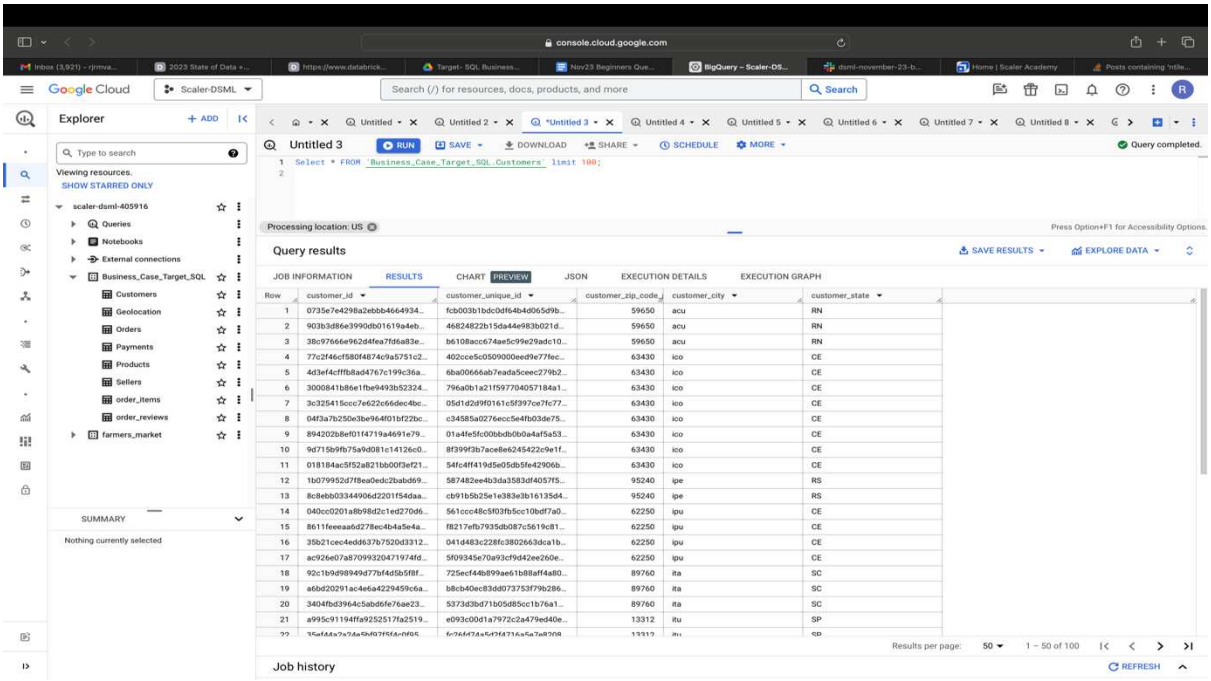
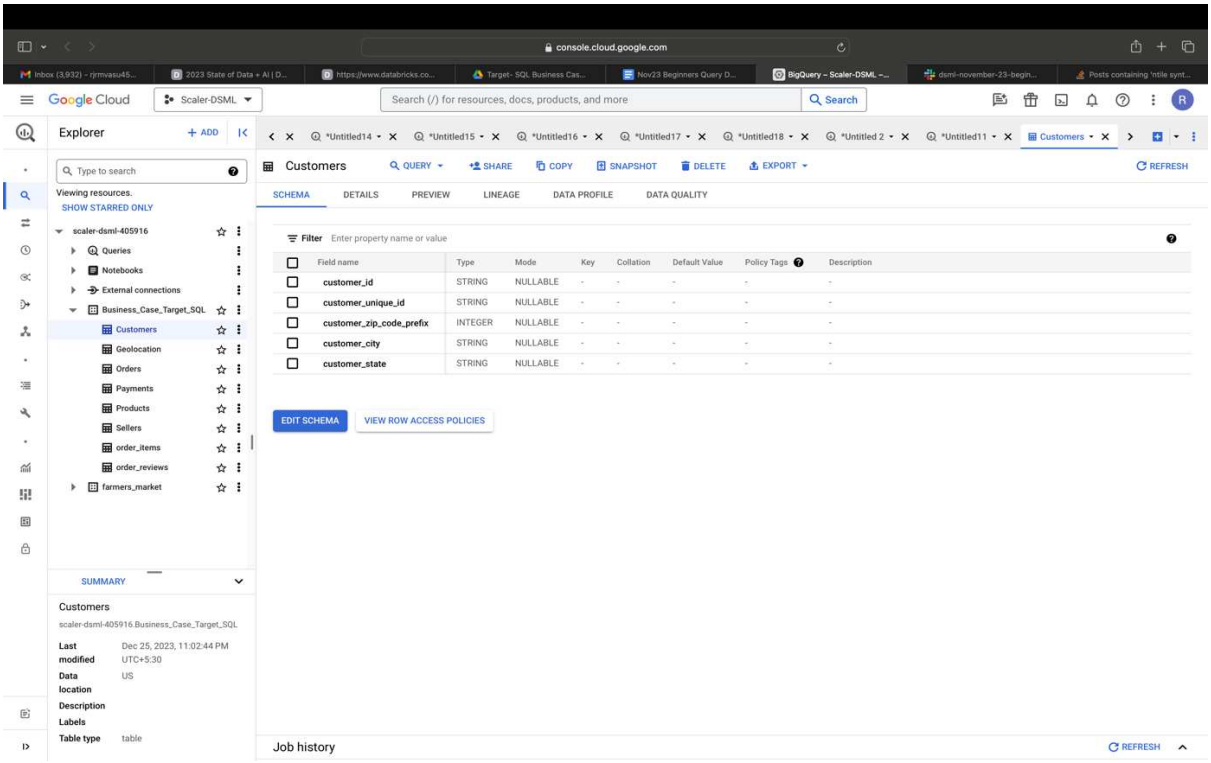
1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**

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

Code: Select * FROM `Business_Case_Target_SQL.Customers` limit 100;

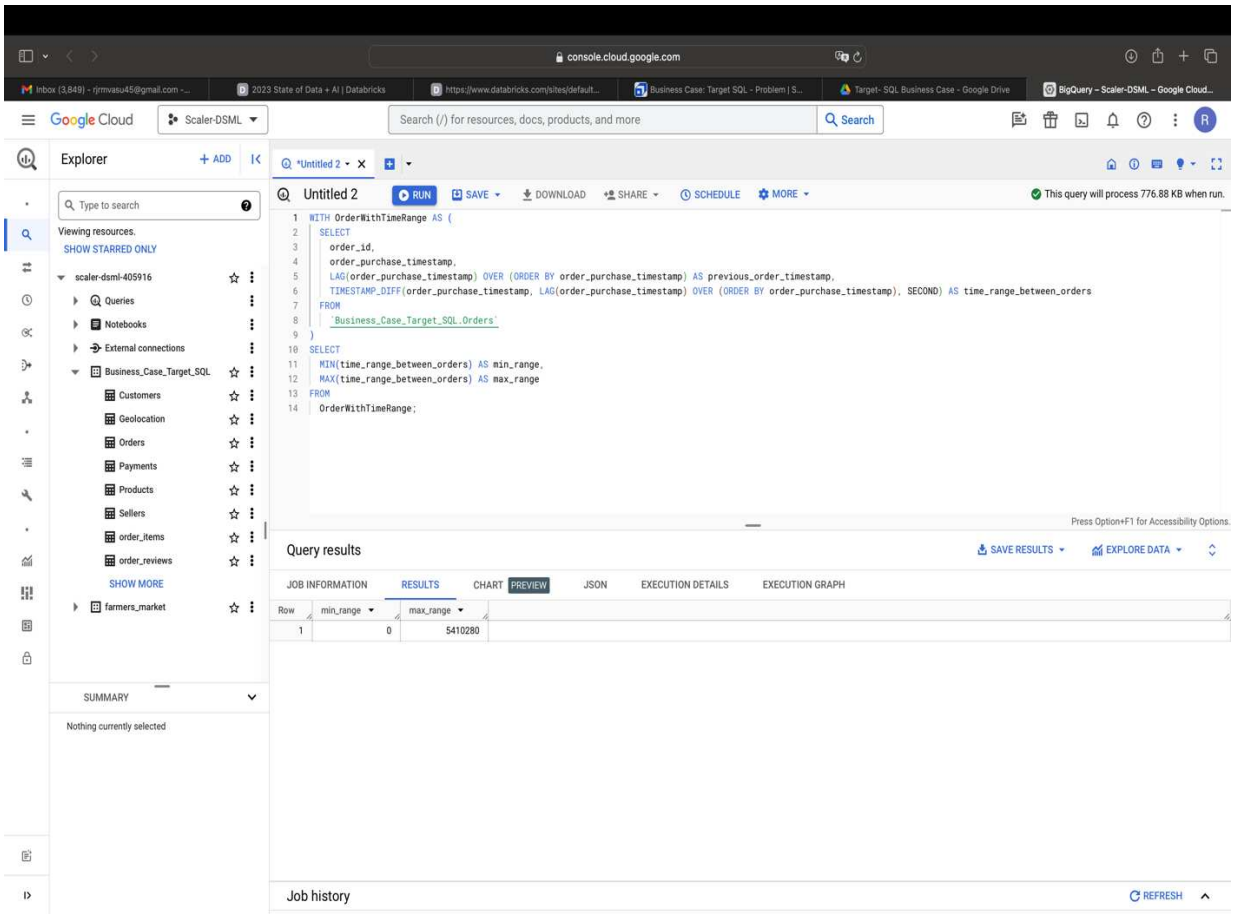




1.2 Get the time range between which the orders were placed.

Code:

```
WITH OrderWithTimeRange AS (
SELECT
order_id,
order_purchase_timestamp,
LAG(order_purchase_timestamp) OVER (ORDER BY order_purchase_timestamp) AS previous_order_timestamp,
TIMESTAMP_DIFF(order_purchase_timestamp, LAG(order_purchase_timestamp) OVER (ORDER BY order_purchase_timestamp), HOUR) AS time_range_between_orders
FROM
`Business_Case_Target_SQL.Orders`
)
SELECT
MIN(time_range_between_orders) AS min_range,
MAX(time_range_between_orders) AS max_range
FROM
OrderWithTimeRange;
```

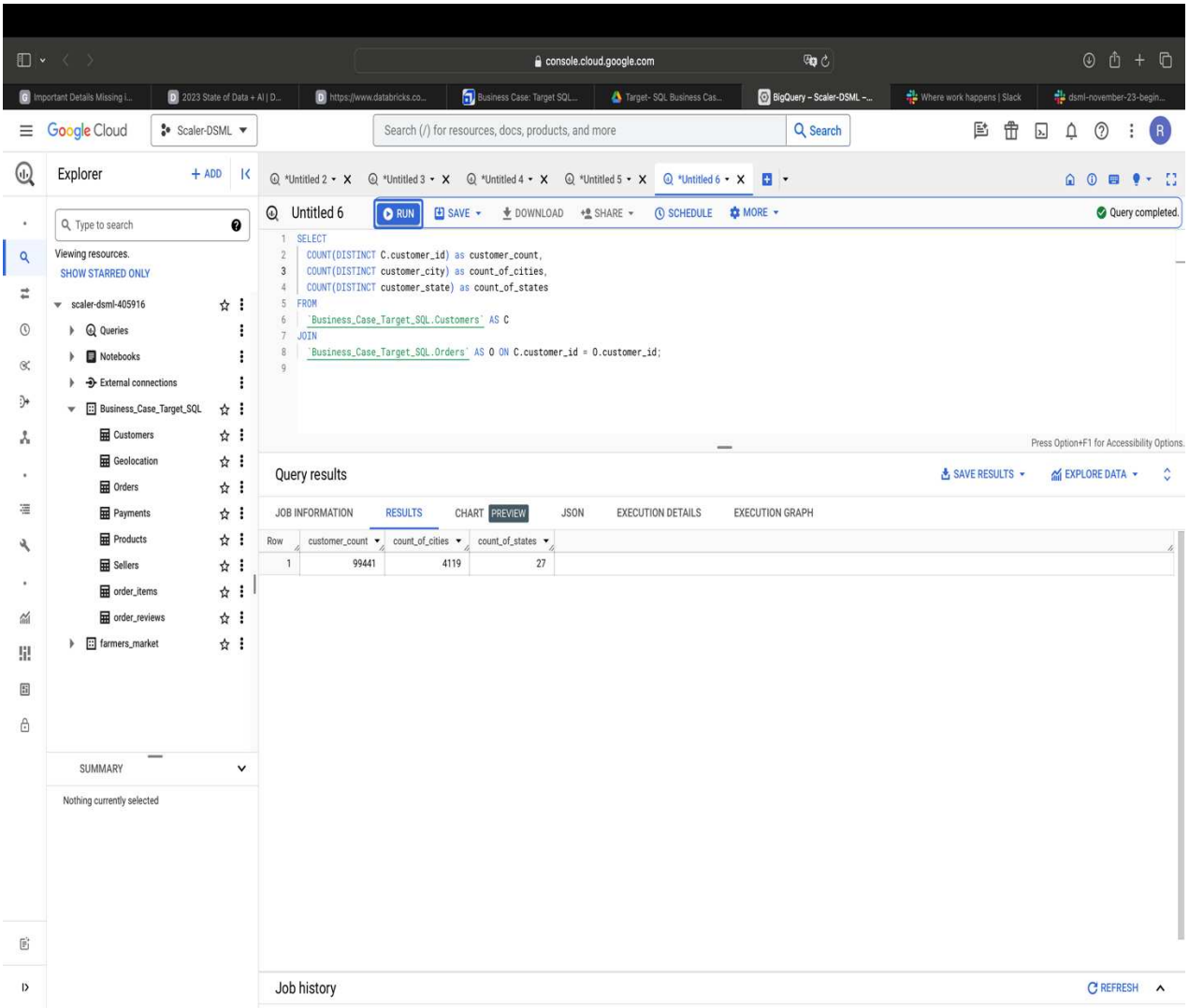


Comment: *Many orders were placed at the same time and hence the difference is zero at the lower end and 62 days is the difference at the higher end. The above query I have used 'Seconds' to calculate the time difference.*

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

Code:

```
SELECT
  COUNT(DISTINCT C.customer_id) as customer_count,
  COUNT(DISTINCT customer_city) as count_of_cities,
  COUNT(DISTINCT customer_state) as count_of_states
FROM
  `Business_Case_Target_SQL.Customers` AS C
JOIN
  `Business_Case_Target_SQL.Orders` AS O ON C.customer_id = O.customer_id;
```

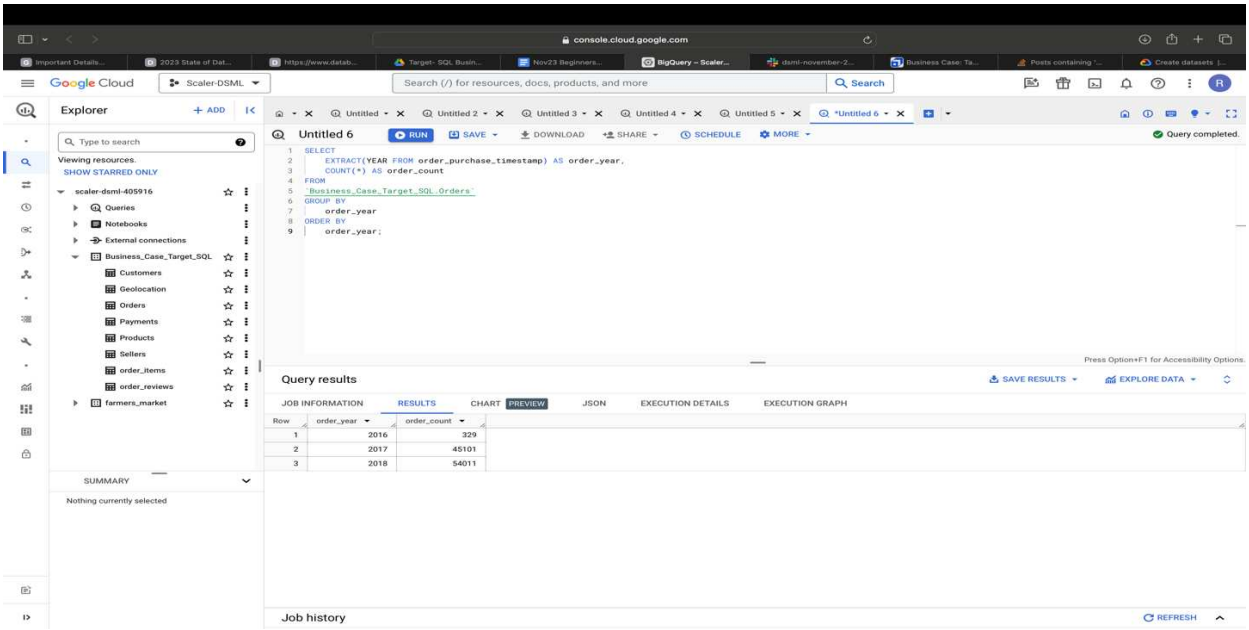


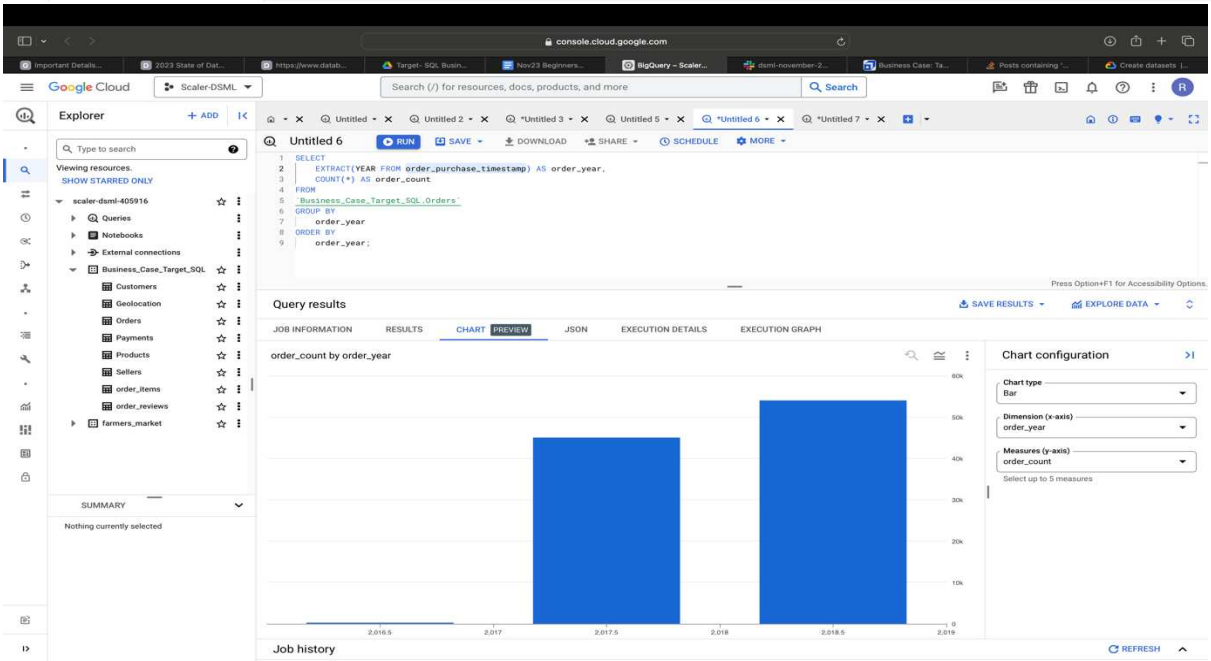
Comment:In total 99441 customers placed orders from 4119 cities which are located in 27 states in Brazil.

2.In-depth Exploration:

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

```
SELECT
    EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
    COUNT(*) AS order_count
FROM
`    Business_Case_Target_SQL.Orders`
GROUP BY
    order_year
ORDER BY
    order_year;
```



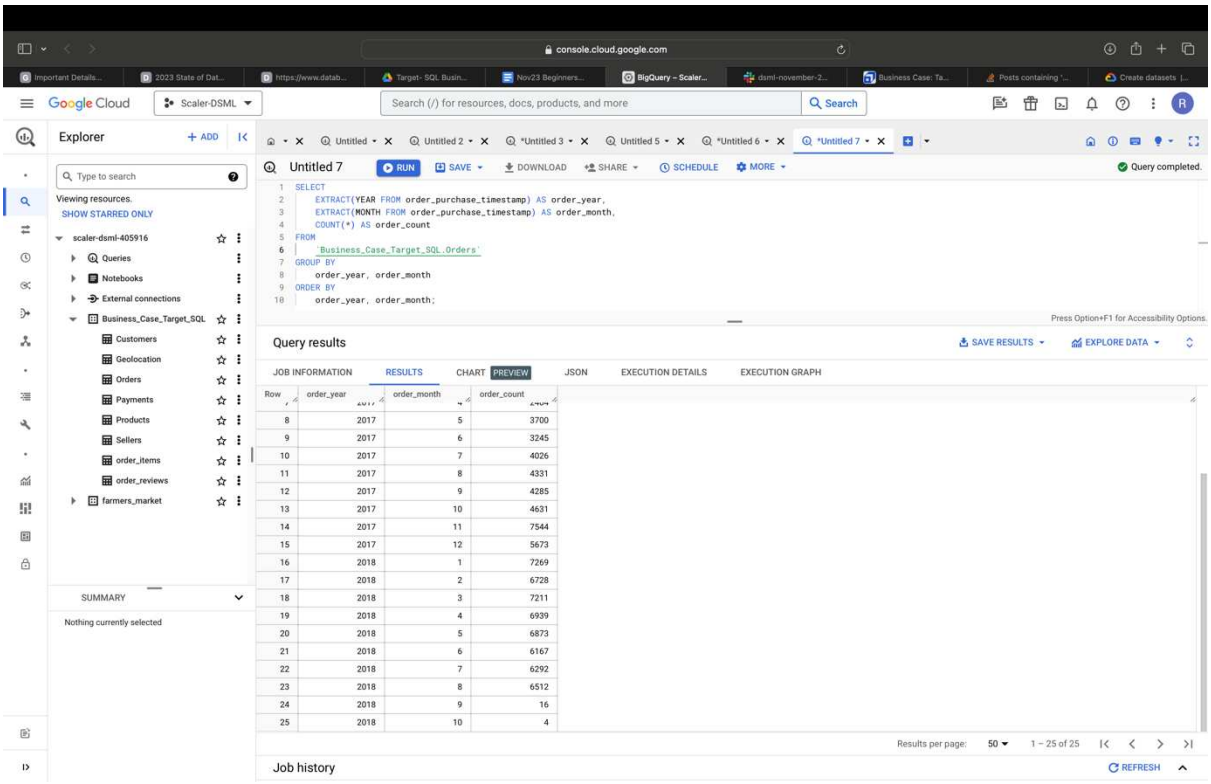


Comment: There is a 19.7% increase in no. of orders from 2017 to 2018 in spite of a massive dip in orders for Sep and Oct of 2018. Also we have compared 12 months of 2017 data compared with 10 months 0f 2018 data.Not enough data available for 2016 to be compared.

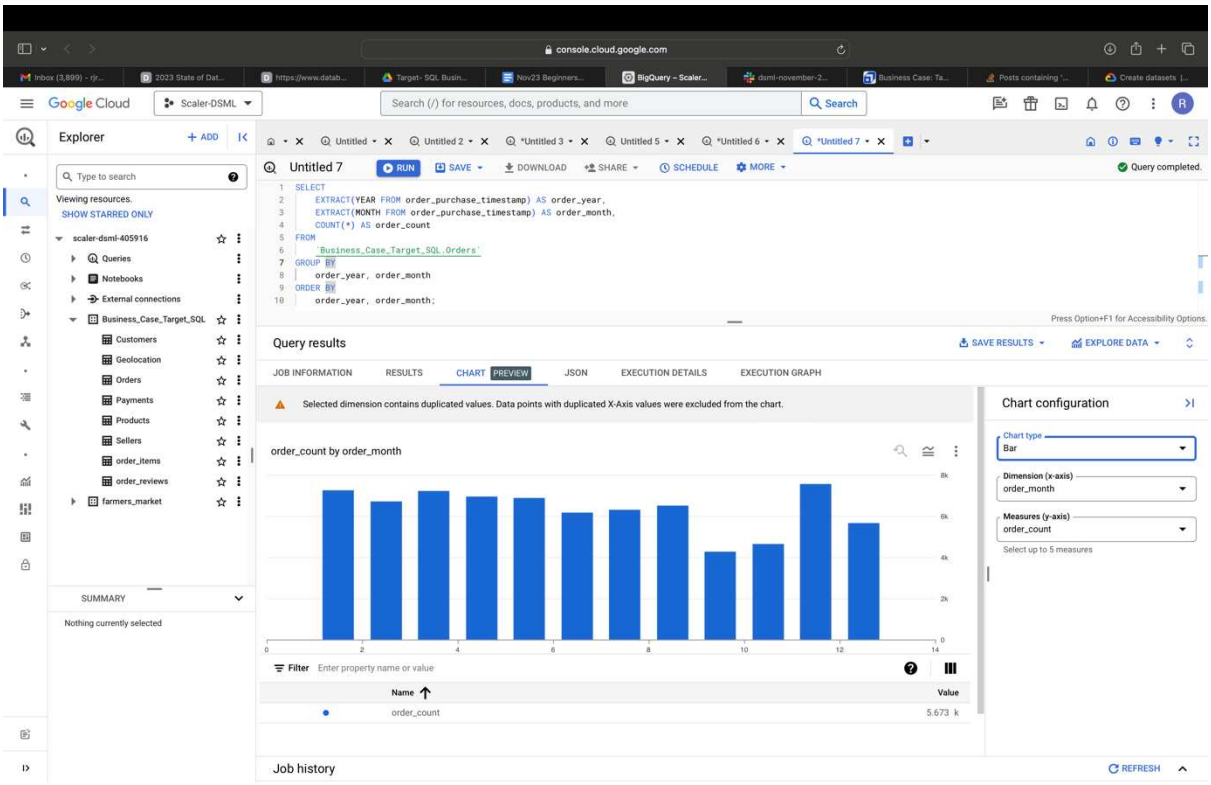
2.2 Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

Code:

```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
  EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
  COUNT(*) AS order_count
FROM
  `Business_Case_Target_SQL.Orders`
GROUP BY
  order_year, order_month
ORDER BY
  order_year, order_month;
```



Comment: From Feb 2017 there was a steady increase in the no. of orders placed moving from 2000 orders per month to 7500 in the month of Nov 2017 and after a slight fall in Dec 2017 the orders steadied at 6000+ range till August 2018. Not sure about the double digit and single digit order recorded for the month of Sep and Oct 2018 which also resembles the trend during the Sep, Oct and Dec data of 2016.

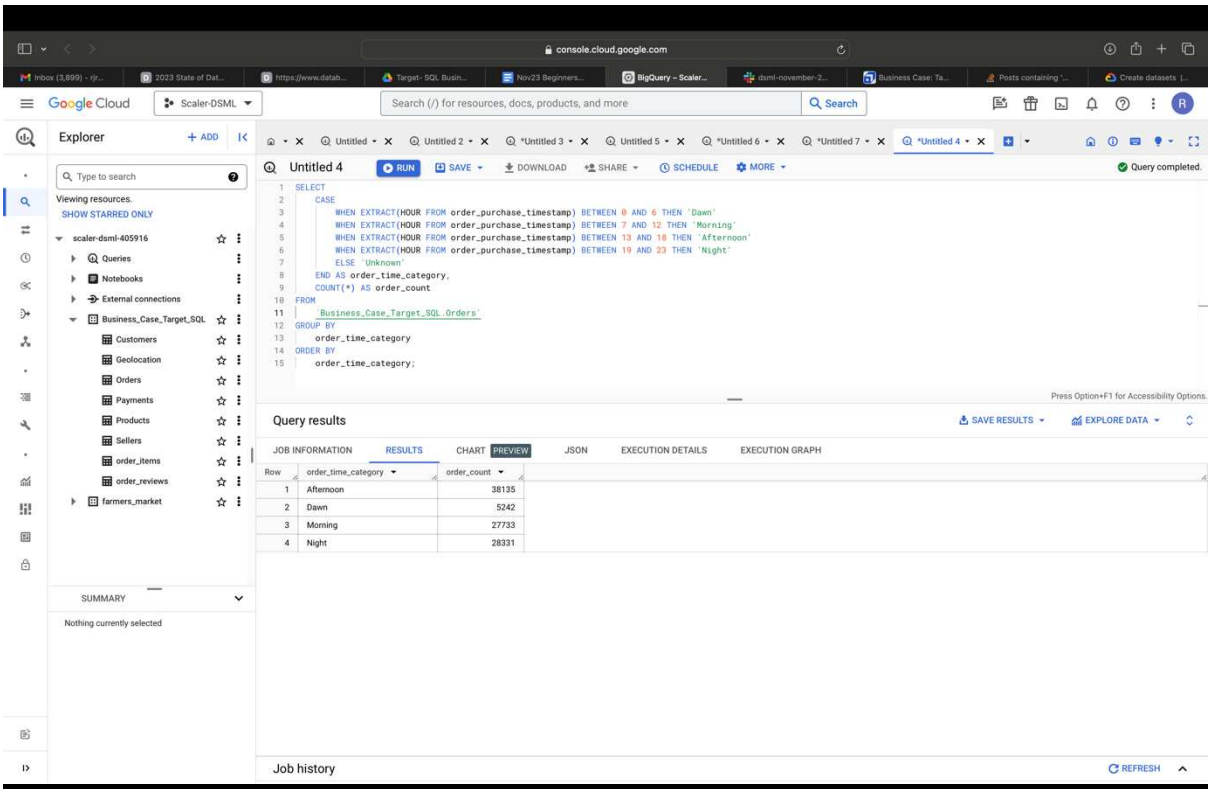


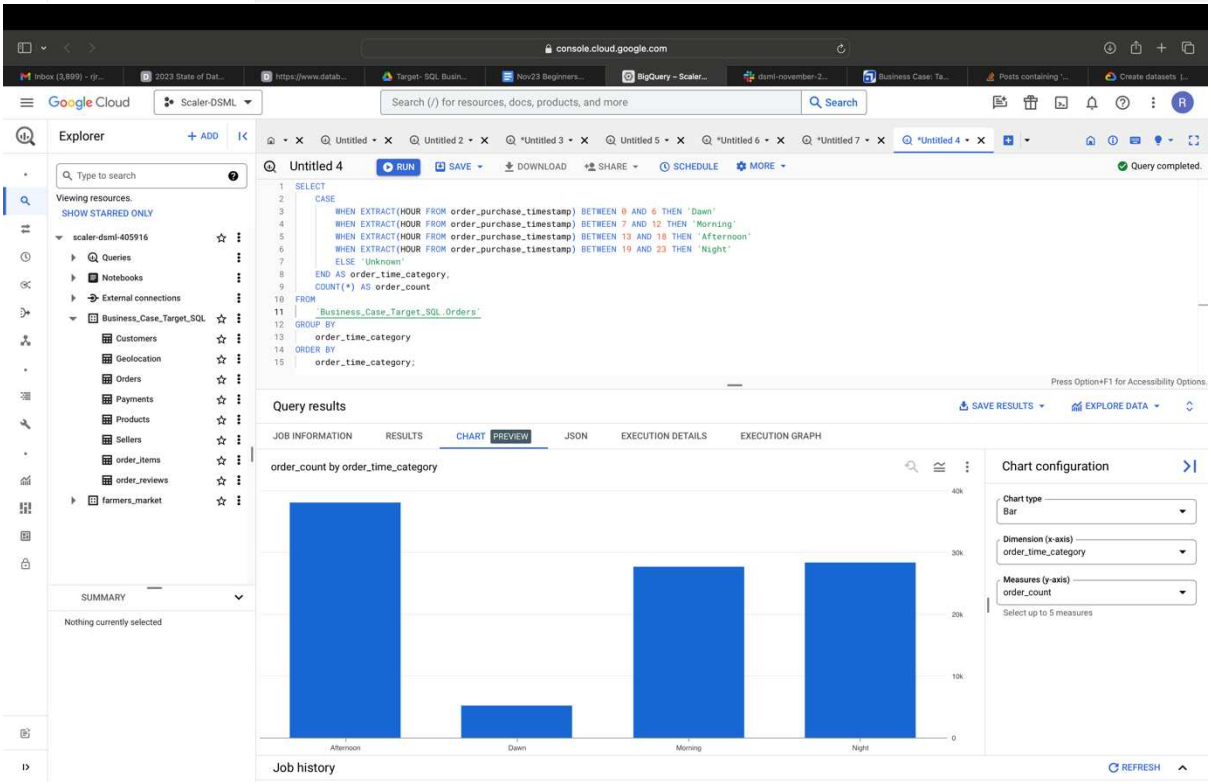
2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

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

Code:

```
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'
    ELSE 'Unknown'
  END AS order_time_category,
  COUNT(*) AS order_count
FROM
  `Business_Case_Target_SQL.Orders`
GROUP BY
  order_time_category
ORDER BY
  order_time_category;
```





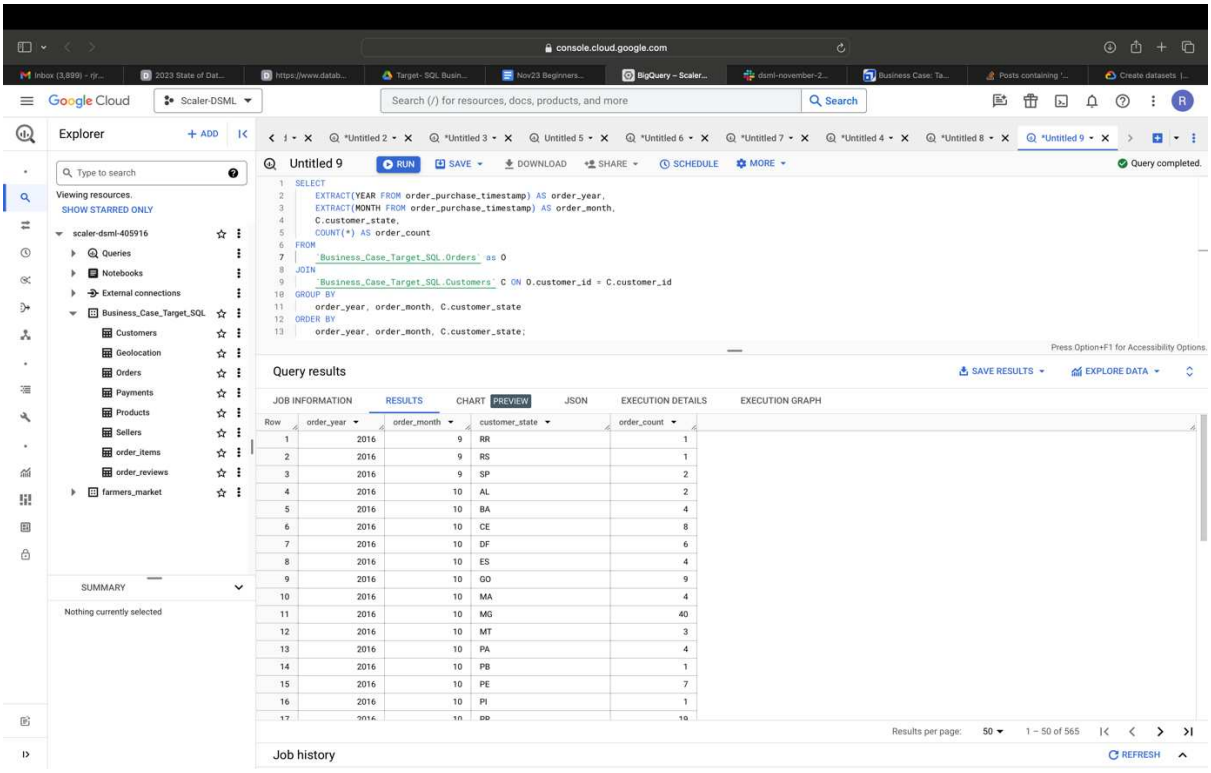
Comment: 95% of the orders were placed during Afternoon and Night and Morning time. Around 38.5% of the orders were placed during Afternno

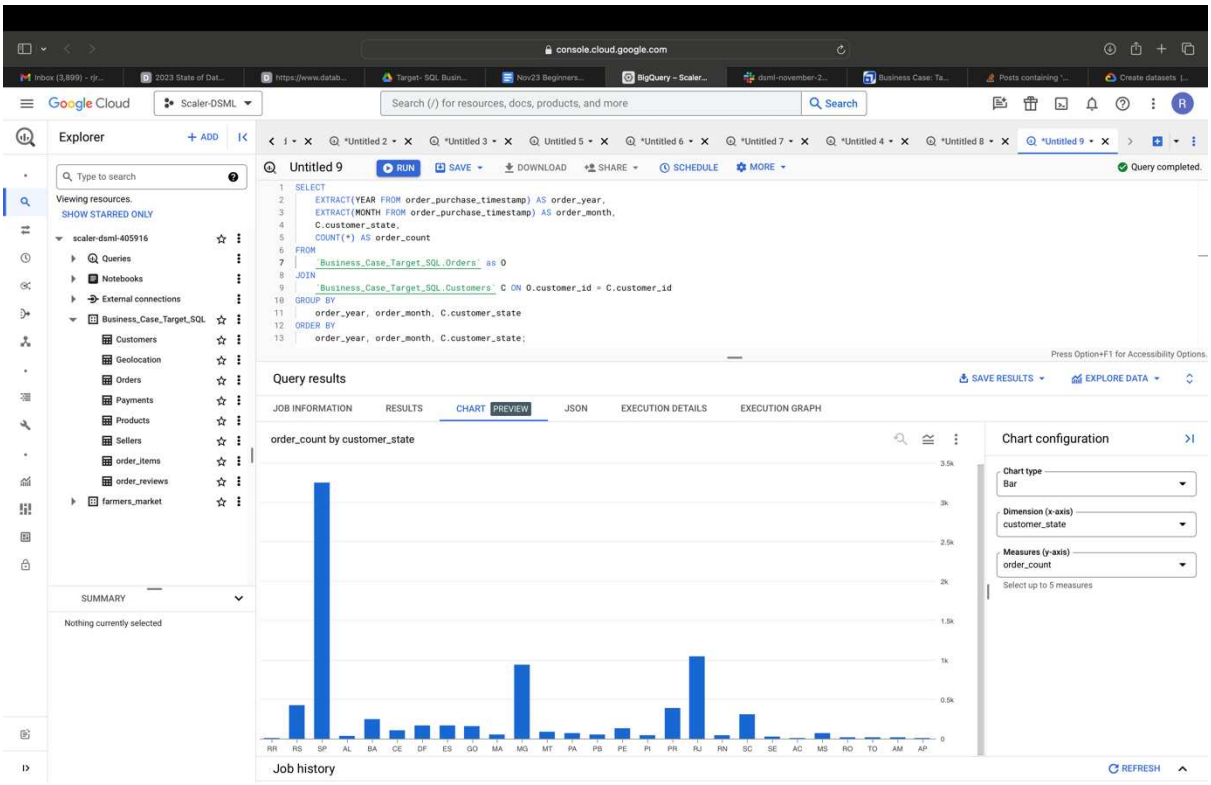
3.Evolution of E-commerce orders in the Brazil region:

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

Code:

```
SELECT
    EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
    EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
    C.customer_state,
    COUNT(*) AS order_count
FROM
    `Business_Case_Target_SQL.Orders` as O
JOIN
    `Business_Case_Target_SQL.Customers` C ON O.customer_id =
C.customer_id
GROUP BY
    order_year, order_month, C.customer_state
ORDER BY
    order_year, order_month, C.customer_state;
```



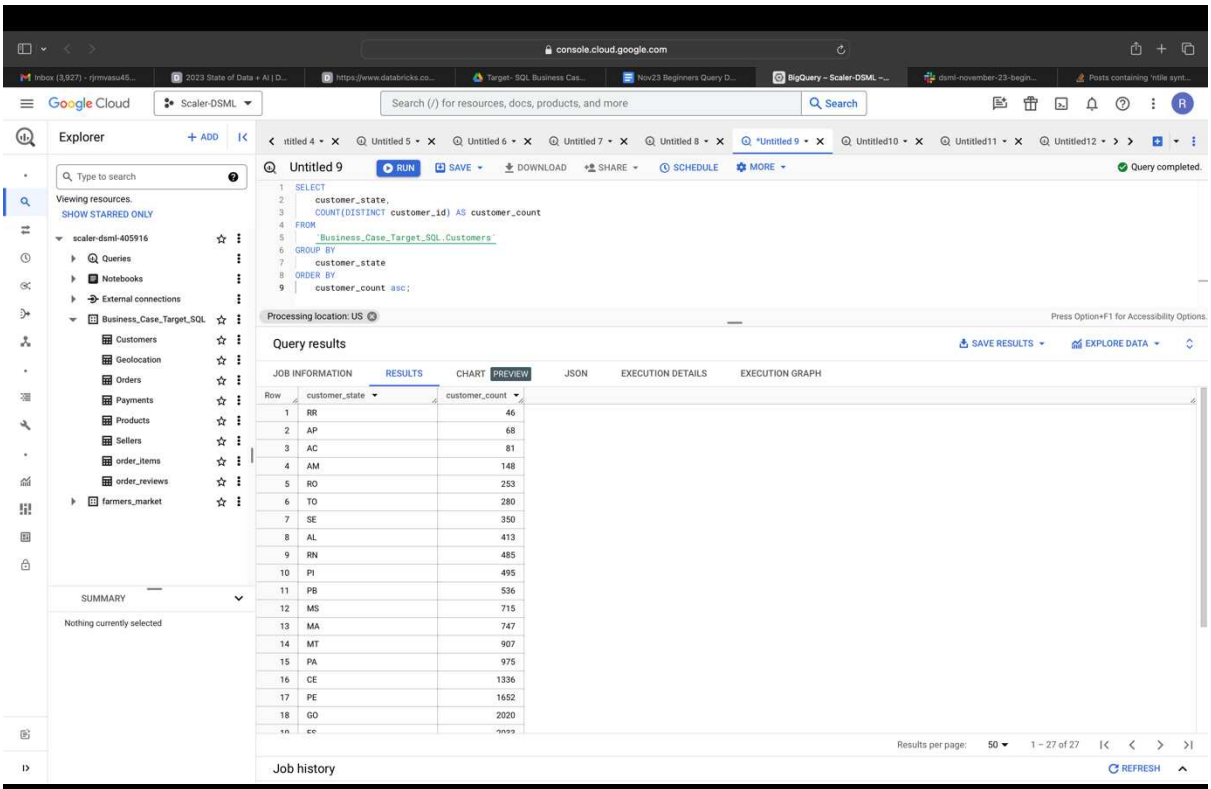


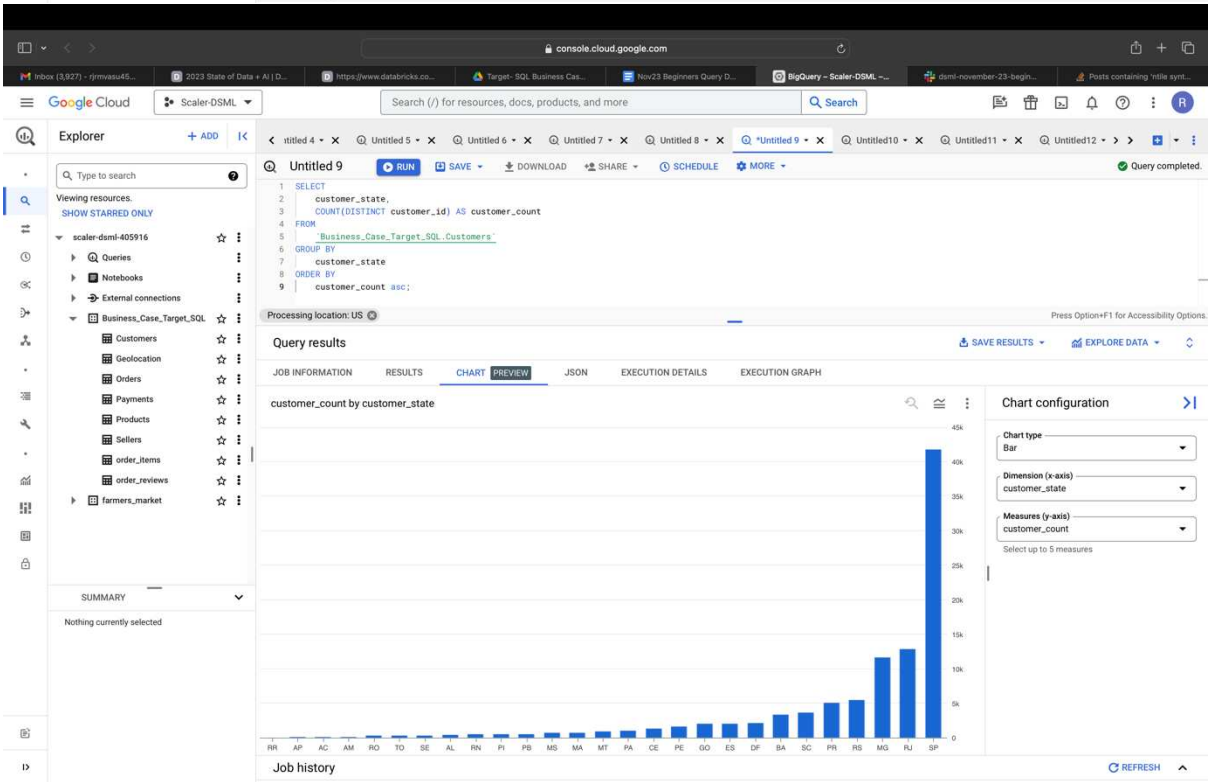
Comment: *Sao Paulo(SP) state being the most populated and having the highest economic activity in Brazil is clearly in the no. 1 spot by having the most no. of orders. The next state Rio De Janiero(RJ) and Minas Gerais(MG) which are 2nd and 3rd spot put together still fall short of total orders placed in SP. The key factor here is the population as RJ & MG have less than 40% & 50% population of SP respectively. Though RJ has less population than MG the orders placed from there are higher.*

3.2 How are the customers distributed across all the states?

Code:

```
SELECT
    customer_state,
    COUNT(DISTINCT customer_id) AS customer_count
FROM
    `Business_Case_Target_SQL.Customers`
GROUP BY
    customer_state
ORDER BY
    customer_state;
```





Comment: *As mentioned in the previous graph, the no. of customers from each state clearly shows the population density and economic activity of each state.*

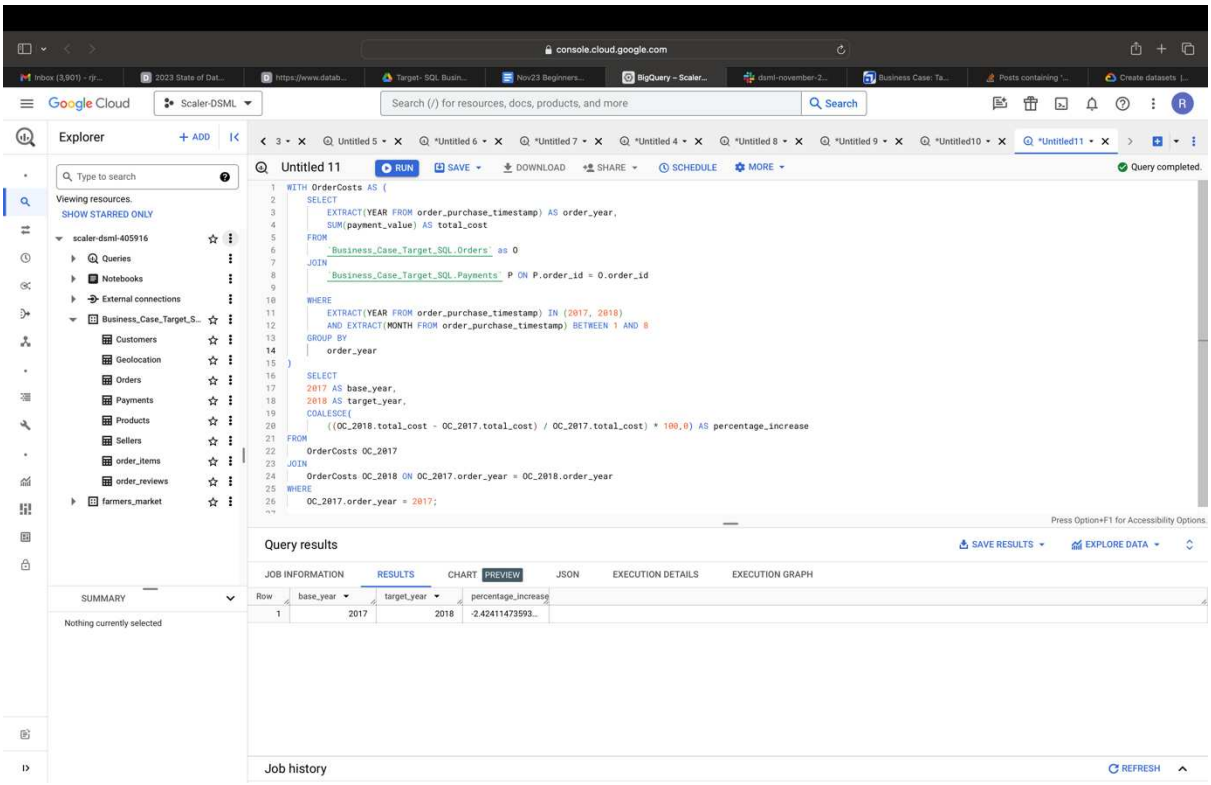
4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

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.

Code:

```
WITH OrderCosts AS (
  SELECT
    EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
    SUM(payment_value) AS total_cost
  FROM
    `Business_Case_Target_SQL.Orders` as O
  JOIN
    `Business_Case_Target_SQL.Payments` P ON P.order_id = O.order_id

  WHERE
    EXTRACT(YEAR FROM order_purchase_timestamp) IN (2017, 2018)
    AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8
  GROUP BY
    order_year
)
  SELECT
  2017 AS base_year,
  2018 AS target_year,
  COALESCE(
    ((OC_2018.total_cost - OC_2017.total_cost) / OC_2017.total_cost) * 100,0) AS percentage_increase
FROM
  OrderCosts OC_2017
JOIN
  OrderCosts OC_2018 ON OC_2017.order_year = OC_2018.order_year
WHERE
  OC_2017.order_year = 2017;
```

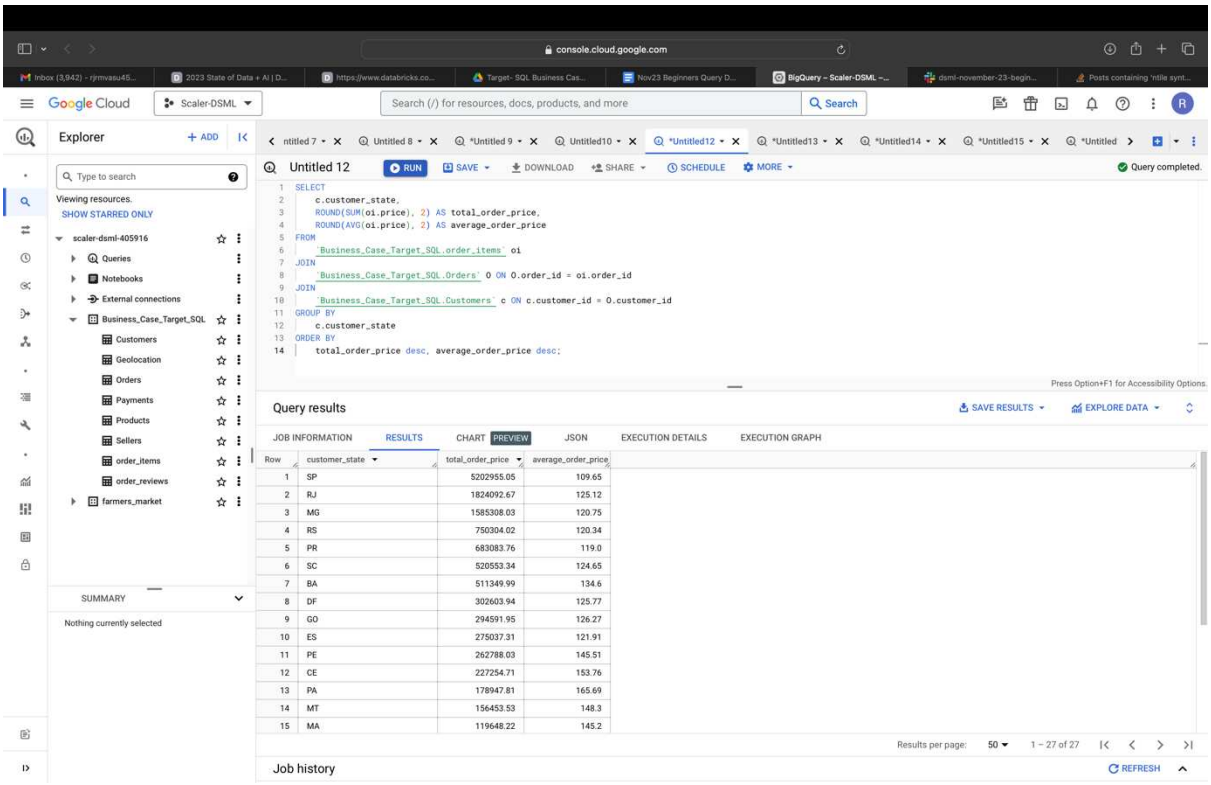


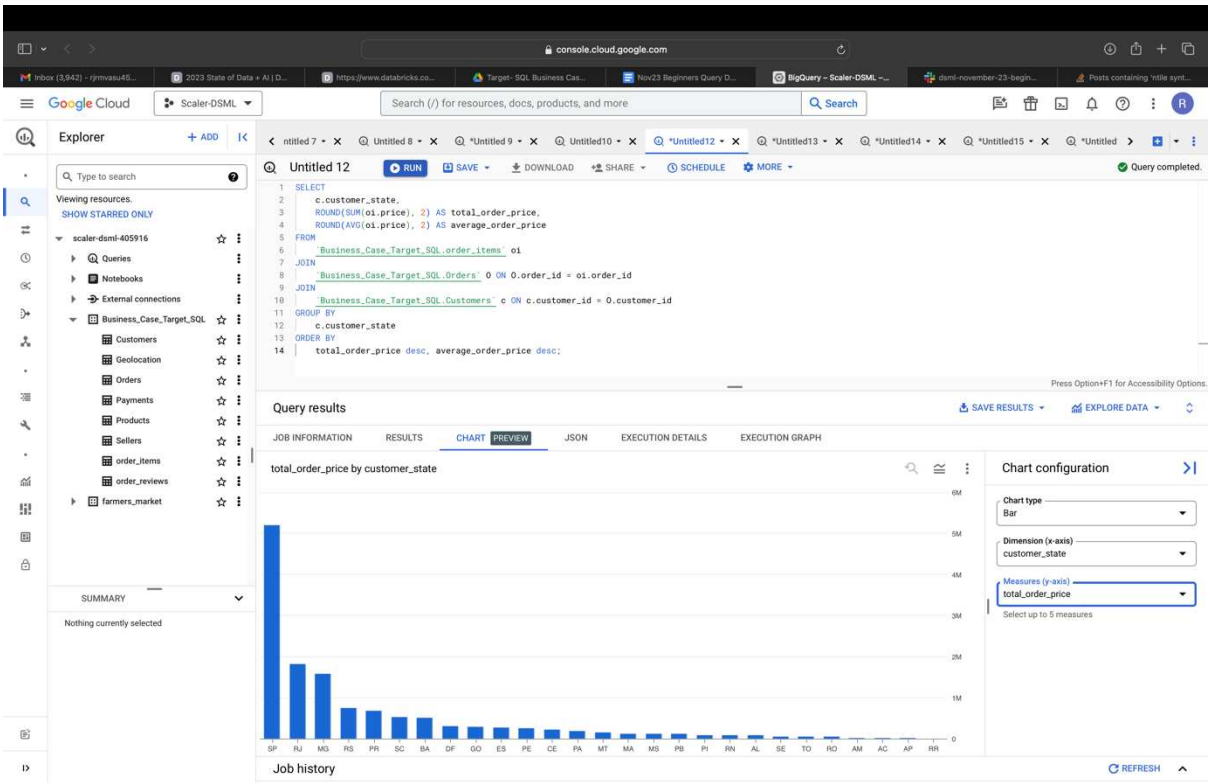
Comment: *As per the data there is a slight decrease in the cost of orders in 2018 when compared to 2017.*

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

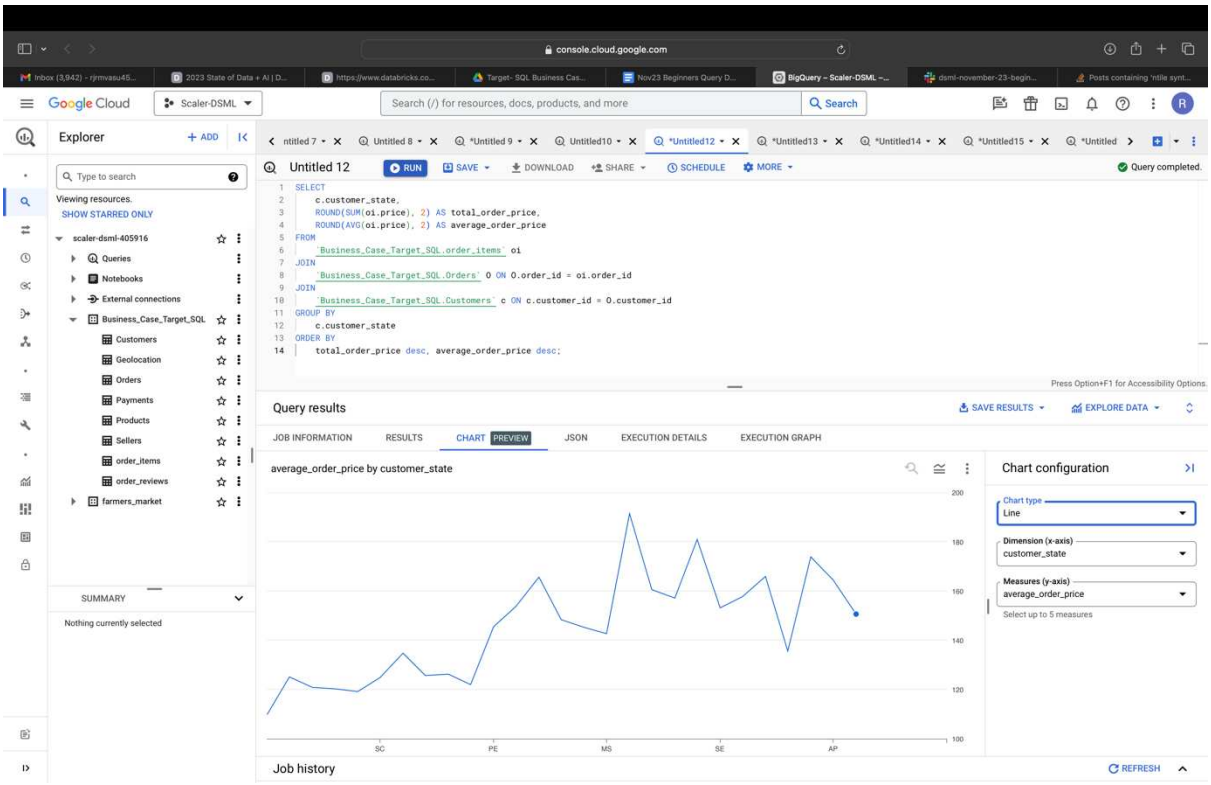
Code:

```
SELECT
  c.customer_state,
  ROUND(SUM(oi.price), 2) AS total_order_price,
  ROUND(AVG(oi.price), 2) AS average_order_price
FROM
  `Business_Case_Target_SQL.order_items` oi
JOIN
  `Business_Case_Target_SQL.Orders` O ON O.order_id = oi.order_id
JOIN
  `Business_Case_Target_SQL.Customers` c ON c.customer_id = O.customer_id
GROUP BY
  c.customer_state
ORDER BY
  c.customer_state;
```





Comment: *The top 5 states were most orders placed contribute to the total value of the orders.*

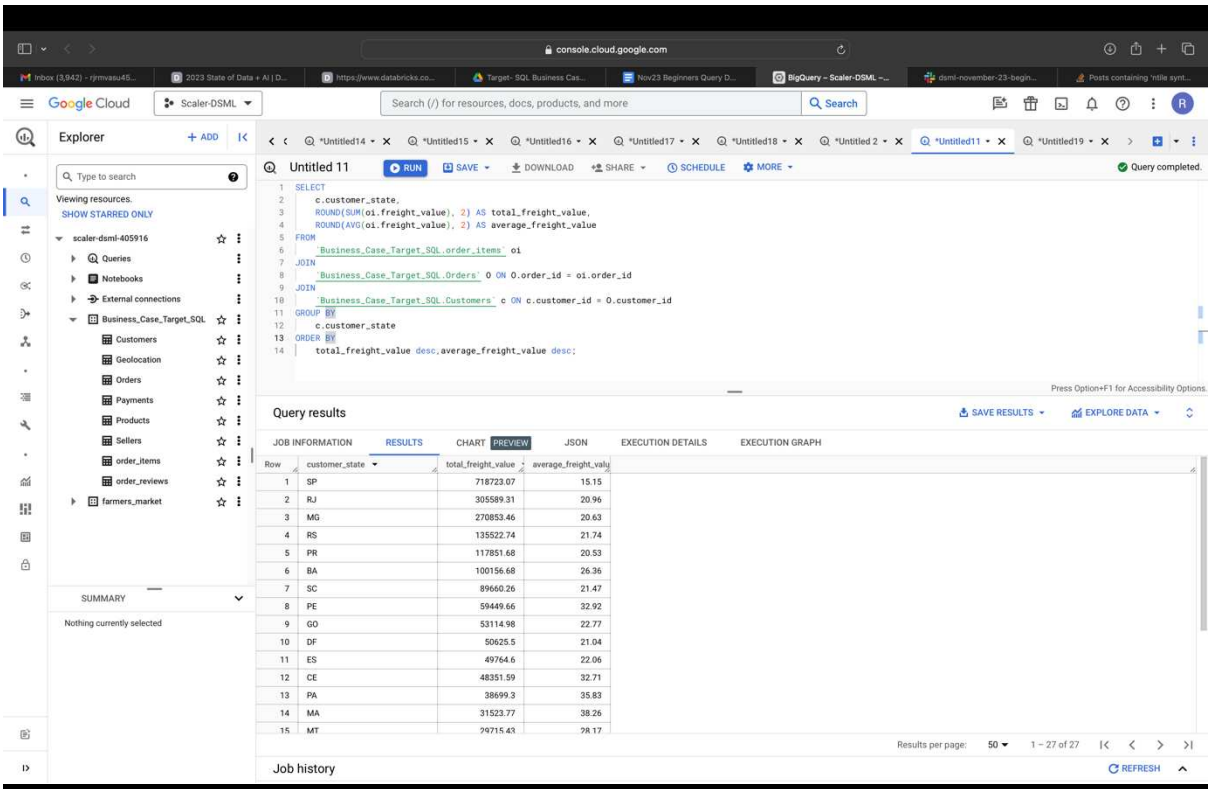


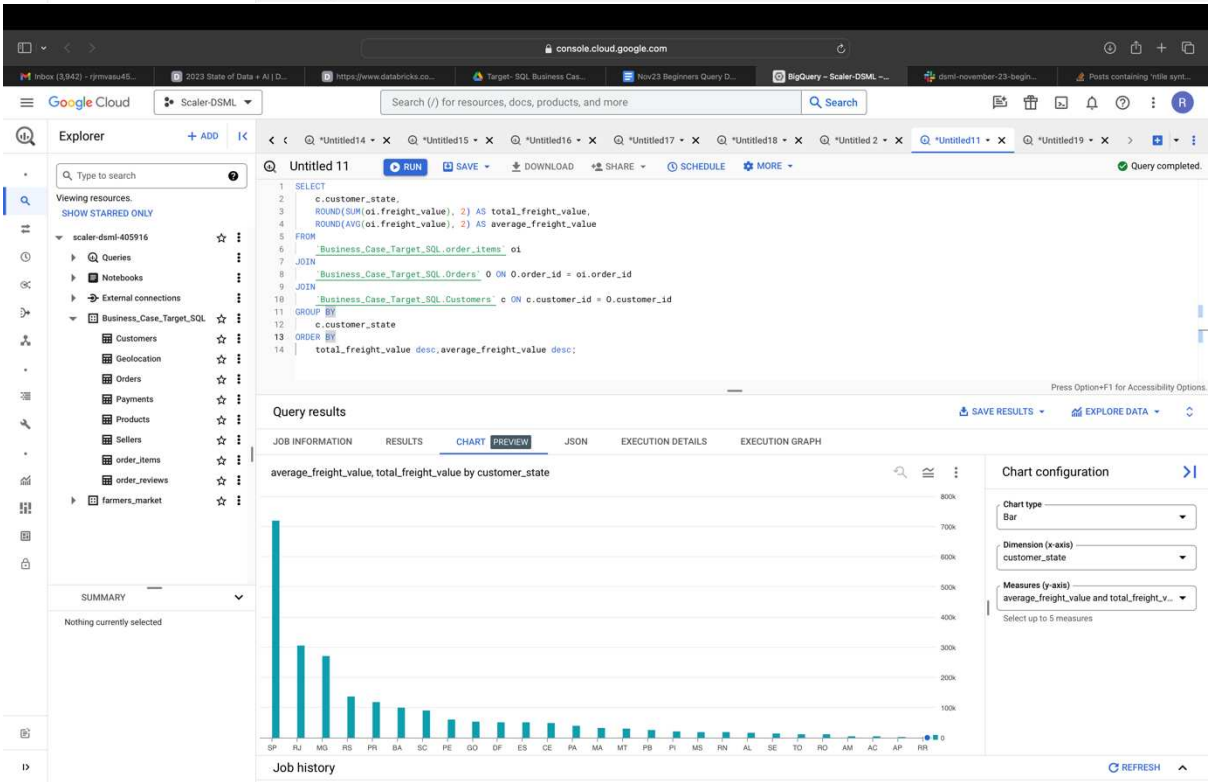
Comment: *Though the top 5 states were most orders placed have the lesser average order price but it is not consistent with where lower the orders, higher the average is the story for the rest of the states. Need deeper analysis to find the other contributing factors.*

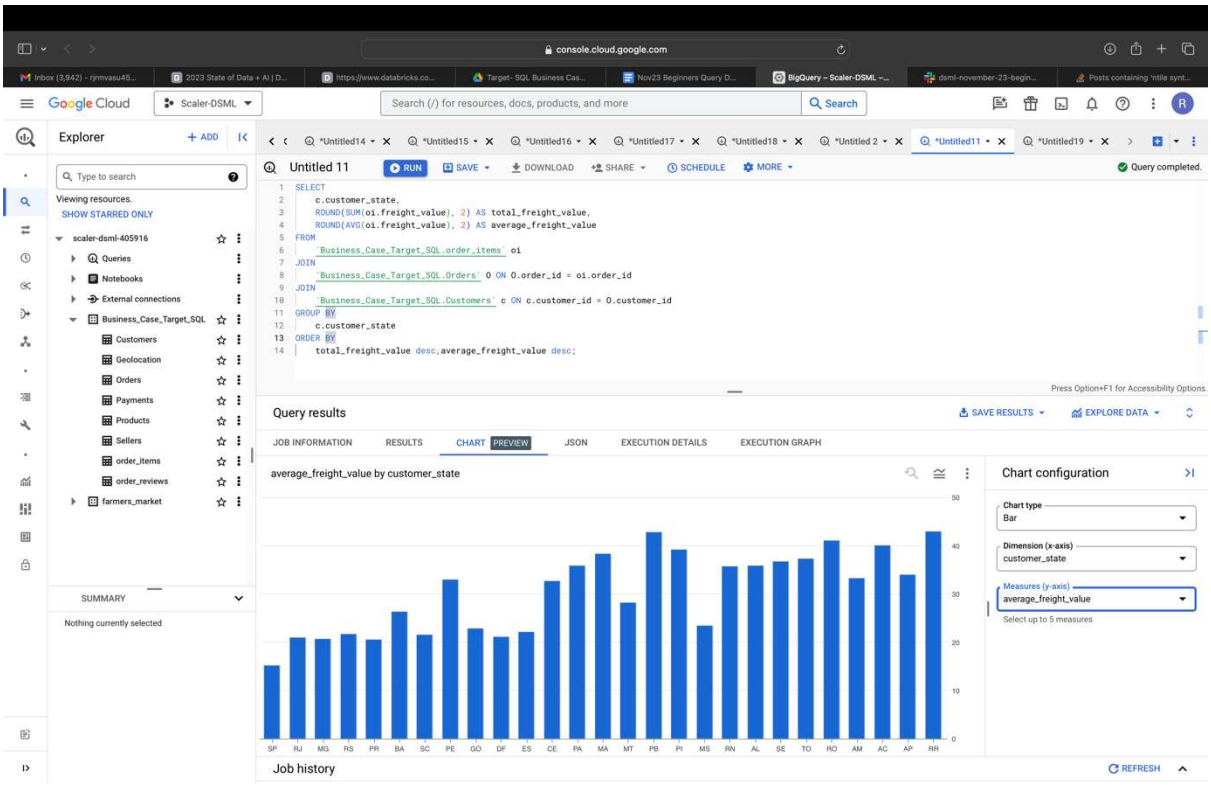
4.3 Calculate the Total & Average value of order freight for each state.

Code:

```
SELECT
  c.customer_state,
  ROUND(SUM(oi.price), 2) AS total_freight_value,
  ROUND(AVG(oi.price), 2) AS average_freight_value
FROM
  `Business_Case_Target_SQL.order_items` oi
JOIN
  `Business_Case_Target_SQL.Orders` O ON O.order_id = oi.order_id
JOIN
  `Business_Case_Target_SQL.Customers` c ON c.customer_id = O.customer_id
GROUP BY
  c.customer_state
ORDER BY
  c.customer_state;
```







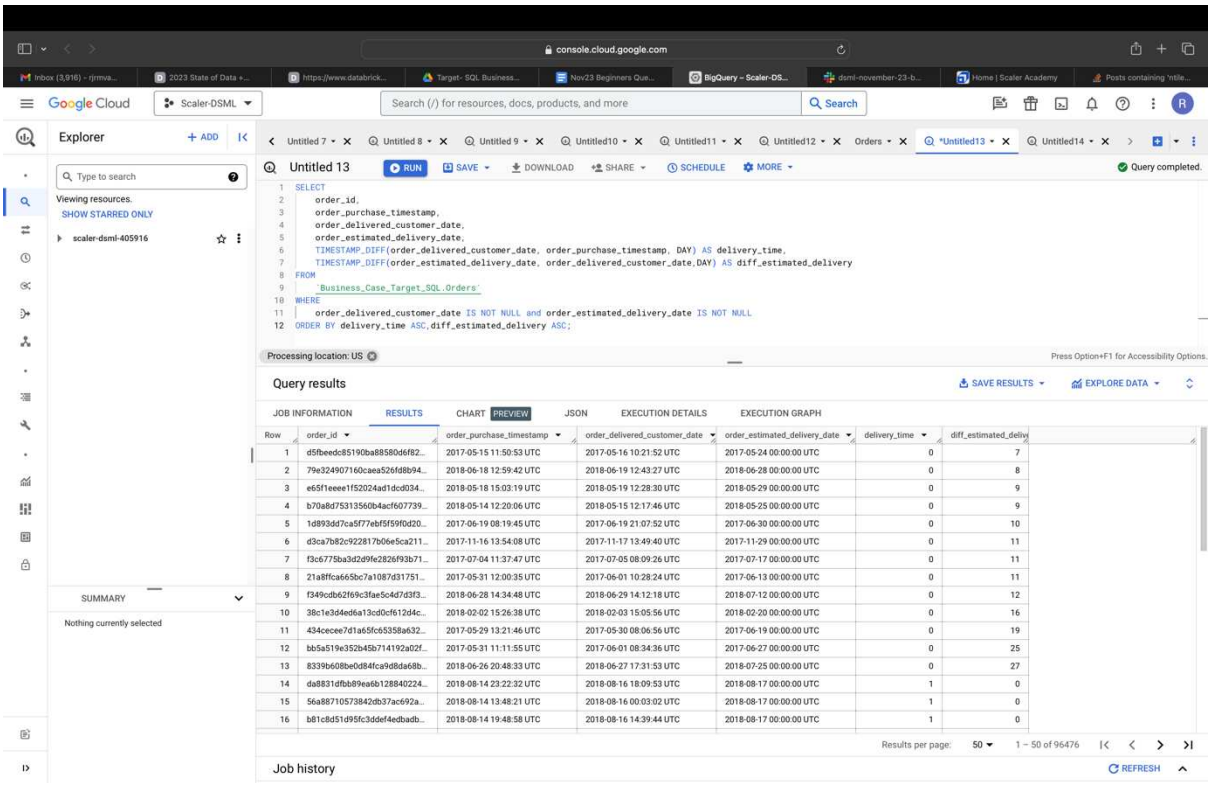
Comment: *The top 5 states were most orders placed contribute to the majority of the freight value. Also due to the no. of orders being high for top states the average freight value is lower however it is not consistent for the rest of the states which requires more deeper analysis.*

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

5.1 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



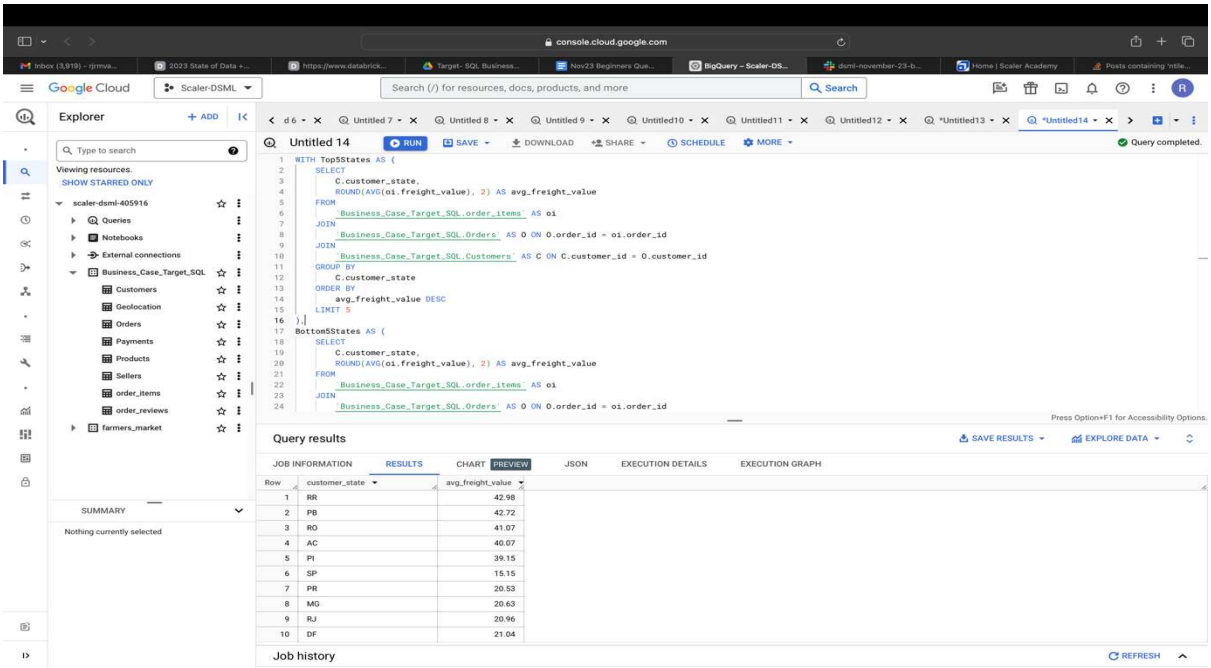
5.2 Find out the top 5 states with the highest & lowest average freight value.

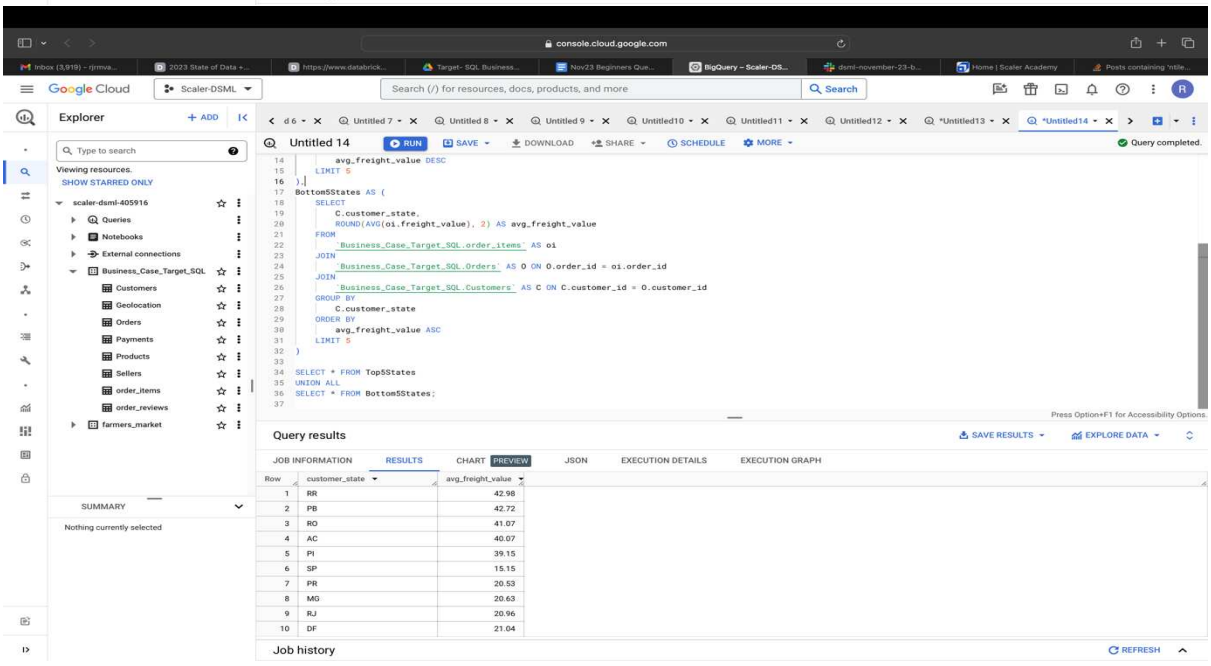
Code:

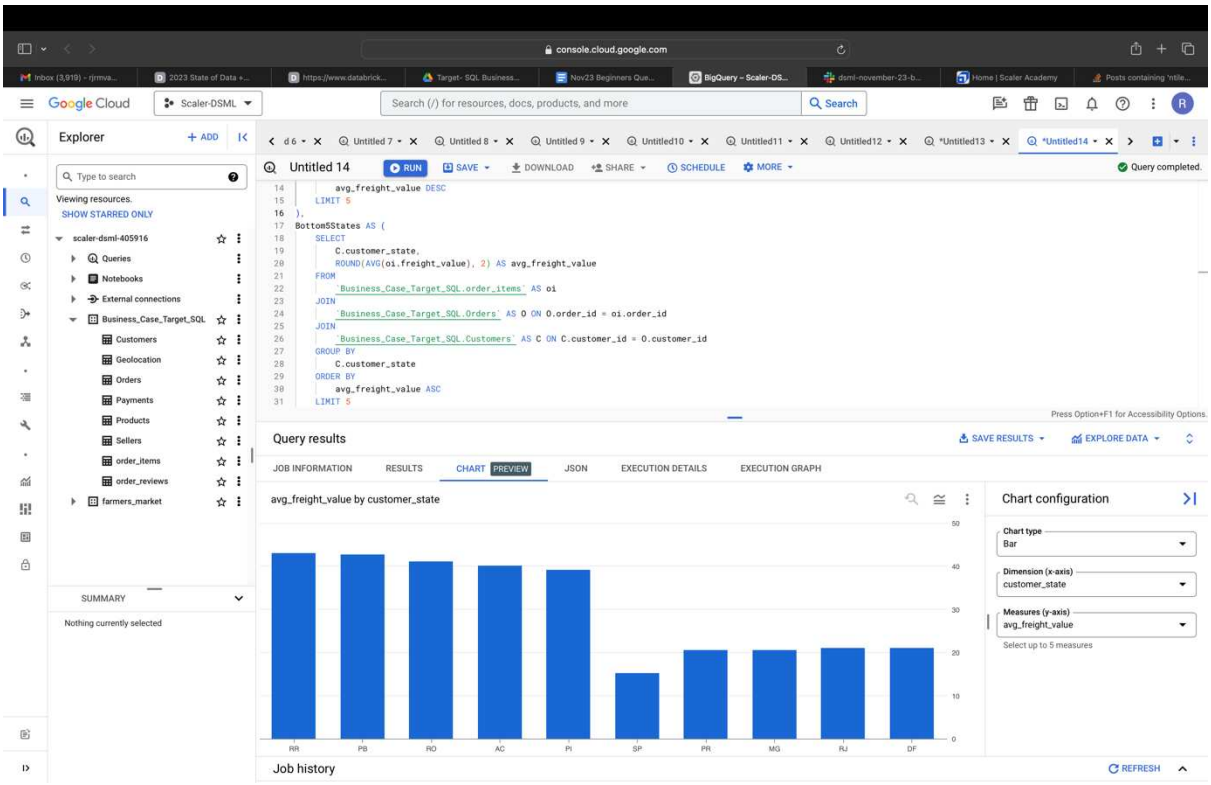
```
WITH Top5States AS (
  SELECT
    C.customer_state,
    ROUND(AVG(oi.freight_value), 2) AS avg_freight_value
  FROM
    `Business_Case_Target_SQL.order_items` AS oi
  JOIN
    `Business_Case_Target_SQL.Orders` AS O ON O.order_id = oi.order_id
  JOIN
    `Business_Case_Target_SQL.Customers` AS C ON C.customer_id = O.customer_id
  GROUP BY
    C.customer_state
  ORDER BY
    avg_freight_value DESC
  LIMIT 5
),
Bottom5States AS (
  SELECT
    C.customer_state,
    ROUND(AVG(oi.freight_value), 2) AS avg_freight_value
  FROM
    `Business_Case_Target_SQL.order_items` AS oi
  JOIN
    `Business_Case_Target_SQL.Orders` AS O ON O.order_id = oi.order_id
  JOIN
```

```
    `Business_Case_Target_SQL.Customers` AS C ON C.customer_id = O.customer_id
  GROUP BY
    C.customer_state
  ORDER BY
    avg_freight_value ASC
  LIMIT 5
)

SELECT * FROM Top5States
UNION ALL
SELECT * FROM Bottom5States;
```







Comment: *The lesser the orders more the average of the freight value which can be seen with states like AC, RR PB,RO & AC, the state which placed most orders having the lowest average of the freight value like SP,MG,RJ, etc.*

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

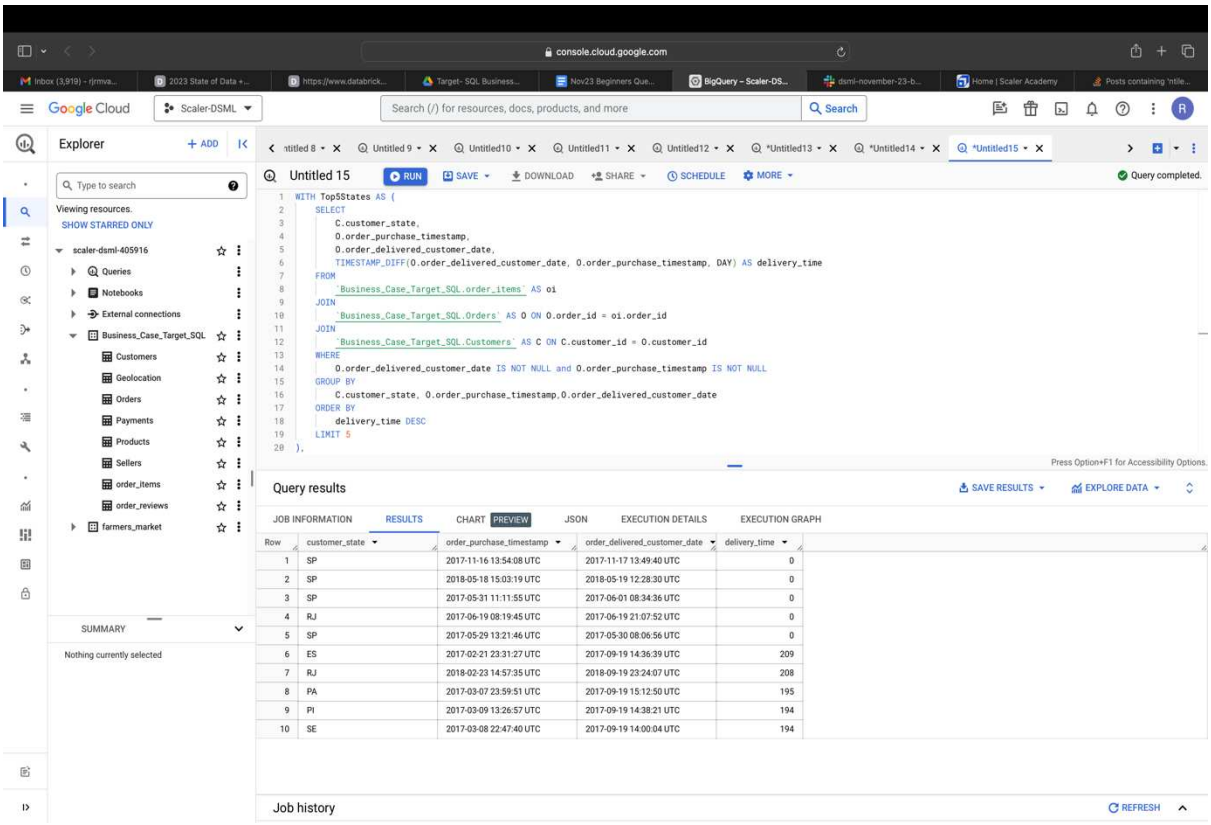
Code:

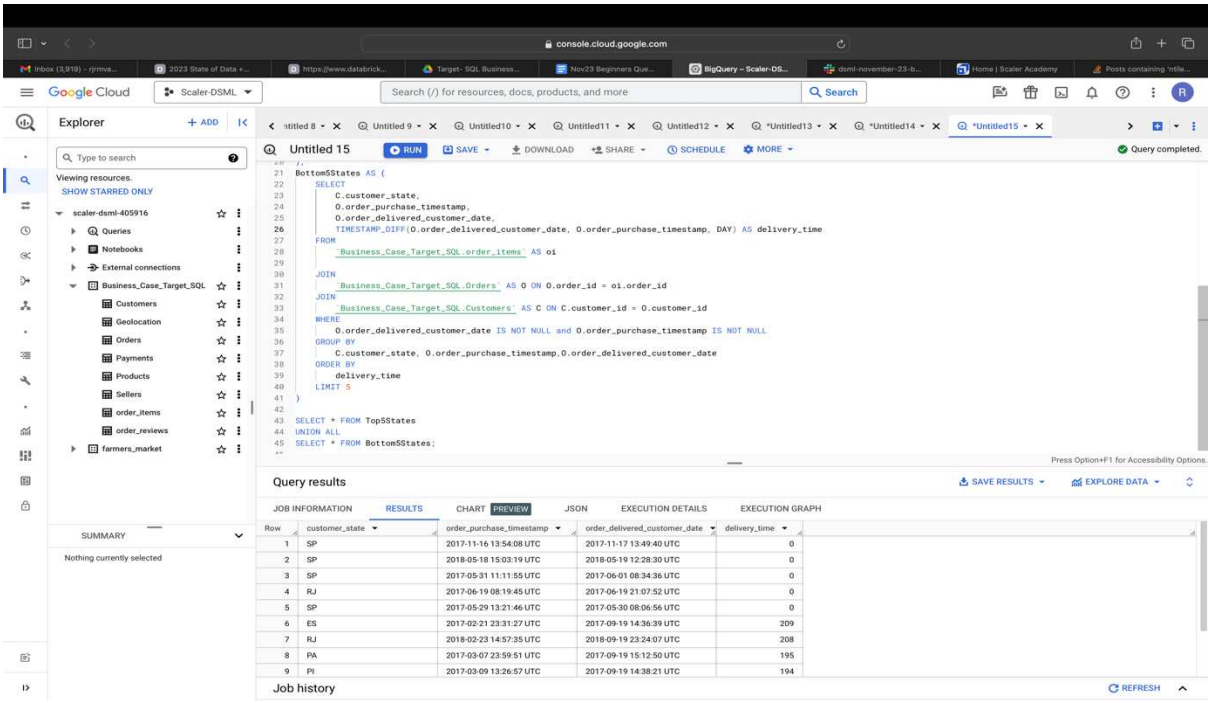
```
WITH Top5States AS (
    SELECT
        C.customer_state,
        O.order_purchase_timestamp,
        O.order_delivered_customer_date,
        TIMESTAMP_DIFF(O.order_delivered_customer_date,
O.order_purchase_timestamp, DAY) AS delivery_time
    FROM
        `Business_Case_Target_SQL.order_items` AS oi
    JOIN
        `Business_Case_Target_SQL.Orders` AS O ON O.order_id =
oi.order_id
    JOIN
        `Business_Case_Target_SQL.Customers` AS C ON C.customer_id =
O.customer_id
    WHERE
        O.order_delivered_customer_date IS NOT NULL and
O.order_purchase_timestamp IS NOT NULL
    GROUP BY
```

```
        C.customer_state,
O.order_purchase_timestamp,O.order_delivered_customer_date
    ORDER BY
        delivery_time DESC
    LIMIT 5
),
Bottom5States AS (
    SELECT
        C.customer_state,
        O.order_purchase_timestamp,
        O.order_delivered_customer_date,
        TIMESTAMP_DIFF(O.order_delivered_customer_date,
O.order_purchase_timestamp, DAY) AS delivery_time
    FROM
        `Business_Case_Target_SQL.order_items` AS oi

    JOIN
        `Business_Case_Target_SQL.Orders` AS O ON O.order_id =
oi.order_id
    JOIN
        `Business_Case_Target_SQL.Customers` AS C ON C.customer_id =
O.customer_id
    WHERE
        O.order_delivered_customer_date IS NOT NULL and
O.order_purchase_timestamp IS NOT NULL
    GROUP BY
        C.customer_state,
O.order_purchase_timestamp,O.order_delivered_customer_date
    ORDER BY
        delivery_time
    LIMIT 5
)

SELECT * FROM Top5States
UNION ALL
SELECT * FROM Bottom5States;
```



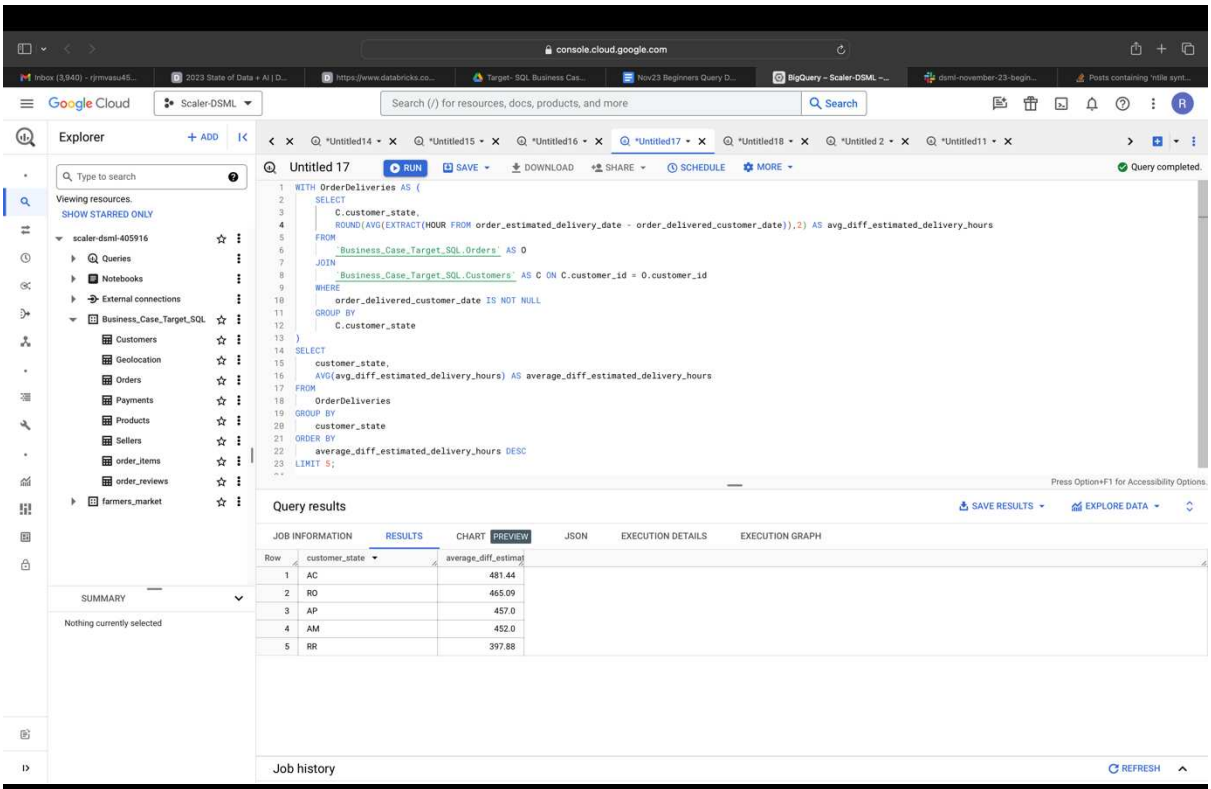


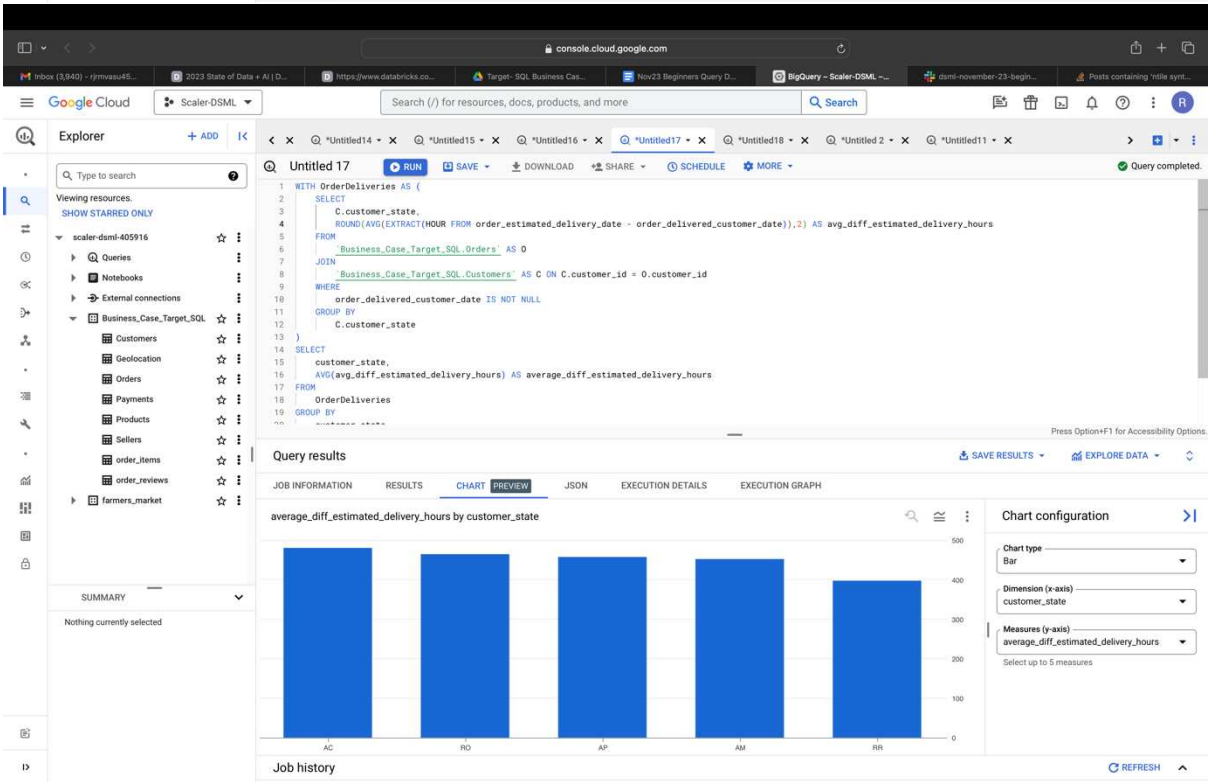
Comment: *Again SP and RJ with the better infrastructure and connectivity among other states has the lowest(fastest) delivery time.*

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

Code:

```
WITH OrderDeliveries AS (
  SELECT
    C.customer_state,
    AVG(EXTRACT(HOUR FROM order_delivered_customer_date - order_purchase_timestamp)) AS
avg_delivery_time_hours,
    AVG(EXTRACT(HOUR FROM order_estimated_delivery_date - order_delivered_customer_date)) AS
avg_diff_estimated_delivery_hours
  FROM
    `Business_Case_Target_SQL.Orders` AS O
  JOIN
    `Business_Case_Target_SQL.Customers` AS C ON C.customer_id = O.customer_id
  WHERE
    order_delivered_customer_date IS NOT NULL
  GROUP BY
    C.customer_state
)
SELECT
  customer_state,
  AVG(avg_delivery_time_hours) AS average_delivery_time_hours,
  AVG(avg_diff_estimated_delivery_hours) AS average_diff_estimated_delivery_hours
FROM
  OrderDeliveries
GROUP BY
  customer_state
ORDER BY
  avg_diff_estimated_delivery_hours DESC
LIMIT 5;
```





Comment: *Top 5 states were the actual delivery time is much ahead of the estimated delivery date/time.*

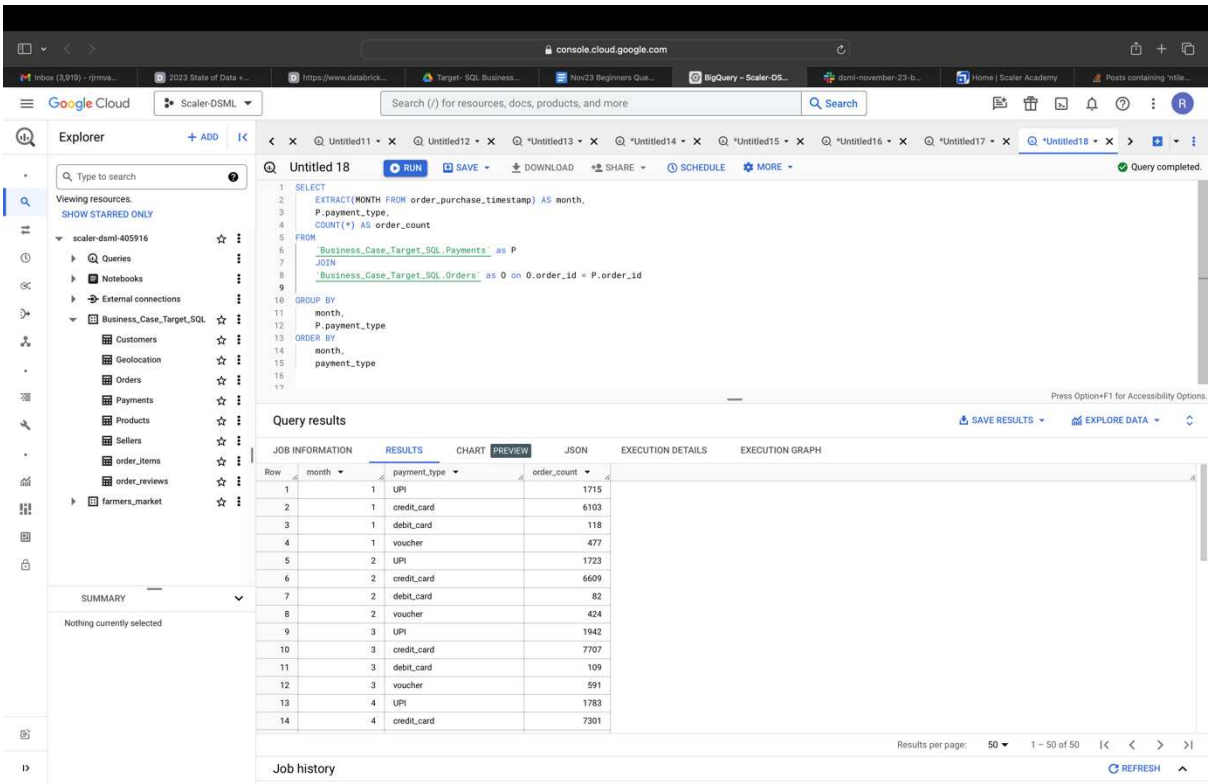
6. Analysis based on the payments:

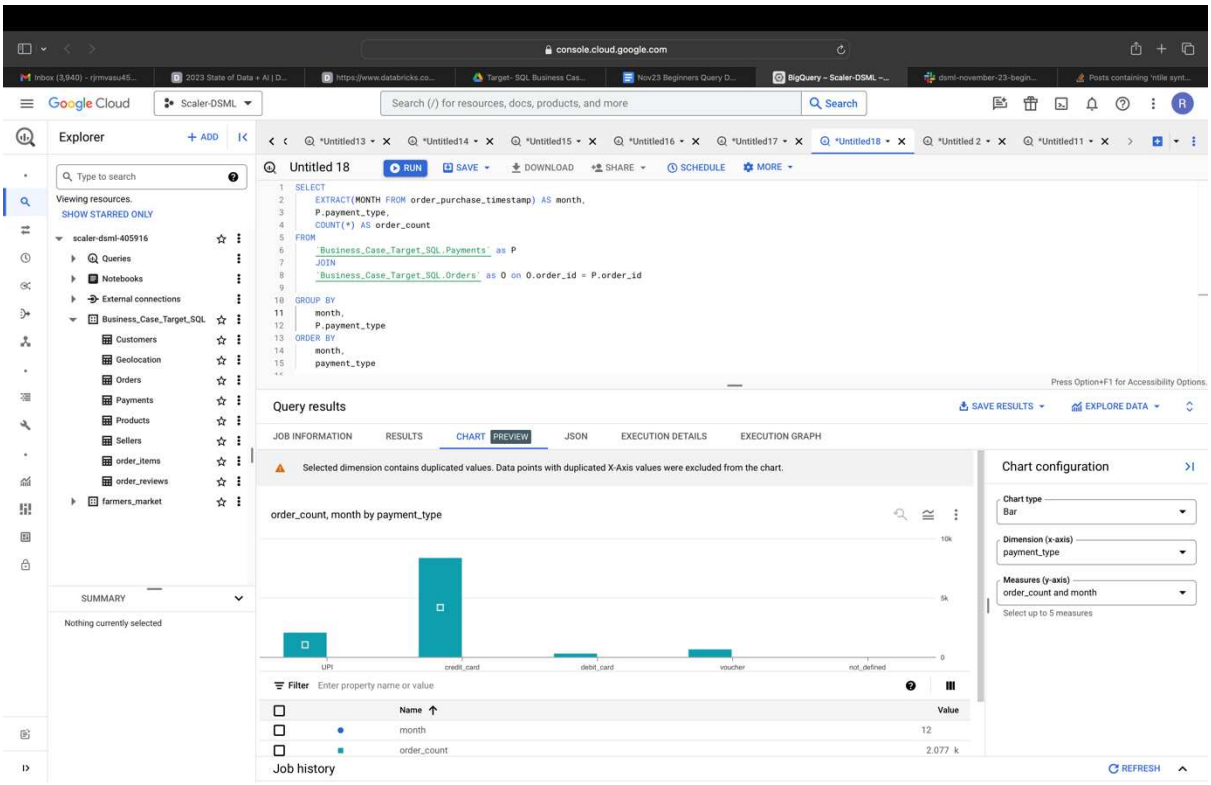
6.1 Find the month on month no. of orders placed using different payment types.

Code:

```
SELECT
  EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
  P.payment_type,
  COUNT(*) AS order_count
FROM
  `Business_Case_Target_SQL.Payments` as P
  JOIN
  `Business_Case_Target_SQL.Orders` as O on O.order_id = P.order_id

GROUP BY
  month,
  P.payment_type
ORDER BY
  month,
  payment_type
```



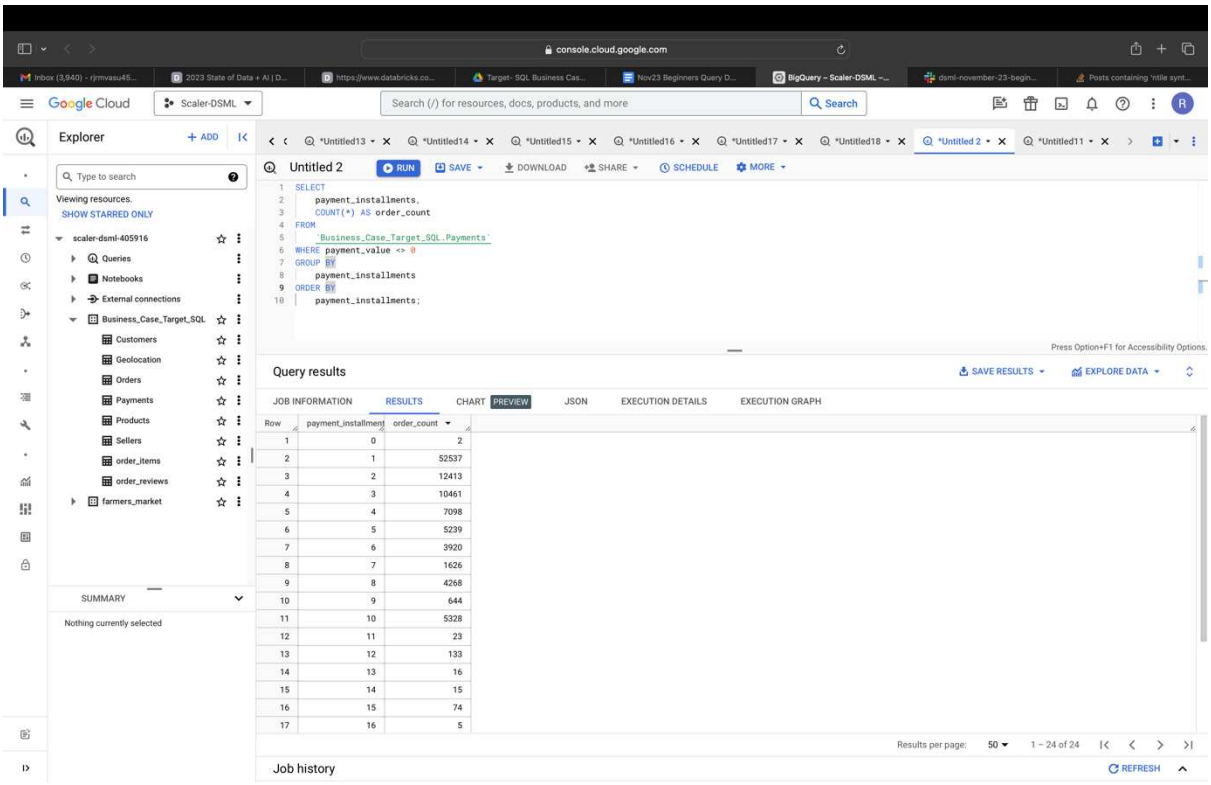


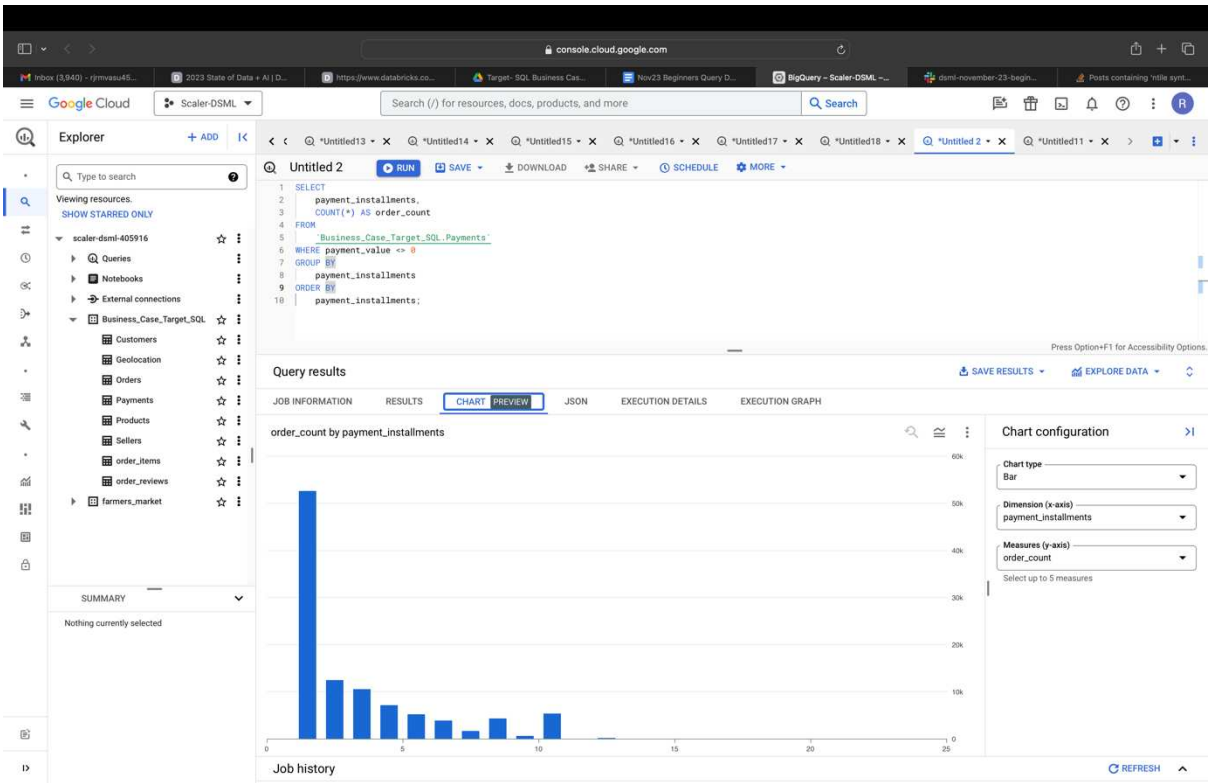
COMMENT: *Overall Credit Card payment type is the most used payment method by customers to place orders.*

6.2 Find the no. of orders placed on the basis of the payment instalments that have been paid.

Code:

```
SELECT
  payment_installments,
  COUNT(*) AS order_count
FROM
  `Business_Case_Target_SQL.Payments`
GROUP BY
  payment_installments
ORDER BY
    payment_installments;
```





Comment: *More than 50% of the orders the payments were made in a single payment.*