**Project: Target-SQL Business Case**

* **Mrudul Jain**

Q1. 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
2. Time period for which the data is given
3. Cities and States covered in the dataset

Ans:

1. SELECT table\_name, column\_name, data\_type

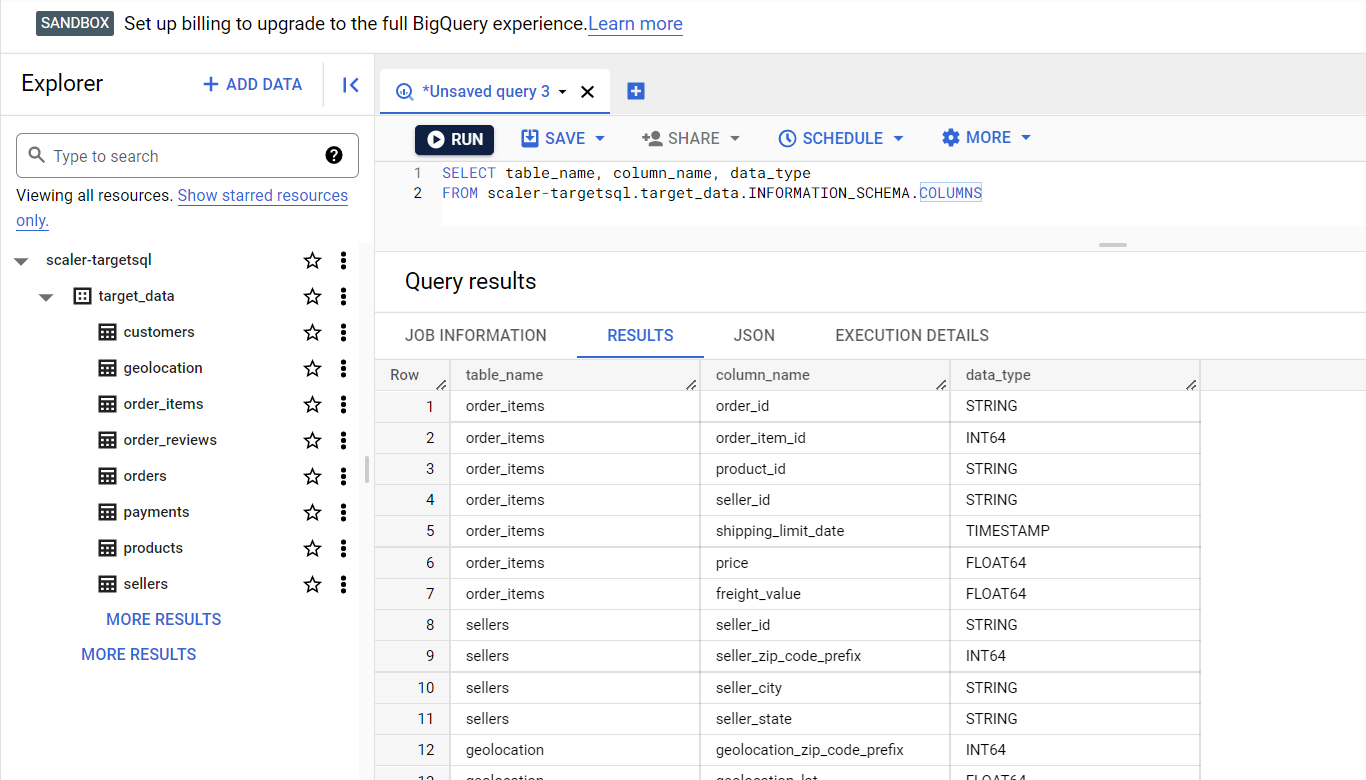
FROM scaler-targetsql.target\_data.INFORMATION\_SCHEMA.COLUMNS

To get specific table:

SELECT table\_name, column\_name, data\_type

FROM scaler-targetsql.target\_data.INFORMATION\_SCHEMA.COLUMNS

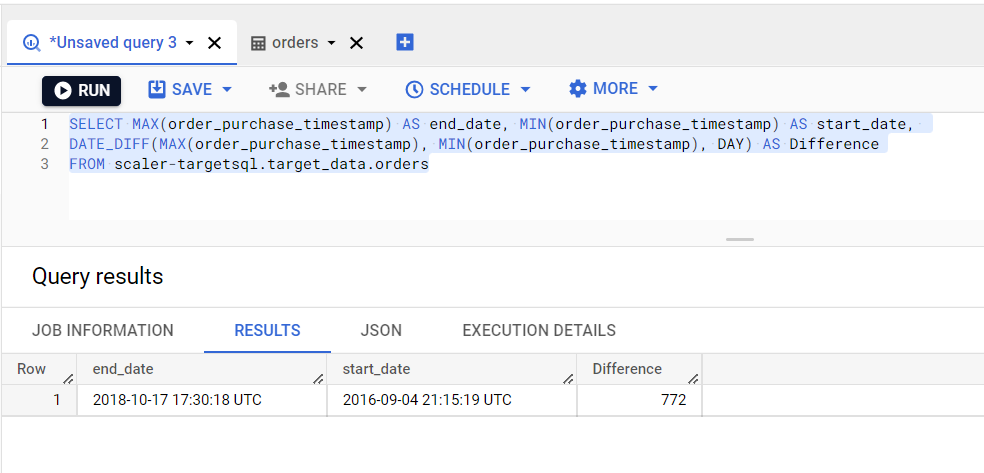
WHERE table\_name = "orders"

****

1. SELECT MAX(order\_purchase\_timestamp) AS end\_date, MIN(order\_purchase\_timestamp) AS start\_date,

DATE\_DIFF(MAX(order\_purchase\_timestamp), MIN(order\_purchase\_timestamp), DAY) AS Difference

FROM scaler-targetsql.target\_data.orders

****

1. SELECT DISTINCT geolocation\_city AS city, geolocation\_state AS state, FROM scaler-targetsql.target\_data.geolocation

To get count of cities within each state:

SELECT geolocation\_state AS state, COUNT(DISTINCT(geolocation\_city)) AS city\_count,

FROM scaler-targetsql.target\_data.geolocation

GROUP BY geolocation\_state

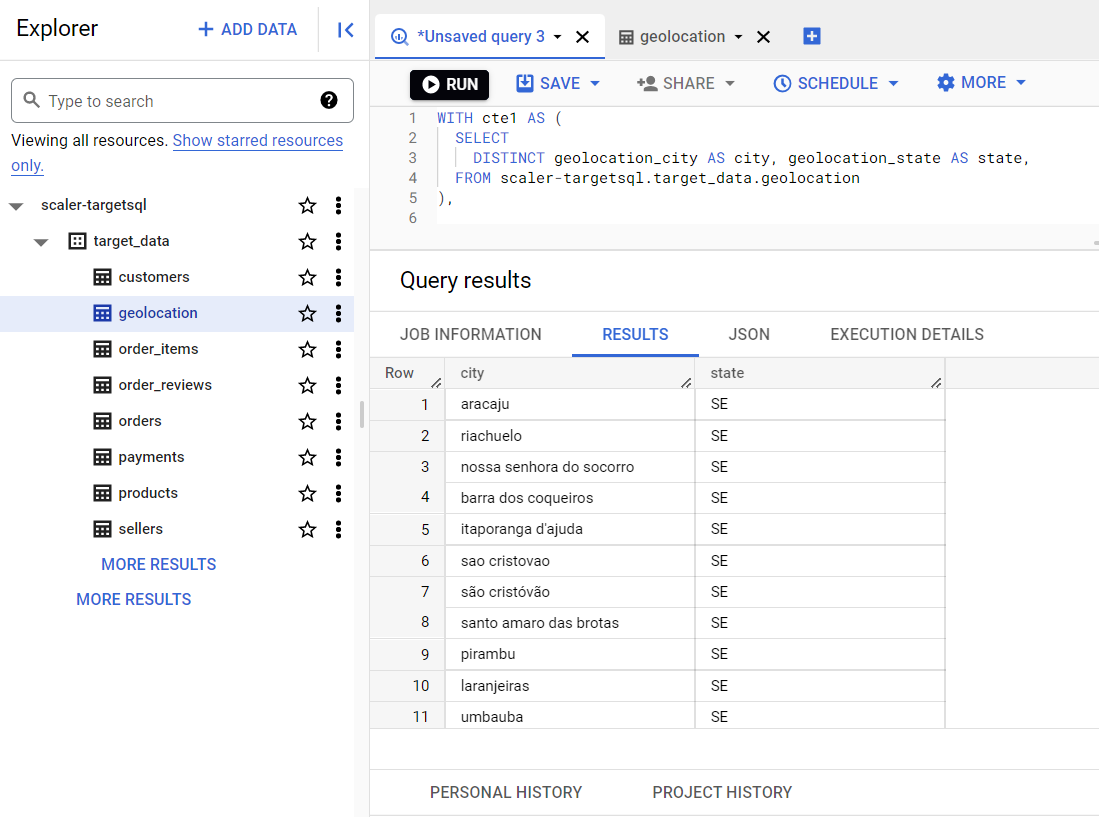
To get total count of cities and states

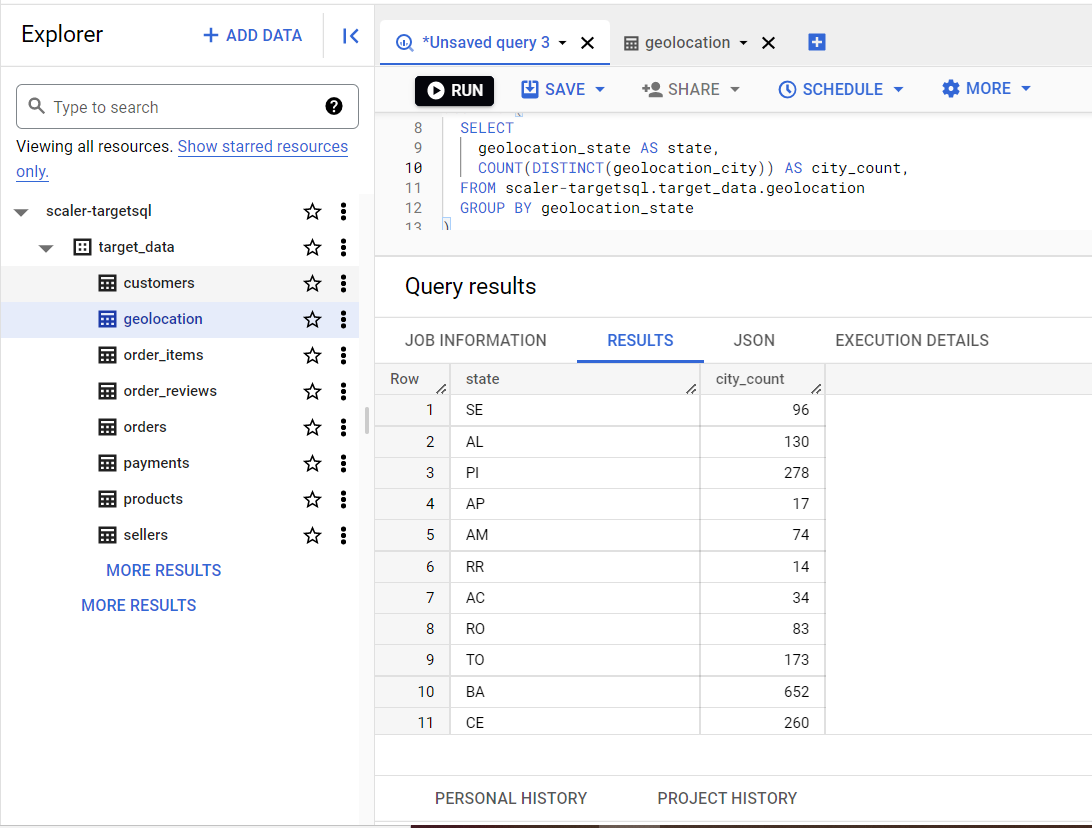
SELECT

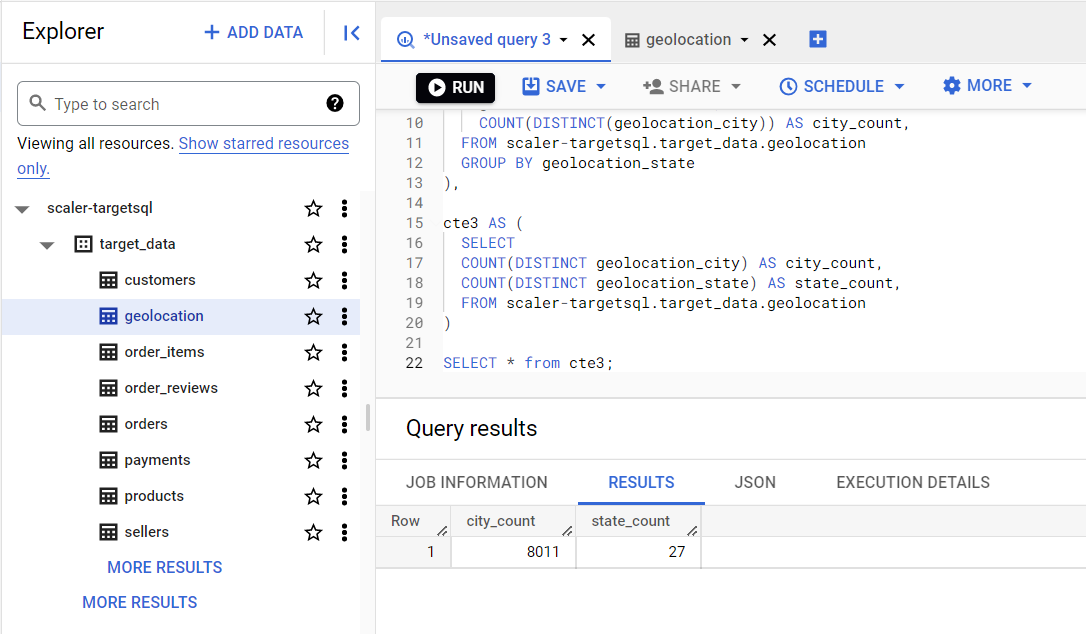
COUNT(DISTINCT geolocation\_city) AS city\_count,

COUNT(DISTINCT geolocation\_state) AS state\_count,

FROM scaler-targetsql.target\_data.geolocation

****

****

****

Q2. 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?
  2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Ans:

1. SELECT

DISTINCT EXTRACT(MONTH FROM order\_purchase\_timestamp) AS months,

COUNT(orders.order\_id) OVER (PARTITION BY EXTRACT(MONTH FROM order\_purchase\_timestamp)) AS order\_count,

ROUND(SUM(payments.payment\_value) OVER (PARTITION BY EXTRACT(MONTH FROM order\_purchase\_timestamp)) , 2) AS total\_purchase\_value

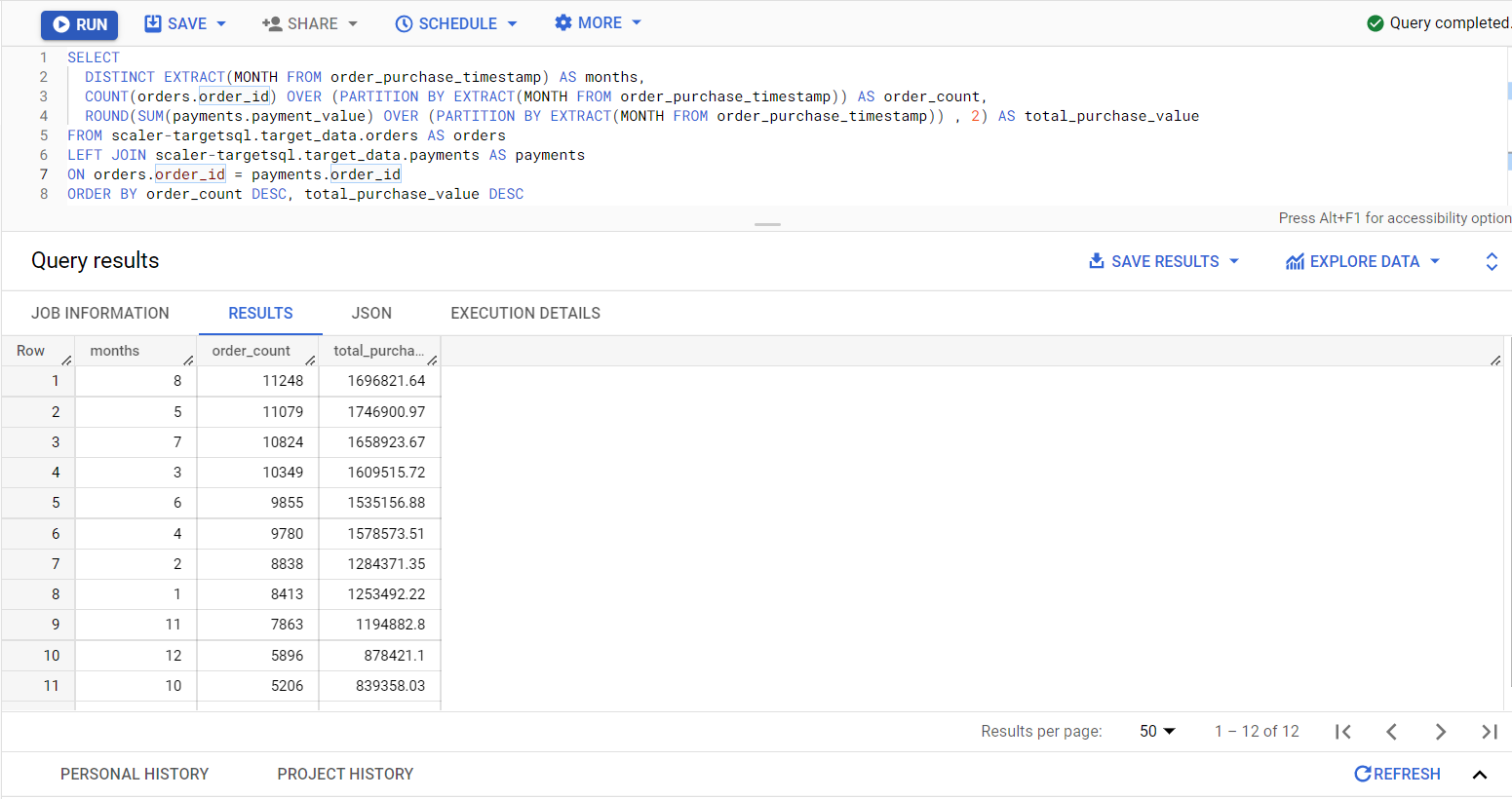
FROM scaler-targetsql.target\_data.orders AS orders

LEFT JOIN scaler-targetsql.target\_data.payments AS payments

ON orders.order\_id = payments.order\_id

ORDER BY order\_count DESC, total\_purchase\_value DESC

// This query checks seasonality based on order count and total purchase value of good. As we can clearly see from the output(in screenshot). In the winter months (Sept 🡪 Feb) the sales are low as compared to the summer months(March 🡪 August)



To see yearly trend:

SELECT

EXTRACT(YEAR FROM order\_purchase\_timestamp) AS years,

COUNT(orders.order\_id) AS total\_orders,

SUM(price) AS total\_price

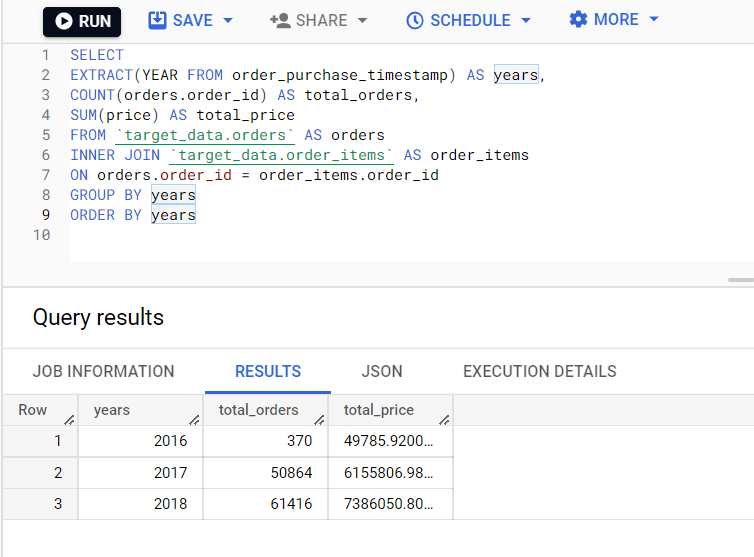
FROM `target\_data.orders` AS orders

INNER JOIN `target\_data.order\_items` AS order\_items

ON orders.order\_id = order\_items.order\_id

GROUP BY years

ORDER BY years



To see quarterly trend:

SELECT

EXTRACT(YEAR FROM order\_purchase\_timestamp) AS years,

EXTRACT(QUARTER FROM order\_purchase\_timestamp) AS quarters,

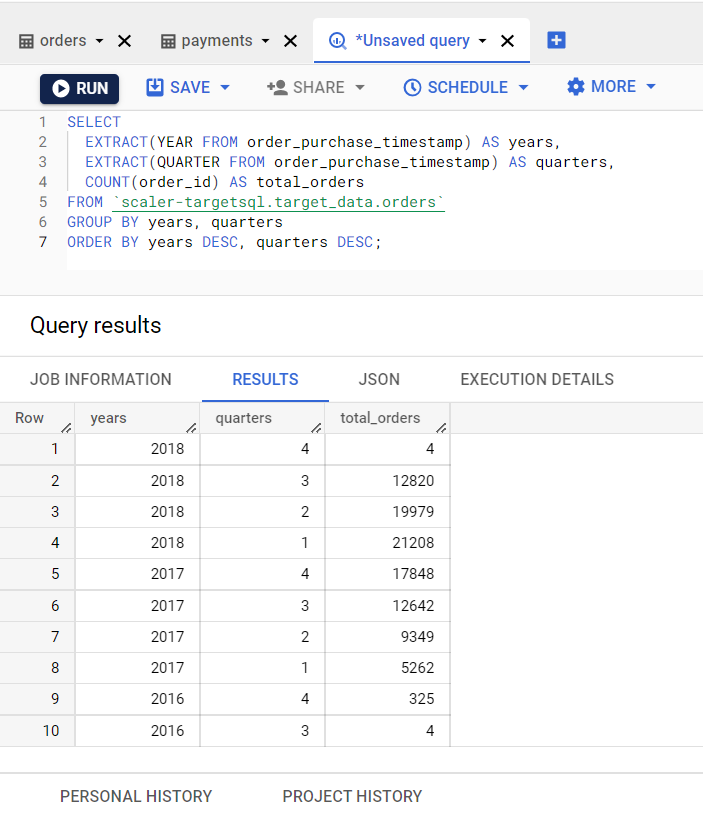
COUNT(order\_id) AS total\_orders

FROM `scaler-targetsql.target\_data.orders`

GROUP BY years, quarters

ORDER BY years DESC, quarters DESC;

// As we can see, in Q3 and Q4 of 2016, there were very few sales as compared to Q3 and Q4 of 2017. And there has been an increase in sales in all the quarters from 2017 to 2018. So, the market for e-commerce increased very quickly from 2016 🡪 2017 but it has slowed down by Q3 of 2018



1. SELECT

   CASE

     WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) <= 6

       THEN 'DAWN (12AM - 6AM)'

     WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) <= 12

       THEN 'MORNING (6AM - 12PM)'

     WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) <= 18

       THEN 'AFTERNOON (12PM - 6PM)'

     WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) <= 24

       THEN 'EVENING (6PM - 12AM)'

     ELSE 'MISC'

   END AS Timings,

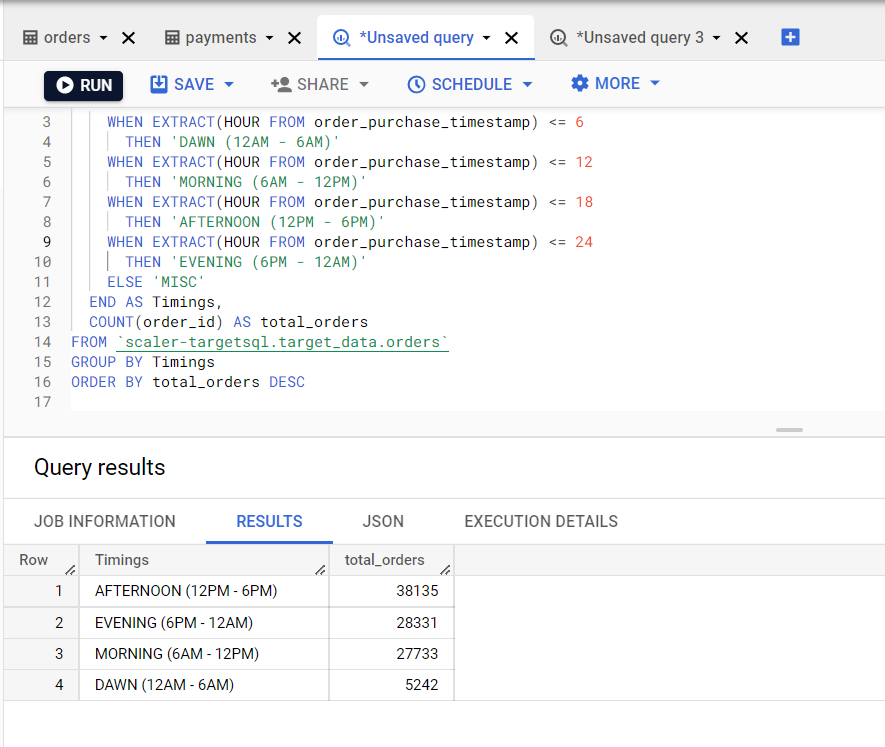
   COUNT(order\_id) AS total\_orders

FROM `scaler-targetsql.target\_data.orders`

GROUP BY Timings

ORDER BY total\_orders DESC

// Most customers shop during the afternoon hours in Brazil followed by the morning and night hours (equal). The dawn/mid-night hours are the least shopped. So, in-case we need to do server maintenance we can do it during midnight time. And our servers need to be able to handle most load during the day (6am 🡪 12am) timings



Q3. Evolution of E-commerce orders in the Brazil region:

* 1. Get month on month orders by region, states
  2. How are customers distributed in Brazil

Ans:

1. SELECT

DISTINCT EXTRACT(MONTH FROM order\_purchase\_timestamp) AS month

   COUNT(order\_id) AS order\_count,

   geolocation.geolocation\_state AS state\_name

FROM `scaler-targetsql.target\_data.geolocation` AS geolocation

LEFT JOIN `scaler-targetsql.target\_data.customers` AS customers

ON geolocation.geolocation\_zip\_code\_prefix = customers.customer\_zip\_code\_prefix

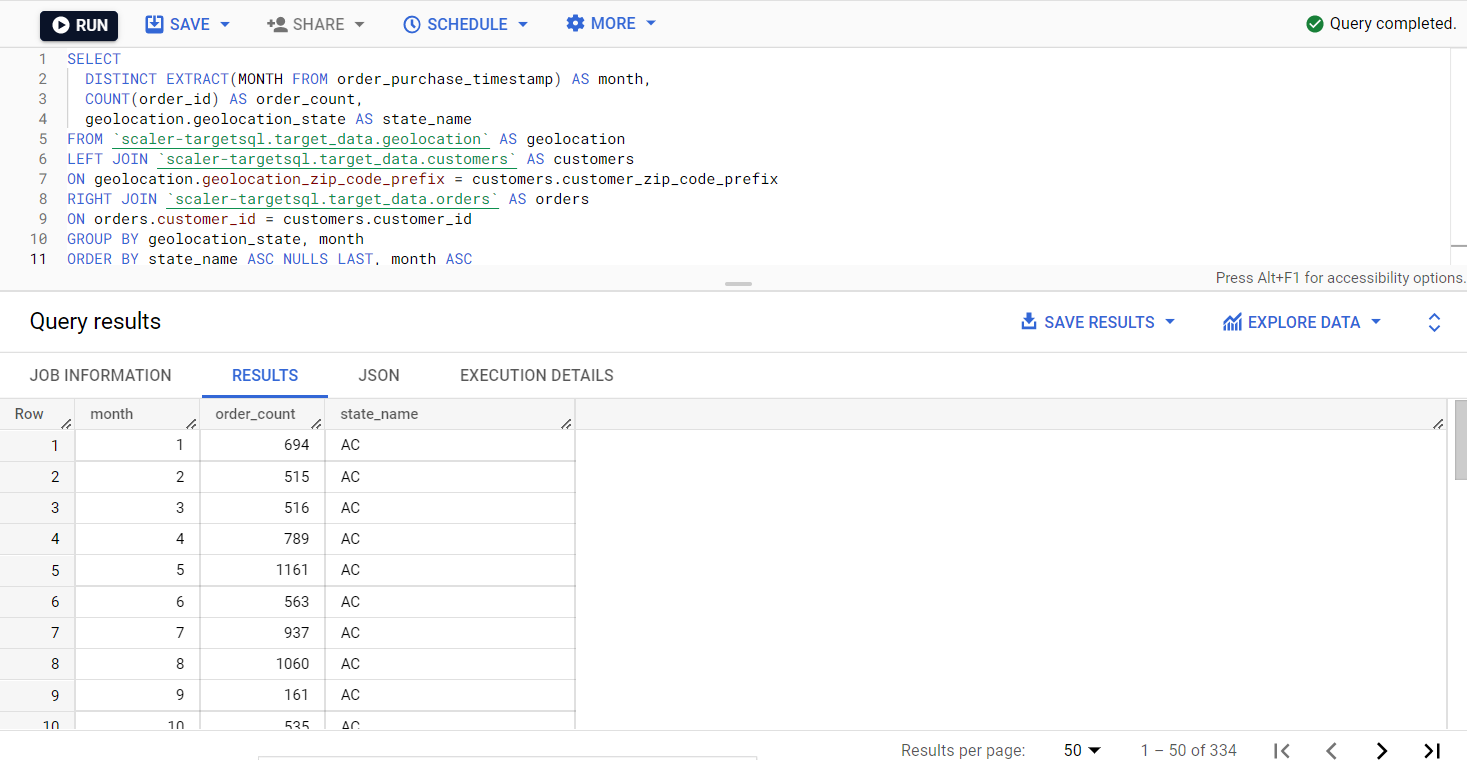
RIGHT JOIN `scaler-targetsql.target\_data.orders` AS orders

ON orders.customer\_id = customers.customer\_id

GROUP BY geolocation\_state, month

ORDER BY state\_name ASC NULLS LAST, month ASC

// This query gets total orders by month, to get total orders by city we can replace geolocation\_state in GROUP BY with geolocation\_city. There is nothing such as “regions” within the database, so we can’t get details by region



To get count of orders by a specific city/state:

SELECT DISTINCT EXTRACT(MONTH FROM order\_purchase\_timestamp) AS month,

COUNT(order\_id) AS order\_count,

geolocation.geolocation\_city AS city\_name,

geolocation.geolocation\_state AS state\_name

FROM `scaler-targetsql.target\_data.geolocation` AS geolocation

LEFT JOIN `scaler-targetsql.target\_data.customers` AS customers

ON geolocation.geolocation\_zip\_code\_prefix = customers.customer\_zip\_code\_prefix

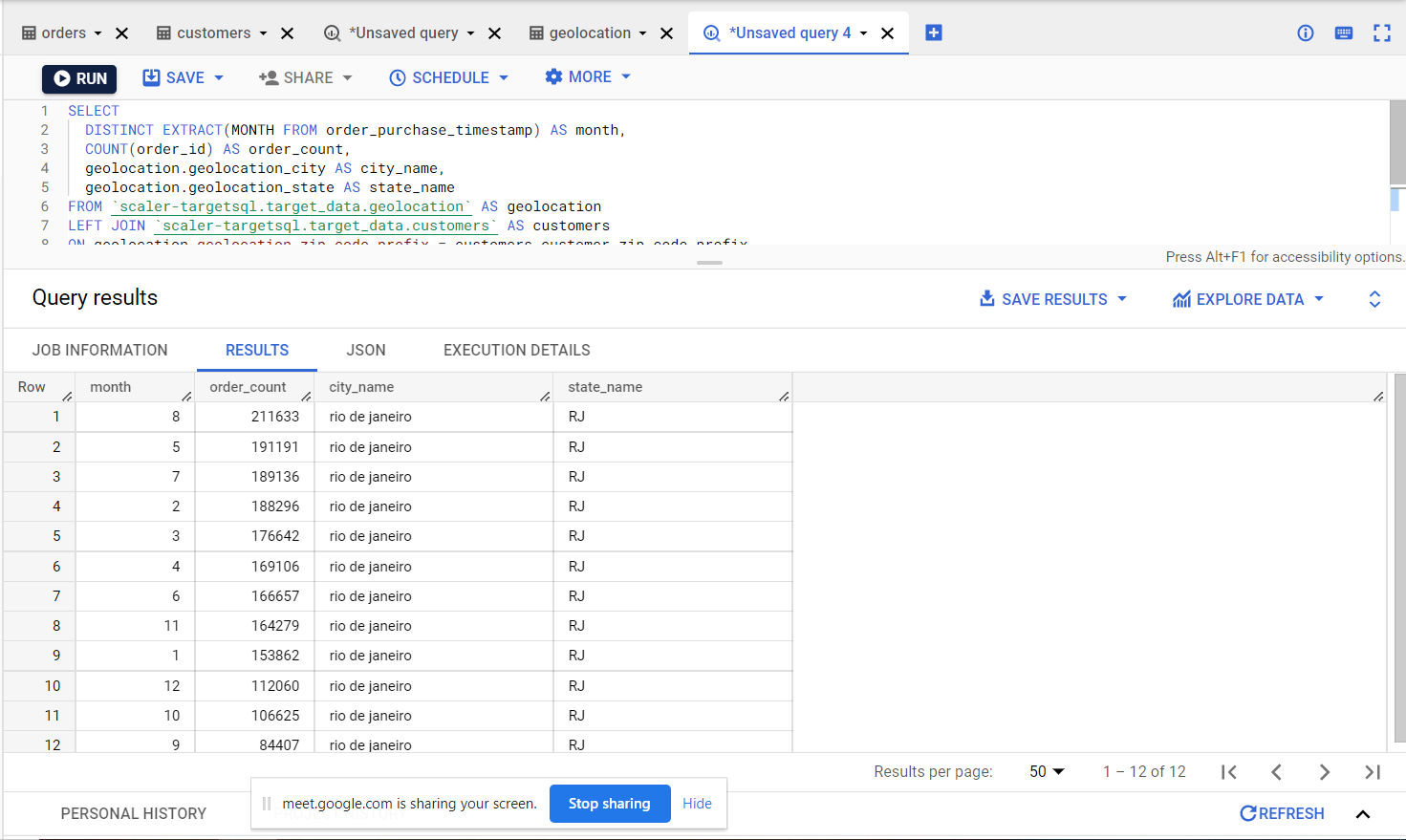
RIGHT JOIN `scaler-targetsql.target\_data.orders` AS orders

ON orders.customer\_id = customers.customer\_id

GROUP BY geolocation\_city, geolocation\_state, month

HAVING geolocation\_city = "rio de janeiro" AND geolocation\_state = "RJ"

ORDER BY order\_count DESC;



1. SELECT COUNT(order\_id) AS order\_count, geolocation.geolocation\_state AS state\_name

FROM `scaler-targetsql.target\_data.geolocation` AS geolocation

LEFT JOIN `scaler-targetsql.target\_data.customers` AS customers

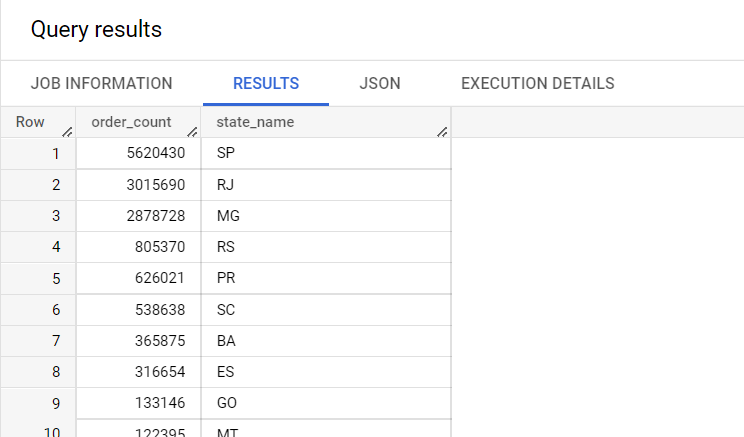
ON geolocation.geolocation\_zip\_code\_prefix = customers.customer\_zip\_code\_prefix

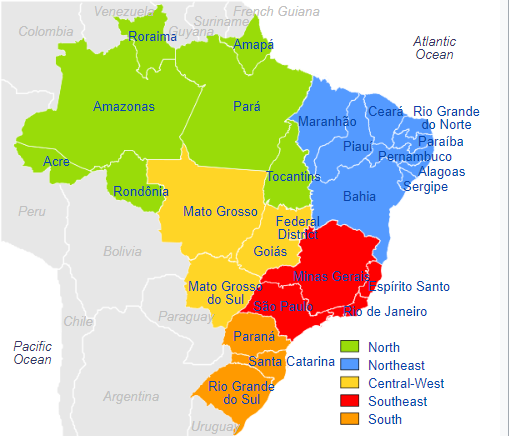
RIGHT JOIN `scaler-targetsql.target\_data.orders` AS orders

ON orders.customer\_id = customers.customer\_id

GROUP BY geolocation\_state

ORDER BY order\_count DESC



// As we can see here, most of the customers are distributed in the south-east and southern regions of Brazil. There is no “region” field in the database hence I have used an official map for this purpose  
  


Q.4) Impact on Economy: Analyze the money movemented 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)
  2. Mean & Sum of price and freight value by customer state

Ans:

1. WITH cte1 AS (

  SELECT

  SUM(price) AS total\_sum,

  EXTRACT(YEAR FROM order\_purchase\_timestamp) AS year

FROM `target\_data.order\_items` AS order\_items

  INNER JOIN `target\_data.orders` AS orders

  ON orders.order\_id = order\_items.order\_id

  WHERE EXTRACT(MONTH FROM order\_purchase\_timestamp) <= 8

  GROUP BY year

),

cte2 AS (

  SELECT \* FROM cte1 WHERE year = 2017

),

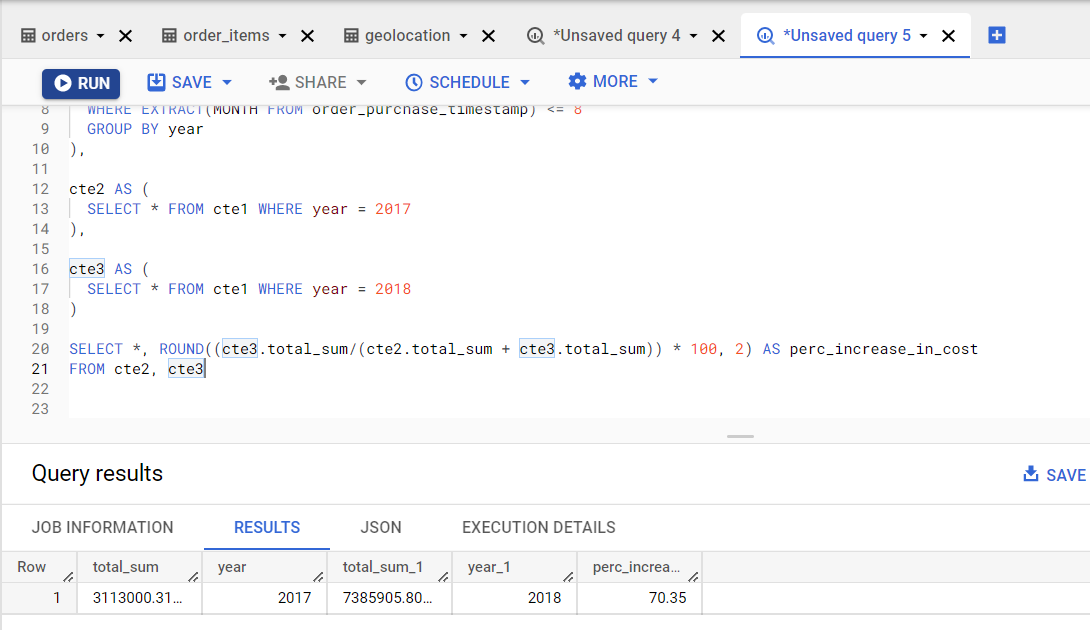
cte3 AS (

  SELECT \* FROM cte1 WHERE year = 2018

)

SELECT \*, ROUND((cte3.total\_sum/(cte2.total\_sum + cte3.total\_sum)) \* 100, 2) AS perc\_increase\_in\_cost

FROM cte2, cte3



1. SELECT

customer\_state AS state,

SUM(price) AS sum\_price,

AVG(price) AS avg\_price,

SUM(freight\_value) AS sum\_frieght,

AVG(freight\_value) AS avg\_frieght

FROM `target\_data.orders` AS orders

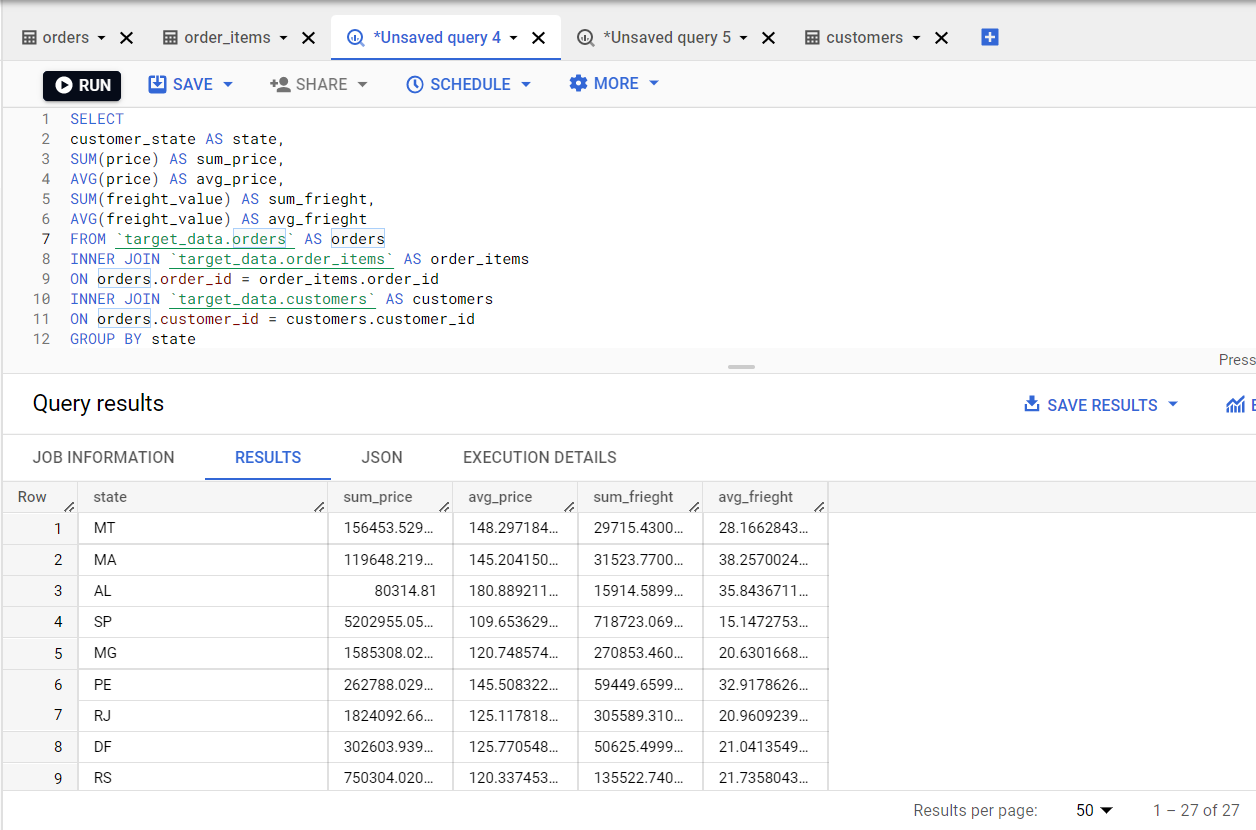
INNER JOIN `target\_data.order\_items` AS order\_items

ON orders.order\_id = order\_items.order\_id

INNER JOIN `target\_data.customers` AS customers

ON orders.customer\_id = customers.customer\_id

GROUP BY state



Q.5) Analysis on sales, freight and delivery time

1. Calculate days between purchasing, delivering and estimated delivery. Create columns:
   1. time\_to\_delivery = order\_purchase\_timestamp-order\_delivered\_customer\_date
   2. diff\_estimated\_delivery = order\_estimated\_delivery\_date-order\_delivered\_customer\_date
2. Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery
3. Sort the data to get the following:
   1. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5
   2. Top 5 states with highest/lowest average time to delivery
   3. Top 5 states where delivery is really fast/ not so fast compared to estimated date

Ans:

1. // Before, diving all data into columns, I noticed that order\_delivered customer date as some NULL values, so we will check them first and remove them in main query

NULL value check:

SELECT COUNT(\*), 'null\_tally' AS narrative

FROM `target\_data.orders` AS orders

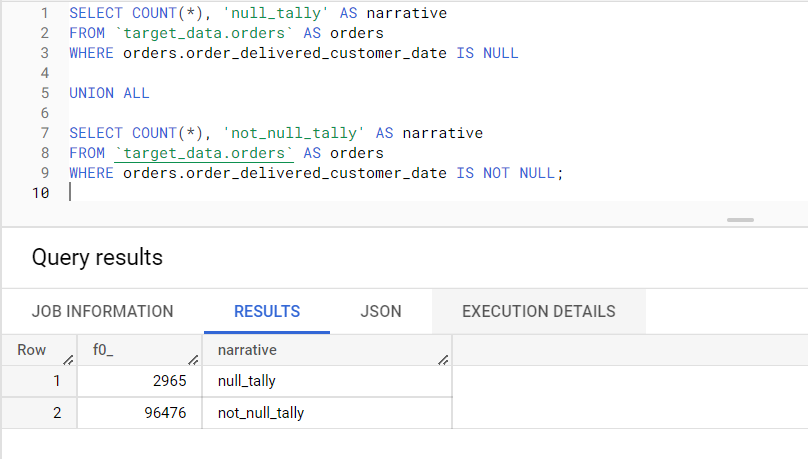
WHERE orders.order\_delivered\_customer\_date IS NULL

UNION ALL

SELECT COUNT(\*), 'not\_null\_tally' AS narrative

FROM `target\_data.orders` AS orders

WHERE orders.order\_delivered\_customer\_date IS NOT NULL;



Main Query:

SELECT

  EXTRACT(DATE FROM order\_purchase\_timestamp) AS purchasing\_time,

  EXTRACT(DATE FROM order\_delivered\_customer\_date) AS actual\_delivery\_date,

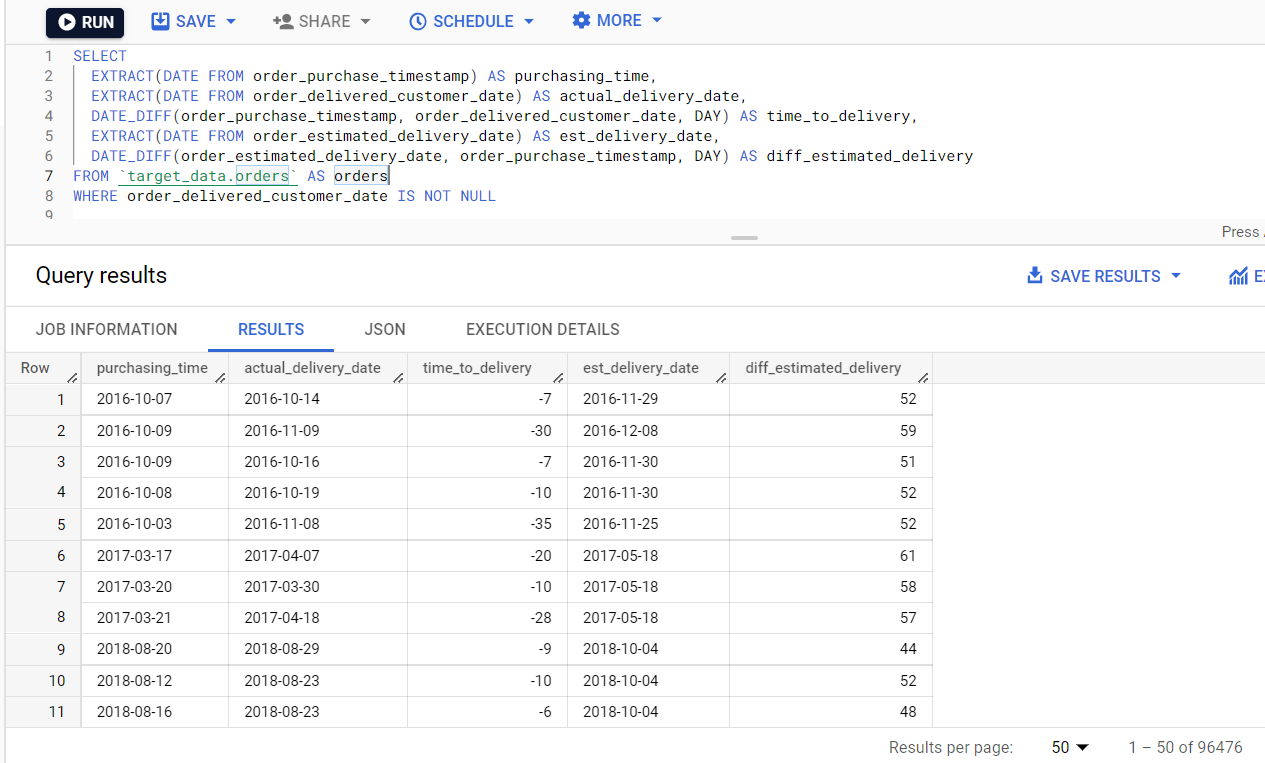
  DATE\_DIFF(order\_purchase\_timestamp, order\_delivered\_customer\_date, DAY) AS time\_to\_delivery,

  EXTRACT(DATE FROM order\_estimated\_delivery\_date) AS est\_delivery\_date,

  DATE\_DIFF(order\_estimated\_delivery\_date, order\_purchase\_timestamp, DAY) AS diff\_estimated\_delivery

FROM `target\_data.orders` AS orders

WHERE order\_delivered\_customer\_date IS NOT NULL



1. SELECT

  AVG(freight\_value) AS avg\_frieght,

  AVG(DATE\_DIFF(order\_purchase\_timestamp, order\_delivered\_customer\_date, DAY)) AS avg\_time\_to\_delivery,

  AVG(DATE\_DIFF(order\_estimated\_delivery\_date, order\_purchase\_timestamp, DAY)) AS avg\_diff\_estimated\_delivery,

  customer\_state AS state

FROM `target\_data.orders` AS orders

INNER JOIN `target\_data.order\_items` AS order\_items

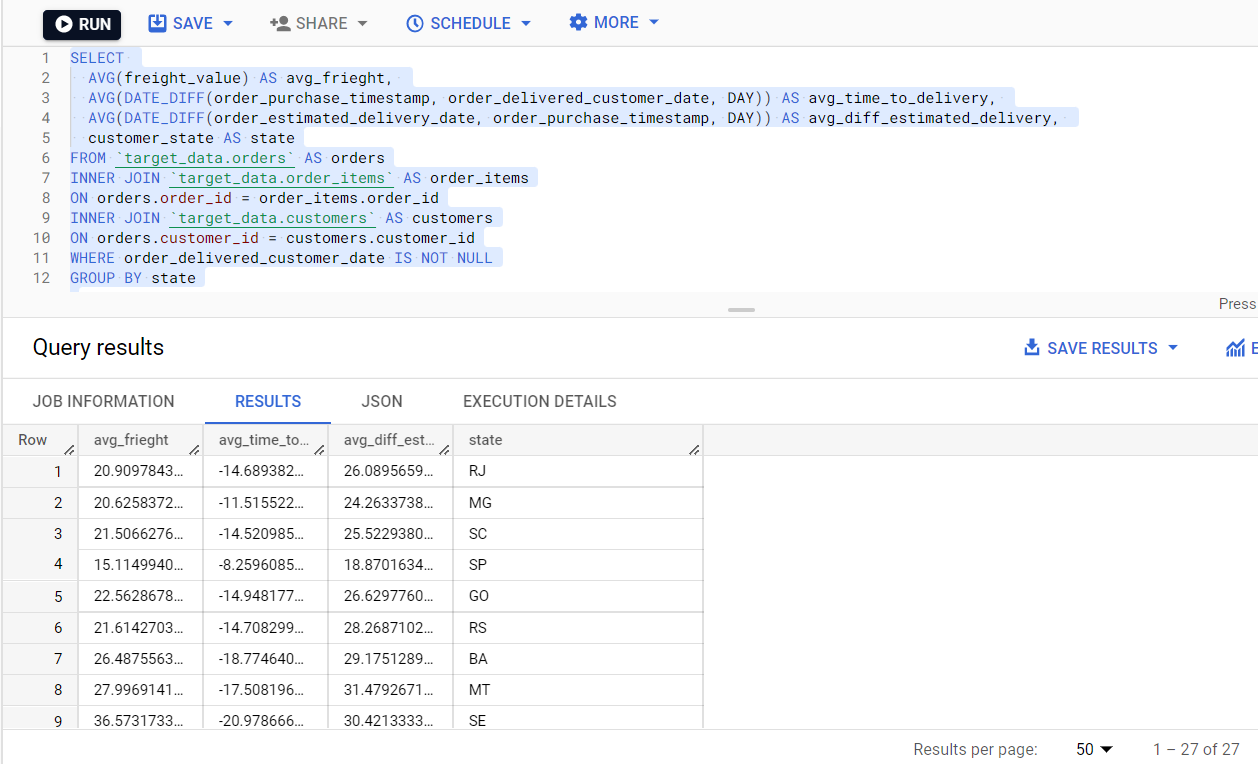
ON orders.order\_id = order\_items.order\_id

INNER JOIN `target\_data.customers` AS customers

ON orders.customer\_id = customers.customer\_id

WHERE order\_delivered\_customer\_date IS NOT NULL

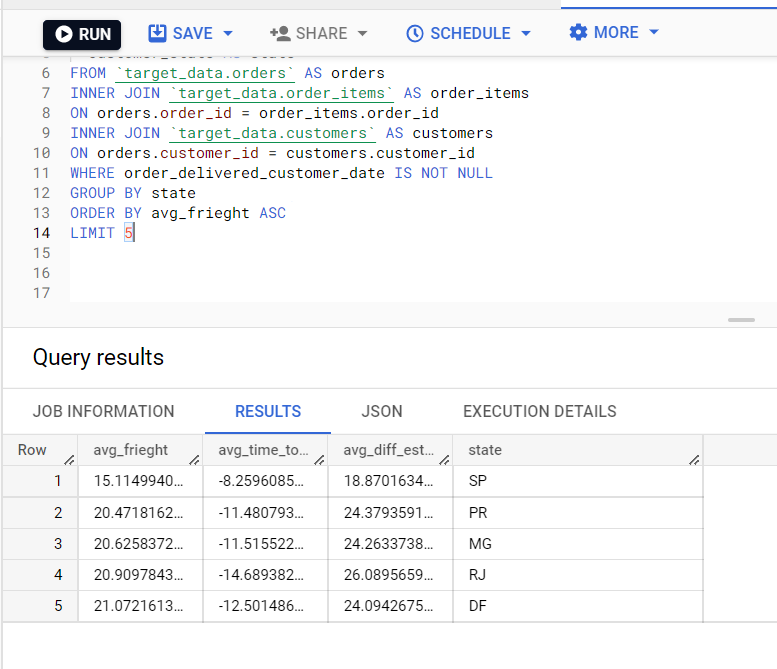
GROUP BY state



1. Top 5 avg\_frieght ASC :

ORDER BY avg\_frieght ASC

LIMIT 5

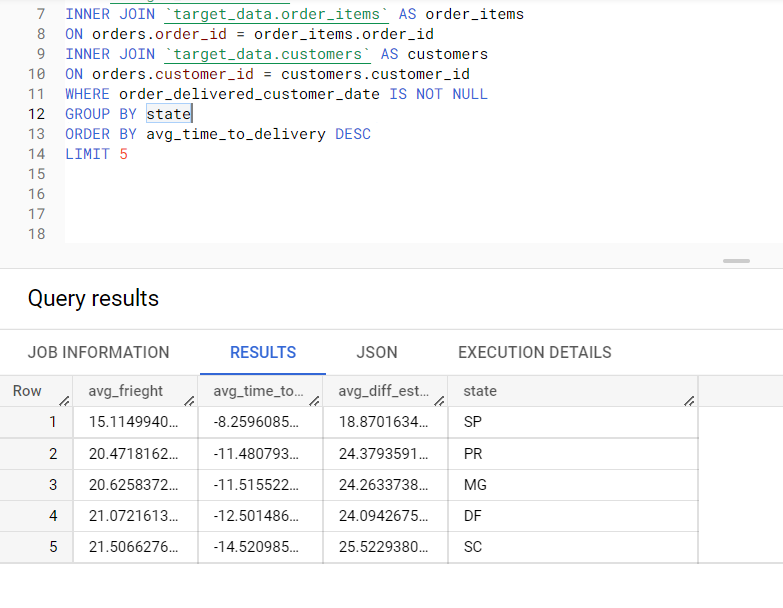


Top 5 avg\_delivery\_time ASC :

ORDER BY avg\_time\_to\_delivery DESC

LIMIT 5

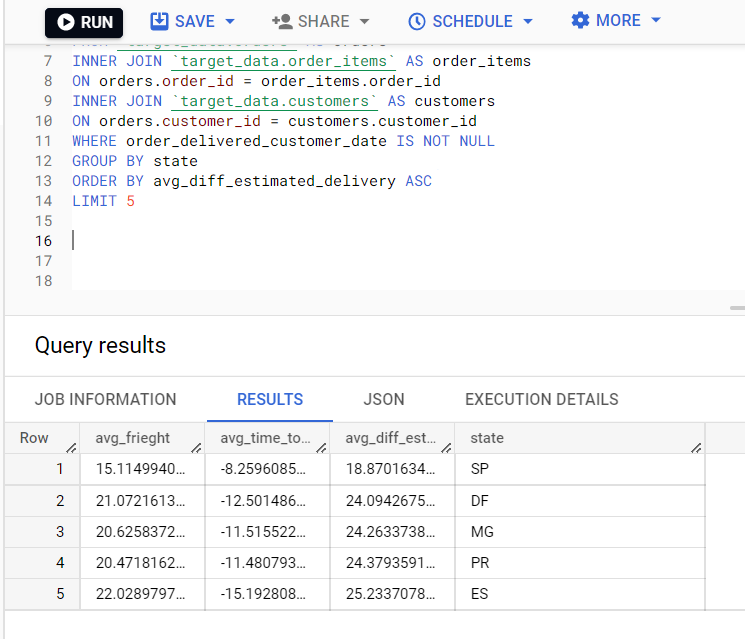
// Descending order because values are negative, higher negative value (more towards 0) = less delivery time



Top 5 states where Avg. Difference between expected and actual time is least:

ORDER BY avg\_diff\_estimated\_delivery ASC

LIMIT 5



Q.6) Payment type analysis:

1. Month over Month count of orders for different payment types
2. Distribution of payment instalments and count of orders

Ans:

1. SELECT

   EXTRACT(MONTH FROM order\_purchase\_timestamp) AS month,

   COUNT(payments.order\_id) AS cnt\_orders,

   payment\_type

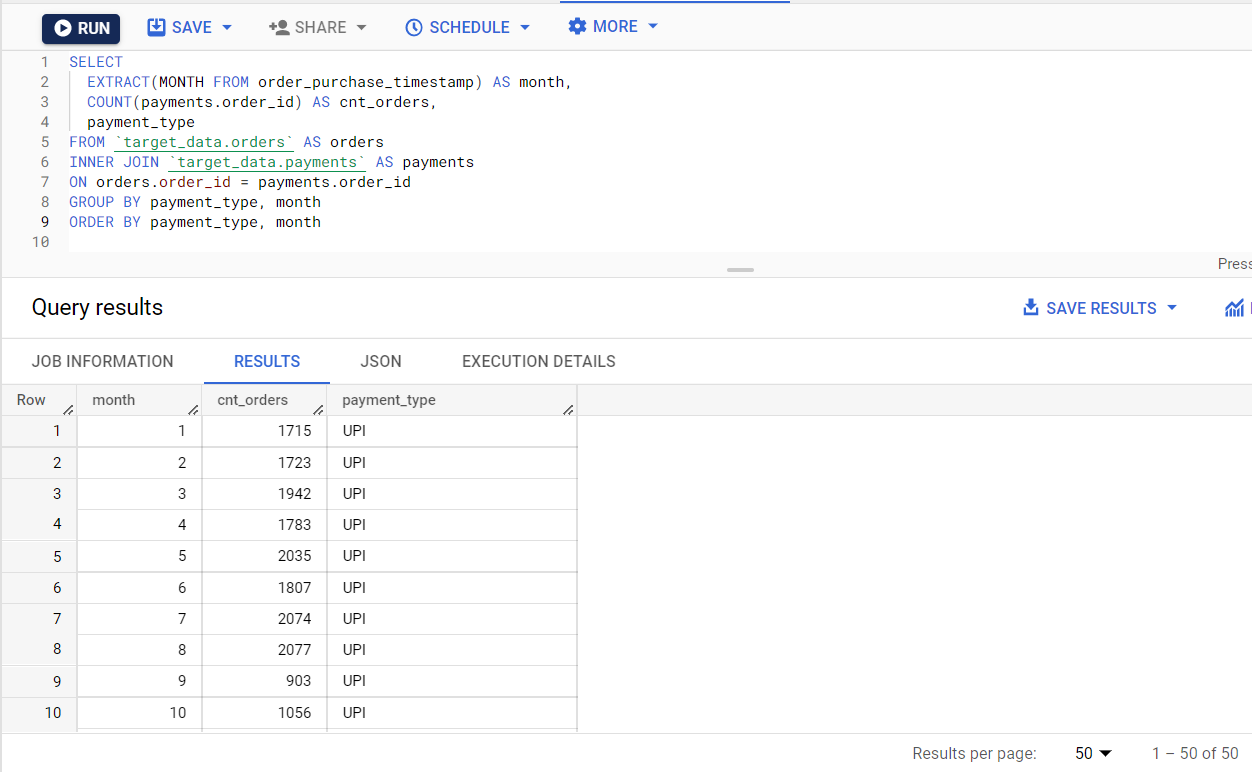
FROM `target\_data.orders` AS orders

INNER JOIN `target\_data.payments` AS payments

ON orders.order\_id = payments.order\_id

GROUP BY payment\_type, month

ORDER BY payment\_type, month



1. SELECT

   COUNT(payments.order\_id) AS cnt\_orders,

   payment\_installments

FROM `target\_data.payments` AS payments

GROUP BY payment\_installments

ORDER BY cnt\_orders DESC

