TARGET-SQL

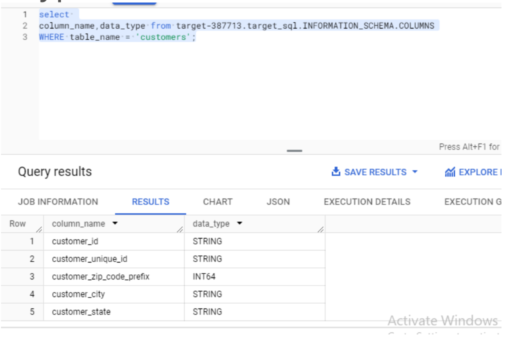
Context:

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 analyzing 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.

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

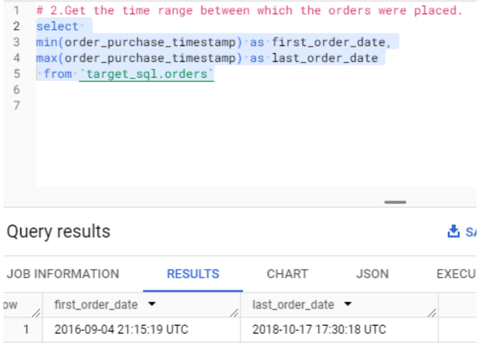


```
select
column_name,data_type from target-387713.target_sql.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers';
```

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

```
select
column_name, data_type from target-387713.target_sql.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers';
```

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



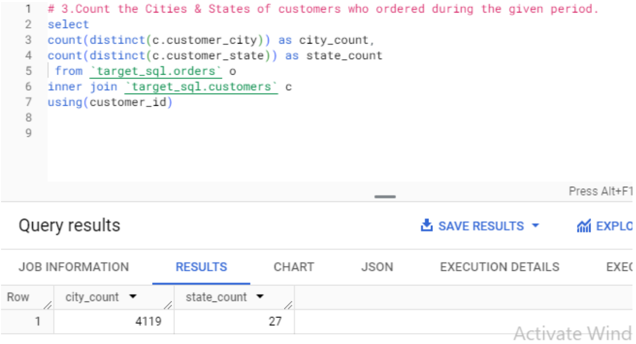
```
# 2.Get the time range between which the orders were placed.
select
min(order_purchase_timestamp) as first_order_date,
max(order_purchase_timestamp) as last_order_date
 from `target_sql.orders`
```

ow	first_order_date	last_order_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

```
select
min(order_purchase_timestamp) as first_order,
max(order_purchase_timestamp) as last_order
  from `target_sql.orders`
```

6.

7. Count the Cities & States of customers who ordered during the given period.
8.
9.



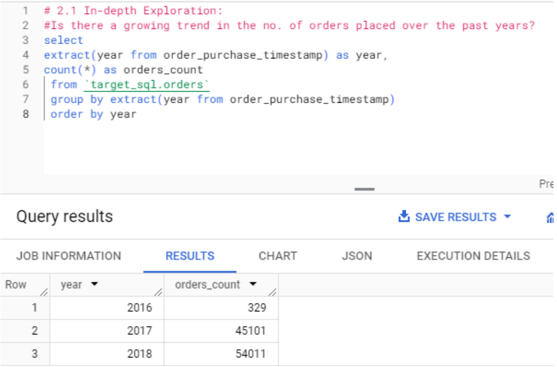
```
# 3.Count the Cities & States of customers who ordered during the given period.
select
count(distinct(c.customer_city)) as city_count,
count(distinct(c.customer_state)) as state_count
 from `target_sql.orders` o
inner join `target_sql.customers` c
using(customer_id)
```

Row	city_count	state_count
1	4119	27

2.1 In-depth Exploration:

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 order_count
from `target_sql.orders`
group by year
order by year
```

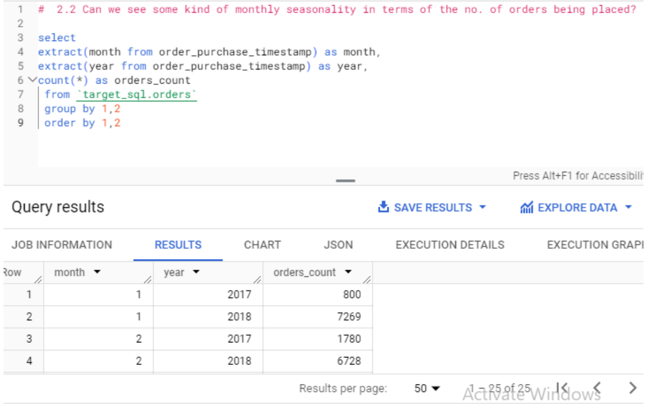


```
# 2.1 In-depth Exploration:
#Is there a growing trend in the no. of orders placed over the past years?
select
extract(year from order_purchase_timestamp) as year,
count(*) as orders_count
 from `target_sql.orders`
 group by extract(year from order_purchase_timestamp)
 order by year
```

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

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

```
select
extract(month from order_purchase_timestamp) as month,
extract(year from order_purchase_timestamp) as year,
count(*) as order_count
from `target_sql.orders`
group by 1,2
order by 1,2
```



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

select
extract(month from order_purchase_timestamp) as month,
extract(year from order_purchase_timestamp) as year,
count(*) as orders_count
 from `target_sql.orders`
 group by 1,2
 order by 1,2
```

Row	month	year	orders_count
1	1	2017	800
2	1	2018	7269
3	2	2017	1780
4	2	2018	6728

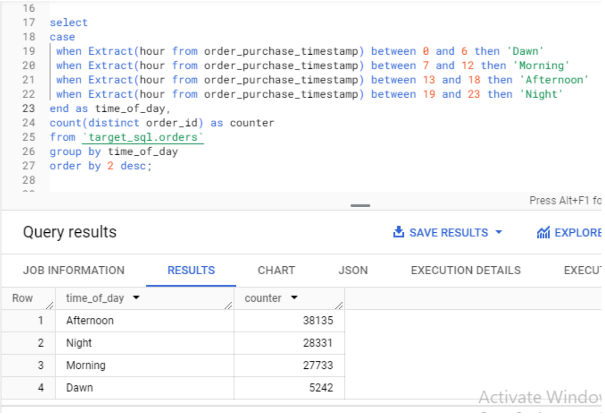
2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

0-6 hrs : Dawn

7-12 hrs : Mornings

13-18 hrs : Afternoon

19-23 hrs : Night



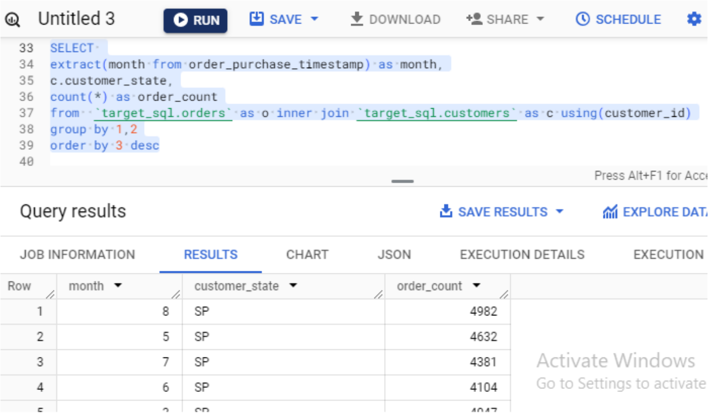
```
select
case
 when Extract(hour from order_purchase_timestamp) between 0 and 6 then 'Dawn'
 when Extract(hour from order_purchase_timestamp) between 7 and 12 then 'Morning'
 when Extract(hour from order_purchase_timestamp) between 13 and 18 then 'Afternoon'
 when Extract(hour from order_purchase_timestamp) between 19 and 23 then 'Night'
end as time_of_day,
count(distinct order_id) as counter
from `target_sql.orders`
group by time_of_day
order by 2 desc;
```

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

```
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 time_of_day,
COUNT(DISTINCT order_id) as order_count
from `target_sql.orders`
group by time_of_day
ORDER BY order_count desc;
```

3.1 Evolution of E-commerce orders in the Brazil region:

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

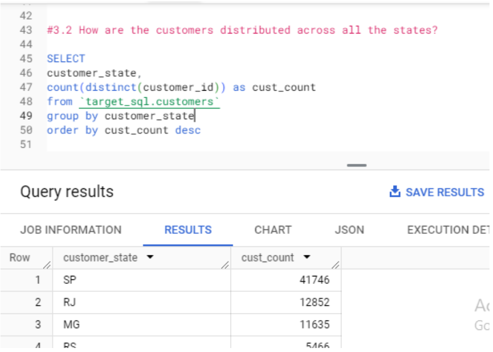


```
SELECT
extract(month from order_purchase_timestamp) as month,
c.customer_state,
count(*) as order_count
from  `target_sql.orders` as o inner join `target_sql.customers` as c using(customer_id)
group by 1,2
order by 3 desc
```

Row	month	customer_state	order_count
1	8	SP	4982
2	5	SP	4632
3	7	SP	4381
4	6	SP	4104

```
select
extract(month from order_purchase_timestamp) as month,
c.customer_state,
count(*) as order_count
from `target_sql.orders` o inner join `target_sql.customers` c using (customer_id)
group by 1,2
order by 3 desc;
```

3.2How are the customers distributed across all the states?



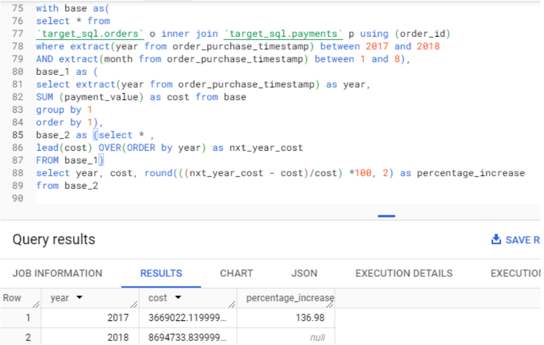
```
#3.2 How are the customers distributed across all the states?

SELECT
customer_state,
count(distinct(customer_id)) as cust_count
from `target_sql.customers`
group by customer_state
order by cust_count desc
```

Query results

Row	customer_state	cust_count
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466

```
select customer_state,
count(distinct customer_id) as cus_count
from `target_sql.customers`
group by 1
order by 2 desc
```

```
#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.
```



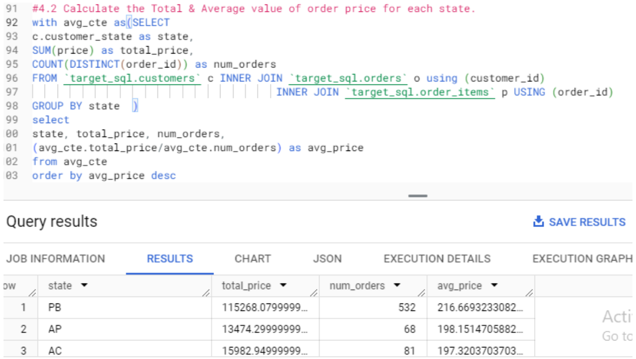
```
with base as(
select * from
`target_sql.orders` o inner join `target_sql.payments` p using (order_id)
where extract(year from order_purchase_timestamp) between 2017 and 2018
AND extract(month from order_purchase_timestamp) between 1 and 8),
base_1 as (
select extract(year from order_purchase_timestamp) as year,
SUM (payment_value) as cost from base
group by 1
order by 1),
base_2 as (select * ,
lead(cost) OVER(ORDER by year) as nxt_year_cost
FROM base_1)
select year, cost, round(((nxt_year_cost - cost)/cost) *100, 2) as percentage_increase
from base_2
```

Query results

Row	year	cost	percentage_increase
1	2017	3669022.119999...	136.98
2	2018	8694733.839999...	null

```
with base as(
select * from
`target_sql.orders` o inner join `target_sql.payments` p using (order_id)
where extract(year from order_purchase_timestamp) between 2017 and 2018
AND extract(month from order_purchase_timestamp) between 1 and 8),
base_1 as (
select extract(year from order_purchase_timestamp) as year,
SUM (payment_value) as cost from base
group by 1
order by 1),
base_2 as (select * ,
lead(cost) OVER(ORDER by year) as nxt_year_cost
FROM base_1)
select year, cost, round(((nxt_year_cost - cost)/cost) *100, 2) as percentage_increase
from base_2
```

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



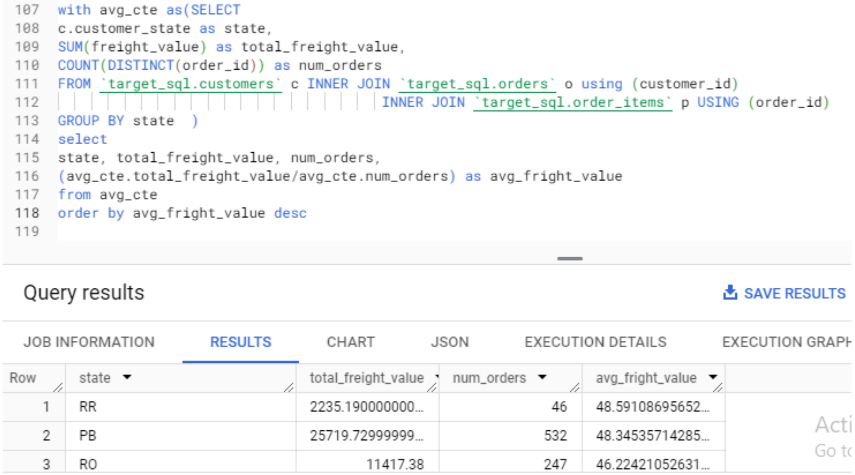
```
#4.2 Calculate the Total & Average value of order price for each state.
with avg_cte as(SELECT
c.customer_state as state,
SUM(price) as total_price,
COUNT(DISTINCT(order_id)) as num_orders
FROM `target_sql.customers` c INNER JOIN `target_sql.orders` o using (customer_id)
                                            INNER JOIN `target_sql.order_items` p USING (order_id)
GROUP BY state  )
select
state, total_price, num_orders,
(avg_cte.total_price/avg_cte.num_orders) as avg_price
from avg_cte
order by avg_price desc
```

Query results

ow	state	total_price	num_orders	avg_price
1	PB	115268.0799999...	532	216.6693233082...
2	AP	13474.29999999...	68	198.1514705882...
3	AC	15982.94999999...	81	197.3203703703...

```
with avg_cte as(SELECT
c.customer_state as state,
SUM(price) as total_price,
COUNT(DISTINCT(order_id)) as num_orders
FROM `target_sql.customers` c INNER JOIN `target_sql.orders` o using (customer_id)
                                       INNER JOIN `target_sql.order_items` p USING (order_id)
GROUP BY state  )
select
state, total_price, num_orders,
(avg_cte.total_price/avg_cte.num_orders) as avg_price
from avg_cte
order by avg_price desc
```

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

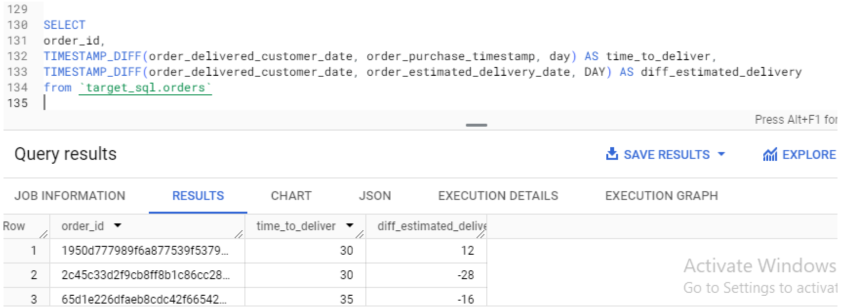


```
with avg_cte as(SELECT
c.customer_state as state,
SUM(freight_value) as total_freight_value,
COUNT(DISTINCT(order_id)) as num_orders
FROM `target_sql.customers` c INNER JOIN `target_sql.orders` o using (customer_id)
                                          INNER JOIN `target_sql.order_items` p USING (order_id)
GROUP BY state  )
select
state, total_freight_value, num_orders,
(avg_cte.total_freight_value/avg_cte.num_orders) as avg_fright_value
from avg_cte
order by avg_fright_value desc
```

Query results

Row	state	total_freight_value	num_orders	avg_fright_value
1	RR	2235.190000000...	46	48.59108695652...
2	PB	25719.72999999...	532	48.34535714285...
3	RO	11417.38	247	46.22421052631...

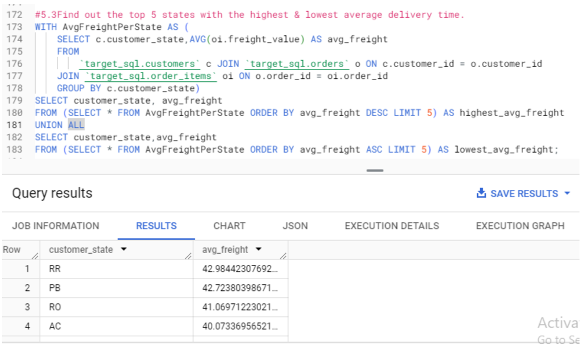
```
with avg_cte as(SELECT
c.customer_state as state,
SUM(freight_value) as total_freight_value,
COUNT(DISTINCT(order_id)) as num_orders
FROM `target_sql.customers` c INNER JOIN `target_sql.orders` o using (customer_id)
                                       INNER JOIN `target_sql.order_items` p USING (order_id)
GROUP BY state  )
select
state, total_freight_value, num_orders,
(avg_cte.total_freight_value/avg_cte.num_orders) as avg_fright_value
from avg_cte
order by avg_fright_value desc


#5.1 Analysis based on sales, freight and delivery time.
#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_delivered_customer_date - order_estimated_delivery_date
```



```

SELECT
order_id,
TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS time_to_deliver,
TIMESTAMP_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY) AS diff_estimated_delivery
from `target_sql.orders`

```

Row	order_id	time_to_deliver	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	-28
3	65d1e226dfaeb8cdc42f66542...	35	-16

```
SELECT
order_id,
TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
time_to_deliver,
TIMESTAMP_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY) AS
diff_estimated_delivery
from `target_sql.orders`



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

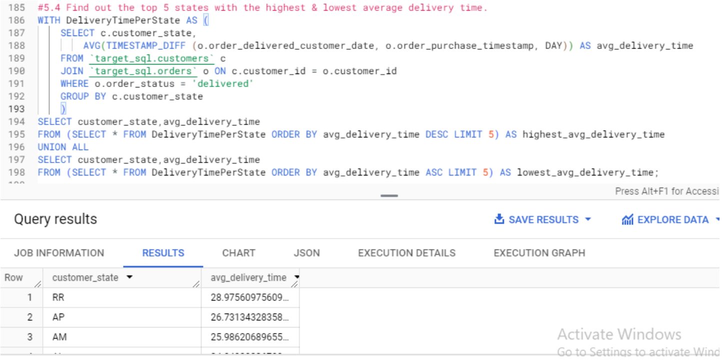


```
#5.3Find out the top 5 states with the highest & lowest average delivery time.
WITH AvgFreightPerState AS (
    SELECT c.customer_state,AVG(oi.freight_value) AS avg_freight
    FROM
        `target_sql.customers` c JOIN `target_sql.orders` o ON c.customer_id = o.customer_id
    JOIN `target_sql.order_items` oi ON o.order_id = oi.order_id
    GROUP BY c.customer_state)
SELECT customer_state, avg_freight
FROM (SELECT * FROM AvgFreightPerState ORDER BY avg_freight DESC LIMIT 5) AS highest_avg_freight
UNION ALL
SELECT customer_state,avg_freight
FROM (SELECT * FROM AvgFreightPerState ORDER BY avg_freight ASC LIMIT 5) AS lowest_avg_freight;
```

Row	customer_state	avg_freight
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...

```
WITH AvgFreightPerState AS (
    SELECT c.customer_state,AVG(oi.freight_value) AS avg_freight
    FROM
        `target_sql.customers` c JOIN `target_sql.orders` o ON c.customer_id = o.customer_id
    JOIN `target_sql.order_items` oi ON o.order_id = oi.order_id
    GROUP BY c.customer_state)
SELECT customer_state, avg_freight
FROM (SELECT * FROM AvgFreightPerState ORDER BY avg_freight DESC LIMIT 5) AS
highest_avg_freight
UNION ALL
SELECT customer_state,avg_freight
FROM (SELECT * FROM AvgFreightPerState ORDER BY avg_freight ASC LIMIT 5) AS
lowest_avg_freight;
```

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

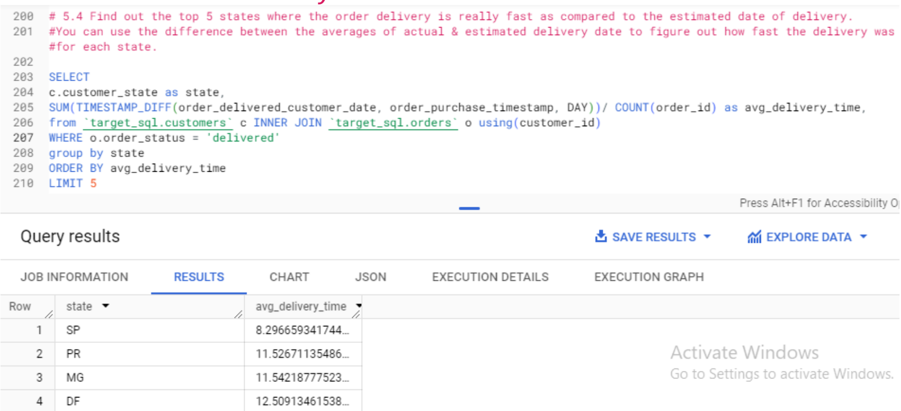


```
#5.4 Find out the top 5 states with the highest & lowest average delivery time.
WITH DeliveryTimePerState AS (
    SELECT c.customer_state,
        AVG(TIMESTAMP_DIFF (o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS avg_delivery_time
    FROM `target_sql.customers` c
    JOIN `target_sql.orders` o ON c.customer_id = o.customer_id
    WHERE o.order_status = 'delivered'
    GROUP BY c.customer_state
    )
SELECT customer_state,avg_delivery_time
FROM (SELECT * FROM DeliveryTimePerState ORDER BY avg_delivery_time DESC LIMIT 5) AS highest_avg_delivery_time
UNION ALL
SELECT customer_state,avg_delivery_time
FROM (SELECT * FROM DeliveryTimePerState ORDER BY avg_delivery_time ASC LIMIT 5) AS lowest_avg_delivery_time;
```

Row	customer_state	avg_delivery_time
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...

```
WITH DeliveryTimePerState AS (
    SELECT c.customer_state,
        AVG(TIMESTAMP_DIFF (o.order_delivered_customer_date, o.order_purchase_timestamp, DAY))
AS avg_delivery_time
    FROM `target_sql.customers` c
    JOIN `target_sql.orders` o ON c.customer_id = o.customer_id
    WHERE o.order_status = 'delivered'
    GROUP BY c.customer_state
    )
SELECT customer_state,avg_delivery_time
FROM (SELECT * FROM DeliveryTimePerState ORDER BY avg_delivery_time DESC LIMIT 5) AS
highest_avg_delivery_time
UNION ALL
SELECT customer_state,avg_delivery_time
```

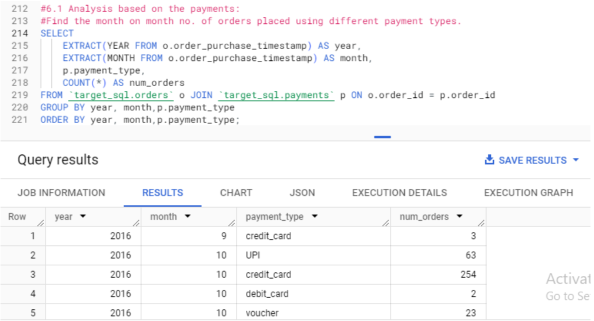
```
FROM (SELECT * FROM DeliveryTimePerState ORDER BY avg_delivery_time ASC LIMIT 5) AS
lowest_avg_delivery_time;


# 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.
```



```
# 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.

SELECT
c.customer_state as state,
SUM(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))/ COUNT(order_id) as avg_delivery_time,
from `target_sql.customers` c INNER JOIN `target_sql.orders` o using(customer_id)
WHERE o.order_status = 'delivered'
group by state
ORDER BY avg_delivery_time
LIMIT 5
```

Row	state	avg_delivery_time
1	SP	8.296659341744...
2	PR	11.52671135486...
3	MG	11.54218777523...
4	DF	12.50913461538...

```
SELECT
c.customer_state as state,
SUM(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))/
COUNT(order_id) as avg_delivery_time,
from `target_sql.customers` c INNER JOIN `target_sql.orders` o using(customer_id)
WHERE o.order_status = 'delivered'
group by state
ORDER BY avg_delivery_time
LIMIT 5


#6.1 Analysis based on the payments:
#Find the month on month no. of orders placed using different payment types.
```



```
#6.1 Analysis based on the payments:
#Find the month on month no. of orders placed using different payment types.
SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
    p.payment_type,
    COUNT(*) AS num_orders
FROM `target_sql.orders` o JOIN `target_sql.payments` p ON o.order_id = p.order_id
GROUP BY year, month,p.payment_type
ORDER BY year, month,p.payment_type;
```

Row	year	month	payment_type	num_orders
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	254
4	2016	10	debit_card	2
5	2016	10	voucher	23

```
SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
    p.payment_type,
    COUNT(*) AS num_orders
FROM `target_sql.orders` o JOIN `target_sql.payments` p ON o.order_id = p.order_id
GROUP BY year, month,p.payment_type
```

```
ORDER BY year, month,p.payment_type;
```

BUSINESS INSIGHT

1.Geographical Insights:

Identification of regions with high customer concentration: Understanding which states or cities have the highest number of customers and orders can help Target allocate resources effectively, tailor marketing efforts, and optimize inventory management for specific regions.

Geographic trends in order volume: Recognizing patterns in order volume across different states and cities over time can inform Target's expansion strategies, promotional campaigns, and logistical operations.

2.Trend Analysis:

Seasonal trends in order volume: Recognizing monthly or yearly patterns in order volume can help Target anticipate demand fluctuations, adjust inventory levels accordingly, and plan marketing campaigns to capitalize on peak seasons.

Time-of-day ordering patterns: Identifying the most common times of day when orders are placed can optimize staffing schedules, logistics, and customer support resources to ensure efficient order processing and delivery.

3.Economic Impact Analysis:

Cost analysis and pricing strategies: Analyzing the percentage increase in order costs over time can inform pricing strategies, cost management initiatives, and decisions regarding discounts, promotions, or adjustments to product pricing.

Freight cost optimization: Understanding the distribution of freight costs across different states can help Target negotiate better shipping rates, optimize delivery routes, and implement strategies to reduce shipping expenses while maintaining service quality.

4.Sales and Delivery Analysis:

Delivery performance and customer satisfaction: Monitoring delivery times and comparing them to estimated delivery dates can provide insights into operational efficiency, customer satisfaction levels, and areas for improvement in delivery logistics and service quality.

Regional differences in delivery performance: Identifying states with the fastest or slowest delivery times can help Target focus improvement efforts, address logistical challenges, and enhance customer experiences in specific regions.

5.Payment Analysis:

Payment method preferences: Understanding the popularity of different payment methods among customers can inform payment processing strategies, partnerships with payment providers, and initiatives to enhance payment security and convenience.

Revenue attribution by payment type: Analyzing revenue generated through various payment methods can help Target assess the effectiveness of marketing campaigns, loyalty programs, and incentives tied to specific payment options.

These insights can guide Target's decision-making processes, resource allocation, and strategic initiatives to optimize operational efficiency, enhance customer experiences, and drive business growth in the Brazilian market.