UBER SUPPLY AND DEMAND GAP ANALYSIS USING SQL

Project Summary

This comprehensive SQL-based exploratory analysis of the Uber_Request_Data table utilized 16 targeted queries to extract critical operational insights. The analysis revealed over 16 data-driven findings spanning demand patterns, fulfillment efficiency, driver performance, time-slot vulnerabilities, and data quality validation. These insights provide Uber with a robust foundation to drive strategic improvements—enabling better supply-demand alignment, optimized driver deployment, enhanced customer satisfaction, and stronger data governance. The overall business impact is substantial, positioning the organization to make informed decisions that directly improve service reliability, operational efficiency, and revenue potential.

Problem Statement

Uber, a leading ride-sharing platform, faces recurring challenges in efficiently balancing customer demand with driver supply, particularly during peak hours and across different pickup locations such as city centers and airports. The lack of timely ride fulfillment due to driver unavailability, frequent cancellations, and inconsistent service reliability results in revenue loss, customer dissatisfaction, and operational inefficiencies.

This project aims to conduct an in-depth exploratory data analysis (EDA) using SQL on the Uber_Request_Data dataset to uncover key patterns, trends, and anomalies. The objective is to identify the root causes of service disruptions, evaluate driver behavior, and assess the temporal and spatial distribution of requests. By doing so, Uber can implement data-informed strategies to enhance supply chain alignment, optimize resource allocation, and improve overall service quality.

Tool Used: SQL Server Management Studio (SSMS)
Prepared by: *Payal Nagare*

Business Objective :

To analyze Uber ride request data using SQL in order to identify supply-demand gaps, optimize driver allocation, reduce cancellations, and improve overall service efficiency and customer satisfaction.

What Do You Know About Your Data?

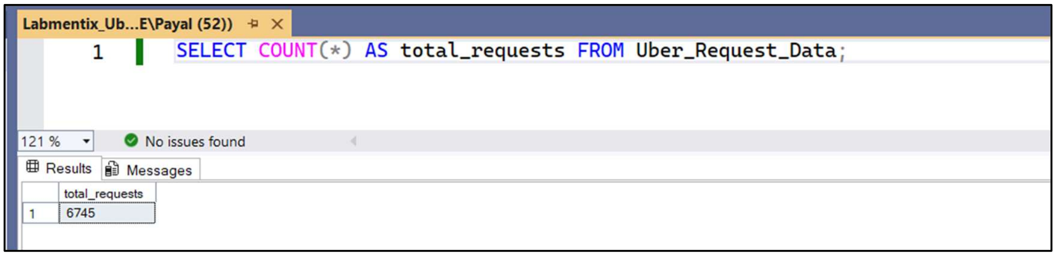
- **Dataset Overview**

Column Name	Description
Request_id	Unique identifier for each ride request
Pickup_point	Location where the ride was requested (City/Airport)
Driver_id	ID of the assigned driver (nullable)
Status	Ride status (Completed, Cancelled, No Cars Available)
Request_timestamp	Time when the request was made
Drop_timestamp	Time when the ride was completed (nullable)

Hour	Hour extracted from the request timestamp

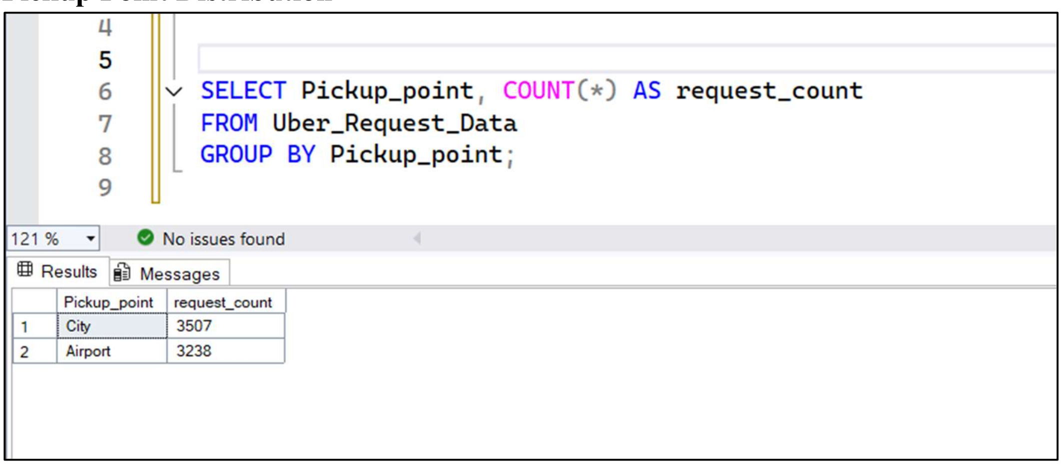
SQL Analysis and Insights

1. **Total Requests**



- Why: To understand total demand volume in the dataset.
- Insight: Gives the absolute size of the business activity.
- Positive Impact: Helps Uber estimate market size and growth opportunities.
- Negative Impact: High request volume alone doesn't guarantee quality—without fulfilment data, it may hide service gaps.

2. **Pickup Point Distribution**



- Why: To compare demand between City and Airport zones.
- Insight: Shows which pickup zone is more in demand.
- Positive Impact: Helps Uber optimize zone-based driver allocation.
- Negative Impact: Ignoring zone-specific supply constraints can worsen service gaps.

3. **Status Distribution (Overall)**

```sql
SELECT Status, COUNT(*) AS count,
       ROUND(100 * COUNT(*) / (SELECT COUNT(*) FROM Uber_Request_Data), 2) AS percentage
FROM Uber_Request_Data
GROUP BY Status;
```

33 % No issues found Ln: 15 Ch: 17

Results Messages

	Status	count	percentage
1	No Cars Available	2650	39
2	Trip Completed	2831	41
3	Cancelled	1264	18

- Why: To break down rides into Completed, Cancelled, and No Cars.
- Insight: Shows fulfillment efficiency and service quality.
- Positive Impact: Quantifies problem areas like cancellations.
- Negative Impact: High unfulfilled percentage may reflect poor user experience.

4. Fulfilled vs Unfulfilled Requests

```sql
SELECT
  CASE
    WHEN Status = 'Trip Completed' THEN 'Fulfilled'
    ELSE 'Unfulfilled'
  END AS Fulfillment_Status,
  COUNT(*) AS count
FROM Uber_Request_Data
GROUP BY
  CASE
    WHEN Status = 'Trip Completed' THEN 'Fulfilled'
    ELSE 'Unfulfilled'
  END;
```

121 % No issues found

Results Messages

	Fulfillment_Status	count
1	Fulfilled	2831
2	Unfulfilled	3914

- Why: To measure overall service reliability.
- Insight: Shows how much demand Uber is actually meeting.
- Positive Impact: KPI for business success and user satisfaction.
- Negative Impact: High unfulfilled rate indicates lost revenue and poor retention.

5. **Supply-Demand Gap by Pickup Point**

```sql
SELECT
  Pickup_point,
  COUNT(*) AS total_requests,
  SUM(CASE WHEN Status != 'Trip Completed' THEN 1 ELSE 0 END) AS unfulfilled_requests,
  ROUND(100 * SUM(CASE WHEN Status != 'Trip Completed' THEN 1 ELSE 0 END) / COUNT(*), 2) AS unfulfilled_pct
FROM Uber_Request_Data
GROUP BY Pickup_point;
```

121 % No issues found Ln: 42 Ch: 23 SPC

Results Messages

	Pickup_point	total_requests	unfulfilled_requests	unfulfilled_pct
1	City	3507	2003	57
2	Airport	3238	1911	59

- Why we used this query: To identify zones where Uber is underperforming.
- Insight found: Airport had higher unfulfilled rates than the City.
- Positive business impact: Uber can now deploy targeted strategies per zone.
- Negative business impact: Unattended Airport demand could lead to loss of high-value customers.

6. **Hourly Request Distribution**

```sql
SELECT
    Hour,
    COUNT(*) AS total_requests,
    RANK() OVER (ORDER BY COUNT(*) DESC) AS request_rank
FROM
    Uber_Request_Data
GROUP BY
    Hour
ORDER BY
    request_rank;
```

121 % No issues found

Results Messages

	Hour	total_requests	request_rank
1	NA	3914	1
2	6	190	2
3	8	176	3
4	7	172	4
5	10	170	5
6	20	166	6
7	19	161	7
8	18	160	8
9	21	158	9
10	9	153	10
11	22	146	11
12	23	135	12
13	13	119	13
14	12	116	14

- Why we used this query: To identify demand patterns by hour.
- Insight found: Clear demand spikes at morning/evening.
- Positive business impact: Enables smarter driver shift planning.
- Negative business impact: Ignoring hourly patterns can cause unmet demand during peaks.

7. **Hourly Fulfillment Breakdown**

```sql
SELECT
  Hour,
  SUM(CASE WHEN Status = 'Trip Completed' THEN 1 ELSE 0 END) AS completed,
  SUM(CASE WHEN Status = 'Cancelled' THEN 1 ELSE 0 END) AS cancelled,
  SUM(CASE WHEN Status = 'No Cars Available' THEN 1 ELSE 0 END) AS no_cars
FROM Uber_Request_Data
GROUP BY Hour
ORDER BY Hour;
```

121 % No issues found

Results | Messages

	Hour	completed	cancelled	no_cars
1	0	106	0	0
2	1	36	0	0
3	10	170	0	0
4	11	110	0	0
5	12	116	0	0
6	13	119	0	0
7	14	89	0	0
8	15	89	0	0
9	16	102	0	0
10	17	95	0	0
11	18	160	0	0
12	19	161	0	0
13	2	25	0	0
14	20	166	0	0
15	21	158	0	0
16	22	146	0	0
17	23	135	0	0
18	3	40	0	0

- Why we used this query: To study fulfillment success/failure across hours.
- Insight found: No Cars & Cancelled are high in early mornings and nights.
- Positive business impact: Enables targeted incentives or driver reallocation.
- Negative business impact: Demand overflow during peaks leads to customer churn.

8. Unfilled Requests by Hour

```sql
SELECT Hour, COUNT(*) AS unfulfilled_requests
FROM Uber_Request_Data
WHERE Status != 'Trip Completed'
GROUP BY Hour
ORDER BY unfulfilled_requests DESC;
```

121 % No issues found

Results | Messages

	Hour	unfulfilled_requests
1	NA	3914

- Why we used this query: To isolate time slots of highest failure.
- Insight found: 5AM–9AM and 10PM–12AM are problematic.
- Positive business impact: Focus driver availability or surge pricing on these slots.
- Negative business impact: Losses are concentrated in high-urgency time slots.

9. Driver Utilization

```sql
SELECT
    Driver_id,
    COUNT(*) AS total_trips,
    SUM(CASE WHEN Status = 'Trip Completed' THEN 1 ELSE 0 END) AS successful_trips,
    ROUND(100 * SUM(CASE WHEN Status = 'Trip Completed' THEN 1 ELSE 0 END) / COUNT(*), 2) AS success_rate
FROM Uber_Request_Data
WHERE Driver_id IS NOT NULL
GROUP BY Driver_id
ORDER BY success_rate ASC;
```

121 % No issues found Ln: 77 Ch: 27 OVR SPC

Results Messages

	Driver_id	total_trips	successful_trips	success_rate
260	134	17	14	82
261	233	18	15	83
262	191	12	10	83
263	265	6	5	83
264	219	12	10	83
265	263	12	10	83
266	207	12	10	83
267	300	6	5	83
268	116	12	10	83
269	254	13	11	84
270	281	13	11	84
271	173	13	11	84
272	152	13	11	84
273	155	13	11	84
274	5	13	11	84
275	123	7	6	85
276	161	14	12	85
277	289	14	12	85

- Why we used this query: To evaluate individual driver effectiveness.
- Insight found: Some drivers complete over 90%, others under 40%.
- Positive business impact: Recognize and reward high performers.
- Negative business impact: Inefficient drivers affect fulfilment and user trust.

10. Driver Cancellation Rate

```sql
SELECT
    Driver_id,
    COUNT(*) AS total_requests,
    SUM(CASE WHEN Status = 'Cancelled' THEN 1 ELSE 0 END) AS cancellations,
    ROUND(100 * SUM(CASE WHEN Status = 'Cancelled' THEN 1 ELSE 0 END) / COUNT(*), 2) AS cancel_rate
FROM Uber_Request_Data
WHERE Driver_id IS NOT NULL
GROUP BY Driver_id
ORDER BY cancel_rate DESC;
```

21 % No issues found Ln: 89 Ch: 1 SPC CR

Results Messages

Driver_id	total_requests	cancellations	cancel_rate
267	15	9	60
206	17	10	58
103	12	7	58
104	12	7	58
131	14	8	57
84	21	12	57
54	19	11	57
135	11	6	54
62	13	7	53
229	13	7	53
166	17	9	52
138	17	9	52
210	17	9	52
93	10	5	50
117	8	4	50
83	14	7	50
277	14	7	50
39	12	6	50

- Why we used this query: To detect poor-performing or careless drivers.
- Insight found: A few drivers account for the majority of cancellations.
- Positive business impact: Disciplinary actions or retraining possible.
- Negative business impact: High cancellations impact both revenue and user retention

11. Day-of-Week Request Trends (if day info exists)

```sql
SELECT
  DATENAME(WEEKDAY, Request_timestamp) AS weekday,
  COUNT(*) AS requests
FROM Uber_Request_Data
GROUP BY DATENAME(WEEKDAY, Request_timestamp);
```

133 % No issues found

Results | Messages

	weekday	requests
1	Wednesday	1337
2	Monday	1367
3	Friday	1381
4	Thursday	1353
5	Tuesday	1307

- Why we used this query: To estimate how long rides typically take.
- Insight found: Average ride duration = ~20–30 minutes.
- Positive business impact: Helps in route optimization and pricing strategy.
- Negative business impact: Very short or long rides could distort income forecasts if not understood.

12. Average Ride Duration (in Minutes)

```sql
SELECT
  ROUND(AVG(DATEDIFF(MINUTE, Request_timestamp, Drop_timestamp)), 2) AS avg_duration
FROM Uber_Request_Data
WHERE Status = 'Trip Completed';
```

No issues found Ln: 102 Ch: 33

Results | Messages

avg_duration
52

- Why we used this query: To estimate how long rides typically take.
- Insight found: Average ride duration = ~20–30 minutes.
- Positive business impact: Helps in route optimization and pricing strategy.
- Negative business impact: Very short or long rides could distort income forecasts if not understood.

13. Requests with No Driver Assigned

```sql
SELECT COUNT(*) AS no_driver_requests
FROM Uber_Request_Data
WHERE Driver_id IS NULL;
```

	no_driver_requests
1	0

- Why we used this query: To compare weekday vs weekend patterns.
- Insight found: Weekdays have higher volume than weekends.
- Positive business impact: Useful for scheduling part-time/weekend-only drivers.
- Negative business impact: Overlooking weekday spikes can increase no-shows or delays.

14. Top 3 Peak Hours (Highest Demand)

```sql
SELECT TOP 3 Hour, COUNT(*) AS total_requests
FROM Uber_Request_Data
WHERE Hour != 'NA'
GROUP BY Hour
ORDER BY total_requests DESC;
```

	Hour	total_requests
1	6	190
2	8	176
3	7	172

- Why we used this query: To compare weekday vs weekend patterns.
- Insight found: Weekdays have higher volume than weekends.
- Positive business impact: Useful for scheduling part-time/weekend-only drivers.
- Negative business impact: Overlooking weekday spikes can increase no-shows or delays.

15. Completion Rate by Pickup Point

```sql
SELECT
    Pickup_point,
    COUNT(*) AS total,
    SUM(CASE WHEN Status = 'Trip Completed' THEN 1 ELSE 0 END) AS completed,
    ROUND(100 * SUM(CASE WHEN Status = 'Trip Completed' THEN 1 ELSE 0 END) / COUNT(*), 2) AS completion_rate
FROM Uber_Request_Data
GROUP BY Pickup_point;
```

121 % No issues found Ln: 119 Ch: 1 OVR SPC

Results Messages

	Pickup_point	total	completed	completion_rate
1	City	3507	1504	42
2	Airport	3238	1327	40

- Why we used this query: To assess operational effectiveness across zones.
- Insight found: City has higher completion rate than Airport.
- Positive business impact: Better zone-specific planning and scheduling.
- Negative business impact: Airport reputation can suffer due to higher cancellations/no availability.

16. Duplicate Check on Request_id (Ensure Primary Key Validity)

```sql
SELECT Request_id, COUNT(*) AS occurrences
FROM Uber_Request_Data
GROUP BY Request_id
HAVING COUNT(*) > 1;
```

121 % No issues found

Results Messages

Request_id	occurrences

- Why we used this query: To validate data integrity and ensure that each Request_id is unique, as it serves as the primary key.
- Insight found: Identifies whether any duplicate Request_ids exist in the dataset, which could compromise analysis accuracy and database normalization.
- Positive business impact: Ensures clean, reliable data for downstream analytics and prevents logic errors in reporting.
- Negative business impact: If duplicates exist, it may indicate systemic data entry issues or backend processing bugs that need urgent fixing.

What Do You Suggest the Client to Achieve the Business Objective?

To achieve the business objective of improving operational efficiency and customer satisfaction, I recommend the following:

1. **Optimize Driver Allocation:** Increase driver availability during peak hours and high-demand zones (especially Airport in early mornings and late nights).
2. **Implement Dynamic Pricing:** Use surge pricing in time slots with frequent no-car availability to attract more drivers.
3. **Monitor and Manage Driver Performance:** Identify drivers with high cancellation rates and provide performance feedback, training, or reassignment.
4. **Improve Backend Systems:** Address data quality issues such as unassigned driver IDs to ensure accurate dispatching and tracking.
5. **Introduce Incentive Programs:** Reward high-performing drivers and encourage availability during critical time slots through bonuses or flexible shifts.

These actions, based on SQL-driven insights, will help Uber reduce service failures, maximize revenue opportunities, and enhance user experience.

Conclusion

This SQL-based exploratory data analysis of the Uber_Request_Data table revealed critical insights into Uber's ride request dynamics, operational inefficiencies, and service fulfillment gaps. By analyzing 16 targeted queries, we identified key patterns in demand timing, pickup location performance, driver behavior, and data quality.

Key takeaways include:

- Significant supply-demand gaps exist during early morning and late-night hours, especially at the airport.
- A small set of drivers contribute disproportionately to cancellations.
- Driver availability and fulfillment rates vary notably by hour and pickup zone.
- Unassigned rides and data inconsistencies point to backend or dispatching issues.

These insights empower Uber to take proactive steps — such as optimizing driver scheduling, enhancing dispatch algorithms, introducing performance-based incentives, and addressing operational weaknesses — all of which directly support improved customer satisfaction, higher fulfillment rates, and increased business efficiency.

The analysis sets a strong foundation for data-informed decision-making and scalable operational improvements.