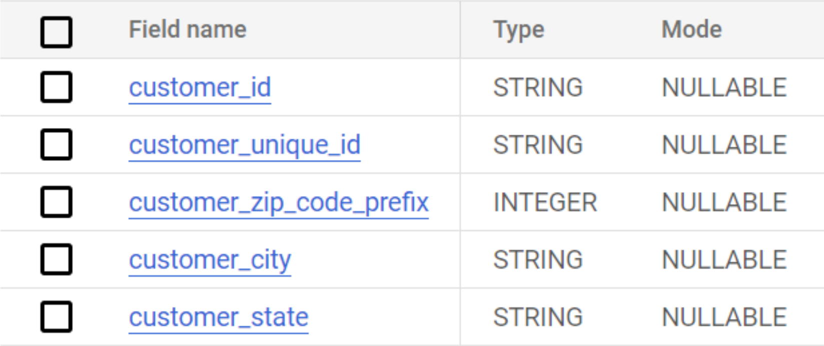
Business Case: Target SQL

Context:
Target is a globally renowned brand and a prominent retailer in the United States. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver.
This particular business case focuses on the operations of Target in Brazil and provides insightful information about 100,000 orders placed between 2016 and 2018. The dataset offers a comprehensive view of various dimensions including the order status, price, payment and freight performance, customer location, product attributes, and customer reviews.
By analysing this extensive dataset, it becomes possible to gain valuable insights into Target's operations in Brazil. The information can shed light on various aspects of the business, such as order processing, pricing strategies, payment and shipping efficiency, customer demographics, product characteristics, and customer satisfaction levels.

Problem Statement:
Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

1.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**

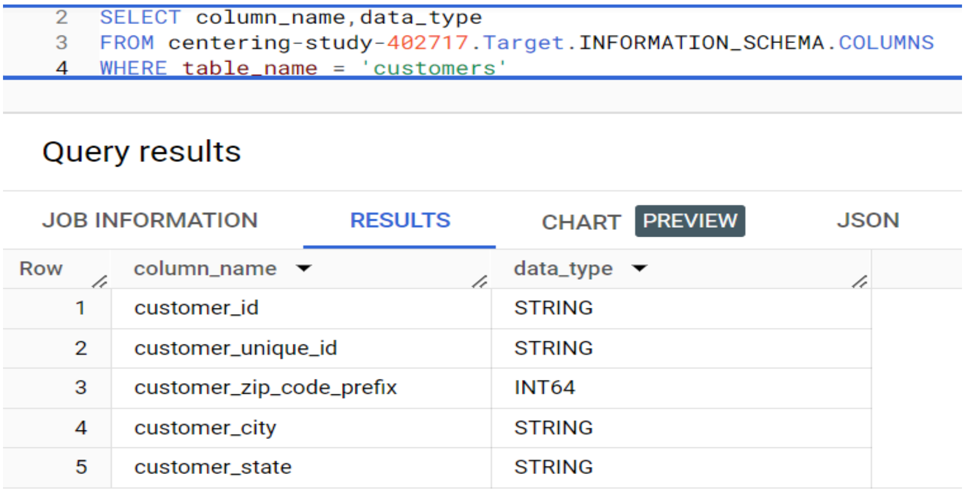


□	Field name	Type	Mode
□	customer_id	STRING	NULLABLE
□	customer_unique_id	STRING	NULLABLE
□	customer_zip_code_prefix	INTEGER	NULLABLE
□	customer_city	STRING	NULLABLE
□	customer_state	STRING	NULLABLE

Syntax:

SELECT columns_name, data_type FROM your_project_id. your_dataset_id. INFORMATION_SCHEMA.COLUMNS WHERE table_name = 'TableName';

Query:
SELECT column_name, data_type FROM centering-study-402717.Target. INFORMATION_SCHEMA.COLUMNS WHERE table_name = 'customers'



```sql
SELECT column_name,data_type
FROM centering-study-402717.Target.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers'
```

Query results

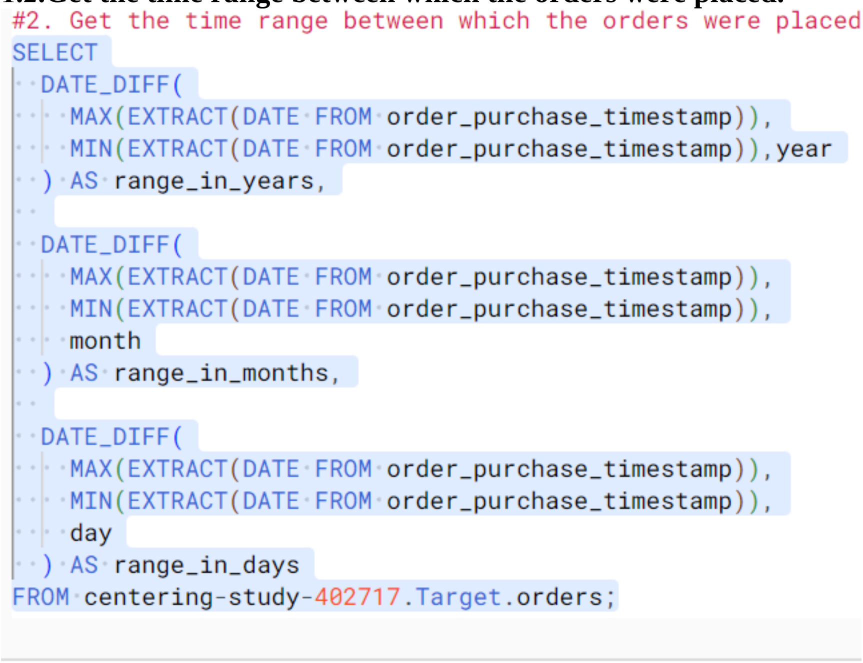
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

MySQL: To retrieve the data types of all columns in the "customers" table in MySQL, you can use the following query:
Query: **DESCRIBE customers;**

Insights:

- All columns in customer are saves as strings(varchar), except **customer_zip_code_prefix** with an integer.
- This indicates that the data primarily consists of text, and any information about dates or numeric values has been encoded as strings.

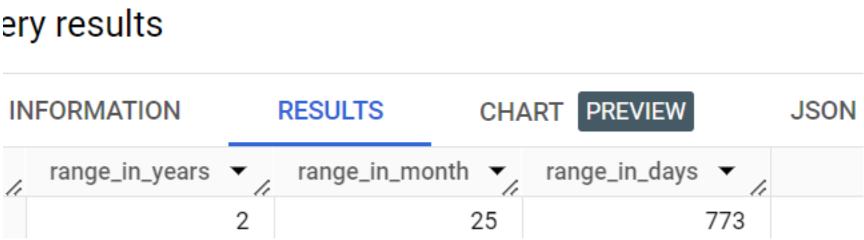
1.2.Get the time range between which the orders were placed.



```sql
#2. Get the time range between which the orders were placed
SELECT
  DATE_DIFF(
    MAX(EXTRACT(DATE FROM order_purchase_timestamp)),
    MIN(EXTRACT(DATE FROM order_purchase_timestamp)),year
  ) AS range_in_years,

  DATE_DIFF(
    MAX(EXTRACT(DATE FROM order_purchase_timestamp)),
    MIN(EXTRACT(DATE FROM order_purchase_timestamp)),
    month
  ) AS range_in_months,

  DATE_DIFF(
    MAX(EXTRACT(DATE FROM order_purchase_timestamp)),
    MIN(EXTRACT(DATE FROM order_purchase_timestamp)),
    day
  ) AS range_in_days
FROM centering-study-402717.Target.orders;
```



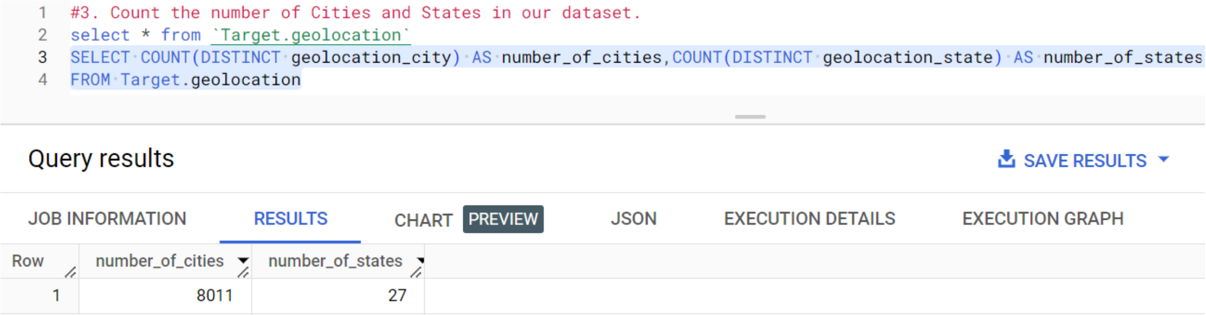
ery results

range_in_years	range_in_month	range_in_days
2	25	773

Insights:

1.The orders were made over a period spanning 2 years, 25 months and 773 days.
2.To understand order trends, seasonal variations, and overall order patterns during a specific timeframe, it's important to determine the time duration covered by the orders.

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



```
#3. Count the number of Cities and States in our dataset.
select * from `Target.geolocation`
SELECT COUNT(DISTINCT geolocation_city) AS number_of_cities,COUNT(DISTINCT geolocation_state) AS number_of_states
FROM Target.geolocation
```

Query results

Row	number_of_cities	number_of_states
1	8011	27

Query: SELECT COUNT (DISTINCT geolocation_city) AS number_of_cities,
COUNT (DISTINCT geolocation_state) AS number_of_states
FROM Target.geolocation

Insight:

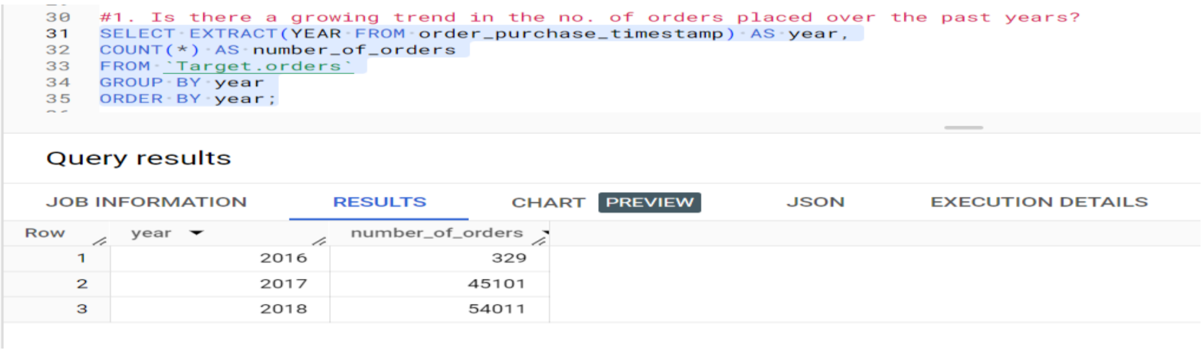
1.In this case, there are 27 different states and 8011 different cities in the dataset. This information helps us understand where our customers.
2.By checking where our customers live (cities and states), we can see if many are in one place or spread out. This helps us know which areas are most important, where we're doing best, and how far our company reaches in different locations.

In-depth-Exploration

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

SELECT EXTRACT (YEAR FROM order_purchase_timestamp) AS year,
COUNT (*) AS number_of_orders FROM `Target. Orders`
GROUP BY year ORDER BY year;

Screenshot:



```
#1. Is there a growing trend in the no. of orders placed over the past years?
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
COUNT(*) AS number_of_orders
FROM `Target.orders`
GROUP BY year
ORDER BY year;
```

Query results

Row	year	number_of_orders
1	2016	329
2	2017	45101
3	2018	54011

Insights:

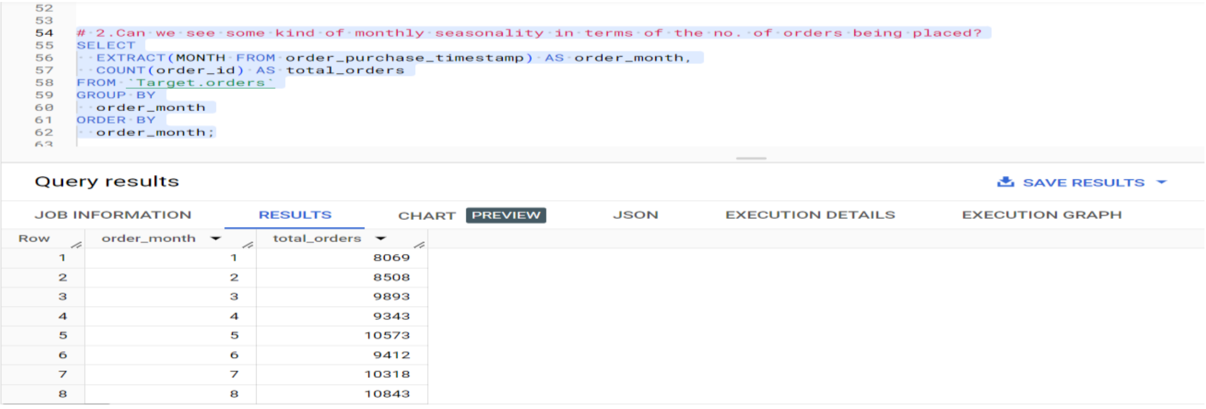
Looking at the data, we notice a positive trend in recent years with the number of orders steadily increasing, which is a positive sign. However, if we see fluctuations or a decline in orders, it indicates a different pattern or trend.

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

Query:

```
SELECT EXTRACT (MONTH FROM order_purchase_timestamp) AS order_month,
            COUNT (order_id) AS total_orders
FROM `Target. orders`
GROUP BY order_month
ORDER BY order_month;
```

Screenshot:



```
# 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(order_id) AS total_orders
FROM `Target.orders`
GROUP BY
  order_month
ORDER BY
  order_month;
```

Row	order_month	total_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

Insights:

- ➢ The months with increased orders are usually when businesses offer attractive discounts and promotions to attract more customers. These high-demand periods allow companies to generate more revenue and clear out old inventory.
- ➢ Conversely, during slower months, businesses can focus on inventory management and strategize ways to engage their customers. This includes planning for future sales and marketing campaigns to boost sales during these quieter times.
- ➢ Understanding these order patterns empowers businesses to adapt, improve their financial stability, and deliver what customers want, when they want it.

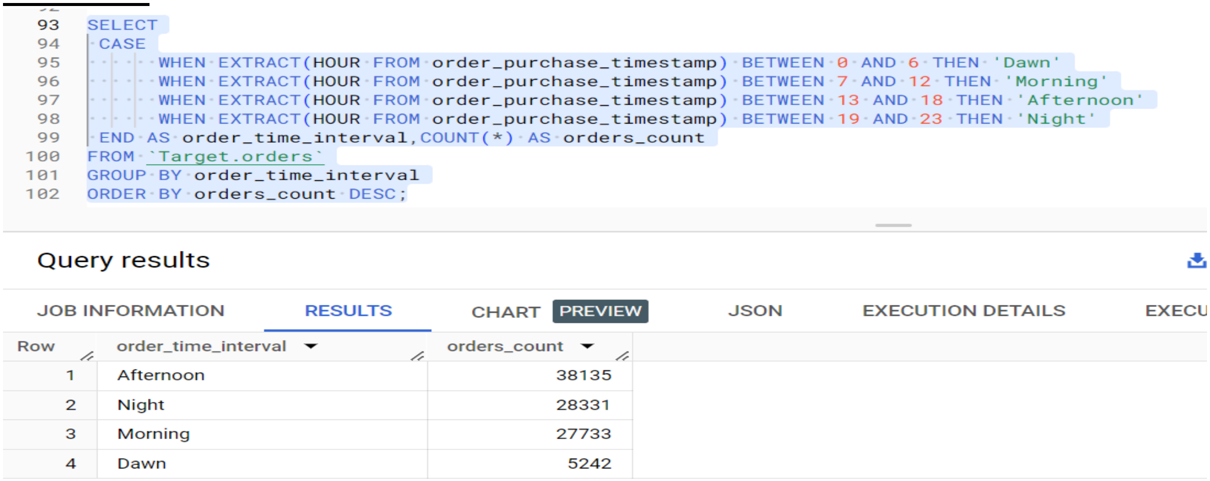
2.3During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

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

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'
    WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 19 AND 23 THEN 'Night'
 END AS order_time_interval, COUNT (*) AS orders_count
FROM `Target.orders`
GROUP BY order_time_interval
ORDER BY orders_count DESC;
```

Screenshot:



```
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_interval,COUNT(*) AS orders_count
FROM `Target.orders`
GROUP BY order_time_interval
ORDER BY orders_count DESC;
```

Query results

Row	order_time_interval	orders_count
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

Insights:

- ➢ The analysis groups orders into different parts of the day (like morning, afternoon, and night) based on the time they were made. Then, it shows which time period has the most orders.
- ➢ This helps us understand when Brazilian customers prefer to shop. For instance, it seems they like shopping in the afternoon. Knowing this, businesses can plan to have more staff or special promotions in the afternoons to serve customers better. On the other hand, very few people shop in the early morning.

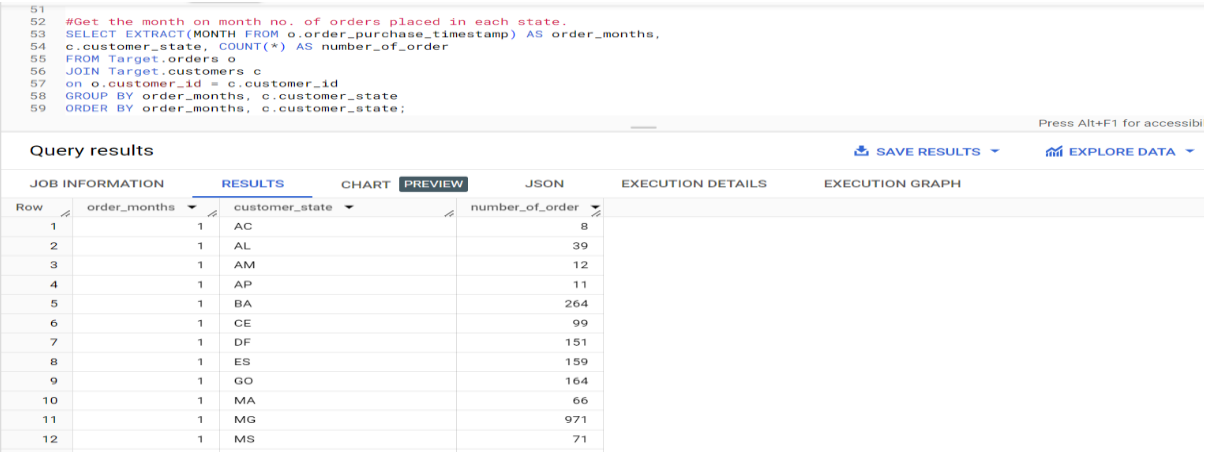
Evolution of E-commerce orders in the Brazil region:

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

Query: SELECT EXTRACT (MONTH FROM o. order_purchase_timestamp) AS order_months,

```
c.customer_state, COUNT(*) AS number_of_order
FROM Target.orders o
JOIN Target.customers c
on o.customer_id = c.customer_id
GROUP BY order_months, c.customer_state
ORDER BY order_months, c.customer_state;
```

Screenshot:



```
#Get the month on month no. of orders placed in each state.
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) AS order_months,
c.customer_state, COUNT(*) AS number_of_order
FROM Target.orders o
JOIN Target.customers c
on o.customer_id = c.customer_id
GROUP BY order_months, c.customer_state
ORDER BY order_months, c.customer_state;
```

Query results

Row	order_months	customer_state	number_of_order
1	1	AC	8
2	1	AL	39
3	1	AM	12
4	1	AP	11
5	1	BA	264
6	1	CE	99
7	1	DF	151
8	1	ES	159
9	1	GO	164
10	1	MA	66
11	1	MG	971
12	1	MS	71

Insights:

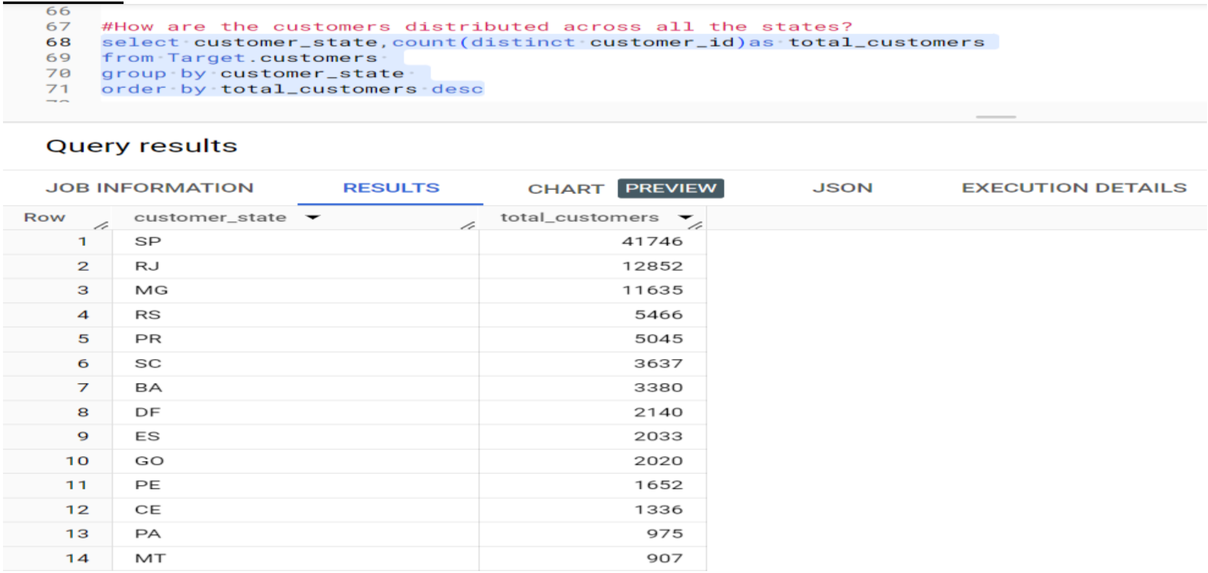
- ➢ By looking at the results of this query, we can understand how many orders each state gets each month. This helps us see if there are any patterns or changes over time. For example, we notice that the state called SP consistently has the most orders every month.
- ➢ With this information, we can focus our marketing efforts in states where orders are increasing, identify and fix problems in states with decreasing orders, or manage our products better based on how orders change in different states.

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

Query:

```
select customer_state, count (distinct customer_id) as total_customers
from Target.customers
group by customer_state
order by total_customers desc
```

Screenshot:



```
#How are the customers distributed across all the states?
select customer_state,count(distinct customer_id)as total_customers
from Target.customers
group by customer_state
order by total_customers desc
```

Query results

Row	customer_state	total_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
11	PE	1652
12	CE	1336
13	PA	975
14	MT	907

Insights:

- ➢ By checking the data, we can figure out which states have the most customers (like SP) and which ones have fewer (like RR). This can help us with things like marketing, finding new places to do business, and taking care of our customers better.
- ➢ Looking at where our customers are from can help us plan where we might want to grow next and make good decisions for our business strategy.

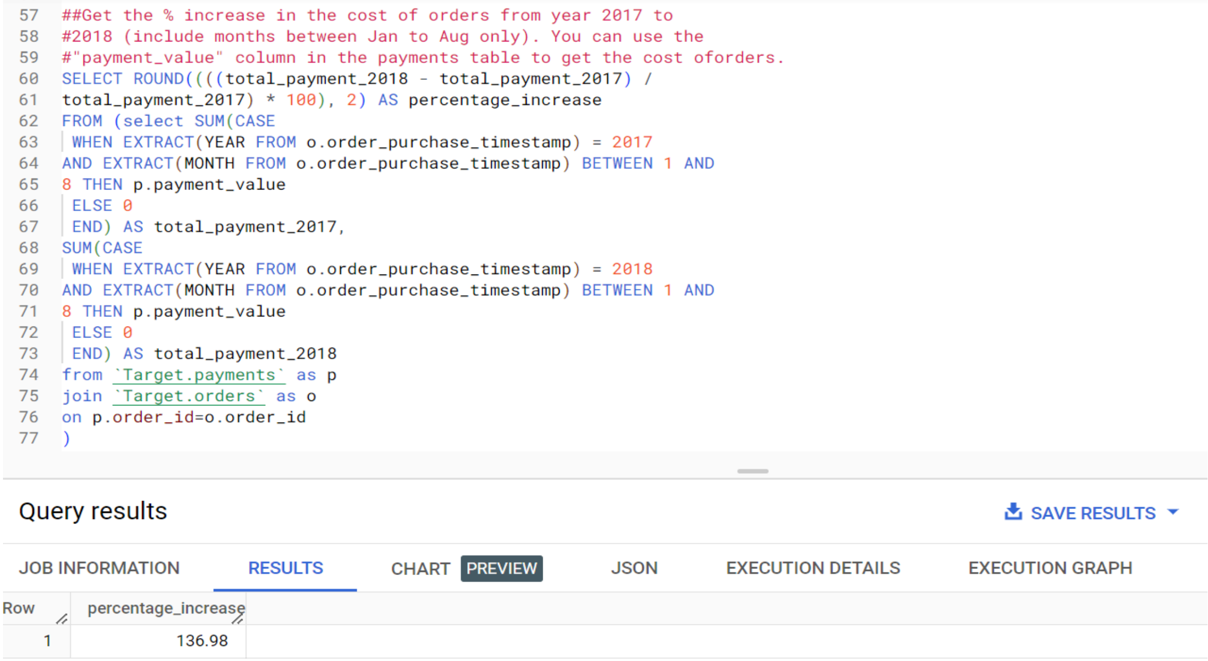
4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

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.

Query:

```
SELECT ROUND ((((total_payment_2018 - total_payment_2017) /
total_payment_2017) * 100), 2) AS percentage 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_payment_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_payment_2018
from `Target.payments` as p
join `Target.orders` as o
on p.order_id=o.order_id
)
```

Screenshot:



```
##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 oforders.
SELECT ROUND((((total_payment_2018 - total_payment_2017) /
total_payment_2017) * 100), 2) AS percentage_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_payment_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_payment_2018
from `Target.payments` as p
join `Target.orders` as o
on p.order_id=o.order_id
)
```

Row	percentage_increase
1	136.98

Insights:

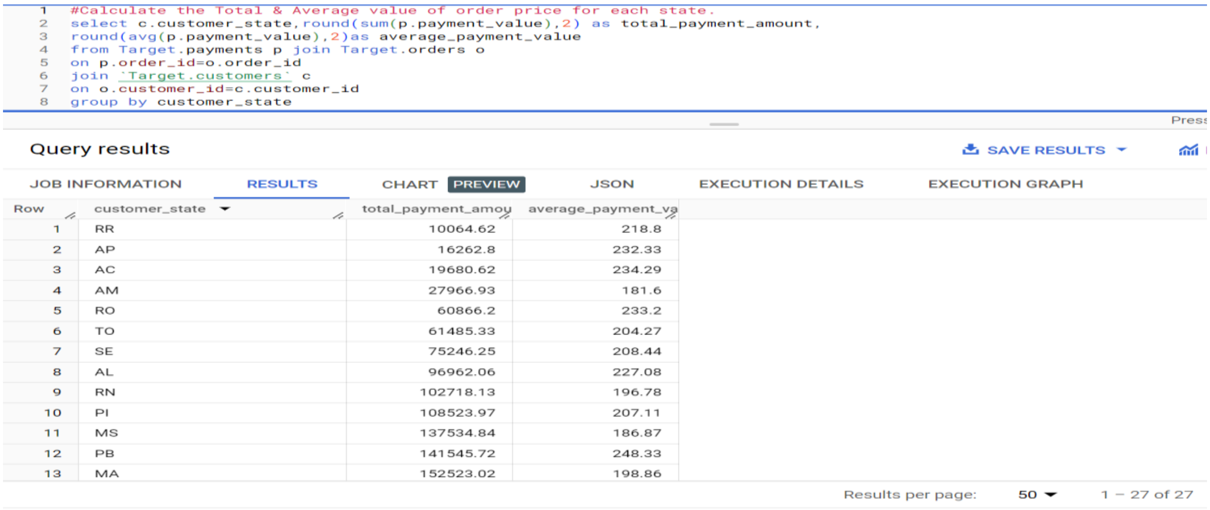
- ➢ We're only looking at orders made between January and August for both 2017 and 2018.
- ➢ To calculate the percentage increase, we're comparing the prices each month in 2017 to the prices in the same months in 2018.
- ➢ The results show that prices increased by about 137% from 2017 to 2018.

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

Query:

```
select c.customer_state,
         round(sum(p. payment_value),2) as total_payment_amount,
         round(avg (p. payment_value),2) as average_payment_value
         from Target.payments p join Target.orders o
         on p.order_id=o.order_id
         join `Target.customers` c
         on o.customer_id=c.customer_id
         group by customer_state
```

Screenshot:



```
#Calculate the Total & Average value of order price for each state.
select c.customer_state,round(sum(p.payment_value),2) as total_payment_amount,
round(avg(p.payment_value),2)as average_payment_value
from Target.payments p join Target.orders o
on p.order_id=o.order_id
join `Target.customers` c
on o.customer_id=c.customer_id
group by customer_state
```

Row	customer_state	total_payment_amou	average_payment_va
1	RR	10064.62	218.8
2	AP	16262.8	232.33
3	AC	19680.62	234.29
4	AM	27966.93	181.6
5	RO	60866.2	233.2
6	TO	61485.33	204.27
7	SE	75246.25	208.44
8	AL	96962.06	227.08
9	RN	102718.13	196.78
10	PI	108523.97	207.11
11	MS	137534.84	186.87
12	PB	141545.72	248.33
13	MA	152523.02	198.86

Insights:

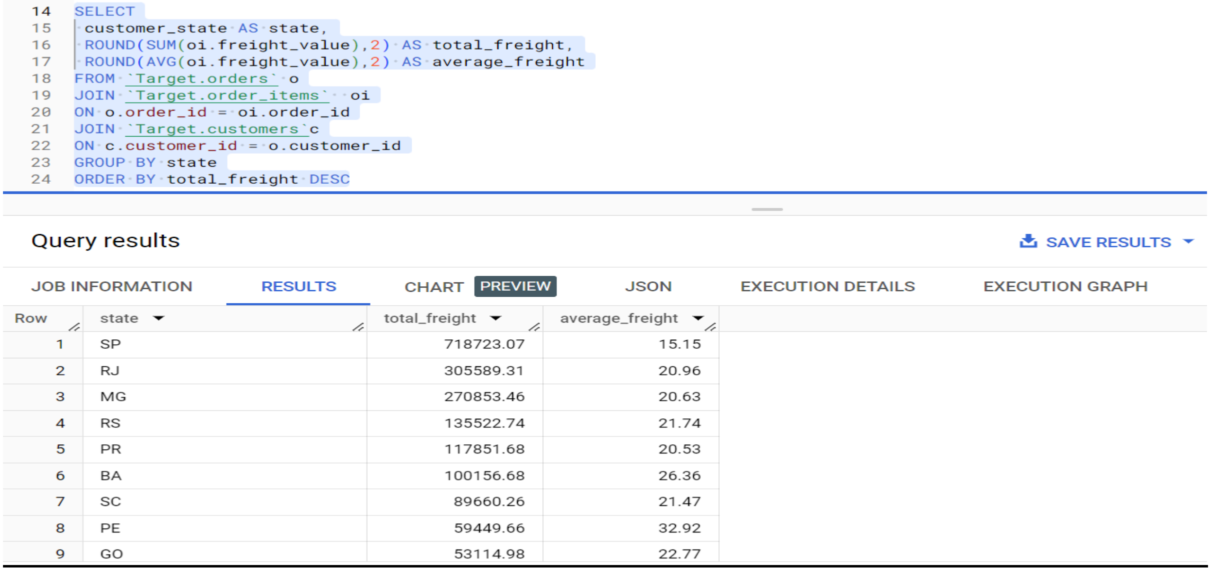
- ➢ The "total_payment_amount" column adds up all the money spent on orders in each state, showing the total order value.
- ➢ The "average_payment_value" column tells us how much people typically spend on orders in each state, showing the average order price.
- ➢ By looking at the results, we can spot states where people spend a lot of money on orders, which might be good for business.
- ➢ Comparing the average order prices across states helps us see where people tend to spend more or less, which can guide our marketing and pricing strategies.
- ➢ To make the best decisions based on this data, we should also consider factors like population, economic conditions, and customer behaviour in each state.

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

Query:

```
SELECT customer_state AS state,
ROUND(SUM(oi.freight_value),2) AS total_freight,
ROUND(AVG(oi.freight_value),2) AS average_freight
FROM `Target.orders` o
JOIN `Target.order_items`  oi
ON o.order_id = oi.order_id
JOIN `Target.customers`c
ON c.customer_id = o.customer_id
GROUP BY state
ORDER BY total_freight DESC
```

Screenshot:



```
SELECT
 customer_state AS state,
 ROUND(SUM(oi.freight_value),2) AS total_freight,
 ROUND(AVG(oi.freight_value),2) AS average_freight
FROM `Target.orders` o
JOIN `Target.order_items`  oi
ON o.order_id = oi.order_id
JOIN `Target.customers`c
ON c.customer_id = o.customer_id
GROUP BY state
ORDER BY total_freight DESC
```

Row	state	total_freight	average_freight
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77

Insights:

- ➢ By looking at the data, we can identify states, like SP, where the total cost of shipping is high. This might mean that these places have expensive shipping or logistical challenges.
- ➢ When we want to improve how we ship things or set prices, it's useful to find areas where shipping costs are higher or lower by comparing the average shipping costs across states.
- ➢ Understanding why shipping costs vary between states can give us insights into how people in different places like to ship things, where our suppliers are, or what customers prefer. This information helps us make our processes better and save money.

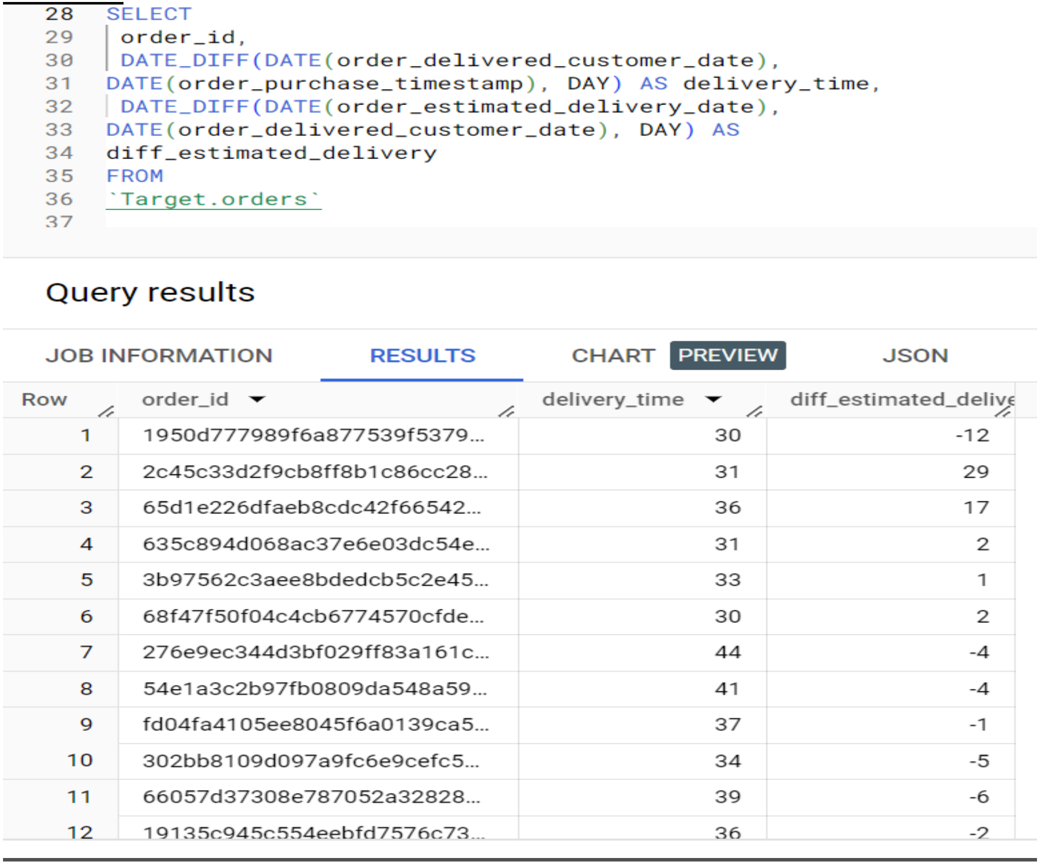
5.**Analysis based on sales, freight and delivery time.**

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.
   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:
   - **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
   - **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date.

Query:

```
SELECT
 order_id,
 DATE_DIFF(DATE(order_delivered_customer_date),
DATE(order_purchase_timestamp), DAY) AS delivery_time,
 DATE_DIFF(DATE(order_estimated_delivery_date),
DATE(order_delivered_customer_date), DAY) AS
diff_estimated_delivery
FROM
`Target.orders`
```

Screenshot:



```
SELECT
 order_id,
 DATE_DIFF(DATE(order_delivered_customer_date),
DATE(order_purchase_timestamp), DAY) AS delivery_time,
 DATE_DIFF(DATE(order_estimated_delivery_date),
DATE(order_delivered_customer_date), DAY) AS
diff_estimated_delivery
FROM
`Target.orders`
```

Query results

Row	order_id	delivery_time	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	31	29
3	65d1e226dfaeb8cdc42f66542...	36	17
4	635c894d068ac37e6e03dc54e...	31	2
5	3b97562c3aee8bdedcb5c2e45...	33	1
6	68f47f50f04c4cb6774570cfde...	30	2
7	276e9ec344d3bf029ff83a161c...	44	-4
8	54e1a3c2b97fb0809da548a59...	41	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	34	-5
11	66057d37308e787052a32828...	39	-6
12	19135c945c554eebfd7576c73...	36	-2

- ➢ By looking at the delivery time and the difference between estimated and actual delivery dates, we can understand how well our delivery process is working. We can see if orders are late or early compared to what we expected.
- ➢ We can dig deeper into these numbers to find patterns, unusual cases, or things that might be causing delays or differences in delivery times.

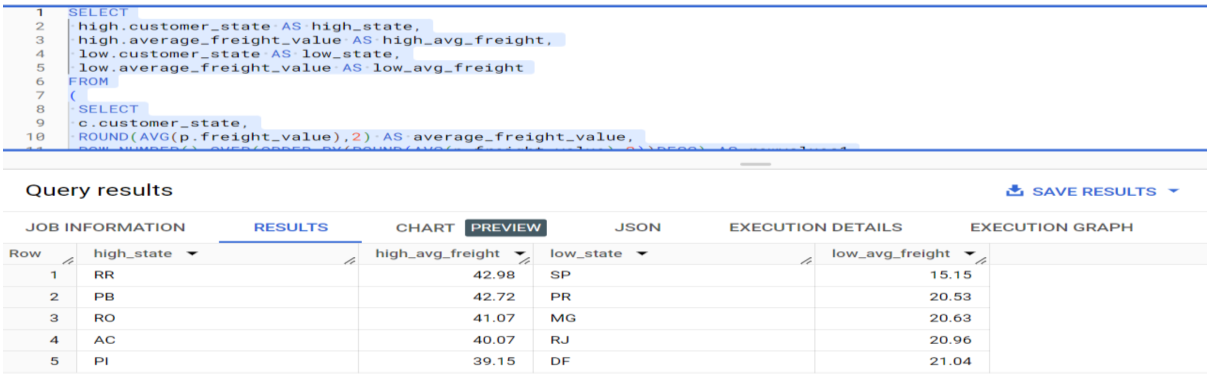
- These findings help us make customers happier by setting better expectations, improving the delivery process, and making our logistics (the way we move things around) work better.

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

Query:

```
SELECT
 high.customer_state AS high_state,
 high.average_freight_value AS high_avg_freight,
 low.customer_state AS low_state,
 low.average_freight_value AS low_avg_freight
FROM
(SELECT
 c.customer_state,
 ROUND(AVG(p.freight_value),2) AS average_freight_value,
 ROW_NUMBER() OVER(ORDER BY(ROUND(AVG(p.freight_value),2))DESC) AS rowvalues1
 FROM Target.orders AS o
 JOIN Target.order_items AS p
 ON o.order_id = p.order_id
 JOIN Target.customers AS c
 ON o.customer_id = c.customer_id
 GROUP BY
 c.customer_state
 ORDER BY
 average_freight_value DESC
 LIMIT
 5) AS high
JOIN
(SELECT
 c.customer_state,
 ROUND(AVG(p.freight_value),2) AS average_freight_value,
 ROW_NUMBER() OVER(ORDER BY (ROUND(AVG(p.freight_value),2)))AS rowvalues2
 FROM Target.orders AS o
 JOIN Target.order_items AS p
 ON o.order_id = p.order_id
 JOIN Target.customers AS c
 ON o.customer_id = c.customer_id
 GROUP BY
 c.customer_state
 ORDER BY
 average_freight_value
 LIMIT
 5) AS low
ON high.rowvalues1 = low.rowvalues2;
```

Screenshot:



```
SELECT
 high.customer_state AS high_state,
 high.average_freight_value AS high_avg_freight,
 low.customer_state AS low_state,
 low.average_freight_value AS low_avg_freight
FROM
(
 SELECT
 c.customer_state,
 ROUND(AVG(p.freight_value),2) AS average_freight_value,
```

Row	high_state	high_avg_freight	low_state	low_avg_freight
1	RR	42.98	SP	15.15
2	PB	42.72	PR	20.53
3	RO	41.07	MG	20.63
4	AC	40.07	RJ	20.96
5	PI	39.15	DF	21.04

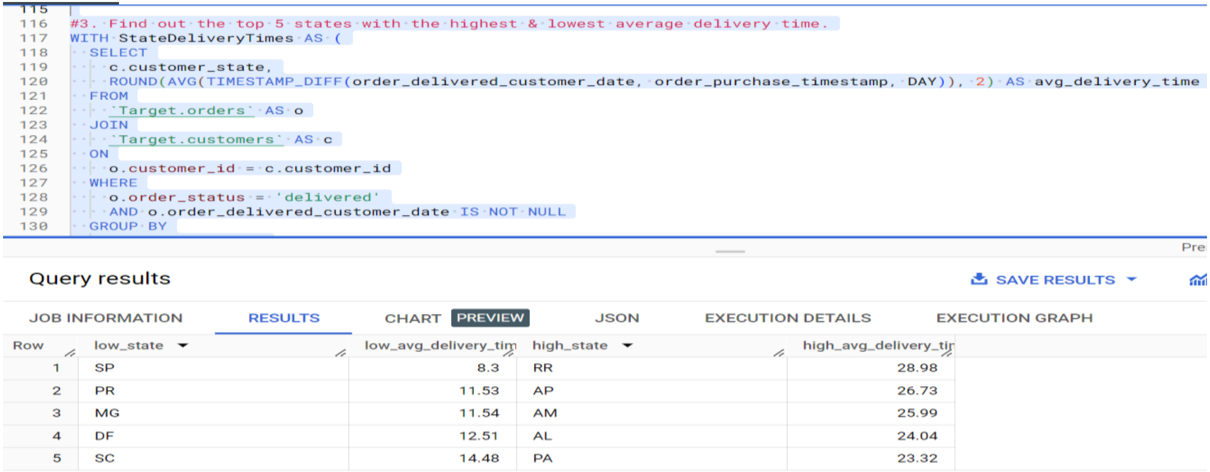
Insights:

- ➢ Some states, such as RR and PB, have higher shipping costs, likely due to factors like remote locations and increased transportation expenses.
- ➢ We have an opportunity to reduce costs by optimizing logistics operations. Identifying states with lower shipping costs, such as SP and PR, provides potential cost-saving options.
- ➢ This data can guide us in creating specific strategies, including negotiating for better freight rates and finding ways to minimize supply chain costs.
- ➢ When analysing the data, consider additional variables such as distance, transportation infrastructure, carrier availability, and regional economic differences.

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

```
WITH StateDeliveryTimes AS (
 SELECT
   c.customer_state,
   ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)), 2) AS
avg_delivery_time
 FROM
   `Target.orders` AS o
 JOIN
   `Target.customers` AS c
 ON
   o.customer_id = c.customer_id
 WHERE
   o.order_status = 'delivered'
   AND o.order_delivered_customer_date IS NOT NULL
 GROUP BY
   c.customer_state
)
SELECT
 sd_low.customer_state AS low_state,
 sd_low.avg_delivery_time AS low_avg_delivery_time,
 sd_high.customer_state AS high_state,
 sd_high.avg_delivery_time AS high_avg_delivery_time
FROM
 (SELECT
   customer_state,
   avg_delivery_time,
   ROW_NUMBER() OVER (ORDER BY avg_delivery_time) AS rownum_low
  FROM StateDeliveryTimes
  ORDER BY rownum_low
  LIMIT 5) AS sd_low
JOIN
 (SELECT
   customer_state,
   avg_delivery_time,
   ROW_NUMBER() OVER (ORDER BY avg_delivery_time DESC) AS rownum_high
  FROM StateDeliveryTimes
  ORDER BY rownum_high
  LIMIT 5) AS sd_high
ON
 sd_low.rownum_low = sd_high.rownum_high;
```

screenshot:



```
#3. Find out the top 5 states with the highest & lowest average delivery time.
WITH StateDeliveryTimes AS (
  SELECT
    c.customer_state,
    ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)), 2) AS avg_delivery_time
  FROM
    `Target.orders` AS o
  JOIN
    `Target.customers` AS c
  ON
    o.customer_id = c.customer_id
  WHERE
    o.order_status = 'delivered'
    AND o.order_delivered_customer_date IS NOT NULL
  GROUP BY
```

Query results

Row	low_state	low_avg_delivery_tim	high_state	high_avg_delivery_tir
1	SP	8.3	RR	28.98
2	PR	11.53	AP	26.73
3	MG	11.54	AM	25.99
4	DF	12.51	AL	24.04
5	SC	14.48	PA	23.32

Insights:

- **Efficient Delivery Operations:**
  Identify efficient delivery areas like SP and PR for faster delivery and areas like RR and AP for longer delivery times.
- **Customer Satisfaction and Efficiency**:
  Use these insights to improve customer satisfaction, enhance operational efficiency, and set realistic delivery expectations based on regional times.
- **Additional Factors:**
  Consider factors such as population density, urban vs. rural locations, customer expectations, and logistical challenges when analysing the data.
- **Focus on Delivery Efficiency:**
  Concentrate on areas where delivery operations can be optimized to enhance customer experiences and operational efficiency.

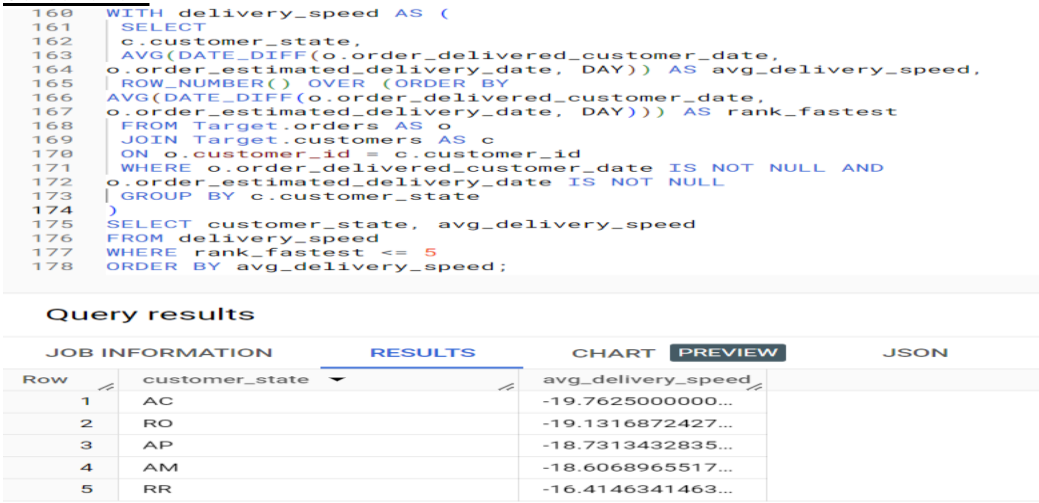
5.4. **Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.**

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

Query:

```
WITH 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,
 ROW_NUMBER() OVER (ORDER BY
AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_estimated_delivery_date, DAY))) AS rank_fastest
 FROM Target.orders AS o
 JOIN Target.customers AS c
 ON o.customer_id = c.customer_id
 WHERE o.order_delivered_customer_date IS NOT NULL AND
o.order_estimated_delivery_date IS NOT NULL
 GROUP BY c.customer_state
)
SELECT customer_state, avg_delivery_speed
FROM delivery_speed
WHERE rank_fastest <= 5
ORDER BY avg_delivery_speed;
```

Screenshot:



```
WITH 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,
  ROW_NUMBER() OVER (ORDER BY
AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_estimated_delivery_date, DAY))) AS rank_fastest
  FROM Target.orders AS o
  JOIN Target.customers AS c
  ON o.customer_id = c.customer_id
  WHERE o.order_delivered_customer_date IS NOT NULL AND
o.order_estimated_delivery_date IS NOT NULL
  GROUP BY c.customer_state
)
SELECT customer_state, avg_delivery_speed
FROM delivery_speed
WHERE rank_fastest <= 5
ORDER BY avg_delivery_speed;
```

Query results

Row	customer_state	avg_delivery_speed
1	AC	-19.7625000000...
2	RO	-19.1316872427...
3	AP	-18.7313432835...
4	AM	-18.6068965517...
5	RR	-16.4146341463...

Insights:

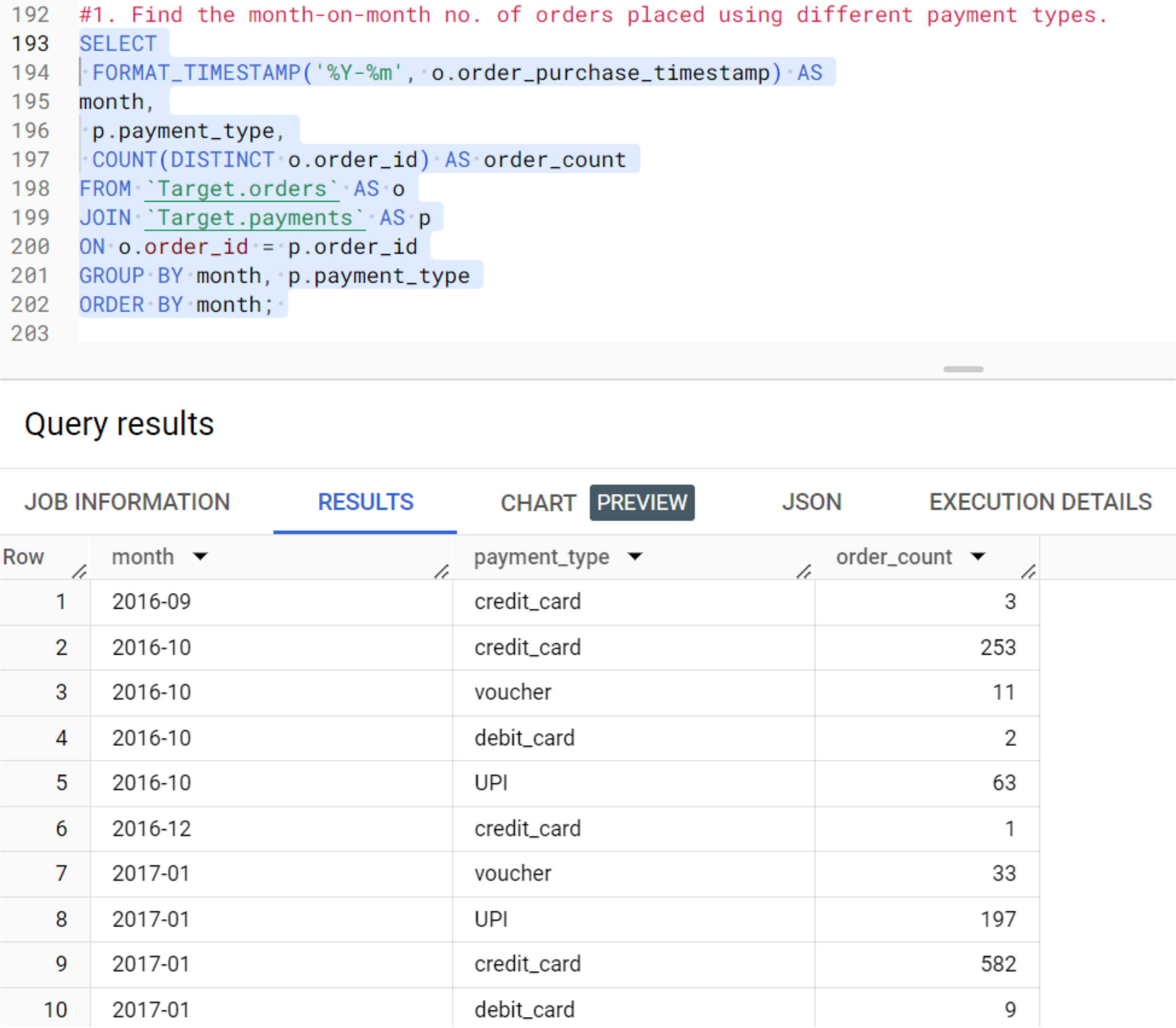
- **Faster Delivery, Happy Customers:**
  Our company operates in states like AC, RO, AP, and AM, where delivery times are the fastest.We can promote our quick and reliable service to attract more clients and boost customer satisfaction.
- **Improving Operations and Growth:**
  This data can help us enhance our operations and make our customers happier.
  We can optimize our logistics and even consider expanding into areas known for fast order delivery.

6.Analysis based on the payments

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

```
SELECT
 FORMAT_TIMESTAMP('%Y-%m', o.order_purchase_timestamp) AS
month,
 p.payment_type,
 COUNT(DISTINCT o.order_id) AS order_count
FROM `Target.orders` AS o
JOIN `Target.payments` AS p
ON o.order_id = p.order_id
GROUP BY month, p.payment_type
ORDER BY month;
```

Screenshot:



```
#1. Find the month-on-month no. of orders placed using different payment types.
SELECT
 FORMAT_TIMESTAMP('%Y-%m', o.order_purchase_timestamp) AS
month,
 p.payment_type,
 COUNT(DISTINCT o.order_id) AS order_count
FROM `Target.orders` AS o
JOIN `Target.payments` AS p
ON o.order_id = p.order_id
GROUP BY month, p.payment_type
ORDER BY month;
```

Query results

Row	month	payment_type	order_count
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

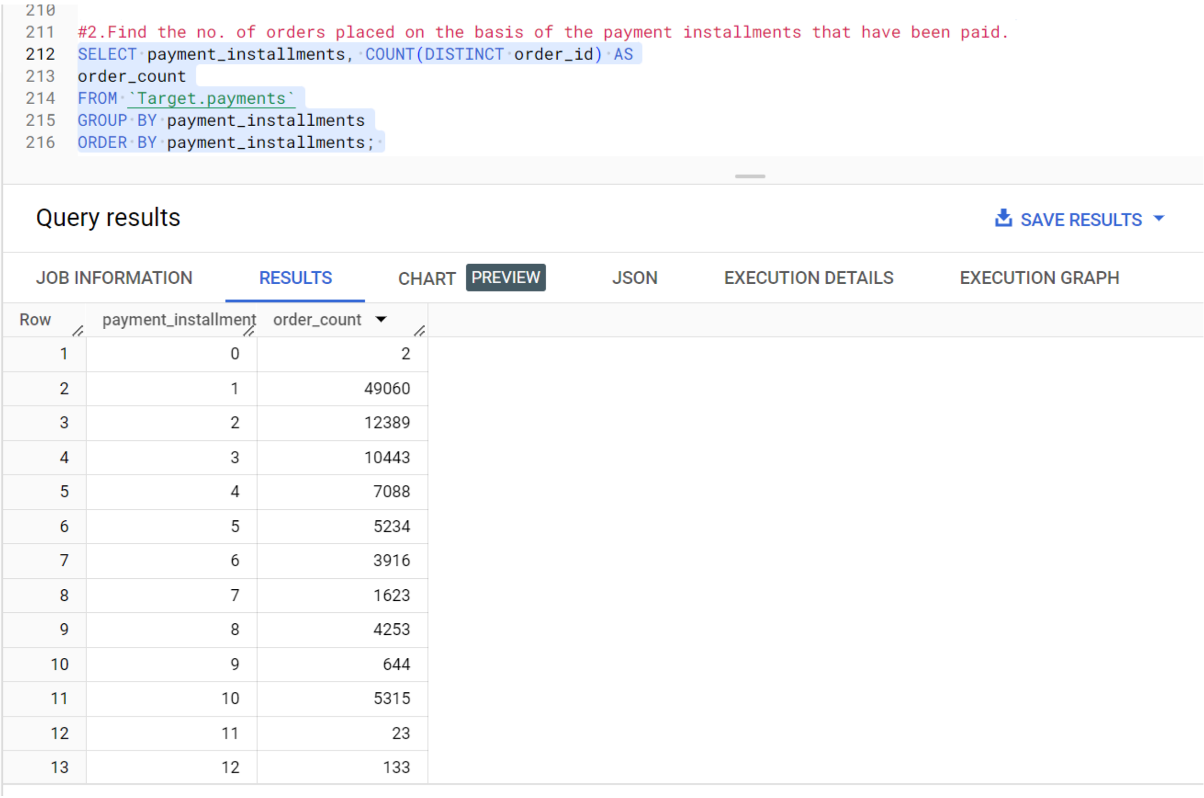
Insights:

- The data shows that credit card payment was most frequently used as a payment method in November 2017.
- Analyzing month-to-month trends in order counts can help us understand seasonality, identify peak months, and evaluate the impact of marketing efforts or external factors on consumer behavior.
- By recognizing payment preferences in different months, businesses can optimize their payment processes, tailor marketing campaigns, and improve the overall customer experience to better meet customer demands and expectations.

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

```
SELECT payment_installments, COUNT(DISTINCT order_id) AS
order_count
FROM `Target.payments`
GROUP BY payment_installments
ORDER BY payment_installments;
```

Screenshot:



```
#2.Find the no. of orders placed on the basis of the payment installments that have been paid.
SELECT payment_installments, COUNT(DISTINCT order_id) AS
order_count
FROM `Target.payments`
GROUP BY payment_installments
ORDER BY payment_installments;
```

Row	payment_installment	order_count
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644
11	10	5315
12	11	23
13	12	133

Insights:

- **Single Payment Installments:**
  There were 49,060 orders where customers opted for just one payment installment.
- **Uncovering Payment Preferences:**
  This analysis helps us uncover whether customers have particular preferences when it comes to payment options.
- **Customer Budgeting Insights**:
  It can provide insights into whether customers tend to Favor a specific number of payment installments, indicating their budgeting or financing choices.
- **Understanding Customer Behavior:**
  By examining the distribution of orders based on payment installments, we can gain a better understanding of customer shopping habits and their inclination towards flexible payment methods.