Business Case-Target SQL

Target is one of the world's most recognized brands and one of America's leading retailers.This business case has information of 100k orders from 2016 to 2018 made at Target in Brazil. Its features allow viewing an order from multiple dimensions.
Data is available in 8 csv files:
1. customers.csv
2. geolocation.csv
3. order_items.csv
4. payments.csv
5. reviews.csv
6. orders.csv
7. products.csv
8. sellers.csv

Q1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**

1) **Data type of all columns in the "customers" table.**

```
SELECT table_name, column_name, data_type FROM
scaler-dsml-sql-390311.
Target_Brazil_Operations_2016_2018.
INFORMATION_SCHEMA.COLUMNS
WHERE table_name= 'customers'
```

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

Each feature or columns of different CSV files are described below:
The customers.csv contain following features:
customer_id : Id of the consumer who made the purchase.
customer_unique_id :Unique Id of the consumer.
customer_zip_code_prefix : Zip Code of the location of the - consumer.
customer_city : Name of the City from where order is made.
customer_state :State Code from where order is made.

Insights and Recommendations :-

- We have 99,441 customers with data available.
- We have 96096 numbers of Unique Customers ids.
- 14994 different locations of customers
- Customers are from 4119 cities and 27 states from Brazil.
- total 99441 customers are there in given data.

Q.2 Get the time range between which the orders were placed.

```
1  SELECT  MIN(order_purchase_timestamp) AS
   min_timestamp, MAX(order_purchase_timestamp) AS
   max_timestamp
2  FROM scaler-dsml-sql-390311.
   Target_Brazil_Operations_2016_2018.orders
```

Press Alt+F1 for Accessibility O

Query results SAVE RESULTS

JOB INFORMATION | RESULTS | JSON | EX

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

Insights and Recommendations :-

(a)Time range: This result of this query provides the earliest (min_timestamp) and latest (max_timestamp)purchase timestamp from the order table. This gives you the overall time range during which the orders were placed. (b) Analysis: By knowing the time range of the orders, you can determine the duration for which the data was analyzed. This can be useful for reporting or understanding the timeframe of the dataset.

Q.3. Count the number of Cities and States in our dataset.

```
1  SELECT COUNT(DISTINCT geolocation_city) AS
   num_cities, COUNT(DISTINCT geolocation_state) AS
   num_states
2  FROM scaler-dsml-sql-390311.
   Target_Brazil_Operations_2016_2018.geolocation
```

Press Alt+F1 for Accessibility

Query results SAVE RESULTS

JOB INFORMATION | RESULTS | JSON | E

w	num_cities	num_states
1	8011	27

Insights and Recommendations :- In this, geolocation, is the name of a table containing the city and state info. By using the count(Distinct Column) we can count the number of unique

values in each column. The result will be returned with 2 columns ‘num_cities’ and ‘num_states’ representing the counts of unique cities & states resp.

Q2)In-depth Exploration

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-390311.
Target_Brazil_Operations_2016_2018.orders
GROUP BY order_year
ORDER BY order_year
```

Press Alt+F1 for Accessibility Op

Query results SAVE RESULTS

JOB INFORMATION | RESULTS | JSON | EX

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

Insights and Recommendations :-

By examining the results, we can determine whether there is a growing trend in the number of orders placed over the past years. If the ‘num_orders’ values consistently increase from year to year, it suggests a growing trend. Conversely, if the values remain relatively stable or decrease, it indicates a different pattern.

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-390311.
Target_Brazil_Operations_2016_2018.orders
GROUP BY order_month
ORDER BY order_month
```

JOB INFORMATION | RESULTS

Row	order_month	num_orders
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
11	11	7544
12	12	5674

Insights and Recommendations :-

i)Monthly order volume:- The query provides information on the number of orders placed in each month. By analyzing the “num_orders” column, we can identify which months had higher or lower order volumes.

ii) Seasonal pattern:- By examining the order counts across different months, we can identify any recurring seasonal patterns.

Some Recommendations: i) Allocate resources such as staffing, inventory and market efforts based on the identified seasonal patterns. Increase resources during high demand months to meet customer demands and optimize operational efficiency.

ii) Plan targeted promotions and marketing campaigns during peak months or low demand periods to boost sales and drive customer engagement.

iii) It utilizes the information on monthly order volumes to improve demand forecasting and optimized inventory management.

iv) Analyze the factors contributing to the higher order volumes during specific months. Identify customer preferences, behavior and trends to personalize marketing messages and enhance customer engagement.

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

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

```sql
SELECT
  CASE
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 0 AND EXTRACT(HOUR FROM order_purchase_timestamp) <= 6 THEN 'Dawn'
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 7 AND EXTRACT(HOUR FROM order_purchase_timestamp) <= 12 THEN 'Morning'
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 13 AND EXTRACT(HOUR FROM order_purchase_timestamp) <= 18 THEN 'Afternoon'
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 19 AND EXTRACT(HOUR FROM order_purchase_timestamp) <= 23 THEN 'Night'
  END AS time_of_day,
  COUNT(*) AS order_count
FROM scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders
GROUP BY time_of_day
ORDER BY order_count DESC
```

JOB INFORMATION | RESULTS | JSON | EXEC

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

Insights and Recommendations :-

This query categorizes the orders based on the hour of the day they were placed. It assigns labels('Dawn', 'Morning', 'Afternoon', 'Night') to each time period as per our definition. The result will include two columns- 'time_of_day" and "order_count" column representing the number of orders placed in each time period.

The output of the query will show the time of day('Dawn', 'Morning', 'Afternoon', 'Night') when Brazilian customers mostly placed their orders, along with the corresponding order counts. The result will be sorted in desc order of order counts, allowing you to identify the predominant time period.

3) Evolution of E-commerce orders in the Brazil region:

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

```sql
SELECT c.customer_state,
format_date('%y-%m',datetime(o.order_purchase_timestamp)) as year_month,
count(*) as no_of_orders from scaler-dsml-sql-390311.
Target_Brazil_Operations_2016_2018.orders o
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers
c on c.customer_id=o.customer_id
group by c.customer_state, year_month
order by no_of_orders
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | Expand

Row	customer_state	year_month	no_of_orders
1	SC	18-09	1
2	PI	18-10	1
3	RJ	18-10	1
4	PB	16-10	1
5	RR	18-05	1
6	AM	17-06	1
7	AP	17-07	1
8	TO	17-07	1
9	PR	16-12	1

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	year_month	no_of_orders
10	PI	16-10	1
11	RR	17-07	1
12	RR	17-09	1
13	RR	16-10	1
14	RR	16-09	1
15	RS	16-09	1
16	MS	17-01	1
17	AP	18-02	2
18	AP	18-08	2
19	AP	17-06	2
20	AP	18-06	2

Insights and Recommendations :-

i) The query allows us to observe the month-on-month variations in the number of orders placed in each state. You can identify states that consistently have higher or lower order volumes over time.
ii) By analyzing the number_of_orddr column, we can determine the fluctuations in order volume month by month for each state.This analysis can help to identify peak months, low demand periods or any seasonality in e-commerce orders.
Some Recommendations: i) Identify states with consistently high order volumes and focus on implementing targeted marketing strategies & campaigns to further increase customer engagement and capture market share in those regions.
ii)Analyze the monthly variations in order volumes to plan seasonal promotions and discounts. By aligning promotional activities with peak months or specific periods of high demand, we can maximize sales & customer acquisition.

By leveraging the insights gained from analyzing month-on-month numbers of orders placed in each state, you can make data driven decisions to drive growth, improve operational efficiency and enhance customer satisfaction in the Brazil e-commerce market.

2) How are the customers distributed across all the states?

```
SELECT  customer_state, count(distinct customer_id) as num_customer
FROM scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers
GROUP BY customer_state
ORDER BY num_customer
```

Row	customer_state	num_customer
1	RR	46
2	AP	68
3	AC	81
4	AM	148
5	RO	253
6	TO	280
7	SE	350
8	AL	413
9	RN	485
10	PI	495
11	PB	536

Insights and Recommendations :-

Insights:

- This query provides the number of distinct customers for each state in the given table.
- We can observe the customer distribution across different states in Brazil.
- We can identify states with a high number of customers & those with a low number of customers.

Recommendations:

- Focus on states with a low number of customers to explore market expansion opportunities and target those regions for marketing and customer acquisition efforts.
- Analyze states with a high number of customers to understand the factors contributing to their success and implement strategies to enhance customer retention and satisfaction.
- Conduct further market research on states with varying customer counts to understand local demographics, preferences, and competitor landscapes.
- Allocate resources strategically based on the customer distribution to maximize business opportunities in each state.

4) Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

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.

```sql
SELECT round((total_2018 - total_2017) / total_2017 * 100 ,4) AS
total_percent_increase
FROM (SELECT
SUM (CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2018 AND
EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8 THEN
p.payment_value ELSE 0 END) AS total_2018,
SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017 AND
EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8 THEN
p.payment_value ELSE 0 END) AS total_2017
FROM scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.payments  p
JOIN scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o ON p.order_id = o.order_id) subquery
```

JOB INFORMATION | RESULTS

Row	total_percent_increas
1	136.9769

Insights and Recommendations :-

Insights-

The query will return the 'total_percent_increase' which represents the percentage increase in the cost of orders from 2017-2018 for the specific months.

Recommendations-

- Analyze contributing factors- Identify the factors that led to the increase in order cost and analyze their impact on overall revenue and profitability.
- Access the impact of pricing changes during the period. Ensure your pricing aligns with customer expectations and conduct a price sensitivity analysis if necessary.
- It analyzes the products or categories that experienced the highest increase in order cost. Optimize cost without compromising quality and focus on high performance products.
- It improves the overall customer experience by delivering exceptional service, streamlining the ordering process, and ensuring timely product delivery.

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

```
select g.geolocation_state as state, round(sum(oi.price),2) as total_price
,round(avg(price),2) as avg_price from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
```

Row	state	total_price	avg_price
1	MT	22777072.82	156.63
2	MA	9020091.01	150.95
3	AL	7191886.1	196.64
4	SP	711838740.52	111.28
5	MG	397190155.87	121.18
6	PE	17545068.94	137.42
7	RJ	440142503.48	127.81
8	DF	13141649.62	124.66
9	RS	111183139.56	120.18
10	SE	3976184.44	146.11
11	PR	85392469.28	119.21

Insights and Recommendations :-

Insight: The query provides insights into the total price and average price of order items for different states. This allows you to analyze sales performance and price trends across various regions.

Recommendations:
This query will join the necessary tables(order_items,orders,customers, and geolocation) and calculate the total and average order price for each state by grouping the results based on the state column(geolocation_state). The sum() function calculates the total price by summing the 'price' column from the 'order_items' table, and the AVG() function calculates the average price. The ROUND() function is used to round the values to two decimal places for better readability.

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

```
select g.geolocation_state as state,round(sum(oi.freight_value),2) as total_freight
,round(avg(freight_value),2) as avg_freight
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders  o
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
```

Row	state	total_freight	avg_freight
1	SP	98574572.43	15.41
2	RJ	71966793.75	20.9
3	PR	14432159.77	20.15
4	SC	13472314.62	21.55
5	DF	2214955.55	21.01
6	MG	67058347.09	20.46
7	PA	3409472.09	36.53
8	BA	11345094.0	27.22
9	GO	3590268.56	23.17
10	RS	19910834.35	21.52
11	TO	743027.53	37.36

Insights and Recommendations :-

Insights-

- The query calculates the total freight value for each state by summing up the freight values of all order items associated with orders from that state. This information provides insights into the overall freight costs incurred by customers in different states. It helps identify states with higher or lower total freight costs.
- The query also calculates the average freight value for each state by taking the average of the freight values of all order items associated with orders from that state. This metric indicates the average freight cost per order for customers in different states. It helps identify states with higher or lower average freight costs.

Recommendations-

- To improve the performance of the query, we can consider adding appropriate indexes on the columns used for joining the tables. This can speed up the query execution and improve overall performance.
- Data Validation:Ensure that the tables 'order_items', 'orders', 'customers' and 'geolocation' exist in the database and have the correct structure. Also, verify that the join conditions and column names are accurate.
- The query rounds the 'total_freight, and 'avg_freight' values to two decimal places. Make sure this level of precision is sufficient for the intended use of the results. Adjust the rounding as needed.
- If you want to analyze a specific time period or a subset of data, you can add appropriate date filters to limit the data retrieved from the "orders" table. This can be done by adding a 'WHERE' clause to the query and specifying the desired date range or any other relevant conditions.
- The query groups the results by the 'state' column, indicating that the output will provide freight information for each state. Ensure that the 'geolocation_state column in the 'geolocation' table contains the correct state names and that it aligns with the expected output.
- Once the query has been executed successfully, analyze the results to gain insights into the total and average freight values for each state in Brazil. Identify states with high and low freight costs and compare them to historical data or other relevant metrics. This analysis can help identify patterns, optimize logistics, and make informed business decisions

Q.5 Analysis based on sales, freight and delivery time.

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.
   You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
   - **time_to_deliver = order_delivered_customer_date - order_purchase_timestamp**
   - **diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date**

```
Select order_id, date_diff(order_delivered_customer_date,
order_purchase_timestamp,day) as time_to_deliver,
date_diff(order_estimated_delivery_date, order_delivered_customer_date,day)
as diff_estimated_delivery
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders
```

Row	order_id	time_to_deliver	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

Insights and Recommendations :-

Time to Deliver: The column "time_to_deliver" represents the number of days it took for an order to be delivered to the customer. This can provide insights into the efficiency of the delivery process. By analyzing this data, you can identify trends, patterns, and potential bottlenecks in the delivery system. For example, you can track the average delivery time over time and compare it to service level agreements (SLAs) or industry benchmarks to evaluate performance.

Difference in Estimated Delivery: The column "diff_estimated_delivery" represents the difference in days between the estimated delivery date and the actual delivered date. This can help assess the accuracy of the estimated delivery dates provided to customers. By analyzing this difference, you can evaluate the reliability of your estimated delivery estimates.

Recommendation:

- Monitor the time to deliver metric regularly and set targets for improvement. Identify any outliers or delays and investigate the reasons behind them. Consider implementing measures like optimizing logistics, improving coordination with delivery partners, or addressing any operational issues that may contribute to longer delivery times.
- Track the difference in estimated delivery dates and identify any significant gaps between the estimated and actual delivery times. If there is a consistent delay, it may indicate a need to adjust the estimation process or communicate more realistic delivery expectations to customers. Consider factors like transportation logistics, order processing time, and any external factors affecting the delivery timeline.

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

For Top 5 states with the highest average freight value:-

```sql
select g.geolocation_state as state ,round(avg(freight_value),3) as avg_highest_value
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers  c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation  g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
having state is not null
order by avg_highest_value desc
```

Row	state	avg_highest_value
1	PB	42.773
2	RR	42.47
3	PI	39.477
4	AC	39.098
5	MA	38.075
6	RO	37.429
7	TO	37.361
8	PA	36.527
9	AP	35.655
10	SE	34.673
11	AM	34.622

The top 5 states with the lowest average freight value:-

```sql
select g.geolocation_state as state ,round(avg(freight_value),3) as avg_lowest_value
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
having state is not null
order by avg_lowest_value asc
limit 5
```

Row	state	avg_lowest_value
1	SP	15.41
2	PR	20.148
3	MG	20.459
4	RJ	20.898
5	DF	21.011

Insights and Recommendations :-

High-Freight States: The top 5 states with the highest average freight value indicate that customers in these regions may be located farther away from distribution centers or face logistical challenges, resulting in higher shipping costs. To optimize freight expenses in these states, consider the following recommendations:

- Explore partnerships with local logistics providers or warehouses to reduce transportation costs.
- Analyze the possibility of establishing regional distribution centers to streamline operations and reduce delivery distances.
- Evaluate pricing strategies and consider incorporating shipping costs into product pricing to manage customer expectations.

Low-Freight States: It would be beneficial to analyze the states with the lowest average freight value to identify potential cost-saving opportunities. These states may have more accessible transportation networks or closer proximity to distribution centers. Consider the following recommendations:

- Analyze the supply chain and distribution network in low-freight states to identify best practices that can be implemented in high-freight states.
- Consider expanding marketing efforts in these states to capitalize on the lower shipping costs and attract more customers.
- Explore opportunities to consolidate shipments and optimize routes in low-freight states to further reduce transportation expenses.

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

The top 5 states with the highest average delivery time:-

```
select g.geolocation_state as state,
round(avg(date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)),3) as time_to_deliver_highest
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
order by time_to_deliver_highest desc
limit 5
```

Row	state	time_to_deliver_high
1	AP	30.405
2	AM	24.38
3	RR	23.982
4	AL	22.87
5	PA	22.733

The top 5 states with the lowest average delivery time**:-**

```sql
select g.geolocation_state as state ,
round(avg(date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)),3) as time_to_deliver_lowest
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
order by time_to_deliver_lowest
limit 5
```

Row	state	time_to_deliver_lowe
1	SP	8.442
2	PR	10.998
3	MG	11.357
4	DF	12.437
5	RJ	14.395

Insights and Recommendations :-

Insights and Recommendations for Query 1:

- The query selects the geolocation state, 'g.geolocation_state' , and calculates the average time it takes to deliver an order, 'round(avg(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp, day)), as time_to_deliver_highest.
- By grouping the results by state and ordering them in descending order by the average delivery time, the query retrieves the top 5 states with the longest average delivery times.
- This information can be useful for identifying regions or states where the delivery process may need improvement. It can help identify potential bottlenecks or inefficiencies in the delivery process.
- Based on the results, you can focus on these states and investigate the reasons behind the longer delivery times. Possible factors could include logistic challenges, lack of delivery infrastructure, or issues with shipping carriers.
- Once the issues are identified, appropriate measures can be taken to improve the delivery time in these states. This may involve optimizing the logistics network, partnering with reliable shipping companies, or streamlining the order fulfillment process.

Insights and Recommendations for Query 2:

- Similar to the first query, this query selects the geolocation state, 'g.geolocation_state' , and calculates the average time it takes to deliver an order, 'round(avg(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp, day)), as time_to_deliver_lowest.
- The query groups the results by state and orders them in ascending order by the average delivery time, retrieving the top 5 states with the shortest average delivery times.
- These results can provide insights into regions or states where the delivery process is efficient and successful. It can help identify best practices and successful strategies for faster order delivery.

- You can analyze the factors contributing to the shorter delivery times in these states and replicate those practices in other regions. This may involve studying the logistics infrastructure, delivery networks, or operational strategies in these states and implementing them in areas with longer delivery times.
- Furthermore, we can compare the performance of different states and use the insights to establish benchmarks and set goals for improving delivery times across the board.

Overall, by analyzing the results from both queries, we can identify areas for improvement in terms of delivery efficiency and implement strategies to optimize the delivery process in different states.

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.

The top 5 states where the order delivery is fast:-

```
select g.geolocation_state as state ,
round(avg(date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)),3) as estimated_top_delivery
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
order by estimated_top_delivery desc
limit 5
```

Row	state	estimated_top_delive
1	RR	20.882
2	AM	20.561
3	RO	19.104
4	AC	18.564
5	AP	15.648

The top 5 states where the order delivery is Slow:-

```
select g.geolocation_state as state
,round(avg(date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)),3) as estimated_slow_delivery
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.order_items oi
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on oi.order_id=o.order_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.customers c
on c.customer_id=o.customer_id
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.geolocation g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
group by state
order by estimated_slow_delivery
limit 5
```

Row	state	estimated_slow_delivery
1	AL	8.457
2	SE	8.743
3	MA	9.017
4	CE	10.016
5	ES	10.089

Insights and Recommendations :-

Insights and Recommendations for Query 1:

The query calculates the average estimated delivery time for orders in each state and ranks them based on the estimated top delivery. The top 5 states with the fastest estimated delivery times are returned.

- The query considers the 'order_items', 'orders', 'customers', and 'geolocation tables to gather the necessary data.
- The 'geolocation_state' column from the 'geolocation' table is used to determine the state.
- The 'date_diff' function calculates the difference between the estimated delivery date and the delivered customer date in days.
- The average delivery time is rounded to 3 decimal places using the 'round' function.
- The results are grouped by the state column and ordered in descending order based on the estimated top delivery time.

Recommendation:

- Consider adding appropriate indexes to the relevant columns used in the join operations ('order_id', 'customer_id', 'customer_zip_code_prefix) to improve query performance.
- Monitor the performance of the query execution and optimize it further if required, especially if dealing with large datasets.
- Ensure that the('order_estimated_delivery_date') and ('order_delivered_customer_date') columns have the correct data type to perform the date difference calculation accurately.
- Regularly update the 'geolocation' table with the latest data to maintain accurate state information.

Insights and Recommendations for Query 2:

The query calculates the average estimated delivery time for orders in each state and ranks them based on the estimated slow delivery. The top 5 states with the slowest estimated delivery times are returned.

- The query structure and data sources are the same as in the 1st query.
- The 'estimated_slow_delivery' column is calculated using the 'round' and 'date_diff' functions.
- The results are grouped by the state column and ordered in ascending order based on the estimated slow delivery.

Recommendation-

- Consider adding appropriate indexes to the relevant columns used in the join operations ('order_id', 'customer_id', 'customer_zip_code_prefix) to improve query performance.
- Compare the results of this query with the first query to identify states that have both the fastest and slowest estimated delivery times.

- Analyze the underlying factors that contribute to slow delivery times in the identified states and take appropriate measures to improve the delivery process in those areas.
- Monitor the impact of any interventions or improvements made based on the analysis and adjust strategies accordingly.

6) Analysis based on the payments:

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

```
select distinct format_date('%Y-%m', datetime(o.order_purchase_timestamp)) as
year_month, p.payment_type
,count(distinct o.order_id) as no_of_orders
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.payments p
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on p.order_id=o.order_id
group by year_month,p.payment_type
order by year_month
```

Row	year_month	payment_type	no_of_orders
1	2016-09	credit_card	3
2	2016-10	credit_card	253
3	2016-10	voucher	11
4	2016-10	debit_card	2
5	2016-10	UPI	63
6	2016-12	credit_card	1
7	2017-01	voucher	33
8	2017-01	UPI	197
9	2017-01	credit_card	582
10	2017-01	debit_card	9

Insights and Recommendations :-

- By examining the year_month column in the results, you can identify trends and patterns in the number of orders over time. This information can be useful for understanding the seasonal variations or identifying any significant changes in customer behavior.
- The payment_type column provides insights into the different payment methods used by customers. You can analyze the distribution of payment types over time to identify any shifts in customer preferences or the effectiveness of specific payment options.
- By reviewing the no_of_orders column, you can assess the performance of different payment types across various time periods. This analysis can help identify which payment methods are more popular or successful in driving sales.

Recommendation-

- Promote the use of credit cards: Since credit cards are the most preferred payment option, consider offering special incentives or discounts to customers who use credit cards for making purchases. This can help increase customer engagement and loyalty.
- Enhance UPI payment experience: As UPI is gaining popularity, ensure that the payment process is seamless and user-friendly. Provide clear instructions on how to use UPI for payment and address any potential issues that customers may face.

- Explore partnerships with voucher providers: Since voucher payments have shown some usage, consider collaborating with voucher providers or offering exclusive deals for customers who use vouchers as a payment method. This can attract more customers and increase sales.

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

```
select distinct format_date('%Y-%m', datetime(o.order_purchase_timestamp)) as
year_month, p.payment_installments
,count(distinct o.order_id) as no_of_orders
from scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.payments p
join scaler-dsml-sql-390311.Target_Brazil_Operations_2016_2018.orders o
on p.order_id=o.order_id
group by year_month,p.payment_installments
order by year_month
```

Row	year_month	payment_installment	no_of_orders
1	2016-09	1	1
2	2016-09	2	1
3	2016-09	3	1
4	2016-10	1	127
5	2016-10	2	30
6	2016-10	3	43
7	2016-10	4	26
8	2016-10	5	20
9	2016-10	6	18
10	2016-10	7	13
11	2016-10	8	3
12	2016-10	9	3

Insights and Recommendations :-

Insights-

- The query uses the 'payment' tables and the 'orders'
- It calculates the number of orders('no_of_orders') for each combination of 'year_month' (formatted as %y-%m) and 'payment_installments'.
- The 'distinct' keyword is used to count unique values, ensuring that each order is counted only once.
- The result is ordered by 'year_month'.

Recommendation-

- Identify the payment installments that are most frequently chosen by customers. Consider promoting these installment options to attract more customers and increase order volume.
- Identify trends over time: Analyze the number of orders placed over different months and years (year_month) Look for any patterns or trends to understand how the business is performing over time. This information can be valuable for forecasting and planning purposes.

- Investigate the impact of payment installments on order volume: Compare the number of orders across different payment installments. Determine whether customers tend to place more orders with a specific number of payment installments. This can help optimize pricing strategies or payment options to attract more customers.
- Monitor changes in order volume: Track changes in the number of orders over time to identify any significant fluctuations. Sudden increases or decreases in order volume may indicate the effectiveness of marketing campaigns, changes in customer behavior, or external factors affecting the business.