Target Retail Store
Business Insights & Data Analysis Using SQL

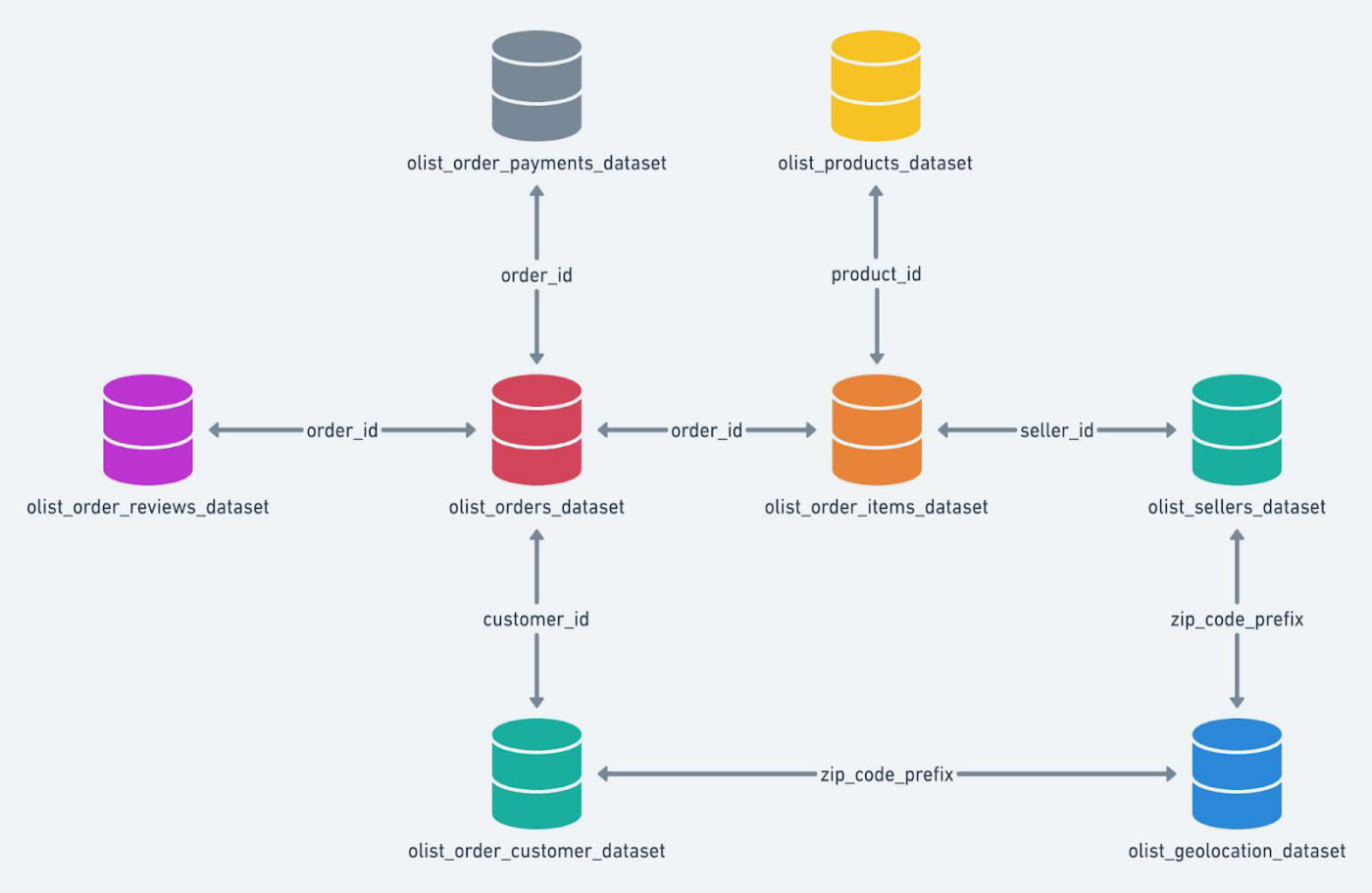
ADHARSHA TN

Problem Statement: - As a **Data Analyst**, I am responsible for analyzing the given datasets to extract meaningful insights and provide data-driven recommendations. The objective is to identify key trends, enhance business strategies, and support informed decision-making using **SQL-based data analysis**.

Report Structure: -

1. Understanding the Dataset.
2. Initial Exploration.
3. In-Depth Exploration.
4. Evolution of E-commerce orders in the Brazil region
5. Impact on Economy.
6. Sales, Freight, and Delivery Time Analysis.
7. Payments Analysis.
8. Actionable Insights & Recommendations

Data Set Schema:



1. Checking the structure & characteristics of the datasets:

A. Data type of all columns in the "customers & orders" table.

```
SELECT
    column_name,
    data_type,
FROM `Target_case_study.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers';
```

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Results per page: 50 1 – 5 of 5

INFERENCE: - The Customer table comprises five columns. Among them, Customer_id, Customer_unique_id, Customer_City, and Customer_state hold string data types, while customer_zip_code_prefix holds integer data type.

```
SELECT
    column_name,
    data_type,
FROM `Target_case_study.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'orders';
```

Query results SAVE RESULTS OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	column_name	data_type
1	order_id	STRING
2	customer_id	STRING
3	order_status	STRING
4	order_purchase_timestamp	TIMESTAMP
5	order_approved_at	TIMESTAMP
6	order_delivered_carrier_date	TIMESTAMP
7	order_delivered_customer_date	TIMESTAMP
8	order_estimated_delivery_date	TIMESTAMP

2. Initial Exploration.

A. Getting the time range between which, the orders were placed.

```
select
  extract (date from min(order_purchase_timestamp)) as First_order_date,
  extract (date from max(order_purchase_timestamp)) as Last_order_date,
  date_diff(extract (date from max(order_purchase_timestamp)),
            extract (date from min(order_purchase_timestamp)),day) as
total_days
from `Target_case_study.orders`
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	First_order_date	Last_order_date	total_days
1	2016-09-04	2018-10-17	773

INFERENCE: - Based on the given time range, the first order was placed on September 4, 2016, and the last order was placed on October 17, 2018. The time period between the first and last order is 773 Days.

B. Count the Cities & States of customers who ordered during the given period.

```
select
  count(distinct(c.customer_city)) as count_city,
  count(distinct(c.customer_state)) as count_state
from `Target_case_study.orders` as o
left join `Target_case_study.customers`as c
on o.customer_id = c.customer_id
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	count_city	count_state
1	4119	27

INFERENCE: - The orders were received from 4,119 cities across 27 states.

C. The Count of reviews received from the customers.

```
SELECT
  review_score,
  COUNT(review_score) as review_Count
from `Target_case_study.order_reviews`
group by review_score
order by review_Count desc
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	review_score	review_Count
1	5	57328
2	4	19142
3	1	11424
4	3	8179
5	2	3151

INFERENCE: - The majority of reviews are **5-star ratings**, followed by **4-star ratings** in frequency.

3. In-depth Exploration:

A. Is there a growing trend in the no. of orders placed over the past years?

```
WITH t1 AS (
    SELECT
        EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
        COUNT(order_id) AS yearly_sales_count
    FROM `Target_case_study.orders`
    WHERE order_status <> "canceled"
    GROUP BY year
)
SELECT *,
    LAG(yearly_sales_count) OVER (ORDER BY year) AS previous_year_sales,
    ROUND(
        ((yearly_sales_count - LAG(yearly_sales_count) OVER (ORDER BY
year)) /
        LAG(yearly_sales_count) OVER (ORDER BY year)) * 100, 2
    ) AS YoY_growth_percentage
FROM t1
ORDER BY year;
```

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

Row	year	yearly_sales_count	previous_year_sal...	YoY_growth_perc...
1	2016	303	null	null
2	2017	44836	303	14697.36
3	2018	53677	44836	19.72

Inference:- Since 2016 data is incomplete, the 14,697.36% growth is not fully reliable for trend analysis. The actual growth rate from 2016 to 2017 might be Differ if all 2016 data were available.
A 19.72% increase in orders from 44,836 (2017) to 53,677 (2018) indicates sustained demand and business expansion.

B. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

```
with t1 as (SELECT
  Extract(month from order_purchase_timestamp) as Month,
  Extract(year from order_purchase_timestamp) as Year,
  count(distinct(order_id)) as orders_count
from `Target_case_study.orders`
WHERE order_status <> "canceled"
group by Year,Month
order by orders_count desc)

SELECT
  Month,
  sum(case when year = 2016 then orders_count else 0 END) as Order_of_2016,
  sum(case when year = 2017 then orders_count else 0 END) as Order_of_2017,
  sum(case when year = 2018 then orders_count else 0 END) as Order_of_2018,
  SUM(orders_count) AS Total_Orders
FROM T1
GROUP BY month
ORDER BY MONTH
```

Row	Month	Order_of_2016	Order_of_2017	Order_of_2018	Total_Orders
1	1	0	797	7235	8032
2	2	0	1763	6655	8418
3	3	0	2649	7185	9834
4	4	0	2386	6924	9310
5	5	0	3671	6849	10520
6	6	0	3229	6149	9378
7	7	0	3998	6251	10249
8	8	0	4304	6428	10732
9	9	2	4265	1	4268
10	10	300	4605	0	4905
11	11	0	7507	0	7507
12	12	1	5662	0	5663

Inference:- The analysis reveals that peak order months vary by year, with significant spikes in May, July, and August across multiple years, while specific months like November 2017, January 2018, and March 2018 had the highest orders in their respective years. This suggests that demand patterns are influenced by seasonal trends, promotions, or external factors. To maximize revenue, businesses should implement dynamic pricing and strategic resource allocation tailored to each year's peak months.

c. During what time of the day, do the Brazilian customers mostly place them orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs.: Dawn
- 7-12 hrs.: Mornings
- 13-18 hrs.: Afternoon
- 19-23 hrs.: Night

```
SELECT
  CASE
  WHEN EXTRACT (hour FROM order_purchase_timestamp) BETWEEN 0 AND 6 THEN
"Dawn"
  WHEN EXTRACT (hour FROM order_purchase_timestamp) BETWEEN 7 AND 12 THEN
"Morning"
  WHEN EXTRACT (hour FROM order_purchase_timestamp) BETWEEN 13 AND 18 THEN
"Afternoon"
  WHEN EXTRACT (hour FROM order_purchase_timestamp) BETWEEN 19 AND 23 THEN
"Night"
  end as order_time,
  count(distinct(order_id)) order_count
FROM `Target_case_study.orders`
group by order_time
ORDER BY order_count desc
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	order_time	order_count
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

Inference: - The highest number of orders occurred between 13:00 and 18:00, indicating peak demand during the Afternoon hours. This trend could inform businesses to focus resources and staffing during this time for optimal efficiency and customer service.

4. Evolution of E-commerce orders in the Brazil region:

A. Get the month on month no. of orders placed in each state.

```
With T1 AS(SELECT
  c.customer_state,
  extract(month from o.order_purchase_timestamp) as month,
  count(distinct(o.order_id)) as order_count
FROM `Target_case_study.orders` as o
left join `Target_case_study.customers` as c
on o.customer_id = c.customer_id
group by c.customer_state,month)

SELECT
  customer_state,
  SUM(CASE WHEN MONTH = 1 THEN order_count ELSE 0 END) AS Jan,
  SUM(CASE WHEN MONTH = 2 THEN order_count ELSE 0 END) AS Feb,
  SUM(CASE WHEN MONTH = 3 THEN order_count ELSE 0 END) AS Mar,
  SUM(CASE WHEN MONTH = 4 THEN order_count ELSE 0 END) AS Apr,
  SUM(CASE WHEN MONTH = 5 THEN order_count ELSE 0 END) AS May,
  SUM(CASE WHEN MONTH = 6 THEN order_count ELSE 0 END) AS Jun,
  SUM(CASE WHEN MONTH = 7 THEN order_count ELSE 0 END) AS Jul,
  SUM(CASE WHEN MONTH = 8 THEN order_count ELSE 0 END) AS Aug,
  SUM(CASE WHEN MONTH = 9 THEN order_count ELSE 0 END) AS Sep,
  SUM(CASE WHEN MONTH = 10 THEN order_count ELSE 0 END) AS Oct,
  SUM(CASE WHEN MONTH = 11 THEN order_count ELSE 0 END) AS Nov,
  SUM(CASE WHEN MONTH = 12 THEN order_count ELSE 0 END) AS Dec,
  sum(order_count) as Total_Orders
FROM T1
Group by customer_state
order by Total_Orders Desc
```

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_s...	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Total_Orders
1	SP	3351	3357	4047	3967	4632	4104	4381	4982	1648	1908	3012	2357	41746
2	RJ	990	1176	1302	1172	1321	1128	1288	1307	612	725	1048	783	12852
3	MG	971	1063	1237	1061	1190	1080	1111	1177	511	600	943	691	11635
4	RS	427	473	569	488	559	526	565	599	279	276	422	283	5466
5	PR	443	460	504	500	524	478	523	556	183	225	378	271	5045
6	SC	345	316	362	351	379	321	356	365	157	189	303	193	3637
7	BA	264	273	340	318	368	307	405	323	170	170	250	192	3380
8	DF	151	196	207	183	208	220	243	232	97	104	168	131	2140
9	ES	159	186	182	188	228	204	206	200	93	104	170	113	2033
10	GO	164	176	199	177	226	184	192	213	88	117	157	127	2020
11	PE	113	146	153	154	174	140	210	170	76	87	126	103	1652
12	CE	99	101	126	143	136	121	140	130	77	74	108	81	1336
13	PA	82	83	109	107	75	92	96	104	41	58	70	58	975

Inference:- The state of SP consistently registers the highest number of orders each month, while states like RR, AP, AM, AC, RO, and TO consistently record the lowest number of orders.

To enhance overall sales, it's crucial to allocate more attention and resources towards improving performance in these states with lower order numbers.

B. How are the customers distributed across all the states?

```
SELECT
  Customer_state,
  count(distinct(customer_id)) as customer_count
FROM  `Target_case_study.customers`
group by customer_state
order by customer_count desc
```

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	customer_count
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Results per page: 50 | 1 – 27 of 27

Query results | SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	customer_count
18	PI	495
19	RN	485
20	AL	413
21	SE	350
22	TO	280
23	RO	253
24	AM	148
25	AC	81
26	AP	68
27	RR	46

Results per page: 50 | 1 – 27 of 27

Inference:-

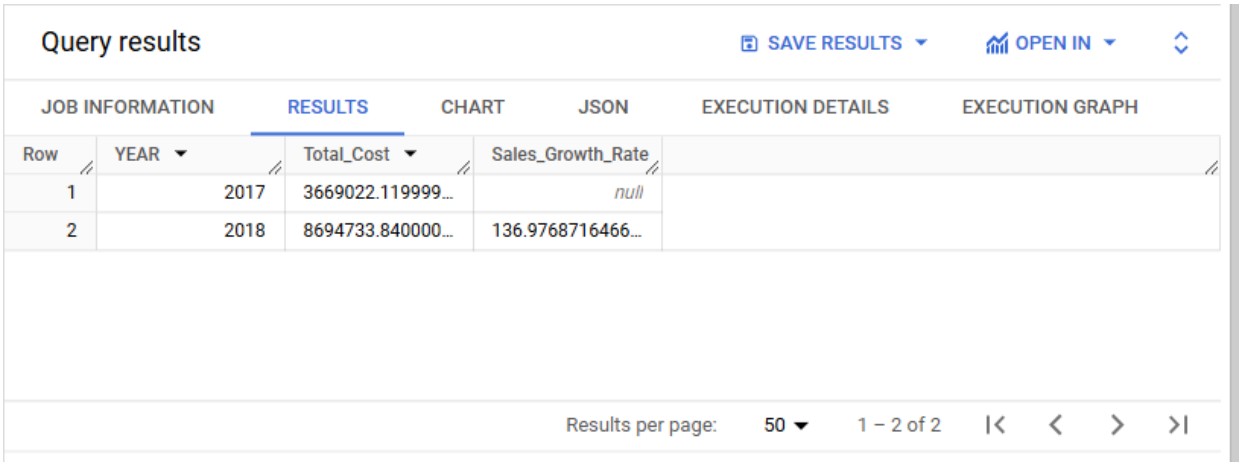
- The SP state has the highest number of customers, while RR, AP, and AC states have fewer than 100 customers each.
- The SP state has the highest number of customers, while RR, AP, and AC states have fewer than 100 customers each.

5. Impact on Economy

A. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

```
With T1 as (SELECT
  extract(year from o.order_purchase_timestamp) as year,
  sum(p.payment_value) as total_cost
from `Target_case_study.orders` as o
left join `Target_case_study.payments`as p
on o.order_id = p.order_id
where extract(month from o.order_purchase_timestamp) between 1 and 8
group by year)

SELECT
  YEAR,
  Total_Cost,
  (Total_Cost - (lag (Total_Cost) over (order by year))) / lag (Total_Cost)
over (order by year)*100 as Sales_Growth_Rate
from T1
```



INFERENCE: - From 2017 to 2018, the cost to orders surged by 136.97%, reflecting robust growth, when considering only January to August orders in both years.

B. Calculate the Total & Average value of order price for each state.

```
SELECT
  c.customer_state ,
  round(sum(P.payment_value),2) as Total_sales,
  round(avg(P.payment_value),2) as Avg_sales
from `Target_case_study.customers`as c
inner join `Target_case_study.orders` as o
on c.customer_id = o.customer_id
inner join `Target_case_study.payments`as P
on o.order_id = P.order_id
group by c.customer_state
order by Total_sales desc
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	Total_sales	Avg_sales
1	SP	5998226.96	137.5
2	RJ	2144379.69	158.53
3	MG	1872257.26	154.71
4	RS	890898.54	157.18
5	PR	811156.38	154.15
6	SC	623086.43	165.98
7	BA	616645.82	170.82
8	DF	355141.08	161.13
9	GO	350092.31	165.76

Results per page: 50 | 1 – 27 of 27

Inference: - The highest order price was placed by customers from the state of SP, while the highest average order price was observed among customers from the state of PB.

c. Calculate the Total & Average value of order freight for each state?

```
SELECT
  c.customer_state ,
  round(sum(oi.freight_value),2) as Total_Fright_Value,
  round(avg(oi.freight_value),2) as Avg_Fright_Value
from `Target_case_study.customers`as c
inner join `Target_case_study.orders` as o
on c.customer_id = o.customer_id
inner join `Target_case_study.order_items`as oi
on o.order_id = oi.order_id
group by c.customer_state
order by Avg_Fright_Value desc
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	Total_Fright_Value	Avg_Fright_Value
1	RR	2235.19	42.98
2	PB	25719.73	42.72
3	RO	11417.38	41.07
4	AC	3686.75	40.07
5	PI	21218.2	39.15
6	MA	31523.77	38.26
7	TO	11732.68	37.25
8	SE	14111.47	36.65
9	AL	15914.59	35.84
10	PA	38699.3	35.83

Results per page: 50 1 – 27 of 27

Inference: - Highest Freight value paid by SP state With Lowest Avg Freight Price & Highest Avg Freight value Paid by RR State with Less total Fright Price.

The state with high average freight costs but low total expenditure, signaling a need to review our freight pricing policies. Exploring volume discounts, shipping consolidation, and negotiation strategies with partners can help optimize freight expenses.

6. Sales, Freight, and Delivery Time Analysis.

A. Find the no. of days taken to deliver each order from the order's purchase date as delivery time. Also, calculate the difference (in days) between the estimated & actual delivery date of an order. Do this in a single query?

```
SELECT
 order_id,
 DATE_DIFF(EXTRACT(DATE FROM order_delivered_customer_date),EXTRACT
(DATE FROM order_purchase_timestamp),day) AS
Total_days_taken_to_delivery,
 DATE_DIFF(EXTRACT(DATE FROM order_estimated_delivery_date),EXTRACT
(DATE FROM order_delivered_customer_date),day) AS
Diff_estimated_delivery
FROM `Target_case_study.orders`
where order_status = "delivered"
order by Total_days_taken_to_delivery desc
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	order_id	Total_days_taken_to_	Diff_estimated_deliv
1	ca07593549f1816d26a572e06...	210	-181
2	1b3190b2dfa9d789e1f14c05b...	208	-188
3	440d0d17af552815d15a9e41a...	196	-165
4	285ab9426d6982034523a855f...	195	-166
5	2fb597c2f772eca01b1f5c561b...	195	-155
6	0f4519c5f1c541ddec9f21b3bd...	194	-161
7	47b40429ed8cce3aee9199792...	191	-175
8	2fe324febf907e3ea3f2aa9650...	190	-167
9	c27815f7e3dd0b926b5855262...	188	-162
10	2d7561026d542c8dbd8f0daea...	188	-159

Inference: - The longest delivery took 210 days, while the shortest delivery was fulfilled on the same day as the order. Notably, one order was fulfilled 188 days after the estimated date, while another was completed 147 days ahead of schedule.

B. Find out the top 5 states with the highest & lowest average freight value.

```
(SELECT
  c.customer_state,
  round(AVG(oi.freight_value),2) as avg_freight_value
from `Target_case_study.order_items`as oi
inner join `Target_case_study.orders`as o
on oi.order_id = o.order_id
inner join `Target_case_study.customers` as c
on o.customer_id = c.customer_id
group by c.customer_state
order by avg_freight_value desc
limit 5)
union all
(SELECT
  c.customer_state,
  round(AVG(oi.freight_value),2) as avg_freight_value
from `Target_case_study.order_items`as oi
inner join `Target_case_study.orders`as o
on oi.order_id = o.order_id
inner join `Target_case_study.customers` as c
on o.customer_id = c.customer_id
group by c.customer_state
order by avg_freight_value
limit 5)
```

Query results

SAVE RESULTS | OPEN IN

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	avg_freight_value
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15
6	SP	15.15
7	PR	20.53
8	MG	20.63
9	RJ	20.96
10	DF	21.04

Inference: - Top 5 Rows given the Highest Average Freight Value, While the following 5 rows are Lowest Freight Value.

C. Find out the top 5 states with the highest & lowest average delivery time.

```
with T1 AS (
SELECT
  c.customer_state,
  avg(date_diff(o.order_estimated_delivery_date,o.order_purchase_times
tamp,day)) as Avg_time_to_delivery,
  dense_rank()over (order by
avg(date_diff(o.order_estimated_delivery_date,o.order_purchase_timesta
mp,day))desc) Highest_time_taken,
  dense_rank()over (order by
avg(date_diff(o.order_estimated_delivery_date,o.order_purchase_timesta
mp,day))asc) Lowest_time_taken
FROM `Target_case_study.orders` as o
left join `Target_case_study.customers` as c
on o.customer_id = c.customer_id
where o.order_status = "delivered"
group by c.customer_state)

SELECT
  customer_state,
  round(Avg_time_to_delivery,2)
FROM T1
WHERE Highest_time_taken between 1 and 5 or Lowest_time_taken between 1
and 5
order by Avg_time_to_delivery desc
```

Query results

SAVE RESULTS | OPEN IN

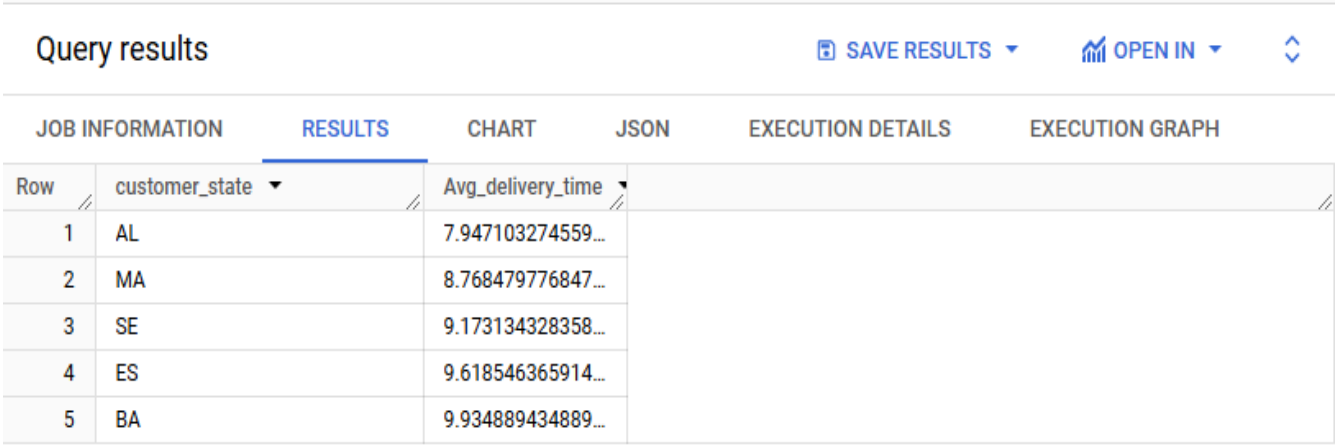
JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	f0_
1	AP	45.87
2	RR	45.63
3	AM	44.92
4	AC	40.72
5	RO	38.39
6	ES	25.22
7	PR	24.25
8	MG	24.19
9	DF	23.95
10	SP	18.78

INFERENCE: - The top 5 rows reflect the Highest average time taken for product delivery, while the subsequent rows represent the Lowest delivery times. Therefore, it's essential to concentrate on the Top 5 States to identify areas for improvement and enhance customer satisfaction by reducing delivery times.

D. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
SELECT
 c.customer_state,
 AVG(DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_
date,day)) AS Avg_delivery_time
FROM `Target_case_study.orders` as o
left join `Target_case_study.customers`as c
ON o.customer_id = c.customer_id
where order_status = "delivered"
GROUP BY c.customer_state
order by Avg_delivery_time
LIMIT 5
```



Row	customer_state	Avg_delivery_time
1	AL	7.947103274559...
2	MA	8.768479776847...
3	SE	9.173134328358...
4	ES	9.618546365914...
5	BA	9.934889434889...

Inference: - We found out the top 5 states where deliveries are really fast compared to the expected time. In AL state customers get their orders about 7.9 days earlier than expected.

7. Analysis based on the payments:

A. Find the month on month no. of orders placed using different payment types.

```
WITH T1 AS (
SELECT
  EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Number,
  FORMAT_TIMESTAMP('%B', o.order_purchase_timestamp) AS Month,
  P.payment_type,
  count(distinct(o.order_id)) as order_count
FROM `Target_case_study.orders` O
Inner Join `Target_case_study.payments` P
on o.order_id = P.order_id
group by Month,Number, P.payment_type)
,
T2 AS(SELECT
  Number,
  Month,
  sum(case when payment_type = "credit_card" then order_count else 0 end
) as Credit_Card,
  sum(case when payment_type = "UPI" then order_count else 0 end ) as UPI,
  sum(case when payment_type = "voucher" then order_count else 0 end ) as
voucher,
  sum(case when payment_type = "debit_card" then order_count else 0 end
) as debit_card,
  sum(case when payment_type = "not_defined" then order_count else 0 end
) as Others
FROM T1
GROUP BY MONTH,Number),
T3 AS (SELECT
  13 as Number,
  "Total_orders" as Month,
  SUM(credit_card) as Credit_Card,
  SUM(UPI) AS UPI,
  SUM(voucher) as voucher,
  sum(debit_card) as debit_card,
  sum(others) as others
FROM T2
GROUP BY month),
T4 AS (SELECT
  14 as Number,
  "%_Total_Orders" as month,
  round(credit_card/(select count(*) from
`Target_case_study.orders`)*100,2) as credit_card,
```

```
  round(UPI/(select count(*) from `Target_case_study.orders`)*100,2) as
UPI,
  round(voucher/(select count(*) from `Target_case_study.orders`)*100,2)
as voucher,
  round(debit_card/(select count(*) from
`Target_case_study.orders`)*100,2) as debit_card,
  round(Others/(select count(*) from `Target_case_study.orders`)*100,4)
as Others
from T3)

SELECT * FROM T2
UNION ALL
SELECT * FROM T3
UNION ALL
SELECT * FROM T4
order by Number
```

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

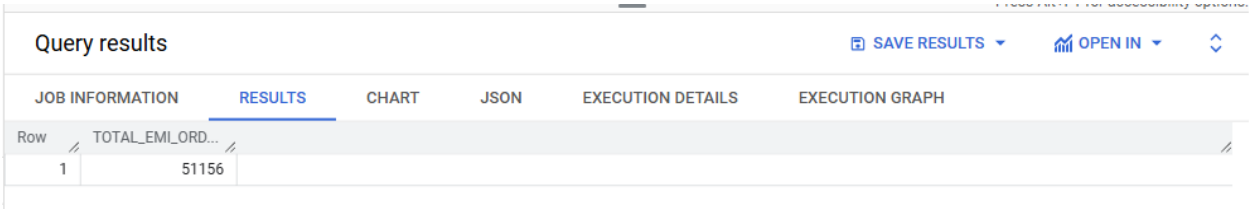
Row	Number	Month	Credit_Card	UPI	voucher	debit_card	Others
1	1	January	6093.0	1715.0	337.0	118.0	0.0
2	2	February	6582.0	1723.0	288.0	82.0	0.0
3	3	March	7682.0	1942.0	395.0	109.0	0.0
4	4	April	7276.0	1783.0	353.0	124.0	0.0
5	5	May	8308.0	2035.0	374.0	81.0	0.0
6	6	June	7248.0	1807.0	373.0	208.0	0.0
7	7	July	7810.0	2074.0	417.0	264.0	0.0
8	8	August	8235.0	2077.0	430.0	311.0	2.0
9	9	September	3277.0	903.0	189.0	43.0	1.0
10	10	October	3763.0	1056.0	223.0	54.0	0.0
11	11	November	5867.0	1509.0	267.0	70.0	0.0
12	12	December	4364.0	1160.0	220.0	64.0	0.0
13	13	Total_orders	76505.0	19784.0	3866.0	1528.0	3.0
14	14	%_Total_Orders	76.94	19.9	3.89	1.54	0.003

Inference: -

- Credit Card is the most used payment method across all months, contributing 76.94% of total transactions.
- Peak Credit Card Usage: May (8308 orders) and August (8235 orders) have the highest credit card transactions.
- Lowest Credit Card Usage: September (3277 orders) and October (3763 orders) show a significant dip in credit card transactions.
- September and October might need promotional offers or discounts to boost transactions.

B. Find the no. of orders placed on the basis of the payment installments that have been paid.

```
SELECT
  COUNT (DISTINCT(order_id)) AS TOTAL_EMI_ORDERS
FROM `Target_case_study.payments`
WHERE payment_installments > 1 and payment_installments <>
payment_sequential
```



Inference:- A total of 51,170 orders have been paid through the EMI method, with the consideration that at least 1 EMI payment was made for each order.

Recommendations: -

- **Leverage Peak Seasons:** Since orders peak in May, July, and August, implement targeted promotions and discounts during these months to maximize revenue.

- **Boost Off-Peak Months:** September and October show a dip in credit card transactions. Offering special discounts, cashback, or loyalty points can help stimulate purchases.

- **Encourage Repeat Purchases:** Majority of orders are one-time. Personalized discounts, loyalty programs, and targeted email campaigns can improve retention.

- **Expand in Underperforming States:** SP leads in orders, while RR, AP, AM, AC, RO, and TO have the lowest. Invest in regional marketing and logistics to improve penetration in low-order regions.

- **Reduce Long Delivery Delays:** The longest delivery took 210 days, and some orders were fulfilled 188 days late. Strengthen supply chain partnerships, optimize routing, and introduce real-time tracking.

- **Improve Freight Cost Efficiency:** RR pays the highest average freight cost but has low total freight expenditure. Offer bulk shipping discounts, optimize last-mile delivery, and negotiate better rates with carriers.

- **Expand Payment Instalments:** 51,170 orders used EMI. Promote instalment plans further to encourage high-value purchases.

- **Incentivize Alternative Payment Methods:** Credit card transactions dominate (76.94%), while UPI and vouchers are underutilized. Introducing cashback offers and additional discounts on alternative payment methods can boost usage.