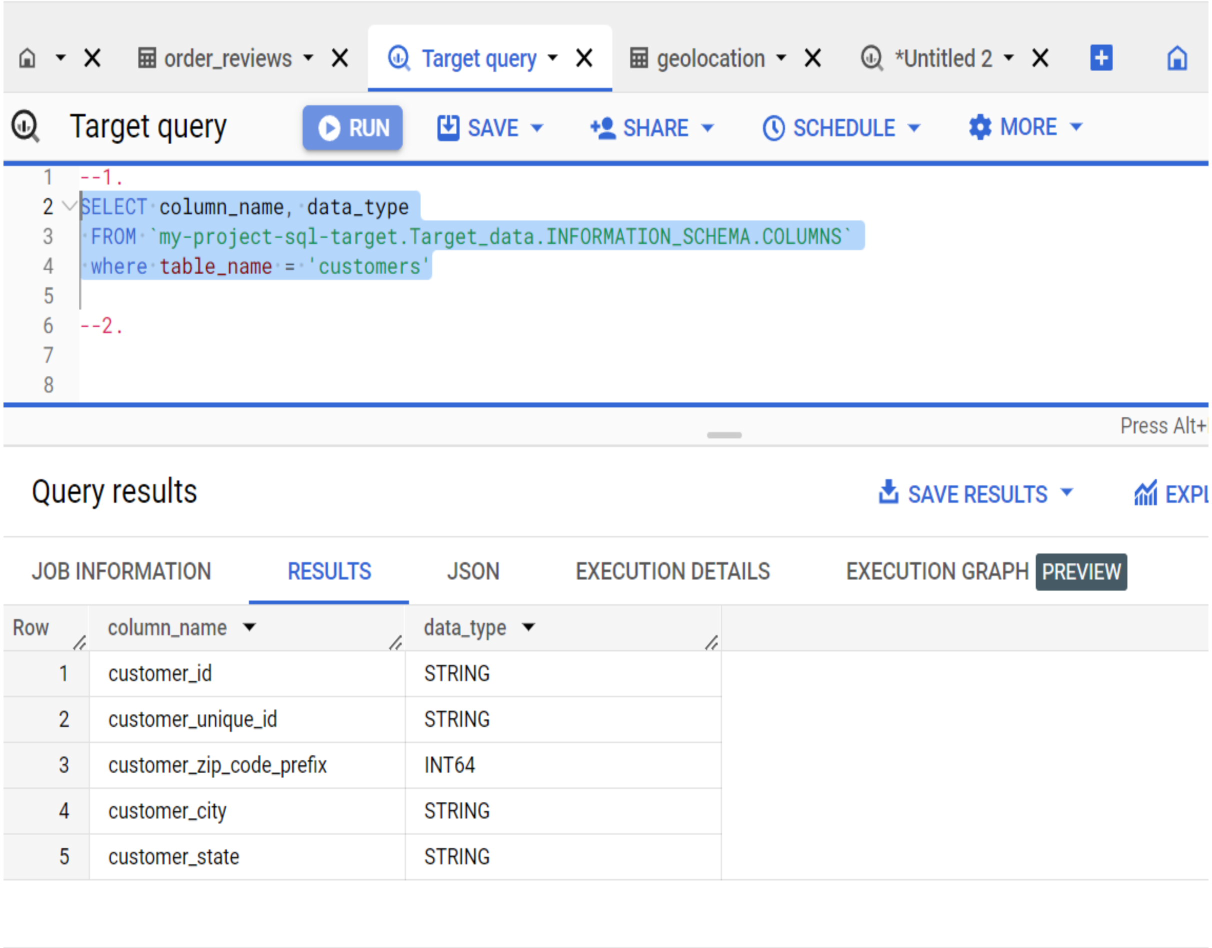
Problem Statement:

Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

What does 'good' look like?

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

```
SELECT column_name, data_type
 FROM `my-project-sql-target.Target_data.INFORMATION_SCHEMA.COLUMNS`
 where table_name = 'customers'
```

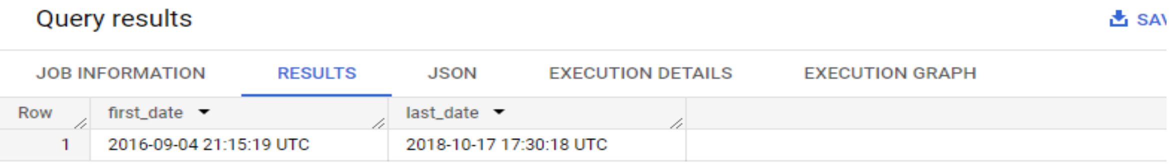


```
--1.
SELECT column_name, data_type
 FROM `my-project-sql-target.Target_data.INFORMATION_SCHEMA.COLUMNS`
 where table_name = 'customers'

--2.
```

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

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

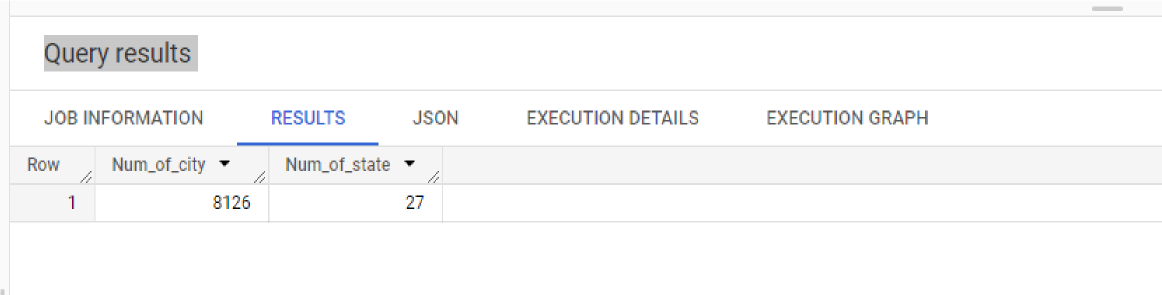
```
SELECT
min(order_purchase_timestamp) as first_date,
max(order_purchase_timestamp) as last_date,
from `Target_data.orders`
```



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

3. Count the number of Cities and States in our dataset.

```
with cte as
(
select
geolocation_city as city,
geolocation_state as state
from `Target_data.geolocation`
group by geolocation_city,geolocation_state

union all

select
customer_city as city,
customer_state as state
from `Target_data.customers`

union all

select
seller_city as city,
seller_state as state
from `Target_data.sellers`)

select count(distinct city) as Num_of_city,
count(distinct state) as Num_of_state
```

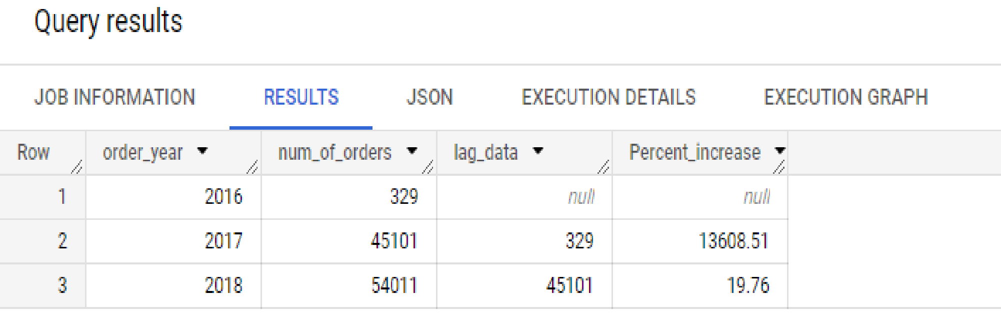


Row	Num_of_city	Num_of_state
1	8126	27

2. **In-depth Exploration:**

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

```
with cte as
(select order_year, count(order_id) as num_of_orders
from(
select order_id,
extract(year from order_purchase_timestamp) as order_year
from `Target_data.orders`
)tbl

group by order_year
ORDER BY order_year )
select *,lag(num_of_orders) over(order by order_year) as
lag_data,round(((num_of_orders-lag(num_of_orders) over(order by
order_year))/lag(num_of_orders) over(order by order_year)) *100,2) as
Percent_increase
from cte
ORDER BY order_year
```



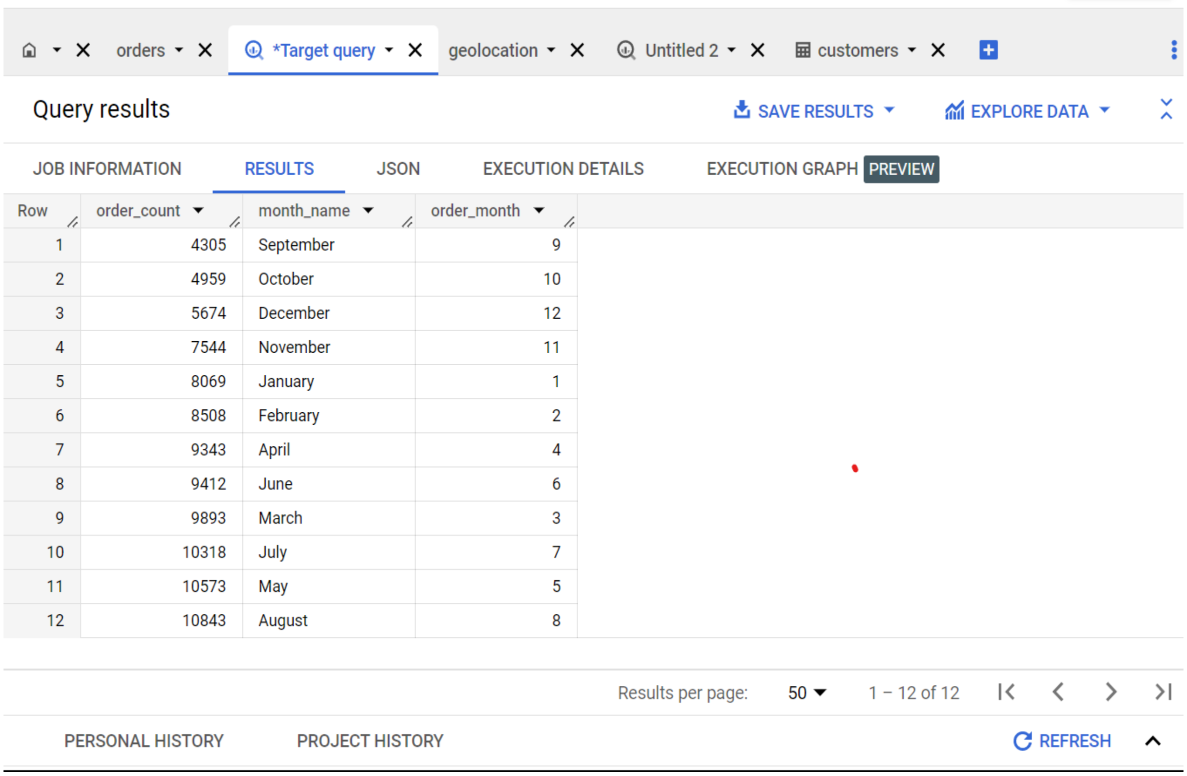
Row	order_year	num_of_orders	lag_data	Percent_increase
1	2016	329	null	null
2	2017	45101	329	13608.51
3	2018	54011	45101	19.76

Insights:

- With the minimum dataset in 2016 it is not meaningful to compare it to the complete dataset of 2017, but comparing 2017 to 2018 it is very evident that there is a significant jump of almost 20% is observed.Yes there is a an uptrend in orders

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

```
select count(order_id)as order_count,month_name,order_month
from(
select order_id,
FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) as
month_name,
extract(month from order_purchase_timestamp) as order_month
from `Target_data.orders`)tbl

group by month_name,order_month
order by order_count
```



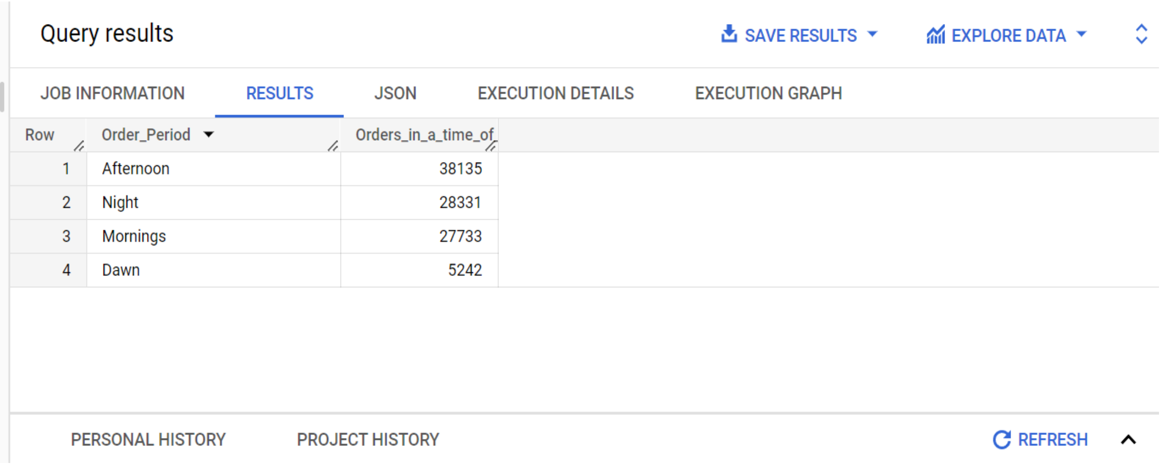
Row	order_count	month_name	order_month
1	4305	September	9
2	4959	October	10
3	5674	December	12
4	7544	November	11
5	8069	January	1
6	8508	February	2
7	9343	April	4
8	9412	June	6
9	9893	March	3
10	10318	July	7
11	10573	May	5
12	10843	August	8

Insights:

- Highest number of orders were placed in the month of August
- Lowest number of orders were placed in the month of September

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_Period,count(Order_Period) as Orders_in_a_time_of_day
from
(select order_id,
      case  when extract(hour from order_purchase_timestamp) between
00 and 06
            then 'Dawn'
            when extract(hour from order_purchase_timestamp) between
07 and 12
            then 'Mornings'
             when extract(hour from order_purchase_timestamp) between
13 and 18
            then 'Afternoon'
             when extract(hour from order_purchase_timestamp) between
19 and 23
            then 'Night'

      end as Order_Period
  from `Target_data.orders`)tbl

  group by Order_Period
  order by Orders_in_a_time_of_day desc
```



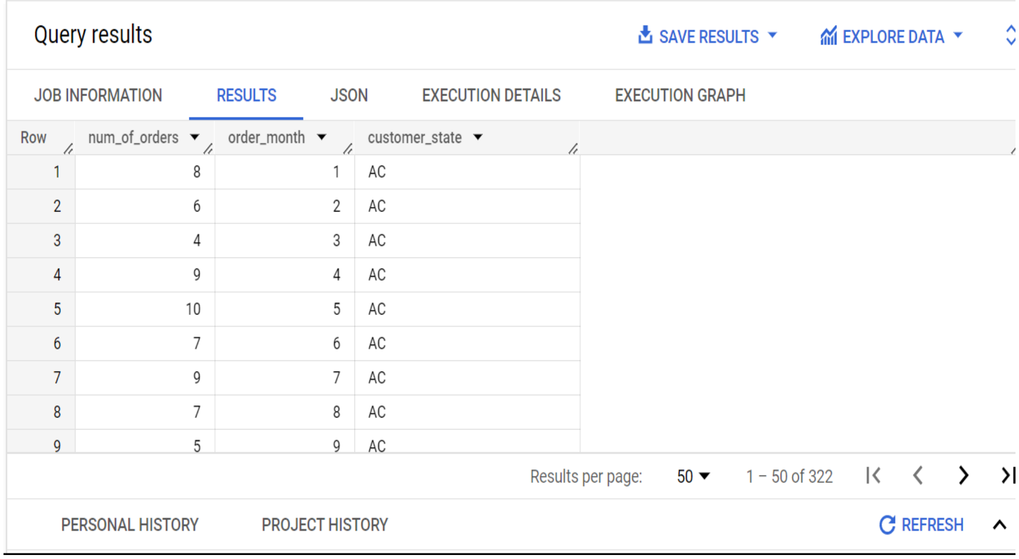
Row	Order_Period	Orders_in_a_time_of
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

Insights:

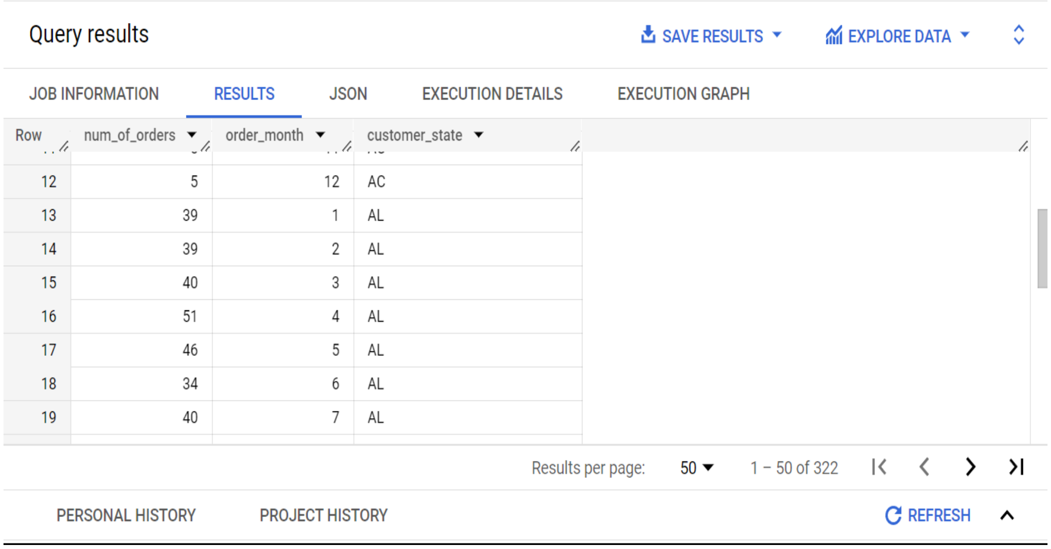
- From the result it is evident that Brazilian customers mostly place their orders in the afternoon and least place in the dawn time

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

```
select count(order_id) as num_of_orders, order_month,customer_state

from (
  select order_id,c.customer_state, extract(month from
order_purchase_timestamp) as order_month

  from `Target_data.customers` c
   right join `Target_data.orders`o
  on c.customer_id= o.customer_id
) tbl

group by order_month,customer_state
order by customer_state,order_month
```



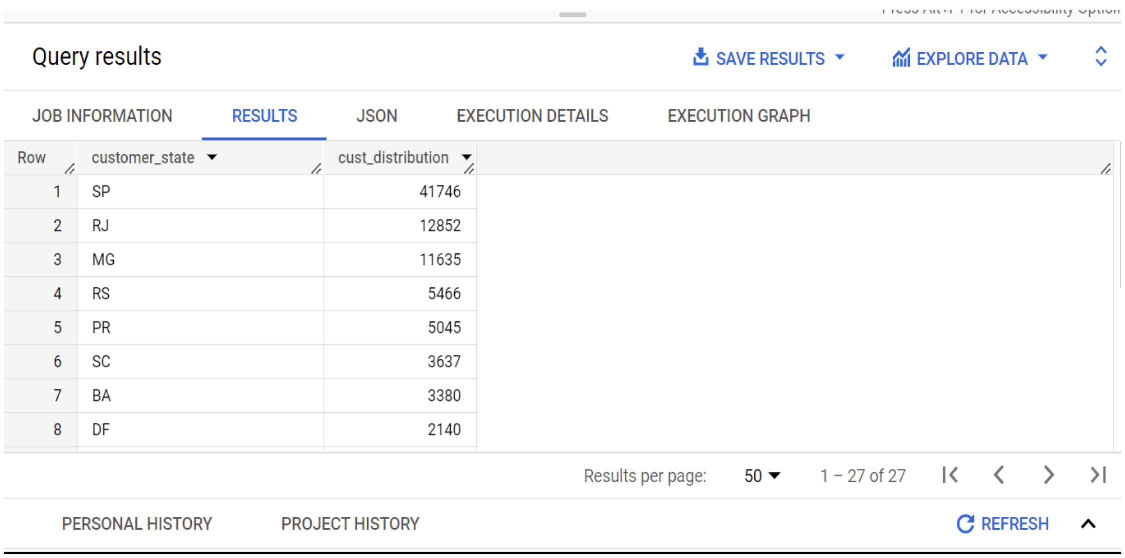
Row	num_of_orders	order_month	customer_state
1	8	1	AC
2	6	2	AC
3	4	3	AC
4	9	4	AC
5	10	5	AC
6	7	6	AC
7	9	7	AC
8	7	8	AC
9	5	9	AC



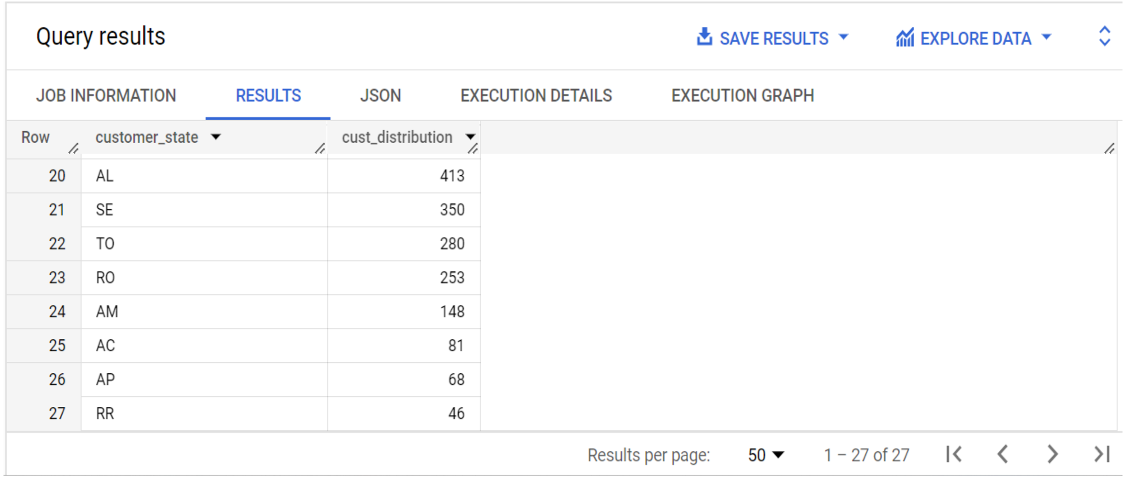
Row	num_of_orders	order_month	customer_state
12	5	12	AC
13	39	1	AL
14	39	2	AL
15	40	3	AL
16	51	4	AL
17	46	5	AL
18	34	6	AL
19	40	7	AL

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

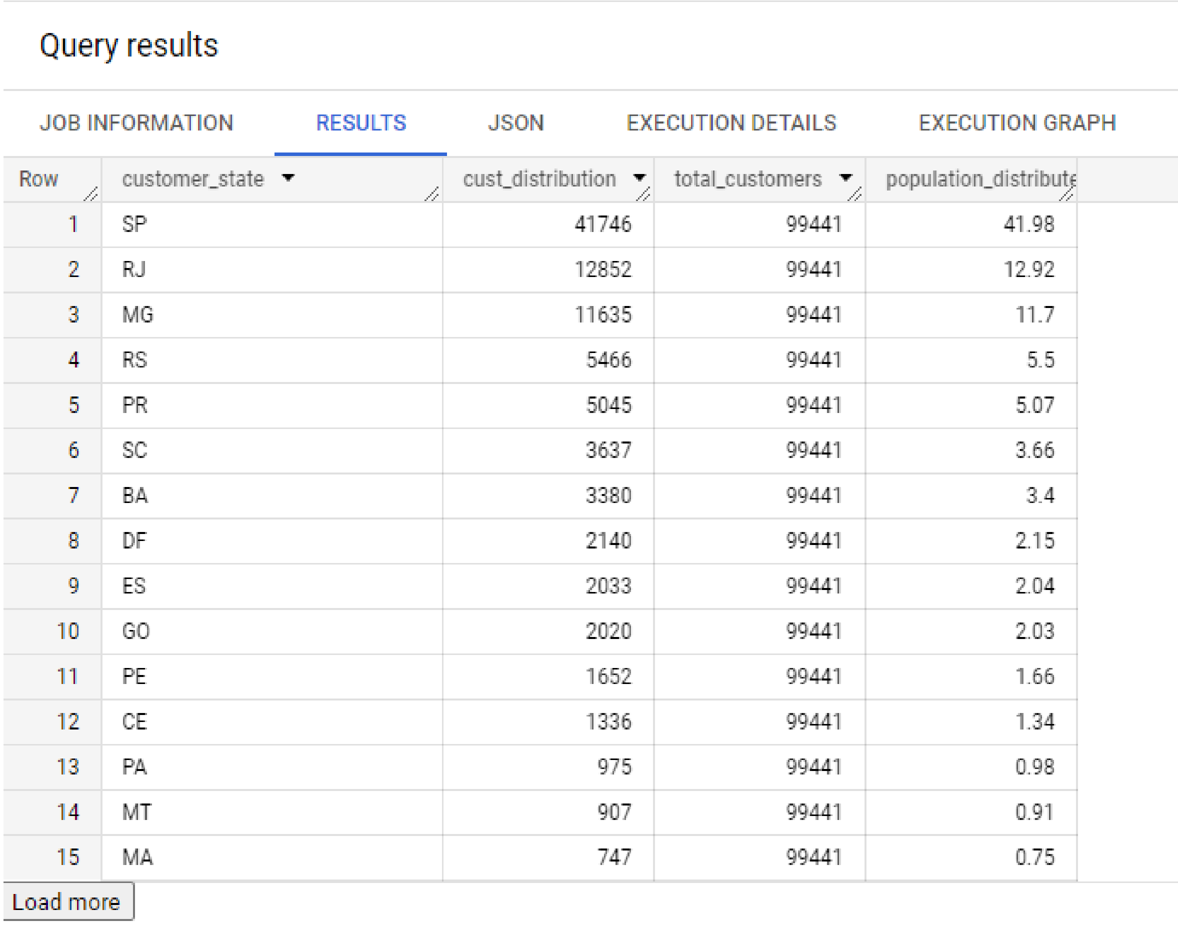
```
with cte as

(
select *, sum(cust_distribution) over () as total_customers
from
(select customer_state, count(distinct(customer_id)) as cust_distribution
from `Target_data.customers`
group by customer_state
order by cust_distribution desc) tbl
)

select *, round(cust_distribution/total_customers *100,2) as
population_distributed_percent
from cte
order by cust_distribution desc
```



Row	customer_state	cust_distribution
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140



Row	customer_state	cust_distribution
20	AL	413
21	SE	350
22	TO	280
23	RO	253
24	AM	148
25	AC	81
26	AP	68
27	RR	46



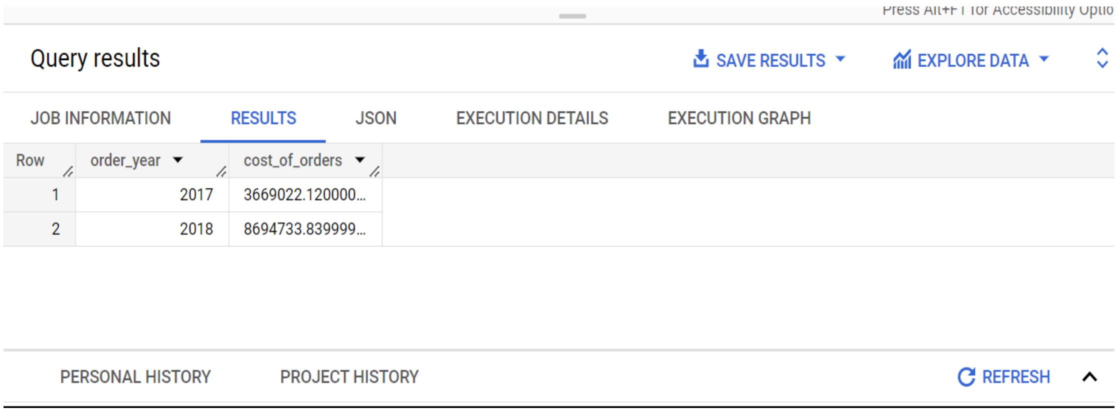
Row	customer_state	cust_distribution	total_customers	population_distribute
1	SP	41746	99441	41.98
2	RJ	12852	99441	12.92
3	MG	11635	99441	11.7
4	RS	5466	99441	5.5
5	PR	5045	99441	5.07
6	SC	3637	99441	3.66
7	BA	3380	99441	3.4
8	DF	2140	99441	2.15
9	ES	2033	99441	2.04
10	GO	2020	99441	2.03
11	PE	1652	99441	1.66
12	CE	1336	99441	1.34
13	PA	975	99441	0.98
14	MT	907	99441	0.91
15	MA	747	99441	0.75

Insights:

- State SP has the highest number of customers and customers distribution of more than 40% is observed
- State RR has the lowest number of customers and customers distribution is only 0.05%

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

```
with cte as
(
select o.order_id,p.payment_value,extract(month from
order_purchase_timestamp) as order_month,extract(year from
order_purchase_timestamp) as order_year
from `Target_data.payments` p
join `Target_data.orders` o
on p.order_id=o.order_id)

select order_year,sum(payment_value) as cost_of_orders
from cte
where (order_month between 1 and 8) and order_year in(2017,2018)
group by order_year
order by order_year
```



Row	order_year	cost_of_orders
1	2017	3669022.120000...
2	2018	8694733.839999...

Insights:

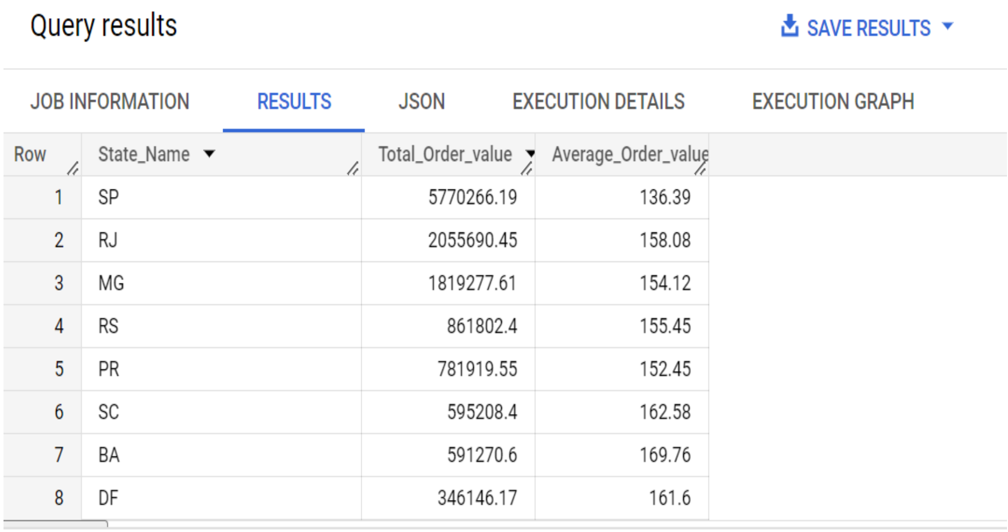
(% increase in the cost of orders from year 2017 to 2018)

```
=(8694733.839999-3669022.1200)/3669022.1200 *100
=136.97%
```

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

```
SELECT
c.customer_state as State_Name,
Round(Sum(payment_value),2) as Total_Order_value,
Round(avg(payment_value),2) as Average_Order_value,

from `Target_data.customers` as c
inner join `Target_data.orders` as o on c.customer_id =
o.customer_id
inner join `Target_data.payments` as p on o.order_id = p.order_id
where order_status = 'delivered'
group by State_Name
order by Total_Order_value desc
```

Following table shows the top 3 states with highest total order values



Row	State_Name	Total_Order_value	Average_Order_value
1	SP	5770266.19	136.39
2	RJ	2055690.45	158.08
3	MG	1819277.61	154.12
4	RS	861802.4	155.45
5	PR	781919.55	152.45
6	SC	595208.4	162.58
7	BA	591270.6	169.76
8	DF	346146.17	161.6

Following table shows the top 3 states with highest average order values

Insights:

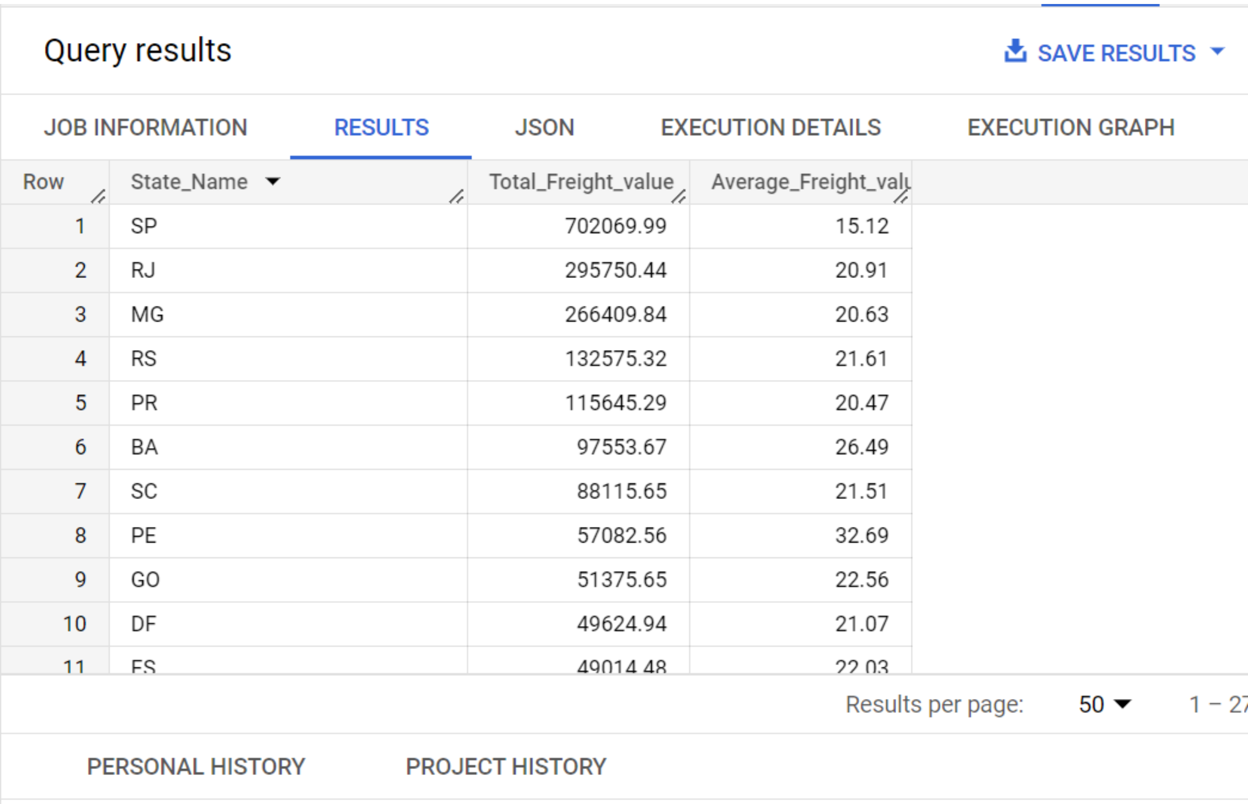
- State SP has the highest order price
- State RR has the lowest order price
- State PB has the highest average price
- State SP has the lowest average price

From this it shows state SP is highly populated to get the maximum order price and lowest in average price shows orders of less worth is ordered a lot in that state

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

```
SELECT
c.customer_state as State_Name,
Round(Sum(freight_value),2) as Total_Freight_value,
Round(avg(freight_value),2) as Average_Freight_value

from `Target_data.customers` as c
inner join `Target_data.orders` as o on c.customer_id = o.customer_id
inner join `Target_data.order_items` as p on o.order_id = p.order_id
where order_status = 'delivered'

group by State_Name
order by Total_Freight_value desc
```

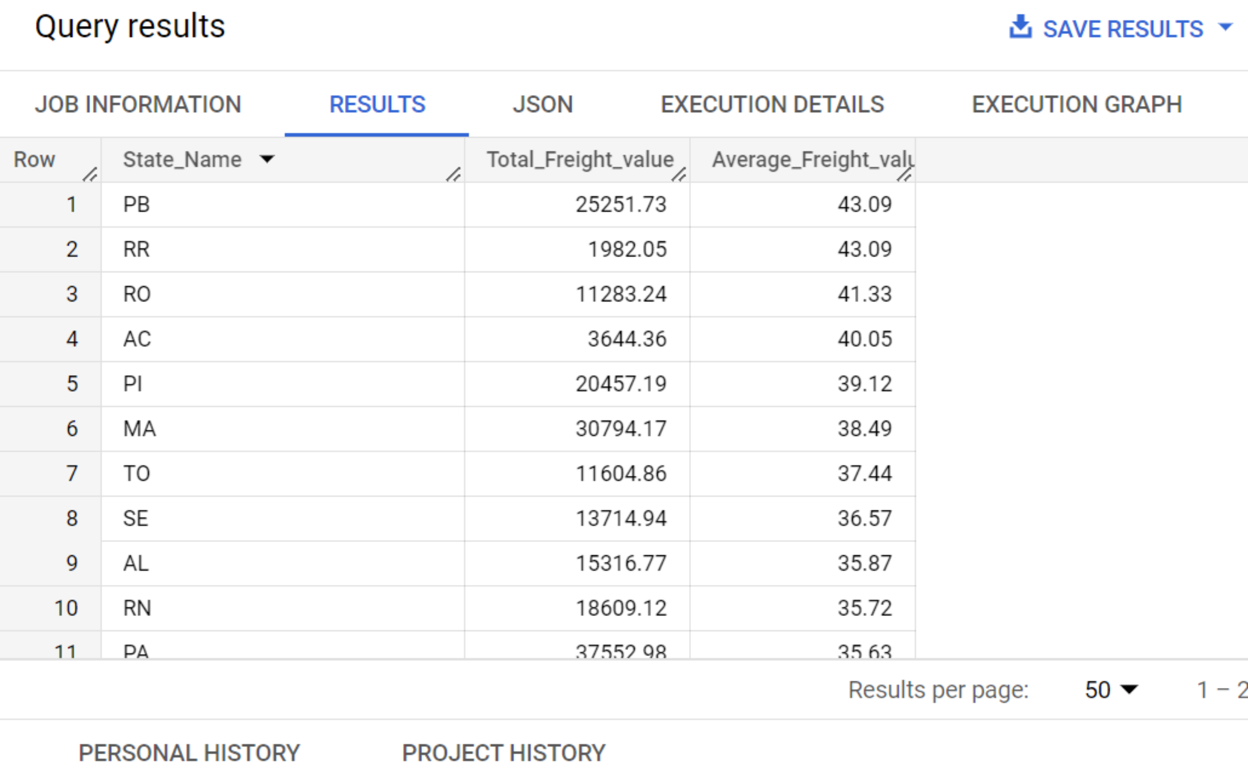
The following table is arranged on highest total freight values



Row	State_Name	Total_Freight_value	Average_Freight_valu
1	SP	702069.99	15.12
2	RJ	295750.44	20.91
3	MG	266409.84	20.63
4	RS	132575.32	21.61
5	PR	115645.29	20.47
6	BA	97553.67	26.49
7	SC	88115.65	21.51
8	PE	57082.56	32.69
9	GO	51375.65	22.56
10	DF	49624.94	21.07
11	ES	49014.48	22.03

Results per page: 50 1 – 27

```
The following table is arranged on highest average freight values
```



Row	State_Name	Total_Freight_value	Average_Freight_valu
1	PB	25251.73	43.09
2	RR	1982.05	43.09
3	RO	11283.24	41.33
4	AC	3644.36	40.05
5	PI	20457.19	39.12
6	MA	30794.17	38.49
7	TO	11604.86	37.44
8	SE	13714.94	36.57
9	AL	15316.77	35.87
10	RN	18609.12	35.72
11	PA	37552.98	35.63

Insights:

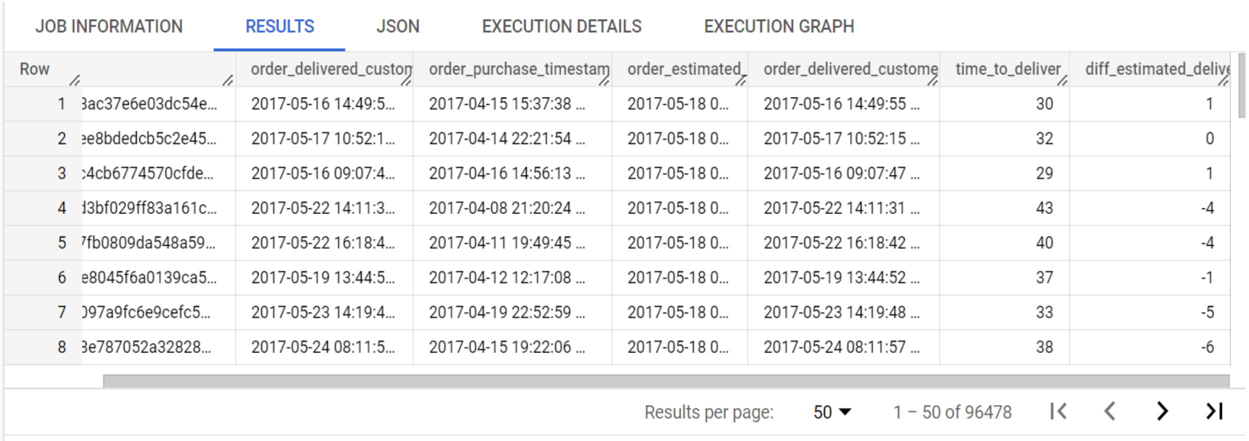
- State SP has the highest freight value
- State RR has the lowest freight value
- State PB has the average highest freight value
- State RN has the average lowest freight value

5. **Analysis based on sales, freight and delivery time.**
    1. Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
       Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
       Do this in a single query.

       You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
        - **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp

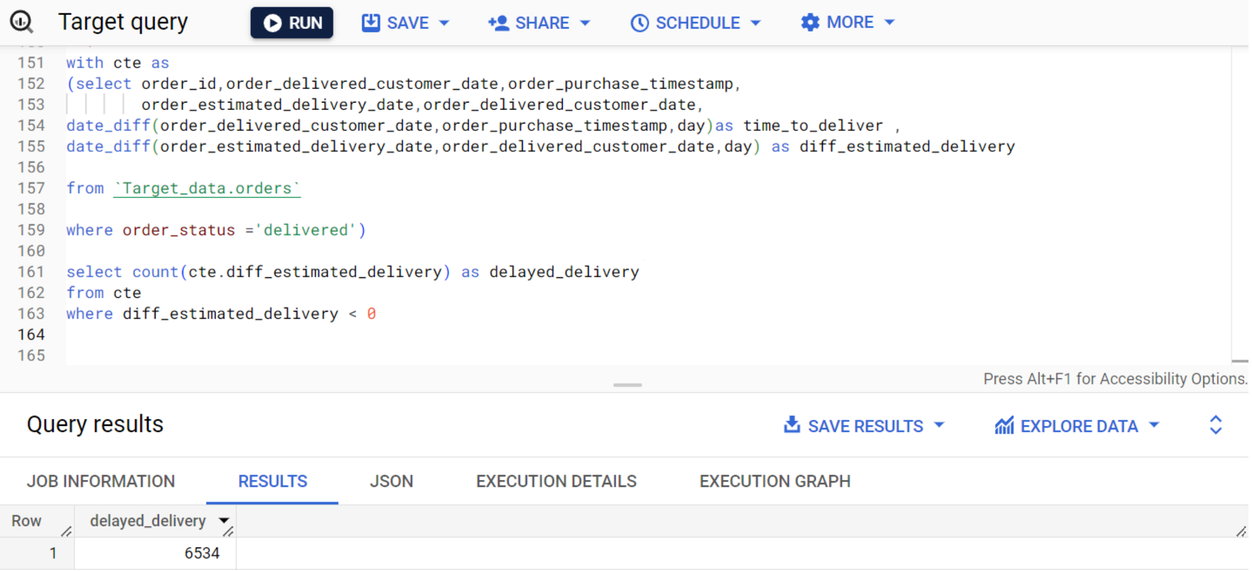
- **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

```
select
order_id,order_delivered_customer_date,order_purchase_timestamp,
        order_estimated_delivery_date,order_delivered_customer_dat
e,
date_diff(order_delivered_customer_date,order_purchase_timestamp,d
ay)as time_to_deliver ,
date_diff(order_estimated_delivery_date,order_delivered_customer_d
ate,day) as diff_estimated_delivery

from `Target_data.orders`

where order_status ='delivered'
```



Row		order_delivered_custon	order_purchase_timestam	order_estimated_	order_delivered_custome	time_to_deliver	diff_estimated_delive
1	3ac37e6e03dc54e...	2017-05-16 14:49:5...	2017-04-15 15:37:38 ...	2017-05-18 0...	2017-05-16 14:49:55 ...	30	1
2	ee8bdedcb5c2e45...	2017-05-17 10:52:1...	2017-04-14 22:21:54 ...	2017-05-18 0...	2017-05-17 10:52:15 ...	32	0
3	:4cb6774570cfde...	2017-05-16 09:07:4...	2017-04-16 14:56:13 ...	2017-05-18 0...	2017-05-16 09:07:47 ...	29	1
4	13bf029ff83a161c...	2017-05-22 14:11:3...	2017-04-08 21:20:24 ...	2017-05-18 0...	2017-05-22 14:11:31 ...	43	-4
5	7fb0809da548a59...	2017-05-22 16:18:4...	2017-04-11 19:49:45 ...	2017-05-18 0...	2017-05-22 16:18:42 ...	40	-4
6	e8045f6a0139ca5...	2017-05-19 13:44:5...	2017-04-12 12:17:08 ...	2017-05-18 0...	2017-05-19 13:44:52 ...	37	-1
7	097a9fc6e9cefc5...	2017-05-23 14:19:4...	2017-04-19 22:52:59 ...	2017-05-18 0...	2017-05-23 14:19:48 ...	33	-5
8	3e787052a32828...	2017-05-24 08:11:5...	2017-04-15 19:22:06 ...	2017-05-18 0...	2017-05-24 08:11:57 ...	38	-6

Results per page: 50 1 – 50 of 96478



```
with cte as
(select order_id,order_delivered_customer_date,order_purchase_timestamp,
        order_estimated_delivery_date,order_delivered_customer_date,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day)as time_to_deliver ,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as diff_estimated_delivery

from `Target_data.orders`

where order_status ='delivered')

select count(cte.diff_estimated_delivery) as delayed_delivery
from cte
where diff_estimated_delivery < 0
```

Press Alt+F1 for Accessibility Options.

Query results

Row	delayed_delivery
1	6534

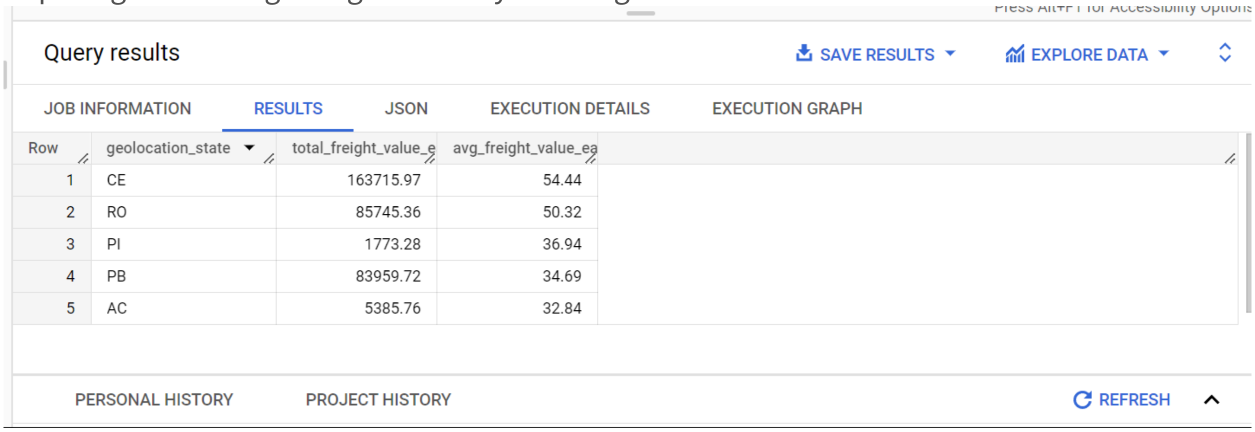
Insights:

- Total of 6534 delayed deliveries have been done. This must be taken into consideration

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

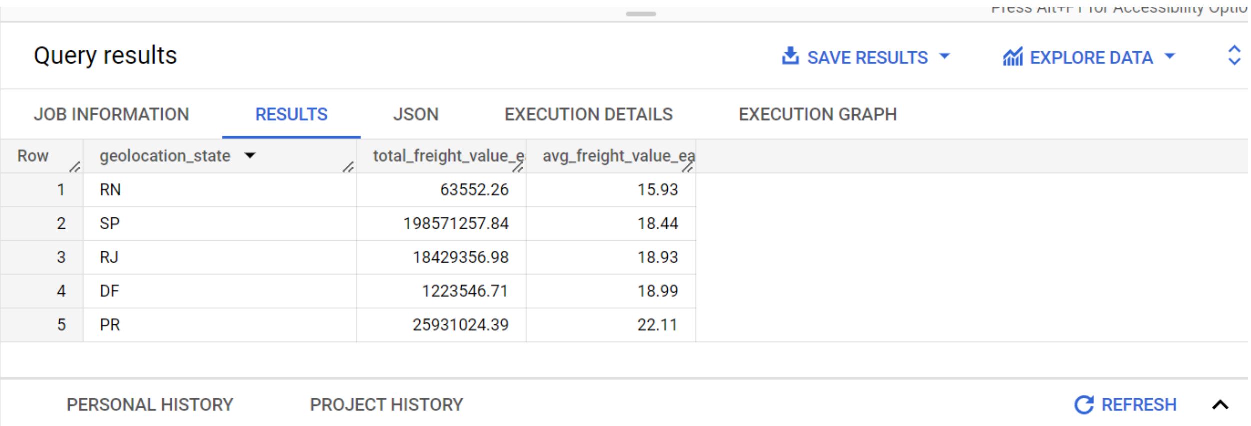
```
with cte as
(
select *

from `Target_data.order_items` o
join `Target_data.sellers` s
on o.seller_id = s.seller_id
 join `Target_data.geolocation` g
on g.geolocation_zip_code_prefix= s.seller_zip_code_prefix)

select geolocation_state,round(sum(freight_value),2) as
total_freight_value_each_state,round(avg(freight_value),2) as
avg_freight_value_each_state
from cte

group by geolocation_state
order by avg_freight_value_each_state desc
limit 5
```

Top 5 highest average freight value by ordering in desc:



Row	geolocation_state	total_freight_value_e	avg_freight_value_ea
1	CE	163715.97	54.44
2	RO	85745.36	50.32
3	PI	1773.28	36.94
4	PB	83959.72	34.69
5	AC	5385.76	32.84

Top 5 lowest average freight value by ordering in asc:

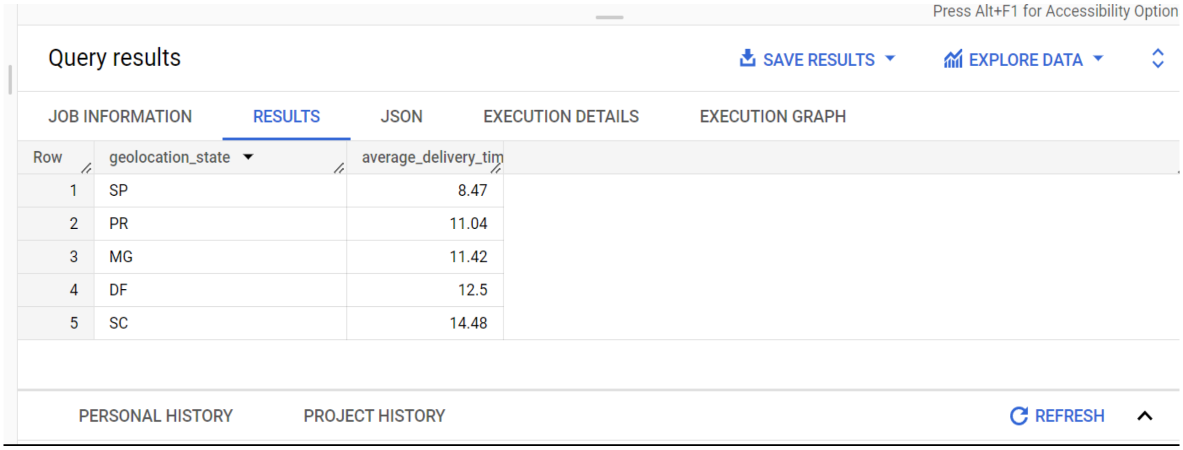


Row	geolocation_state	total_freight_value_e	avg_freight_value_ea
1	RN	63552.26	15.93
2	SP	198571257.84	18.44
3	RJ	18429356.98	18.93
4	DF	1223546.71	18.99
5	PR	25931024.39	22.11

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

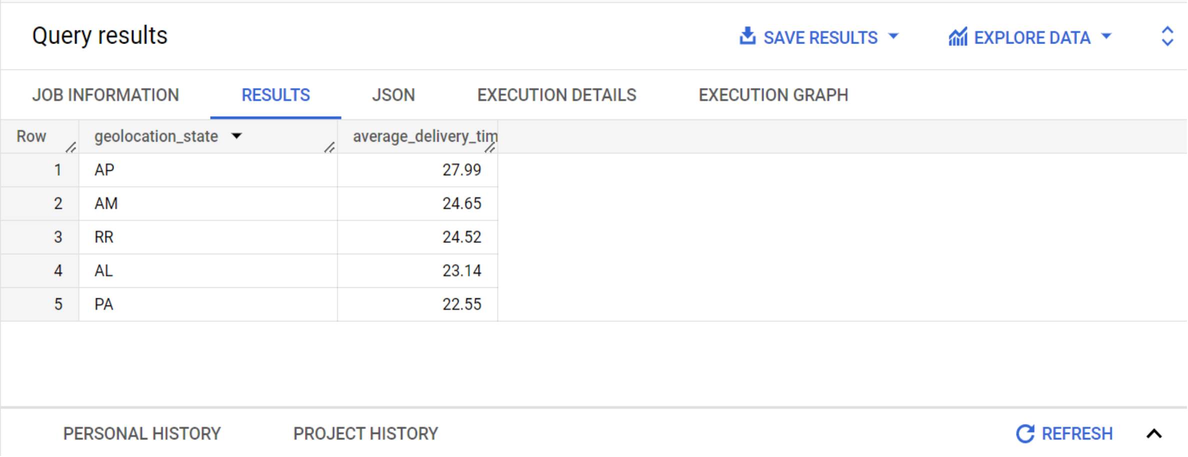
```
with cte as
(
select
o.order_id,o.order_delivered_customer_date,o.order_purchase_timestamp,

date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)
as time_to_deliver_in_days ,
g.geolocation_state

from `Target_data.orders` o
join `Target_data.customers`c
on o.customer_id = c.customer_id
join `Target_data.geolocation` g
on c.customer_zip_code_prefix = g.geolocation_zip_code_prefix

where order_status ='delivered' and
date_diff(order_delivered_customer_date,order_purchase_timestamp,day)
is  not null

order by time_to_deliver_in_days )


select geolocation_state,round(avg(time_to_deliver_in_days),2) as
average_delivery_time_days
from cte
group by geolocation_state
order by average_delivery_time_days
limit 5
```

Top 5 states with lowest average delivery time:



Row	geolocation_state	average_delivery_tim
1	SP	8.47
2	PR	11.04
3	MG	11.42
4	DF	12.5
5	SC	14.48

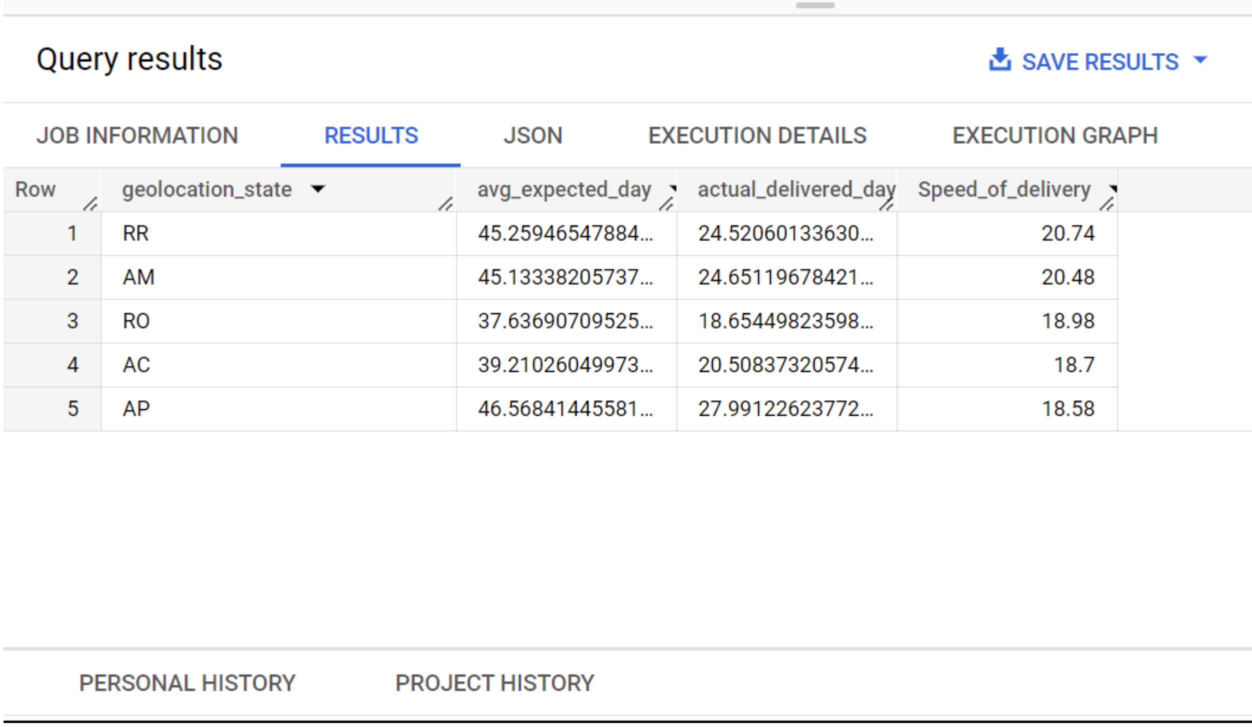
Top 5 states with highest average delivery time:



Row	geolocation_state	average_delivery_tim
1	AP	27.99
2	AM	24.65
3	RR	24.52
4	AL	23.14
5	PA	22.55

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 cte as
(select
order_id,order_delivered_customer_date,order_purchase_timestamp,g.geoloca
tion_state,
        order_estimated_delivery_date,order_delivered_customer_date,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day)as
actual_delivery_day ,
date_diff(order_estimated_delivery_date,order_purchase_timestamp,day) as
expected_delivery_day

from `Target_data.orders` o
join `Target_data.customers`c
on o.customer_id = c.customer_id
join `Target_data.geolocation` g
on c.customer_zip_code_prefix = g.geolocation_zip_code_prefix

where order_status ='delivered')

select geolocation_state,avg(expected_delivery_day)as
avg_expected_day,avg(actual_delivery_day) as
actual_delivered_day,round(avg(expected_delivery_day)-
avg(actual_delivery_day)) as Speed_of_delivery
from cte
group by geolocation_state

order by Speed_of_delivery desc
limit 5
```



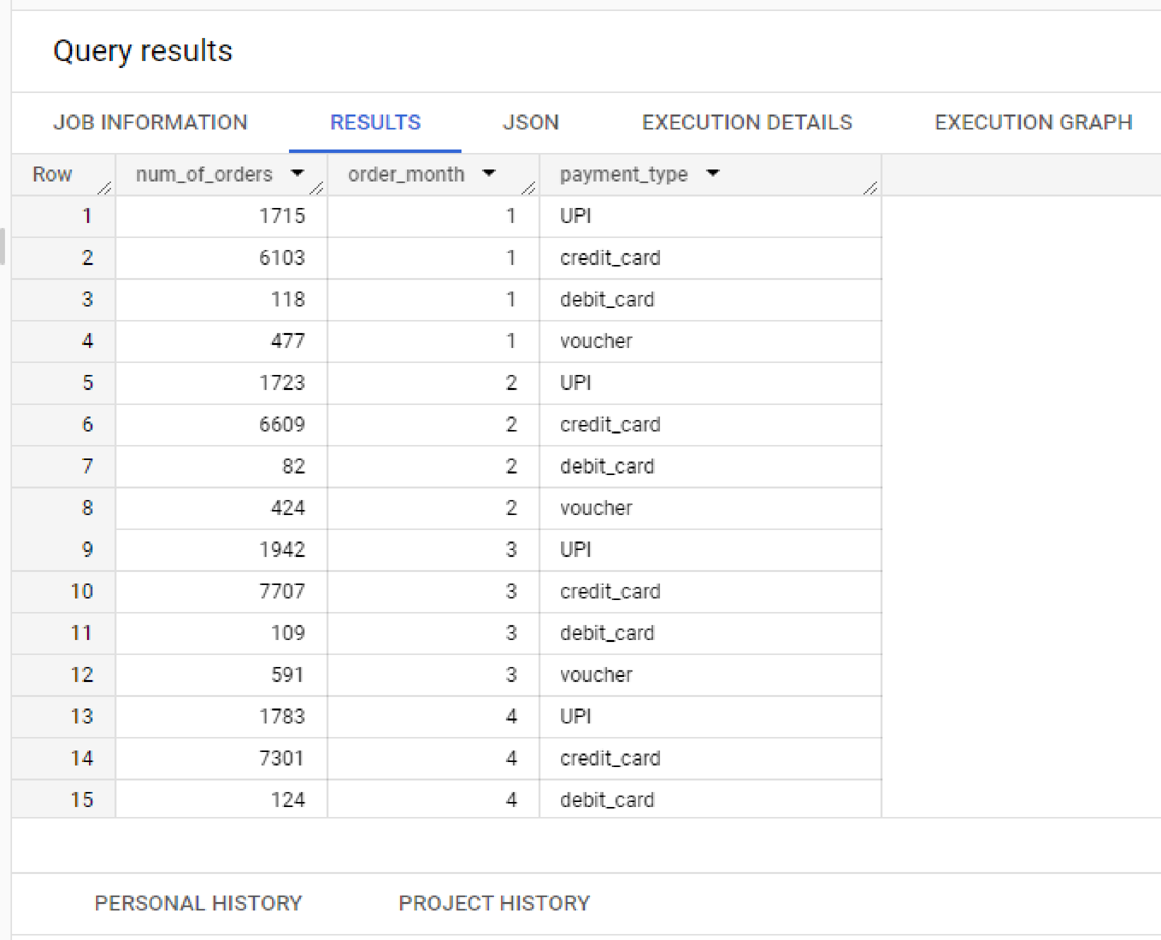
Row	geolocation_state	avg_expected_day	actual_delivered_day	Speed_of_delivery
1	RR	45.25946547884...	24.52060133630...	20.74
2	AM	45.13338205737...	24.65119678421...	20.48
3	RO	37.63690709525...	18.65449823598...	18.98
4	AC	39.21026049973...	20.50837320574...	18.7
5	AP	46.56841445581...	27.99122623772...	18.58

Insights:

Speed of delivery is difference between average estimated delivery days and average actual delivered days , so arranging in descending order will get us the avg quick deliveries. Listed above the top 5 states

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

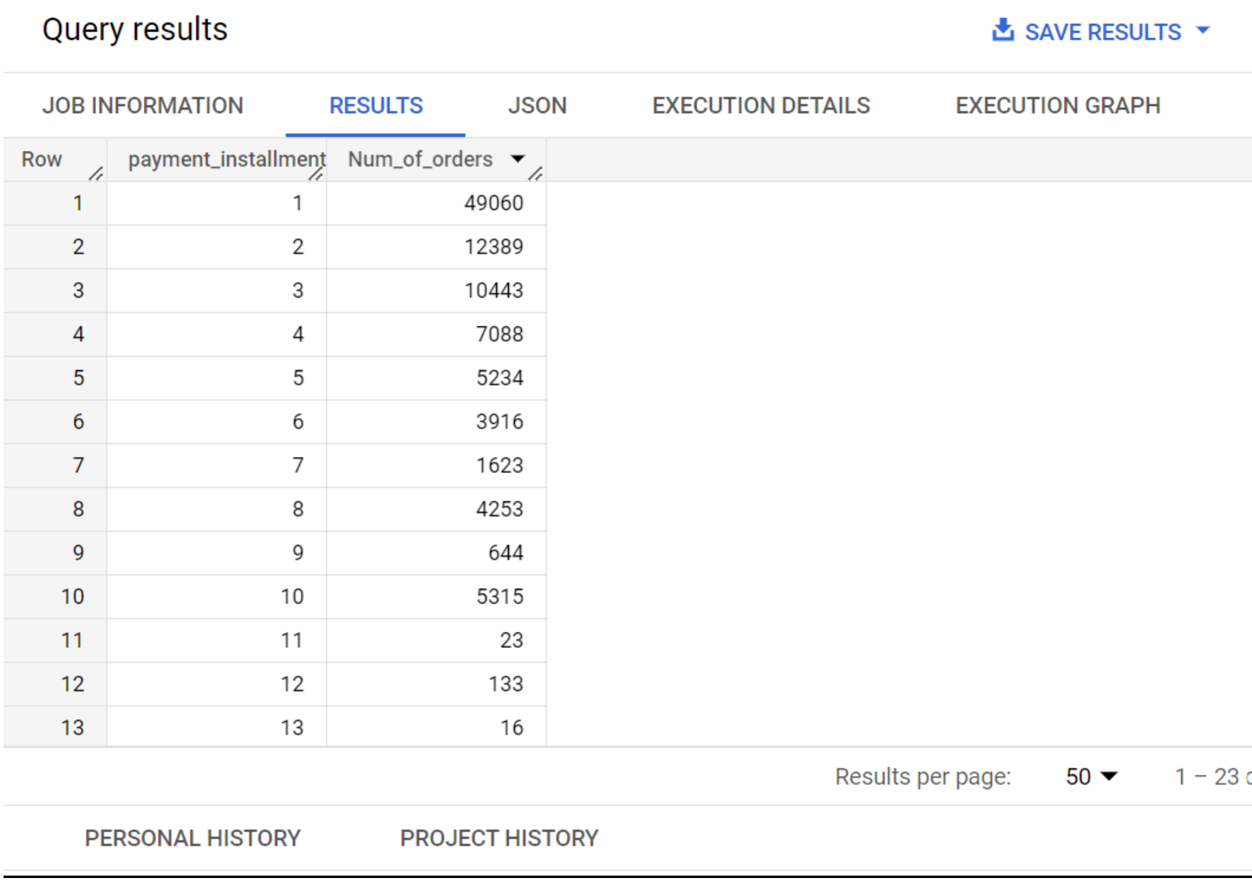
```
select count(o.order_id) as num_of_orders, extract(month from
o.order_purchase_timestamp) as order_month,payment_type

from `Target_data.orders` o
join `Target_data.payments` p
on o.order_id = p.order_id

group by  payment_type,order_month
order by order_month,payment_type
```



Row	num_of_orders	order_month	payment_type
1	1715	1	UPI
2	6103	1	credit_card
3	118	1	debit_card
4	477	1	voucher
5	1723	2	UPI
6	6609	2	credit_card
7	82	2	debit_card
8	424	2	voucher
9	1942	3	UPI
10	7707	3	credit_card
11	109	3	debit_card
12	591	3	voucher
13	1783	4	UPI
14	7301	4	credit_card
15	124	4	debit_card

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

```
select payment_installments,count(distinct(order_id)) as Num_of_orders
from `Target_data.payments`
where payment_installments >0
group by payment_installments
order by payment_installments
```



Row	payment_installment	Num_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