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

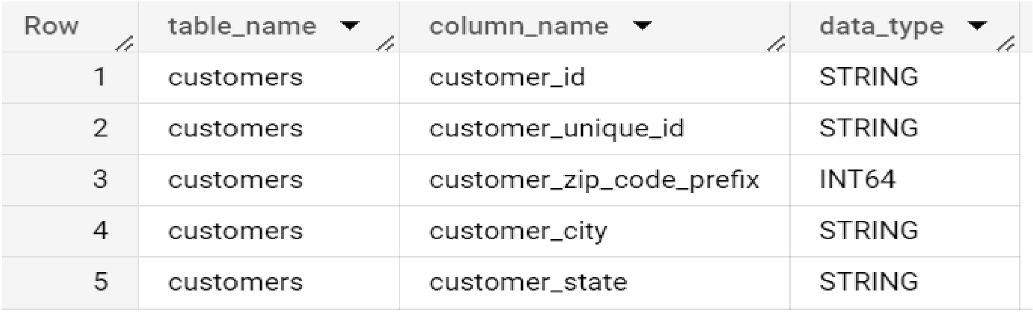
Q1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

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

SQL QUERY:

```
SELECT
  table_name,
  column_name,
  data_type
FROM
`my-first-sql-388003.my_project.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name='customers'
```

Output:



Row	table_name	column_name	data_type
1	customers	customer_id	STRING
2	customers	customer_unique_id	STRING
3	customers	customer_zip_code_prefix	INT64
4	customers	customer_city	STRING
5	customers	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.

Similarly, to get the details of the other 7 tables the following query was used

```
SELECT table_name, column_name, data_type FROM
my-first-sql-388003.my_project.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = ’table_name’
```

Insights and Recommendations :

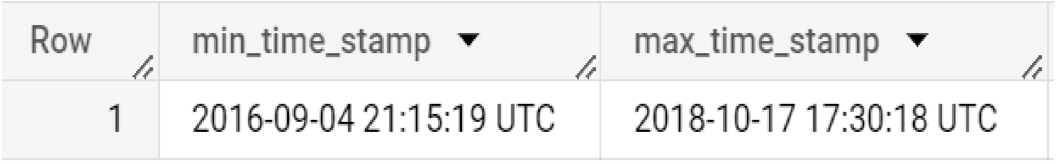
- We have 99,441 customers with data available.
- We have 96096 numbers of Unique Customers ids.
- 14994 different locations of customers
- Customers are from 8126 cities and 27 states from Brazil.
- total 99441 customers are there in given data.

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

SQL QUERY:

```
SELECT
  MIN(order_purchase_timestamp) AS min_time_stamp,
  MAX(order_purchase_timestamp) AS max_time_stamp
FROM
  `my_project.orders`
```

Output:



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

Insights and Recommendations :

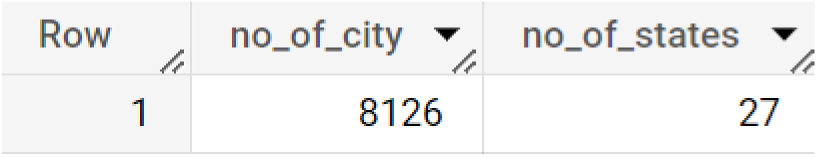
- Here the first order was placed on **2016-09-04** at **21:15:19** and last order was placed on 2018-10-17 at 17:30:18
- Time range: This result of this query provides the earliest **(min_time_stamp)** and latest **(max_time_stamp)** purchase timestamp from the order table. This gives you the overall time range during which the orders were placed.
- 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.

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

SQL QUERY:

```
WITH
 cte AS (
 SELECT
  customer_city AS city,
  customer_state AS state
 FROM
  `my_project.customers`
 UNION ALL
 SELECT
  seller_city AS city,
  seller_state AS state
 FROM
  `my_project.sellers`
 UNION ALL
 SELECT
  geolocation_city AS city,
  geolocation_state AS state
 FROM
  `my_project.geolocation` )
SELECT
 COUNT(DISTINCT city) AS no_of_city,
 COUNT(DISTINCT state) AS no_of_states
FROM
 cte
```

Output:



Row	no_of_city	no_of_states
1	8126	27

Insights and Recommendations :

- Here in the given data set total no of city 8126 are present and total no of states 27.
- In this, geolocation, customers and sellers by combining all tables 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 ‘**no_of_city**’ and ‘**no_of_states**’ representing the counts of unique cities & states respectively.

Q2. In-depth Exploration

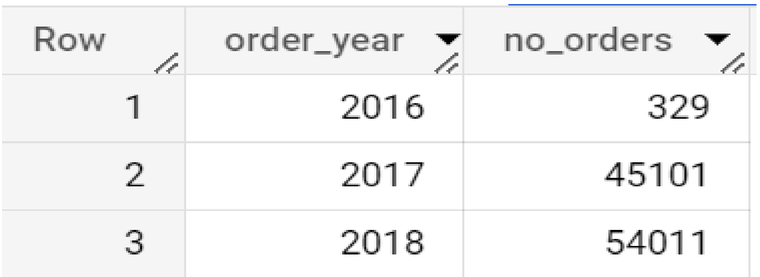
1) Is there a growing trend in the no. of orders placed over the past years?

SQL QUERY:

```
SELECT
 EXTRACT(year
 FROM
  order_purchase_timestamp) order_year,
 COUNT(*) no_orders
```

```
FROM
  my_project.orders
GROUP BY
  order_year
ORDER BY
  no_orders
```

Output:



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

Insights and Recommendations :

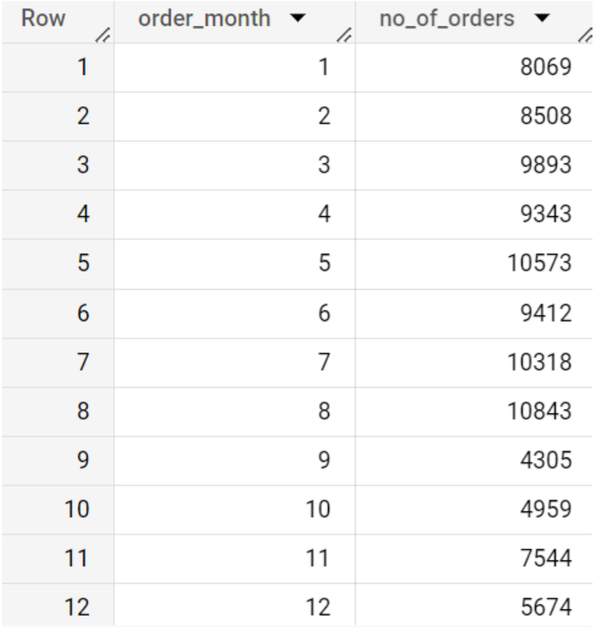
From the analysis of the dataset, it is observed that the number of orders placed has been increasing year by year. This indicates a positive trend in customer engagement and business growth over time.

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

SQL QUERY:

```
SELECT
  EXTRACT(month
  FROM
    order_purchase_timestamp) order_month,
  COUNT(*) no_of_orders
FROM
  `my_project.orders`
GROUP BY
  order_month
ORDER BY
  order_month
```

Output:



Row	order_month	no_of_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
11	11	7544
12	12	5674

Insights and Recommendations:

- **Monthly Order Volumes**: The analysis reveals fluctuating order volumes throughout the year, ranging from 4,305 to 10,873 orders per month.
- **Seasonal Patterns**: There is a noticeable increase in order volumes during certain months, such as February, April, May, and August, while other months show relatively lower volumes
- **Resource Allocation**: Allocate resources based on the identified seasonal patterns. Increase staffing, inventory levels, and marketing efforts during high-demand months to meet customer demands effectively.
- **Promotions and Campaigns**: Plan targeted promotions and marketing campaigns during peak months or periods of lower demand to stimulate sales and drive customer engagement.

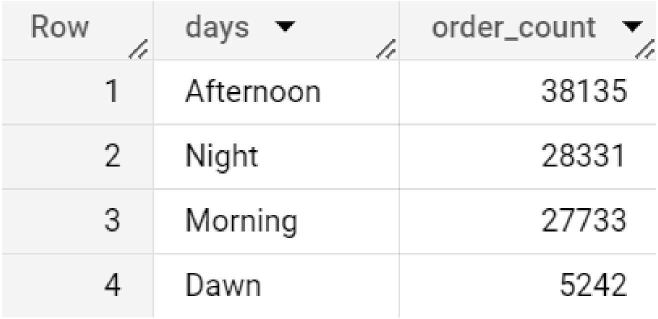
3) During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- **a) 0-6 hrs : Dawn**
- **b) 7-12 hrs : Mornings**
- **c) 13-18 hrs : Afternoon**
- **d) 19-23 hrs : Night**

SQL QUERY:

```
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 days,
 COUNT(*) as order_count
FROM
 `my_project.orders`
GROUP BY days
ORDER BY order_count  DESC
```

Output:



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

Insights and Recommendations:

- **Peak Order Times**: The data indicates that customers in Brazil tend to place orders predominantly in the afternoon and night, with a significant number of orders also being placed during the morning.
- **Staffing Optimization**: Based on the peak order times, allocate appropriate staffing resources to handle customer inquiries, process orders, and provide support during these periods of high customer activity. This ensures efficient customer service and reduces response times.
- **Marketing and Promotions**: Schedule marketing campaigns and promotions to coincide with the peak order times. By targeting customers during these periods, you can maximize their engagement and conversion rates.

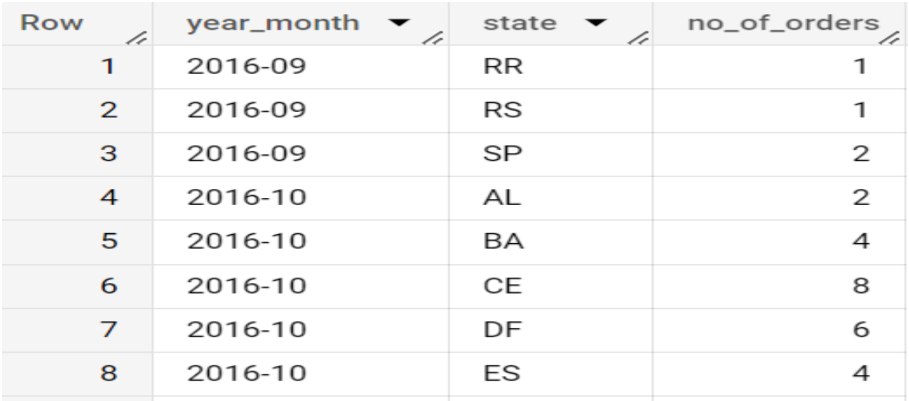
Q3) Evolution of E-commerce orders in the Brazil region:

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

SQL QUERY:

```
SELECT
  FORMAT_DATE('%Y-%m', DATETIME(o.order_purchase_timestamp ))
year_month,
  c.customer_state AS state,
  COUNT(*)  AS no_of_orders
FROM
  my_project.customers c
JOIN
  my_project.orders o
ON
  o.customer_id=c.customer_id
GROUP BY
  state,
  year_month
ORDER BY
  year_month,
  state
```

Output:



Row	year_month	state	no_of_orders
1	2016-09	RR	1
2	2016-09	RS	1
3	2016-09	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

Insights and Recommendations:

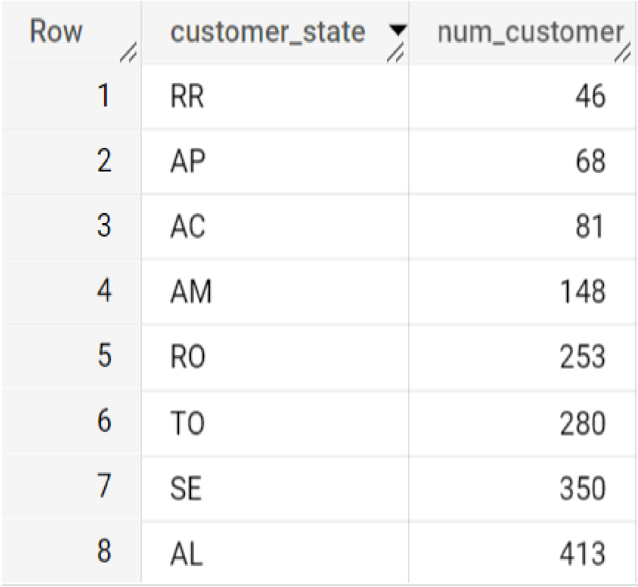
- **Order Volume by State**: Analyze month-on-month variations in the number of orders placed in each state to identify regions with consistently high or low order volumes.
- **Seasonality and Trends**: Identify peak months, low-demand periods, and seasonal patterns in e-commerce orders to optimize marketing and promotional activities.
- **Targeted Marketing Strategies:** Implement targeted marketing strategies and campaigns in states with high order volumes to increase customer engagement and capture a larger market share.
- **Seasonal Promotions**: Plan seasonal promotions and discounts aligned with peak months or periods of high demand to maximize sales and customer acquisition.

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

SQL QUERY:

```
SELECT
  customer_state,
  COUNT(DISTINCT customer_id) AS num_customer
FROM
  `my_project.customers`
GROUP BY
  customer_state
ORDER BY
  num_customer
```

Output:



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

Insights and Recommendations:

- 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.
- 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.

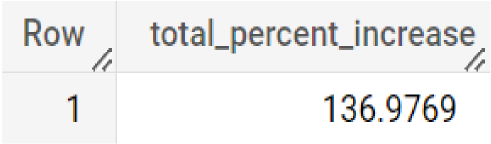
Q4) Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

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

SQL QUERY:

```
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)=2017
AND EXTRACT(month FROM o.order_purchase_timestamp) BETWEEN 1
AND 8 THEN p.payment_value ELSE 0 END) AS total_2017,
    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
  FROM
    `my_project.payments` p
  JOIN
    `my_project.orders` o
  ON
    p.order_id=o.order_id) x
```

Output:



Row	total_percent_increase
1	136.9769

Insights and Recommendations :

- The query will return the **‘total_percent_increase**’ which represents the percentage increase in the cost of orders 2017 and 2018 for the specific months.
- 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.

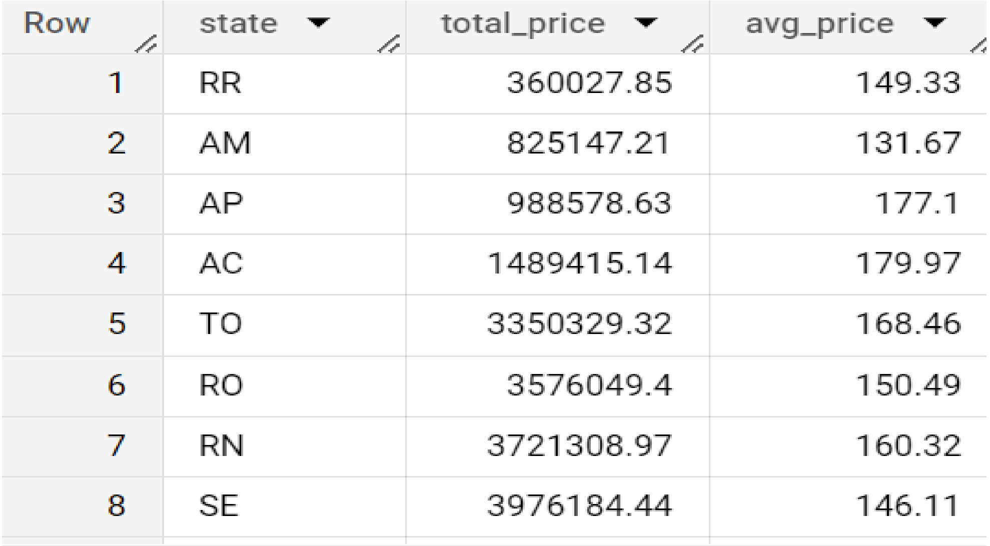
2) Calculate the Total & Average value of order price for each state.

SQL QUERY:

```
SELECT
  c.customer_state AS state,
  ROUND(SUM(oi.price),2) AS total_price,
  ROUND(AVG(oi.price),2) AS avg_price
FROM
  `my_project.order_items` oi
JOIN
  `my_project.orders` o
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
```

```
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  total_price
```

Output:



Row	state	total_price	avg_price
1	RR	360027.85	149.33
2	AM	825147.21	131.67
3	AP	988578.63	177.1
4	AC	1489415.14	179.97
5	TO	3350329.32	168.46
6	RO	3576049.4	150.49
7	RN	3721308.97	160.32
8	SE	3976184.44	146.11

Insights and Recommendations :

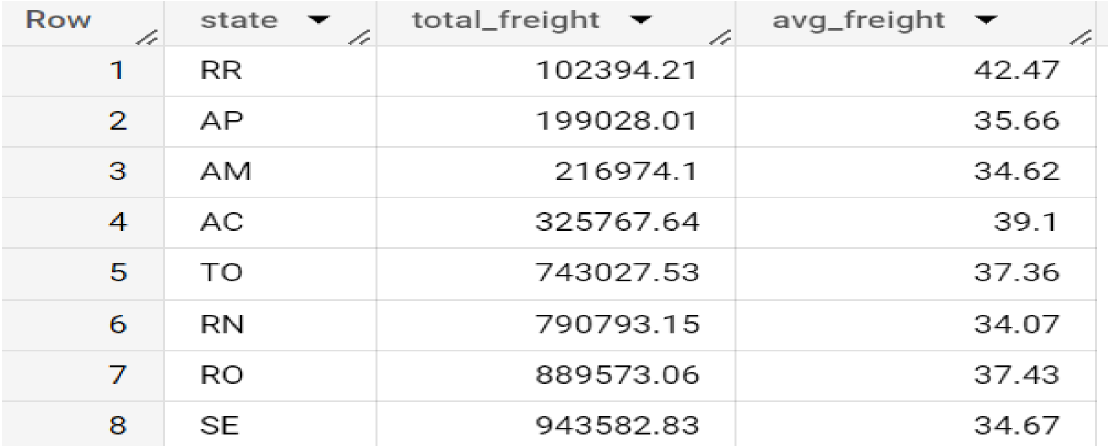
- The query provides insights into the total price and average price of order items for different states. This allows you to analyze sales performance and price trends across various regions
- Here the state with RR has the lowest total and the state with SP has the highest total.

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

SQL QUERY:

```
SELECT
  g.geolocation_state AS state,
  ROUND(SUM(oi.freight_value),2) AS total_freight,
  ROUND(AVG(freight_value),2) AS avg_freight
FROM
  `my_project.order_items` oi
JOIN
  `my_project.orders` o
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  total_freight
```

Output:



Row	state	total_freight	avg_freight
1	RR	102394.21	42.47
2	AP	199028.01	35.66
3	AM	216974.1	34.62
4	AC	325767.64	39.1
5	TO	743027.53	37.36
6	RN	790793.15	34.07
7	RO	889573.06	37.43
8	SE	943582.83	34.67

Insights and Recommendations :

- The query provides insights into the total freight and average freight of order items for different states. This allows you to analyze sales performance and price trends across various regions
- Here the state with RR has the lowest total freight and the state with SP has the highest total freight.

Q5) 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.
Do this in a single query.

You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

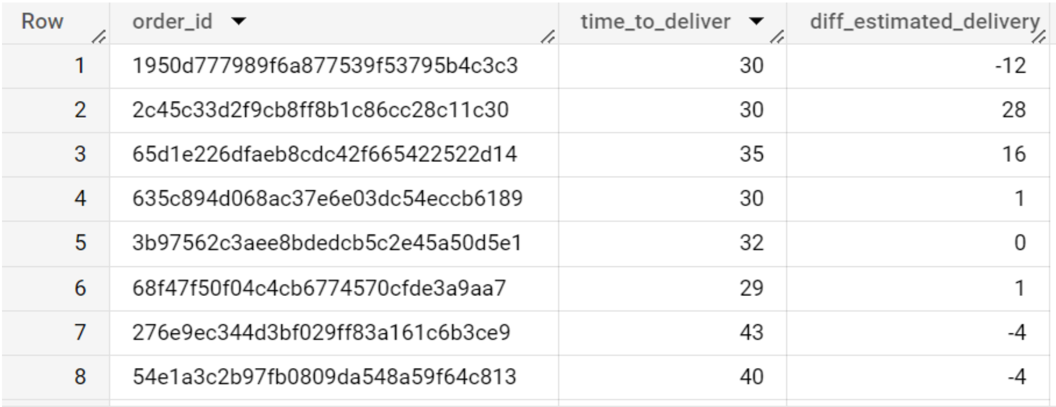
a) time_to_deliver = order_delivered_customer_date - order_purchase_timestamp

b) diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date

SQL QUERY:

```
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
  `my_project.orders`
```

Output:



Row	order_id	time_to_deliver	diff_estimated_delivery
1	1950d777989f6a877539f53795b4c3c3	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28c11c30	30	28
3	65d1e226dfaeb8cdc42f665422522d14	35	16
4	635c894d068ac37e6e03dc54eccb6189	30	1
5	3b97562c3aee8bdedcb5c2e45a50d5e1	32	0
6	68f47f50f04c4cb6774570cfde3a9aa7	29	1
7	276e9ec344d3bf029ff83a161c6b3ce9	43	-4
8	54e1a3c2b97fb0809da548a59f64c813	40	-4

Insights and Recommendations:

- **Time to Deliver**: Analyze the "time_to_deliver" column to assess the efficiency of the delivery process. Monitor the average delivery time, identify any bottlenecks, and strive for continuous improvement by optimizing logistics and addressing operational issues.
- **Difference in Estimated Delivery:** Evaluate the "diff_estimated_delivery" column to measure the accuracy of estimated delivery dates provided to customers. Identify any significant gaps between estimated and actual delivery times, and refine the estimation process or communicate more realistic delivery expectations to enhance customer satisfaction.
- **Operational Efficiency:** Regularly monitor and analyze delivery metrics to ensure timely deliveries. Optimize coordination with delivery partners, streamline order processing, and address any delays or outliers to maintain high service levels and meet customer expectations.

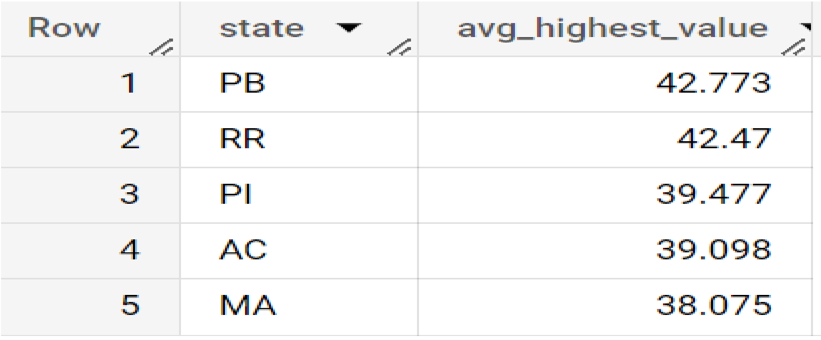
2) Find out the top 5 states with the highest & lowest average freight value.

SQL QUERY:

For the top 5 states highest average freight values.

```
SELECT
  g.geolocation_state AS state,
  ROUND(AVG(freight_value),3) AS avg_highest_value
FROM
  `my_project.order_items` oi
JOIN
  `my_project.orders` o
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  avg_highest_value DESC
LIMIT 5
```

Output:

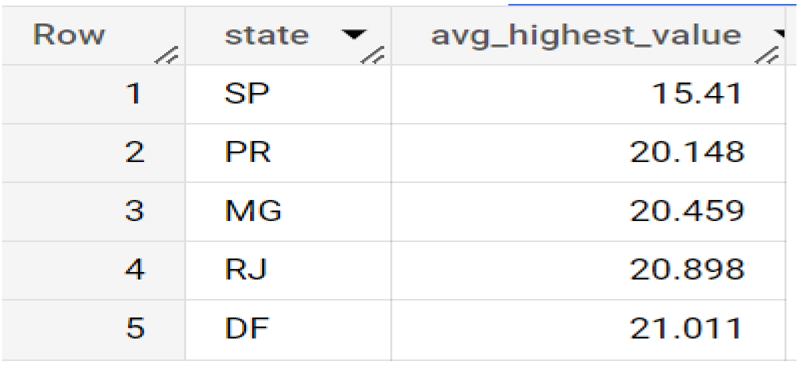


Row	state	avg_highest_value
1	PB	42.773
2	RR	42.47
3	PI	39.477
4	AC	39.098
5	MA	38.075

SQL QUERY:
For the top 5 states with the lowest average freight value:

```
SELECT
  g.geolocation_state AS state,
  ROUND(AVG(freight_value),3) AS avg_highest_value
FROM
  `my_project.order_items` oi
JOIN
  `my_project.orders` o
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  avg_highest_value
LIMIT 5
```

Output:



Row	state	avg_highest_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**:
  Local Partnerships: Explore partnerships with local logistics providers or warehouses in high-freight states to reduce transportation costs and improve delivery efficiency.
  Regional Distribution Centers: Analyze the feasibility of establishing regional distribution centers in high-freight states to streamline operations, reduce delivery distances, and minimize shipping expenses.
- **Low-Freight States**:
  Supply Chain Best Practices: Analyze the supply chain and distribution network in low-freight states to identify best practices that can be implemented in high-freight states, such as efficient routing, optimized inventory management, and effective transportation strategies.
  Market Expansion: Capitalize on the lower shipping costs in low-freight states by expanding marketing efforts and targeting customers in these regions. Utilize localized campaigns and promotions to attract more customers and drive sales

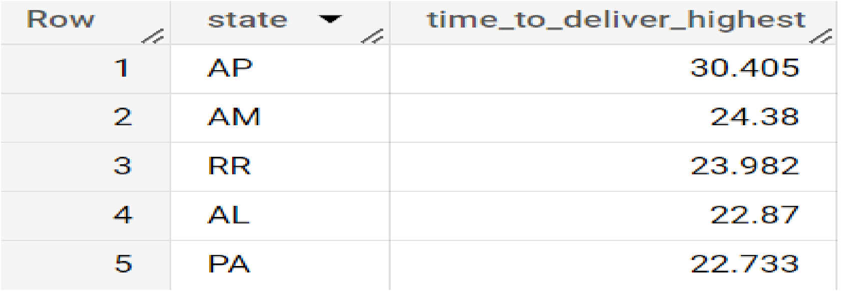
3) Find out the top 5 states with the highest & lowest average delivery time.

SQL Query:

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
  `my_project.orders` o
JOIN
  `my_project.order_items` oi
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.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
```

Output:

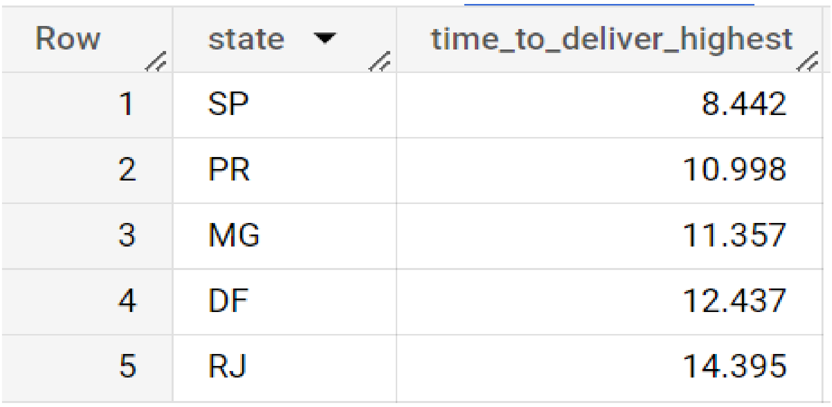


Row	state	time_to_deliver_highest
1	AP	30.405
2	AM	24.38
3	RR	23.982
4	AL	22.87
5	PA	22.733

SQL QUERY:
The top 5 states with the lowest 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
  `my_project.orders` o
JOIN
  `my_project.order_items` oi
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  time_to_deliver_highest
LIMIT 5
```

Output:



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

Insights and Recommendations:

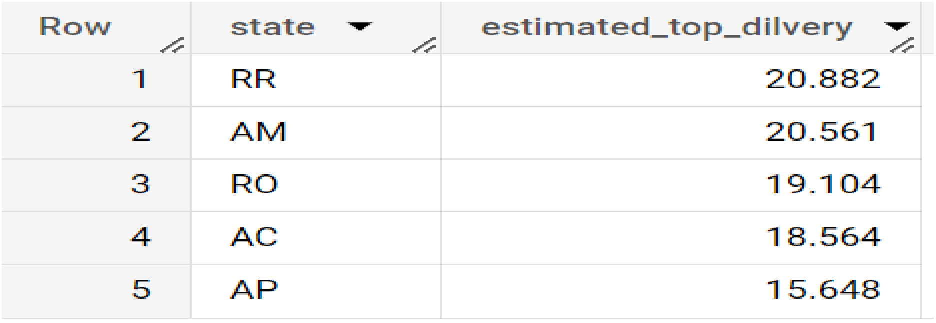
- **High Average States for delivery time**:
  Based on the results, you can focus on these states and investigate the reasons behind the longer delivery times. Possible factors could include logistic challenges, lack of delivery infrastructure, or issues with shipping carriers.
  Once the issues are 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.

- **Low Average States for delivery time.**
  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.

4) **Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.**
SQL QUERY:
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_dilvery
FROM
  `my_project.order_items` oi
JOIN
  `my_project.orders` o
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  estimated_top_dilvery DESC
LIMIT 5
```

Output:

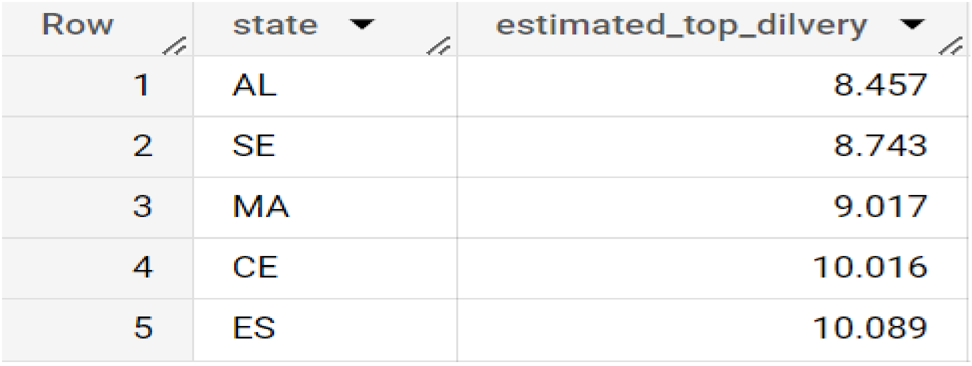


Row	state	estimated_top_dilvery
1	RR	20.882
2	AM	20.561
3	RO	19.104
4	AC	18.564
5	AP	15.648

SQL QUERY:
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_top_dilvery
FROM
  `my_project.order_items` oi
JOIN
  `my_project.orders` o
ON
  oi.order_id=o.order_id
JOIN
  `my_project.customers` c
ON
  c.customer_id=o.customer_id
JOIN
  `my_project.geolocation` g
ON
  c.customer_zip_code_prefix=g.geolocation_zip_code_prefix
GROUP BY
  state
ORDER BY
  estimated_top_dilvery
LIMIT 5
```

Output:



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

Insights and Recommendations:

The top 5 states where the order delivery is fast:

1. Identify states with the fastest estimated delivery times: The query calculates the average estimated delivery time for orders in each state and retrieves the top 5 states with the fastest estimated delivery times.
2. Monitor query performance: Consider adding appropriate indexes to the relevant columns used in the join operations to optimize query performance, especially if dealing with large datasets.

The top 5 states where the order delivery is slow:

1. Identify states with the slowest estimated delivery times: The query calculates the average estimated delivery time for orders in each state and retrieves the top 5 states with the slowest estimated delivery times.
2. Analyze underlying factors: Compare the results of this query with the first query to identify states that appear in both lists. Analyze the underlying factors contributing to slow delivery times in these states and take appropriate measures to improve the delivery process.

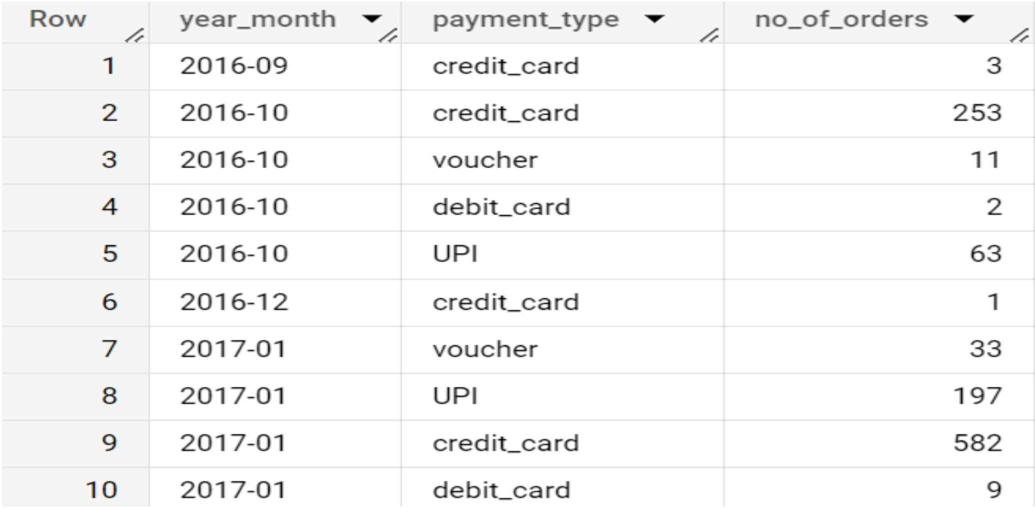
Q6) Analysis based on the payments.

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

SQL QUERY:

```
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
  my_project.payments p
JOIN
  `my_project.orders` o
ON
  p.order_id=o.order_id
GROUP BY
  year_month,
  p.payment_type
ORDER BY
  year_month
```

Output:



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:

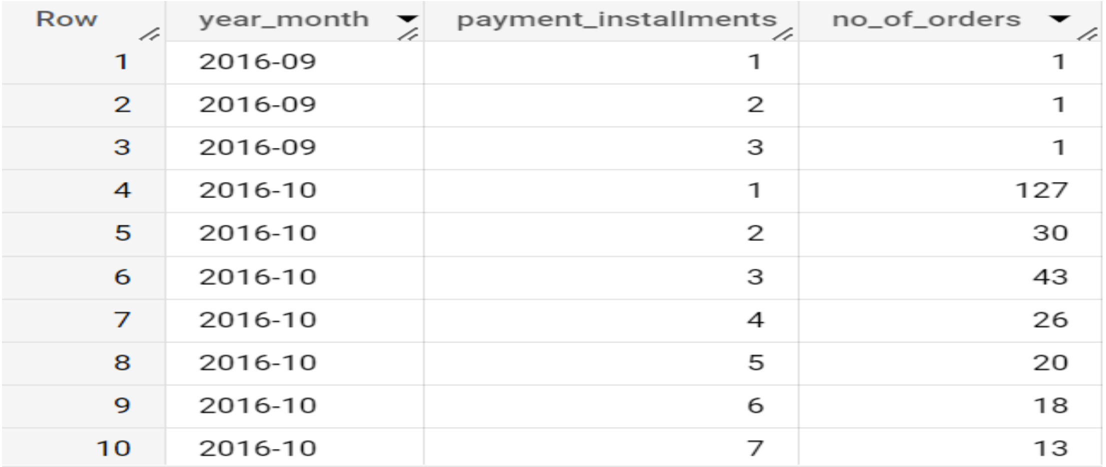
- By examining the year_month column in the results, you can identify trends and patterns in the number of orders over time. This information can be useful for understanding the seasonal variations or identifying any significant changes in customer behavior.
- The payment_type column provides insights 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.
- 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.

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

SQL QUERY:

```
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
  my_project.payments p
JOIN
  `my_project.orders` o
ON
  p.order_id=o.order_id
GROUP BY
  year_month,
  p.payment_installments
ORDER BY
  year_month
```

Output:



Row	year_month	payment_installments	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

Insights and Recommendations:

- 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’.
- 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.