**Target Business Case Study**

**🛒 Target Brazil E-Commerce Case Study**

**📌 Project Overview**

Analyzed 100,000+ e-commerce orders from Target Brazil (2016–2018) to uncover operational insights across order processing, pricing, payments, freight, customer behavior, and delivery performance.

**🎯 Objectivess**

As a data analyst at Target, the goal was to extract actionable insights from a multi-table dataset and recommend improvements in logistics, pricing, and customer experience.

**📂 Dataset Summary**

* Source: [Google Drive Folder](https://drive.google.com/drive/folders/1TGEc66YKbD443nslRi1bWgVd238gJCnb)
* Format: 8 CSV files
* Tables: orders, order\_items, customers, sellers, payments, reviews, products, geolocation

**🔍 Key Explorations & Insights**

**🧭 Exploratory Analysis**

* Identified data types and structure across all tables
* Mapped customer distribution by city and state
* Extracted order time range and seasonal patterns

**📈 Order Trends**

* Analyzed yearly and monthly growth in order volume
* Segmented order times into Dawn, Morning, Afternoon, Night
* Tracked monthly order counts per state

**💰 Economic Impact**

* Calculated % increase in payment value from Jan–Aug (2017 vs 2018)
* Aggregated total & average order price and freight by state

**🚚 Delivery Performance**

* Computed delivery time and deviation from estimated delivery
* Ranked states by:
* Highest/lowest average freight cost
* Fastest/slowest delivery time
* Most efficient delivery vs estimated date

**💳 Payment Analysis**

* Monthly breakdown of payment types used
* Distribution of orders by installment count

**🛠️ Tools & Skills Used**

* Python (pandas, NumPy, Seaborn)
* Data cleaning & transformation
* Time-series analysis
* Geo-mapping & state-level aggregation
* Business storytelling with data

--------------------------------------------------------------

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

**1. Data type of all columns in the "customers" table.**

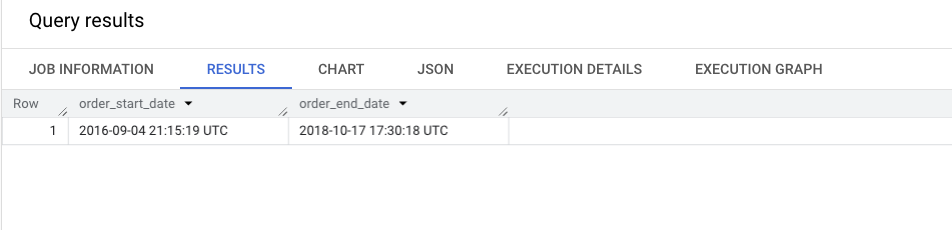


**2. Get the time range between which the orders were placed.**

SELECT MIN(order\_purchase\_timestamp) AS order\_start\_date,

MAX(order\_purchase\_timestamp) AS order\_end\_date,

FROM `TARGET\_SQL.orders`



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

WITH min\_max\_dates AS (

SELECT

MIN(order\_purchase\_timestamp) AS min\_date,

MAX(order\_purchase\_timestamp) AS max\_date

FROM `TARGET\_SQL.orders`

)

SELECT COUNT(DISTINCT cust.customer\_id) AS cities\_state\_count

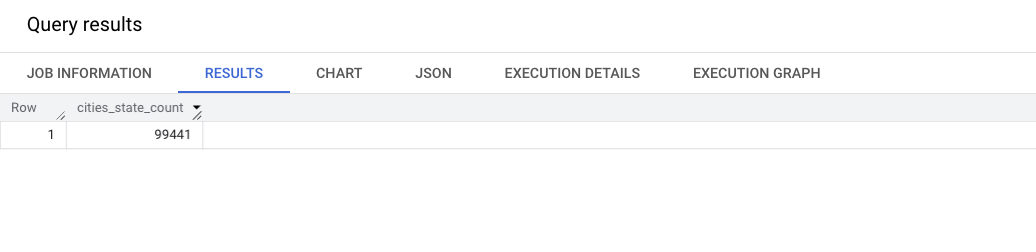
FROM `TARGET\_SQL.customers` AS cust

JOIN `TARGET\_SQL.orders` AS ord

ON cust.customer\_id = ord.customer\_id

JOIN min\_max\_dates AS dates

ON ord.order\_purchase\_timestamp BETWEEN dates.min\_date AND dates.max\_date;



**----------------------------------------------------------------------------------------------------------**

**In-depth Exploration:**

**1. Is there a growing trend in the no. of orders placed over the past years?**

WITH order\_data AS (

SELECT

order\_id,

order\_purchase\_timestamp,

FORMAT\_DATETIME('%Y-%m', order\_purchase\_timestamp) AS past\_years

FROM `TARGET\_SQL.orders`

)

SELECT

past\_years,

COUNT(\*) AS order\_month\_count,

COUNT(\*)-LAG(COUNT(\*)) OVER(ORDER BY past\_years) AS growth\_trend

FROM order\_data

GROUP BY past\_years

ORDER BY past\_years

LIMIT 10



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

WITH order\_data AS (

SELECT

FORMAT\_DATETIME('%m', order\_purchase\_timestamp) AS past\_month,

COUNT(order\_id) AS order\_month\_count

FROM `TARGET\_SQL.orders`

GROUP BY past\_month

)

SELECT

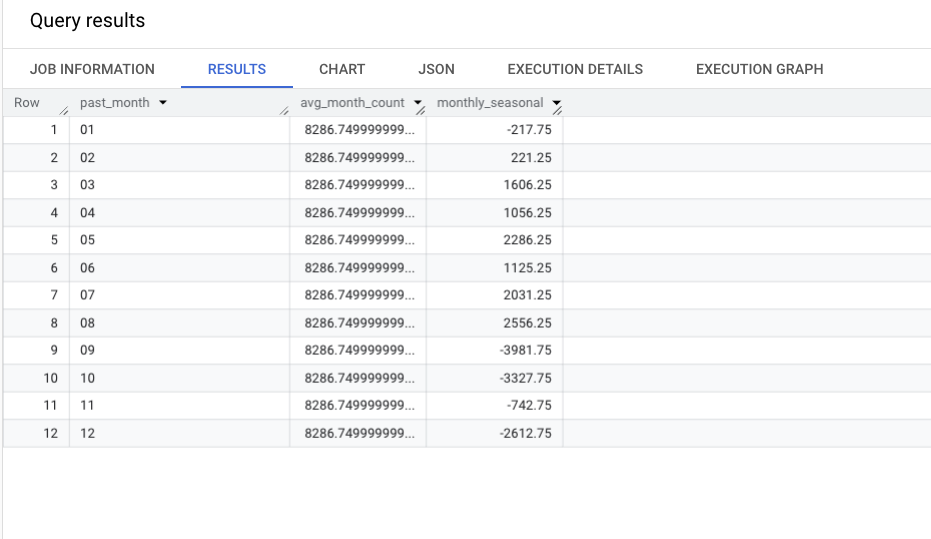
past\_month,

AVG(order\_month\_count) OVER() AS avg\_month\_count,

order\_month\_count - ROUND(AVG(order\_month\_count) OVER(), 2) AS monthly\_seasonal

FROM order\_data

ORDER BY past\_month



**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 order\_purchase\_timezone, count(order\_purchase\_timezone) FROM (

SELECT order\_purchase\_timestamp,

CASE

WHEN extract(hour FROM order\_purchase\_timestamp) between 0 and 6 THEN 'Dawan'

WHEN extract(hour FROM order\_purchase\_timestamp) between 7 and 12 THEN 'Mornings'

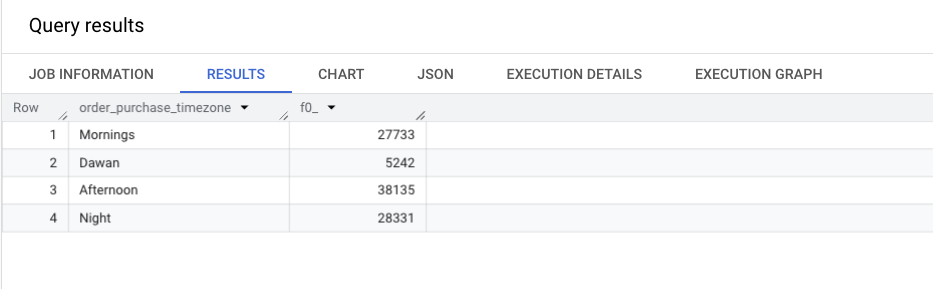
WHEN extract(hour FROM order\_purchase\_timestamp) between 13 and 18 THEN 'Afternoon'

ELSE 'Night'

END AS order\_purchase\_timezone

FROM `TARGET\_SQL.orders`)

GROUP BY order\_purchase\_timezone



-------------------------------------------------------------------------------------------------------------------------------------------------------

**Evolution of E-commerce orders in the Brazil region:**

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

SELECT geo.geolocation\_state,

extract(month FROM ord.order\_purchase\_timestamp) AS order\_month,

count(ord.order\_id) AS month\_on\_month

FROM `TARGET\_SQL.geolocation` AS geo

JOIN `TARGET\_SQL.customers` AS cust

ON geo.geolocation\_zip\_code\_prefix = cust.customer\_zip\_code\_prefix

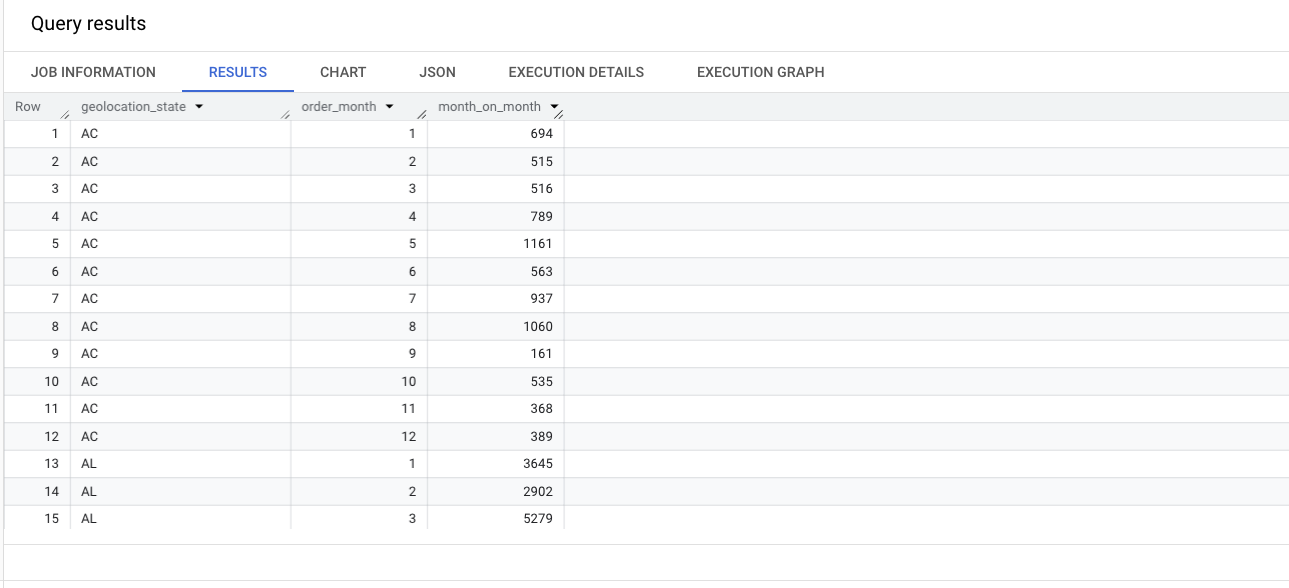
JOIN `TARGET\_SQL.orders` AS ord

ON cust.customer\_id = ord.customer\_id

GROUP BY geo.geolocation\_state, order\_month

order by geo.geolocation\_state, order\_month

limit 15



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

SELECT geo.geolocation\_state,count(distinct cust.customer\_id) AS no\_customers

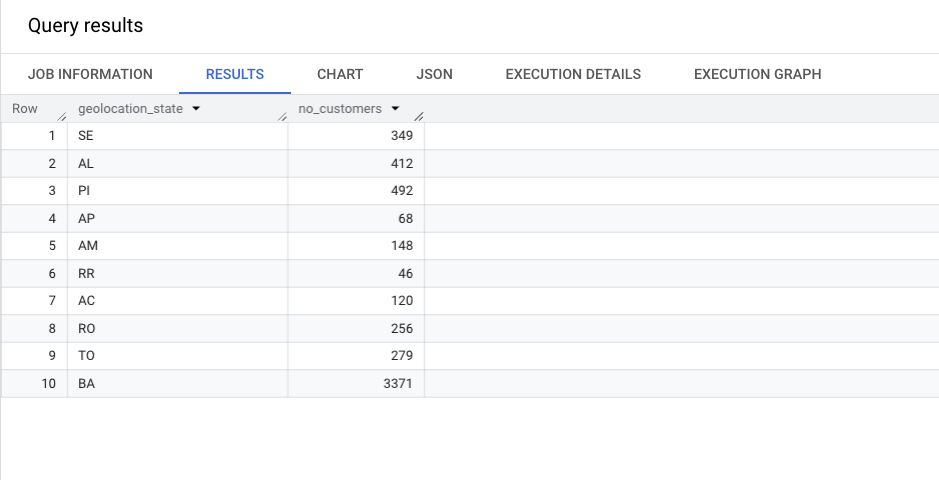
FROM `TARGET\_SQL.geolocation` AS geo

JOIN `TARGET\_SQL.customers` AS cust

ON geo.geolocation\_zip\_code\_prefix = cust.customer\_zip\_code\_prefix

GROUP BY geo.geolocation\_state

limit 10



---------------------------------------------------------------------------------------------------------------------------------------------------------

**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).**

WITH monthly\_avg\_payment AS (

SELECT

EXTRACT(YEAR FROM ord.order\_purchase\_timestamp) AS year\_x,

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

AVG(pmt.payment\_value) AS avg\_payment

FROM `TARGET\_SQL.orders` AS ord

JOIN `TARGET\_SQL.payments` AS pmt

ON ord.order\_id = pmt.order\_id

WHERE EXTRACT(YEAR FROM ord.order\_purchase\_timestamp) IN (2017, 2018)

AND EXTRACT(MONTH FROM ord.order\_purchase\_timestamp) BETWEEN 1 AND 8

GROUP BY year\_x, month\_x

)

SELECT

year\_x,

month\_x,

avg\_payment,

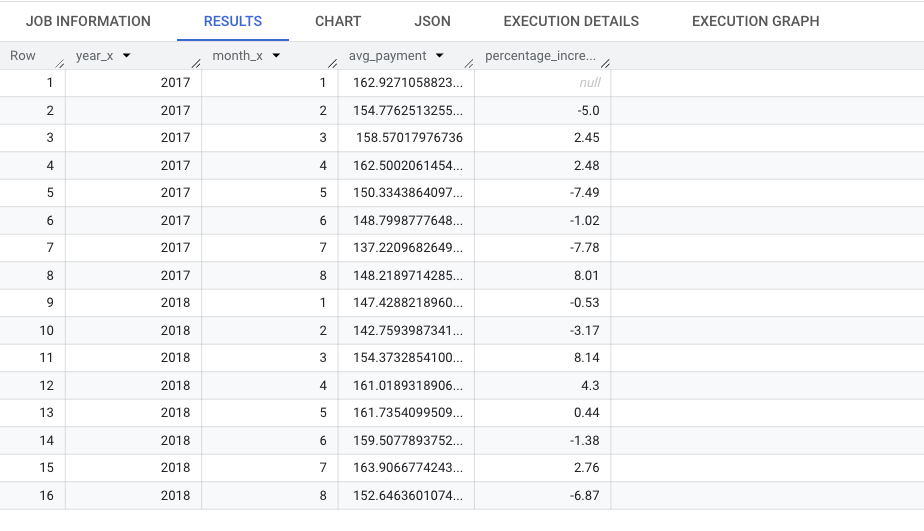
ROUND(

(avg\_payment - LAG(avg\_payment) OVER (ORDER BY year\_x, month\_x)) \* 100

/ LAG(avg\_payment) OVER (ORDER BY year\_x, month\_x), 2) AS percentage\_increase

FROM monthly\_avg\_payment

ORDER BY year\_x, month\_x;

****

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

SELECT

geo.geolocation\_state,

SUM(pmt.payment\_value) AS total\_price,

AVG(pmt.payment\_value) AS avg\_price

FROM `TARGET\_SQL.geolocation` AS geo

JOIN `TARGET\_SQL.customers` AS cust

ON geo.geolocation\_zip\_code\_prefix = cust.customer\_zip\_code\_prefix

JOIN `TARGET\_SQL.orders` AS ord

ON ord.customer\_id = cust.customer\_id

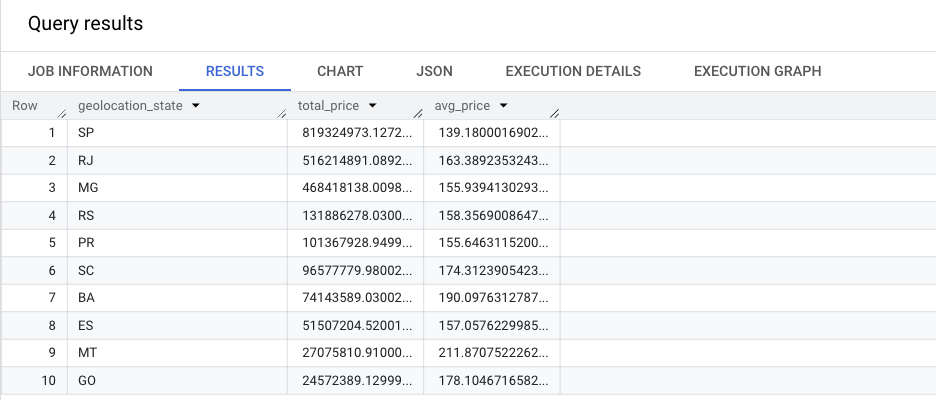
JOIN `TARGET\_SQL.payments` AS pmt

ON ord.order\_id = pmt.order\_id

GROUP BY geo.geolocation\_state

ORDER BY total\_price DESC

LIMIT 10;



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

SELECT

geo.geolocation\_state,

SUM(orditm.freight\_value) as total\_freight\_value,

AVG(orditm.freight\_value) as avg\_freight\_value

FROM `TARGET\_SQL.order\_items` AS orditm

JOIN `TARGET\_SQL.orders` AS ord

ON orditm.order\_id = ord.order\_id

JOIN `TARGET\_SQL.customers` AS cust

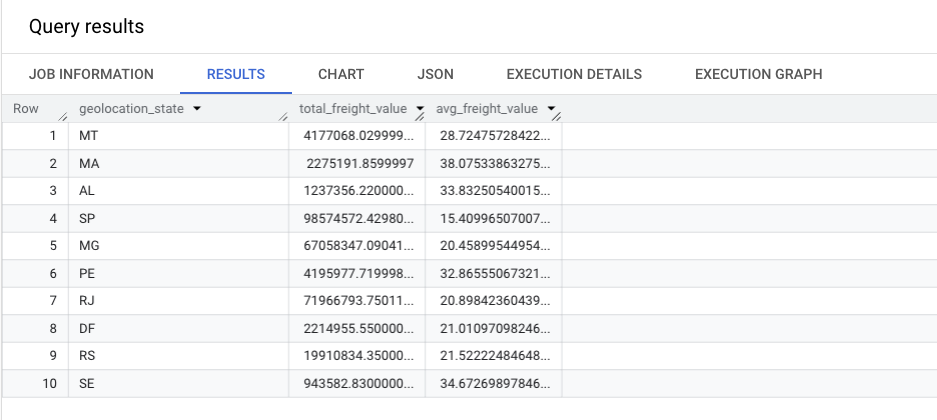
ON ord.customer\_id = cust.customer\_id

JOIN `TARGET\_SQL.geolocation` AS geo

ON cust.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix

GROUP BY geo.geolocation\_state

LIMIT 10

****

**-------------------------------------------------------------------------------------------------------------**

**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:**

* + **time\_to\_deliver = order\_delivered\_customer\_date - order\_purchase\_timestamp**
  + **diff\_estimated\_delivery = order\_delivered\_customer\_date - order\_estimated\_delivery\_date**

SELECT

order\_id,

customer\_id,

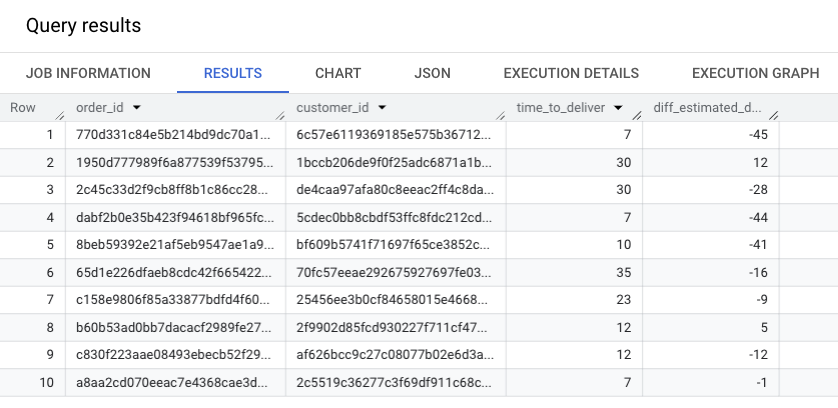
date\_diff(order\_delivered\_customer\_date, order\_purchase\_timestamp, day) as time\_to\_deliver,

date\_diff(order\_delivered\_customer\_date, order\_estimated\_delivery\_date, day) as diff\_estimated\_delivery

FROM `TARGET\_SQL.orders`

WHERE order\_delivered\_customer\_date IS NOT NULL

LIMIT 10;



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

WITH freight\_value AS(

SELECT

geo.geolocation\_state as state,

AVG(orditm.freight\_value) as avg\_freight\_value

FROM `TARGET\_SQL.order\_items` AS orditm

JOIN `TARGET\_SQL.orders` AS ord

ON orditm.order\_id = ord.order\_id

JOIN `TARGET\_SQL.customers` AS cust

ON ord.customer\_id = cust.customer\_id

JOIN `TARGET\_SQL.geolocation` AS geo

ON cust.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix

GROUP BY geo.geolocation\_state

)

-- HIGHEST

(SELECT \*

FROM freight\_value

ORDER BY avg\_freight\_value

LIMIT 5)

UNION ALL

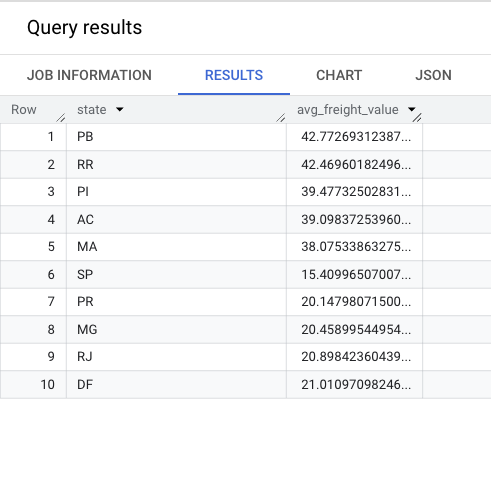
-- Lowest

(SELECT \*

FROM freight\_value

ORDER BY avg\_freight\_value DESC

LIMIT 5)



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

WITH high\_low as (

SELECT

geo.geolocation\_state as state,

AVG(DATE\_DIFF(ord.order\_estimated\_delivery\_date, ord.order\_delivered\_customer\_date, day)) as diff\_estimated\_delivery

FROM `TARGET\_SQL.orders` AS ord

JOIN `TARGET\_SQL.customers` AS cust

ON ord.customer\_id = cust.customer\_id

JOIN `TARGET\_SQL.geolocation` AS geo

ON cust.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix

GROUP BY geo.geolocation\_state

)

(SELECT state, diff\_estimated\_delivery

FROM high\_low

ORDER BY diff\_estimated\_delivery DESC

LIMIT 5)

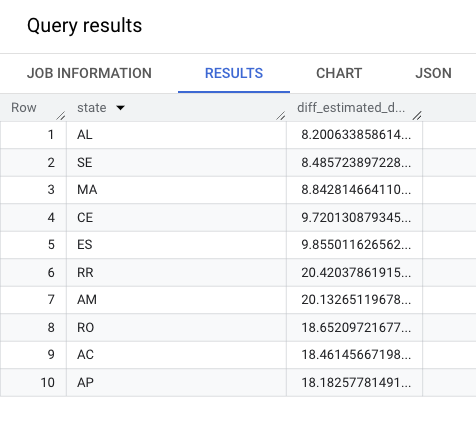
UNION ALL

(SELECT state, diff\_estimated\_delivery

FROM high\_low

ORDER BY diff\_estimated\_delivery ASC

LIMIT 5)



**4. 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 top\_five as (

SELECT

geo.geolocation\_state as state,

AVG(DATE\_DIFF(ord.order\_estimated\_delivery\_date, ord.order\_purchase\_timestamp, day)) as est\_date,

AVG(DATE\_DIFF(ord.order\_delivered\_customer\_date, ord.order\_purchase\_timestamp, day)) as delivered\_date

FROM `TARGET\_SQL.orders` AS ord

JOIN `TARGET\_SQL.customers` AS cust

ON ord.customer\_id = cust.customer\_id

JOIN `TARGET\_SQL.geolocation` AS geo

ON cust.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix

GROUP BY geo.geolocation\_state

)

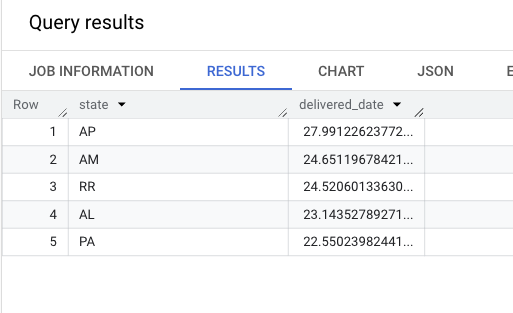
select top\_five.state, top\_five.delivered\_date

from top\_five

where top\_five.delivered\_date < top\_five.est\_date

ORDER BY top\_five.delivered\_date DESC

limit 5



------------------------------------------------------------------------------------------------------------------------------------------------------

**Analysis based on the payments**

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

SELECT pmt.payment\_type,

EXTRACT(YEAR FROM ord.order\_purchase\_timestamp) AS year\_x,

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

COUNT(pmt.order\_id) AS order\_count

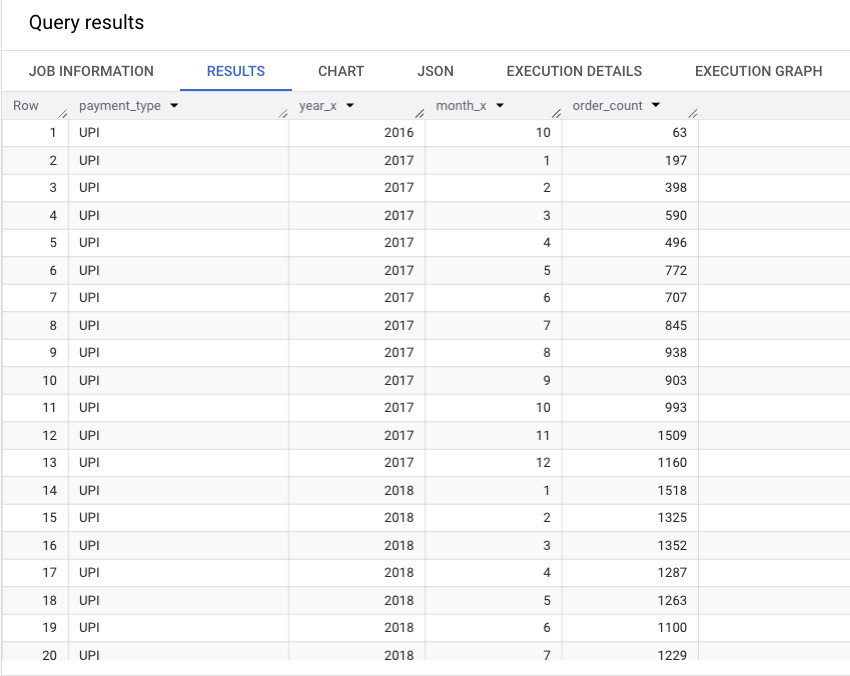
FROM `TARGET\_SQL.payments` AS pmt

JOIN `TARGET\_SQL.orders` AS ord

ON pmt.order\_id = ord.order\_id

GROUP BY pmt.payment\_type ,EXTRACT(YEAR FROM ord.order\_purchase\_timestamp), EXTRACT(MONTH FROM ord.order\_purchase\_timestamp)

ORDER BY pmt.payment\_type, year\_x, month\_x



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

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

FROM `TARGET\_SQL.payments`

WHERE payment\_installments > 0

GROUP BY payment\_type

