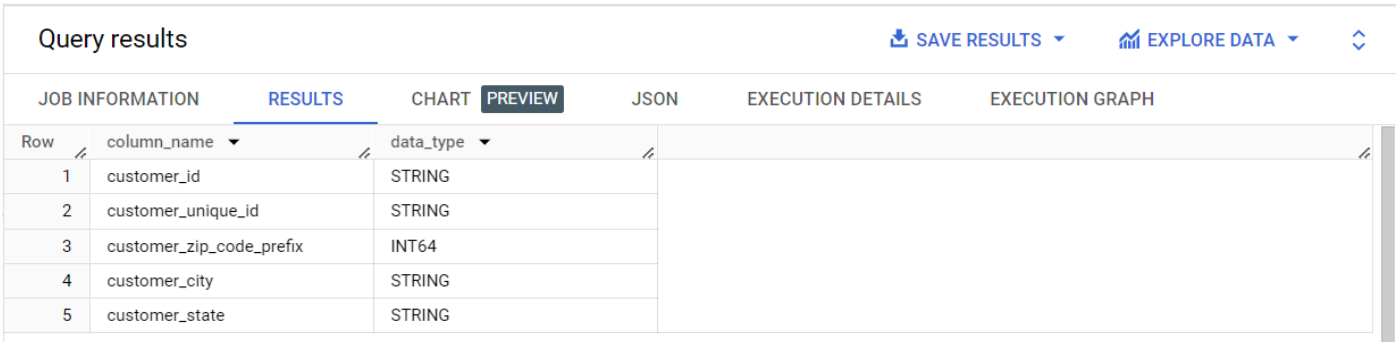
BUSINESS CASE STUDY: TARGET

1.1 Data type of all columns in the "customers" table

```
SELECT column_name, data_type
FROM `scaler-dsml-sql-409613.Target_SQL_Business_Case.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers';
```



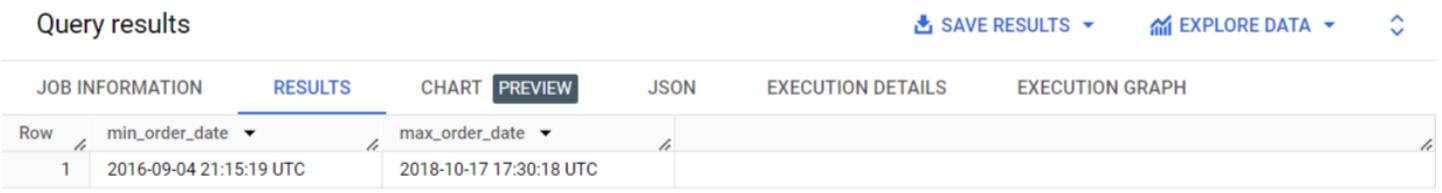
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Insight: The provided dataset contains crucial customer information such as unique IDs, location details, and state codes.

Action: Ensure data integrity and accuracy in customer records to facilitate personalized marketing and customer service initiatives.

1.2 Get the time range between which the orders were placed.

```
SELECT MIN(order_purchase_timestamp) AS min_order_date, MAX(order_purchase_timestamp)
AS max_order_date
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders;
```



Row	min_order_date	max_order_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insight: The dataset spans from September 2016 to October 2018, providing a comprehensive view of two years' worth of order data.

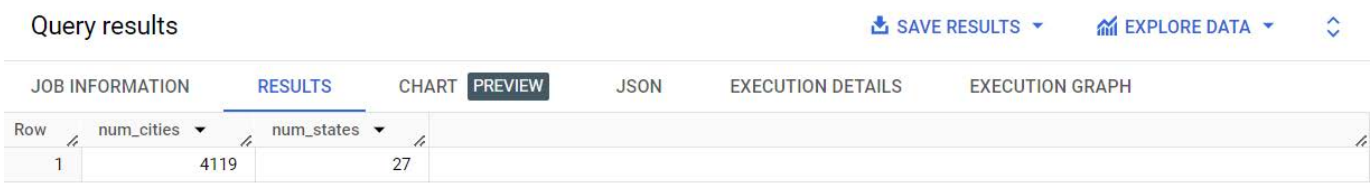
Action: Use this timeframe to analyze trends, identify patterns, and make informed business decisions.

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

```
SELECT COUNT(DISTINCT customer_city) AS num_cities #total_cities and states,
COUNT(DISTINCT customer_state) AS num_states

FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o

JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id;
```



Row	num_cities	num_states
1	4119	27

Insight: The dataset includes orders from 4,119 cities across 27 states in Brazil, indicating a broad geographical reach.

Action: Conduct targeted marketing campaigns and logistics optimization efforts tailored to different regions for enhanced customer engagement and operational efficiency.

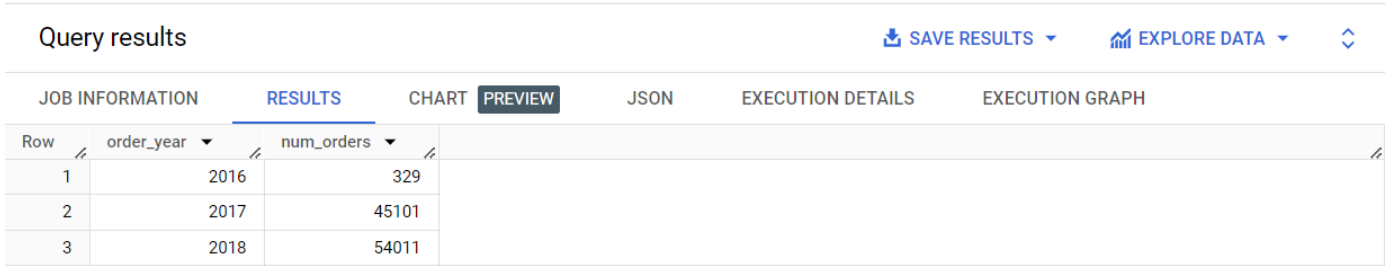
2.1 Is there a growing trend in the no. of orders placed over the past years?

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year, COUNT(*) AS
num_orders

FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders

GROUP BY order_year

ORDER BY order_year;
```



Row	order_year	num_orders
1	2016	329
2	2017	45101
3	2018	54011

Insight: There's a notable increase in the number of orders over the years, with significant growth from 2016 to 2017 and further expansion in 2018.

Action: Allocate resources to meet growing demand, including scaling up inventory, logistics, and customer service capabilities.

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

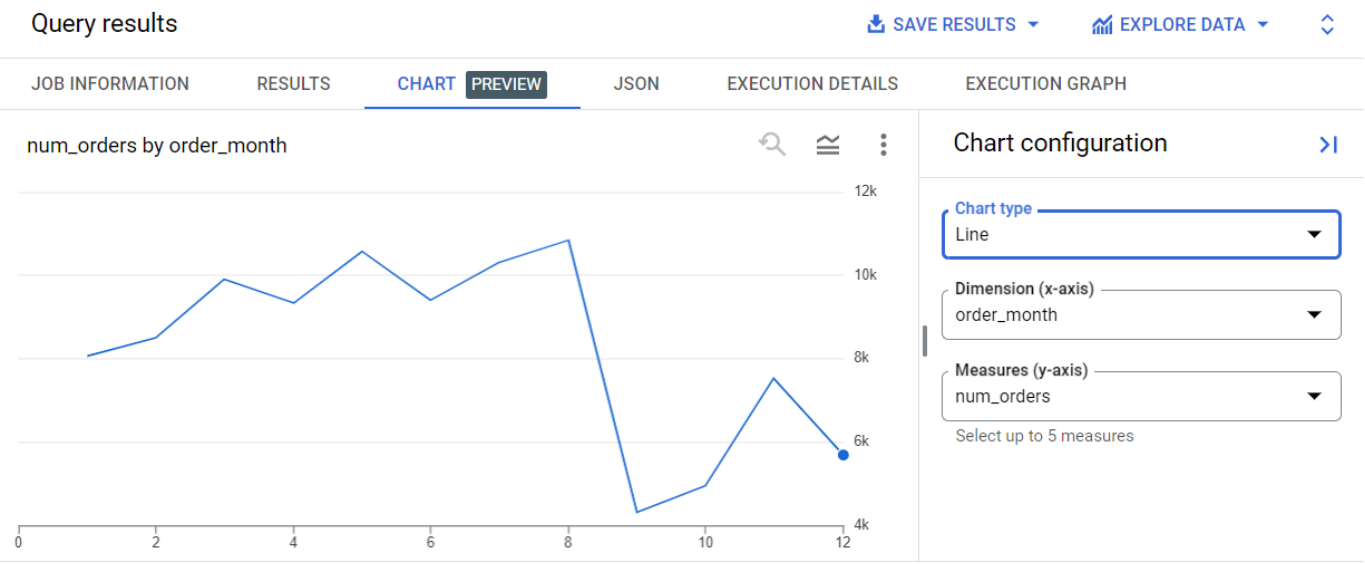
```
SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month, COUNT(*) AS
num_orders
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders
GROUP BY order_month
ORDER BY order_month;
```

Query results

SAVE RESULTS EXPLORE DATA

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

Row	order_month	num_orders
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959



Insight: Monthly order volumes exhibit seasonal fluctuations, with higher demand observed during certain months, such as May and August.

Action: Plan promotional campaigns and inventory management strategies to align with peak demand periods and maximize sales opportunities.

2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or 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'
        ELSE 'Night'
    END AS order_time_period,
    COUNT(*) AS num_orders
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders
GROUP BY order_time_period;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	order_time_period	num_orders
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

Insight: Most orders are placed during the afternoon, suggesting that customers prefer shopping during this time of day.

Action: Optimize marketing efforts and promotions to coincide with peak shopping hours to drive sales and enhance customer satisfaction.

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

```
SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS order_year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS order_month,
    c.customer_state,
    COUNT(*) AS num_orders
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id
```

```
GROUP BY order_year, order_month, c.customer_state
ORDER BY order_year, order_month, c.customer_state;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	order_year	order_month	customer_state	num_orders
1	2016	9	RR	1
2	2016	9	RS	1
3	2016	9	SP	2
4	2016	10	AL	2
5	2016	10	BA	4
6	2016	10	CE	8
7	2016	10	DF	6
8	2016	10	ES	4
9	2016	10	GO	9
10	2016	10	MA	4

Insight: The dataset provides insights into regional order trends, with orders distributed across various states each month.

Action: Analyze state-level order patterns to identify opportunities for market expansion and targeted promotional activities.

3.2 How are the customers distributed across all the states?

```
SELECT customer_state, COUNT(DISTINCT customer_id) AS num_customers
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.customers
GROUP BY customer_state
ORDER BY num_customers DESC;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	num_customers
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

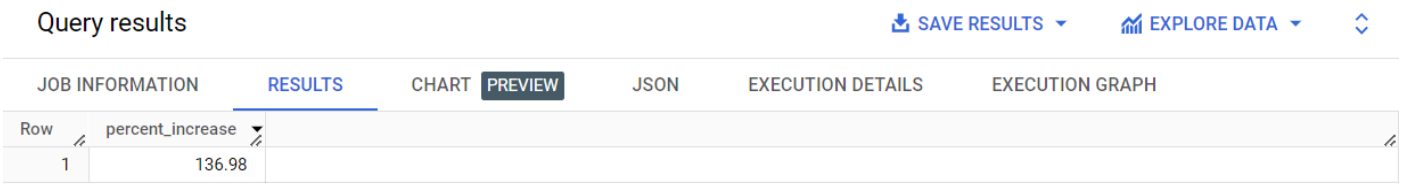
Results per page: 50 | 1 – 27 of 27

Insight: São Paulo (SP) has the highest number of customers, indicating a strong presence in Brazil's most populous state.

Action: Leverage insights from high-density regions to refine marketing strategies and replicate successful approaches in other states to increase market penetration.

4.1 Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
You can use the "payment_value" column in the payments table to get the cost of orders.

```
SELECT
    ROUND(((SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2018 THEN
p.payment_value ELSE 0 END) -
            SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017 THEN
p.payment_value ELSE 0 END)) /
           SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017 THEN
p.payment_value ELSE 0 END)) * 100, 2) AS percent_increase
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.payments p ON o.order_id =
p.order_id
WHERE EXTRACT(YEAR FROM o.order_purchase_timestamp) IN (2017, 2018)
AND EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8;
```



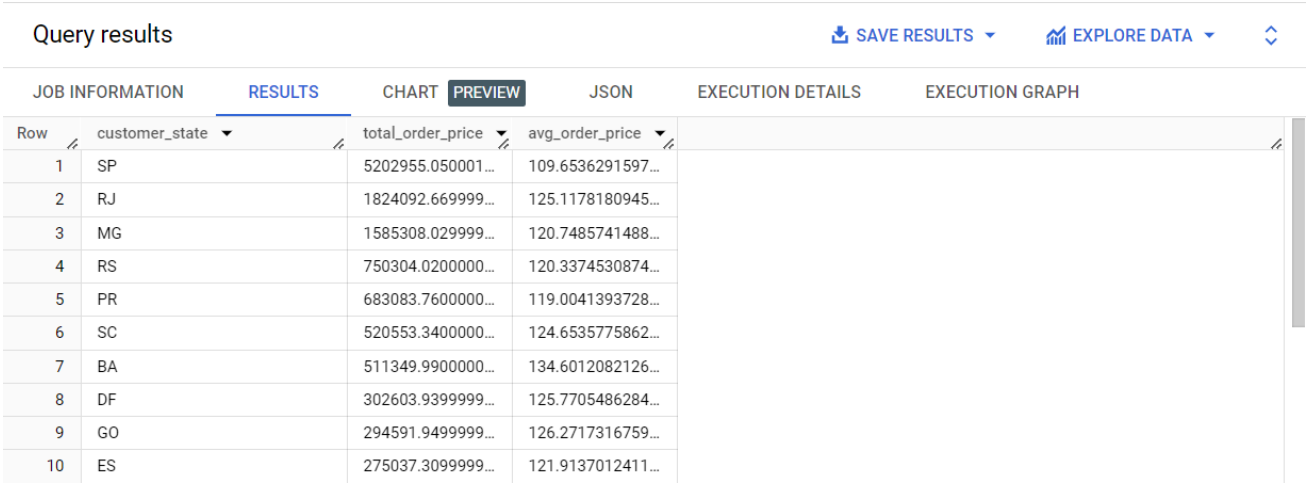
Insight: There was a substantial increase in the cost of orders from 2017 to 2018, indicating potential changes in consumer behavior or pricing strategies.

Action: Conduct further analysis to understand the factors driving the increase and adjust pricing strategies accordingly to maintain competitiveness.

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

```
SELECT
    c.customer_state,
```

```
    SUM(p.price) AS total_order_price,
    AVG(p.price) AS avg_order_price
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.order_items p ON o.order_id =
p.order_id
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id
GROUP BY c.customer_state
ORDER BY total_order_price DESC;
```



Row	customer_state	total_order_price	avg_order_price
1	SP	5202955.050001...	109.6536291597...
2	RJ	1824092.669999...	125.1178180945...
3	MG	1585308.029999...	120.7485741488...
4	RS	750304.0200000...	120.3374530874...
5	PR	683083.7600000...	119.0041393728...
6	SC	520553.3400000...	124.6535775862...
7	BA	511349.9900000...	134.6012082126...
8	DF	302603.9399999...	125.7705486284...
9	GO	294591.9499999...	126.2717316759...
10	ES	275037.3099999...	121.9137012411...

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

```
SELECT
    c.customer_state,
    SUM(p.freight_value) AS total_freight_value,
    AVG(p.freight_value) AS avg_freight_value
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.order_items p ON o.order_id =
p.order_id
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id
GROUP BY c.customer_state
ORDER BY total_freight_value DESC;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	total_freight_value	avg_freight_value
1	SP	718723.0699999833	15.147275390419248
2	RJ	305589.31000000035	20.96092393168248
3	MG	270853.46000000357	20.630166806306541
4	RS	135522.74000000212	21.735804330392945
5	PR	117851.68000000139	20.531651567944248
6	BA	100156.67999999883	26.363958936562248
7	SC	89660.260000000431	21.470368773946436
8	PE	59449.6599999999	32.917862679955796
9	GO	53114.979999999865	22.766815259322794
10	DF	50625.499999999811	21.041354945968383

Insight: There are variations in order value and freight costs across different states, reflecting regional economic dynamics and logistical challenges.

Action: Tailor pricing and shipping strategies to account for regional differences and optimize profitability in each market.

5.1 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.

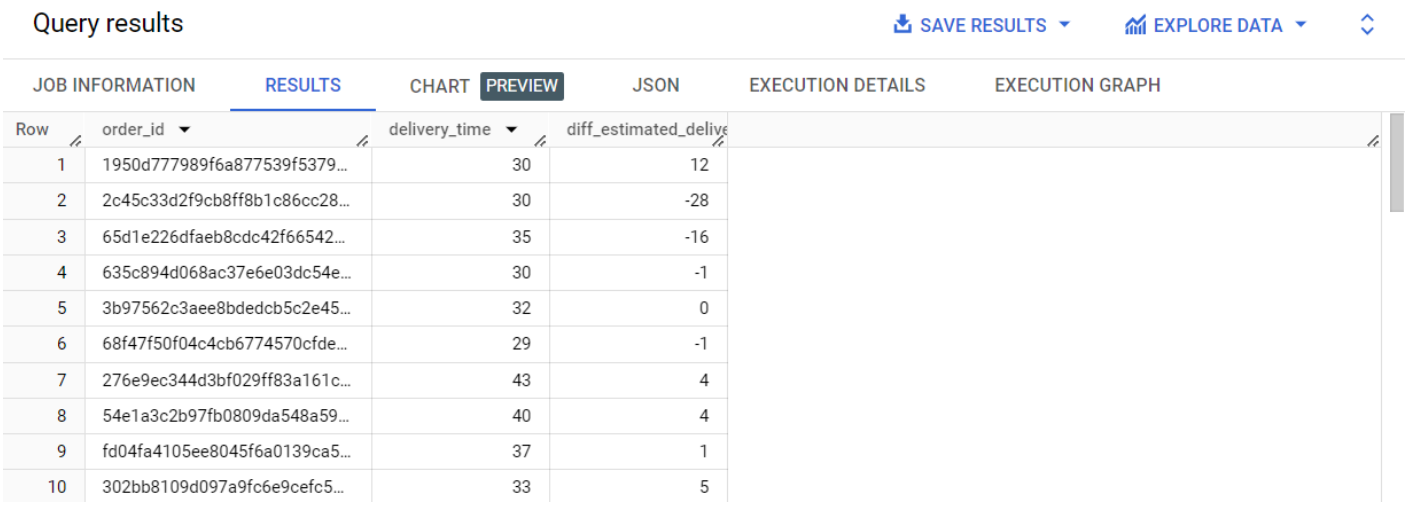
```
SELECT

    o.order_id,

    DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY) AS
delivery_time,

    DATE_DIFF(o.order_delivered_customer_date,o.order_estimated_delivery_date, DAY) AS
diff_estimated_delivery

FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o;
```



Row	order_id	delivery_time	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	-28
3	65d1e226dfaeb8cdc42f66542...	35	-16
4	635c894d068ac37e6e03dc54e...	30	-1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	-1
7	276e9ec344d3bf029ff83a161c...	43	4
8	54e1a3c2b97fb0809da548a59...	40	4
9	fd04fa4105ee8045f6a0139ca5...	37	1
10	302bb8109d097a9fc6e9cefc5...	33	5

Insight: Some orders experienced delays in delivery compared to the estimated delivery date, potentially impacting customer satisfaction.

Action: Enhance logistics operations and communication with customers to minimize delivery delays and improve overall service quality.

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

```
(
SELECT
    'Highest' AS category,
    c.customer_state,
    AVG(oi.freight_value) AS avg_freight_value
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.order_items oi ON o.order_id =
oi.order_id
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id
GROUP BY c.customer_state
ORDER BY avg_freight_value DESC
LIMIT 5
)

UNION ALL

(
```

```
SELECT
    'Lowest' AS category,
    c.customer_state,
    AVG(oi.freight_value) AS avg_freight_value
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.order_items oi ON o.order_id =
oi.order_id
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id
GROUP BY c.customer_state
ORDER BY avg_freight_value ASC
LIMIT 5
);
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	category	customer_state	avg_freight_value
1	Highest	RR	42.98442307692...
2	Highest	PB	42.72380398671...
3	Highest	RO	41.06971223021...
4	Highest	AC	40.07336956521...
5	Highest	PI	39.14797047970...
6	Lowest	SP	15.14727539041...
7	Lowest	PR	20.53165156794...
8	Lowest	MG	20.63016680630...
9	Lowest	RJ	20.96092393168...
10	Lowest	DF	21.04135494596...

Insight: Freight costs vary significantly among states, influencing overall order profitability and customer pricing perceptions.

Action: Explore opportunities to negotiate better freight rates and optimize shipping routes to reduce costs and improve competitiveness.

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

```
(
SELECT
    'Highest' AS category,
    c.customer_state,
```

```
    AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS
avg_delivery_time

FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o

JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id

GROUP BY c.customer_state

ORDER BY avg_delivery_time DESC

LIMIT 5

)


UNION ALL


(

SELECT

    'Lowest' AS category,

    c.customer_state,

    AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS
avg_delivery_time

FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o

JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id

GROUP BY c.customer_state

ORDER BY avg_delivery_time ASC

LIMIT 5

);
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	category	customer_state	avg_delivery_time
1	Lowest	SP	8.298061489072...
2	Lowest	PR	11.52671135486...
3	Lowest	MG	11.54381329810...
4	Lowest	DF	12.50913461538...
5	Lowest	SC	14.47956019171...
6	Highest	RR	28.97560975609...
7	Highest	AP	26.73134328358...
8	Highest	AM	25.98620689655...
9	Highest	AL	24.04030226700...
10	Highest	PA	23.31606765327...

Insight: Delivery times vary across states, with some regions experiencing faster or slower order fulfillment processes.

Action: Invest in infrastructure and logistics capabilities to improve delivery speed and reliability, particularly in regions with longer average delivery times.

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

You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

```
WITH state_delivery_speed AS (

    SELECT

        c.customer_state,

        AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_estimated_delivery_date,
DAY)) AS avg_delivery_speed,

        RANK() OVER (ORDER BY AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_estimated_delivery_date, DAY)) DESC) AS delivery_speed_rank

    FROM

        scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o

    JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.customers c ON o.customer_id =
c.customer_id

    GROUP BY

        c.customer_state

)

SELECT

    customer_state,
```

```
    avg_delivery_speed
FROM
    state_delivery_speed
WHERE
    delivery_speed_rank <= 5;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	avg_delivery_speed
1	MA	-8.76847977684797
2	BA	-9.93488943488941
3	AL	-7.9471032745591943
4	SE	-9.1731343283582127
5	ES	-9.6185463659147885

Insight: Certain states demonstrate efficient delivery operations, exceeding customer expectations by delivering orders ahead of the estimated delivery date.

Action: Identify and replicate best practices from top-performing states to enhance delivery performance and customer satisfaction nationwide.

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

```
SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS order_year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS order_month,
    p.payment_type,
    COUNT(*) AS num_orders
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.orders o
JOIN scaler-dsml-sql-409613.Target_SQL_Business_Case.payments p ON o.order_id =
p.order_id
GROUP BY order_year, order_month, p.payment_type
ORDER BY order_year, order_month, p.payment_type;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	order_year	order_month	payment_type	num_orders
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	254
4	2016	10	debit_card	2
5	2016	10	voucher	23
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	583
9	2017	1	debit_card	9
10	2017	1	voucher	61

Insight: Payment method preferences vary across months, reflecting changing consumer behaviors and market trends.

Action: Analyze payment data to understand shifting consumer preferences and tailor payment options and promotions accordingly.

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

```
SELECT
    p.payment_installments,
    COUNT(*) AS num_orders
FROM scaler-dsml-sql-409613.Target_SQL_Business_Case.payments p
GROUP BY p.payment_installments
ORDER BY p.payment_installments;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	payment_installment	num_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644

Insight: Installment payment plans are popular among customers, providing flexibility and affordability in purchasing.

Action: Develop targeted marketing campaigns and promotional offers to promote installment payment options and attract customers with varying budget constraints.