**Target - Business Case SOLUTION**

**Q - What does ‘good’ look like?**

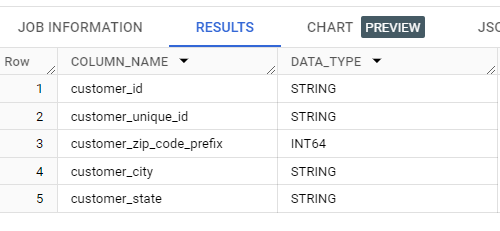
1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**
   1. **Data type of all columns in the “customers” table.**

**ANS A**- SELECT COLUMN\_NAME, DATA\_TYPE

FROM target-406519.Target.INFORMATION\_SCHEMA.COLUMNS

WHERE table\_name = 'customers'

**SCREENSHOT**



**INSIGHTS** - As per result, it is concluded that most of data are store in the form of string (varchar) except customer \_zip\_code\_prefix**.**

* 1. **Get the time range between which the orders were placed.**

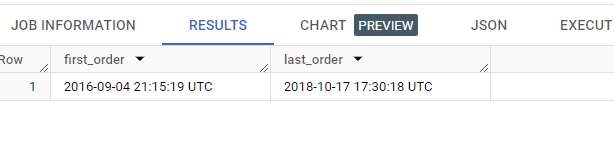
**ANS B**- SELECT

min(order\_purchase\_timestamp) as first\_order,

max(order\_purchase\_timestamp) as last\_order

FROM `Target.orders`

**Screenshot-**



**INSIGHTS** – This help to understand product timeline in market. This help in time range of product and it popularity, order pattern.

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

**UNIQUE VALUE - USE KEYWORD DISTINCT**

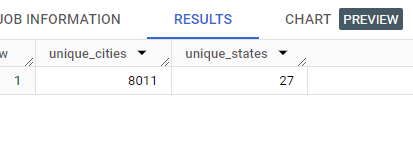
ANS C - SELECT

COUNT(DISTINCT geolocation\_city) AS unique cities,

COUNT(DISTINCT geolocation\_state) AS unique tates

FROM `Target.geolocation`

**Screenshot-**



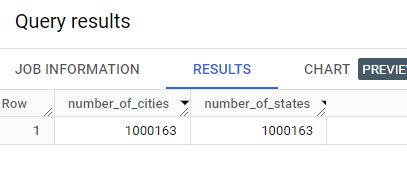
**ANS - repeated orders from CITIES and STATES**

SELECT

COUNT( geolocation\_city) AS number\_of\_cities,

COUNT( geolocation\_state) AS number\_of\_states

FROM `Target.geolocation`

****

**INSIGHTS** – This help in knowing the product popularity among the no of cities and in states. This two results gave us the unique value and repeated orders of the products.

1. **In-depth Exploration:**
   1. Is there a growing trend in the no. of orders placed over the past years?

ANS -

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

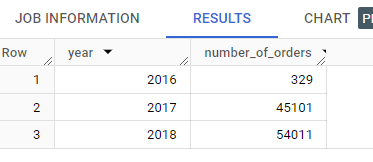
COUNT(\*) AS number\_of\_orders

FROM `Target.orders`

GROUP BY 1

ORDER BY 1;

Screenshot



**Insights** – In the first year of product launch, there was a few sales and growth jumps to five times. We observe the pattern increase over a span of year.

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

ANS B -

WITH cte AS

(

SELECT

\*,

EXTRACT ( DATE FROM order\_purchase\_timestamp) AS order\_date,

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

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

FROM `target-406519.Target.orders`

)

SELECT

 order\_month,

 order\_year,

\_id) AS total\_orders

FROM cte

GROUP BY

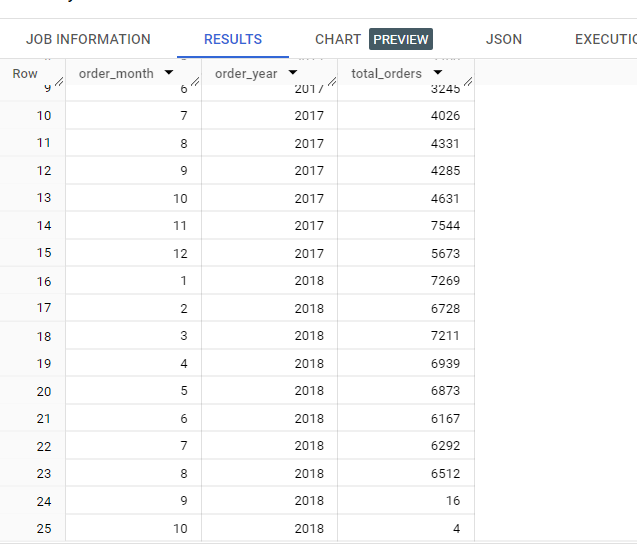
  order\_month,

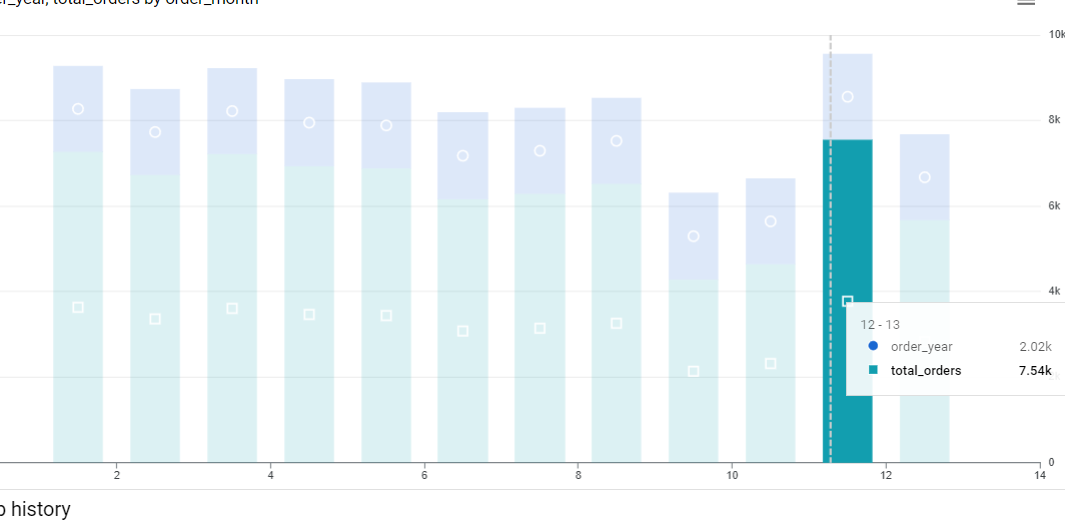
 order\_year

ORDER BY

 order\_year,

 order\_month





**Insights** – Seasonal effect were observe sales were increase on black Friday and New Year in Jan month. Understanding the sales behaviour and consumer activity help business to boost revenue on particular season by managing inventory of the products and also in planning discount ahead of time.

* 1. 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 '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'

  END AS order\_time\_interval,

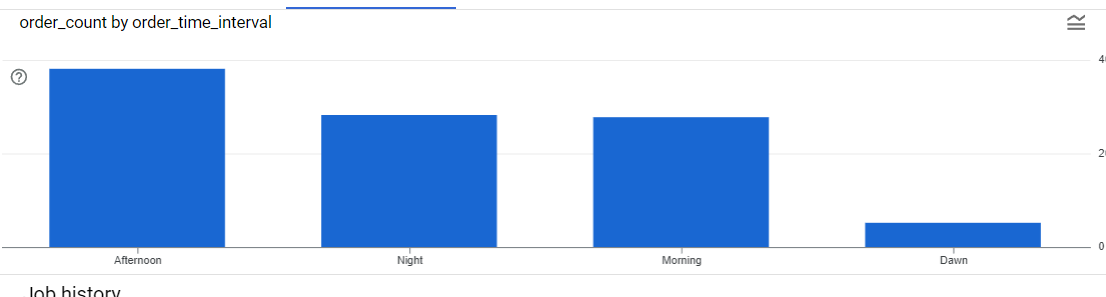
  COUNT(\*) AS order\_count

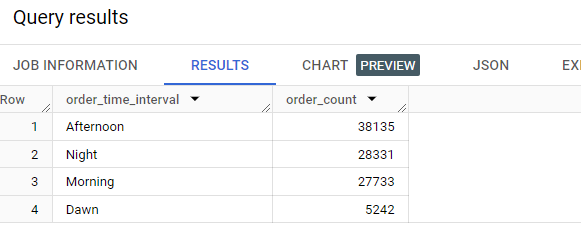
FROM `target-406519.Target.orders`

GROUP BY order\_time\_interval

ORDER BY order\_count DESC;

Screenshot





**Insights** – This brings that most of customer were active all time except Dawn time. This help the business to understand when will be product will in high demand.

1. **Evolution of E-commerce orders in the Brazil region:** 
   1. Get the month on month no. of orders placed in each state.

Solution -SELECT EXTRACT(MONTH FROM o.order\_purchase\_timestamp) AS

order\_month, c.customer\_state, COUNT(\*) AS number\_of\_order

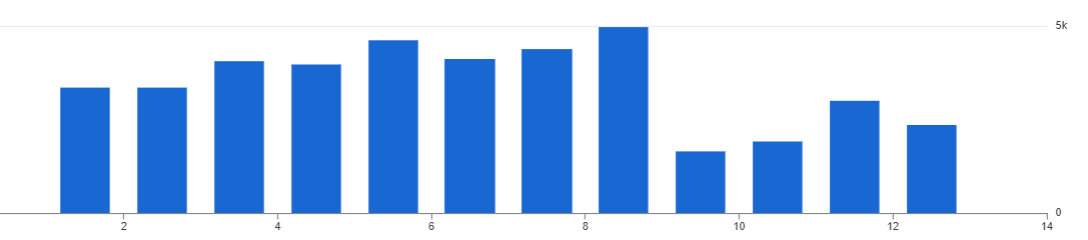
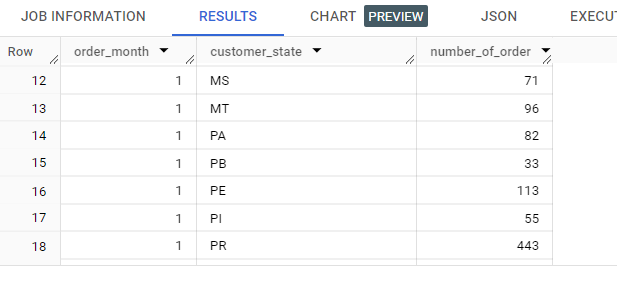
FROM `target-406519.Target.orders` AS o

JOIN `target-406519.Target.customers` AS c

GROUP BY order\_month, c.customer\_state

ORDER BY order\_month, c.customer\_state

ON o.customer\_id = c.customer\_id



**Insights** – This help us to target the potential products on particular month to bring most revenue and also inform marking team to market to product in that area.

* 1. **How are the customers distributed across all the states?**

SELECT customer\_state,

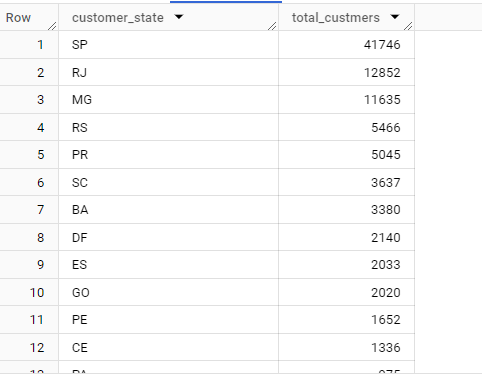
COUNT(DISTINCT customer\_id) AS total\_custmers

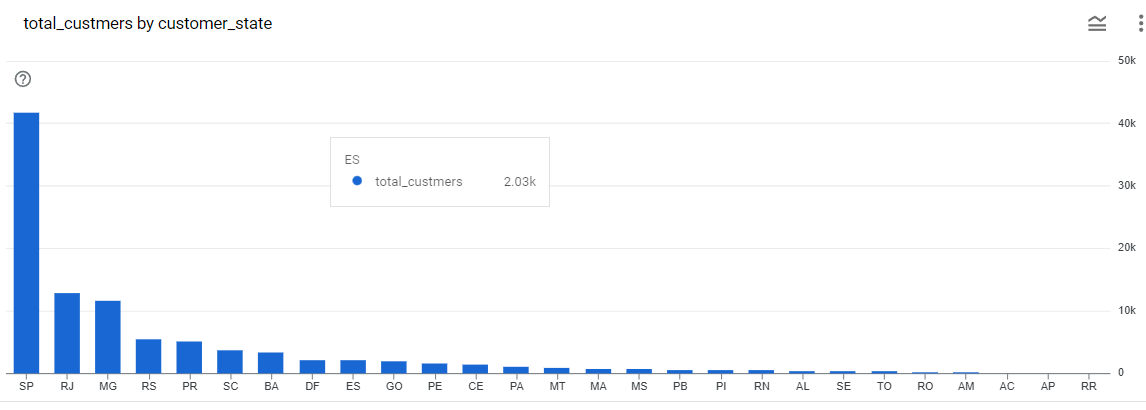
FROM `target-406519.Target.customers`

GROUP BY 1

ORDER BY 1 DESC

SCREENSHOTS





**Insights** – This show that the most wanted state having the highest demand of product. This helps production team to evaluate how much product they need in particular states.

1. **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

ROUND((((total\_payment\_2018 - total\_payment\_2017) /

total\_payment\_2017) \* 100), 2) AS percentage\_increase

FROM (

 SELECT

 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

 ELSE 0

 END) AS total\_payment\_2017,

 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

 ELSE 0

 END) AS total\_payment\_2018

 FROM

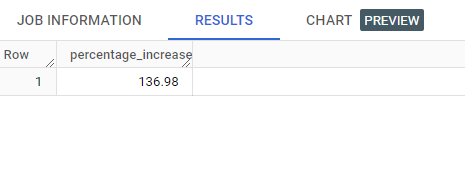
 `target-406519.Target.payments` AS p

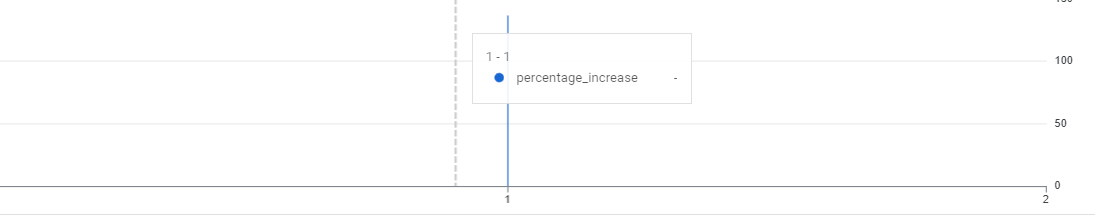
 JOIN`target-406519.Target.orders` AS o

 ON p.order\_id = o.order\_id

 );

**Screenshot**





**Insights** - This query show the amount of increase of sales between orders placed from January to August are considered.

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

SELECT customer\_state,

  ROUND(SUM(p.payment\_value),2) AS total\_order\_price,

  ROUND(AVG(p.payment\_value),2) AS average\_order\_price

FROM `target-406519.Target.payments` AS p

JOIN `target-406519.Target.orders` AS o

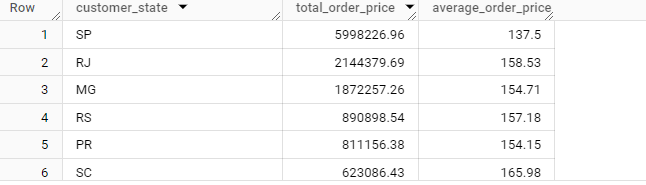
ON p.order\_id = o.order\_id

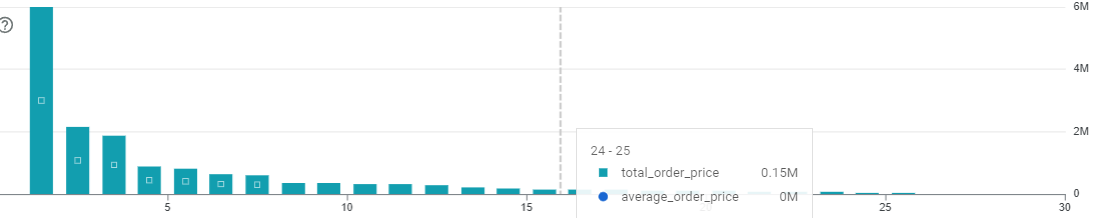
JOIN `target-406519.Target.customers` AS c

ON o.customer\_id = c.customer\_id

GROUP BY 1

ORDER BY 2 DESC





**Insights –** This query brought analyst attention on particular total\_order price among states and average no of orders.

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

SELECT

customer\_state AS state,

 ROUND(SUM(oi.freight\_value),2) AS total\_freight,

ROUND(AVG(oi.freight\_value),2) AS average\_freight

FROM `target-406519.Target.orders` AS o

JOIN `target-406519.Target.order\_items` AS oi

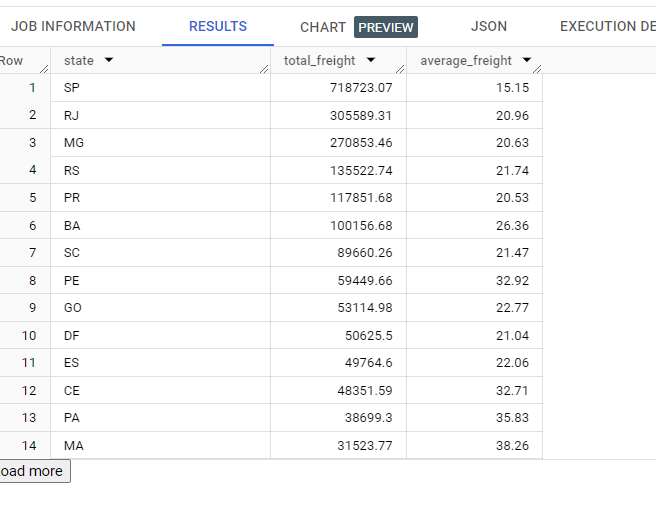
ON o.order\_id = oi.order\_id

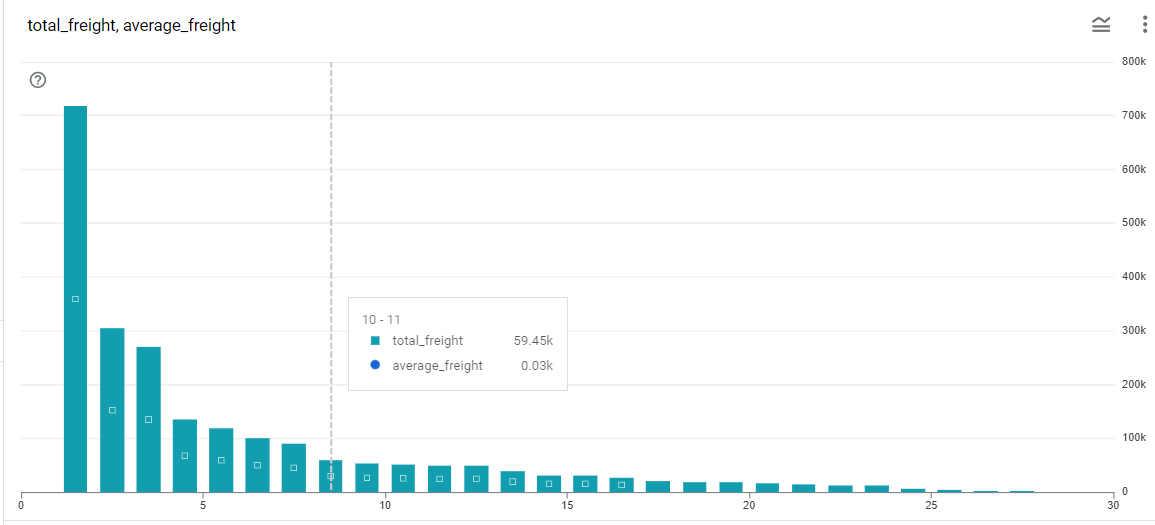
JOIN `target-406519.Target.customers` AS c

ON c.customer\_id = o.customer\_id

GROUP BY 1

ORDER BY 2 DESC





**Insights**

1. **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.

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

SELECT

order\_id,

DATE\_DIFF(DATE(order\_delivered\_customer\_date),

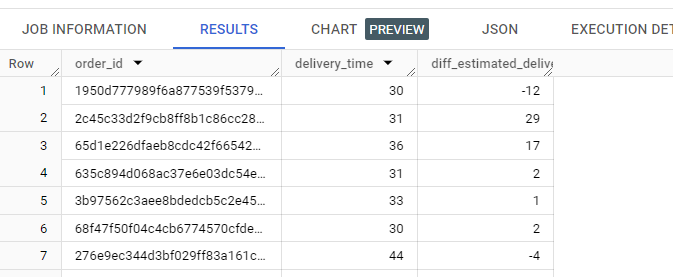
DATE(order\_purchase\_timestamp), DAY) AS delivery\_time,

DATE\_DIFF(DATE(order\_estimated\_delivery\_date),

DATE(order\_delivered\_customer\_date), DAY) AS

diff\_estimated\_delivery

FROM `target-406519.Target.orders`



**Insights -**

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

SELECT

 high.customer\_state AS high\_state,

high.average\_freight\_value AS high\_avg\_freight,

low.customer\_state AS low\_state,

 low.average\_freight\_value AS low\_avg\_freight

FROM

(

SELECT

c.customer\_state,

 ROUND(AVG(p.freight\_value),2) AS average\_freight\_value,

ROW\_NUMBER() OVER(ORDER BY

(ROUND(AVG(p.freight\_value),2))DESC) AS rowval1

 FROM `target-406519.Target.orders`   AS o

JOIN `target-406519.Target.order\_items` AS p

ON o.order\_id = p.order\_id

JOIN `target-406519.Target.customers`  AS c

 ON o.customer\_id = c.customer\_id

 GROUP BY

c.customer\_state

ORDER BY

 average\_freight\_value DESC

 LIMIT

 5

) AS high

JOIN

(

SELECT

c.customer\_state,

ROUND(AVG(p.freight\_value),2) AS average\_freight\_value,

ROW\_NUMBER() OVER(ORDER BY (ROUND(AVG(p.freight\_value),2)))

AS rowval2

FROM `target-406519.Target.orders`   AS o

JOIN `target-406519.Target.order\_items` AS p

 ON o.order\_id = p.order\_id

 JOIN `target-406519.Target.customers`   AS c

ON o.customer\_id = c.customer\_id

 GROUP BY

 c.customer\_state

 ORDER BY

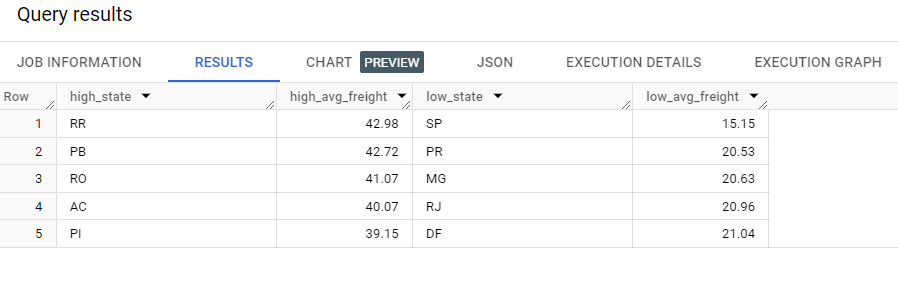
 average\_freight\_value

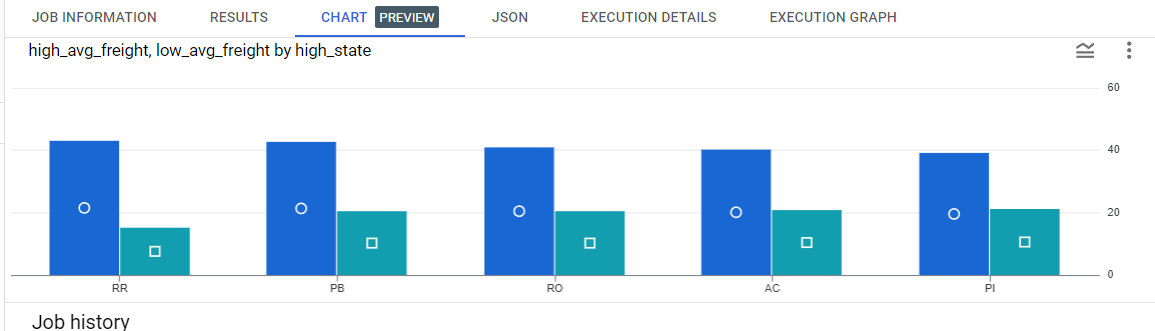
LIMIT

 5

) AS low

ON high.rowval1 = low.rowval2;





**Insights -**

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

WITH cte AS

(

SELECT

 c.customer\_state,

 ROUND(AVG(t1.delivery\_time),2) AS avg\_delivery\_time

FROM

 (

 SELECT

 \*,

TIMESTAMP\_DIFF(order\_delivered\_customer\_date,

order\_purchase\_timestamp, day) AS delivery\_time,

FROM

`target-406519.Target.orders`

 WHERE

 order\_status = 'delivered' AND

order\_delivered\_customer\_date IS NOT NULL

 ORDER BY

 order\_purchase\_timestamp

) AS t1

JOIN

`target-406519.Target.customers` AS c

ON t1.customer\_id = c.customer\_id

GROUP BY

 c.customer\_state

ORDER BY

 avg\_delivery\_time

)

SELECT

c1.customer\_state AS low\_state,

c1.avg\_delivery\_time AS low\_avg\_delivery\_time,

c2.customer\_state AS high\_state,

c2.avg\_delivery\_time AS high\_avg\_delivery\_time

FROM

(

 SELECT

\*,

ROW\_NUMBER() OVER (ORDER BY cte.avg\_delivery\_time DESC) AS

rowval2

 FROM

 cte

ORDER BY

rowval2

) AS c2

JOIN

(

 SELECT

\*,

ROW\_NUMBER() OVER (ORDER BY cte.avg\_delivery\_time) AS

rowval1

 FROM

cte

 ORDER BY

rowval1

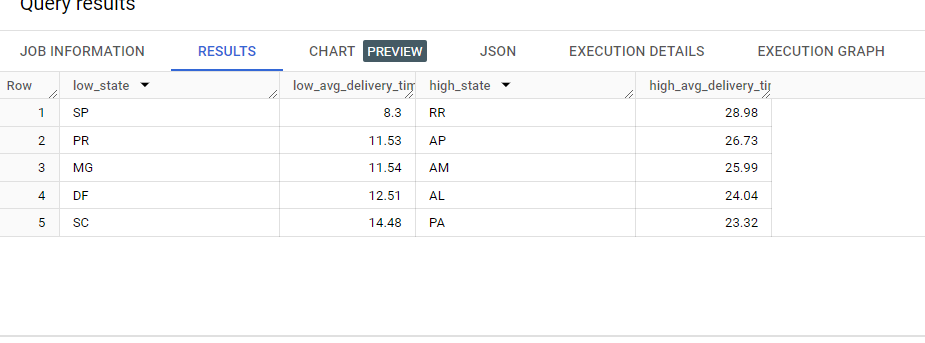
) AS c1

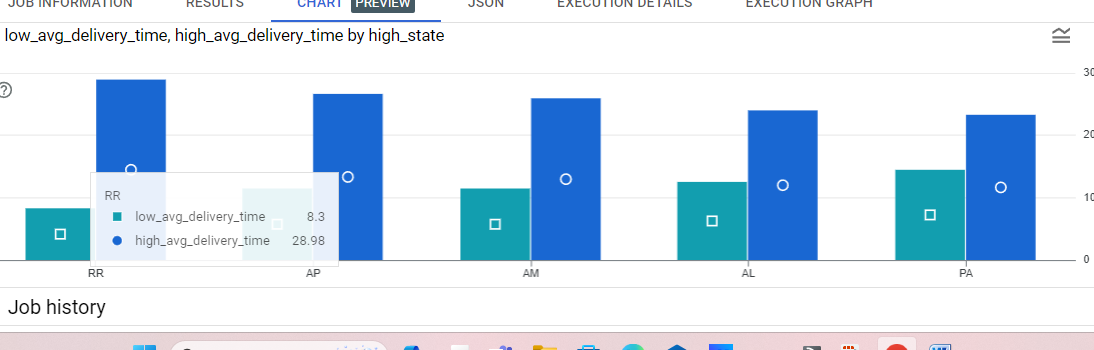
ON

c1.rowval1 = c2.rowval2

LIMIT

5;





**Insights -**

* 1. 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.

 WITH delivery\_speed AS (

 SELECT

 c.customer\_state,

 AVG(DATE\_DIFF(o.order\_delivered\_customer\_date,

o.order\_estimated\_delivery\_date, DAY)) AS avg\_delivery\_speed,

 ROW\_NUMBER() OVER (ORDER BY

AVG(DATE\_DIFF(o.order\_delivered\_customer\_date,

o.order\_estimated\_delivery\_date, DAY))) AS rank\_fastest

 FROM `target-406519.Target.orders`           AS o

 JOIN `target-406519.Target.customers` AS c

 ON o.customer\_id = c.customer\_id

 WHERE o.order\_delivered\_customer\_date IS NOT NULL AND

o.order\_estimated\_delivery\_date IS NOT NULL

 GROUP BY c.customer\_state

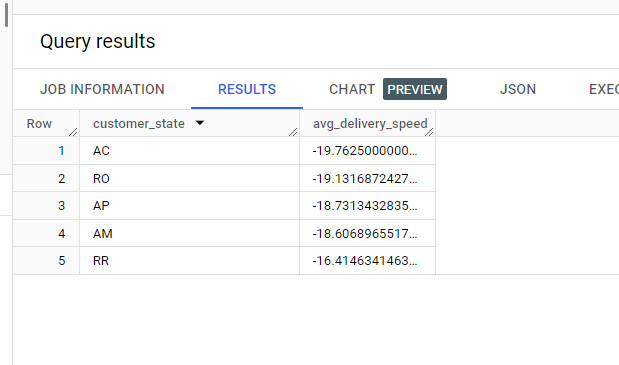
)

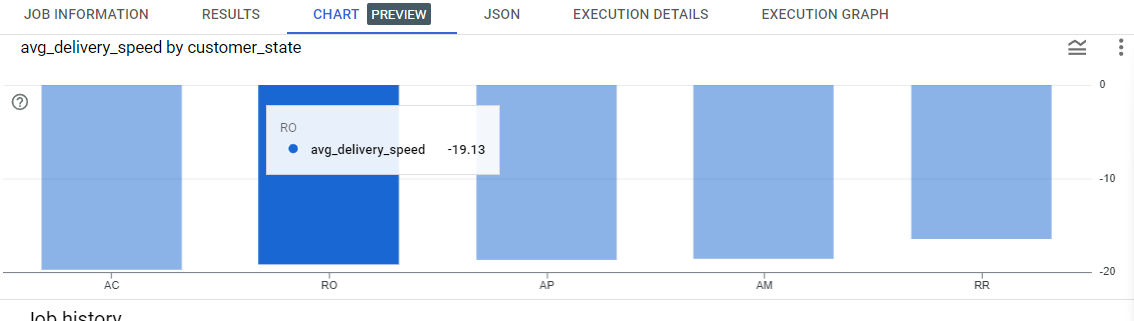
SELECT customer\_state, avg\_delivery\_speed

FROM delivery\_speed

WHERE rank\_fastest <= 5

ORDER BY avg\_delivery\_speed;





**Insights -**

1. **Analysis based on the payments:**
   1. Find the month on month no. of orders placed using different payment types**.**

SELECT

 FORMAT\_TIMESTAMP('%Y-%m', o.order\_purchase\_timestamp) AS

month,

 p.payment\_type,

COUNT(DISTINCT o.order\_id) AS order\_count

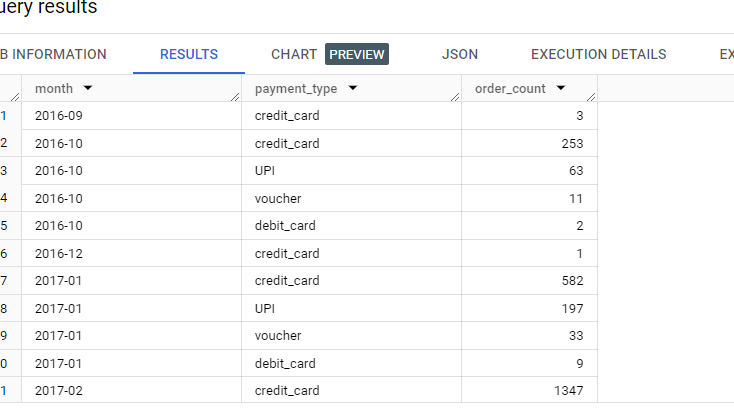
FROM `target-406519.Target.orders` AS o

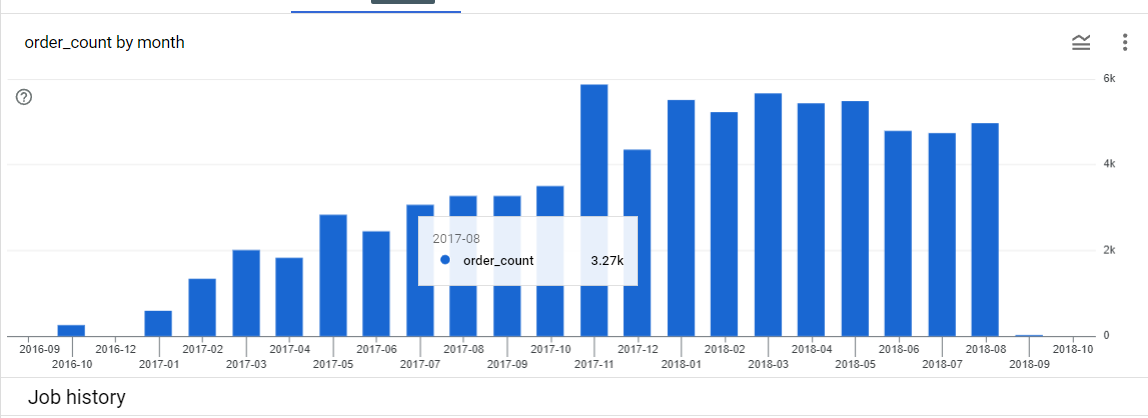
JOIN `target-406519.Target.payments` AS p

ON o.order\_id = p.order\_id

GROUP BY 1, 2

ORDER BY 1





**Insights -**

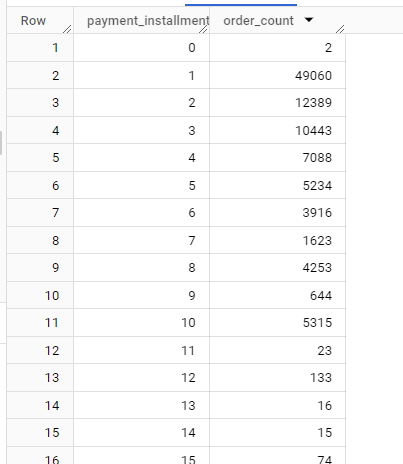
* 1. **Find the no. of orders placed on the basis of the payment installments that have been paid.**

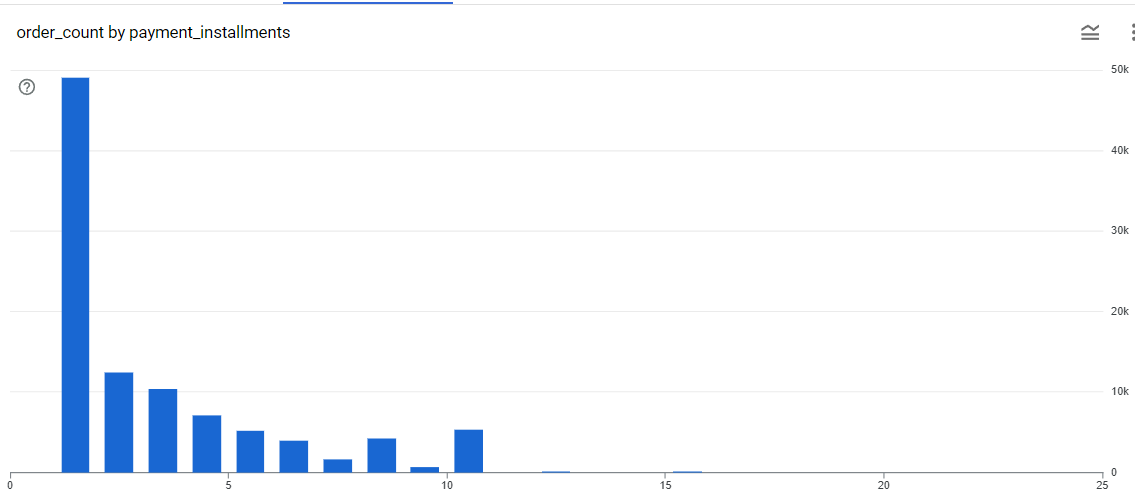
SELECT payment\_installments, COUNT(DISTINCT order\_id) AS order\_count

FROM `target-406519.Target.payments`

GROUP BY 1

ORDER BY 1





**Insights -**