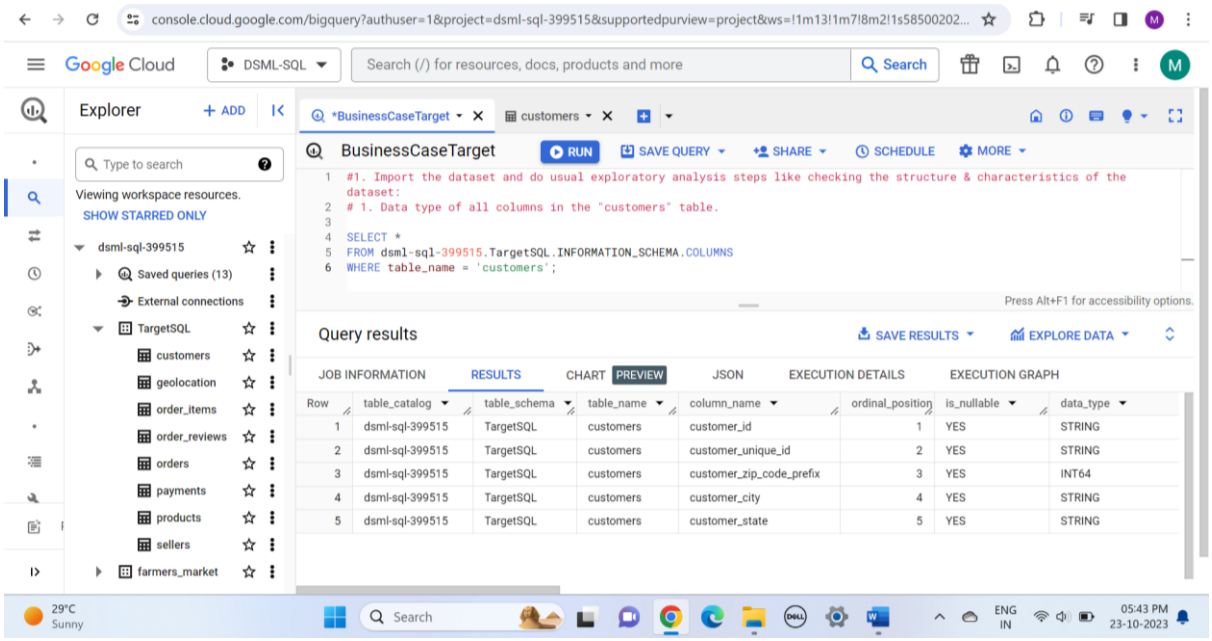
Business Cases: Target SQL

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.

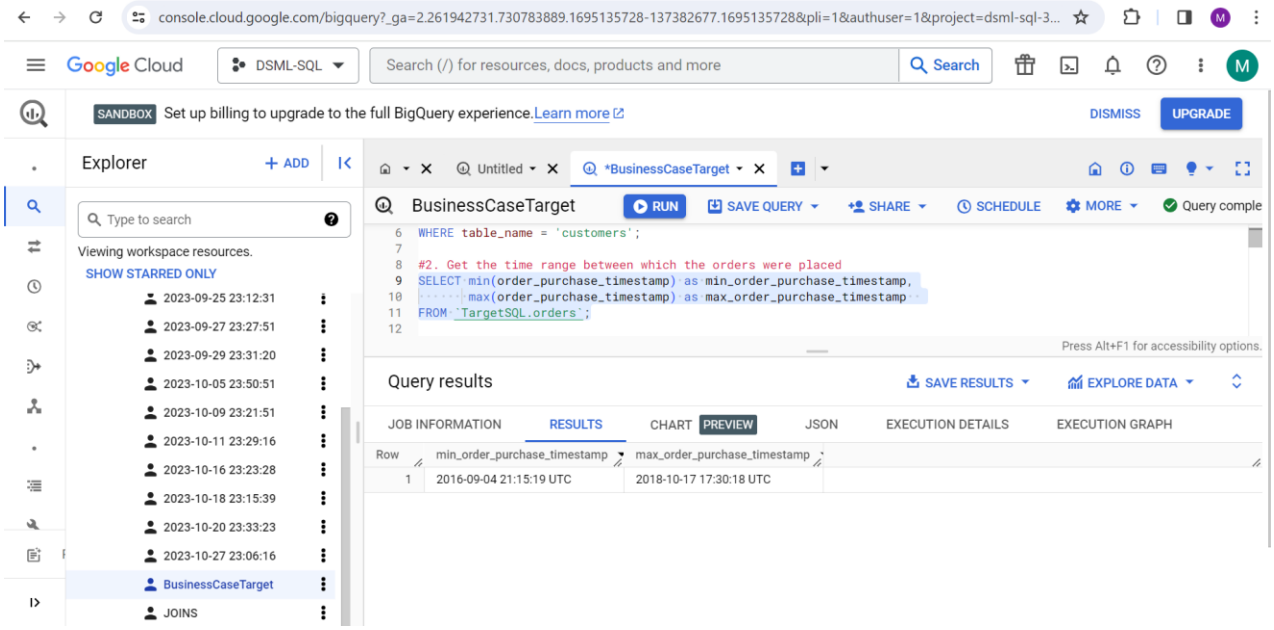
```
SELECT *
FROM dsml-sql-399515.TargetSQL.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers';
```



Insights: From the above query we got to know the Description of the customer table and also data type of the columns

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

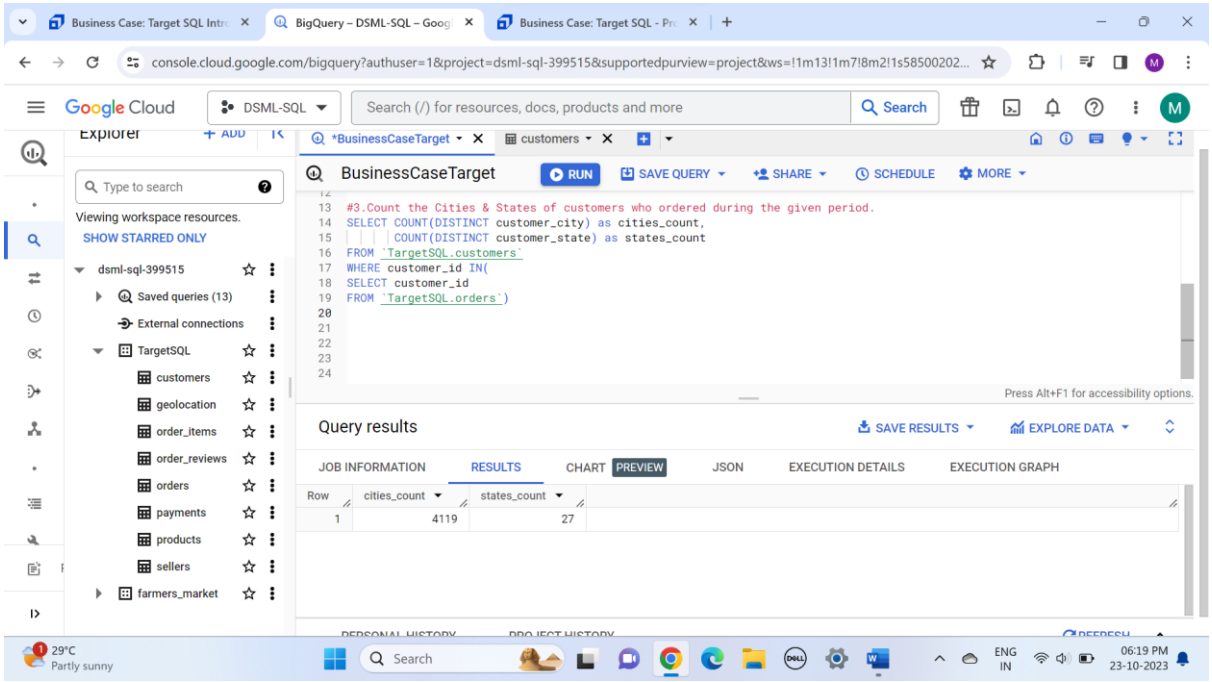
```
SELECT min(order_purchase_timestamp) as min_order_purchase_timestamp,
    max(order_purchase_timestamp) as max_order_purchase_timestamp
FROM `TargetSQL.orders`
```



Insights: From the above query we got to know that the orders were placed between the year 2016 and 2018.

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

```
SELECT COUNT(DISTINCT customer_city) as cities,
    COUNT(DISTINCT customer_state) as staes
FROM `TargetSQL.customers`
WHERE customer_id IN(
SELECT customer_id
FROM `TargetSQL.orders`)
```

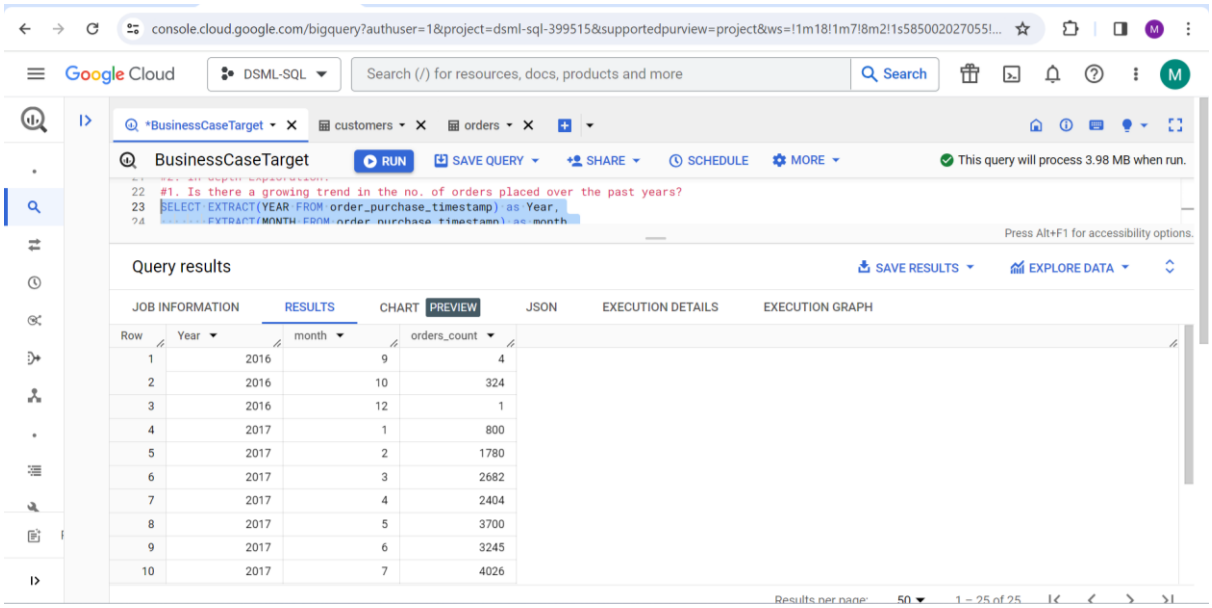


Insights: From the above Query we got to know that there are orders from 27 states and 4119 cities

2. In-depth Exploration:

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,
     EXTRACT(MONTH FROM order_purchase_timestamp) as month,
      count(order_id) as orders_count
FROM `TargetSQL.orders`
GROUP BY 1,2
ORDER BY 1,2;
```

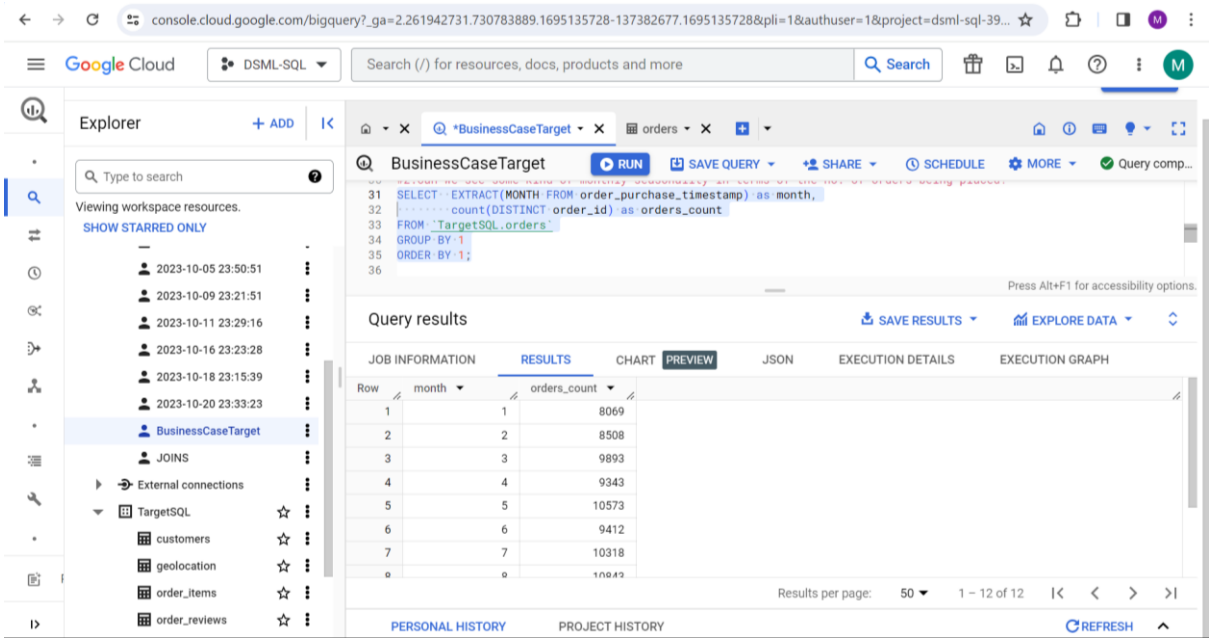


Insights: From the above query we got to know that the no. of orders placed are increasing gradually

Recommendations: It is recommended to maintain adequate storage as there is an increase in the no. of orders placed

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,
      count(DISTINCT order_id) as orders_count
FROM `TargetSQL.orders`
GROUP BY 1
ORDER BY 1;
```

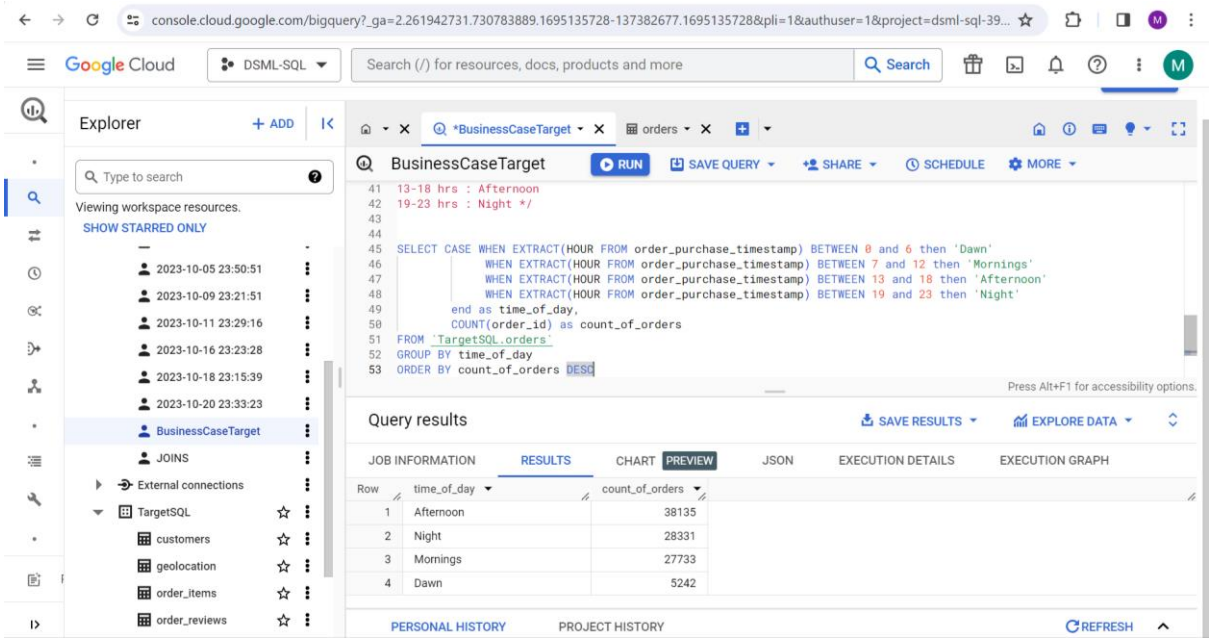


Insights: From the above query we can observe some seasonality in the orders placed. The count of orders generally increases from March to August with fluctuations in between. Additionally, the month of August shows a peak in order count.

Recommendations: It is recommended to maintain adequate quantity of stock in the peak orders months.

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 'Mornings'
       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(order_id) as count_of_orders
FROM `TargetSQL.orders`
GROUP BY time_of_day
ORDER BY count_of_orders DESC
```



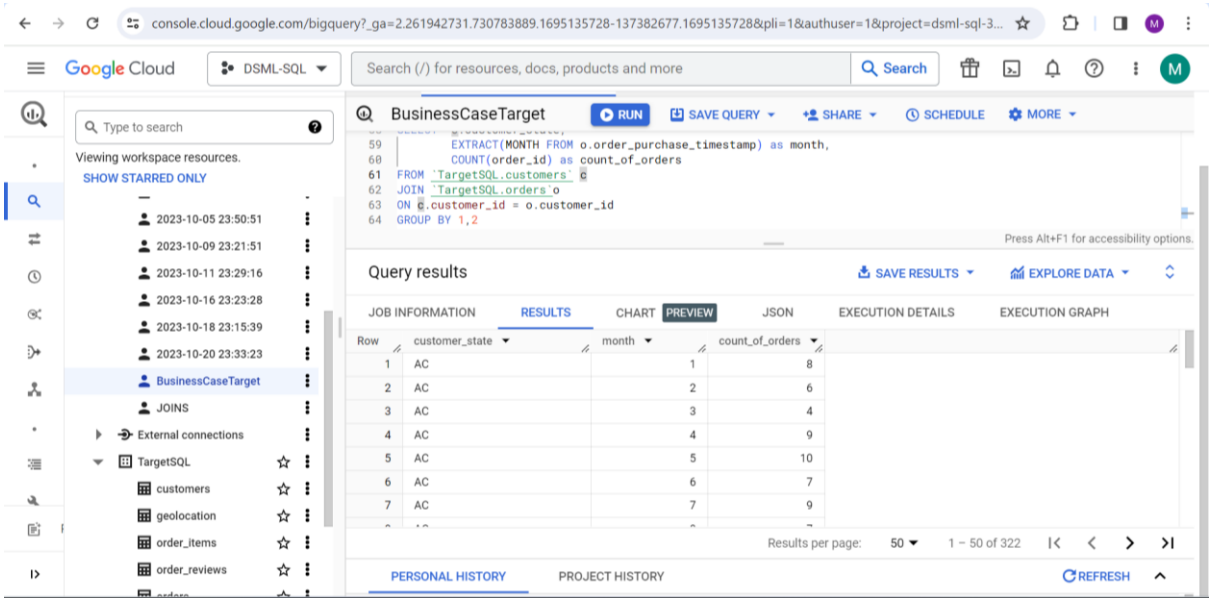
Insights: From the above results we got to know that Brazilian customers tend to place most orders during afternoon and night. This indicates that customers prefer to shop online when they have leisure time or after completing their daily activities.

Recommendations: It is recommended to maintain adequate quantity of stock by identifying peak buying times, companies can allocate resources, such as customer service representatives and inventory, more effectively to meet customer demands and provide a seamless shopping experience

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

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

```
SELECT  c.customer_state,
      EXTRACT(MONTH FROM o.order_purchase_timestamp) as month,
      COUNT(order_id) as count_of_orders
FROM `TargetSQL.customers` c
JOIN `TargetSQL.orders`o
ON c.customer_id = o.customer_id
GROUP BY 1,2
ORDER BY 1,2;
```

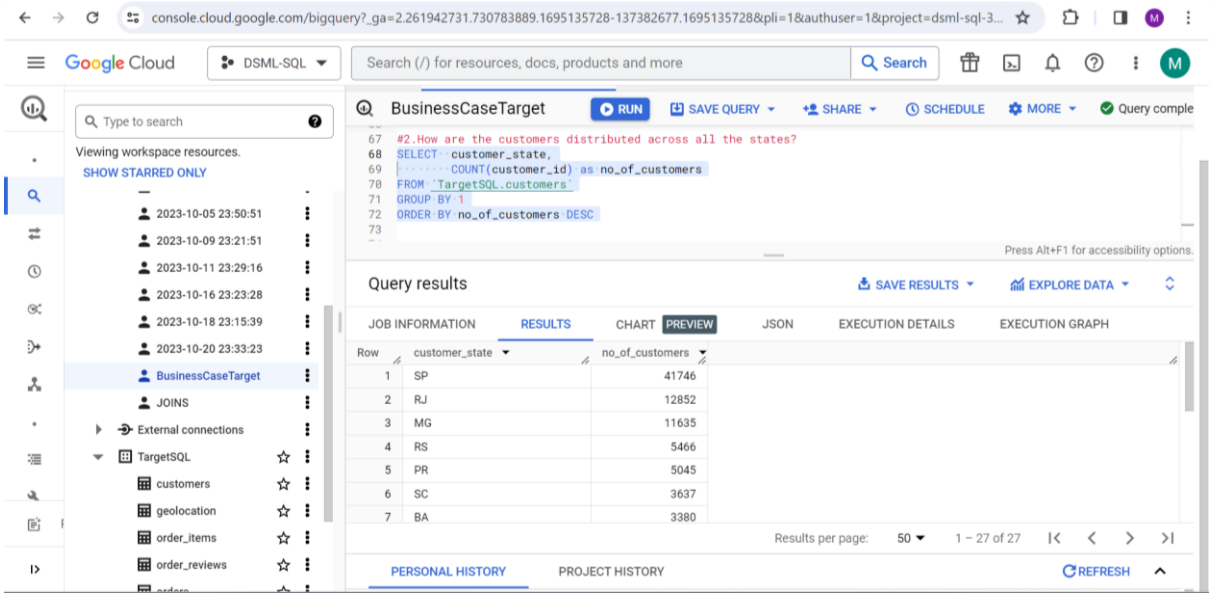


Insights: From the above results we got to know the month-on-month order counts in each state of Brazil and the state SP consistently has the highest number of orders in any given month.

Recommendations: It is recommended to maintain adequate quantity of stock in the state SP as it has the highest number of orders to improve the customer experience.

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

```
SELECT  customer_state,
      COUNT(customer_id) as no_of_customers
FROM `TargetSQL.customers`
GROUP BY 1
ORDER BY no_of_customers DESC
```



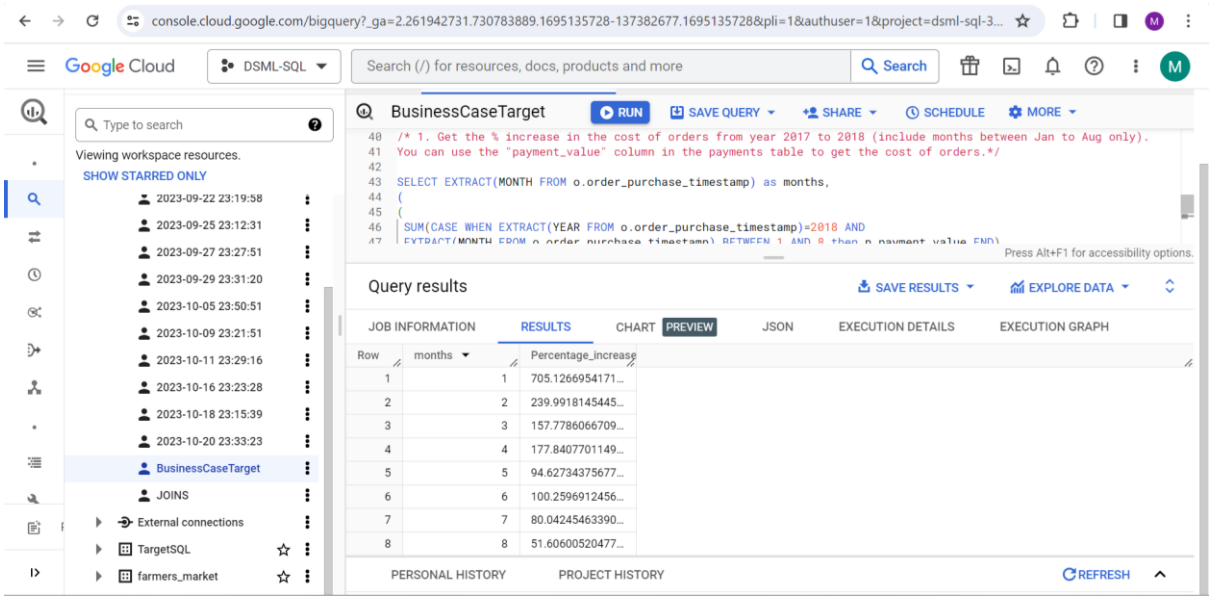
Insights: From the above results we got to know that the state SP has the highest number of customers and the state RR has the lowest number of customers.

Recommendations: It is recommended to maintain adequate quantity of stock the state SP as it has the highest number of customer and it is required to meet customer demands and provide a seamless shopping experience.

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

1. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).You can use the "payment_value" column in the payments table to get the cost of orders.

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) as months,
((
 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 END)
 -
 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 END)
)
/
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
END))*100 as Percentage_increase_payments
FROM `TargetSQL.orders` o
JOIN `TargetSQL.payments` p ON o.order_id=p.order_id
WHERE EXTRACT(YEAR FROM o.order_purchase_timestamp) IN (2017,2018) AND
        EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8
GROUP BY 1
ORDER BY 1
```

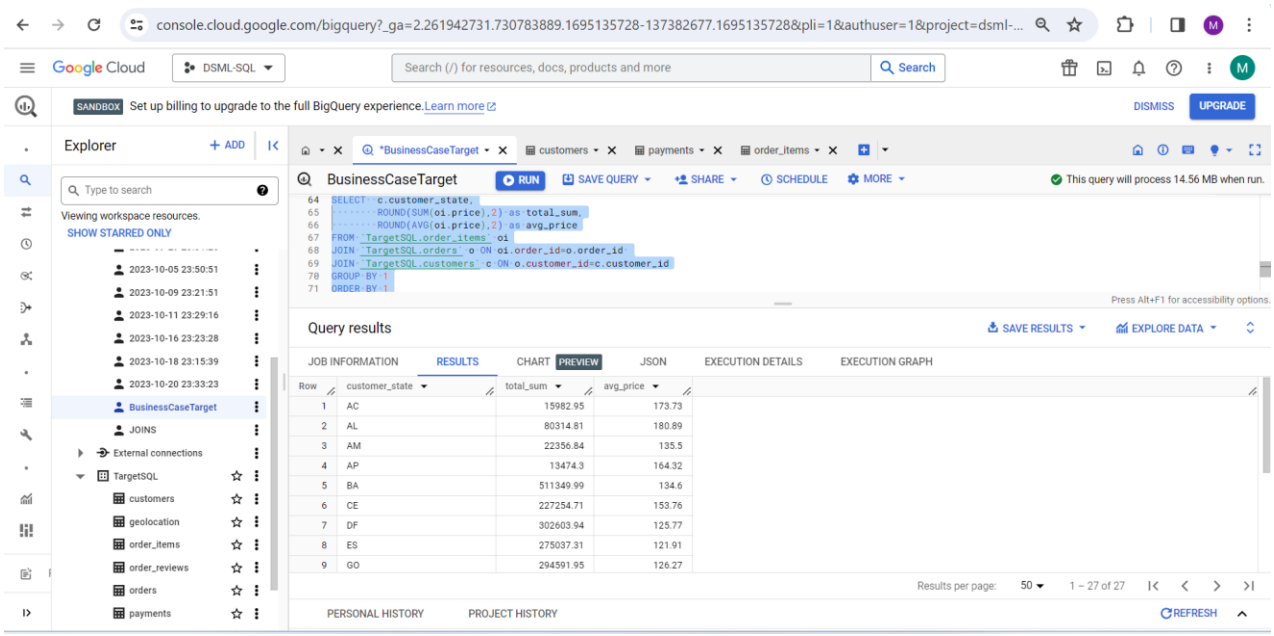


Insights: From the above results we got to know that the January shows the highest percentage increase, followed by February and April.

Recommendation: It is recommended to maintain adequate quantity of stock by identifying peak buying times, and also allocate resources to meet customer demands and provide a seamless shopping experience.

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

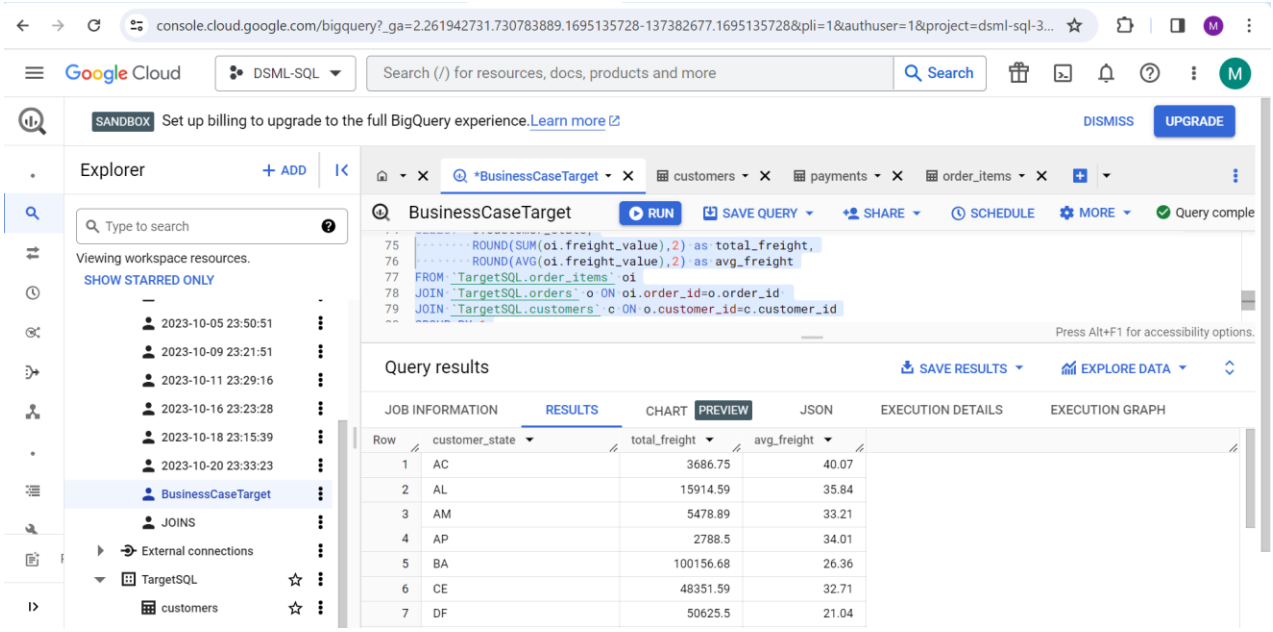
```
SELECT  c.customer_state,
      ROUND(SUM(oi.price),2) as total_sum,
      ROUND(AVG(oi.price),2) as avg_price
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 1
```



Insights: From the above results we got to know that the state SP has the highest total price value and the lowest average price value. The state RR has the lowest total price value and the state PB has the highest average price value.

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

```
SELECT  c.customer_state,
      ROUND(SUM(oi.freight_value),2) as total_freight,
      ROUND(AVG(oi.freight_value),2) as avg_freight
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 1;
```

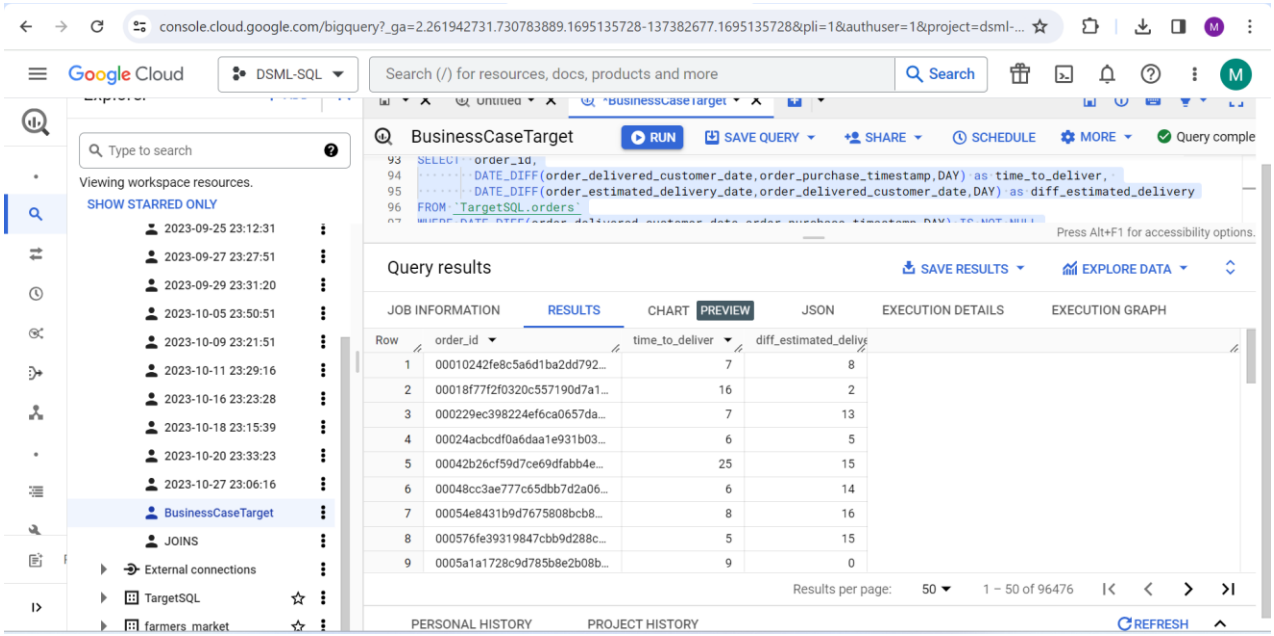


Insights: From the above results we got to know that the state SP has the highest total freight value and the lowest average freight value. The state RR has the lowest total freight value and the state PB has the highest average freight value.

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.

```
SELECT  order_id,
      DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) as time_to_deliver,
      DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) as
diff_estimated_delivery
FROM `TargetSQL.orders`
WHERE DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) IS NOT NULL
AND   DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) IS NOT NULL
ORDER BY 1;
```

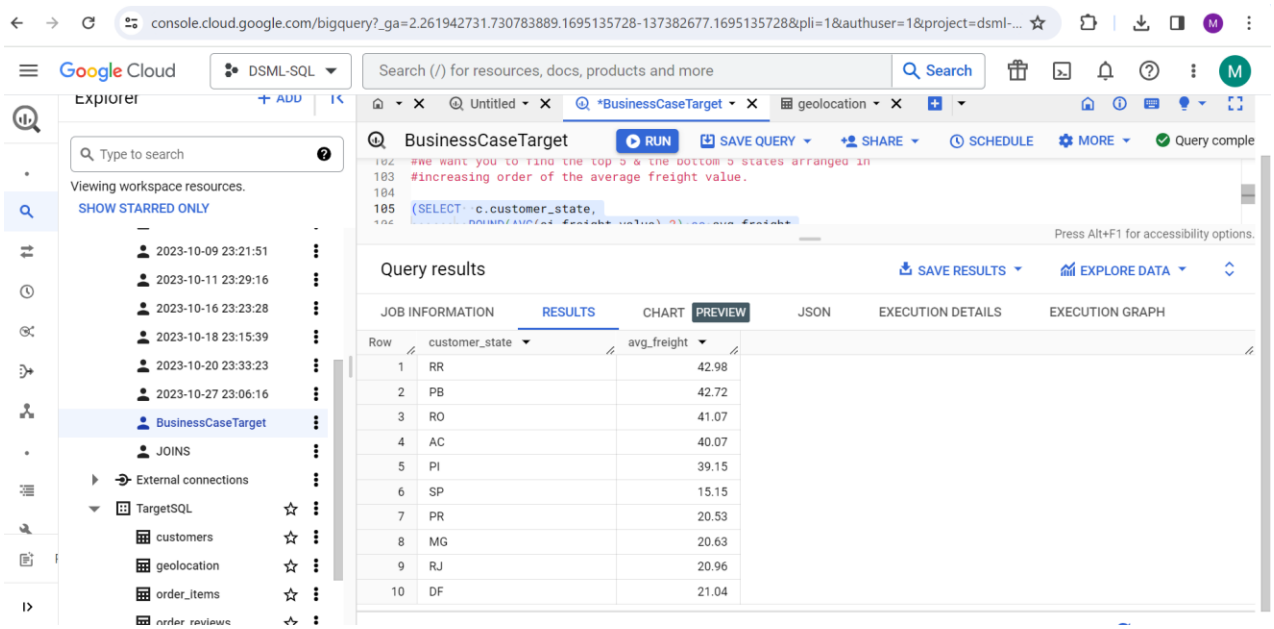


Insights: From the above we got to know the difference between order purchase date and order delivery date and also the difference between order estimate delivery date and order delivery date.

Recommendations: It is recommended to deliver the products soon before order estimate delivery date to improve the customer satisfaction.

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

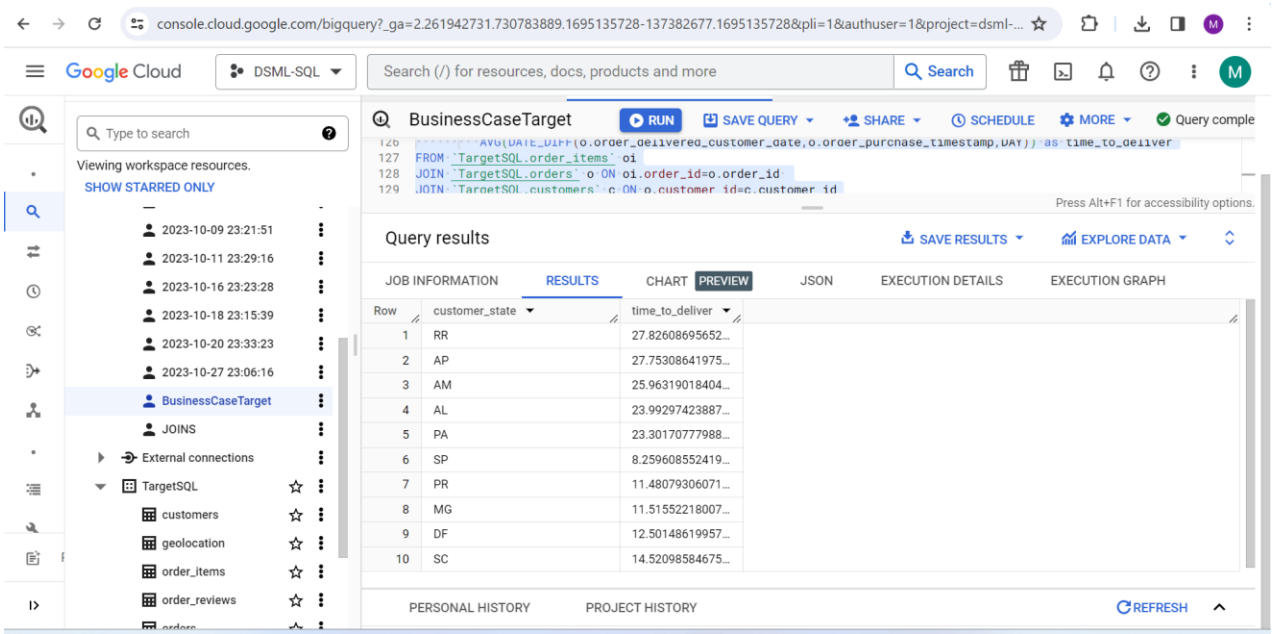
```
(SELECT  c.customer_state,
         ROUND(AVG(oi.freight_value),2) as avg_freight
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5)
UNION ALL
(SELECT  c.customer_state,
         ROUND(AVG(oi.freight_value),2) as avg_freight
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 2
LIMIT 5);
```



Insights: From the above we got to know the top 5 states with the highest & lowest average freight value with state RR being highest and state SP being the lowest.

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

```
(SELECT  c.customer_state,
      AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)) as
time_to_deliver
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5)
UNION ALL
(SELECT  c.customer_state,
      AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)) as
time_to_deliver
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 2
LIMIT 5);
```

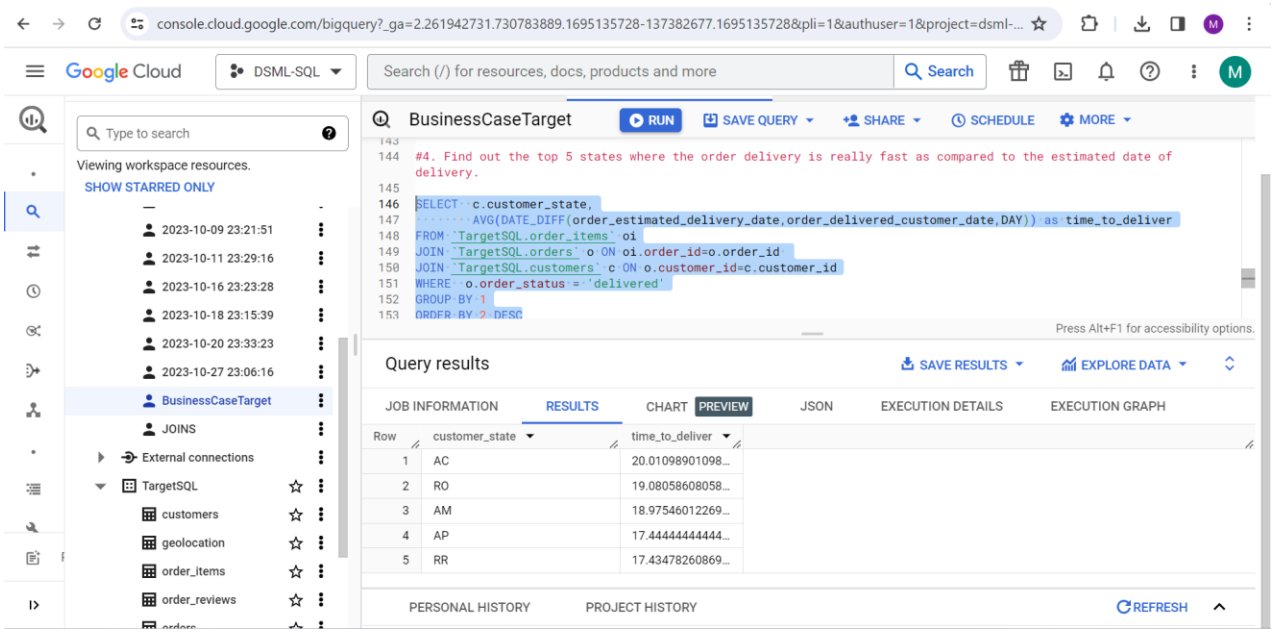


Insights: From the above we got to know the top 5 states with the highest & lowest average delivery time in days out of which the state RR being the highest avg delivery time and the state SP has the lowest avg delivery time.

Recommendations: It is recommended to deliver the products soon so that the avg delivery time should be less to improve the customer satisfaction.

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

```
SELECT  c.customer_state,
      AVG(DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)) as time_to_deliver
FROM `TargetSQL.order_items` oi
JOIN `TargetSQL.orders` o ON oi.order_id=o.order_id
JOIN `TargetSQL.customers` c ON o.customer_id=c.customer_id
WHERE  o.order_status = 'delivered'
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```



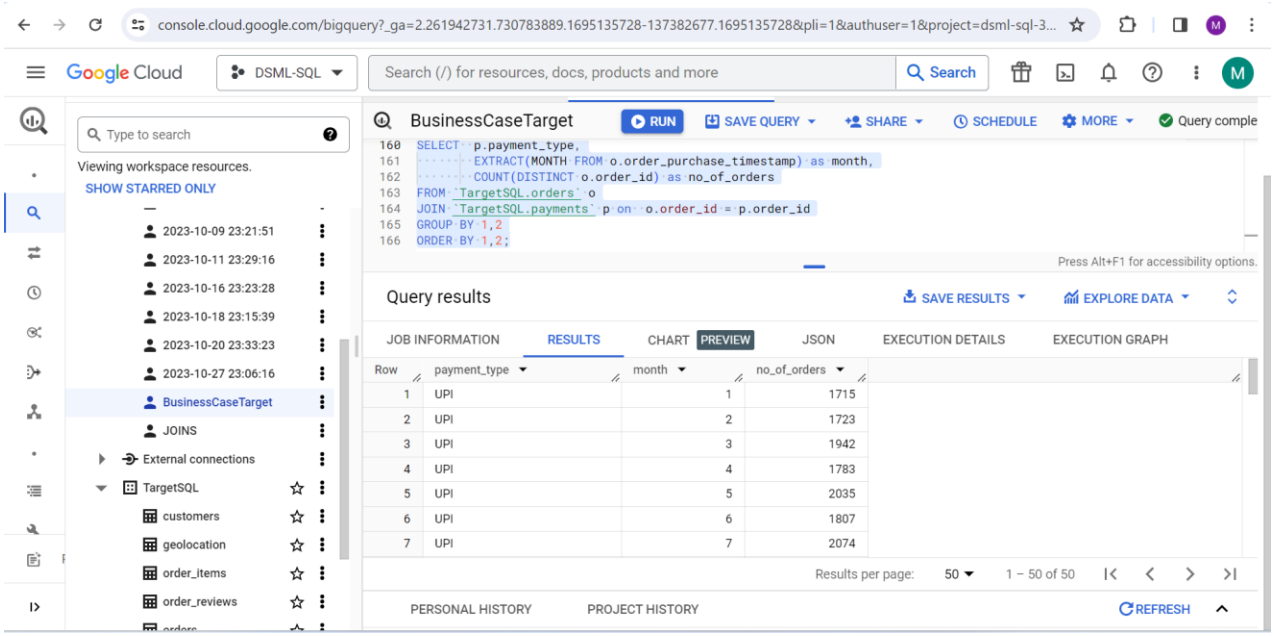
Insights: From the above we got to know 5 states where the order delivery is really fast as compared to the estimated date of delivery.

Recommendations: It is recommended to deliver the products soon improve the customer satisfaction.

6.Analysis based on the payments:

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

```
SELECT  p.payment_type,
      EXTRACT(MONTH FROM o.order_purchase_timestamp) as month,
      COUNT(DISTINCT o.order_id) as no_of_orders
FROM `TargetSQL.orders` o
JOIN `TargetSQL.payments` p on  o.order_id = p.order_id
GROUP BY 1,2
ORDER BY 1,2;
```

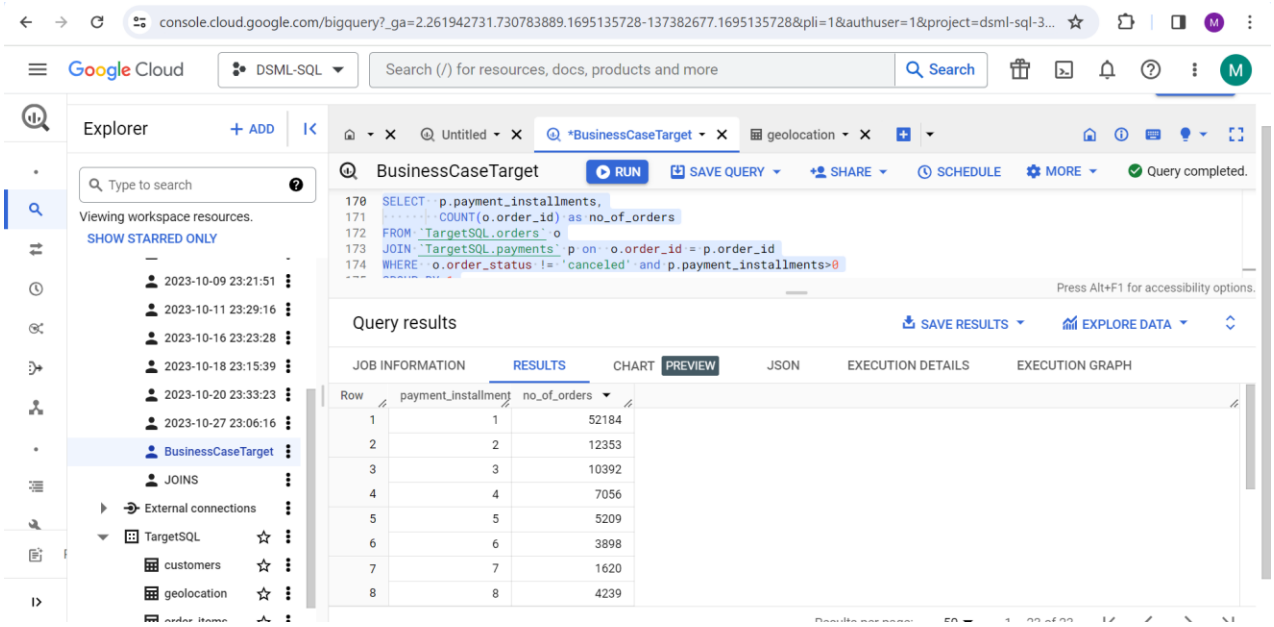


Insights: From the above query we got to know the month on month no. of orders placed using different payment types where Credit card transactions are the most popular payment method, followed by UPI and Debit card transactions are the least preferred option.

Recommendations: It is recommended to maintain less chances of payment failure when a customer makes a purchase through credit card as most of the customer are making the payments through credit card to improve the customer satisfaction.

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

```
SELECT  p.payment_installments,
      COUNT(o.order_id) as no_of_orders
FROM `TargetSQL.orders` o
JOIN `TargetSQL.payments` p on  o.order_id = p.order_id
WHERE  o.order_status != 'canceled' and p.payment_installments>0
GROUP BY 1
ORDER BY 1;
```



Insights: From the above query we got to know that the majority of orders have only one payment installment and the highest number of installment is 24 which has 18 orders.

Recommendations: It is recommended to improve payment options as most of the customers are going for single installment payment there should not be any difficulty while doing the transaction to improve the customer satisfaction.