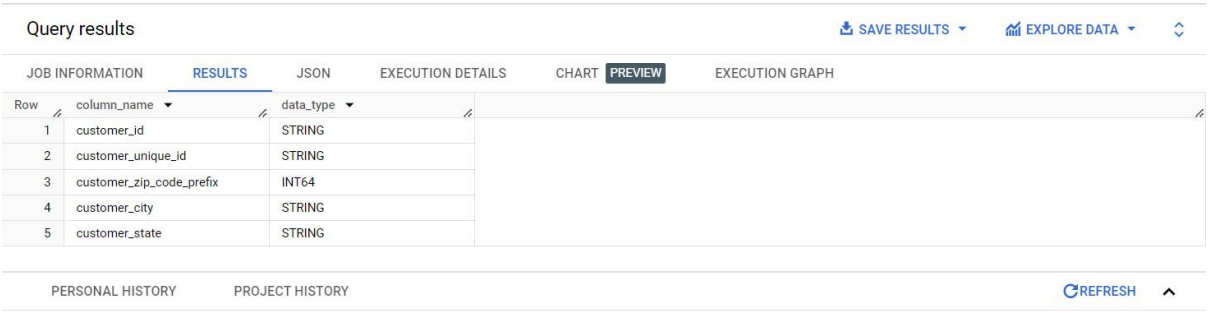
1.Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

A)Data type of all columns in the "customers" table.

Query:

```
SELECT column_name,data_type
FROM `casestudy`.INFORMATION_SCHEMA.COLUMNS
WHERE table_name="customers"
```

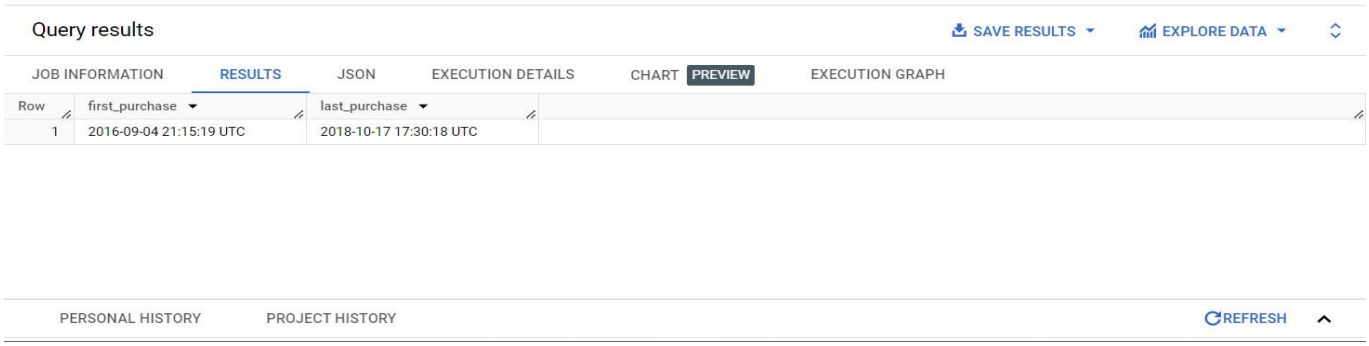


INSIGHT: Datatype “STRING” is most commonly used.

B) Get the time range between which the orders were placed.

Query:

```
SELECT MIN(order_purchase_timestamp) AS first_purchase,MAX(order_purchase_timestamp) AS last_purchase FROM `casestudy.orders`
```



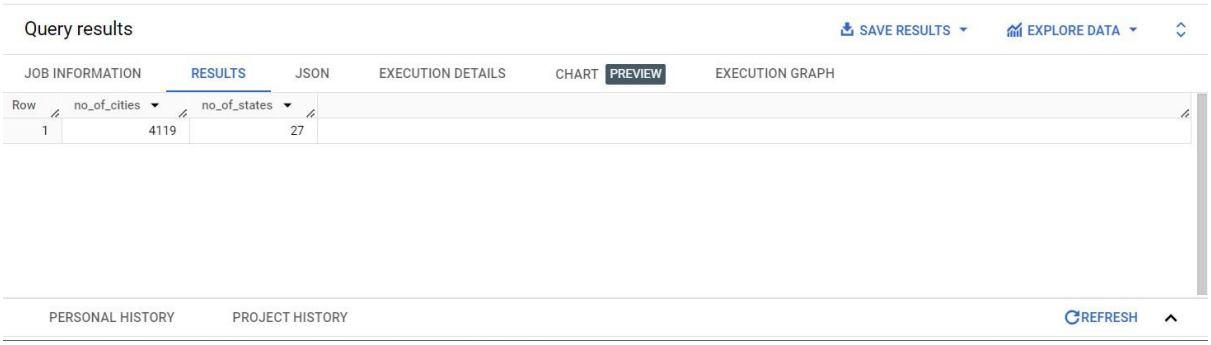
INSIGHT: The orders were placed between 2016 and 2018, indicating a data set spanning two years.

C)Count the Cities & States of customers who ordered during the given period

Query:

```
SELECT COUNT(DISTINCT c.customer_city) AS no_of_cities,

COUNT(DISTINCT c.customer_state) AS states
      FROM `casestudy.customers` c
JOIN `casestudy.orders` o
      ON c.customer_id=o.customer_id
```



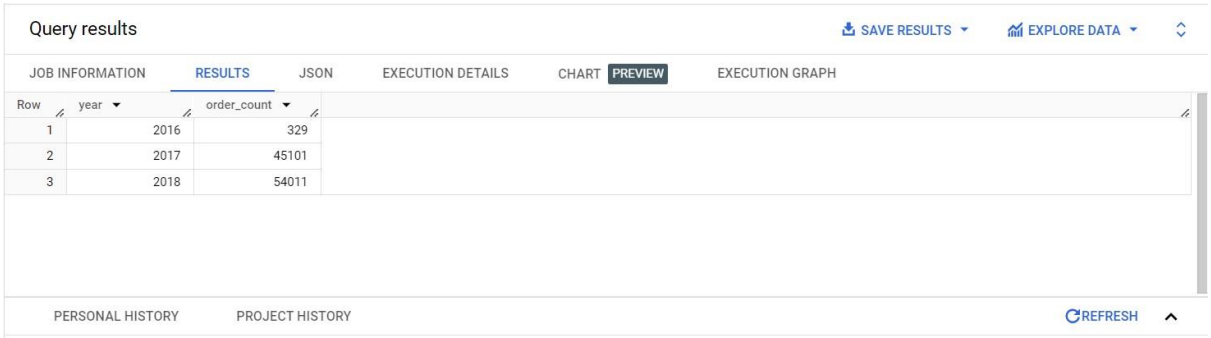
INSIGHT: Unique cities and states were recorded.

2.In-depth Exploration:

A)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 order_count
FROM `casestudy.orders`
GROUP BY year
ORDER BY year
```



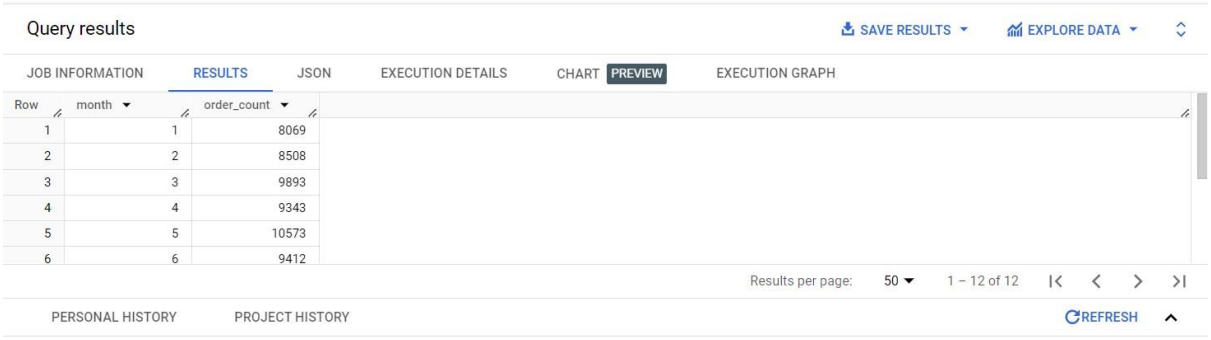
INSIGHT: Yes, there is a growing trend in the number of orders placed, with a significant increase.

B)Can 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 month,COUNT(*) AS order_count
```

```
FROM `casestudy.orders`
GROUP BY month
ORDER BY month
```



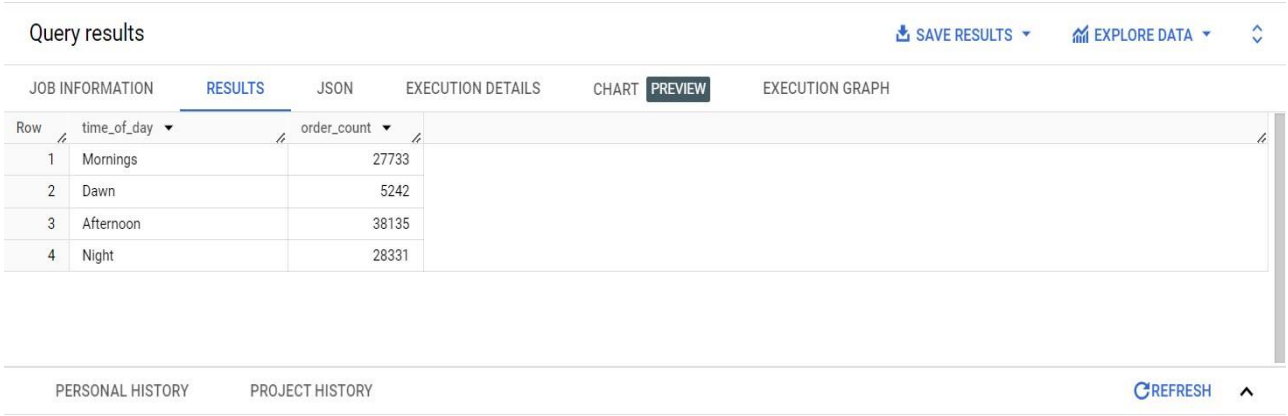
INSIGHT: Yes, there is monthly seasonality in the order data, with varying order counts each month, such as 8,069 in January and 10,573 in May.

C) 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

Query:

```
SELECT
CASE WHEN EXTRACT(HOUR FROM o.order_purchase_timestamp) BETWEEN 0 AND 6 THEN "Dawn"
WHEN EXTRACT(HOUR FROM o.order_purchase_timestamp) BETWEEN 7 AND 12 THEN "Mornings"
WHEN EXTRACT(HOUR FROM o.order_purchase_timestamp) BETWEEN 13 AND 18 THEN "Afternoon"
WHEN EXTRACT(HOUR FROM o.order_purchase_timestamp) BETWEEN 19 AND 23 THEN "Night" END AS time_of_day, COUNT(*) AS order_count
FROM `casestudy.orders` o
JOIN `casestudy.customers` c
ON c.customer_id=o.customer_id GROUP BY
time_of_day
```



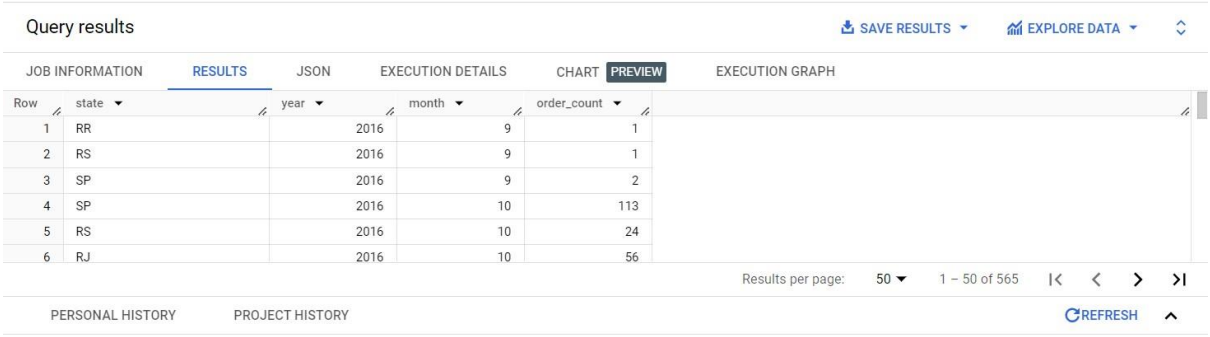
INSIGHT: Brazilian customers mostly place their orders in the afternoon, with 38,135 orders during this time of day.

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

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

Query:

```
SELECT c.customer_state AS state,
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
COUNT(*) AS order_count
FROM `casestudy.customers` c
JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
GROUP BY state,year,month
ORDER BY year,month
```



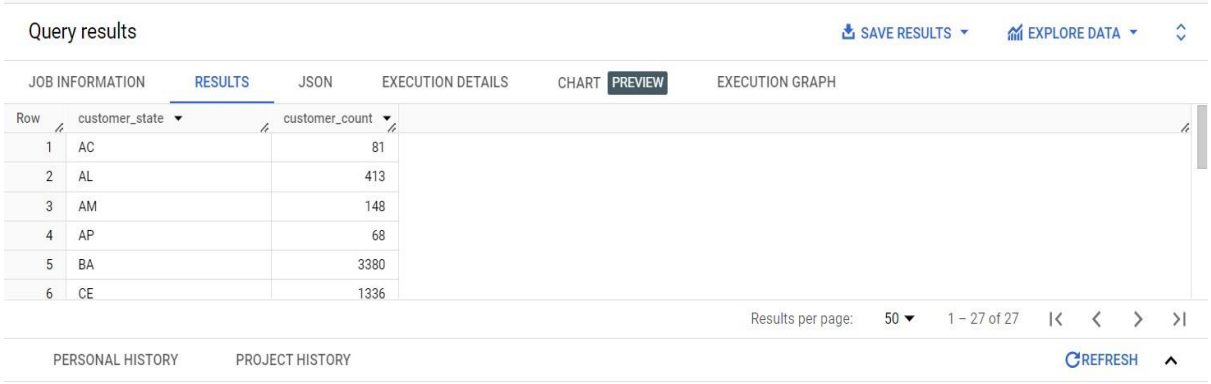
INSIGHT: The evolution of e-commerce orders in Brazil shows a gradual increase over time.

B) How are the customers distributed across all the states

Query:

```
SELECT customer_state,COUNT(*) AS customer_count

FROM `casestudy.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.

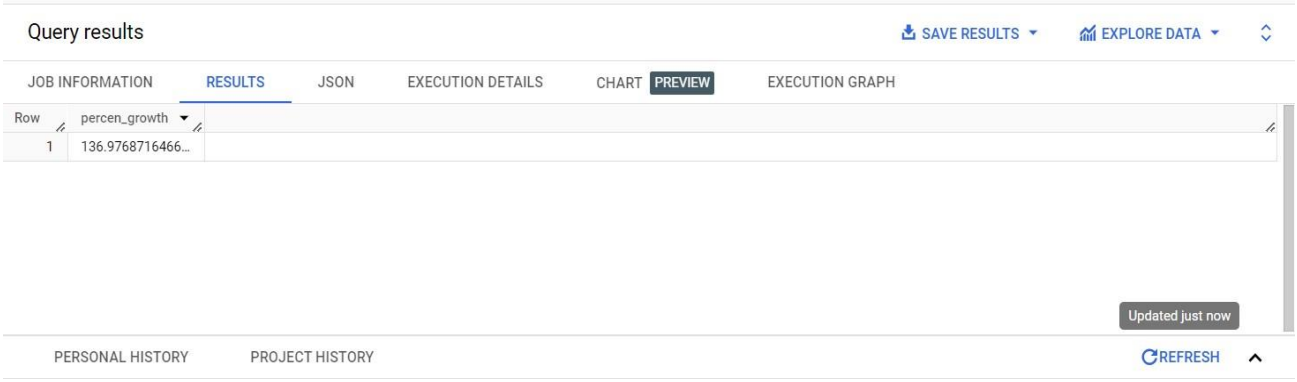
A)Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

You can use the "payment_value" column in the payments table to get the cost of orders.

Query:

```
WITH cte AS (SELECT EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,

EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
SUM(p.payment_value) AS revenue
FROM `casestudy.orders` o
JOIN `casestudy.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 year,month)
SELECT ((SUM(p2.revenue)-SUM(p1.revenue))*100)/SUM(p1.revenue) AS percen_growth FROM
cte p1
JOIN cte p2
ON p1.month=p2.month AND p1.year=2017 AND p2.year=2018
```

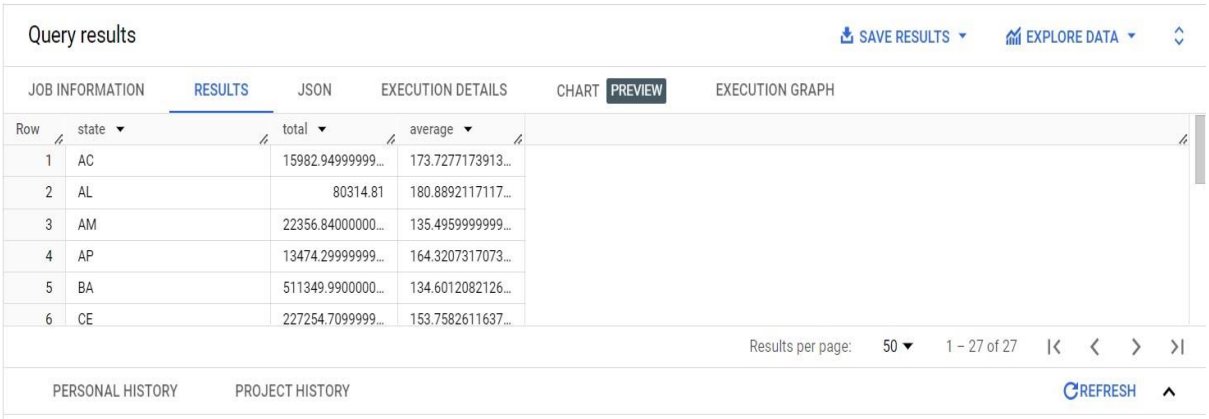


B) Calculate the Total & Average value of order price for each state

Query:

```
SELECT c.customer_state AS state,
SUM(i.price) AS total,AVG(i.price) AS average
FROM `casestudy.customers` c
        JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id

JOIN `casestudy.orderitems` i
        ON o.order_id=i.order_id
GROUP BY state
    ORDER BY state
```

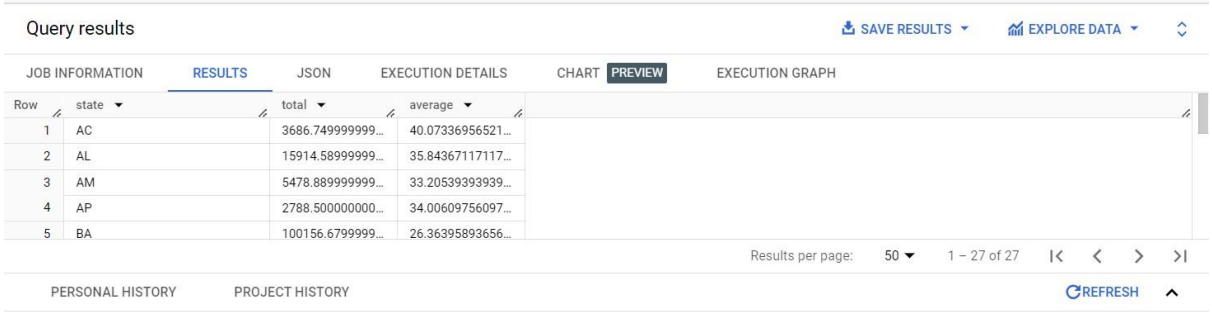


C) Calculate the Total & Average value of order freight for each state

Query:

```
SELECT c.customer_state AS state,
```

```
SUM(i.freight_value) AS total,
        AVG(i.freight_value) AS average
FROM `casestudy.customers` c
       JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
        JOIN `casestudy.orderitems` i
ON o.order_id=i.order_id
   GROUP BY state
ORDER BY state
```



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

A)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.

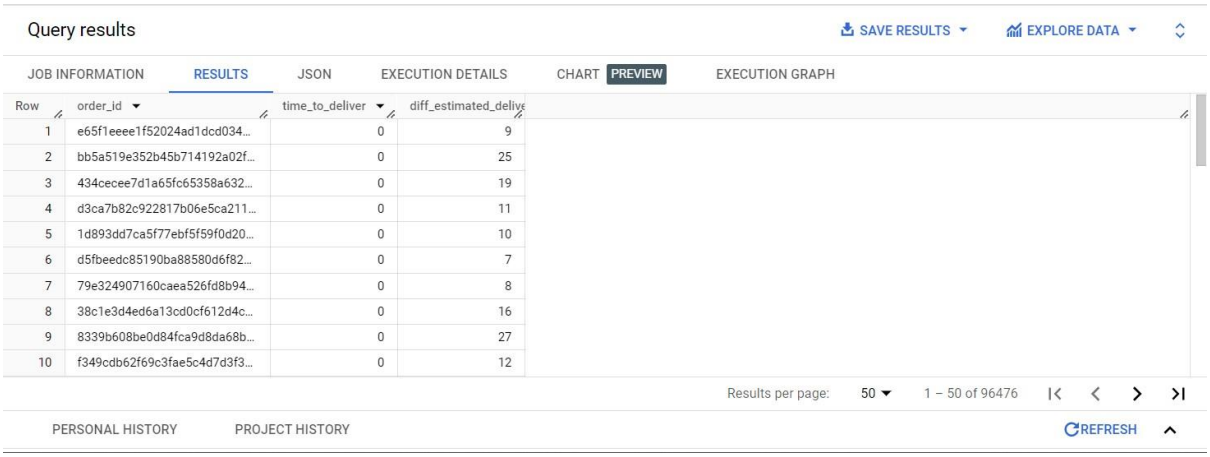
Do this in a single query.

You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

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 `casestudy.orders`
WHERE order_delivered_customer_date IS NOT NULL ORDER BY
time_to_deliver
```



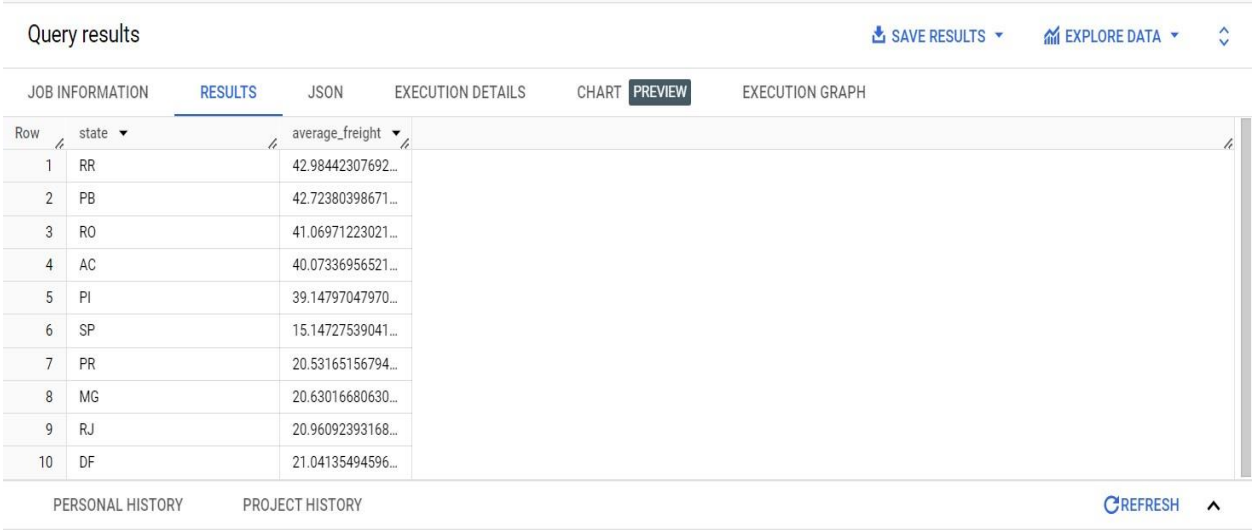
Row	order_id	time_to_deliver	diff_estimated_deliv
1	e65f1eeee1f52024ad1dcd034...	0	9
2	bb5a519e352b45b714192a02f...	0	25
3	434cecee7d1a65fc65358a632...	0	19
4	d3ca7b82c922817b06e5ca211...	0	11
5	1d893dd7ca5f77ebf5f59f0d20...	0	10
6	d5fbeedc85190ba88580d6f82...	0	7
7	79e324907160caea526fd8b94...	0	8
8	38c1e3d4ed6a13cd0cf612d4c...	0	16
9	8339b608be0d84fca9d8da68b...	0	27
10	f349cdb62f69c3fae5c4d7d3f3...	0	12

INSIGHT: The estimated delivery is changing on everyorder.

B) Find out the top 5 states with the highest & lowest average freight value

Query:

```
        (SELECT c.customer_state AS state,
          AVG(i.freight_value) AS average_freight
FROM `casestudy.customers` c
      JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
       JOIN `casestudy.orderitems` i
ON o.order_id=i.order_id
   GROUP BY state
     ORDER BY average_freight DESC
LIMIT 5)
  UNION ALL
(SELECT c.customer_state AS state,
          AVG(i.freight_value) AS average_freight
FROM `casestudy.customers` c
      JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
       JOIN `casestudy.orderitems` i
ON o.order_id=i.order_id
   GROUP BY state
ORDER BY average_freight
  LIMIT 5)
```



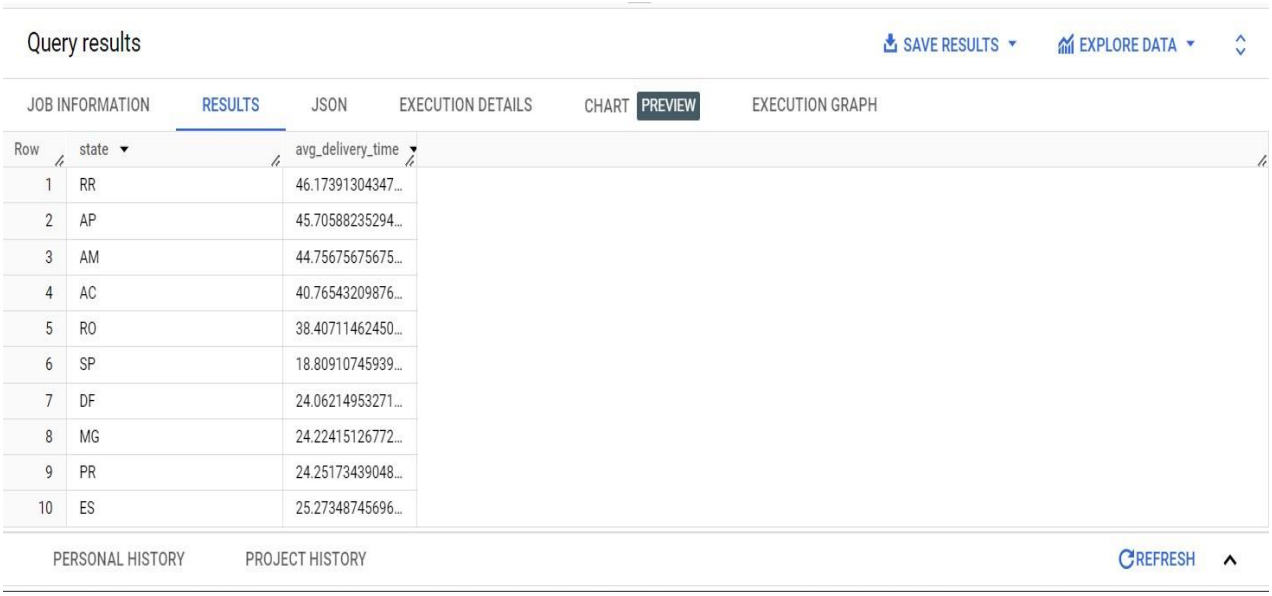
Row	state	average_freight
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...
6	SP	15.14727539041...
7	PR	20.53165156794...
8	MG	20.63016680630...
9	RJ	20.96092393168...
10	DF	21.04135494596...

C) Find out the top 5 states with the highest & lowest average delivery time

Query:

```
(SELECT c.customer_state AS state,

AVG(DATE_DIFF(order_estimated_delivery_date,order_purchase_timestamp,DAY)) AS avg_delivery_time FROM `casestudy.customers` c
JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
GROUP BY state
ORDER BY avg_delivery_time DESC
LIMIT 5)
UNION ALL
(SELECT c.customer_state AS state,
AVG(DATE_DIFF(order_estimated_delivery_date,order_purchase_timestamp,DAY)) AS avg_delivery_time FROM `casestudy.customers` c
JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
GROUP BY state
ORDER BY avg_delivery_time LIMIT 5)
```



Row	state	avg_delivery_time
1	RR	46.17391304347...
2	AP	45.70588235294...
3	AM	44.75675675675...
4	AC	40.76543209876...
5	RO	38.40711462450...
6	SP	18.80910745939...
7	DF	24.06214953271...
8	MG	24.22415126772...
9	PR	24.25173439048...
10	ES	25.27348745696...

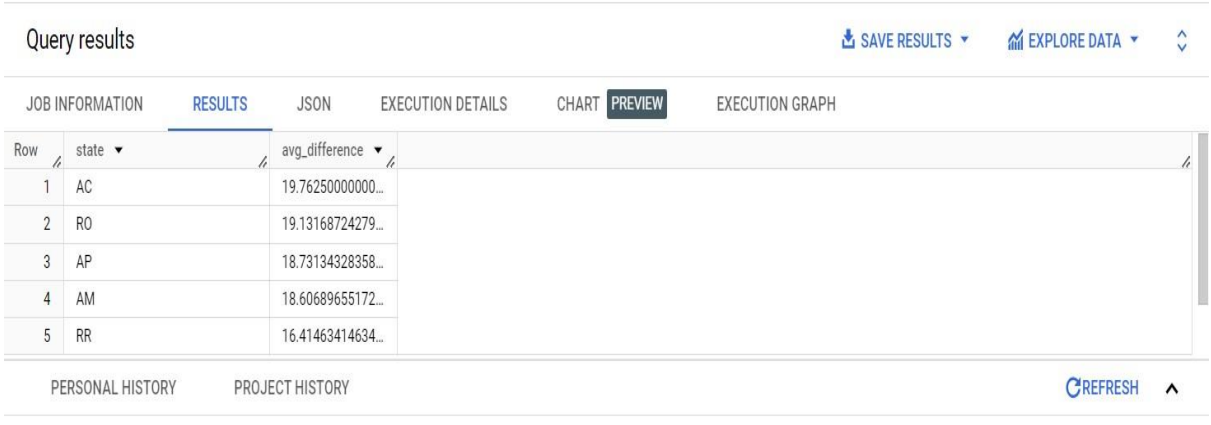
D) Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

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:

```
SELECT c.customer_state AS state,

AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date,DAY)) AS avg_difference FROM `casestudy.customers` c
JOIN `casestudy.orders` o
ON c.customer_id=o.customer_id
WHERE o.order_delivered_customer_date IS NOT NULL
GROUP BY state
ORDER BY avg_difference DESC LIMIT 5
```



Row	state	avg_difference
1	AC	19.76250000000...
2	RO	19.13168724279...
3	AP	18.73134328358...
4	AM	18.60689655172...
5	RR	16.41463414634...

6. **Analysis based on the payments:**

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

Query:

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

Query results

SAVE RESULTS | EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW | EXECUTION GRAPH

Row	year	month	payment_types	order_count
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
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	583
9	2017	1	debit_card	9
10	2017	1	voucher	61

Results per page: 50 | 1 – 50 of 90

PERSONAL HISTORY | PROJECT HISTORY | REFRESH

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

Query:

```
SELECT payment_installments,COUNT(*) AS order_count
    FROM `casestudy.orders` o
JOIN `casestudy.payments` p
    ON o.order_id=p.order_id
WHERE payment_sequential>=1
GROUP BY payment_installments
ORDER BY payment_installments
```

Query results

SAVE RESULTS | EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW | EXECUTION GRAPH

Row	payment_installment	order_count
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239

Results per page: 50 1 – 24 of 24

PERSONAL HISTORY | PROJECT HISTORY

REFRESH