cars—highlighting a significant gap in supply during peak hours and specific routes, especially from the Airport. Through careful data preprocessing, handling of missing values, correction of timestamp inconsistencies, and outlier detection, we ensured high data integrity. The insights gained through SQL queries and Python visualizations are instrumental for recommending improvements in fleet management, driver engagement strategies, and resource allocation. Overall, this project

demonstrates the power of end-to-end data analytics in driving informed business decisions for better customer service and operational efficiency.

Suggestions

1. **Increase driver availability** during peak demand hours, especially for Airport pickups and drop-offs.
2. **Implement dynamic driver incentives** to reduce cancellations and encourage trip acceptance.
3. **Use hourly and weekday trend analysis** to optimize driver distribution across time slots.
4. **Set up predictive alerts** for potential "No Cars Available" situations to proactively deploy drivers.