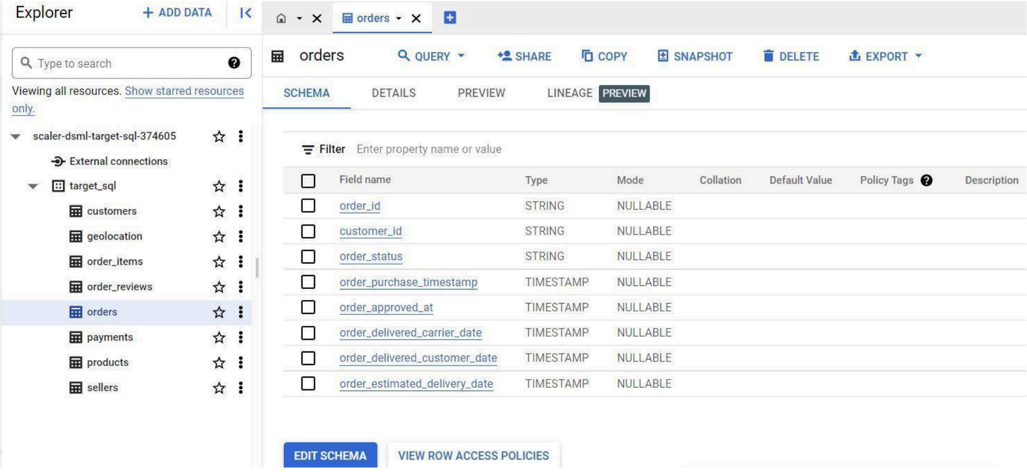
TARGET SQL

By MishelJoy

1. Exploratory analysis steps like checking the structure & characteristics of the dataset

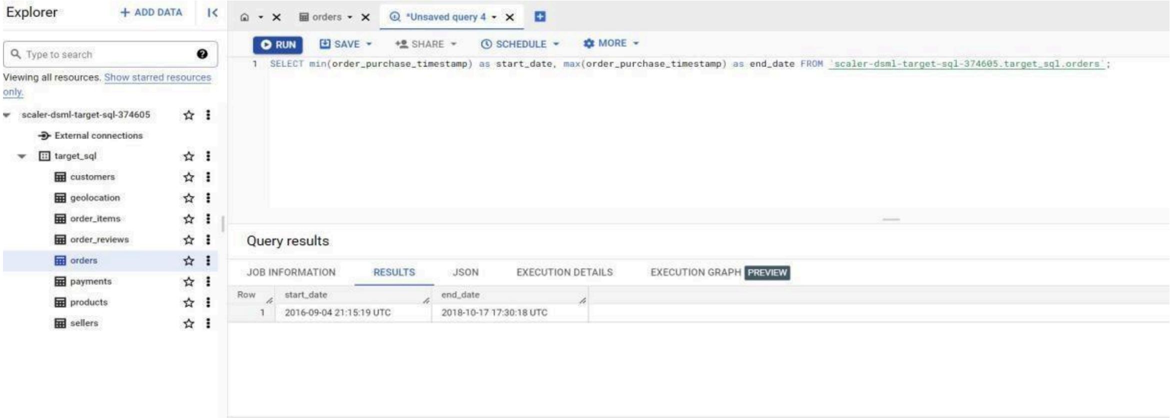
1.1 The data type of columns in a table



	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				

1.2 Period for which the data is given

```
SELECT min(order_purchase_timestamp) as start_date,
 max(order_purchase_timestamp) as end_date
FROM `scaler-dsml-target-sql-374605.target_sql.orders`
```



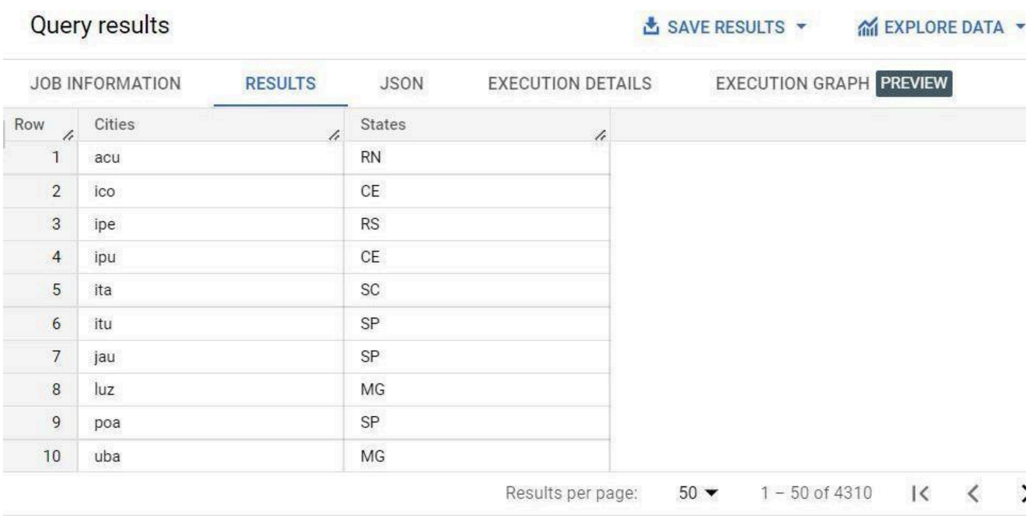
```
SELECT min(order_purchase_timestamp) as start_date, max(order_purchase_timestamp) as end_date FROM `scaler-dsml-target-sql-374605.target_sql.orders`;
```

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

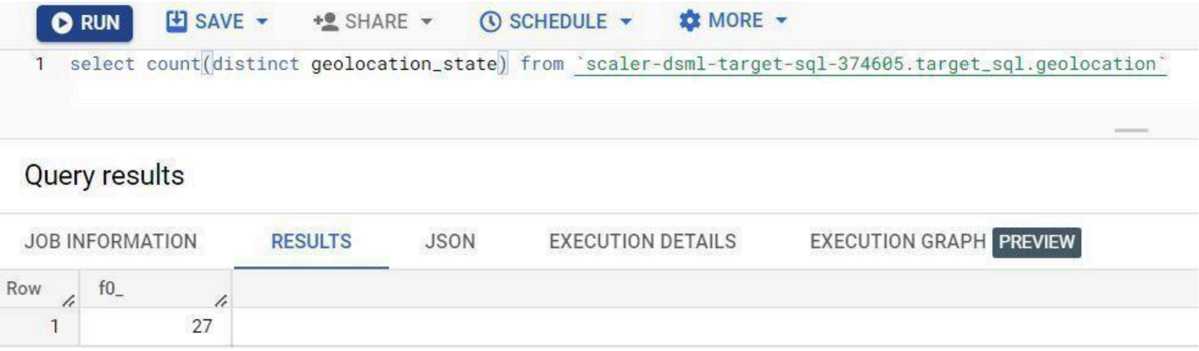
The given dataset is of the period ranging from September 2016 to October 2018

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

```
SELECT distinct customer_city as Cities,
          customer_state as States
FROM `scaler-dsml-target-sql-374605.target_sql.customers` as c
```

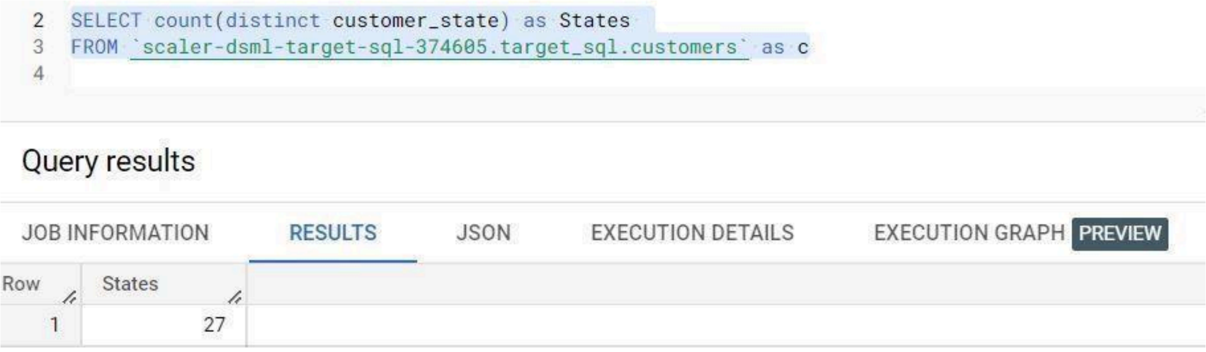


Row	Cities	States
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP
7	jau	SP
8	luz	MG
9	poa	SP
10	uba	MG



```
select count(distinct geolocation_state) from `scaler-dsml-target-sql-374605.target_sql.geolocation`
```

Row	f0_
1	27



```
SELECT count(distinct customer_state) as States
FROM `scaler-dsml-target-sql-374605.target_sql.customers` as c

```

Row	States
1	27

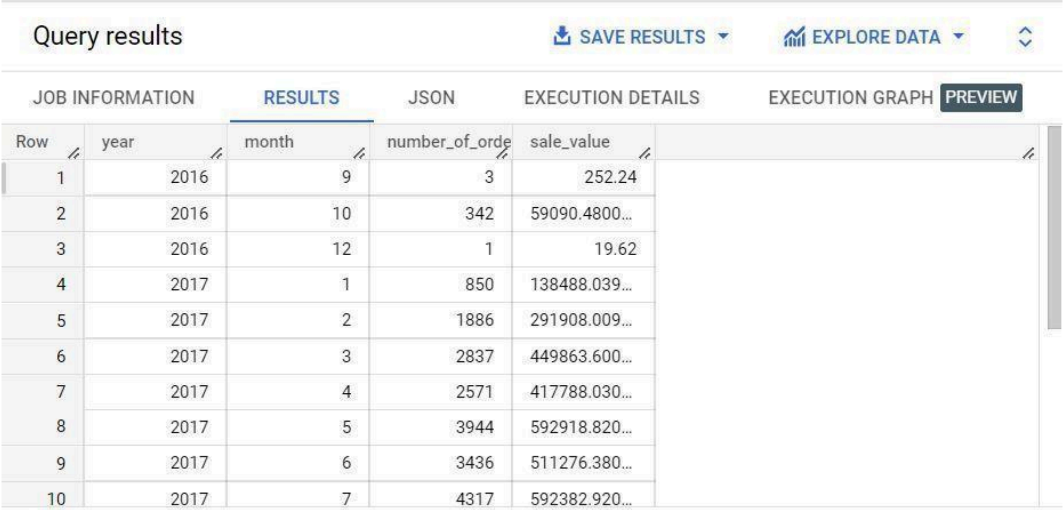
Here we are analyzing states from which the company has customers with the total number of states. Here customers are from all of the 27 states.

Customers from all of the 27 states of Brazil indicate well-established business

2. In-depth Exploration:

2.1 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?

```
SELECT
 extract(year from order_purchase_timestamp) as year,
 extract(month from order_purchase_timestamp) as month,
 count(o.order_id) as number_of_orders,
 sum(p.payment_value) as sale_value
FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o
inner join `target_sql.payments` as p
on o.order_id = p.order_id
group by 1,2
order by 1,2,3
```



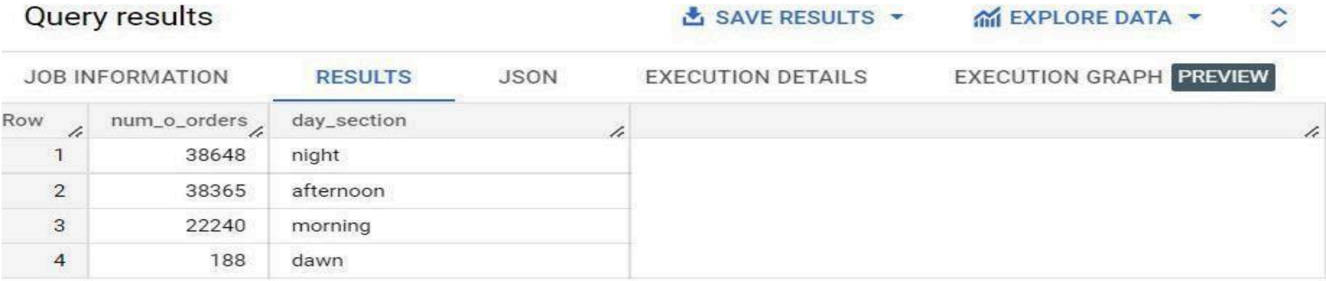
Row	year	month	number_of_orde	sale_value
1	2016	9	3	252.24
2	2016	10	342	59090.4800...
3	2016	12	1	19.62
4	2017	1	850	138488.039...
5	2017	2	1886	291908.009...
6	2017	3	2837	449863.600...
7	2017	4	2571	417788.030...
8	2017	5	3944	592918.820...
9	2017	6	3436	511276.380...
10	2017	7	4317	592382.920...

There has been considerable growth in the number of orders and thus in sale values over the years. However, a considerable dip in values is noticed in September and October of 2018. The maximum business happened in November 2017. There are no noticeable seasonality purchase trends observed. (the dataset is not adequate to have a conclusive opinion on seasonal trends)

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

```
with cte as
(SELECT order_id,
 case
   when time(order_purchase_timestamp)between '05:00:01' and '06:00:00' then 'dawn'
   when time(order_purchase_timestamp)between '06:00:01' and '12:00:00' then 'morning'
   when time(order_purchase_timestamp)between '12:00:01' and '18:00:00' then 'afternoon'
   else 'night'
   end as day_section
   FROM `scaler-dsml-target-sql-374605.target_sql.orders` )
 select
   count(order_id) num_o_orders,
   day_section
   from cte

   group by day_section
   order by num_o_orders desc
```



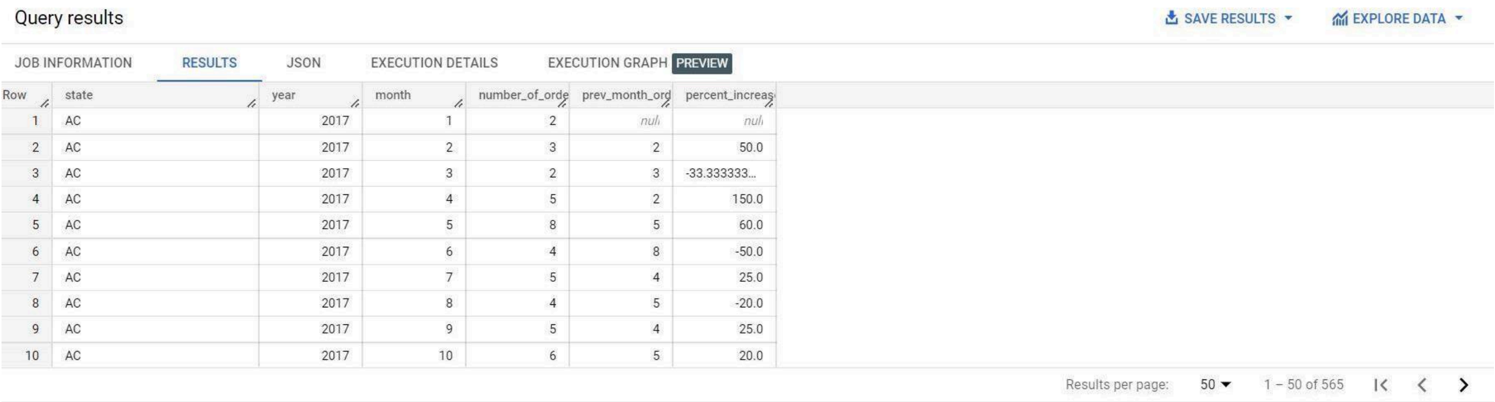
Row	num_o_orders	day_section
1	38648	night
2	38365	afternoon
3	22240	morning
4	188	dawn

Analysis: The purchases are made mostly in the afternoon and night. The purchases made in dawn are really low.
Recommendation: Can keep the website maintenance window in dawn where the number of purchases made is low compared to other windows.

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

3.1 Get month-on-month orders by states

```
with cte1 as(SELECT
 extract(year from order_purchase_timestamp) as year,
 extract(month from order_purchase_timestamp) as month,
 count(o.order_id) as number_of_orders,
 c.customer_state as state
FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o
inner join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id = c.customer_id
group by 1,2,4),
cte2 as(
select
 year,
 month,
 number_of_orders,
 state,
 lag(number_of_orders,1) over(partition by state order by year,month)as prev_month_orders
 from cte1
 order by year,month)
 select
 state,
 year,
 month,
 number_of_orders,
 prev_month_orders,
 ((number_of_orders-prev_month_orders )/prev_month_orders)*100 as percent_increase
 from cte2
 order by state,year, month
```



Query results

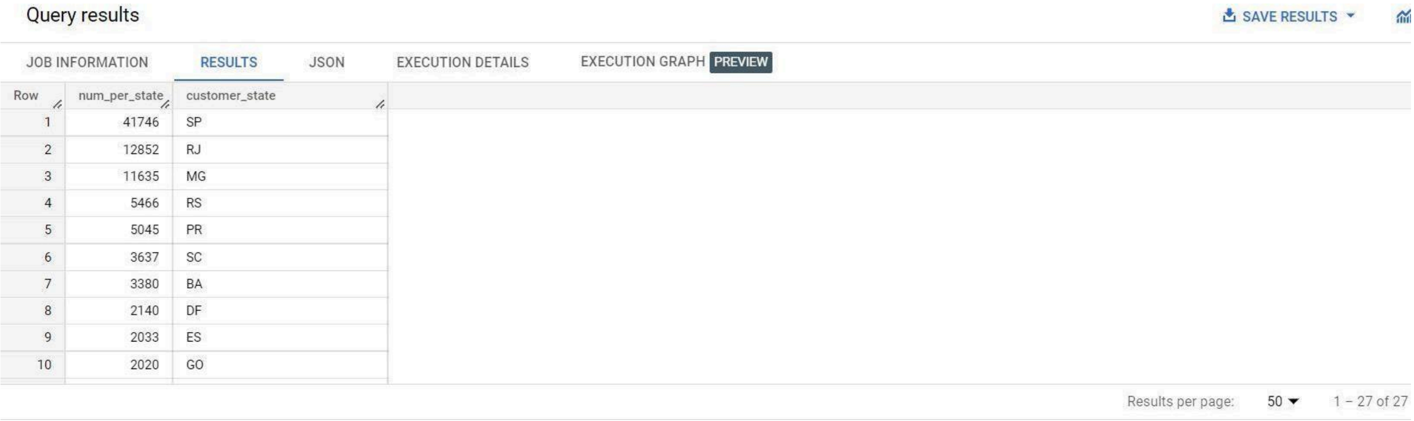
Row	state	year	month	number_of_orde	prev_month_ord	percent_increas
1	AC	2017	1	2	null	null
2	AC	2017	2	3	2	50.0
3	AC	2017	3	2	3	-33.333333...
4	AC	2017	4	5	2	150.0
5	AC	2017	5	8	5	60.0
6	AC	2017	6	4	8	-50.0
7	AC	2017	7	5	4	25.0
8	AC	2017	8	4	5	-20.0
9	AC	2017	9	5	4	25.0
10	AC	2017	10	6	5	20.0

Analysis: The state 'AC' follows a seasonal dip in the number of purchases in March and a hike in the number of purchases made in April.
Recommendation: the warehousing and stock accumulation can be based on these seasonal trends.

3.2 Distribution of customers across the states in Brazil

```
SELECT count(*) as num_per_state,customer_state
FROM`scaler-dsml-target-sql-374605.target_sql.customers` as c
inner join `scaler-dsml-target-sql-374605.target_sql.orders` as o
on c.customer_id = o.customer_id
group by customer_state
```

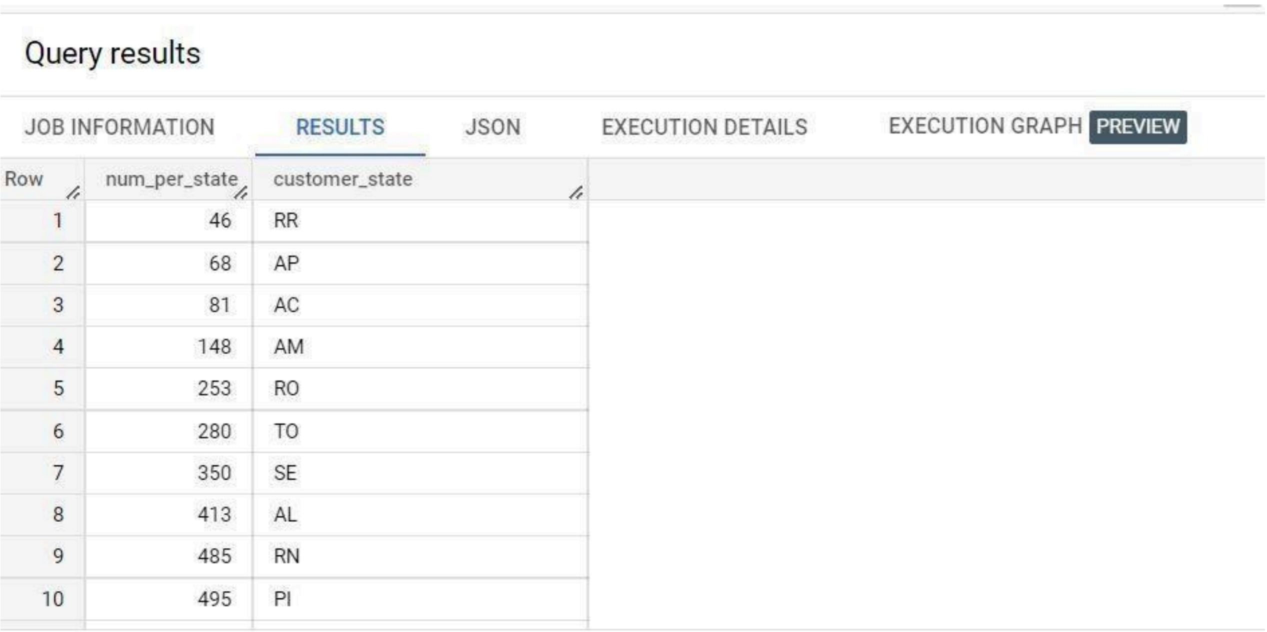
```
order by 1 desc
```



Row	num_per_state	customer_state
1	41746	SP
2	12852	RJ
3	11635	MG
4	5466	RS
5	5045	PR
6	3637	SC
7	3380	BA
8	2140	DF
9	2033	ES
10	2020	GO

Results per page: 50 1 – 27 of 27

```
SELECT count(*) as num_per_state,customer_state
FROM`scaler-dsml-target-sql-374605.target_sql.customers` as c
inner join `scaler-dsml-target-sql-374605.target_sql.orders` as o
on c.customer_id = o.customer_id
group by customer_state
order by 1 asc
```



Row	num_per_state	customer_state
1	46	RR
2	68	AP
3	81	AC
4	148	AM
5	253	RO
6	280	TO
7	350	SE
8	413	AL
9	485	RN
10	495	PI

This table gives the poor-performing states in terms of the number of customers

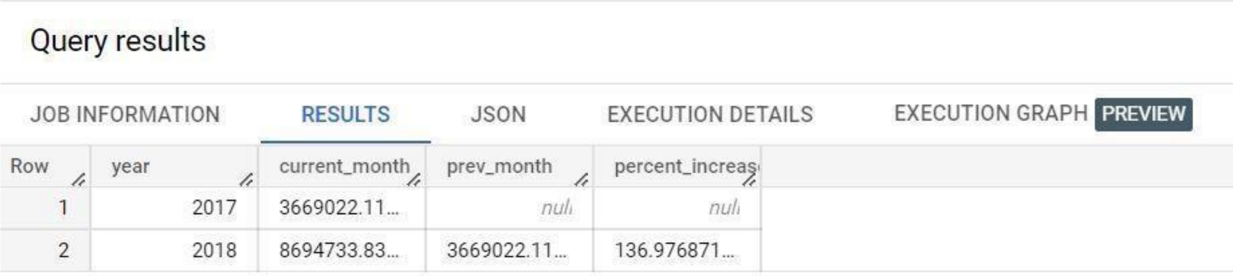
Analysis: The maximum number of customers are from the state ‘SP’.

Recommendation: promotional offers and ads to bring more customers from poor-performing states

4. Impact on the Economy: Analyse the money movement by e-commerce by looking at order prices, freight, and others.

4.1 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 cte1 as
(SELECT
extract(year from order_purchase_timestamp) as year,
sum(payment_value) as current_month
FROM `scaler-dsml-target-sql-374605.target_sql.payments`as p
inner join `scaler-dsml-target-sql-374605.target_sql.orders`as o
on p.order_id = o.order_id
where extract(year from order_purchase_timestamp) in(2017,2018) and
extract(month from order_purchase_timestamp) in(1,2,3,4,5,6,7,8)
group by 1
order by year),
cte2 as
(select
  *,
  lag(current_month,1)over(order by year)as prev_month
  from cte1
  order by year)
select
*,
((current_month-prev_month)/prev_month)*100 as percent_increase
from cte2
order by year
```

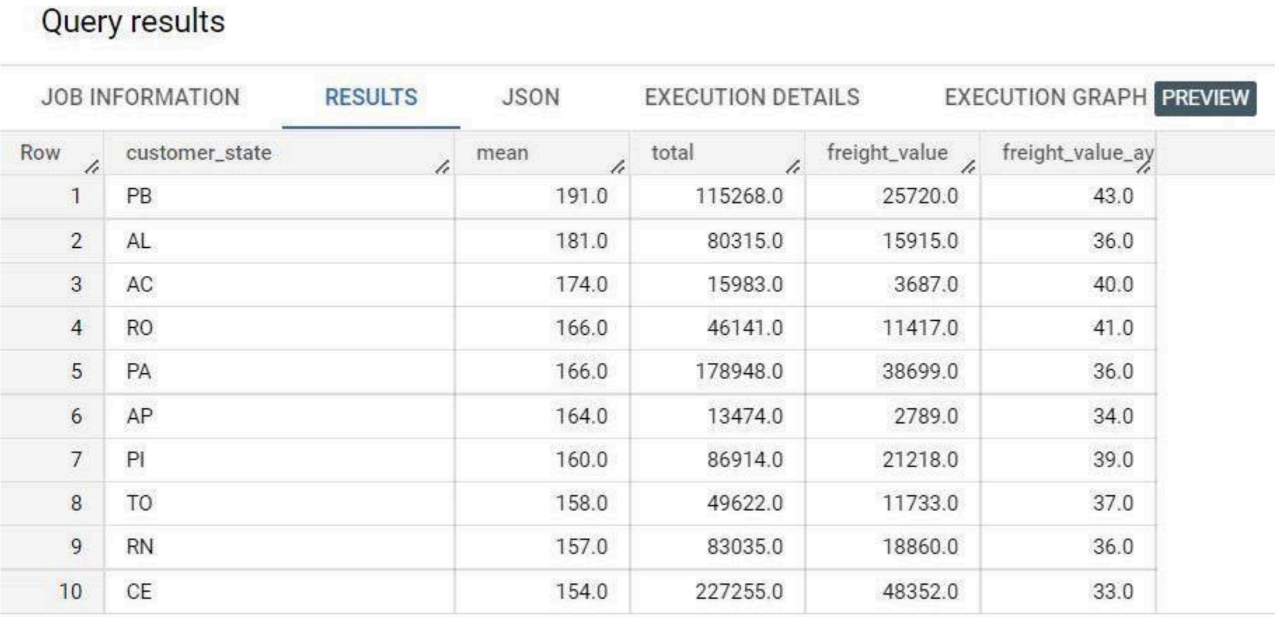


Row	year	current_month	prev_month	percent_increas
1	2017	3669022.11...	null	null
2	2018	8694733.83...	3669022.11...	136.976871...

Analysis: The increase in payment value from 2017 to 2018 is 136.98%.

4.2 Mean & Sum of price and freight value by customer state

```
SELECT
    customer_state,
    round(avg(price)) as mean,
    round(sum(price)) as total,
    round(sum(freight_value)) as freight_value,
    round(avg(freight_value)) as freight_value_avg
 FROM `scaler-dsml-target-sql-374605.target_sql.order_items` as ot
inner join `scaler-dsml-target-sql-374605.target_sql.orders` as o
on ot.order_id = o.order_id
join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
group by customer_state
order by 2 desc,5 desc
```



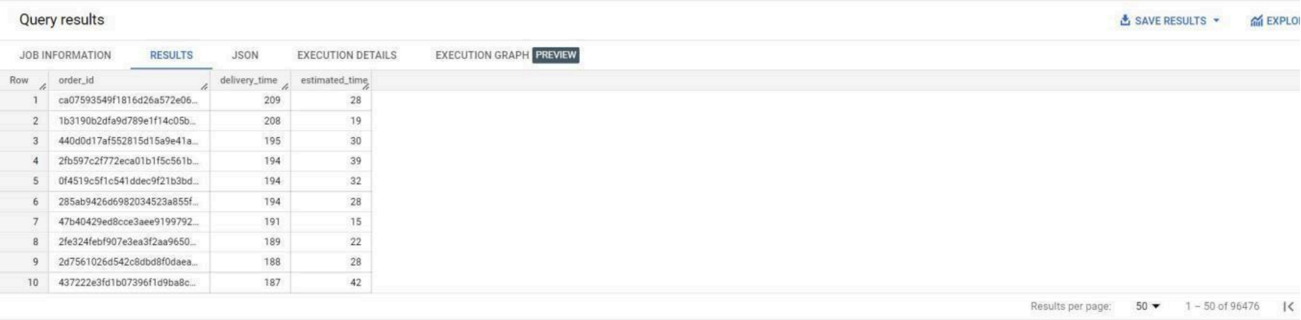
Row	customer_state	mean	total	freight_value	freight_value_av
1	PB	191.0	115268.0	25720.0	43.0
2	AL	181.0	80315.0	15915.0	36.0
3	AC	174.0	15983.0	3687.0	40.0
4	RO	166.0	46141.0	11417.0	41.0
5	PA	166.0	178948.0	38699.0	36.0
6	AP	164.0	13474.0	2789.0	34.0
7	PI	160.0	86914.0	21218.0	39.0
8	TO	158.0	49622.0	11733.0	37.0
9	RN	157.0	83035.0	18860.0	36.0
10	CE	154.0	227255.0	48352.0	33.0

The table provides information about states with the highest commodity prices and logistic expenses. Recommendations: warehousing facilities of these states can be improved, shipping on off-peak days and peak hours to reduce freight values, consolidating smaller shipments, and maintaining a constant shipment volume will also help reduce freight values.

5. Analysis on sales, freight and delivery time

5.1 Calculate days between purchasing, delivering and estimated delivery

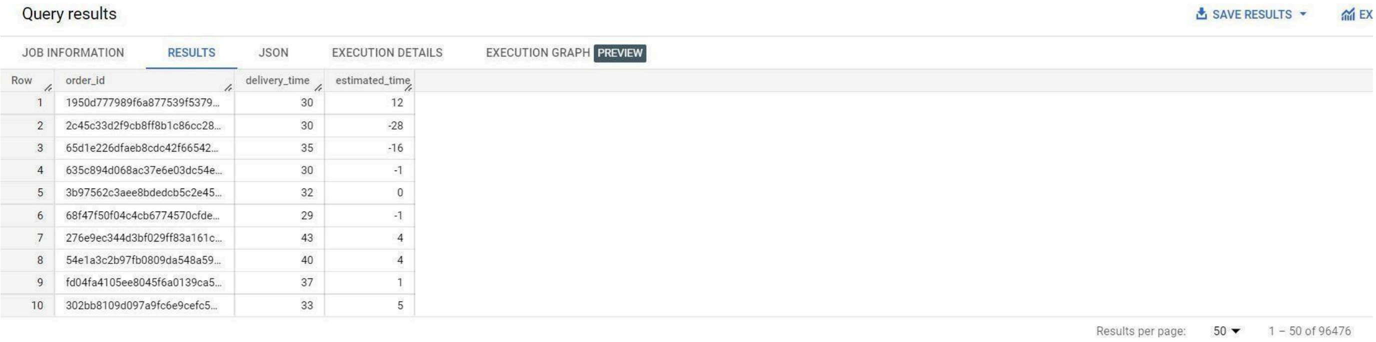
```
SELECT order_id,
date_diff(order_delivered_customer_date, order_purchase_timestamp, day) as delivery_time,
date_diff(order_estimated_delivery_date,order_purchase_timestamp, day) as estimated_time
 FROM `scaler-dsml-target-sql-374605.target_sql.orders`
 where order_delivered_customer_date is not null
 order by 2 desc, 3 desc
```



Row	order_id	delivery_time	estimated_time
1	ca07593549f1816d26a572e06...	209	28
2	1b3190b2dfa9d789e1f14c05b...	208	19
3	440d0d17af552815d15a9e41a...	195	30
4	2fb597c2f772eca01b1f5c561b...	194	39
5	0f4519c5f1c541ddec9f21b3bd...	194	32
6	285ab9426d6982034523a855f...	194	28
7	47b40429ed8cce3aee9199792...	191	15
8	2fe324febf907e3ea3f2aa9650...	189	22
9	2d7561026d542c8dbd8f0daea...	188	28
10	437222e3fd1b07396f1d9ba8c...	187	42

5.2 Find time_to_delivery & diff_estimated_delivery. The formula for the same is given below: time_to_delivery = order_purchase_timestamp-order_delivered_customer_date diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

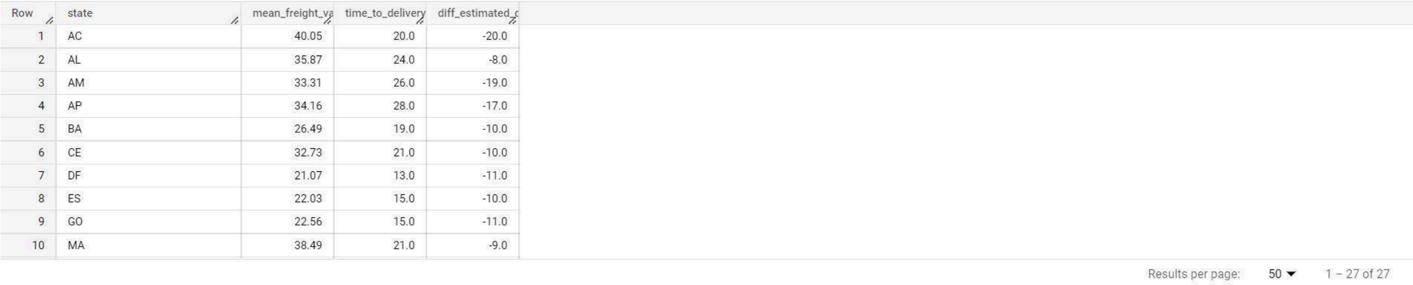
```
SELECT order_id,
date_diff(order_delivered_customer_date, order_purchase_timestamp, day) as delivery_time,
date_diff(order_delivered_customer_date,order_estimated_delivery_date, day) as estimated_time
 FROM `scaler-dsml-target-sql-374605.target_sql.orders`
 where order_delivered_customer_date is not null
```



Row	order_id	delivery_time	estimated_time
1	1950d777989f6a877539f5379...	30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	-28
3	65d1e226dfaeb8cdc42f66542...	35	-16
4	635c894d068ac37e6e03dc54e...	30	-1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	-1
7	276e9ec344d3bf029ff83a161c...	43	4
8	54e1a3c2b97fb0809da548a59...	40	4
9	fd04fa4105ee8045f6a0139ca5...	37	1
10	302bb8109d097a9fc6e9cefc5...	33	5

5.3) Group data by state, take the mean of freight_value, time_to_delivery, diff_estimated_delivery

```
SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) a s
time_to_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day)
)) as diff_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 1
```



Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	AC	40.05	20.0	-20.0
2	AL	35.87	24.0	-8.0
3	AM	33.31	26.0	-19.0
4	AP	34.16	28.0	-17.0
5	BA	26.49	19.0	-10.0
6	CE	32.73	21.0	-10.0
7	DF	21.07	13.0	-11.0
8	ES	22.03	15.0	-10.0
9	GO	22.56	15.0	-11.0
10	MA	38.49	21.0	-9.0

Results per page: 50 ▾ 1 – 27 of 27

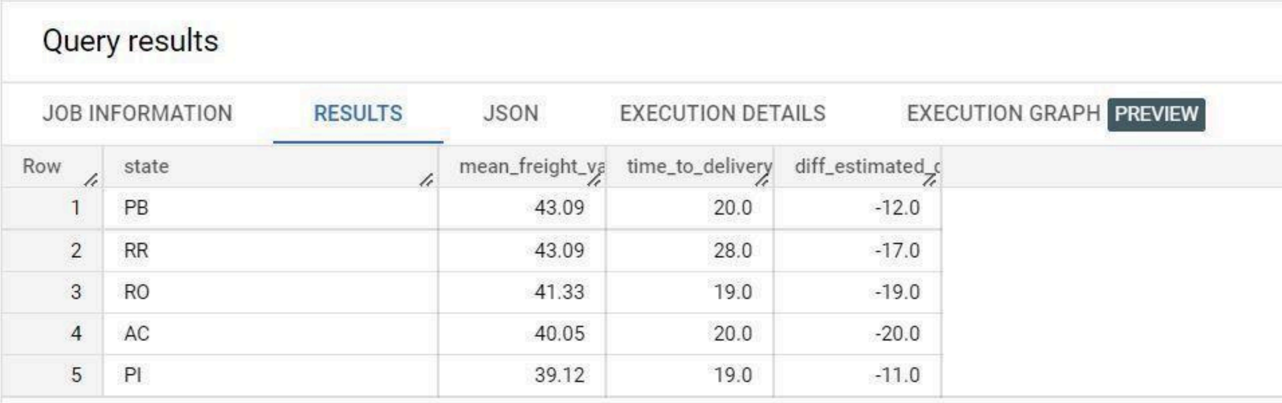
Analysis: Higher diff_estimated_delivery indicates poor delivery services and here all are negative values indicating on average deliveries are done before the estimated time. The higher freight value reduces the profit for the company.

5.4) Sort the data to get the following:

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

Highest

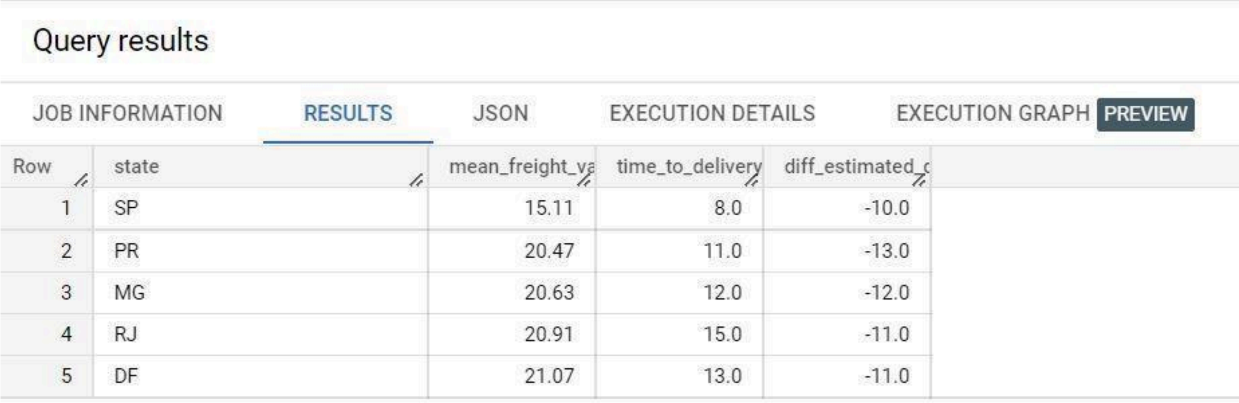
```
5.5     SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) a s
time_to_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day)
)) as diff_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 2 desc
limit 5
```



Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	PB	43.09	20.0	-12.0
2	RR	43.09	28.0	-17.0
3	RO	41.33	19.0	-19.0
4	AC	40.05	20.0	-20.0
5	PI	39.12	19.0	-11.0

Analysis: The highest average freight value is for the state PB

```
5.6.b)SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) a s
time_to_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day)
)) as diff_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 2 asc
limit 5
```



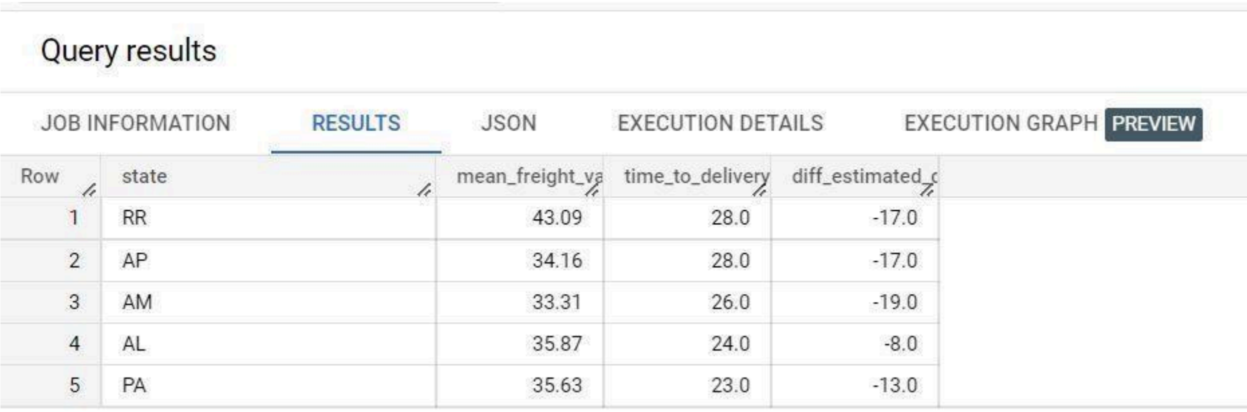
Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	SP	15.11	8.0	-10.0
2	PR	20.47	11.0	-13.0
3	MG	20.63	12.0	-12.0
4	RJ	20.91	15.0	-11.0
5	DF	21.07	13.0	-11.0

Analysis: The

lowest average freight value is for the state SP which has the higher number of customers

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

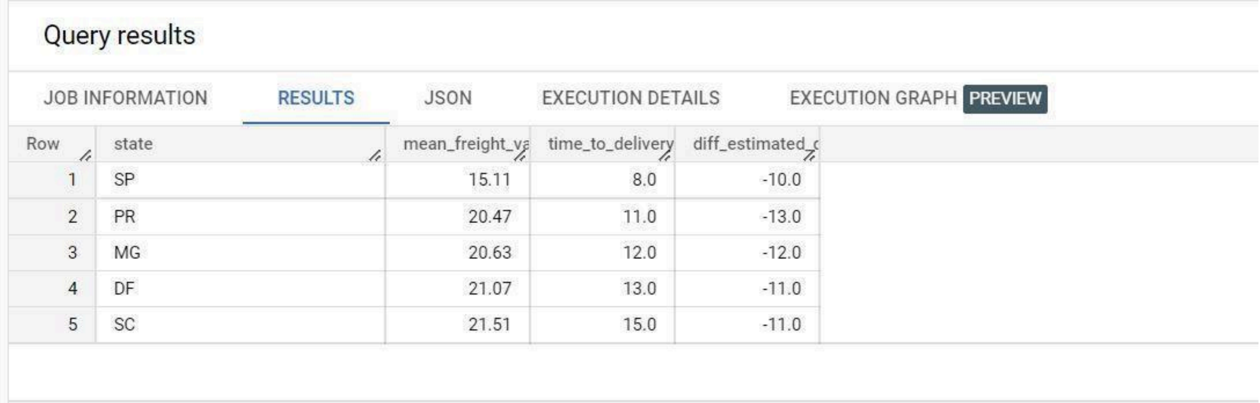
```
5.6.a) SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) a s
time_to_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day)
)) as diff_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 3 desc
limit 5
```



Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	RR	43.09	28.0	-17.0
2	AP	34.16	28.0	-17.0
3	AM	33.31	26.0	-19.0
4	AL	35.87	24.0	-8.0
5	PA	35.63	23.0	-13.0

Analysis: The highest average time to delivery is for the state RR, the freight value is also high for these states

```
5.6.b) SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) a s
time_to_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day)
)) as diff_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 3 asc
limit 5
```

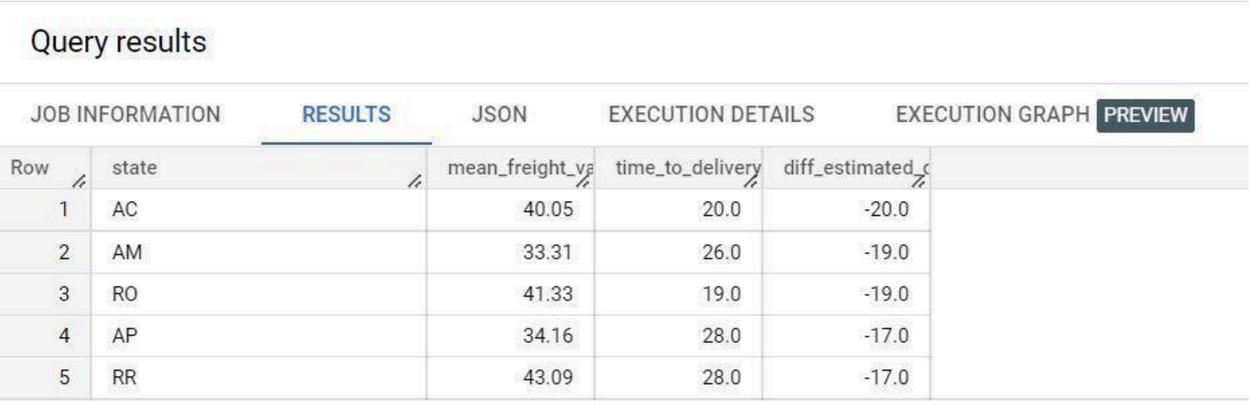


Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	SP	15.11	8.0	-10.0
2	PR	20.47	11.0	-13.0
3	MG	20.63	12.0	-12.0
4	DF	21.07	13.0	-11.0
5	SC	21.51	15.0	-11.0

Analysis: The lowest average time to delivery is for the state SP, the freight value is also low for these states, and these states have a higher number of customers.

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

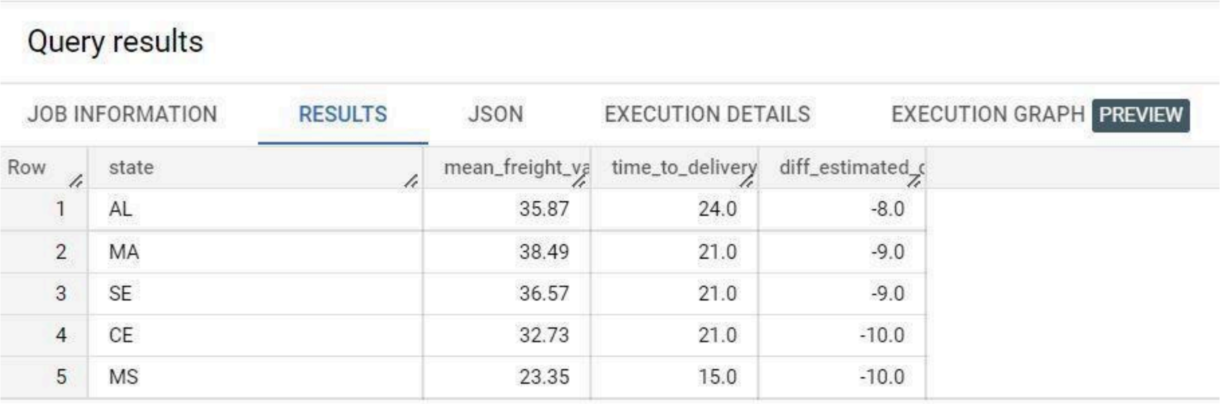
```
5.7.a) SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) a s
time_to_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day)
)) as diff_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 4 asc
limit 5
```



Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	AC	40.05	20.0	-20.0
2	AM	33.31	26.0	-19.0
3	RO	41.33	19.0	-19.0
4	AP	34.16	28.0	-17.0
5	RR	43.09	28.0	-17.0

Analysis: The fastest delivery is for the state AC

```
5.7.b) SELECT
customer_state as state,
round(avg(freight_value),2) as mean_freight_value,
round(avg(date_diff( order_delivered_customer_date,order_purchase_timestamp, day))) as time_t
o_delivery,
round(avg(date_diff( order_delivered_customer_date, order_estimated_delivery_date, day))) as diff
_estimated_delivery
 FROM `scaler-dsml-target-sql-374605.target_sql.orders` as o join
 `scaler-dsml-target-sql-374605.target_sql.order_items` as ot on
 ot.order_id = o.order_id
 join `scaler-dsml-target-sql-374605.target_sql.customers` as c
on o.customer_id= c.customer_id
 where order_delivered_customer_date is not null
group by 1
order by 4 desc
limit 5
```



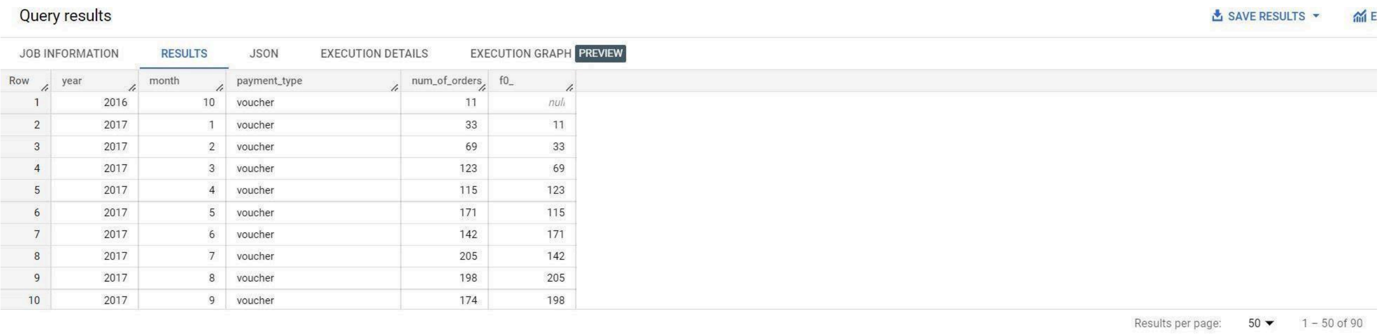
Row	state	mean_freight_va	time_to_delivery	diff_estimated_c
1	AL	35.87	24.0	-8.0
2	MA	38.49	21.0	-9.0
3	SE	36.57	21.0	-9.0
4	CE	32.73	21.0	-10.0
5	MS	23.35	15.0	-10.0

Analysis: The slowest delivery is for the state of AL

6. Payment type analysis:

6.1 Month over Month count of orders for different payment types

```
with cte1 as
(SELECT
  extract(year from order_purchase_timestamp) as year,
  extract(month from order_purchase_timestamp) as month,
  payment_type,
 count(distinct o.order_id) as num_of_orders
 FROM `scaler-dsml-target-sql-374605.target_sql.payments` as p
join `scaler-dsml-target-sql-374605.target_sql.orders` as o
on p.order_id = o.order_id
group by 1,2,3
order by 1,2,3
)
select
*,
lag(num_of_orders)over(partition by payment_type order by year,month) as prev_month
from cte1
```



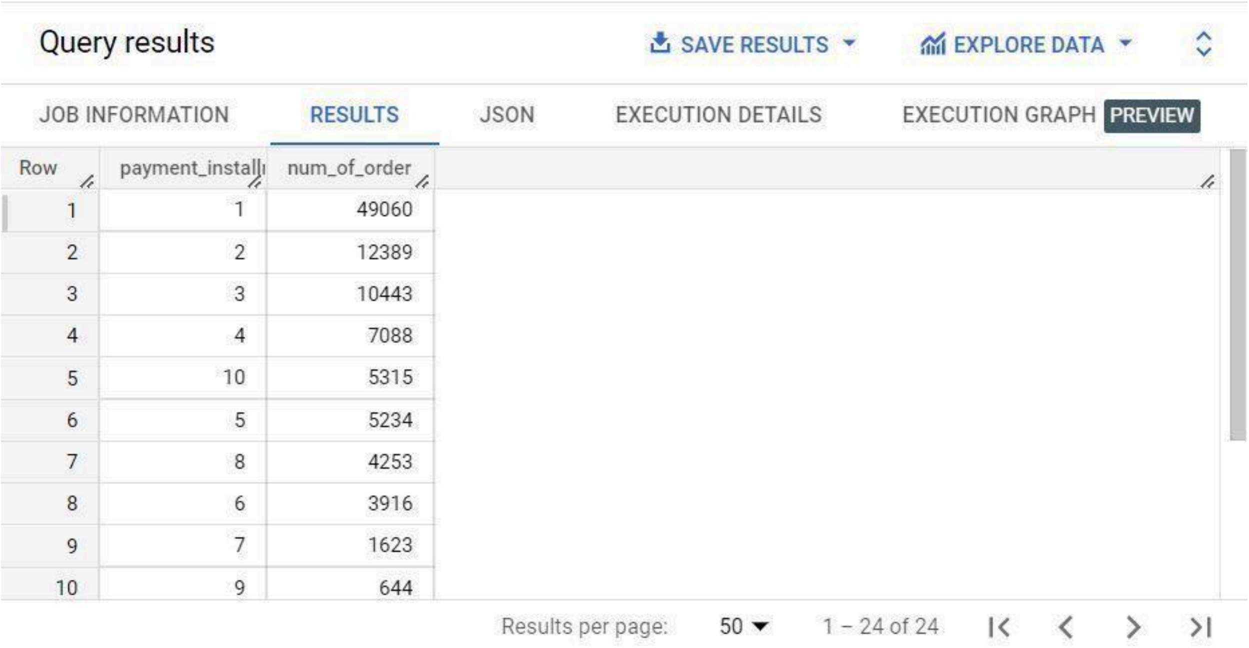
Row	year	month	payment_type	num_of_orders	f0_
1	2016	10	voucher	11	null
2	2017	1	voucher	33	11
3	2017	2	voucher	69	33
4	2017	3	voucher	123	69
5	2017	4	voucher	115	123
6	2017	5	voucher	171	115
7	2017	6	voucher	142	171
8	2017	7	voucher	205	142
9	2017	8	voucher	198	205
10	2017	9	voucher	174	198

Results per page: 50 1 – 50 of 90

Analysis: the most preferable payment mode is a credit card and the least favorable is a debit card

6.2 Count of orders based on the no. of payment instalments

```
SELECT
 payment_installments,
 count(distinct o.order_id) as num_of_order
 FROM `scaler-dsml-target-sql-374605.target_sql.payments` as p
join `scaler-dsml-target-sql-374605.target_sql.orders` as o
on p.order_id = o.order_id
group by 1
order by 2 desc
```



Row	payment_install	num_of_order
1	1	49060
2	2	12389
3	3	10443
4	4	7088
5	10	5315
6	5	5234
7	8	4253
8	6	3916
9	7	1623
10	9	644

Analysis: Most customers prefer single payment. A good amount of Customers also prefer installments from 1-10 installments.

Recommendations:

- There is a dip in the sales in the year 2018. The reason can be the non-availability of shopping offers due to the festive season, or lack of attractiveness of the offers. This is an area that can be worked on to boost sales.
- Almost 2/3rd of the customers are coming from 3 states. Target can focus on other states to attract more customers and boost sales.
- Very less people shop at late night, this is one area where Target can focus on improving sales during this time.
- The average difference between estimated vs delivered date ranges from 8-20 days. The variance can be improved to give a better experience to customers.
- There are states like RR, PB where freight is very high. these areas can be focused on cutting operation costs related to freight.
- The highest average time to deliver a product is 28 days which is very high. This can be worked upon to cut delivery time to make customers more satisfied.

- Most credit card payments have single installment, this information can be used to cross-sell more products to people who use credit cards.