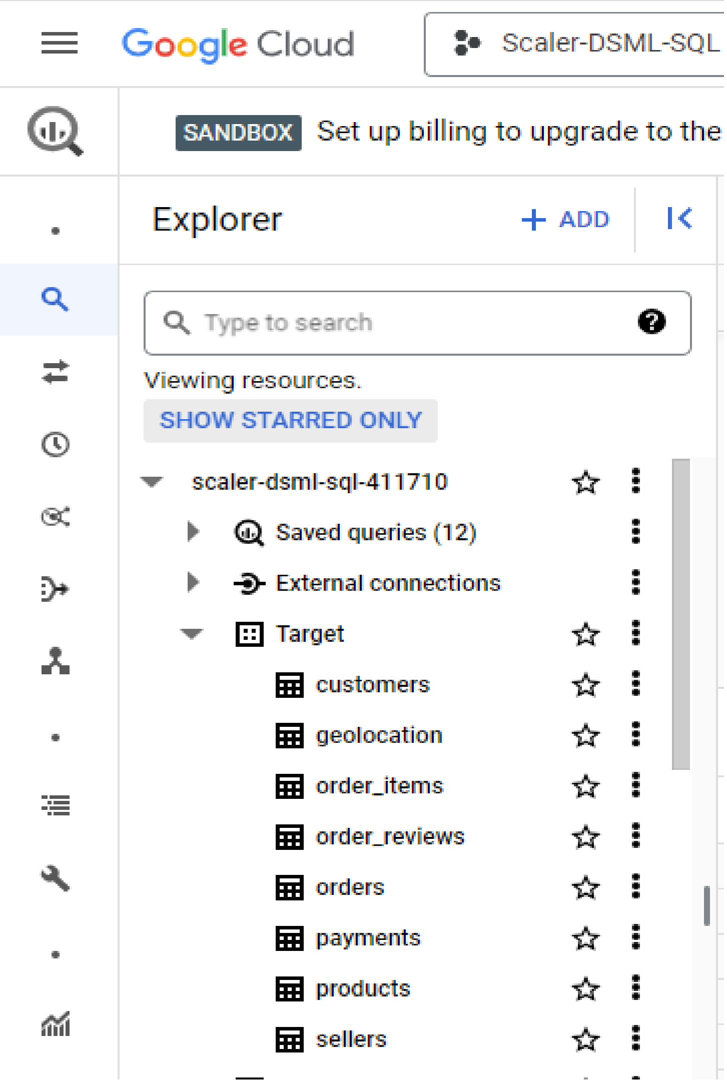
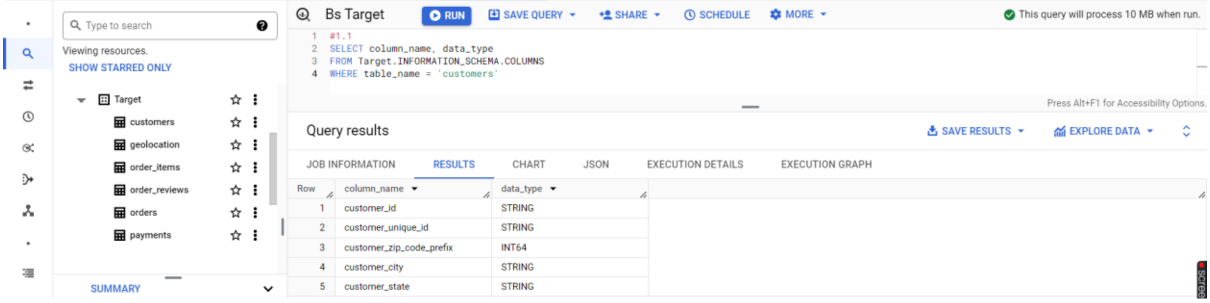
Business Case: Target SQL

1.

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



1.1 Data type of all columns in the "customers" table.



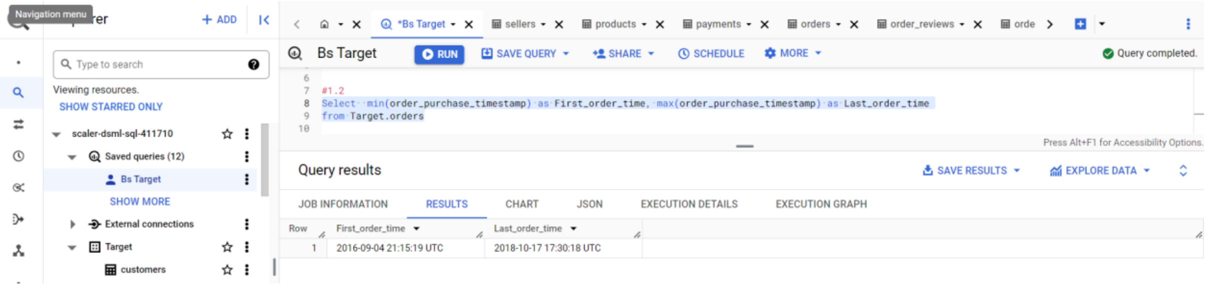
```sql
#1.1
SELECT column_name, data_type
FROM Target.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers'
```

Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Insights: From the above query we come to know datatypes of all the columns in the customers table of Target dataset.

Recommendations : NA

1.2 Get the time range between which the orders were placed.

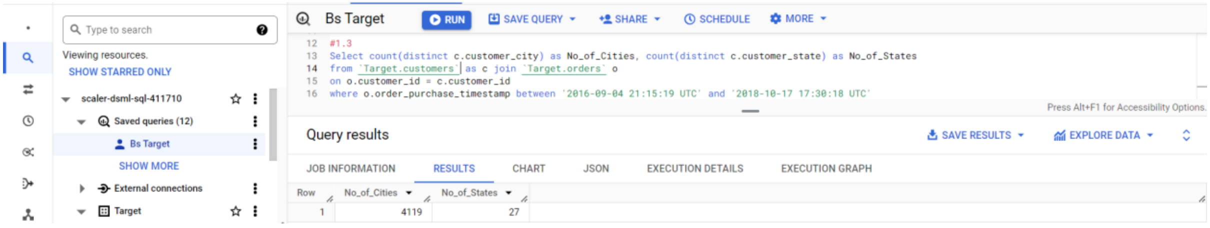


```
#1.2
Select  min(order_purchase_timestamp) as First_order_time, max(order_purchase_timestamp) as Last_order_time
from Target.orders
```

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

Insights: Here, we can find the date of the first order and the last order that were placed in the given time period of the data collected.
Recommendations: NA

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

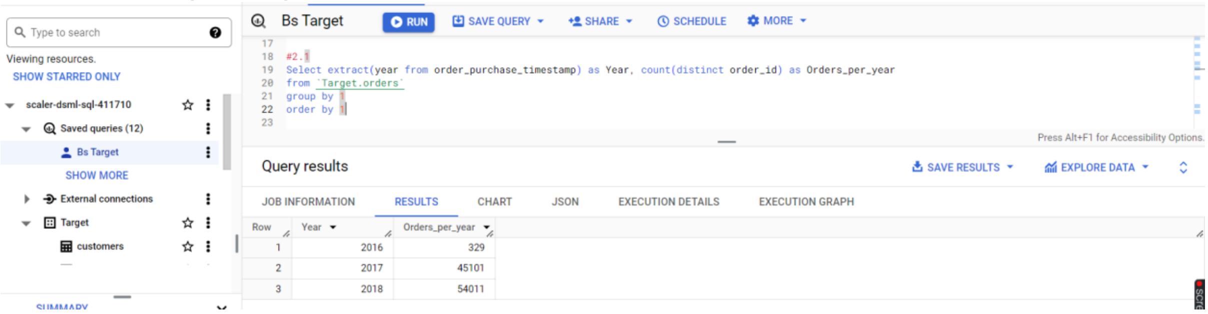


```
#1.3
Select count(distinct c.customer_city) as No_of_Cities, count(distinct c.customer_state) as No_of_States
from `Target.customers` as c join `Target.orders` o
on o.customer_id = c.customer_id
where o.order_purchase_timestamp between '2016-09-04 21:15:19 UTC' and '2018-10-17 17:30:18 UTC'
```

Row	No_of_Cities	No_of_States
1	4119	27

Insights: The above query gives us the count of number of states from which the Target store received its orders over the given time period.
Recommendations: Target can analyse the number of states they could cover across Brazil and plan the business strategies accordingly to spread their footprint across the rest of the Brazil.

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

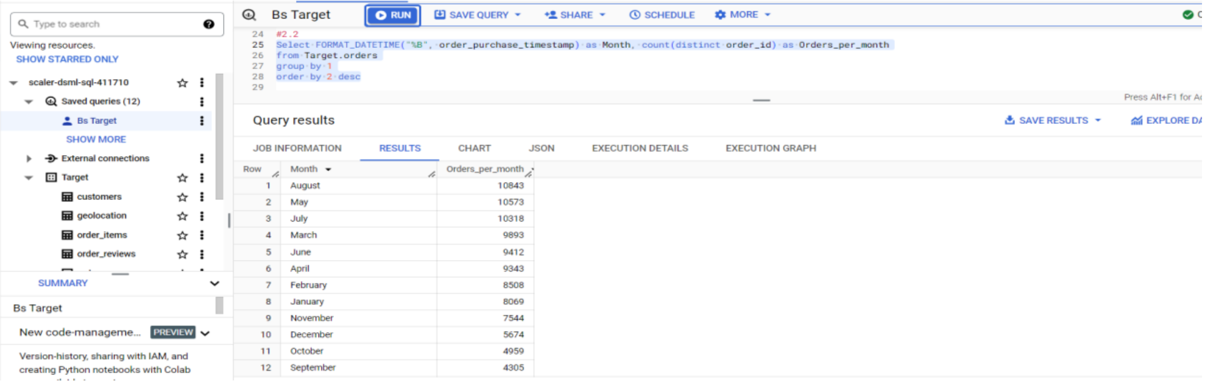


```
#2.1
Select extract(year from order_purchase_timestamp) as Year, count(distinct order_id) as Orders_per_year
from `Target.orders`
group by 1
order by 1
```

Row	Year	Orders_per_year
1	2016	329
2	2017	45101
3	2018	54011

Insights: We can see that the number of orders placed ifrom the year 2016 to 2017 has increased many-folds. And the increase was seen from 2017 to 2018 as well however not as much as seen from year 2016 to 2017.
Recommendations: Target can see what were the changes that led huge amount of increase in sales from 2016 to 2017 and can use the changes that were incorporated during 2017 across other countries where they not seeing good performance with respect to orders.

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



```
#2.2
Select FORMAT_DATETIME("%B", order_purchase_timestamp) as Month, count(distinct order_id) as Orders_per_month
from Target.orders
group by 1
order by 2 desc
```

Query results

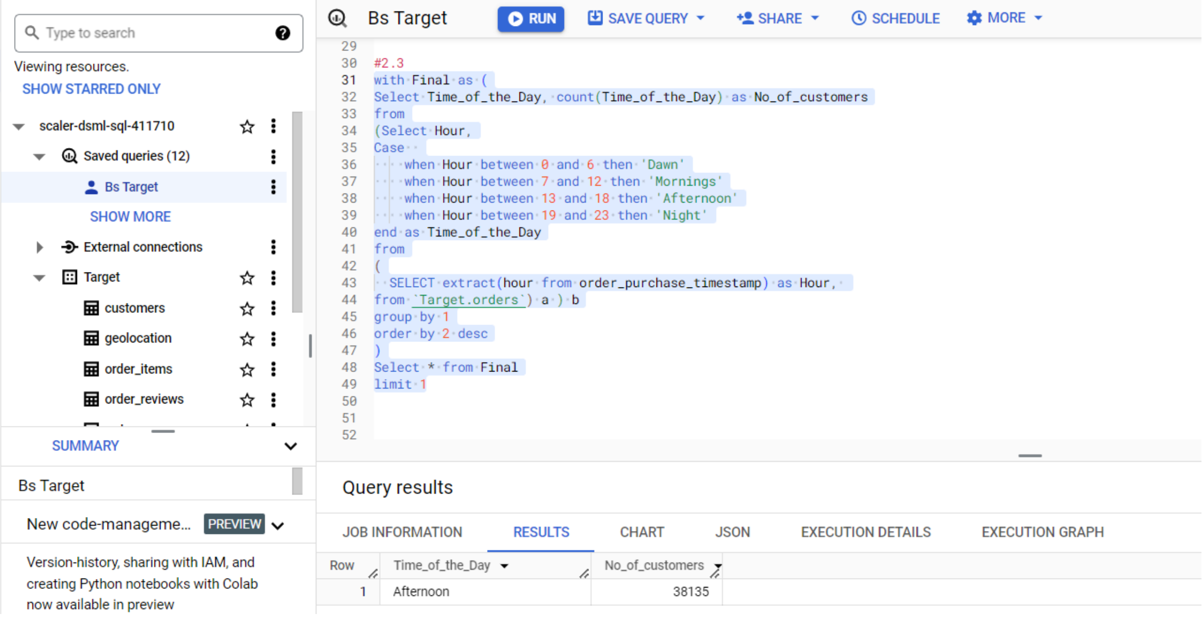
Row	Month	Orders_per_month
1	August	10843
2	May	10573
3	July	10318
4	March	9893
5	June	9412
6	April	9343
7	February	8508
8	January	8069
9	November	7544
10	December	5674
11	October	4959
12	September	4305

Insights: The above mentioned data gives us a clear understanding of number of orders place every month so that the business and understand the months on which they did not perform well and focus on getting better on sales for those months. For ex, August, May and July had orders above 10K and whereas December and September saw orders less than 5K.

Recommendations: Target can try to analyse the reasons behind the huge number of orders during Aug, May, Jul and see if they can use same insights to other months to increase monthly sales. And they can provide some discounts or offers during low performance months to boost their orders

2.3 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



```
#2.3
with Final as (
Select Time_of_the_Day, count(Time_of_the_Day) as No_of_customers
from
(Select Hour,
Case
    when Hour between 0 and 6 then 'Dawn'
    when Hour between 7 and 12 then 'Mornings'
    when Hour between 13 and 18 then 'Afternoon'
    when Hour between 19 and 23 then 'Night'
end as Time_of_the_Day
from
(
 SELECT extract(hour from order_purchase_timestamp) as Hour,
from `Target.orders`) a ) b
group by 1
order by 2 desc
)
Select * from Final
limit 1
```

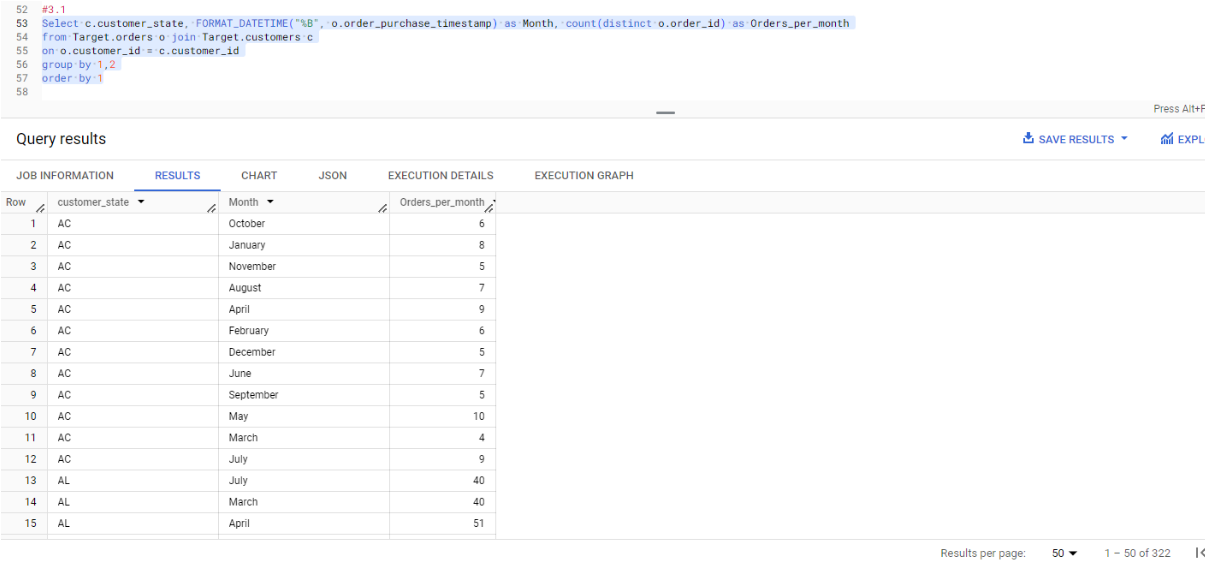
Query results

Row	Time_of_the_Day	No_of_customers
1	Afternoon	38135

Insights: The above-mentioned query and its output give us an understanding of which part of the day do most customers place the orders. As per the result it's the afternoon i.e. 13:00 to 18:00 every day.

Recommendations: Target expect huge number of orders during Afternoons as per the above insight and can plan alignment of staff and other technical facilities and software requirements(ex: billing, inventory Db etc) are available for use and can handle the orders during afternoon without having any difficulties.

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



```
#3.1
Select c.customer_state, FORMAT_DATETIME("%B", o.order_purchase_timestamp) as Month, count(distinct o.order_id) as Orders_per_month
from Target.orders o join Target.customers c
on o.customer_id = c.customer_id
group by 1,2
order by 1
```

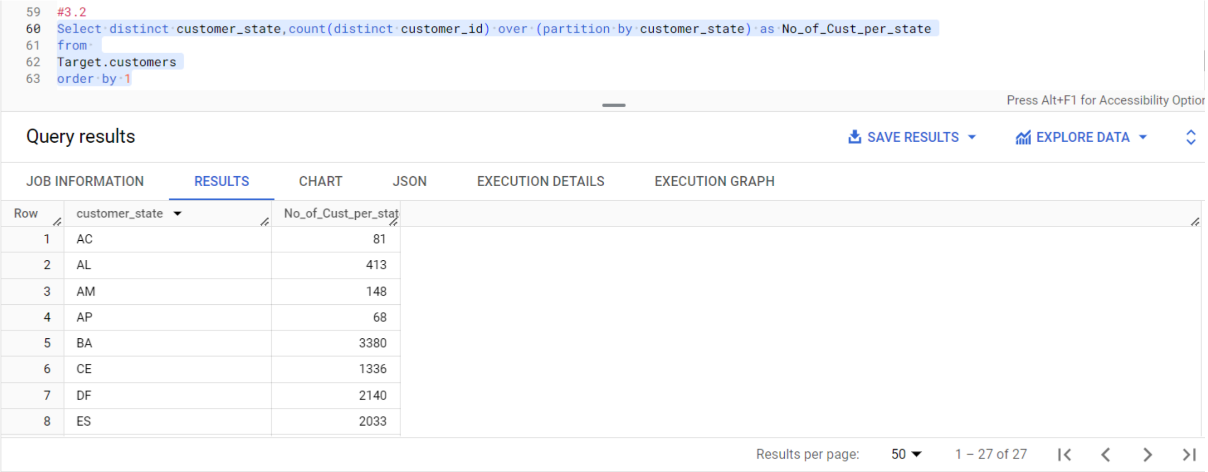
Query results

Row	customer_state	Month	Orders_per_month
1	AC	October	6
2	AC	January	8
3	AC	November	5
4	AC	August	7
5	AC	April	9
6	AC	February	6
7	AC	December	5
8	AC	June	7
9	AC	September	5
10	AC	May	10
11	AC	March	4
12	AC	July	9
13	AL	July	40
14	AL	March	40
15	AL	April	51

Insights: The above data gives the details of the state-wise orders distributed across 12 months of the year for each state.

Recommendations: Target can us the above data and see for which months would they need to more careful so that they have their inventory full enough to accommodate the orders state wise.

3.2 How are the customers distributed across all the states?



```
#3.2
Select distinct customer_state,count(distinct customer_id) over (partition by customer_state) as No_of_Cust_per_state
from
Target.customers
order by 1
```

Query results

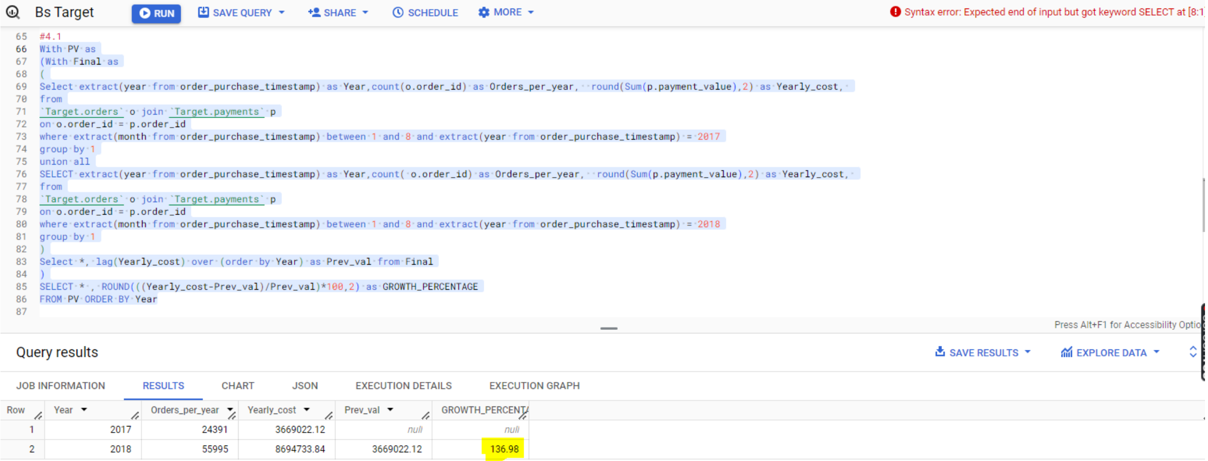
Row	customer_state	No_of_Cust_per_stat
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033

Insights : Th e above data gives us a brief idea of how customers are spread across the states over Brazil.

Recommendations: Target can use the above data to understand the good performing states and bad performing states and take measures to achieve a greater number of orders in poor performing states. And can make strategies to boost their business

4 . 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.



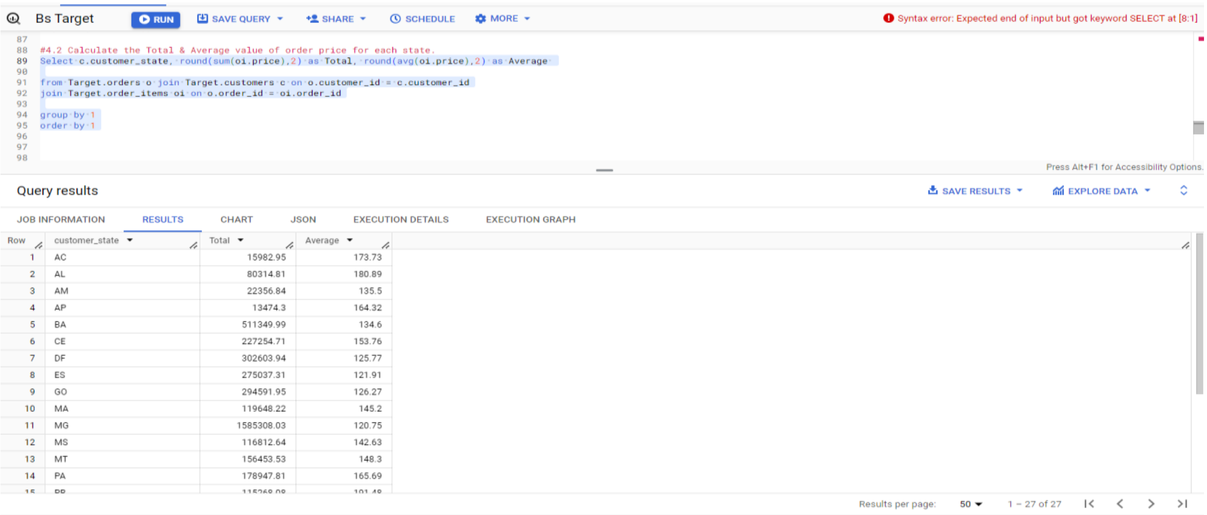
Syntax error: Expected end of input but got keyword SELECT at [8:1

```
#4.1
With PV as
(With Final as
(
Select extract(year from order_purchase_timestamp) as Year,count(o.order_id) as Orders_per_year,  round(Sum(p.payment_value),2) as Yearly_cost, 
from
`Target.orders` o join `Target.payments` p
on o.order_id = p.order_id
where extract(month from order_purchase_timestamp) between 1 and 8 and extract(year from order_purchase_timestamp) = 2017
group by 1
union all
SELECT extract(year from order_purchase_timestamp) as Year,count( o.order_id) as Orders_per_year,  round(Sum(p.payment_value),2) as Yearly_cost, 
from
`Target.orders` o join `Target.payments` p
on o.order_id = p.order_id
where extract(month from order_purchase_timestamp) between 1 and 8 and extract(year from order_purchase_timestamp) = 2018
group by 1
)
Select *, lag(Yearly_cost) over (order by Year) as Prev_val from Final
)
SELECT * , ROUND(((Yearly_cost-Prev_val)/Prev_val)*100,2) as GROWTH_PERCENTAGE
FROM PV ORDER BY Year
```

Row	Year	Orders_per_year	Yearly_cost	Prev_val	GROWTH_PERCENT
1	2017	24391	3669022.12	null	null
2	2018	55995	8694733.84	3669022.12	136.98

Insights: The above highlighted value in yellow gives us the growth percentage of the payment value form the 2017 to 2018 across the months Jan to Aug.
Recommendations: Target can use this data to plan the future year order numbers and can get an idea how much revenue they would be able to generate via payment value in the coming years.

4 . 2 Calculate the Total & Average value of order price for each state.



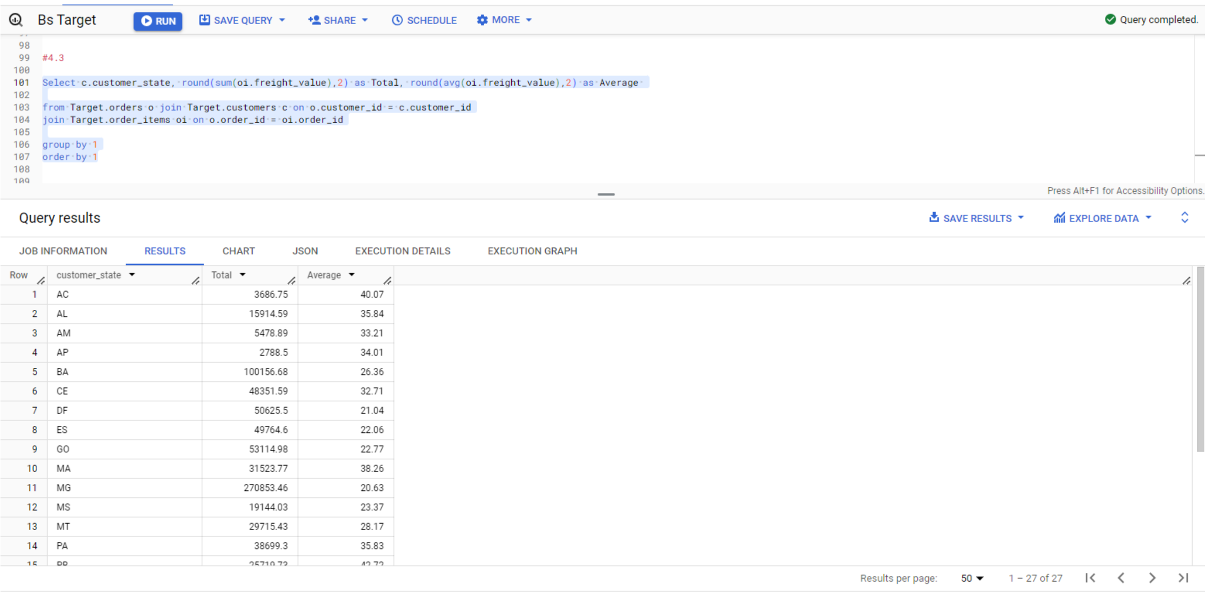
Syntax error: Expected end of input but got keyword SELECT at [8:1]

```
#4.2 Calculate the Total & Average value of order price for each state.
Select c.customer_state, round(sum(oi.price),2) as Total, round(avg(oi.price),2) as Average

from Target.orders o join Target.customers c on o.customer_id = c.customer_id
join Target.order_items oi on o.order_id = oi.order_id

group by 1
order by 1
```

Row	customer_state	Total	Average
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2
11	MG	1585308.03	120.75
12	MS	116812.64	142.63
13	MT	156453.53	148.3
14	PA	178947.81	165.69
15	PB	115268.08	191.48

Insights : The above mentioned data give us an understanding of Total and Average order price with respect to each state.
Recommendations: Target can use the above data to understand the good performing states and bad performing states and take measures to achieve a greater number of orders in poor performing states. And can make strategies to boost their business

4 . 3 Calculate the Total & Average value of order freight for each state.

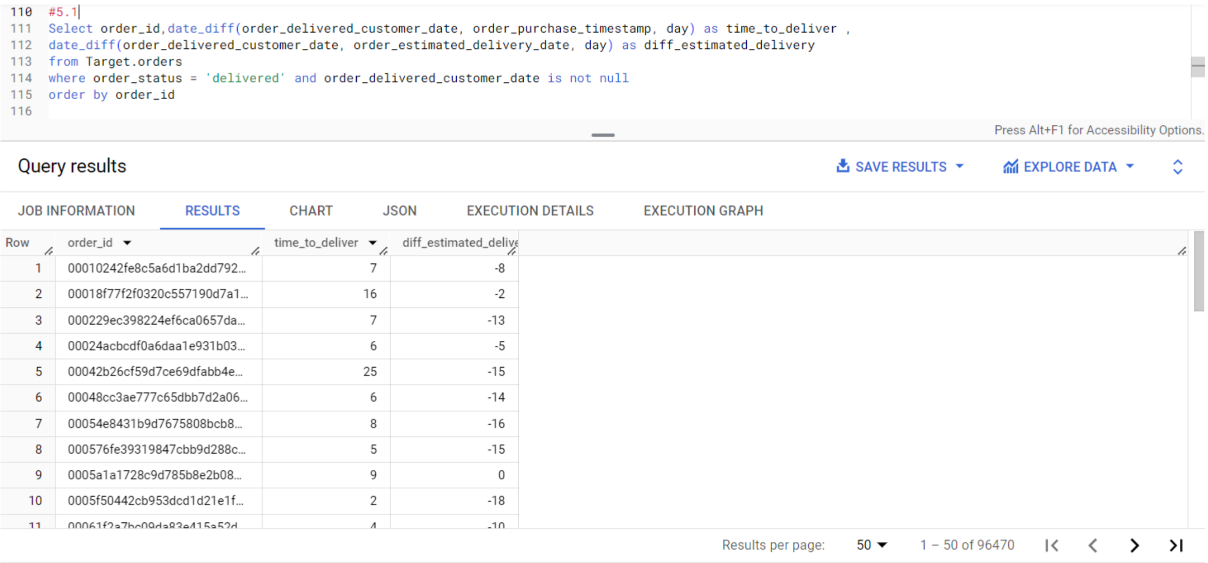


```
#4.3

Select c.customer_state, round(sum(oi.freight_value),2) as Total, round(avg(oi.freight_value),2) as Average

from Target.orders o join Target.customers c on o.customer_id = c.customer_id
join Target.order_items oi on o.order_id = oi.order_id

group by 1
order by 1
```

Row	customer_state	Total	Average
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26
11	MG	270853.46	20.63
12	MS	19144.03	23.37
13	MT	29715.43	28.17
14	PA	38699.3	35.83
15	PB	25719.73	42.72

Insights: The above mentioned data give us an understanding of Total and Average Freight value with respect to each state.

5.1 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.

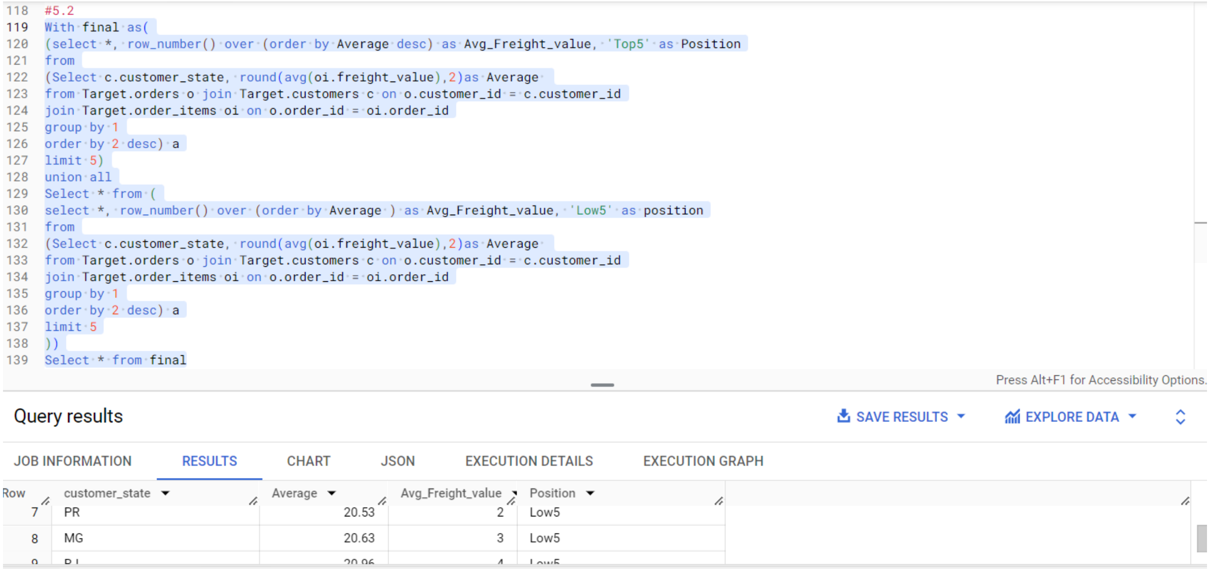


```
#5.1
Select order_id,date_diff(order_delivered_customer_date, order_purchase_timestamp, day) as time_to_deliver ,
date_diff(order_delivered_customer_date, order_estimated_delivery_date, day) as diff_estimated_delivery
from Target.orders
where order_status = 'delivered' and order_delivered_customer_date is not null
order by order_id
```

Row	order_id	time_to_deliver	diff_estimated_delive
1	00010242fe8c5a6d1ba2dd792...	7	-8
2	00018f77f2f0320c557190d7a1...	16	-2
3	000229ec398224ef6ca0657da...	7	-13
4	00024acbcdf0a6daa1e931b03...	6	-5
5	00042b26cf59d7ce69dfabb4e...	25	-15
6	00048cc3ae777c65dbb7d2a06...	6	-14
7	00054e8431b9d7675808bcb8...	8	-16
8	000576fe39319847cbb9d288c...	5	-15
9	0005a1a1728c9d785b8e2b08...	9	0
10	0005f50442cb953dcd1d21e1f...	2	-18
11	00061f2a7bc09da83e415a52d...	4	-10

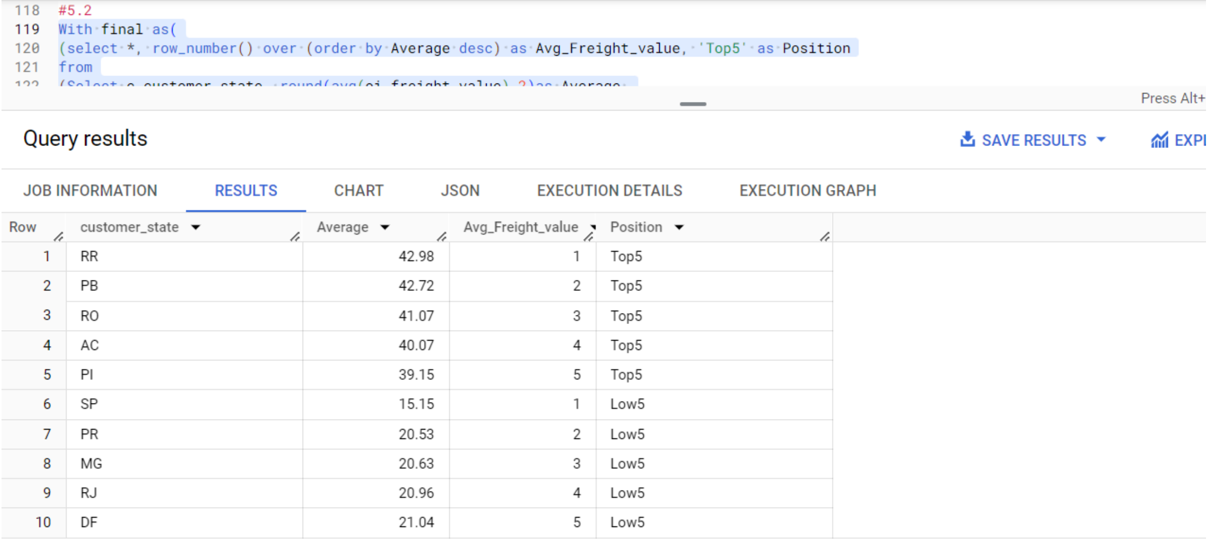
Insights: The above data shows Delivery time for each order and difference between estimated delivery date to actual order delivered date.
Recommendations: Target can use these metrics to analyse how delivery department and logistics dept are working and can use the data to optimize the loop holes.

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



```
#5.2
With final as(
(select *, row_number() over (order by Average desc) as Avg_Freight_value, 'Top5' as Position
from
(Select c.customer_state, round(avg(oi.freight_value),2)as Average
from Target.orders o join Target.customers c on o.customer_id = c.customer_id
join Target.order_items oi on o.order_id = oi.order_id
group by 1
order by 2 desc) a
limit 5)
union all
Select * from (
select *, row_number() over (order by Average ) as Avg_Freight_value, 'Low5' as position
from
(Select c.customer_state, round(avg(oi.freight_value),2)as Average
from Target.orders o join Target.customers c on o.customer_id = c.customer_id
join Target.order_items oi on o.order_id = oi.order_id
group by 1
order by 2 desc) a
limit 5
))
Select * from final
```

Row	customer_state	Average	Avg_Freight_value	Position
7	PR	20.53	2	Low5
8	MG	20.63	3	Low5
9	RJ	20.96	4	Low5



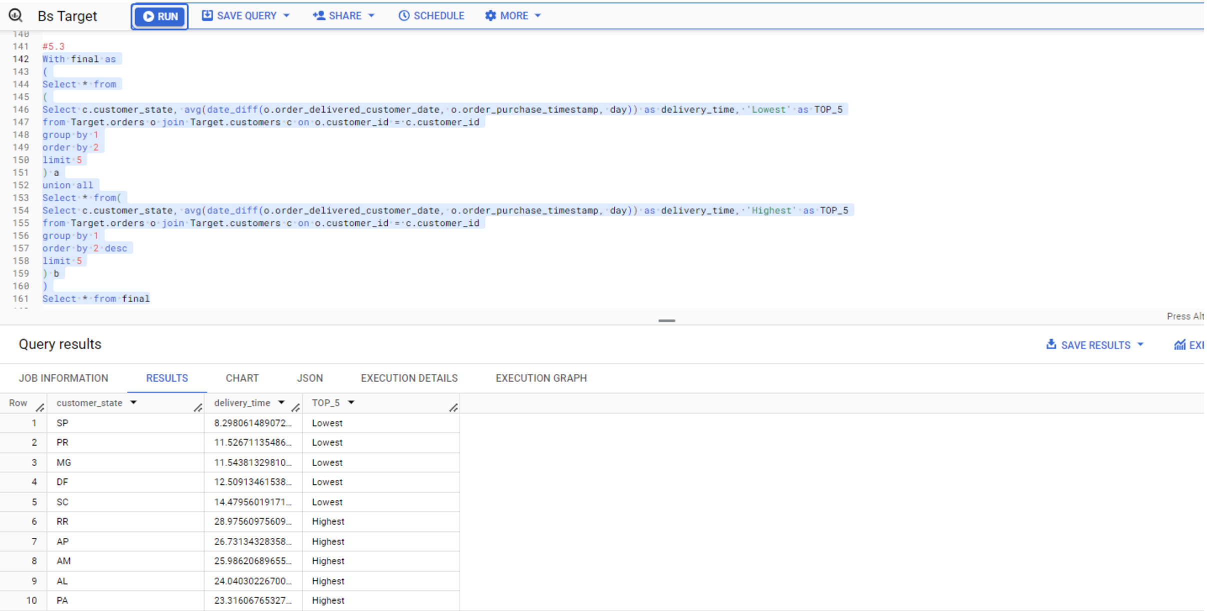
```
#5.2
With final as(
(select *, row_number() over (order by Average desc) as Avg_Freight_value, 'Top5' as Position
from
```

Row	customer_state	Average	Avg_Freight_value	Position
1	RR	42.98	1	Top5
2	PB	42.72	2	Top5
3	RO	41.07	3	Top5
4	AC	40.07	4	Top5
5	PI	39.15	5	Top5
6	SP	15.15	1	Low5
7	PR	20.53	2	Low5
8	MG	20.63	3	Low5
9	RJ	20.96	4	Low5
10	DF	21.04	5	Low5

Insights: The above data displays the top 5 states with the highest & lowest average freight value.

Recommendations: This data can be used to plan how to handle the freight value at the specific states to cut down operational costs.

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



```
#5.3
With final as
(
Select * from
(
Select c.customer_state, avg(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp, day)) as delivery_time, 'Lowest' as TOP_5
from Target.orders o join Target.customers c on o.customer_id = c.customer_id
group by 1
order by 2
limit 5
) a
union all
Select * from(
Select c.customer_state, avg(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp, day)) as delivery_time, 'Highest' as TOP_5
from Target.orders o join Target.customers c on o.customer_id = c.customer_id
group by 1
order by 2 desc
limit 5
) b
)
Select * from final
```

Query results

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

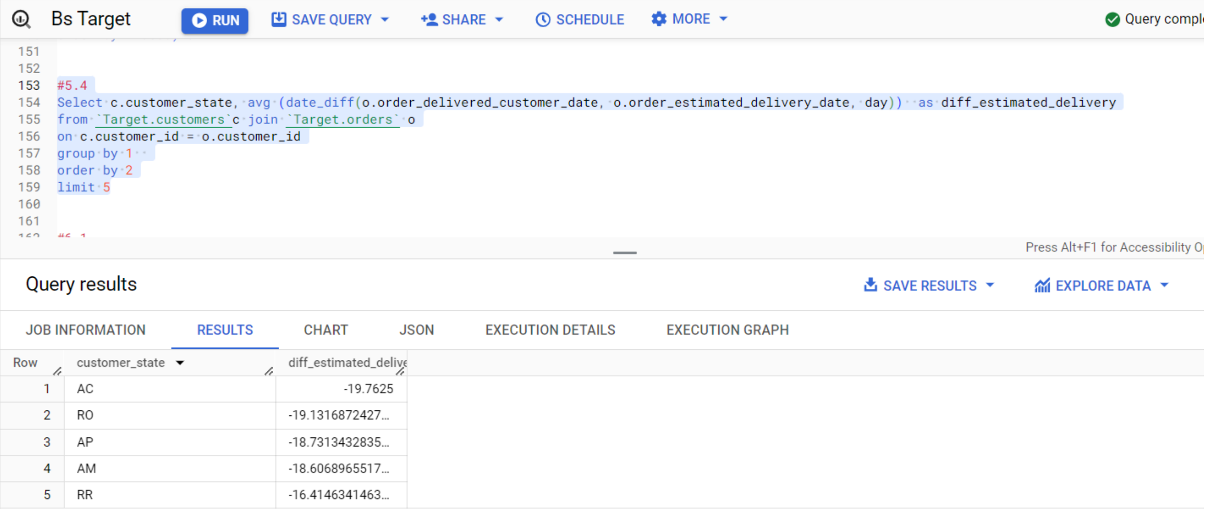
Row	customer_state	delivery_time	TOP_5
1	SP	8.298061489072...	Lowest
2	PR	11.52671135486...	Lowest
3	MG	11.54381329810...	Lowest
4	DF	12.50913461538...	Lowest
5	SC	14.47956019171...	Lowest
6	RR	28.97560975609...	Highest
7	AP	26.73134328358...	Highest
8	AM	25.98620689655...	Highest
9	AL	24.04030226700...	Highest
10	PA	23.31606765327...	Highest

Insights : The above data displays the top 5 states with the highest & lowest average delivery time.

Recommendations: Target can use these metrics to analyse how delivery department and logistics dept are working for these states and take measure to perform better order deliveries.

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.



```
#5.4
Select c.customer_state, avg (date_diff(o.order_delivered_customer_date, o.order_estimated_delivery_date, day))  as diff_estimated_delivery
from `Target.customers`c join `Target.orders` o
on c.customer_id = o.customer_id
group by 1
order by 2
limit 5
```

Query results

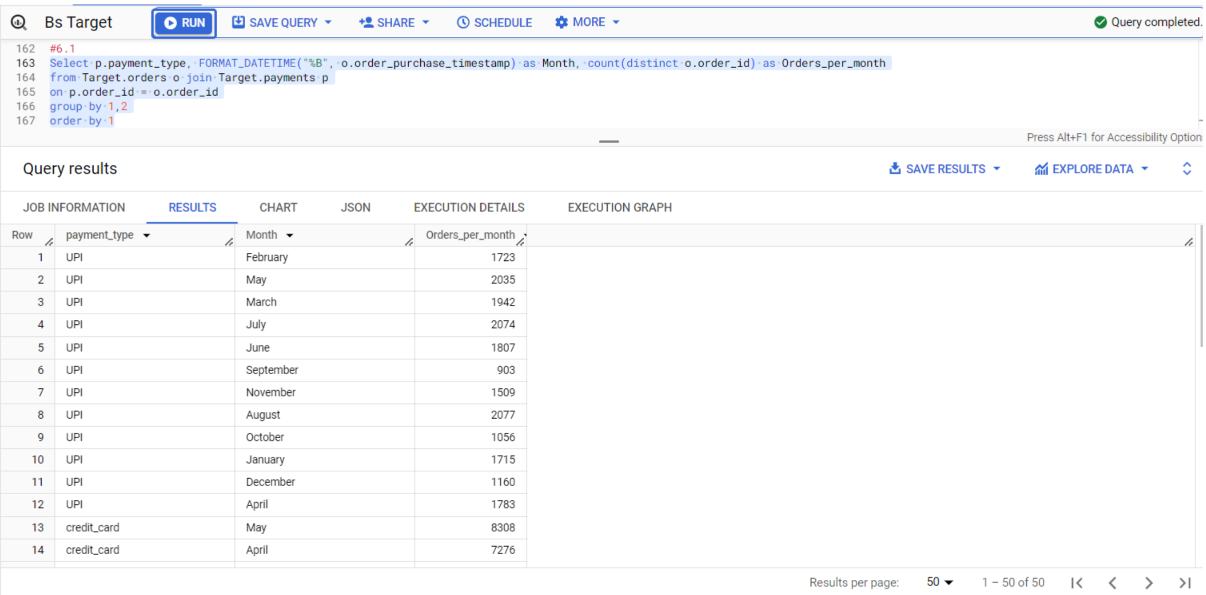
JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	customer_state	diff_estimated_delive
1	AC	-19.7625
2	RO	-19.1316872427...
3	AP	-18.7313432835...
4	AM	-18.6068965517...
5	RR	-16.4146341463...

Insights: These are the best 5 states when it come to delivering orders.

Recommendations: The values are in -ve as the actual delivery date was lesser than the estimated delivery date.

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

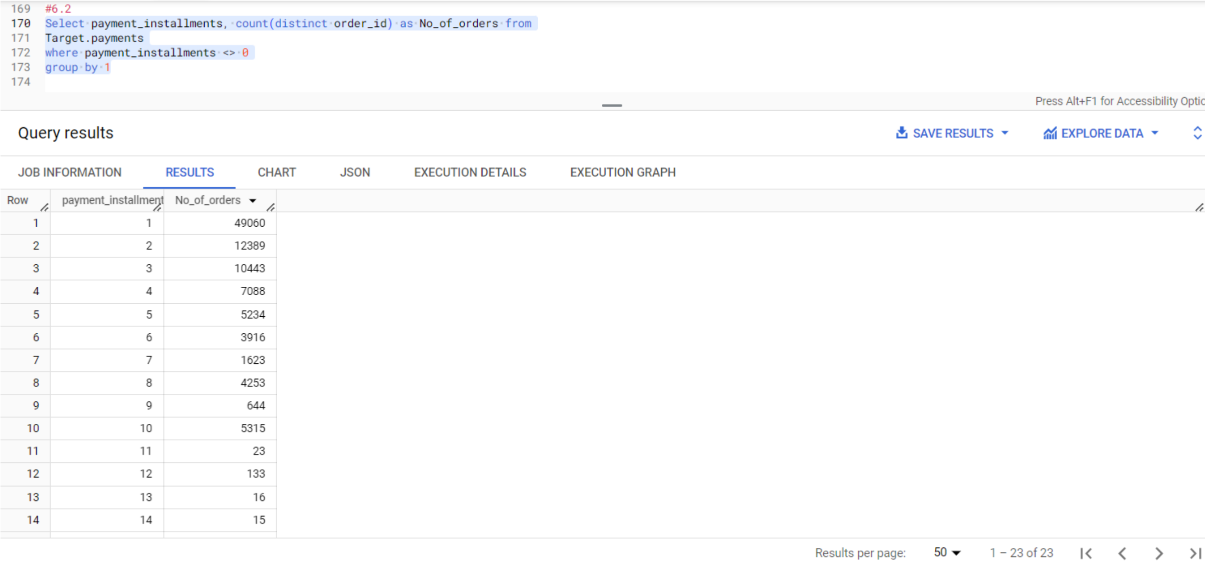


```
#6.1
Select p.payment_type, FORMAT_DATETIME("%B", o.order_purchase_timestamp) as Month, count(distinct o.order_id) as Orders_per_month
from Target.orders o join Target.payments p
on p.order_id = o.order_id
group by 1,2
order by 1
```

Row	payment_type	Month	Orders_per_month
1	UPI	February	1723
2	UPI	May	2035
3	UPI	March	1942
4	UPI	July	2074
5	UPI	June	1807
6	UPI	September	903
7	UPI	November	1509
8	UPI	August	2077
9	UPI	October	1056
10	UPI	January	1715
11	UPI	December	1160
12	UPI	April	1783
13	credit_card	May	8308
14	credit_card	April	7276

Insights: This data give the monthly distribution of no. of orders placed using different payment types.
Recommendations: Target can come to know which is the most frequent used payment type by the customers across various months of the years and provide offers based on payment type to better their revenue.

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



```
#6.2
Select payment_installments, count(distinct order_id) as No_of_orders from
Target.payments
where payment_installments <> 0
group by 1
```

Row	payment_installment	No_of_orders
1	1	49060
2	2	12389
3	3	10443
4	4	7088
5	5	5234
6	6	3916
7	7	1623
8	8	4253
9	9	644
10	10	5315
11	11	23
12	12	133
13	13	16
14	14	15

Insights: The above data gives the no. of orders placed on the basis of the payment installments that have been paid.
Recommendations: This data can be validated to get more insights on payment installments.