

SQL Analytics Project – Ola Rides Dataset

Name: Prem Pushkar

Course: MBA (DTU)

Tools: PostgreSQL, pgAdmin 4

Date: [12-06-2025]

Dataset Overview

This dataset contains 50,000 Ola ride records with details like booking ID, ride time, vehicle type, pickup/drop locations, distance, fare, payment method, and customer/driver ratings. It is used to analyze ride trends, revenue, cancellations, and customer behavior.

OLA-SQL-PROJECT

Column Name	Data Type	Description
booking_id	VARCHAR(40)	Unique ID for each ride booking
ride_timestamp	TIMESTAMP	Combined date and time when the ride was scheduled or started
booking_status	VARCHAR(30)	Status of the booking (Success , Cancelled , Incomplete , etc.)
customer_id	BIGINT	Unique identifier for the customer who booked the ride
vehicle_type	VARCHAR(20)	Type of vehicle (Auto , Mini , Bike , Prime Sedan , etc.)
pickup_location	VARCHAR(50)	Starting location of the ride
drop_location	VARCHAR(50)	Ending location of the ride
avg_vtat	NUMERIC(6,2)	Average Vehicle Turnaround Time — time taken for the vehicle to start ride
avg_ctat	NUMERIC(6,2)	Average Customer Turnaround Time — time taken for the customer to board
cancelled_by_customer	SMALLINT	1 if the ride was cancelled by the customer, otherwise 0
cancelled_by_driver	SMALLINT	1 if the ride was cancelled by the driver, otherwise 0
incomplete_rides	SMALLINT	1 if the ride was incomplete or dropped midway, otherwise 0
booking_value	NUMERIC(10,2)	Amount paid or billed for the ride in INR
payment_method	VARCHAR(30)	Mode of payment (Cash , Wallet , Card , etc.)
ride_distance	NUMERIC(6,2)	Total distance of the ride in kilometers
driver_rating	NUMERIC(3,2)	Rating given by the customer to the driver (out of 5)
customer_rating	NUMERIC(3,2)	Rating given by the driver to the customer (out of

OLA-SQL-PROJECT

Question 1: What is the total number of rides by vehicle type?

```
SELECT vehicle_type, COUNT(*) AS total_rides
```

```
FROM rides
```

```
GROUP BY vehicle_type
```

```
ORDER BY total_rides DESC;
```

The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer tree, which is collapsed. In the center is the main workspace showing the results of a SQL query. The query was:

```
SELECT vehicle_type, COUNT(*) AS total_rides
FROM rides
GROUP BY vehicle_type
ORDER BY total_rides DESC;
```

The results are displayed in a table:

	vehicle_type	total_rides
1	Prime Plus	7230
2	Bike	7204
3	Prime Sedan	7161
4	Prime SUV	7125
5	eBike	7083
6	Auto	7080
7	Mini	6984

At the bottom of the pgAdmin window, it says "Total rows: 7 Query complete 00:00:00.084".

Caption

OLA-SQL-PROJECT

QUESTION 2: What is the total and average booking revenue per vehicle type?

QUERY-SELECT

```
vehicle_type,
```

```
ROUND(AVG(booking_value),2) AS avg_revenue,
```

```
FROM rides
```

```
WHERE booking_status = 'Success'
```

```
GROUP BY vehicle_type;
```

The screenshot shows a PostgreSQL database interface with the following details:

- Object Explorer:** Shows the database structure under "ola_ride_analytics/postgres@PostgreSQL 17".
 - Servers (1)**: PostgreSQL 17
 - Databases (2)**: ola_ride_analytics (selected)
 - Casts**
 - Catalogs**
 - Event Triggers**
 - Extensions**
 - Foreign Data Wrappers**
 - Languages**
 - Publications**
 - Schemas (1)**: public
 - Aggregates**
 - Collations**
 - Domains**
 - FTS Configurations**
 - FTS Dictionaries**
 - FTS Parsers**
 - FTS Templates**
 - Foreign Tables**
 - Functions**
 - Materialized Views**
 - Operators**
 - Procedures**
 - Sequences**
 - Tables (2)**:
 - ola_rides
 - rides
 - Trigger Functions**
 - Types**
 - Views**
 - Subscriptions**
- postres**
- Casts**

SQL Editor: Displays the query results:

	vehicle_type	avg_revenue	total_revenue
1	eBike	1026.95	4875942.18
2	Bike	1021.07	4941984.02
3	Auto	1017.56	4860882.01
4	Prime Sedan	1026.64	4905303.61
5	Prime Plus	1022.00	4895388.81
6	Mini	1024.40	4826956.03
7	Prime SUV	1024.85	4876259.68

Caption

OLA-SQL-PROJECT

QUESTION 3. What are the top 5 pickup-drop location pairs by frequency?

QUERY-SELECT pickup_location, drop_location, COUNT(*) AS rides

FROM rides

GROUP BY pickup_location, drop_location

ORDER BY rides DESC

LIMIT 5;

The screenshot shows a PostgreSQL database management interface. On the left is the Object Explorer pane, which lists various database objects like servers, databases, schemas, and tables. The 'Tables' section under the 'ola_ride_analytics' database is currently selected. In the center is the main workspace showing the results of a query. The query results are presented in a table with three columns: 'pickup_location', 'drop_location', and 'rides'. The data shows five rows of pickup and drop locations with their corresponding ride counts. The bottom status bar indicates 'Total rows: 5' and 'Query complete 00:00:00.151'.

	pickup_location	drop_location	rides
1	Area-8	Area-7	38
2	Area-44	Area-36	37
3	Area-9	Area-43	37
4	Area-26	Area-11	34
5	Area-8	Area-18	34

Caption

OLA-SQL-PROJECT

QUESTION 4: What is the Average ride distance per vehicle?

QUERY: SELECT vehicle_type, ROUND(AVG(ride_distance),2) AS avg_km

FROM rides

GROUP BY vehicle_type;

The screenshot shows a PostgreSQL database management interface. On the left is the Project Explorer sidebar with sections for Servers, Databases, Schemas, and Tables. The main area displays a query results grid for a table named 'vehicle_type'. The grid has two columns: 'vehicle_type' (character varying(40)) and 'avg_km' (numeric). The data shows the average ride distance for various vehicle types: eBike (16.98), Bike (17.07), Auto (17.37), Prime Sedan (17.01), Prime SUV (16.96), Prime Plus (16.66), and Mini (17.26). The bottom status bar indicates 'Total rows: 7' and 'Query complete 00:00:00.112'.

	vehicle_type	avg_km
1	eBike	16.98
2	Bike	17.07
3	Auto	17.37
4	Prime Sedan	17.01
5	Prime SUV	16.96
6	Prime Plus	16.66
7	Mini	17.26

Caption

OLA-SQL-PROJECT

QUESTION 5: Which hour(TOP 5) of the day has the highest number of completed rides?

QUERY- SELECT DATE_PART('hour', ride_timestamp) AS hr, COUNT(*) AS rides

FROM rides

WHERE booking_status = 'Success'

GROUP BY hr

ORDER BY hr DESC

LIMIT 5;

The screenshot shows a PostgreSQL Object Explorer interface. On the left, the tree view displays the database structure under 'ola_ride_analytics'. Under 'Tables (2)', the 'rides' table is selected. The main pane shows the results of a query in a grid format. The grid has three columns: 'hr' (double precision), 'rides' (bigint), and a lock icon. The data is as follows:

hr	rides
23	1327
22	1375
21	1397
20	1374
19	1421

At the bottom of the interface, it says 'Total rows: 5 Query complete 00:00:00.082'.

Caption

OLA-SQL-PROJECT

QUESTION 6: What is the success vs cancellation rate for each vehicle type?

QUERY- SELECT vehicle_type,

```
SUM(CASE WHEN booking_status='Success' THEN 1 END)::float / COUNT(*) *100 AS  
success_pct,
```

```
SUM(cancelled_by_customer)+SUM(cancelled_by_driver) AS cancellations
```

FROM rides

```
GROUP BY vehicle_type;
```

The screenshot shows a PostgreSQL Object Explorer interface. On the left, the tree view displays the database structure under 'PostgreSQL 17' and 'ola_ride_analytics'. Under 'Tables (2)', there are two entries: 'ola_rides' and 'rides'. The main pane shows a query result grid with the following data:

	vehicle_type	success_pct	cancellations
1	eBike	67.03374276436537	1878
2	Bike	67.18489727928927	1949
3	Auto	67.47175141242938	1849
4	Prime Sedan	66.72252478704091	1931
5	Prime SUV	66.77894736842104	1920
6	Prime Plus	66.25172890733056	1981
7	Mini	67.46849942726232	1857

At the bottom of the interface, it says 'Total rows: 7 Query complete 00:00:00.100' and 'LF Ln 159, Col 23'.

Caption

OLA-SQL-PROJECT

QUESTION 7: Which locations have highest driver cancellations?

QUERY- SELECT pickup_location,
SUM(cancelled_by_driver) AS driver_cancel
FROM rides
GROUP BY pickup_location
ORDER BY driver_cancel DESC
LIMIT 10;

The screenshot shows a PostgreSQL database interface with the following details:

- Servers (1)**: PostgreSQL 17
- Databases (2)**: ola_ride_analytics (selected), postgres
- ola_ride_analytics Schema (public)**: Contains various objects like Aggregates, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (rides, ola_rides).
- Tables (2)**: rides, ola_rides
- SQL Query Result**: A table showing the top 10 locations with the highest driver cancellations.

	pickup_location	driver_cancel
1	Area-4	216
2	Area-27	216
3	Area-29	214
4	Area-41	212
5	Area-31	210
6	Area-13	210
7	Area-34	208
8	Area-32	207
9	Area-42	206
10	Area-37	206

Total rows: 10 | Query complete 00:00:00.150 | LF | Ln 16

Caption

OLA-SQL-PROJECT

QUESTION 8: How does booking value vary by ride distance buckets (e.g., 0-5km, 5-10km)?

QUERY- WITH buckets AS (

SELECT CASE

WHEN ride_distance<5 THEN '0-5 km'

WHEN ride_distance<10 THEN '5-10 km'

ELSE '10+ km' END AS bucket,

booking_value

FROM rides

WHERE booking_status='Success')

SELECT bucket,

ROUND(AVG(booking_value),2) AS avg_value

FROM buckets

GROUP BY bucket;

The screenshot shows the pgAdmin interface with the PostgreSQL 11 database selected. The left sidebar displays the database schema, including the ola_ride_analytics schema which contains tables like ola_rides and rides, and various system catalogs and configurations. The main pane shows a query results grid titled 'Data Output'.

	bucket text	avg_value numeric
1	5-10 km	1019.62
2	0-5 km	1021.90
3	10+ km	1023.96

Caption

OLA-SQL-PROJECT

QUESTION 9: What is the most common payment method for successful bookings?

QUERY- SELECT payment_method, COUNT(*) AS cnt

FROM rides

WHERE booking_status='Success'

GROUP BY payment_method

ORDER BY cnt DESC;

The screenshot shows a PostgreSQL database interface with the following details:

- Object Explorer:** On the left, it shows the database structure under "PostgreSQL 17".
 - Databases:** ola_ride_analytics (selected).
 - Schemas:** public (selected).
 - Tables:** ola_rides (selected), rides.
 - Other:** Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Sequences, Triggers, Functions, Materialized Views, Operators, Procedures, Types, Views, Subscriptions.
- Dashboard:** Shows the connection to ola_ride_analytics/postgres@PostgreSQL 17.
- Data Output:** A table showing the count of successful bookings by payment method.

	payment_method	cnt
1	Cash	8529
2	UPI	8411
3	Card	8265
4	Wallet	8198
- SQL:** A SQL editor window with the query:

```
SELECT payment_method, COUNT(*) AS cnt
FROM rides
WHERE booking_status='Success'
GROUP BY payment_method
ORDER BY cnt DESC;
```

Caption

OLA-SQL-PROJECT

QUESTION 10: Top 5 day which had the highest revenue generated ?

QUERY- SELECT CAST(ride_timestamp AS date) AS ride_day,

ROUND(SUM(bookings_value),2) AS day_revenue

FROM rides

WHERE bookings_status = 'Success'

GROUP BY ride_day

ORDER BY day_revenue DESC

LIMIT 5;

The screenshot shows a PostgreSQL database management interface. On the left is the Object Explorer pane, which displays the database schema. It includes a tree view of servers, databases, schemas, and tables. The 'Tables' node under the 'ola_ride_analytics' database is currently selected. In the center is the main workspace, which contains a SQL editor tab and a Data Output tab. The Data Output tab shows the results of the executed query. The results are presented in a table with three columns: 'ride_day', 'date', and 'day_revenue'. The data consists of five rows, each representing a day and its corresponding revenue.

	ride_day	date	day_revenue
1	2024-01-11		1204915.25
2	2024-01-28		1200027.80
3	2024-01-14		1194499.58
4	2024-01-12		1185417.71
5	2024-01-30		1166468.04

Caption

OLA-SQL-PROJECT

QUESTION 11. Which customers have the most successful rides?

QUERY-SELECT customer_id, COUNT(*) AS rides

FROM rides

WHERE booking_status='Success'

GROUP BY customer_id

ORDER BY rides DESC

LIMIT 10;

The screenshot shows a PostgreSQL database management interface. On the left is the Object Explorer tree, which includes a 'Servers' node, a 'PostgreSQL 17' node under it, and a 'Databases' node. Inside 'Databases', there's an 'ola_ride_analytics' database expanded, showing its schema. Under 'Schemas', there's a 'public' schema expanded, showing various objects like Aggregates, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and two tables: 'ola_rides' and 'rides'. There are also Trigger Functions, Types, Views, and Subscriptions. The main pane shows a query editor with the following SQL query:

```
SELECT customer_id, COUNT(*) AS rides
FROM rides
WHERE booking_status='Success'
GROUP BY customer_id
ORDER BY rides DESC
LIMIT 10;
```

The results of this query are displayed in a table titled 'Data Output'. The table has two columns: 'customer_id' and 'rides'. The data is as follows:

	customer_id	rides
1	580524	3
2	583885	3
3	369403	3
4	924373	3
5	579247	3
6	769458	3
7	476504	3
8	299613	3
9	599413	3
10	435197	2

Caption

OLA-SQL-PROJECT

QUESTION 12. What is the average customer rating by vehicle type?

QUERY-SELECT customer_id, COUNT(*) AS rides

FROM rides

GROUP BY customer_id

ORDER BY rides DESC

LIMIT 10;

The screenshot shows a PostgreSQL database interface with the following details:

- Servers:** 1 (PostgreSQL 17)
- Databases:** 2 (ola_ride_analytics, postgres)
- Tables:** 2 (rides, ola_rides)
- Current Connection:** ola_ride_analytics/postgres@PostgreSQL 17*
- Query Results:** A table showing 10 rows of data from the rides table.

	customer_id	rides
1	426604	3
2	517003	3
3	306860	3
4	416967	3
5	458084	3
6	476504	3
7	180400	3
8	299613	3
9	369403	3
10	521629	3

Total rows: 10 | Query complete 00:00:00.116 | LF | Ln 214, (

Caption

OLA-SQL-PROJECT

QUESTION 13.What is the average customer rating by vehicle type?

QUERY:- SELECT vehicle_type,

ROUND(AVG(customer_rating),2) AS avg_cust_rating

FROM rides

WHERE customer_rating>0

GROUP BY vehicle_type;

The screenshot shows the Object Explorer interface of a database management system. The left pane displays a tree structure of database objects under 'Servers (1)' and 'PostgreSQL 17'. Under 'Tables (2)', two tables are listed: 'ola_rides' and 'rides'. The right pane shows the results of a query run against the 'rides' table. The results are displayed in a grid with columns 'vehicle_type' and 'avg_cust_rating'. The data is as follows:

	vehicle_type	avg_cust_rating
1	eBike	4.00
2	Bike	4.00
3	Auto	4.01
4	Prime Sedan	4.00
5	Prime Plus	4.00
6	Mini	3.99
7	Prime SUV	4.01

Caption

OLA-SQL-PROJECT

QUESTION 14: What is the distribution of driver ratings by vehicle type? (e.g., how many below 3.5?)

QUERY-SELECT vehicle_type, COUNT(*) AS low_rated

FROM rides

WHERE driver_rating < 3.5

AND driver_rating>0

GROUP BY vehicle_type;

The screenshot shows the Object Explorer interface of a database management system. The left pane displays a tree structure of database objects under 'ola_ride_analytics' and 'postgres'. The right pane shows the results of a query execution. The query results are presented in a table titled 'Data Output' with columns 'vehicle_type' and 'low_rated'. The data is as follows:

	vehicle_type	low_rated
1	eBike	1041
2	Auto	1059
3	Bike	1114
4	Prime Sedan	1036
5	Prime Plus	1071
6	Mini	1068
7	Prime SUV	1104

Total rows: 7 Query complete 00:00:00.108

Caption

OLA-SQL-PROJECT

QUESTION 15: What is the rebooking rate of customers within 3 days of a ride?

QUERY- WITH cte AS (

```
SELECT customer_id, ride_timestamp,  
LEAD(ride_timestamp) OVER (PARTITION BY customer_id  
ORDER BY ride_timestamp) AS next_ride
```

FROM rides

WHERE booking_status='Success')

SELECT COUNT(*) FILTER (WHERE next_ride IS NOT NULL

```
AND next_ride-ride_timestamp<=INTERVAL '3 day')
```

100.0 / COUNT() AS rebook_pct

FROM cte;

The screenshot shows the Object Explorer on the left and the SQL Editor on the right. The Object Explorer displays the database structure, including the 'ola_ride_analytics' database and its tables ('rides' and 'ola_rides'). The SQL Editor shows the query for calculating the rebooking rate.

rebook_pct	numeric
1	0.38020537077508008263

Caption

OLA-SQL-PROJECT

QUESTION 16: What is the gap between a customer's first and last ride?

QUERY- SELECT customer_id,

AGE(MAX(ride_timestamp), MIN(ride_timestamp)) AS active_span

FROM rides

GROUP BY customer_id

ORDER BY active_span DESC

LIMIT 10;

The screenshot shows the pgAdmin 4 interface with the 'Object Explorer' on the left and a query results window on the right.

Object Explorer:

- Servers (1)
 - PostgreSQL 17
 - Databases (2)
 - ola_ride_analytics
 - Schemas (1)
 - public
 - Tables (2)
 - ola_rides
 - rides

Caption

OLA-SQL-PROJECT

QUESTION 17: Who are the churned customers (no ride in last 60 days)?

QUERY- SELECT customer_id

FROM rides

GROUP BY customer_id

HAVING MAX(ride_timestamp) < NOW() - INTERVAL '60 day';

The screenshot shows a PostgreSQL database interface with the following details:

- Object Explorer:** Shows the database structure with the following hierarchy:
 - Servers (1)
 - PostgreSQL 17
 - Databases (2)
 - ola_ride_analytics (selected)
 - postgres
 - Tables (2)
 - ola_rides
 - rides
 - Triggers
 - Functions
 - Types
 - Views
 - Sequences
 - Operators
 - Procedures
 - Materialized Views
 - Functions
 - Domains
 - Collations
 - Aggregates
 - Schemas (1)
 - public
- Current Connection:** ola_ride_analytics/postgres@PostgreSQL 17*
- Data Output:** Shows the results of the query in a table format.
- Table Headers:** customer_id, bigint
- Table Data:** A list of 48543 rows, each containing a customer ID. The first few rows are: 961274, 151812, 556211, 886289, 254496, 950836, 352579, 329217, 382028, 339484, 196430, 615356, 390749, 487685, 515102, 776736, 103155, 624154, 130544, 677948, 743241, 948468, 723913, 654170, 659905, 294900, 572768.
- Total Rows:** 48543
- Query Complete:** 00:00:00.100
- Page Information:** Showing rows: 1 to 1000, Page No: 1 of 49

Caption

TOTAL- 48543 CUSTOMER

OLA-SQL-PROJECT

QUESTION 18: Calculate the average VTAT (Vehicle Turnaround Time) per vehicle type.

QUERY- SELECT vehicle_type, ROUND(AVG(avg_vtat),2) AS avg_vtat

FROM rides

GROUP BY vehicle_type;

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** On the left, it lists the database structure:
 - Servers (1)
 - PostgreSQL 17 (selected)
 - Databases (2)
 - ola_ride_analytics (selected)
 - Tables (2)
 - ola_rides
 - rides
 - Subscriptions
 - postgres
 - Casts
 - Catalogs
 - Event Triggers
- Connections:** Top right shows the connection status: ola_ride_analytics/postgres@PostgreSQL 17*
- Data Output:** The main pane displays a table with the following data:

	vehicle_type	avg_vtat
1	eBike	10.49
2	Bike	10.45
3	Auto	10.45
4	Prime Sedan	10.59
5	Prime SUV	10.51
6	Prime Plus	10.46
7	Mini	10.42
- Toolbar:** Includes icons for file operations, search, and various database management functions.
- Status Bar:** At the bottom, it shows "Total rows: 7" and "Query complete 00:00:00.111".

Caption

OLA-SQL-PROJECT

QUESTION 19: Calculate average CTAT (Customer Turnaround Time) across time slots.(4 hr gap)

QUERY - SELECT CONCAT(FLOOR(EXTRACT(hour FROM ride_timestamp)/4)*4,':00') AS slot_start,

ROUND(AVG(avg_ctat),2) AS avg_ctat

FROM rides

GROUP BY slot_start

ORDER BY slot_start;

	slot_start	avg_ctat
1	0:00	15.69
2	12:00	15.52
3	16:00	15.54
4	20:00	15.51
5	4:00	15.65
6	8:00	15.47

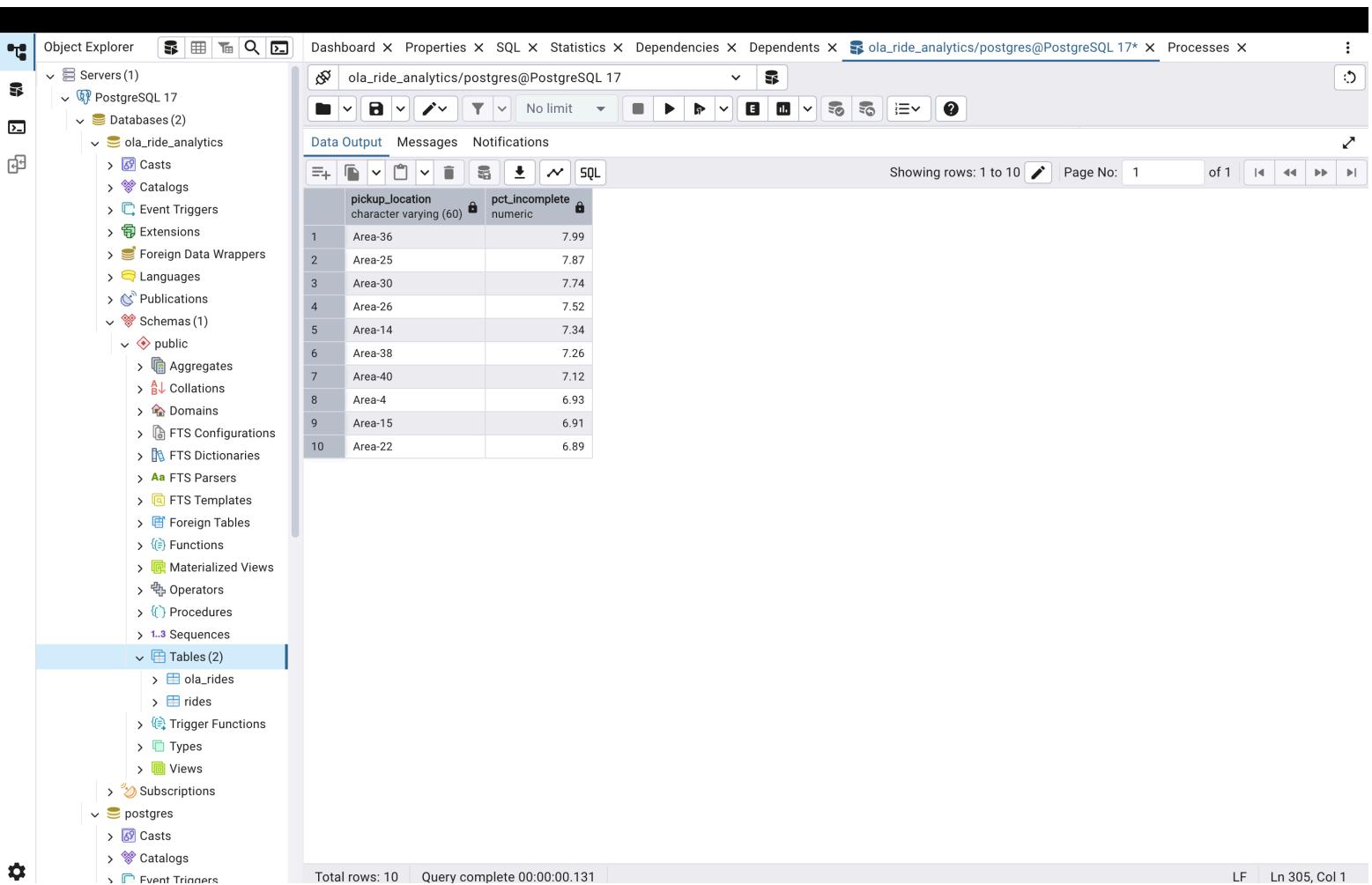
Caption

OLA-SQL-PROJECT

QUESTION 20: What is the percentage of incomplete rides overall and by location?

QUERY-

```
SELECT ROUND(SUM(incomplete_rides)::numeric / COUNT(*)*100,2) AS pct_incomplete  
FROM rides;  
  
SELECT pickup_location,  
       ROUND(SUM(incomplete_rides)::numeric / COUNT(*)*100,2) AS pct_incomplete  
FROM rides  
GROUP BY pickup_location  
ORDER BY pct_incomplete DESC  
LIMIT 10;
```



The screenshot shows a PostgreSQL database management interface. On the left is the Object Explorer pane, which lists the database schema. It includes the 'ola_ride_analytics' database with its tables ('rides', 'ola_rides'), functions, and triggers. Other databases like 'postgres' and 'information_schema' are also listed. The main area is a query editor window titled 'ola_ride_analytics/postgres@PostgreSQL 17'. It displays the results of the SQL query provided above. The results are presented in a table with two columns: 'pickup_location' and 'pct_incomplete'. The data shows the top 10 locations with the highest percentages of incomplete rides, ordered from highest to lowest.

	pickup_location	pct_incomplete
1	Area-36	7.99
2	Area-25	7.87
3	Area-30	7.74
4	Area-26	7.52
5	Area-14	7.34
6	Area-38	7.26
7	Area-40	7.12
8	Area-4	6.93
9	Area-15	6.91
10	Area-22	6.89

Caption

OLA-SQL-PROJECT

QUESTION 21: Identify locations with highest ratio of failed to successful rides.

QUERY- SELECT pickup_location,

```
SUM(CASE WHEN booking_status != 'Success' THEN 1 END)::float /
```

```
COUNT(*) *100 AS fail_pct
```

```
FROM rides
```

```
GROUP BY pickup_location
```

```
ORDER BY fail_pct DESC
```

```
LIMIT 10;
```

	pickup_location	fail_pct
1	Area-27	37.382198952879584
2	Area-36	36.72199170124482
3	Area-37	35.949098621421
4	Area-34	35.54455445544554
5	Area-30	35.317460317460316
6	Area-4	35.294117647058826
7	Area-14	35.115303983228515
8	Area-26	34.959349593495936
9	Area-7	34.942287513116476
10	Area-41	34.81262327416174

Caption

OLA-SQL-PROJECT

QUESTION 22: Identify customers with revenue above the 90th percentile.

QUERY-WITH tot AS (

SELECT customer_id, SUM(booking_value) AS revenue

FROM rides

WHERE booking_status = 'Success'

GROUP BY customer_id),

percentile_90 AS (

SELECT PERCENTILE_CONT(0.9) WITHIN GROUP (ORDER BY revenue) AS p90

FROM tot)

SELECT t.*

FROM tot t

JOIN percentile_90 p ON t.revenue > p.p90;

The screenshot shows a PostgreSQL database management interface. The left sidebar (Object Explorer) displays the database structure, including servers, databases (PostgreSQL 17, ola_ride_analytics), and various schema objects like casts, catalogs, event triggers, extensions, foreign data wrappers, languages, publications, and tables. The 'Tables' section under 'Tables (2)' is currently selected. The main window shows a table with two columns: 'customer_id' (bigint) and 'revenue' (numeric). The table contains 32 rows of data, with the last row being a summary row. The bottom status bar indicates 'Total rows: 3280' and 'Query complete 00:00:00.293'.

	customer_id	revenue
1	556211	1955.97
2	615356	1865.95
3	723913	2128.78
4	179129	3446.89
5	669288	1852.90
6	153907	1882.16
7	139368	2901.78
8	587567	1888.49
9	125646	1915.04
10	258143	1933.52
11	351931	1888.25
12	576348	1964.32
13	106728	1954.77
14	541855	1995.11
15	524745	2159.00
16	599413	4263.67
17	421323	1823.99
18	122390	1957.51
19	236039	1820.32
20	938541	1870.10
21	310632	1934.76
22	357970	1989.28
23	632501	2472.84
24	463798	2861.52
25	565122	1886.09
26	886252	1879.47
27	780294	2193.29
28	1000000	10000.00

Caption

OLA-SQL-PROJECT

QUESTION 23:Find cumulative revenue day by day for the current month.

QUERY- WITH daily AS (SELECT CAST(ride_timestamp AS date) AS day,

SUM(booking_value) AS re

FROM rides

WHERE booking_status = 'Success'

GROUP BY day)

SELECT day,

rev,

SUM(rev) OVER (ORDER BY day) AS cum_rev

FROM daily

ORDER BY day;

The screenshot shows the pgAdmin 4 interface. On the left is the Object Explorer tree, which includes a 'Servers' node, a 'PostgreSQL 17' node, a 'Databases' node with 'ola_ride_analytics' selected, and various schema and table nodes under it. The 'Tables' node is currently selected. At the top, there's a toolbar with various icons. The main area is a SQL results grid titled 'ola_ride_analytics/postgres@PostgreSQL 17'. It has three tabs: 'Data Output' (selected), 'Messages', and 'Notifications'. The results grid shows a table with columns: 'day' (date), 'rev' (numeric), and 'cum_rev' (numeric). The data consists of 27 rows from January 1 to January 27, 2024, showing the cumulative revenue over time.

	day date	rev numeric	cum_rev numeric
1	2024-01-01	1127718.00	1127718.00
2	2024-01-02	1152663.76	2280381.76
3	2024-01-03	1096442.05	3376823.81
4	2024-01-04	1155183.38	4532007.19
5	2024-01-05	1134906.93	5666914.12
6	2024-01-06	1135994.40	6802908.52
7	2024-01-07	1108589.56	7911498.08
8	2024-01-08	1130244.29	9041742.37
9	2024-01-09	1144715.87	10186458.24
10	2024-01-10	1155880.45	11342338.69
11	2024-01-11	1204915.25	12547253.94
12	2024-01-12	1185417.71	13732671.65
13	2024-01-13	1119345.33	14852016.98
14	2024-01-14	1194499.58	16046516.56
15	2024-01-15	1145143.30	17191659.86
16	2024-01-16	1085445.91	18277105.77
17	2024-01-17	1110952.42	19388058.19
18	2024-01-18	1150029.75	20538087.94
19	2024-01-19	1143088.50	21681176.44
20	2024-01-20	1135916.24	22817092.68
21	2024-01-21	1100984.20	23918076.88
22	2024-01-22	1134873.23	25052950.11
23	2024-01-23	1120286.67	26173236.78
24	2024-01-24	1078015.20	27251251.98
25	2024-01-25	1103564.84	28354816.82
26	2024-01-26	1134860.50	29489677.32
27	2024-01-27	1141354.25	30631031.57

Caption

OLA-SQL-PROJECT

QUESTION 24: Rank pickup locations by ride frequency and show top 5.

QUERY- SELECT pickup_location,

COUNT(*) AS rides,

RANK() OVER (ORDER BY COUNT(*) DESC) AS rnk

FROM rides

GROUP BY pickup_location

ORDER BY rnk

LIMIT 5;

The screenshot shows the pgAdmin 4 interface with the following details:

- Servers:** PostgreSQL 17
- Databases:** ola_ride_analytics
- Tables:** ola_rides, rides
- Query Results:** A table showing the top 5 pickup locations by ride frequency.

	pickup_location	rides	rnk
1	Area-39	1097	1
2	Area-4	1054	2
3	Area-8	1048	3
4	Area-29	1041	4
5	Area-9	1037	5

Total rows: 5 | Query complete 00:00:00.138 | LF | Ln 361, 0

Caption

OLA-SQL-PROJECT

QUESTION 25: Calculate monthly average booking value per customer.

QUERY- SELECT DATE_TRUNC('month',ride_timestamp) AS month,

customer_id,

ROUND(AVG(booking_value),2) AS avg_value

FROM rides

WHERE booking_status='Success'

GROUP BY month, customer_id

ORDER BY month, avg_value DESC;

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure under "PostgreSQL 17" and "ola_ride_analytics".
- Dashboard:** Shows the connection status to "ola_ride_analytics/postgres@PostgreSQL 17".
- Data Output:** Displays the results of the executed SQL query.
- SQL Tab:** Shows the raw SQL code of the query.
- Messages:** Shows the execution status: "Query complete 00:00:00.134".
- Notifications:** Shows no notifications.
- Toolbar:** Includes icons for file operations, search, and help.

Table Headers:

month	customer_id	avg_value
timestamp with time zone	bigint	numeric

Table Data:

2024-01-01 00:00:00+05:30	229002	2000.00
2024-01-01 00:00:00+05:30	949778	1999.95
2024-01-01 00:00:00+05:30	873839	1999.94
2024-01-01 00:00:00+05:30	649178	1999.93
2024-01-01 00:00:00+05:30	350191	1999.80
2024-01-01 00:00:00+05:30	704441	1999.80
2024-01-01 00:00:00+05:30	678637	1999.80
2024-01-01 00:00:00+05:30	247827	1999.80
2024-01-01 00:00:00+05:30	677316	1999.72
2024-01-01 00:00:00+05:30	321423	1999.67
2024-01-01 00:00:00+05:30	318276	1999.64
2024-01-01 00:00:00+05:30	629280	1999.64
2024-01-01 00:00:00+05:30	348504	1999.56
2024-01-01 00:00:00+05:30	222264	1999.50
2024-01-01 00:00:00+05:30	809556	1999.29
2024-01-01 00:00:00+05:30	731765	1999.25
2024-01-01 00:00:00+05:30	593396	1999.23
2024-01-01 00:00:00+05:30	672310	1999.05
2024-01-01 00:00:00+05:30	847481	1998.94
2024-01-01 00:00:00+05:30	415303	1998.75
2024-01-01 00:00:00+05:30	970111	1998.73
2024-01-01 00:00:00+05:30	548705	1998.72
2024-01-01 00:00:00+05:30	875739	1998.68
2024-01-01 00:00:00+05:30	802214	1998.60
2024-01-01 00:00:00+05:30	337681	1998.59
2024-01-01 00:00:00+05:30	311050	1998.55
2024-01-01 00:00:00+05:30	222997	1998.49

Caption

TOTAL ROWS =32795

QUESTION 26