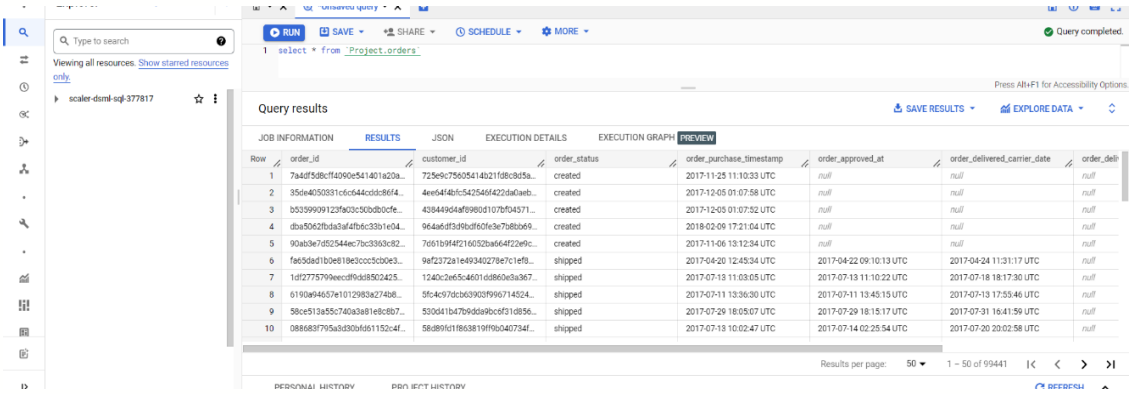
1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset
    1. Data type of columns in a table

```
select * from `Project.orders`
```



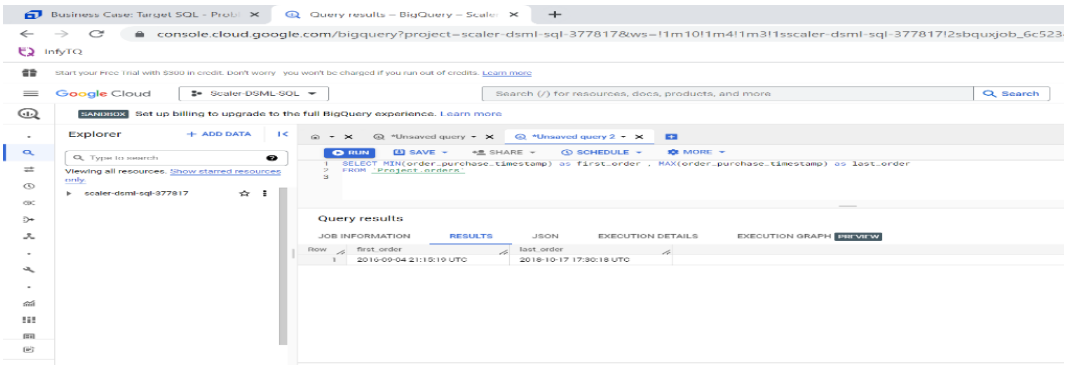
I have used one table name just to show the data inside it

2. Time period for which the data is given

The time period for which the data is available in the orders.csv file

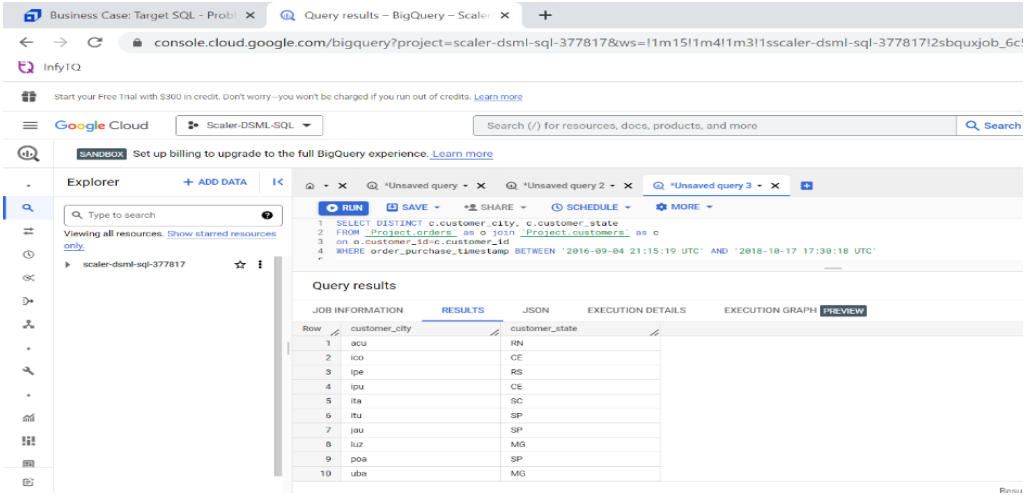
```
SELECT MIN(order_purchase_timestamp) as first_order , MAX(order_purchase_timestamp) as
 last_order

FROM `Project.orders`
```



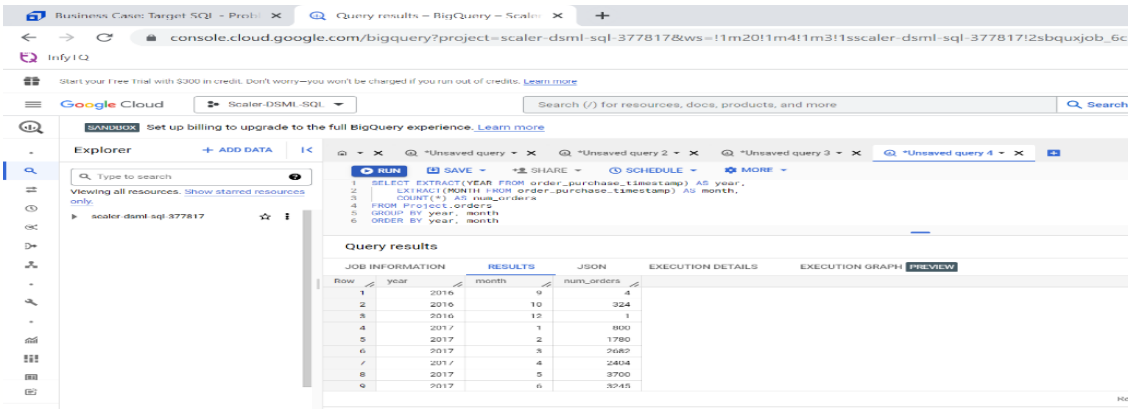
3. Cities and States of customers ordered during the given period

```
SELECT DISTINCT c.customer_city, c.customer_state
FROM `Project.orders` as o join `Project.customers` as c
on o.customer_id=c.customer_id
WHERE order_purchase_timestamp BETWEEN '2016-09-04 21:15:19 UTC' AND '2018-10-
17 17:30:18 UTC'
```



2. In-depth Exploration:
    1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

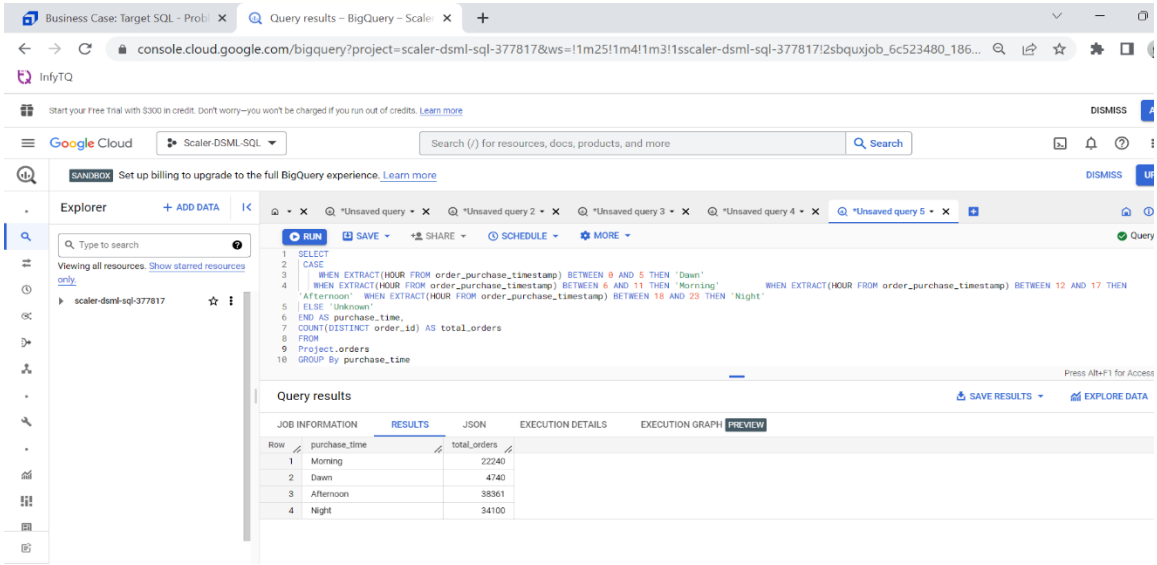
```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
COUNT(*) AS num_orders
FROM Project.orders
GROUP BY year, month
ORDER BY year, month
```



    2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

```
SELECT
 CASE
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 0 AND 5 THEN 'Dawn'
  WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 6 AND 11 THEN 'Morning'
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 12 AND 17 THEN 'Afternoon'
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 18 AND 23 THEN 'Night'
 ELSE 'Unknown'
```

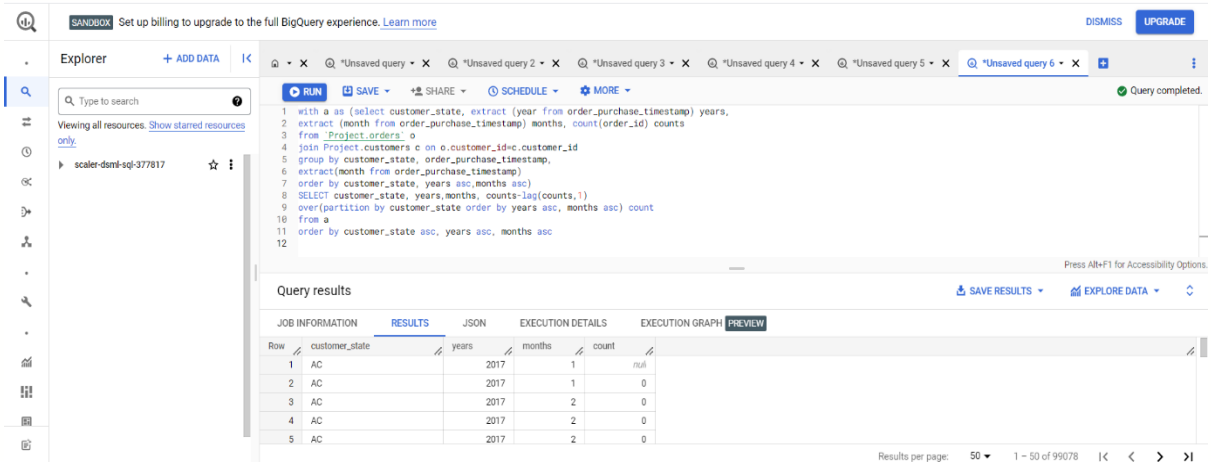
```
END AS purchase_time,
COUNT(DISTINCT order_id) AS total_orders
FROM
Project.orders
GROUP By purchase_time
```



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

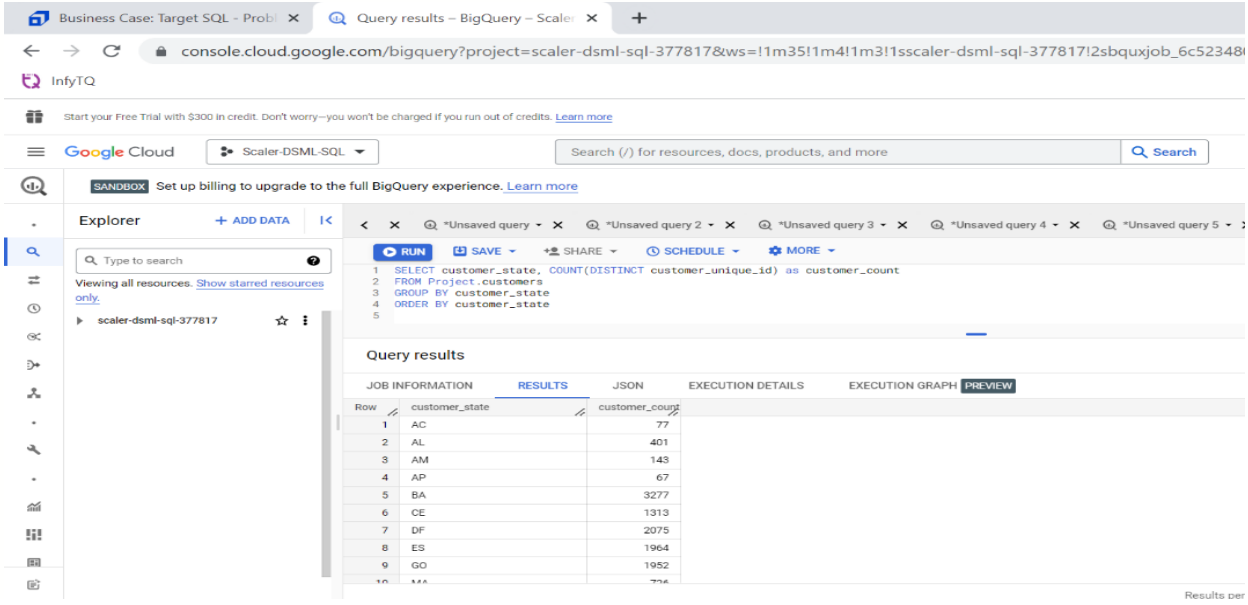
1.Get month on month orders by states

```
with a as (select customer_state, extract (year from order_purchase_timestamp) years,
extract (month from order_purchase_timestamp) months, count(order_id) counts
from `Project.orders` o
join Project.customers c on o.customer_id=c.customer_id
group by customer_state, order_purchase_timestamp,
extract(month from order_purchase_timestamp)
order by customer_state, years asc,months asc)
SELECT customer_state, years,months, counts-lag(counts,1)
over(partition by customer_state order by years asc, months asc) count
from a
order by customer_state asc, years asc, months asc
```



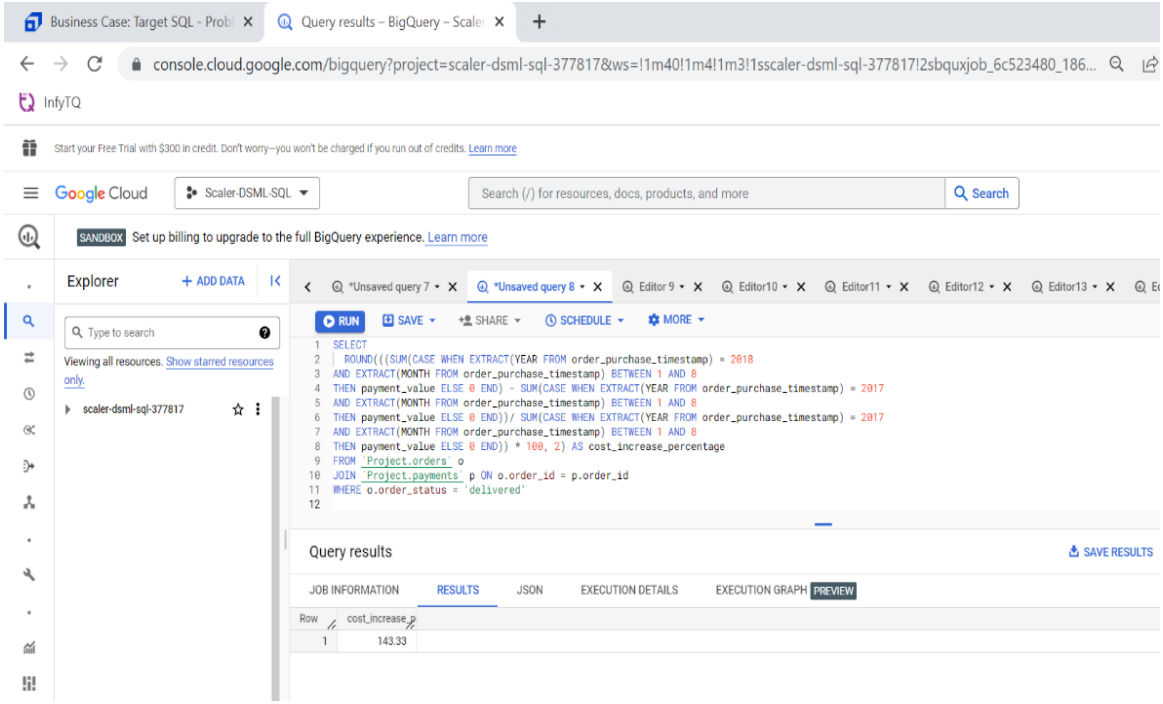
2. Distribution of customers across the states in Brazil

```
SELECT customer_state, COUNT(DISTINCT customer_unique_id) as customer_count
FROM Project.customers
GROUP BY customer_state
ORDER BY customer_state
```



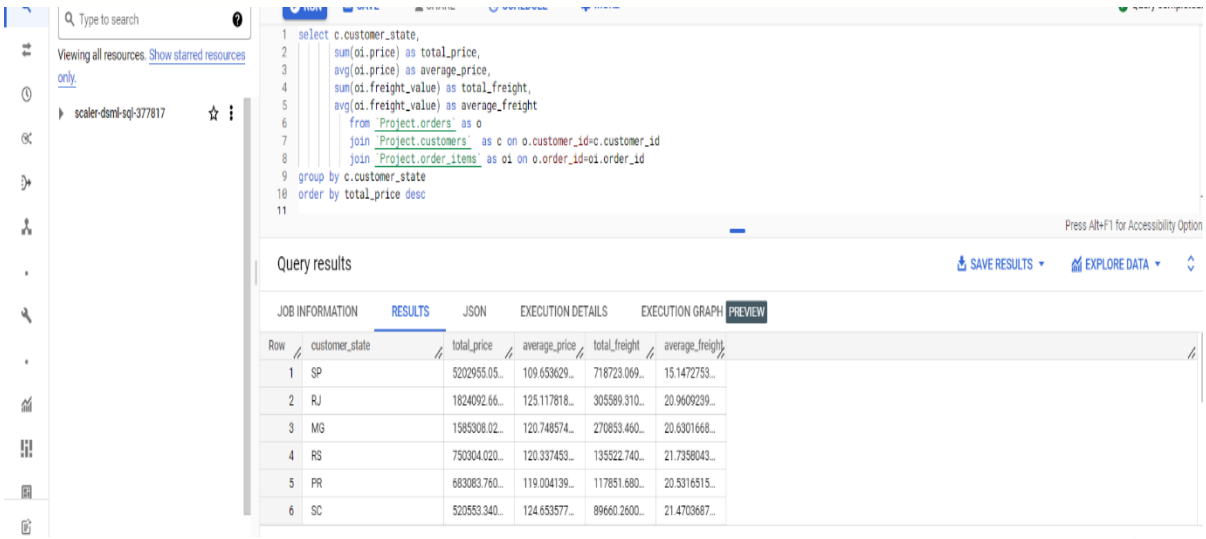
4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.
    1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

```
SELECT
  ROUND(((SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp) = 2018
 AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8
 THEN payment_value ELSE 0 END) - SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_tim
estamp) = 2017
 AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8
 THEN payment_value ELSE 0 END))/ SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_tim
estamp) = 2017
AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8
THEN payment_value ELSE 0 END)) * 100, 2) AS cost_increase_percentage
FROM `Project.orders` o
JOIN `Project.payments` p ON o.order_id = p.order_id
WHERE o.order_status = 'delivered'
```



2. Mean & Sum of price and freight value by customer state

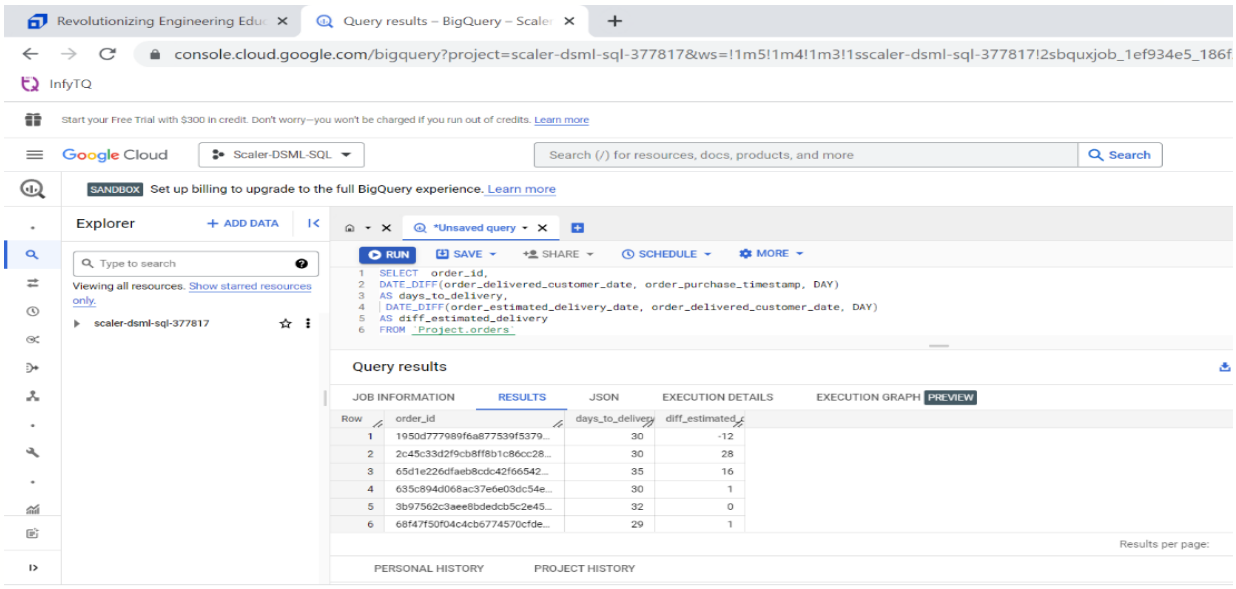
```
select c.customer_state,
    sum(oi.price) as total_price,
    avg(oi.price) as average_price,
    sum(oi.freight_value) as total_freight,
    avg(oi.freight_value) as average_freight
      from `Project.orders` as o
      join `Project.customers`  as c on o.customer_id=c.customer_id
      join `Project.order_items` as oi on o.order_id=oi.order_id
group by c.customer_state
order by total_price desc
```



5. Analysis on sales, freight and delivery time

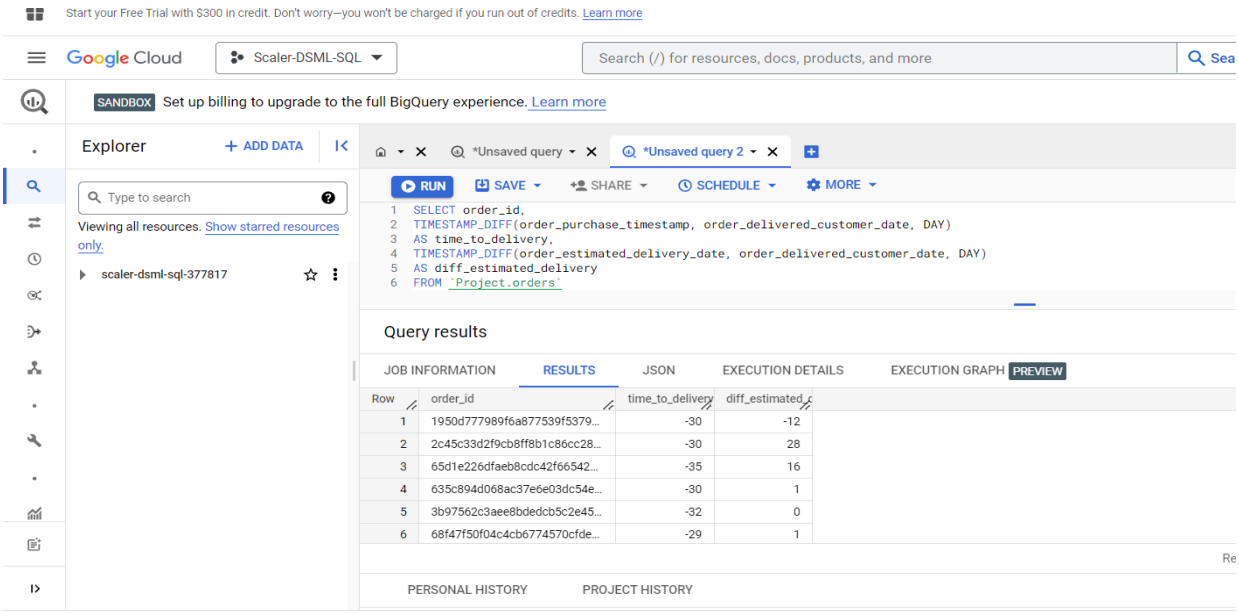
1. Calculate days between purchasing, delivering and estimated delivery

```
SELECT  order_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)
AS days_to_delivery,
 DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)
AS diff_estimated_delivery
FROM `Project.orders`
```



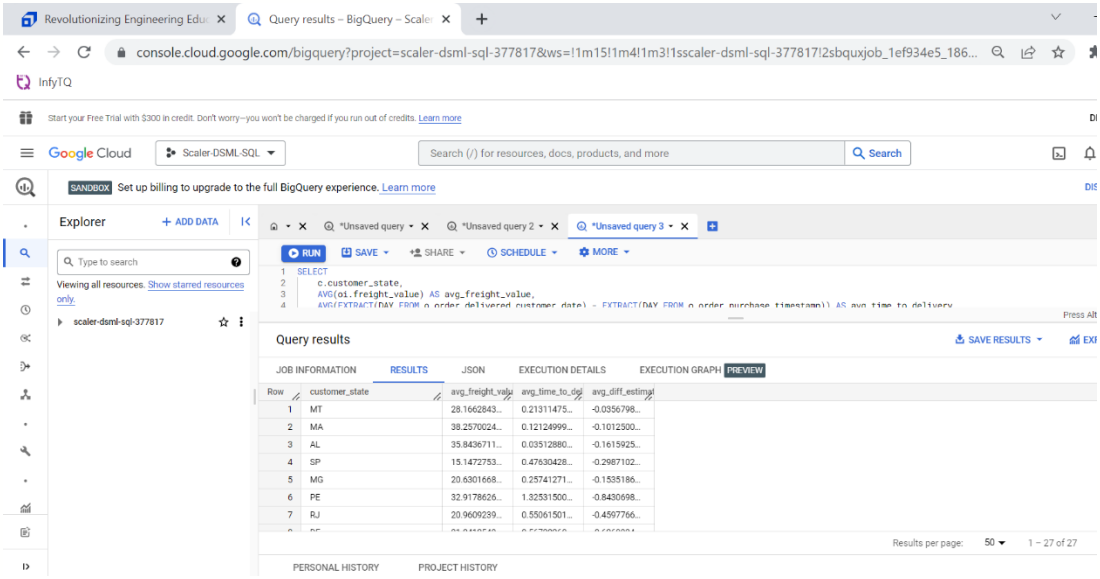
2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:
   - time_to_delivery = order_purchase_timestamp- order_delivered_customer_date
   - diff_estimated_delivery = order_estimated_delivery_date- order_delivered_customer_date

```
SELECT order_id,
TIMESTAMP_DIFF(order_purchase_timestamp, order_delivered_customer_date, DAY)
AS time_to_delivery,
TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)
AS diff_estimated_delivery
FROM `Project.orders`
```



3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

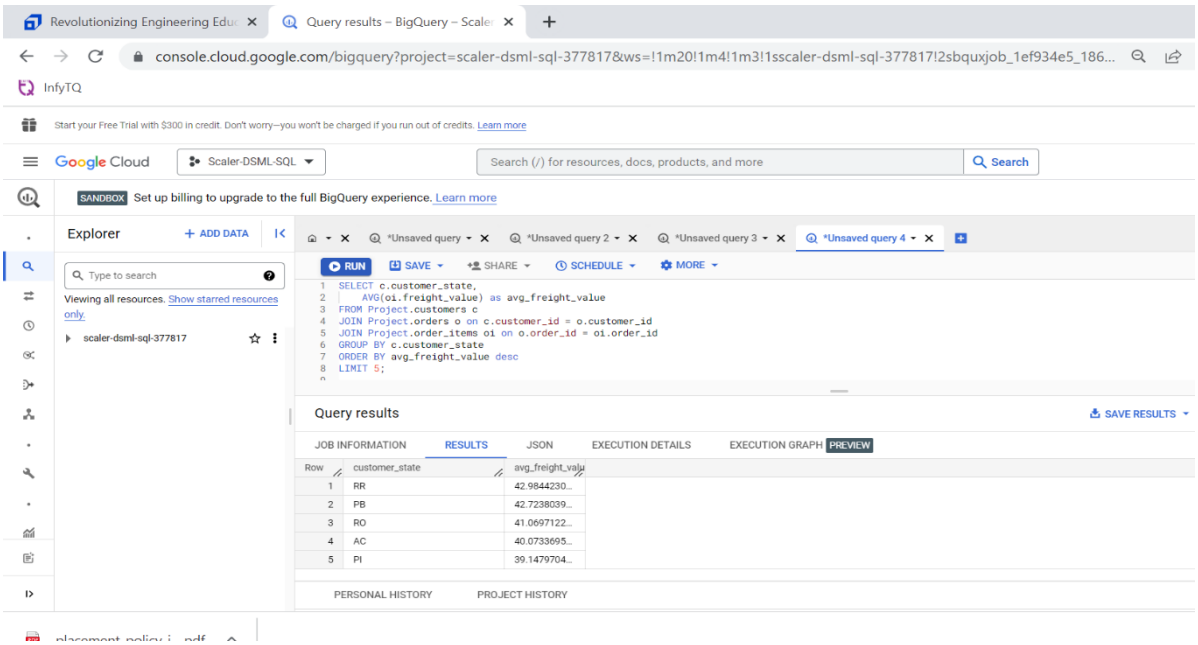
```
SELECT
    c.customer_state,
    AVG(oi.freight_value) AS avg_freight_value,
    AVG(EXTRACT(DAY FROM o.order_delivered_customer_date) - EXTRACT(DAY FROM o.order_purchase_timestamp)) AS avg_time_to_delivery,
    AVG(EXTRACT(DAY FROM o.order_estimated_delivery_date) - EXTRACT(DAY FROM o.order_delivered_customer_date)) AS avg_diff_estimated_delivery
FROM
    Project.customers c
    JOIN Project.orders o ON c.customer_id = o.customer_id
    JOIN Project.order_items oi ON o.order_id = oi.order_id
GROUP BY
    c.customer_state
```



4. Sort the data to get the following:

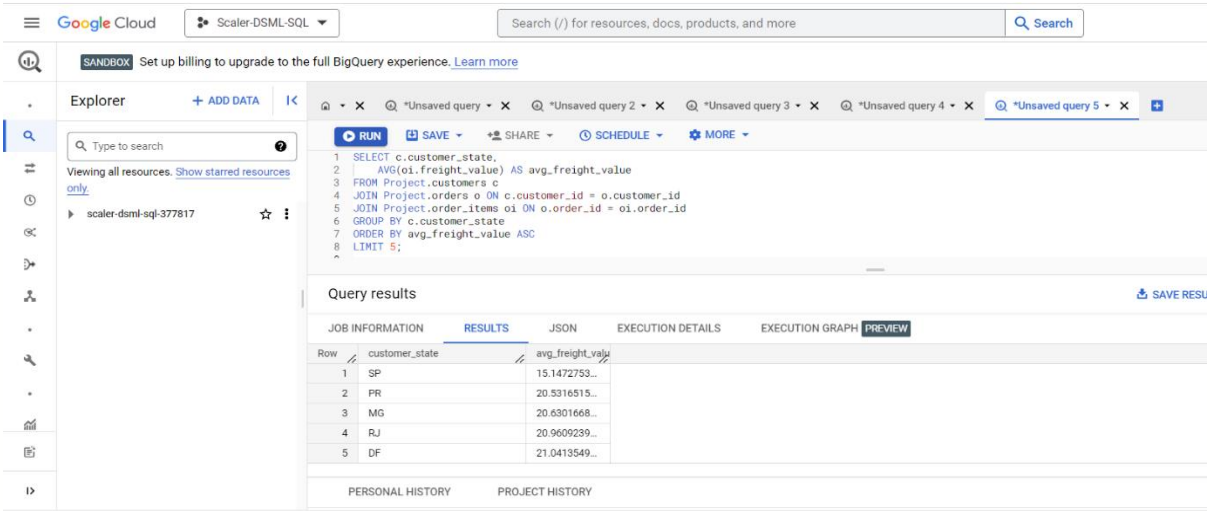
5. 1.Top 5 states with highest average freight value in desc

```
SELECT c.customer_state,
   AVG(oi.freight_value) as avg_freight_value
FROM Project.customers c
JOIN Project.orders o on c.customer_id = o.customer_id
JOIN Project.order_items oi on o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_freight_value desc
LIMIT 5;
```



2.Top 5 states with lowest average freight value in asc

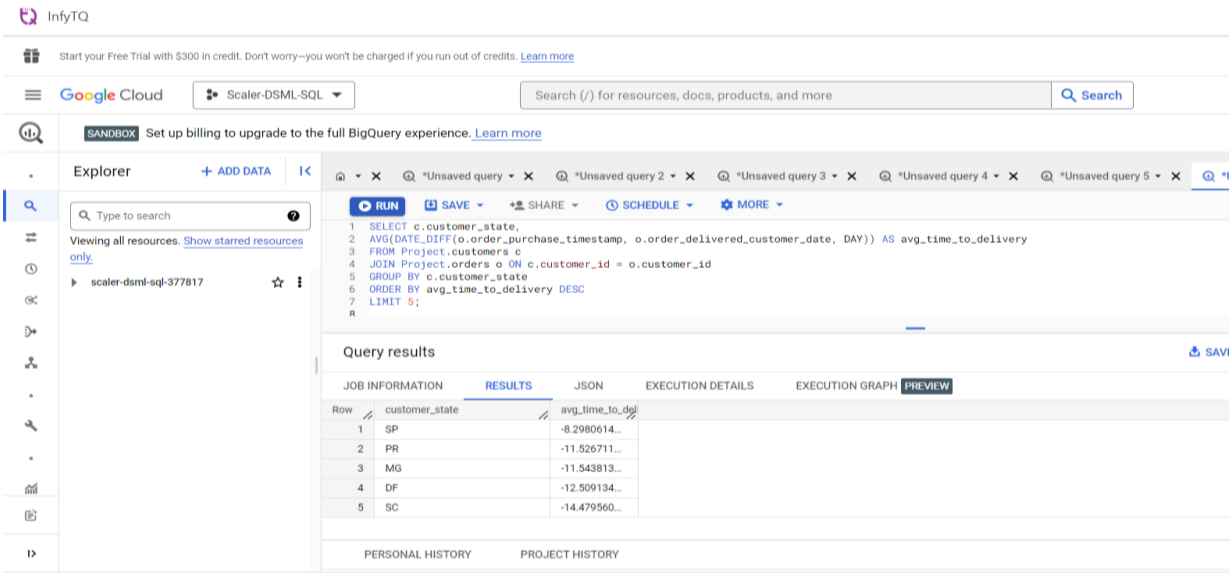
```
SELECT c.customer_state,
   AVG(oi.freight_value) AS avg_freight_value
FROM Project.customers c
JOIN Project.orders o ON c.customer_id = o.customer_id
JOIN Project.order_items oi ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_freight_value ASC
LIMIT 5;
```



6. Top 5 states with highest/lowest average time to delivery

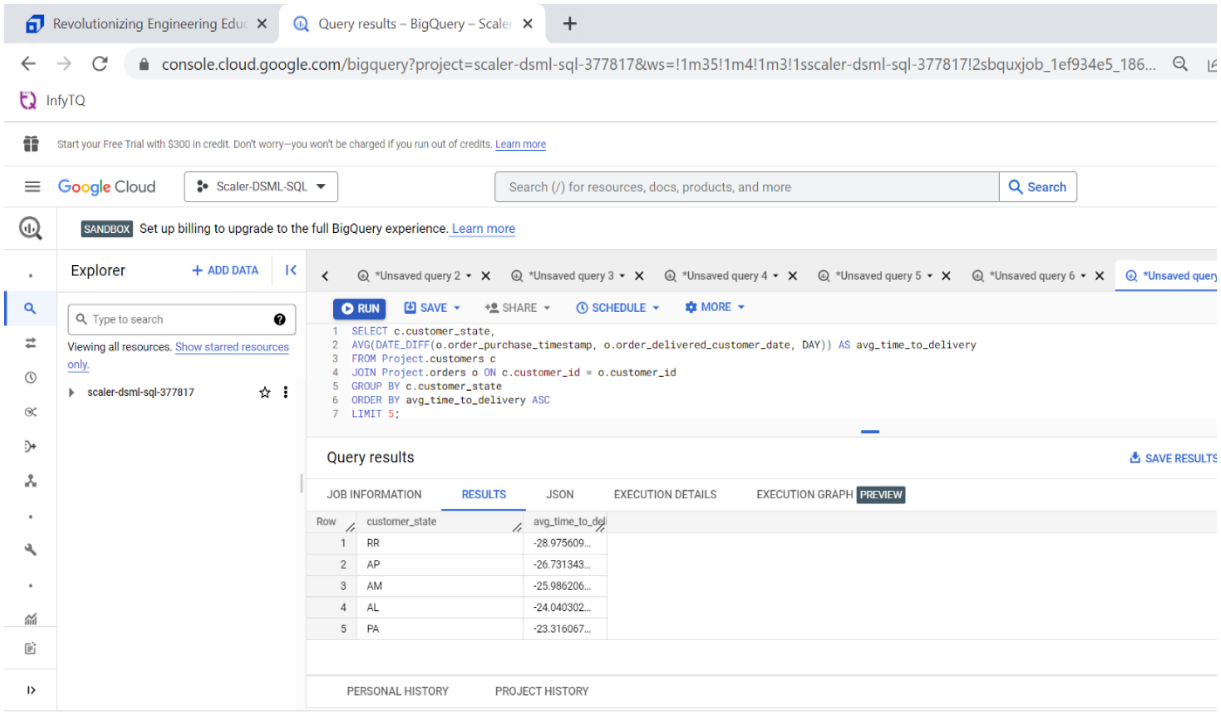
6.1 Top 5 states with highest average time to delivery

```
SELECT c.customer_state,
AVG(DATE_DIFF(o.order_purchase_timestamp, o.order_delivered_customer_date, DAY)) AS avg_time_to_delivery
FROM Project.customers c
JOIN Project.orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery DESC
LIMIT 5;
```



6.2 Top 5 states with lowest average time to delivery

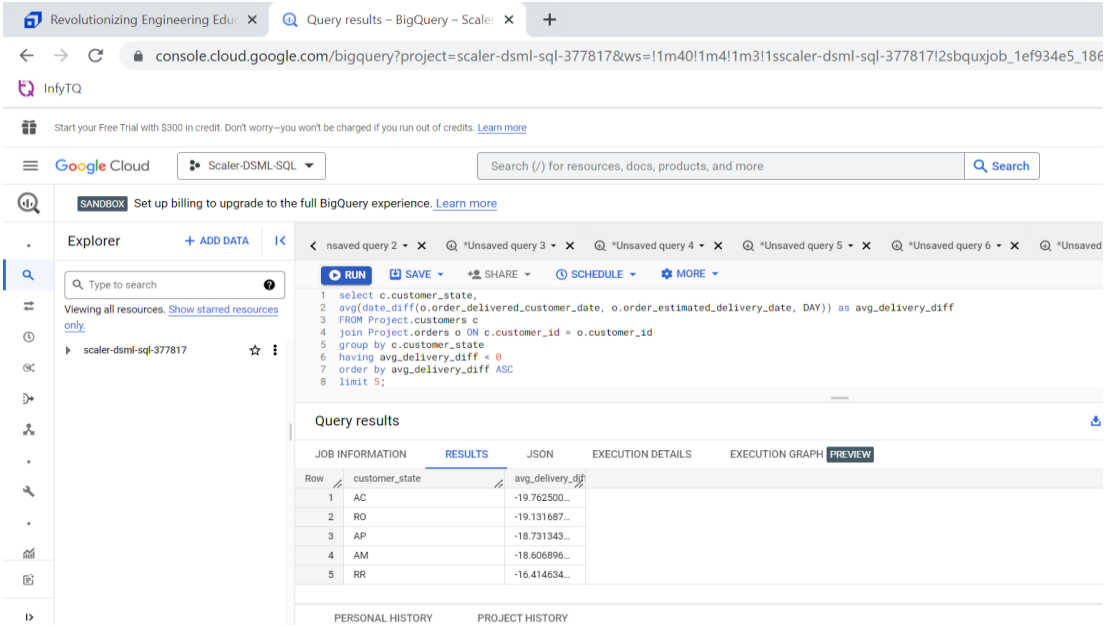
```
SELECT c.customer_state,
AVG(DATE_DIFF(o.order_purchase_timestamp, o.order_delivered_customer_date, DAY)) AS avg_time_to_deliv
ery
FROM Project.customers c
JOIN Project.orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery ASC
LIMIT 5;
```



7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

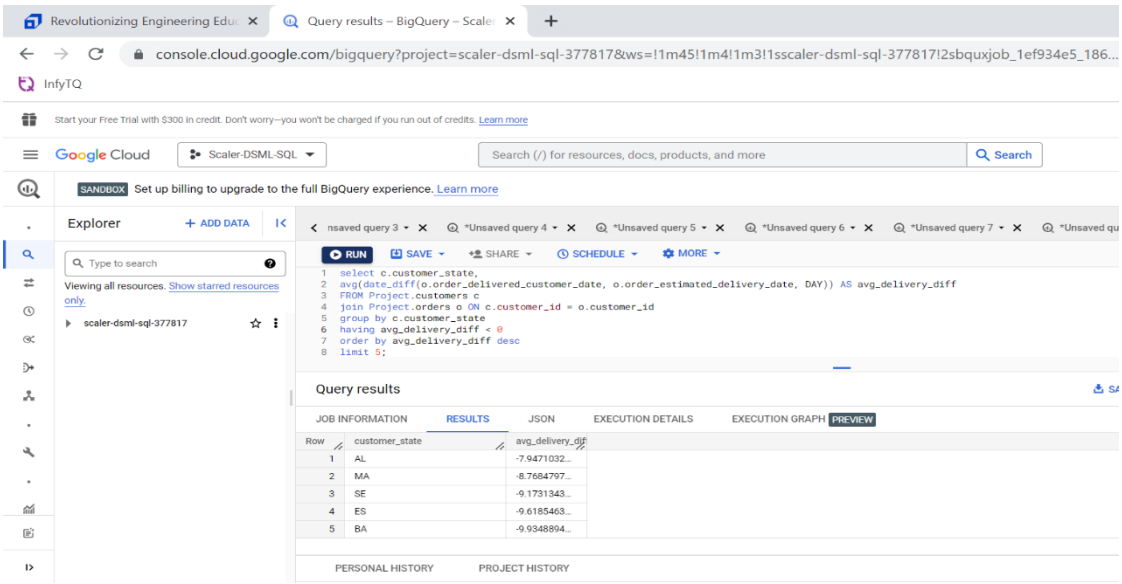
   7.1 Top 5 states where delivery is really fast as compared to estimated date

```
select c.customer_state,
avg(date_diff(o.order_delivered_customer_date, o.order_estimated_delivery_date, DAY)) as avg_delivery_diff
FROM Project.customers c
join Project.orders o ON c.customer_id = o.customer_id
group by c.customer_state
having avg_delivery_diff < 0
order by avg_delivery_diff ASC
limit 5;
```



7.2 Top 5 states where delivery is really not so fast as compared to estimated date

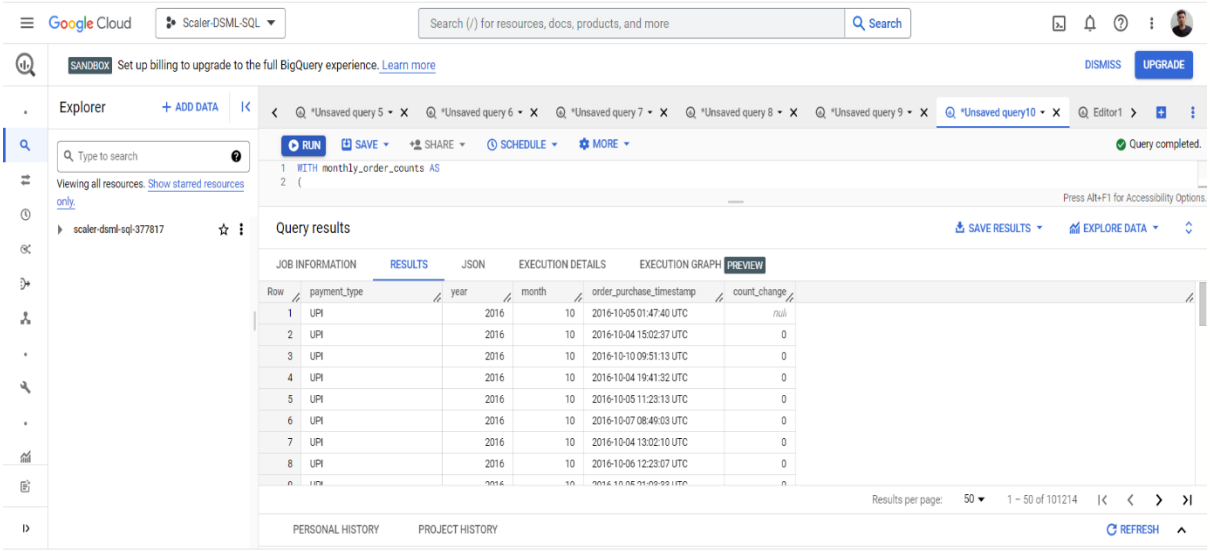
```
select c.customer_state,
avg(date_diff(o.order_delivered_customer_date, o.order_estimated_delivery_date, DAY)) AS avg_delivery_diff
FROM Project.customers c
join Project.orders o ON c.customer_id = o.customer_id
group by c.customer_state
having avg_delivery_diff > 0
order by avg_delivery_diff DESC
limit 5;
```



6. Payment type analysis:

1. Month over Month count of orders for different payment types

```
WITH monthly_order_counts AS
(
  SELECT p.payment_type,
    EXTRACT(YEAR FROM o.order_purchase_timestamp) as year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) as month,
    o.order_purchase_timestamp,
    COUNT(o.order_id) as count
FROM Project.orders as o join Project.payments as p ON o.order_id = p.order_id
  GROUP BY
    p.payment_type,
    EXTRACT(YEAR FROM o.order_purchase_timestamp),
    EXTRACT(MONTH FROM o.order_purchase_timestamp),
    o.order_purchase_timestamp
  ORDER BY p.payment_type, year, month
)
SELECT m.payment_type, m.year, m.month, m.order_purchase_timestamp,
 m.count - LAG(m.count, 1) OVER (PARTITION BY m.payment_type ORDER BY m.year asc, m.month
ASC) as count_change
FROM monthly_order_counts as m
ORDER BY m.payment_type, m.year, m.month
```



2. Count of orders based on the no. of payment installments

```
SELECT
   payment_installments,
   COUNT(*) AS order_counts
FROM `Project.payments`
GROUP BY
  payment_installments
ORDER BY
  payment_installments
```