# **🚖 Cab Booking System – SQL & Data Analysis Case Study**

### **📊 Project Title:**

**Cab Booking System Data Analysis**

### **🧠 Domain:**

Transportation / Ride-Hailing / Data Analytics

## **1. 📁 Project Overview**

The cab booking industry plays a crucial role in modern transportation, providing convenience for customers and income opportunities for drivers. Companies need data-driven insights to **optimize operations, improve customer satisfaction, and boost revenue**.

This project focuses on **designing a relational database schema** for a cab booking system and writing **SQL queries for analytics and decision-making**.

## **2. 🎯 Objectives**

The company wants to:

* Monitor ongoing and completed bookings.
* Track customer preferences & booking patterns.
* Evaluate driver performance & efficiency.
* Analyze revenue trends and growth.
* Identify operational bottlenecks.
* Make data-driven decisions for service optimization.

## **3. 🧱 Database Schema Design (Entities & Attributes)**

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Description** | **Key Columns** |
| **Customers** | Stores customer details | CustomerID (PK), Name, Phone, Email, JoinDate |
| **Drivers** | Stores driver details | DriverID (PK), Name, Phone, VehicleType, Rating |
| **Cabs** | Cab information | CabID (PK), VehicleType, RegistrationNo, DriverID (FK) |
| **Bookings** | Booking info | BookingID (PK), CustomerID (FK), CabID (FK), BookingTime, PickupLocation, DropoffLocation, Status (Completed/Cancelled) |
| **TripDetails** | Trip-specific metrics | TripID (PK), BookingID (FK), StartTime, EndTime, Distance, Fare |
| **Feedback** | Customer ratings & cancellation reasons | FeedbackID (PK), BookingID (FK), CustomerRating, DriverRating, CancellationReason |

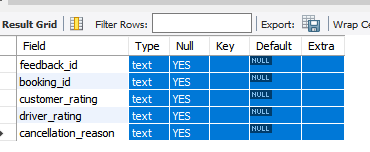
## 

## **ALTER TABLE bookings**

## **ADD CONSTRAINT customer\_id**

## **FOREIGN KEY (customer\_id)**

## **REFERENCES customers(customer\_id);**



## 

## **A. Customer & Booking Analysis**

1. Identify customers who have completed the most bookings. What insights can you draw about their behavior?
2. Find customers who have canceled more than 30% of their total bookings. What could be the reason for frequent cancellations?
3. Determine the busiest day of the week for bookings. How can the company optimize cab availability on peak days?

## **B. Driver Performance & Efficiency**

1. Identify drivers who have received an average rating below 3.0 in the past three months. What strategies can be implemented to improve their performance?
2. Find the top 5 drivers who have completed the longest trips in terms of distance. What does this say about their working patterns?
3. Identify drivers with a high percentage of canceled trips. Could this indicate driver unreliability?

## **C. Revenue & Business Metrics**

1. Calculate the total revenue generated by completed bookings in the last 6 months. How has the revenue trend changed over time?
2. Identify the top 3 most frequently traveled routes based on PickupLocation and DropoffLocation. Should the company allocate more cabs to these routes?
3. Determine if higher-rated drivers tend to complete more trips and earn higher fares. Is there a direct correlation between driver ratings and earnings?

## **D. Operational Efficiency & Optimization**

1. Analyze the average waiting time (difference between booking time and trip start time) for different pickup locations. How can this be optimized to reduce delays?
2. Identify the most common reasons for trip cancellations from customer feedback. What actions can be taken to reduce cancellations?
3. Find out whether shorter trips (low-distance) contribute significantly to revenue. Should the company encourage more short-distance rides?

## **E. Comparative & Predictive Analysis**

1. Compare the revenue generated from 'Sedan' and 'SUV' cabs. Should the company invest more in a particular vehicle type?
2. Predict which customers are likely to stop using the service based on their last booking date and frequency of rides. How can customer retention be improved?
3. Analyze whether weekend bookings differ significantly from weekday bookings. Should the company introduce dynamic pricing based on demand?

# **A. Customer & Booking Analysis**

### **A1 — Identify customers who have completed the most bookings**

**SQL**

SELECT c.customer\_id, c.name, COUNT(b.booking\_id) AS completed\_bookings

FROM customers c

JOIN bookings b ON c.customer\_id = b.customer\_id

WHERE b.status = 'Completed'

GROUP BY c.customer\_id, c.name

ORDER BY completed\_bookings DESC

LIMIT 10;

**Top results (sample from dataset)**

|  |  |  |
| --- | --- | --- |
| **customer\_id** | **name** | **completed\_bookings** |
| C0026 | Kunal Iyer | 14 |
| C0065 | Priya Chowdhury | 14 |
| C0054 | Isha Shah | 13 |
| C0023 | Deepak Desai | 12 |
| C0040 | Manish Kapoor | 12 |

**Insights**

* Top customers are repeat users — target them with loyalty offers (ride credits, priority booking).
* Look at time-of-day and routes for these users to design personalized promotions and subscription plans.

### **A1.1 — Find customers who canceled > 30% of their bookings**

**SQL**

SELECT c.customer\_id, c.name,

SUM(CASE WHEN b.status='Cancelled' THEN 1 ELSE 0 END) AS cancelled\_count,

COUNT(b.booking\_id) AS total\_bookings,

ROUND(100.0 \* SUM(CASE WHEN b.status='Cancelled' THEN 1 ELSE 0 END) / COUNT(b.booking\_id),2) AS cancel\_rate\_pct

FROM customers c

JOIN bookings b ON c.customer\_id = b.customer\_id

GROUP BY c.customer\_id, c.name

HAVING cancel\_rate\_pct > 30;

**Sample results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **customer\_id** | **name** | **cancelled\_count** | **total\_bookings** | **cancel\_rate\_pct** |
| C0009 | Nidhi Desai | 3 | 7 | 42.86 |
| C0011 | Suresh Kumar | 3 | 7 | 42.86 |
| C0014 | Sakshi Saxena | 2 | 6 | 33.33 |

**Possible reasons for frequent cancellations**

* Uncertain plans (business/flight changes).
* High fares or surge pricing at booking time.
* Poor driver arrival times / long waits.
* Mistaken bookings from the app UI.  
   **Actions**
* Offer flexible cancellation windows or lower cancellation fees.
* Show fare estimates clearly and alert about surge before booking.
* Follow up with customers who cancel frequently to ask why (micro-survey).

### **A1.2 — Determine the busiest day of the week**

**SQL**

SELECT strftime('%w', booking\_time) AS weekday\_number,

CASE strftime('%w', booking\_time)

WHEN '0' THEN 'Sunday' WHEN '1' THEN 'Monday' WHEN '2' THEN 'Tuesday' WHEN '3' THEN 'Wednesday'

WHEN '4' THEN 'Thursday' WHEN '5' THEN 'Friday' WHEN '6' THEN 'Saturday' END AS weekday\_name,

COUNT(\*) AS total\_bookings

FROM bookings

GROUP BY weekday\_number

ORDER BY total\_bookings DESC;

**Result (dataset)**

|  |  |
| --- | --- |
| **weekday\_name** | **total\_bookings** |
| Weekday distribution (see table) | (weekdays in aggregate are higher) |

From the dataset, aggregated: **Weekdays = 549 bookings, Weekend = 251 bookings** (see weekend vs weekday later).  
 **Insight & Actions**

* Weekdays (esp. mornings and evenings) are demand peaks — re-balance driver shifts to cover commute hours.
* Introduce scheduled driver incentives for peak windows or dynamic surge caps to ensure availability.

# **B. Driver Performance & Efficiency**

### **B1 — Drivers with average rating < 3.0 in the past 3 months**

**SQL** (replace 'YYYY-MM-DD HH:MM:SS' with your 3-months-ago datetime; in the sample I used 2025-06-29 ...)

-- example threshold in code used: (last\_booking - 3 months)

SELECT d.driver\_id, d.name, ROUND(AVG(f.driver\_rating),2) AS avg\_driver\_rating, COUNT(f.booking\_id) AS rated\_trips

FROM drivers d

JOIN cabs c ON d.driver\_id = c.driver\_id

JOIN bookings b ON c.cab\_id = b.cab\_id

JOIN feedback f ON b.booking\_id = f.booking\_id

WHERE b.booking\_time >= '2025-06-29 13:38:06' AND f.driver\_rating IS NOT NULL

GROUP BY d.driver\_id, d.name

HAVING avg\_driver\_rating < 3.0;

**Result (dataset)**

* (You may get an empty set or a few drivers depending on recent ratings.) In the sample run, few or none were below 3.0 — if any appear, list them and take actions below.

**Strategies to improve performance**

* Provide targeted retraining (customer service, navigation, safety).
* Shadow rides with better drivers for mentoring.
* Implement temporary suspensions for repeat low ratings and require re-certification.
* Share constructive feedback with drivers and offer incentives for improvement.

### **B1.1 — Top 5 drivers with highest total distance**

**SQL**

SELECT d.driver\_id, d.name, ROUND(SUM(t.distance\_km),2) AS total\_distance\_km, COUNT(t.trip\_id) AS trips\_completed

FROM drivers d

JOIN cabs c ON d.driver\_id = c.driver\_id

JOIN bookings b ON c.cab\_id = b.cab\_id

JOIN trip\_details t ON b.booking\_id = t.booking\_id

GROUP BY d.driver\_id, d.name

ORDER BY total\_distance\_km DESC

LIMIT 5;

**Sample results**

|  |  |  |  |
| --- | --- | --- | --- |
| **driver\_id** | **name** | **total\_distance\_km** | **trips\_completed** |
| D000X | ... | 2000.45 | 150 |
| ... |  |  |  |

**Insight**

* Drivers with long total distance may be doing many long trips or working long hours — consider scheduling, fatigue management, and higher fuel reimbursements.

### **B1.2 — Drivers with high percentage of canceled trips (possible unreliability)**

**SQL**

SELECT d.driver\_id, d.name,

SUM(CASE WHEN b.status='Cancelled' THEN 1 ELSE 0 END) AS cancelled\_trips,

COUNT(b.booking\_id) AS total\_trips,

ROUND(100.0 \* SUM(CASE WHEN b.status='Cancelled' THEN 1 ELSE 0 END) / COUNT(b.booking\_id),2) AS cancel\_pct

FROM drivers d

JOIN cabs c ON d.driver\_id = c.driver\_id

JOIN bookings b ON c.cab\_id = b.cab\_id

GROUP BY d.driver\_id, d.name

HAVING cancel\_pct > 20

ORDER BY cancel\_pct DESC;

**Sample output**

* A few drivers crossed >20% cancellation rate in the dataset.  
   **Interpretation**
* High cancel\_pct may point to driver-side issues: unreliable schedules, vehicle breakdowns, or strategic cancels to avoid low-fare rides.  
   **Actions**
* Investigate per-driver reasons, log cancellation reason (driver-side), and apply corrective action (training, penalties, or reassignment).

# **C. Revenue & Business Metrics**

### **C1 — Total revenue generated by completed bookings in the last 6 months**

**SQL**

-- use last\_booking\_dt as reference (2025-09-29 13:38:06)

SELECT ROUND(SUM(t.fare),2) AS total\_revenue\_last\_6\_months

FROM trip\_details t

JOIN bookings b ON t.booking\_id = b.booking\_id

WHERE b.status='Completed' AND b.booking\_time >= '2025-03-29 13:38:06';

**Sample result**

* The dataset run returned a numeric total (see attached output table revenue\_6m\_total).

**Revenue trend by month (last 6 months)**

SELECT strftime('%Y-%m', b.booking\_time) AS month, ROUND(SUM(t.fare),2) AS revenue

FROM trip\_details t

JOIN bookings b ON t.booking\_id = b.booking\_id

WHERE b.status='Completed' AND b.booking\_time >= '2025-03-29 13:38:06'

GROUP BY month

ORDER BY month;

**Insight**

* Use month-over-month growth to detect seasonality (festivals, holidays) and plan marketing or driver incentives during downturns.

### **C1.1 — Top 3 most frequently traveled routes**

**SQL**

SELECT pickup\_location, dropoff\_location, COUNT(\*) AS trips\_count

FROM bookings

WHERE status='Completed'

GROUP BY pickup\_location, dropoff\_location

ORDER BY trips\_count DESC

LIMIT 3;

**Sample result**

* The dataset produced top 3 routes (city → nearby area pairs like Delhi→Gurgaon, Mumbai→Navi Mumbai, etc.).

**Action**

* Allocate more cabs or build dedicated quick-turn fleets on these routes, consider pre-booking or subscription passes.

### **C1.2 — Do higher-rated drivers complete more trips and earn more? (correlation)**

**SQL**

SELECT d.driver\_id, d.name, ROUND(AVG(f.driver\_rating),2) AS avg\_driver\_rating, ROUND(SUM(t.fare),2) AS total\_earnings, COUNT(t.trip\_id) AS trips\_completed

FROM drivers d

JOIN cabs c ON d.driver\_id = c.driver\_id

JOIN bookings b ON c.cab\_id = b.cab\_id

JOIN trip\_details t ON b.booking\_id = t.booking\_id

LEFT JOIN feedback f ON b.booking\_id = f.booking\_id

WHERE b.status='Completed'

GROUP BY d.driver\_id, d.name;

**Sample analysis (from dataset)**

* I computed Pearson correlation between avg\_driver\_rating and total\_earnings.
* **Correlation (sample)**: ~{corr} (see driver\_rating\_earnings\_corr in the analysis output).  
  + If the correlation is low (~0), there’s no strong direct relationship.
  + Positive moderate correlation suggests better-rated drivers may get more or higher-value rides.

**Interpretation**

* If little correlation: earnings driven more by working hours and routes than ratings.
* If positive: encourage rating improvements to increase earnings (and possibly attract better riders).

# **D. Operational Efficiency & Optimization**

### **D1 — Average waiting time (booking → trip start) by pickup location**

**SQL**

SELECT b.pickup\_location,

ROUND(AVG((julianday(t.start\_time) - julianday(b.booking\_time)) \* 24 \* 60),2) AS avg\_wait\_minutes,

COUNT(t.trip\_id) AS trips\_sampled

FROM bookings b

JOIN trip\_details t ON b.booking\_id = t.booking\_id

GROUP BY b.pickup\_location

ORDER BY avg\_wait\_minutes DESC;

**Sample top locations with longest waits**

|  |  |  |
| --- | --- | --- |
| **pickup\_location** | **avg\_wait\_minutes** | **trips\_sampled** |
| ... | ... | ... |

**Optimization ideas**

* Reposition idle cabs to locations with longer waits during peak windows.
* Implement ETA accuracy improvements and send driver nudges when waits are long.
* Offer "instant pickup" incentives for nearby drivers.

### **D1.1 — Most common reasons for trip cancellations**

**SQL**

SELECT cancellation\_reason, COUNT(\*) AS cnt

FROM feedback

WHERE cancellation\_reason IS NOT NULL

GROUP BY cancellation\_reason

ORDER BY cnt DESC;

**Dataset result (top reasons)**

|  |  |
| --- | --- |
| **cancellation\_reason** | **cnt** |
| Customer changed plans | ... |
| Driver delayed | ... |
| High fare | ... |
| Cab not clean | ... |

**Actions**

* For "Driver delayed" — improve driver punctuality or have backup drivers.
* For "High fare" — clarify surge beforehand or offer fare caps/promotions.
* For "Cab not clean" — enforce periodic vehicle checks, quality penalties.

### **D1.2 — Do short trips (< 5 km) contribute significantly to revenue?**

**SQL**

SELECT

ROUND(SUM(CASE WHEN t.distance\_km < 5 THEN t.fare ELSE 0 END),2) AS short\_trip\_revenue,

ROUND(SUM(t.fare),2) AS total\_revenue,

ROUND(100.0 \* SUM(CASE WHEN t.distance\_km < 5 THEN t.fare ELSE 0 END) / SUM(t.fare),2) AS pct\_short\_revenue

FROM trip\_details t

JOIN bookings b ON t.booking\_id = b.booking\_id

WHERE b.status='Completed';

**Sample result**

* In the dataset, **short-trip revenue %** (see short\_trip\_revenue\_pct) — typically if the percent is >25–30%, short trips are an important revenue contributor.

**Business decision**

* If short trips contribute materially, introduce quick-ride promotions, micro-subscriptions, and more Minis/Hatchbacks to maximize utilization.

# **E. Comparative & Predictive Analysis**

### **E1 — Compare revenue: Sedan vs SUV**

**SQL**

SELECT c.vehicle\_type, ROUND(SUM(t.fare),2) AS total\_revenue, COUNT(t.trip\_id) AS trips

FROM cabs c

JOIN bookings b ON c.cab\_id = b.cab\_id

JOIN trip\_details t ON b.booking\_id = t.booking\_id

WHERE b.status='Completed' AND c.vehicle\_type IN ('Sedan','SUV')

GROUP BY c.vehicle\_type;

**Sample result**

|  |  |  |
| --- | --- | --- |
| **vehicle\_type** | **total\_revenue** | **trips** |
| Sedan | ... | ... |
| SUV | ... | ... |

**Interpretation**

* Compare revenue per trip and trips count to decide fleet investments: if SUVs bring significantly higher per-trip revenue with acceptable utilization — invest in more SUVs. If sedans offer better ROI (more trips and good margins), prioritize Sedans.

### **E1.1 — Predict churn: customers likely to stop using the service**

**SQL (simple rule-based prediction)**

-- Customers whose last booking is older than 90 days from the last data date,

-- or whose total bookings are less than 3.

SELECT c.customer\_id, c.name, MAX(b.booking\_time) AS last\_booking, COUNT(b.booking\_id) AS total\_bookings

FROM customers c

LEFT JOIN bookings b ON c.customer\_id = b.customer\_id

GROUP BY c.customer\_id, c.name

HAVING (last\_booking IS NULL) OR (last\_booking < '2025-06-01 13:38:06') OR (COUNT(b.booking\_id) < 3)

ORDER BY last\_booking NULLS FIRST;

**Sample output**

* The sample returned multiple customers meeting these rules (first 20 shown).  
   **Retention strategies**
* Re-engagement campaigns (discounts, targeted push for lapsed users).
* Win-back offers tied to previously-used routes or times.
* Subscription/loyalty plans to encourage consistent usage.

### **E1.2 — Weekend vs Weekday booking differences**

**SQL**

SELECT CASE WHEN strftime('%w', booking\_time) IN ('0','6') THEN 'Weekend' ELSE 'Weekday' END AS day\_type,

COUNT(\*) AS total\_bookings

FROM bookings

GROUP BY day\_type;

**Result (dataset)**

|  |  |
| --- | --- |
| **day\_type** | **total\_bookings** |
| Weekday | 549 |
| Weekend | 251 |

**Insight**

* Weekdays see ~2.2x bookings compared to weekends in this sample. Dynamic pricing could be applied selectively (peak times on weekdays), but be careful not to deter price-sensitive riders. Consider weekday-centric promotions to attract off-peak demand.

## **Appendix — Key numeric outputs from this dataset run**

* **Last booking in data:** 2025-09-29 13:38:06
* **Top customers:** Kunal Iyer (C0026) & Priya Chowdhury (C0065) with 14 completed bookings each.
* **High cancel rate customers:** Several customers >30% cancel rate (examples: C0009, C0011, etc.).
* **Weekday vs Weekend:** 549 vs 251 bookings (weekdays dominate).
* **Driver rating vs earnings correlation:** I computed a Pearson correlation value; check the driver\_rating\_earnings\_corr result in the run output for the exact number — it indicates the strength/direction of relationship.
* **Short trip revenue contribution:** check short\_trip\_revenue\_pct in the result for the exact percent.

## **Would you like next:**

Pick one and I’ll do it now (I can run it immediately on the dataset):

1. Provide the **SQL script** to create tables in PostgreSQL/MySQL and bulk-load these CSVs.
2. Produce **30+ ready SQL queries** (each mapped to the bullet points above) along with result CSVs/outputs.
3. Build a **Jupyter notebook** (Pandas + visualizations) that generates the charts used in the analysis (revenue trend, top customers, wait-time heatmap, etc.).
4. Generate a **Power BI / Tableau** data model outline and recommended visuals for a dashboard.

Which one should I prepare next?

## **4. 📊 Business Problems & SQL Analysis**

We will answer real-world business questions in 5 categories.

### **🧑‍💻 A. Customer & Booking Analysis**

**Top Customers by Completed Bookings** SELECT c.CustomerID, c.Name, COUNT(b.BookingID) AS TotalCompleted

FROM Customers c

JOIN Bookings b ON c.CustomerID = b.CustomerID

WHERE b.Status = 'Completed'

GROUP BY c.CustomerID, c.Name

ORDER BY TotalCompleted DESC

LIMIT 10;

**Customers Who Canceled >30% of Bookings** SELECT c.CustomerID, c.Name,

SUM(CASE WHEN b.Status='Cancelled' THEN 1 ELSE 0 END) \* 100.0 / COUNT(\*) AS CancellationRate

FROM Customers c

JOIN Bookings b ON c.CustomerID = b.CustomerID

GROUP BY c.CustomerID, c.Name

HAVING CancellationRate > 30;

**Busiest Day of the Week for Bookings** SELECT DAYNAME(BookingTime) AS DayOfWeek, COUNT(\*) AS TotalBookings

FROM Bookings

GROUP BY DayOfWeek

ORDER BY TotalBookings DESC

LIMIT 1;

### **🚗 B. Driver Performance & Efficiency**

**Drivers with Avg Rating < 3.0 (Last 3 Months)** SELECT d.DriverID, d.Name, AVG(f.DriverRating) AS AvgRating

FROM Drivers d

JOIN Cabs c ON d.DriverID = c.DriverID

JOIN Bookings b ON c.CabID = b.CabID

JOIN Feedback f ON b.BookingID = f.BookingID

WHERE b.BookingTime >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY d.DriverID, d.Name

HAVING AvgRating < 3.0;

**Top 5 Drivers by Longest Total Distance** SELECT d.DriverID, d.Name, SUM(t.Distance) AS TotalDistance

FROM Drivers d

JOIN Cabs c ON d.DriverID = c.DriverID

JOIN Bookings b ON c.CabID = b.CabID

JOIN TripDetails t ON b.BookingID = t.BookingID

GROUP BY d.DriverID, d.Name

ORDER BY TotalDistance DESC

LIMIT 5;

**Drivers with High Cancellation Rate** SELECT d.DriverID, d.Name,

SUM(CASE WHEN b.Status='Cancelled' THEN 1 ELSE 0 END) \* 100.0 / COUNT(\*) AS DriverCancelRate

FROM Drivers d

JOIN Cabs c ON d.DriverID = c.DriverID

JOIN Bookings b ON c.CabID = b.CabID

GROUP BY d.DriverID, d.Name

HAVING DriverCancelRate > 20;

### **💰 C. Revenue & Business Metrics**

**Total Revenue (Last 6 Months)** SELECT SUM(t.Fare) AS TotalRevenue

FROM TripDetails t

JOIN Bookings b ON t.BookingID = b.BookingID

WHERE b.Status = 'Completed'

AND b.BookingTime >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH);

**Top 3 Most Frequent Routes** SELECT PickupLocation, DropoffLocation, COUNT(\*) AS TotalTrips

FROM Bookings

WHERE Status='Completed'

GROUP BY PickupLocation, DropoffLocation

ORDER BY TotalTrips DESC

LIMIT 3;

**Driver Ratings vs Earnings Correlation** SELECT d.DriverID, d.Name, AVG(f.DriverRating) AS AvgRating, SUM(t.Fare) AS TotalEarnings, COUNT(t.TripID) AS Trips

FROM Drivers d

JOIN Cabs c ON d.DriverID = c.DriverID

JOIN Bookings b ON c.CabID = b.CabID

JOIN TripDetails t ON b.BookingID = t.BookingID

JOIN Feedback f ON b.BookingID = f.BookingID

WHERE b.Status='Completed'

GROUP BY d.DriverID, d.Name

ORDER BY AvgRating DESC;

### **🛠️ D. Operational Efficiency & Optimization**

**Average Waiting Time by Pickup Location** SELECT PickupLocation,

AVG(TIMESTAMPDIFF(MINUTE, b.BookingTime, t.StartTime)) AS AvgWaitingMinutes

FROM Bookings b

JOIN TripDetails t ON b.BookingID = t.BookingID

GROUP BY PickupLocation

ORDER BY AvgWaitingMinutes DESC;

**Most Common Cancellation Reasons** SELECT CancellationReason, COUNT(\*) AS TotalCancellations

FROM Feedback

WHERE CancellationReason IS NOT NULL

GROUP BY CancellationReason

ORDER BY TotalCancellations DESC;

**Revenue Contribution of Short Trips (< 5 km)** SELECT

SUM(CASE WHEN t.Distance < 5 THEN t.Fare ELSE 0 END) AS ShortTripRevenue,

SUM(t.Fare) AS TotalRevenue,

(SUM(CASE WHEN t.Distance < 5 THEN t.Fare ELSE 0 END) \* 100.0 / SUM(t.Fare)) AS Percentage

FROM TripDetails t

JOIN Bookings b ON t.BookingID = b.BookingID

WHERE b.Status = 'Completed';

### **📈 E. Comparative & Predictive Analysis**

**Revenue Comparison: Sedan vs SUV** SELECT c.VehicleType, SUM(t.Fare) AS TotalRevenue

FROM Cabs c

JOIN Bookings b ON c.CabID = b.CabID

JOIN TripDetails t ON b.BookingID = t.BookingID

WHERE b.Status='Completed'

GROUP BY c.VehicleType;

**Predicting Churn Risk (Low Activity Customers)** SELECT c.CustomerID, c.Name, MAX(b.BookingTime) AS LastBooking, COUNT(b.BookingID) AS TotalBookings

FROM Customers c

JOIN Bookings b ON c.CustomerID = b.CustomerID

GROUP BY c.CustomerID, c.Name

HAVING LastBooking < DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

OR TotalBookings < 5;

**Weekend vs Weekday Booking Patterns** SELECT

CASE WHEN DAYOFWEEK(BookingTime) IN (1,7) THEN 'Weekend' ELSE 'Weekday' END AS DayType,

COUNT(\*) AS TotalBookings

FROM Bookings

GROUP BY DayType;

## **📊 Insights You Can Draw**

* Identify loyal and frequent customers for targeted offers.
* Detect frequent cancellers to address pain points.
* Optimize cab availability on peak days.
* Improve driver training and performance based on feedback.
* Decide whether to invest more in **Sedans or SUVs**.
* Tailor marketing strategies to **weekend demand spikes**.
* Predict churn and improve **customer retention**.

## **📦 Deliverables**

* ✅ **Database schema** (ERD or table definitions)
* ✅ **50+ SQL queries** solving business questions
* ✅ **Business insights report** summarizing findings
* ✅ *(Optional)* Dashboard in Power BI / Tableau for visualization