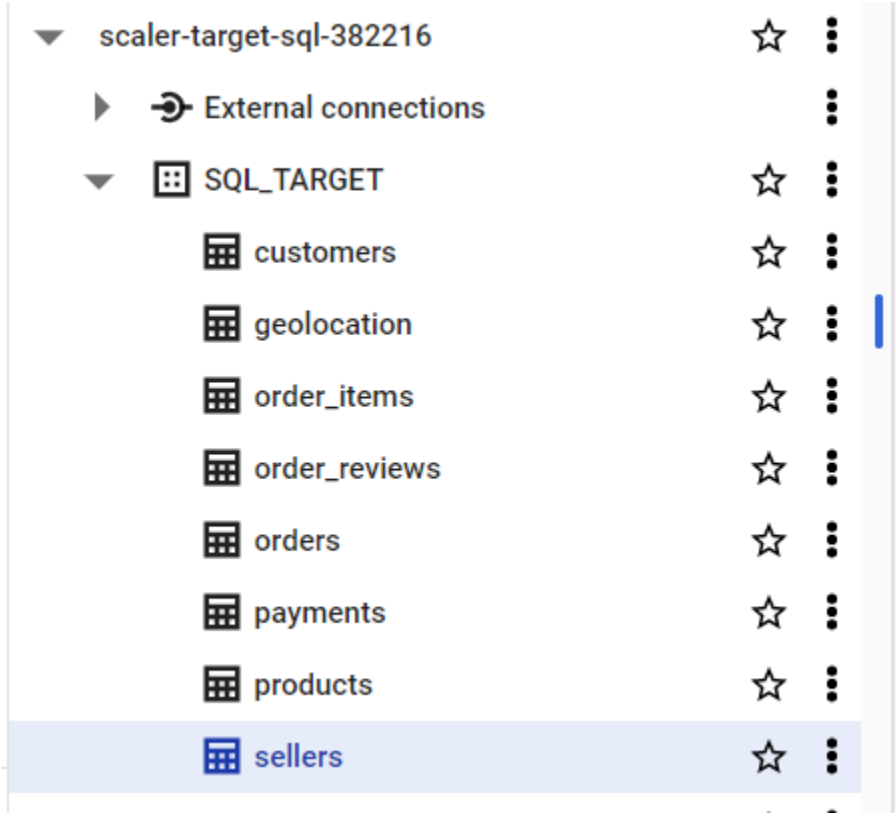
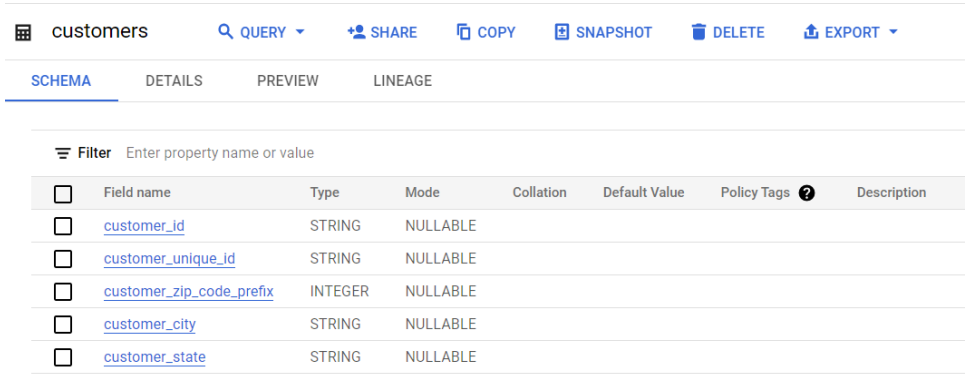
1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

Project and Dataset overview:



A. Data type of columns in a table



Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
customer_id	STRING	NULLABLE				
customer_unique_id	STRING	NULLABLE				
customer_zip_code_prefix	INTEGER	NULLABLE				
customer_city	STRING	NULLABLE				
customer_state	STRING	NULLABLE				

geolocation

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
geolocation_zip_code_prefix	INTEGER	NULLABLE				
geolocation_lat	FLOAT	NULLABLE				
geolocation_lng	FLOAT	NULLABLE				
geolocation_city	STRING	NULLABLE				
geolocation_state	STRING	NULLABLE				

order_items

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
order_id	STRING	NULLABLE				
order_item_id	INTEGER	NULLABLE				
product_id	STRING	NULLABLE				
seller_id	STRING	NULLABLE				
shipping_limit_date	TIMESTAMP	NULLABLE				
price	FLOAT	NULLABLE				
freight_value	FLOAT	NULLABLE				

order_items

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
order_id	STRING	NULLABLE				
order_item_id	INTEGER	NULLABLE				
product_id	STRING	NULLABLE				
seller_id	STRING	NULLABLE				
shipping_limit_date	TIMESTAMP	NULLABLE				
price	FLOAT	NULLABLE				
freight_value	FLOAT	NULLABLE				

orders

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
order_id	STRING	NULLABLE				
customer_id	STRING	NULLABLE				
order_status	STRING	NULLABLE				
order_purchase_timestamp	TIMESTAMP	NULLABLE				
order_approved_at	TIMESTAMP	NULLABLE				
order_delivered_carrier_date	TIMESTAMP	NULLABLE				
order_delivered_customer_date	TIMESTAMP	NULLABLE				
order_estimated_delivery_date	TIMESTAMP	NULLABLE				

payments

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
order_id	STRING	NULLABLE				
payment_sequential	INTEGER	NULLABLE				
payment_type	STRING	NULLABLE				
payment_installments	INTEGER	NULLABLE				
payment_value	FLOAT	NULLABLE				

products

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
product_id	STRING	NULLABLE				
product_category	STRING	NULLABLE				
product_name_length	INTEGER	NULLABLE				
product_description_length	INTEGER	NULLABLE				
product_photos_qty	INTEGER	NULLABLE				
product_weight_g	INTEGER	NULLABLE				
product_length_cm	INTEGER	NULLABLE				
product_height_cm	INTEGER	NULLABLE				
product_width_cm	INTEGER	NULLABLE				

sellers

QUERY SHARE COPY SNAPSHOT DELETE EXPORT

SCHEMA DETAILS PREVIEW LINEAGE

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
seller_id	STRING	NULLABLE				
seller_zip_code_prefix	INTEGER	NULLABLE				
seller_city	STRING	NULLABLE				
seller_state	STRING	NULLABLE				

B. Time period for which the data is given

index	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date	order_delivered_customer_date	order_estir
count	1000	1000	1000	1000	995	995	0	1000
unique	1000	1000	2	999	992	992	0	307
top	7a4df5d8cff4090e541401a20a22bb80	725e9c75605414b21fd8c8d5a1c2f1d6	shipped	2017-01-08 19:27:22+00:00	2017-04-14 22:10:16+00:00	2018-07-30 15:48:00+00:00	NaN	2018-04-12
freq	1	1	995	2	2	2	NaN	12
first	NaN	NaN	NaN	2016-09-04 21:15:19+00:00	2016-10-05 03:10:59+00:00	2016-10-11 16:12:23+00:00	NaN	2016-10-20
last	NaN	NaN	NaN	2018-09-03 09:06:57+00:00	2018-09-03 17:40:06+00:00	2018-09-04 15:25:00+00:00	NaN	2018-09-18

```
SELECT min(order_purchase_timestamp) as start_time,
max(order_purchase_timestamp) as end_time FROM `scaler-target-sql-
382216.SQL_TARGET.orders`
```

RUN SAVE SHARE SCHEDULE MORE

```
1 SELECT min(order_purchase_timestamp) as start_time, max(order_purchase_timestamp) as end_time FROM `scaler-target-sql-382216.SQL_TARGET.orders`
```

Press Alt+F1

Query results

SAVE RESULTS EXPLO

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

This BigQuery table is being used for Change Data Capture(CDC), and this preview was last updated at Fri Mar 31 2023 03:20:04 GMT+0530 (India Standard Time). To pull real-
please query the table.

Row	start_time	end_time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

C. Cities and States of customers ordered during the given period

```
SELECT distinct G.geolocation_city as City, G.geolocation_state as State FROM
`scaler-target-sql-382216.SQL_TARGET.customers` C join `scaler-target-sql-
382216.SQL_TARGET.geolocation` G on
C.customer_zip_code_prefix=G.geolocation_zip_code_prefix LIMIT 100
```

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
SELECT distinct G.geolocation_city as City, G.geolocation_state as State FROM `scaler-target-sql-382216.SQL_TARGET.customers` C join `scaler-target-sql-382216.SQL_TARGET.geolocation` G on C.customer_zip_code_prefix=G.geolocation_zip_code_prefix LIMIT 100
```

Press Alt+F1 for Accessibility Options

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	City	State
1	aracaju	SE
2	riachuelo	SE
3	nossa senhora do socorro	SE
4	barra dos coqueiros	SE
5	itaporanga d'ajuda	SE
6	sao cristovao	SE
7	são cristóvão	SE
8	santo amaro das brotas	SE
9	pirambu	SE
10	umbauba	SE

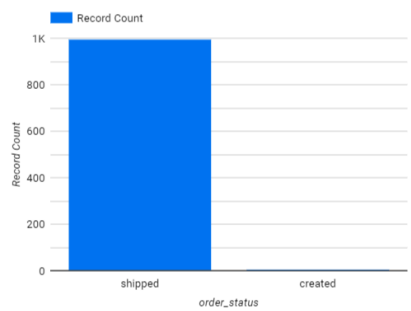
==========

2. In-depth Exploration:

orders

	order_id	Record Count
1.	d55668bb11f39447c1b40421a93...	1
2.	35de4050331c6c644cddc86f4f2d...	1
3.	b5359909123fa03c50bdb0cfed07...	1
4.	dba5062fbda3af4fb6c33b1e040c...	1
5.	90ab3e7d52544ec7bc3363c8268...	1
6.	fa65dad1b0e818e3ccc5cb0e3923...	1
7.	1df2775799eecdf9dd85024255c6...	1
8.	6190a94657e1012983a274b831d...	1
9.	58ce513a55c740a3a81e8c8b75e...	1
10.	088683f795a3d30bfd61152c4fab...	1
11.	aa380313c19905dd1651bd21e75...	1

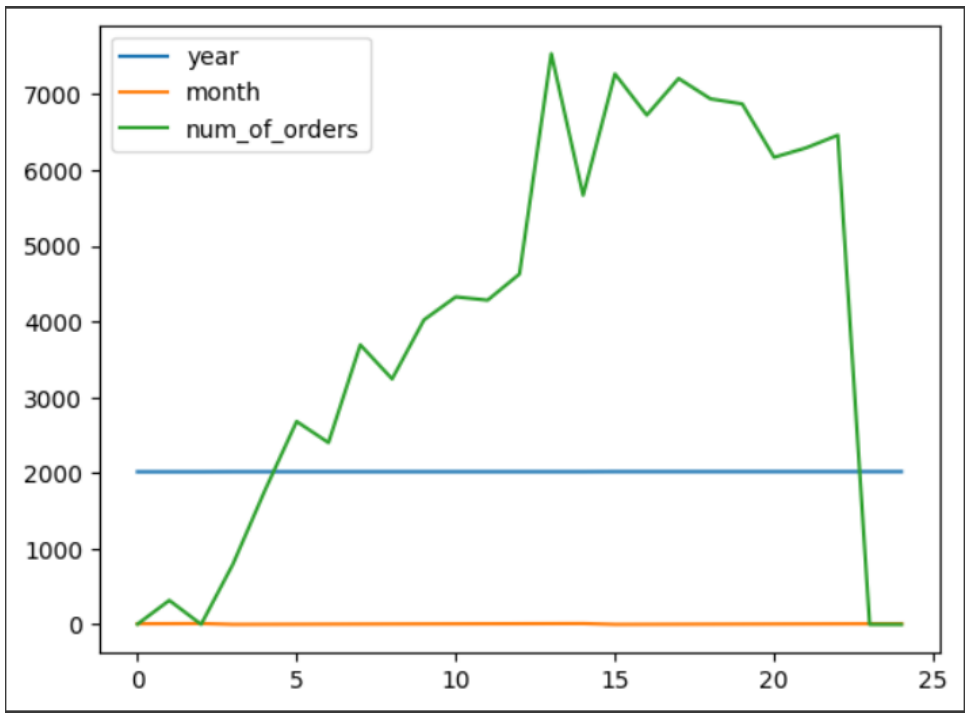
1 - 50 / 1000



a. 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?

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year,EXTRACT(MONTH FROM order_purchase_timestamp) AS month,count(order_approved_at)  FROM `scaler-target-sql-382216.SQL_TARGET.orders` O

group by year, month

order by year , month LIMIT 100
```

index	year	month	num_of_orders
0	2016	9	4
1	2016	10	318
2	2016	12	1
3	2017	1	797
4	2017	2	1766
5	2017	3	2680
6	2017	4	2400
7	2017	5	3691
8	2017	6	3241
9	2017	7	4021
10	2017	8	4325
11	2017	9	4281
12	2017	10	4626
13	2017	11	7535
14	2017	12	5666
15	2018	1	7268
16	2018	2	6724
17	2018	3	7208
18	2018	4	6939
19	2018	5	6872
20	2018	6	6167
21	2018	7	6291
22	2018	8	6459
23	2018	9	1
24	2018	10	0



Monthly Sales have been increasing in terms of number of orders made. ALso it is seen that in Nov 2017 the #orders were increased which could be due to the christmas/festivals.

The Mid year period is consistent in terms of the number of orders and the variance is low whereas around year end the variance is more.

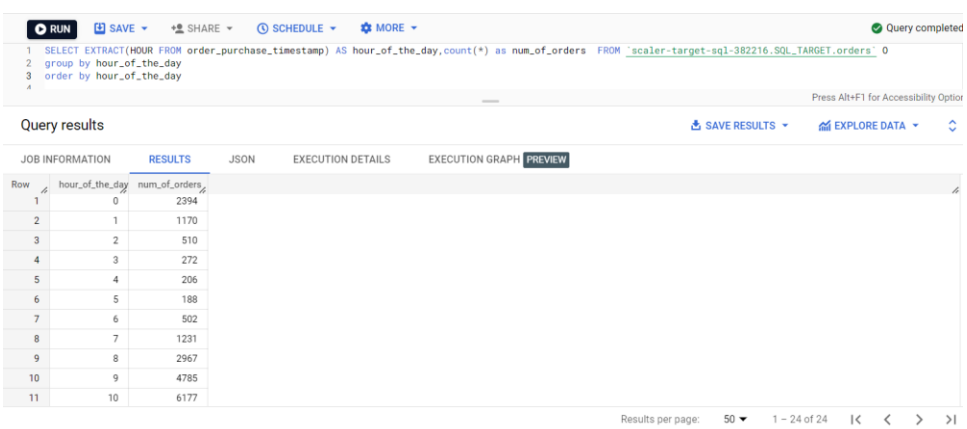
b. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

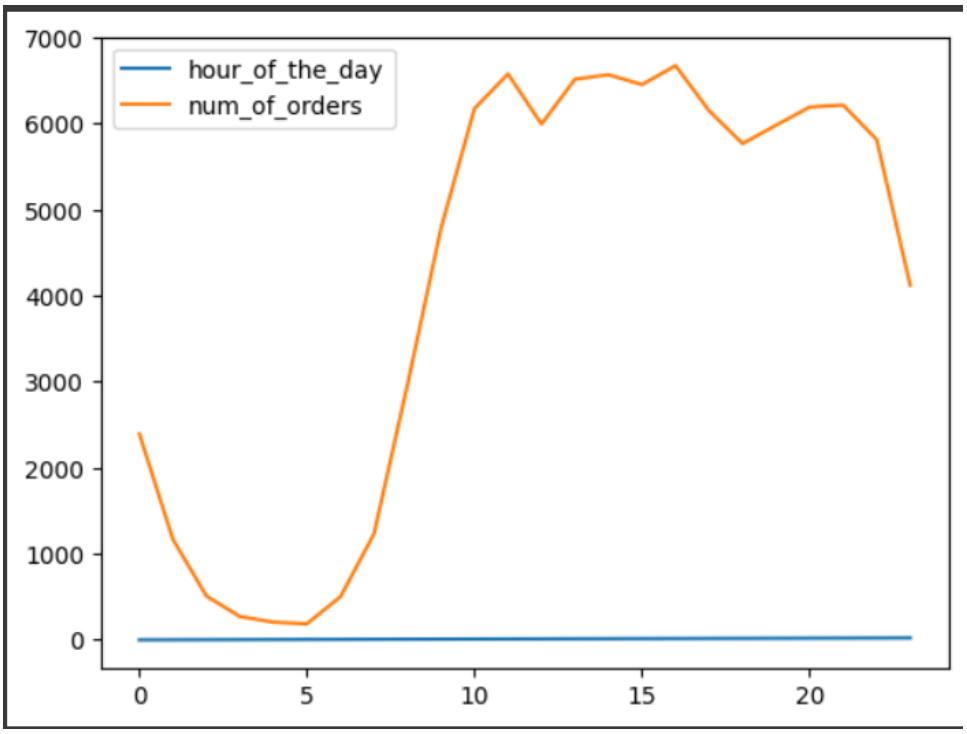
```
select

CASE

    WHEN Q1.hour_of_the_day > 0 and Q1.hour_of_the_day < 06  THEN 'DAWN'
```

```
    WHEN Q1.hour_of_the_day > 06 and Q1.hour_of_the_day < 12  THEN 'MORNING'
    WHEN Q1.hour_of_the_day > 12 and Q1.hour_of_the_day < 18  THEN 'AFTERNOON'
    ELSE 'NIGHT'
END AS Time_of_Day,sum(Q1.num_of_orders) as Order_count
from
(SELECT EXTRACT(HOUR FROM order_purchase_timestamp) AS hour_of_the_day,count(*) as
num_of_orders  FROM `scaler-target-sql-382216.SQL_TARGET.orders` O
group by hour_of_the_day) Q1
group by Time_of_Day
```

Row	Time_of_Day	Order_count
1	MORNING	21738
2	DAWN	2346
3	AFTERNOON	32366
4	NIGHT	42991

```
SELECT EXTRACT(HOUR FROM order_purchase_timestamp) AS hour_of_the_day,count(*) as
num_of_orders  FROM `scaler-target-sql-382216.SQL_TARGET.orders` O
group by hour_of_the_day
order by hour_of_the_day
```

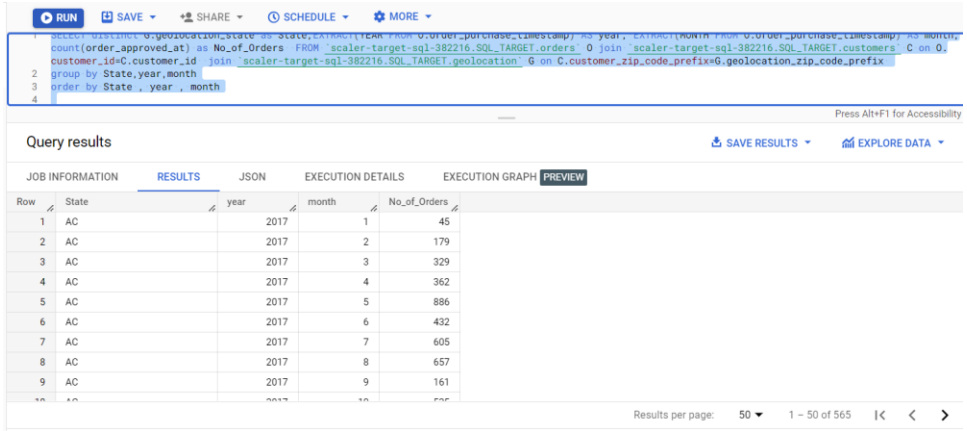


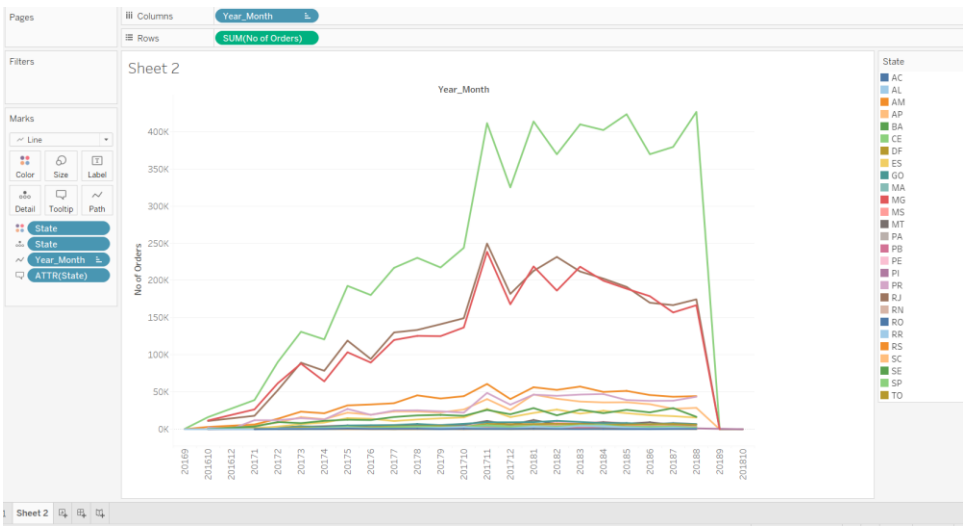



```
From the above graph it is clear that the Brazilian customers tend to order
more between 10 AM and 08 PM, and the transaction density is very low during
Night and early morning hours.

==================================================================
```

3. Evolution of E-commerce orders in the Brazil region:
    a. Get month on month orders by states

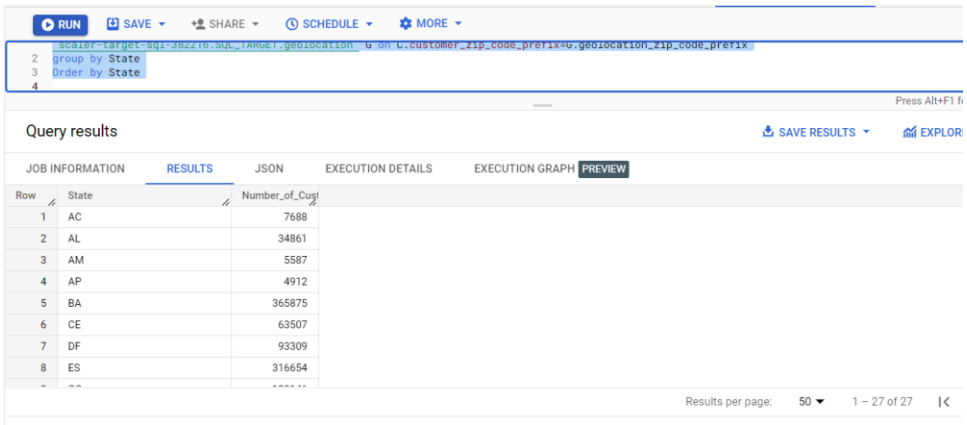
```
SELECT distinct G.geolocation_state as State,EXTRACT(YEAR FROM
O.order_purchase_timestamp) AS year, EXTRACT(MONTH FROM O.order_purchase_timestamp) AS
month,count(order_approved_at) as No_of_Orders  FROM `scaler-target-sql-
382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on
O.customer_id=C.customer_id  join `scaler-target-sql-382216.SQL_TARGET.geolocation` G
on C.customer_zip_code_prefix=G.geolocation_zip_code_prefix

group by State,year,month

order by State , year , month
```

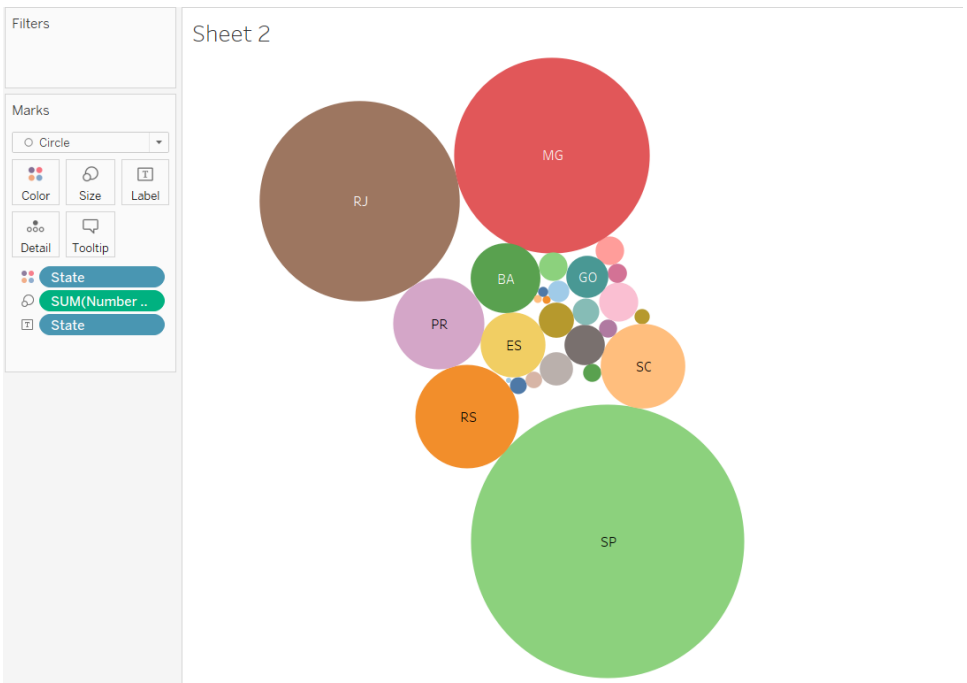




b. Distribution of customers across the states in Brazil

```
SELECT G.geolocation_state as State, count(C.customer_id) as Number_of_Customers FROM
`scaler-target-sql-382216.SQL_TARGET.customers` C join `scaler-target-sql-
382216.SQL_TARGET.geolocation` G on
C.customer_zip_code_prefix=G.geolocation_zip_code_prefix

group by State

Order by State
```





==========

4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.
    a. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

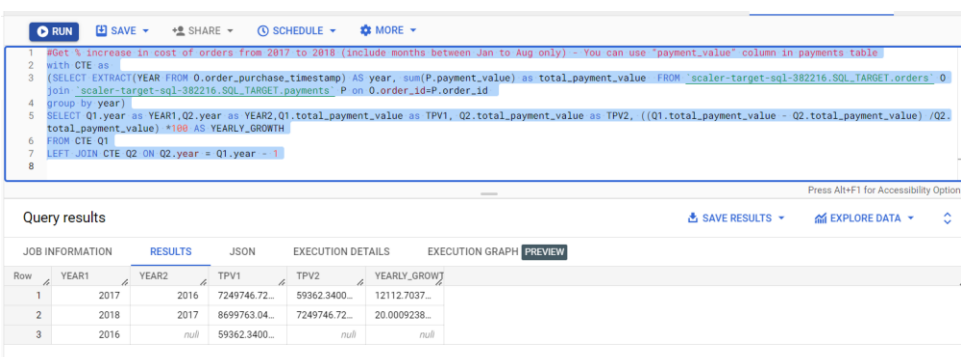
```
SELECT EXTRACT(YEAR FROM O.order_purchase_timestamp) AS year, sum(P.payment_value)
FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-
382216.SQL_TARGET.payments` P on O.order_id=P.order_id

group by year
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	year	f0_
1	2017	7249746.7299996857
2	2018	8699763.0499998648
3	2016	59362.340000000026

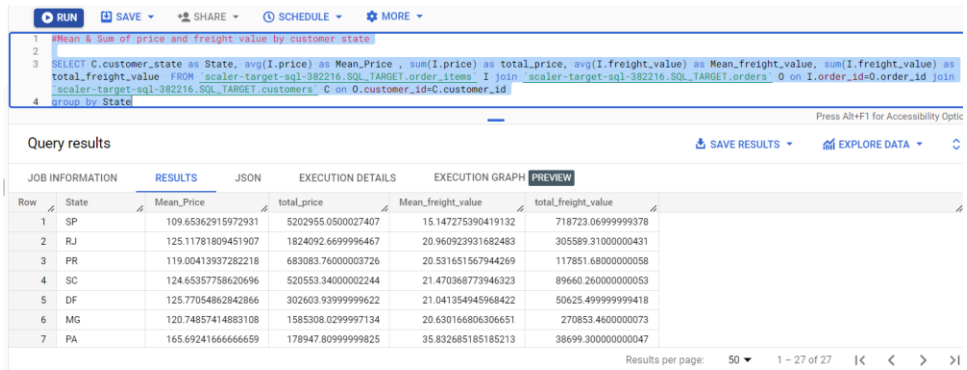
```
#Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

with CTE as

(SELECT EXTRACT(YEAR FROM O.order_purchase_timestamp) AS year, sum(P.payment_value) as total_payment_value  FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.payments` P on O.order_id=P.order_id

group by year)

SELECT Q1.year as YEAR1,Q2.year as YEAR2,Q1.total_payment_value as TPV1, Q2.total_payment_value as TPV2, ((Q1.total_payment_value - Q2.total_payment_value) /Q2.total_payment_value) *100 AS YEARLY_GROWTH

FROM CTE Q1

LEFT JOIN CTE Q2 ON Q2.year = Q1.year - 1
```



b. Mean & Sum of price and freight value by customer state

```
#Mean & Sum of price and freight value by customer state
```

```
SELECT C.customer_state as State, avg(I.price) as Mean_Price , sum(I.price) as
total_price, avg(I.freight_value) as Mean_freight_value, sum(I.freight_value) as
total_freight_value  FROM `scaler-target-sql-382216.SQL_TARGET.order_items` I join
`scaler-target-sql-382216.SQL_TARGET.orders` O on I.order_id=O.order_id join `scaler-
target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id

group by State
```

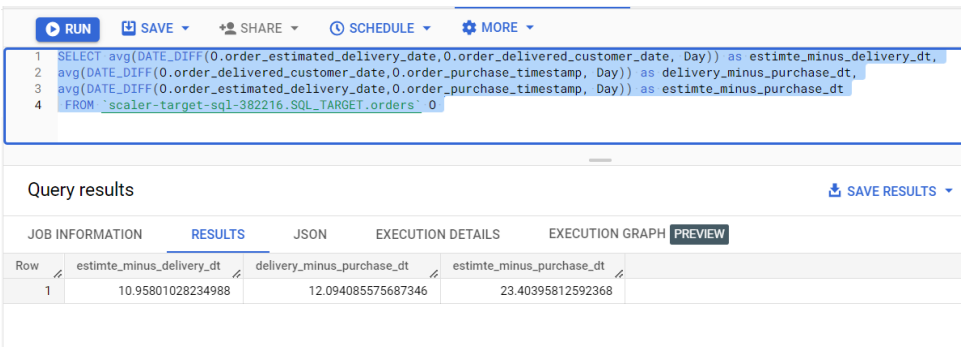


==========

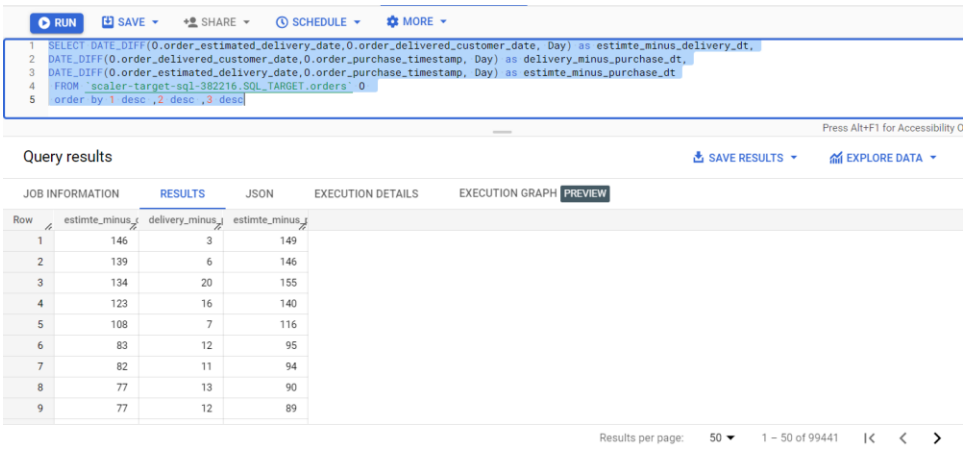
5. Analysis on sales, freight and delivery time

A. Calculate days between purchasing, delivering and estimated delivery

```
SELECT avg(DATE_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date,
Day)) as estimte_minus_delivery_dt,

avg(DATE_DIFF(O.order_delivered_customer_date,O.order_purchase_timestamp, Day)) as
delivery_minus_purchase_dt,

avg(DATE_DIFF(O.order_estimated_delivery_date,O.order_purchase_timestamp, Day)) as
estimte_minus_purchase_dt

 FROM `scaler-target-sql-382216.SQL_TARGET.orders` O
```




```
SELECT DATE_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, Day)
as estimte_minus_delivery_dt,

DATE_DIFF(O.order_delivered_customer_date,O.order_purchase_timestamp, Day) as
delivery_minus_purchase_dt,

DATE_DIFF(O.order_estimated_delivery_date,O.order_purchase_timestamp, Day) as
estimte_minus_purchase_dt

 FROM `scaler-target-sql-382216.SQL_TARGET.orders` O

 order by 1 desc ,2 desc ,3 desc
```



```
SELECT DATE_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, Day) as estimte_minus_delivery_dt,
DATE_DIFF(O.order_delivered_customer_date,O.order_purchase_timestamp, Day) as delivery_minus_purchase_dt,
DATE_DIFF(O.order_estimated_delivery_date,O.order_purchase_timestamp, Day) as estimte_minus_purchase_dt
 FROM `scaler-target-sql-382216.SQL_TARGET.orders` O
 order by 1 desc ,2 desc ,3 desc
```

Query results

Row	estimte_minus_d	delivery_minus_p	estimte_minus_p
1	146	3	149
2	139	6	146
3	134	20	155
4	123	16	140
5	108	7	116
6	83	12	95
7	82	11	94
8	77	13	90
9	77	12	89

B. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:
   - time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
   - diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
SELECT O.order_id,
TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24
as time_to_delivery,

TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date,
HOUR)/24 as diff_estimated_delivery,

 FROM `scaler-target-sql-382216.SQL_TARGET.orders` O

 order by 2 desc ,3 desc
```

RUN SAVE SHARE SCHEDULE MORE

```
SELECT O.order_id, TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24  as time_to_delivery,
TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24 as diff_estimated_delivery,
 FROM `scaler-target-sql-382216.SQL_TARGET.orders` O
 order by 2 desc ,3 desc
```

Press Alt+

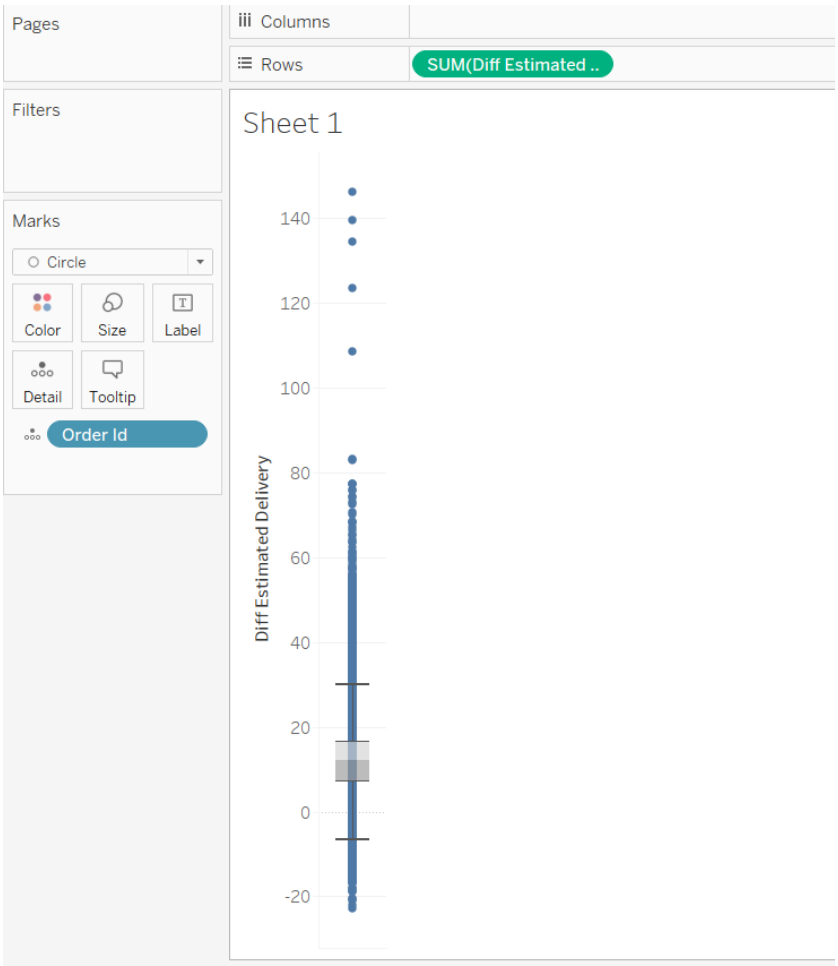
Query results

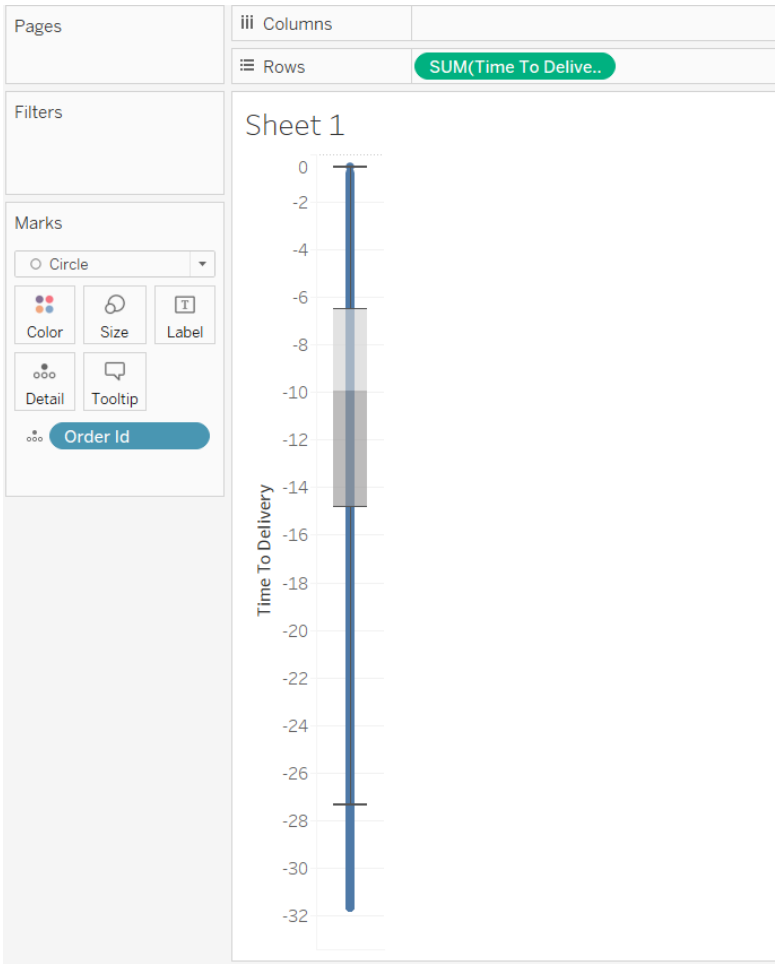
SAVE RESULTS EXP

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	order_id	time_to_delivery	diff_estimated_delivery
1	1d893dd7ca5f77ebf5f59f0d20...	-0.5	10.083333333333334
2	434cecee7d1a65fc65358a632...	-0.75	19.625
3	8339b608be0d84fca9d8da68b...	-0.83333333333333337	27.25
4	f3c6775ba3d2d9fe2826f93b71...	-0.83333333333333337	11.625
5	bb5a519e352b45b714192a02f...	-0.875	25.625
6	e65f1eeee1f52024ad1dcd034...	-0.875	9.4583333333333339
7	21a8ffca665bc7a1087d31751...	-0.91666666666666663	11.541666666666666
8	d5fbeedc85190ba88580d6f82...	-0.91666666666666663	7.541666666666667
9	38c1e3d4ed6a13cd0cf612d4c...	-0.95833333333333337	16.333333333333332

Results per page: 50 1 – 50 of 99441



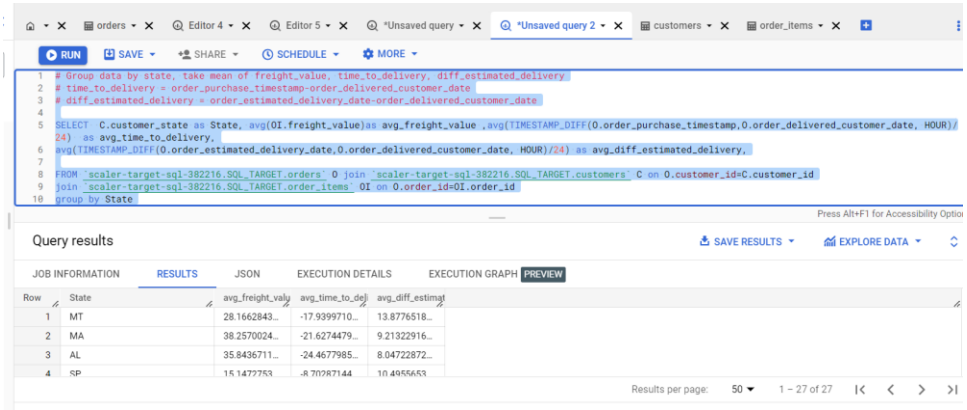


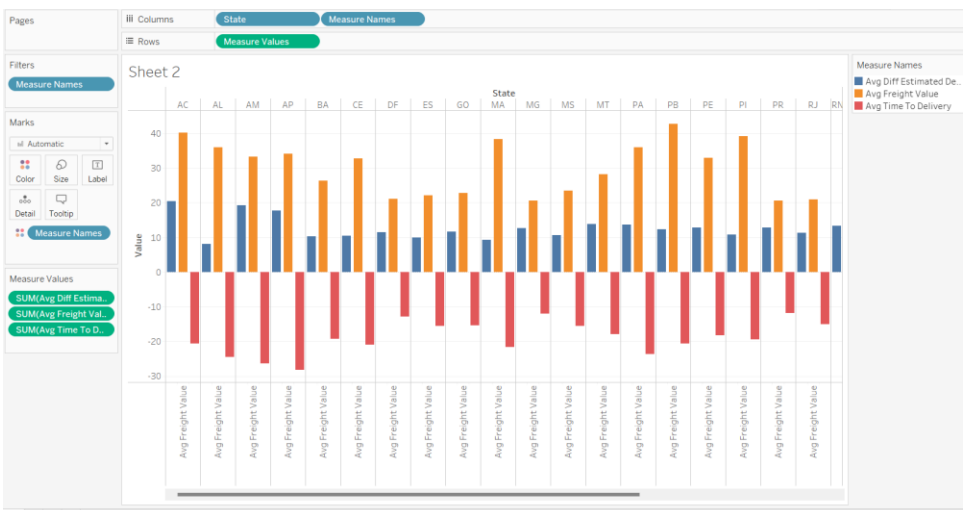
C. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
# Group data by state, take mean of freight_value, time_to_delivery,
diff_estimated_delivery

# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

# diff_estimated_delivery = order_estimated_delivery_date-
order_delivered_customer_date
```

```
SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value
,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date,
HOUR)/24)  as avg_time_to_delivery,

avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date,
HOUR)/24) as avg_diff_estimated_delivery,
```

```
FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id

join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id

group by State
```





D. Sort the data to get the following:

- Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

Lowest:

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
# diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value ,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24)  as avg_time_to_delivery,
avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24) as avg_diff_estimated_delivery,

FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id
join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id
group by State order by avg_freight_value limit 5
```

Press Alt+F1 for Accessibility Options

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	avg_freight_valu	avg_time_to_del	avg_diff_estimat
1	SP	15.1472753...	-8.70287144...	10.4955653...
2	PR	20.5316515...	-11.9266831...	12.7739201...
3	MG	20.6301668...	-11.9630138...	12.6213775...
4	RJ	20.9609239...	-15.1276244...	11.2934958...
5	DF	21.0413549...	-12.9382696...	11.4758138...

Highest :

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
# diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value ,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24)  as avg_time_to_delivery,
avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24) as avg_diff_estimated_delivery,

FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id
join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id
group by State order by avg_freight_value desc limit 5
```

Press Alt+F1 for Accessibility Option

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	avg_freight_valu	avg_time_to_del	avg_diff_estimat
1	RR	42.9844230...	-28.2074275...	17.6014492...
2	PB	42.7238039...	-20.5695392...	12.3553043...
3	RO	41.0697122...	-19.7196275...	19.3237179...
4	AC	40.0733695...	-20.6945970...	20.3163919...
5	PI	39.1479704...	-19.3646430...	10.8381931...

- Top 5 states with highest/lowest average time to delivery

Lowest

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
# diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value ,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24)  as avg_time_to_delivery,
avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24) as avg_diff_estimated_delivery,

FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id
join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id
group by State order by avg_time_to_delivery limit 5
```

Press Alt+F1 for Accessibility Options

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	RR	42.984423076923093	-28.207427536231883	17.601449275362317
2	AP	34.006097560975618	-28.185699588477366	17.75051440329219
3	AM	33.205393939393936	-26.368609406952967	19.207055214723923
4	AL	35.843671171171152	-24.467798594847782	8.0472287275565932
5	PA	35.832685185185177	-23.733357052498413	13.553723908918411

Highest:

RUN SAVE SHARE SCHEDULE MORE Query completed

```
# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
# diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value ,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24)  as avg_time_to_delivery,
avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24) as avg_diff_estimated_delivery,

FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id
join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id
group by State order by avg_time_to_delivery desc limit 5
```

Press Alt+F1 for Accessibility Optior

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.147275390419248	-8.7028714409778978	10.49556535251091
2	PR	20.531651567944248	-11.926683188764931	12.773920162860671
3	MG	20.630166806306541	-11.963013857706892	12.621377512838386
4	DF	21.041354945968383	-12.938269639065819	11.475813871196051
5	SC	21.470368773946436	-14.980234260614958	10.856251016756175

- Top 5 states where delivery is really fast/ not so fast compared to estimated date

Lowest

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
# diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value ,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24)  as avg_time_to_delivery,
avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24) as avg_diff_estimated_delivery,

FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id
join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id
group by State order by avg_diff_estimated_delivery asc limit 5
```

Press Alt+F1 for Accessibility Option

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	AL	35.843671171171152	-24.467798594847782	8.0472287275565932
2	MA	38.25700242718446	-21.627447916666689	9.2132291666666628
3	SE	36.653168831168855	-21.446777777777772	9.3109999999999911
4	ES	22.058776595744682	-15.62833333333333	9.9339513108614224
5	BA	26.363958936562248	-19.227135939904066	10.273893565028521

PERSONAL HISTORY PROJECT HISTORY REFRESH

Highest

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
# time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
# diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT  C.customer_state as State, avg(OI.freight_value)as avg_freight_value ,avg(TIMESTAMP_DIFF(O.order_purchase_timestamp,O.order_delivered_customer_date, HOUR)/24)  as avg_time_to_delivery,
avg(TIMESTAMP_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date, HOUR)/24) as avg_diff_estimated_delivery,

FROM `scaler-target-sql-382216.SQL_TARGET.orders` O join `scaler-target-sql-382216.SQL_TARGET.customers` C on O.customer_id=C.customer_id
join `scaler-target-sql-382216.SQL_TARGET.order_items` OI on O.order_id=OI.order_id
group by State order by avg_diff_estimated_delivery desc limit 5
```

Press Alt+F1 for Accessibility Options

Query results SAVE RESULTS EXPLORE DATA

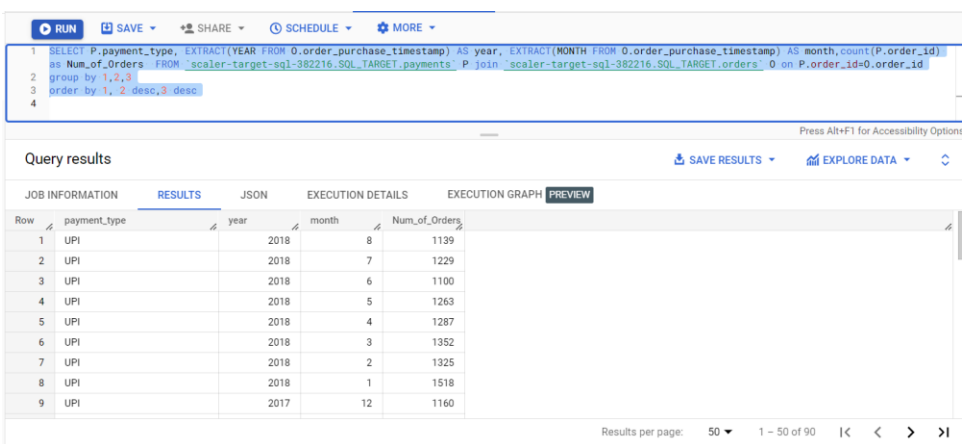
JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	AC	40.073369565217405	-20.694597069597069	20.316391941391945
2	RO	41.069712230215842	-19.719627594627593	19.323717948717952
3	AM	33.205393939393936	-26.368609406952967	19.207055214723923
4	AP	34.006097560975618	-28.185699588477366	17.75051440329219
5	RR	42.984423076923093	-28.207427536231883	17.601449275362317

6. Payment type analysis:

A. Month over Month count of orders for different payment types

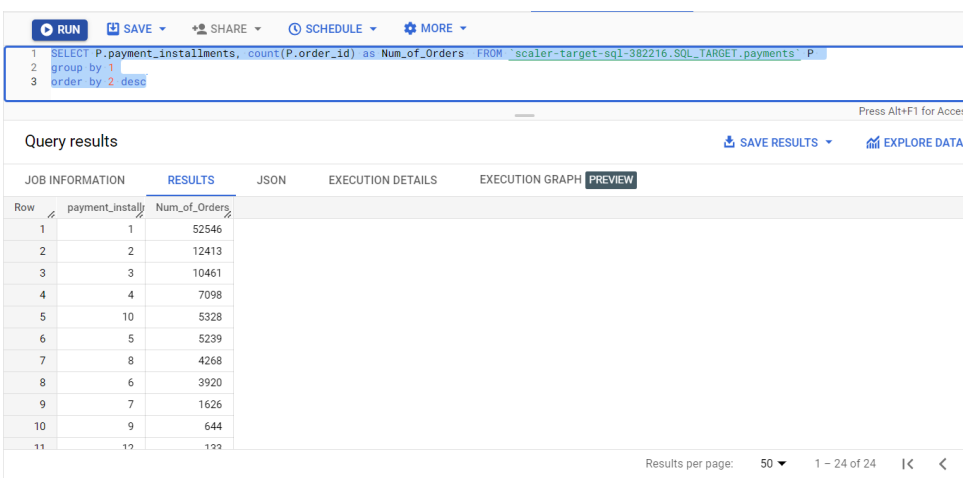
```
SELECT P.payment_type, EXTRACT(YEAR FROM O.order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM O.order_purchase_timestamp) AS month,count(P.order_id) as
Num_of_Orders  FROM `scaler-target-sql-382216.SQL_TARGET.payments` P join `scaler-
target-sql-382216.SQL_TARGET.orders` O on P.order_id=O.order_id

group by 1,2,3

order by 1, 2 desc,3 desc
```



Row	payment_type	year	month	Num_of_Orders
1	UPI	2018	8	1139
2	UPI	2018	7	1229
3	UPI	2018	6	1100
4	UPI	2018	5	1263
5	UPI	2018	4	1287
6	UPI	2018	3	1352
7	UPI	2018	2	1325
8	UPI	2018	1	1518
9	UPI	2017	12	1160

B. Count of orders based on the no. of payment installments

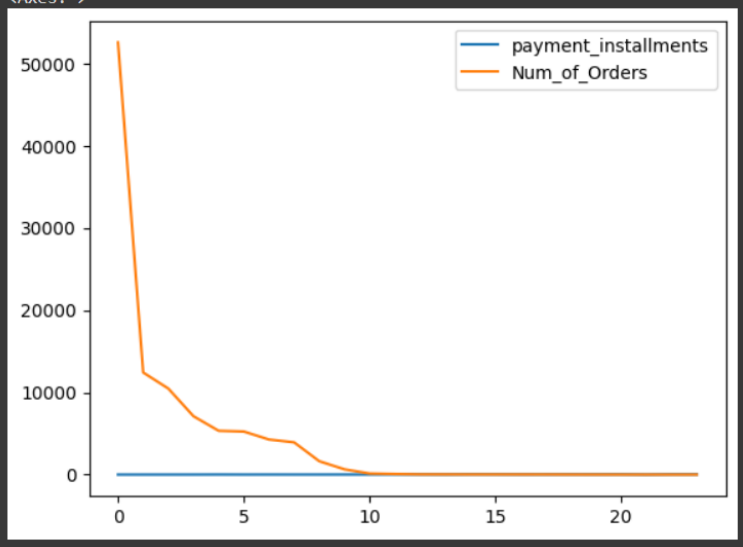
```
SELECT P.payment_installments, count(P.order_id) as Num_of_Orders  FROM `scaler-target-sql-382216.SQL_TARGET.payments` P

group by 1

order by 2 desc
```



index	payment_installments	Num_of_Orders
0	1	52546
1	2	12413
2	3	10461
3	4	7098
4	10	5328
5	5	5239
6	8	4268
7	6	3920
8	7	1626
9	9	644
10	12	133
11	15	74
12	18	27
13	11	23
14	24	18
15	20	17
16	13	16
17	14	15
18	17	8
19	16	5
20	21	3
21	0	2
22	22	1
23	23	1

```
results.plot()
```

```
<Axes: >
```



==========

Insights and Recommendations:

more than 50% customers are paying in single installment -- there is sharp drop after 6 and 10 installments

-- sudden drop at 7 installments could be linked to human behaviorism of choosing simple composite number of installments rather than difficult to calculate numbers like 7 , 11 , 13 etc

--> work out plans to promote easy installments options to help in more sales

The proportion of debit card payments is increasing whereas credit card and UPI payments decreased slightly in 2018 -- so check if promotional offers from debit card service providers has helped this change? and how more UPI payments could be achieved

AL MA SE ES BA -- have lowest diff between estimated and actual delivery --- check the reasons for the longer delivery time or check if the prediction/estimation is inaccurate
-- for these states the freight value is high - so it would help to see the root causes for higher freight values

RR AP AM AL PA -- have highest delivery time - who also have high freight values - so check for reasons behind the higher transportation/logistics cost
-- look for the ways to optimize logistics of delivery cycles

Total payment value has increased by 20% in 2018 compared to 2017 -- the order volume is increased by same percentage
-- which indicates the prices/expenses etc are still in proportion

SP RS MG RJ PR - are the top 5 states accounting for majority of the orders and other states have very less contribution to the sales --
-- look for the ways to improve the market penetration in the remaining states to increase the overall sales

==========