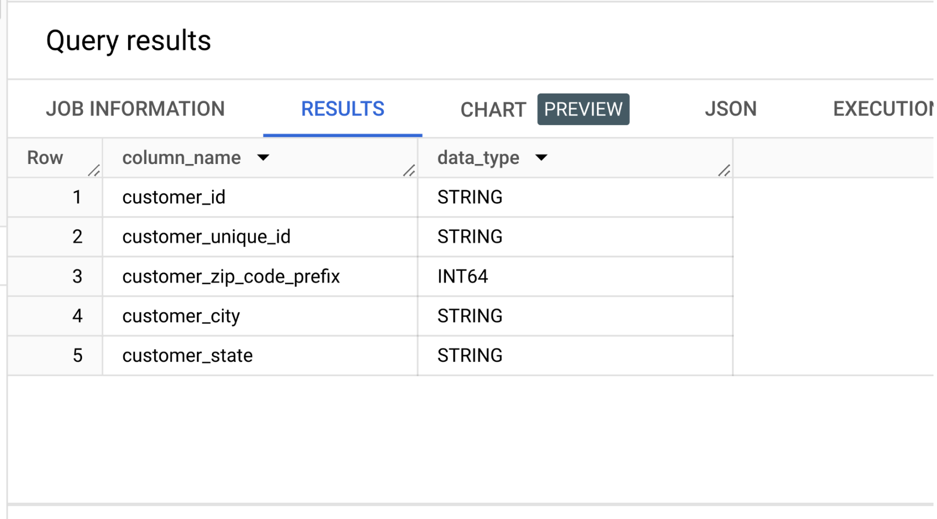
Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

Data type of all columns in the “customers” table

```
Select column_name,
data_type
from scaler-dsml-sql-402117.target_sql.INFORMATION_SCHEMA.COLUMNS
where table_name = 'customers'
```

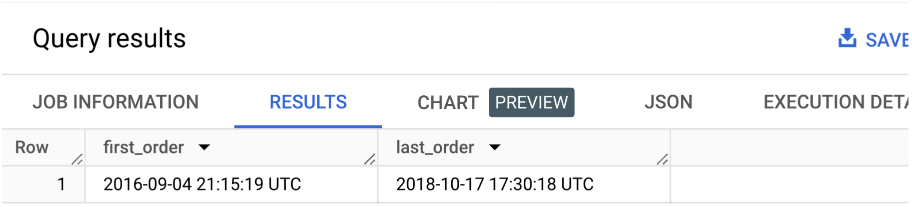


Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Insights : The datatypes in the table are majorly strings.

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

```
Select
min(order_purchase_timestamp) as first_order,
max(order_purchase_timestamp) as last_order
from `target_sql.orders`
```

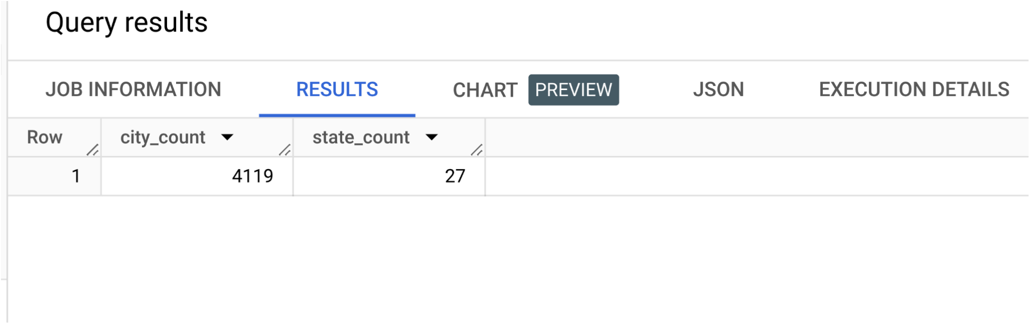


Row	first_order	last_order
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights : The first order date was in 2016 and the last was in 2018.

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

```
Select count(distinct customer_city) as city_count,
count(distinct customer_state) as state_count
from `Project_Target.customers` c
join `Project_Target.orders`p
on c.customer_id = p.customer_id
where p.order_purchase_timestamp between '2016-09-04' and '2018-10-17'
```



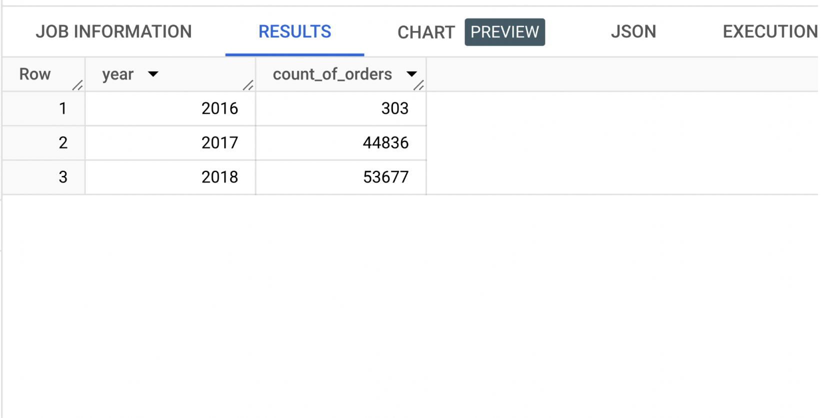
Row	city_count	state_count
1	4119	27

insights : The Cities & States of customers who ordered during the given period is 4119 and 27.

In-depth Exploration:

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

```
Select extract (year from order_purchase_timestamp) as year,
count(order_id) as count_of_orders
from `Project_Target.orders`
where order_status != 'canceled'
group by year
order by year
```

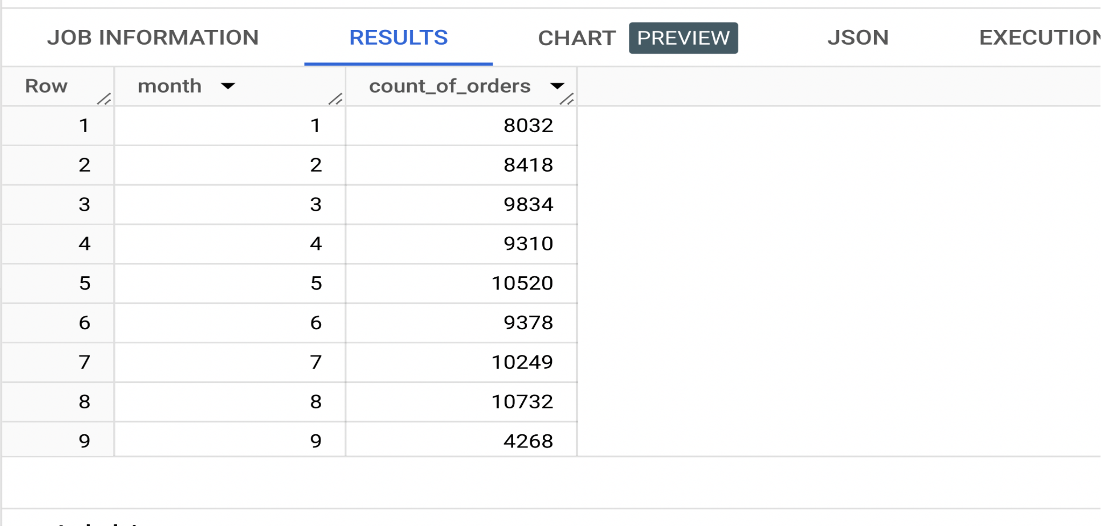


Row	year	count_of_orders
1	2016	303
2	2017	44836
3	2018	53677

Insights : The count of orders have increased over the years.
Assumptions : The orders that has been canceled have not been counted as placed

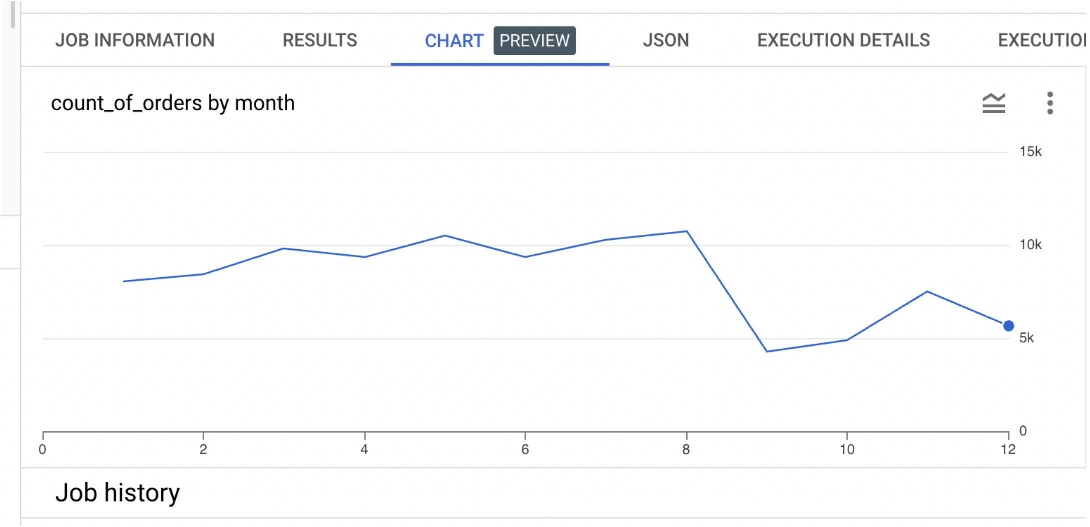
B.Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

```
Select extract(MONTH from order_purchase_timestamp) as month, count(order_id) as
count_of_orders
from `Project_Target.orders`
where order_status != 'canceled'
group by month
order by month, count_of_orders
```



Row	month	count_of_orders
1	1	8032
2	2	8418
3	3	9834
4	4	9310
5	5	10520
6	6	9378
7	7	10249
8	8	10732
9	9	4268

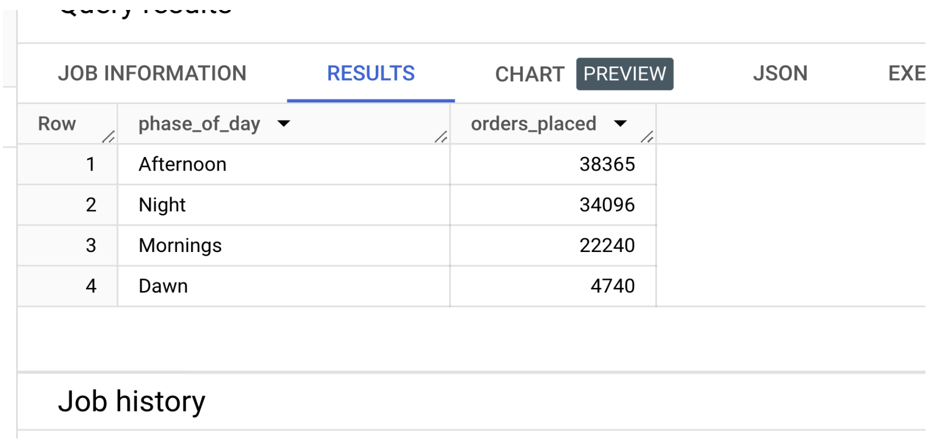
Insights : The top 3 seasonal months for sales are August, July and May.
Assumptions : The orders that has been canceled have not been counted as placed



C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

```
select
case when hours between '00:00:00' and '06:00:00' then 'dawn'
when hours between '06:00:01' and '12:00:00' then 'Morning'
when hours between '12:00:01' and '18:00:00' then 'Afternoon'
else 'Night'
end as time_of_day, count(order_id) as count_of_orders
```

```
from
(Select
extract(TIME from order_purchase_timestamp) as hours,order_id
from`Project_Target.orders`
)
group by time_of_day
order by count_of_orders
```



Query results

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXE

Row	phase_of_day	orders_placed
1	Afternoon	38365
2	Night	34096
3	Mornings	22240
4	Dawn	4740

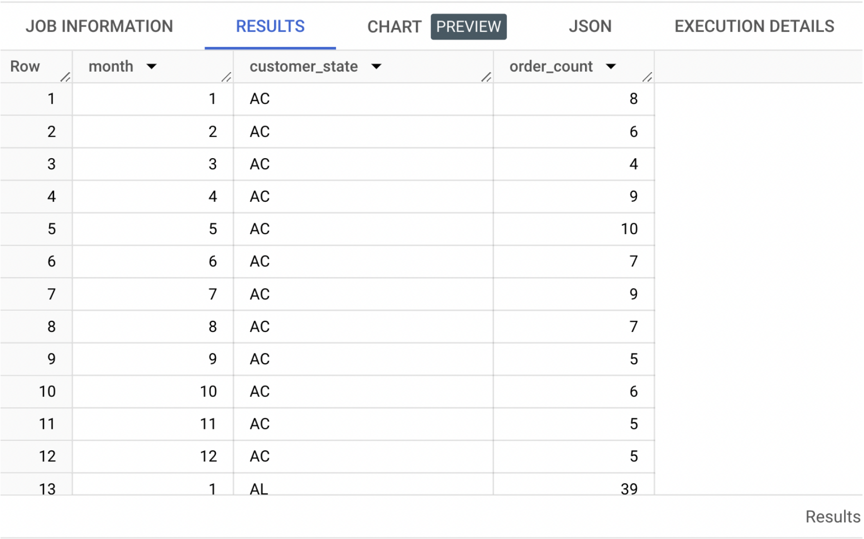
Job history

Insights : The placement of orders increases during the **afternoon**.

Evolution of E-commerce orders in the Brazil region:

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

```
Select month,a.customer_state,count(a.order_id) as order_count
from
(Select extract(MONTH from order_purchase_timestamp) as
month,o.order_id,customer_state
from `Project_Target.customers`c
join `Project_Target.orders`o
on c.customer_id = o.customer_id)a
group by month,customer_stat
order by customer_state, month
```

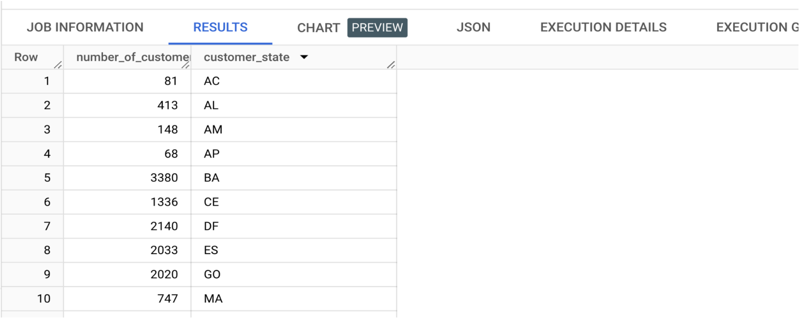


Row	month	customer_state	order_count
1	1	AC	8
2	2	AC	6
3	3	AC	4
4	4	AC	9
5	5	AC	10
6	6	AC	7
7	7	AC	9
8	8	AC	7
9	9	AC	5
10	10	AC	6
11	11	AC	5
12	12	AC	5
13	1	AL	39

Insights : The output shows the different fluctuations in the order per month for different states, according to the season, it keeps on increasing and decreasing.

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

```
Select count(customer_unique_id) as number_of_customers,customer_state
        from `Project_Target.customers`
        group by customer_state
        order by customer_state
```



Row	number_of_custome	customer_state
1	81	AC
2	413	AL
3	148	AM
4	68	AP
5	3380	BA
6	1336	CE
7	2140	DF
8	2033	ES
9	2020	GO
10	747	MA

Insights : The customers are segmented on the area basis, the top three states on customer distribution are SP,RJ and M.

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

A. Get the % increase in the cost of orders from year 2017 to 2018 *(include months between Jan to Aug only)*.

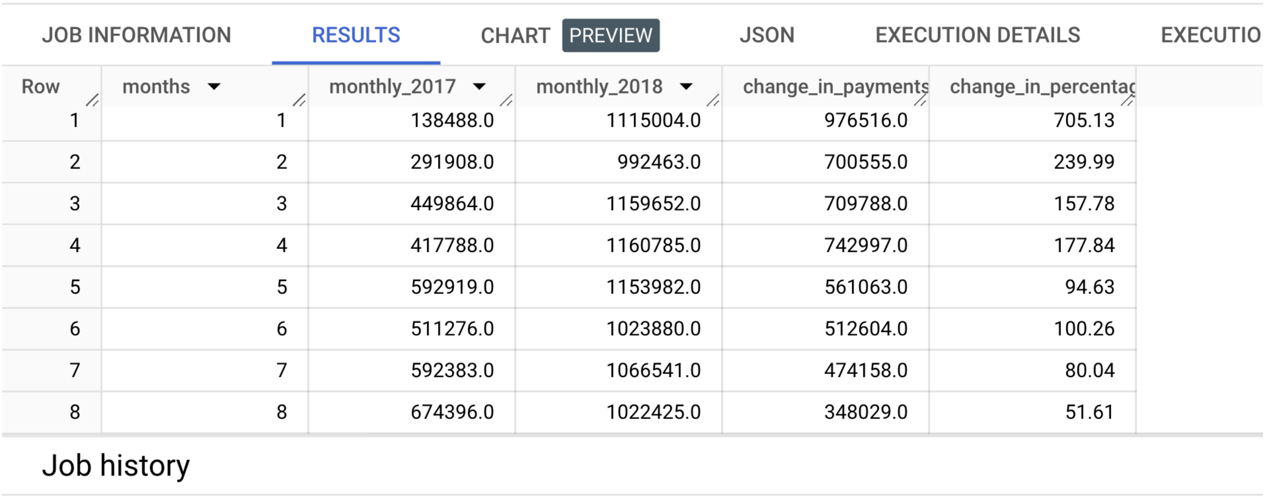
```sql
Select a.months,a.monthly_2017,b. monthly_2018,b.monthly_2018-a.monthly_2017 as
change_in_payments,
round(((b.monthly_2018-a.monthly_2017)/monthly_2017)*100,2) as change_in_percentage
from

(Select extract(year from order_purchase_timestamp) as years,
extract(month from order_purchase_timestamp) as months,round(sum(payment_value),0) as
monthly_2017
from `Project_Target.orders` o
join `Project_Target.payments `p
on o.order_id = p.order_id
where extract(year from order_purchase_timestamp) = 2017 and extract(month from
order_purchase_timestamp) between 1 and 8
group by years,months
order by months,years)a

join

(
Select extract(year from order_purchase_timestamp) as years,
extract(month from order_purchase_timestamp) as months,round(sum(payment_value),0) as
monthly_2018
from `Project_Target.orders` o
join `Project_Target.payments `p
on o.order_id = p.order_id
where extract(year from order_purchase_timestamp) = 2018 and extract(month from
order_purchase_timestamp) between 1 and 8
group by years,months
order by months,years)b

on a.months = b.months
order by months
```

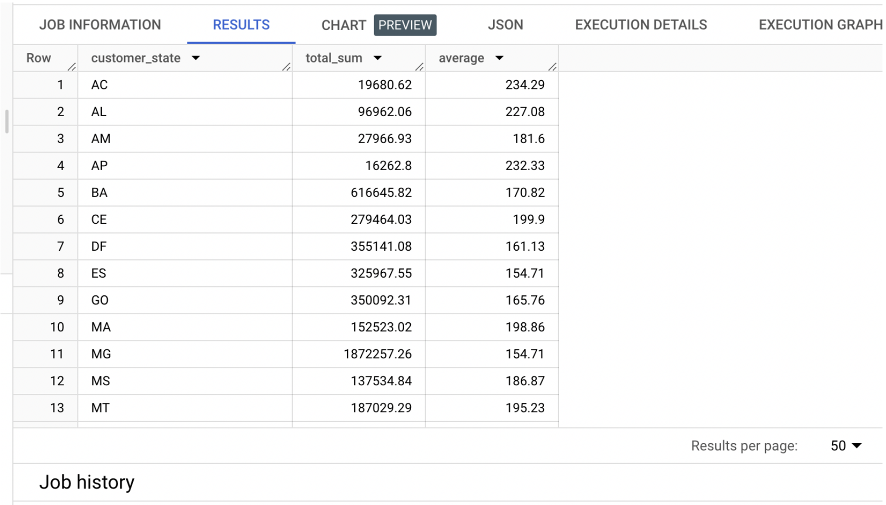


Row	months	monthly_2017	monthly_2018	change_in_payments	change_in_percentag
1	1	138488.0	1115004.0	976516.0	705.13
2	2	291908.0	992463.0	700555.0	239.99
3	3	449864.0	1159652.0	709788.0	157.78
4	4	417788.0	1160785.0	742997.0	177.84
5	5	592919.0	1153982.0	561063.0	94.63
6	6	511276.0	1023880.0	512604.0	100.26
7	7	592383.0	1066541.0	474158.0	80.04
8	8	674396.0	1022425.0	348029.0	51.61

Insights : The sales have increased throughout the months from Jan to Aug from the past year(2017)

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

```
Select customer_state, round(sum(payment_value),2) as total_sum,
round(avg(payment_value),2)  as average
from `Project_Target.customers`c
join `Project_Target.orders`o
on c.customer_id = o.customer_id
join `Project_Target.payments `p
on o.order_id = p.order_id
group by customer_state
order by customer_state,total_sum,average
```



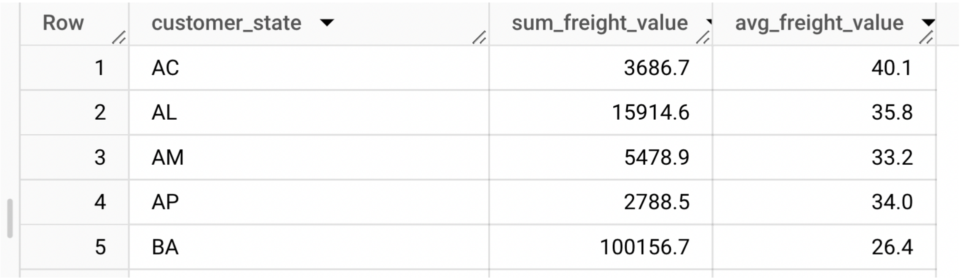
Row	customer_state	total_sum	average
1	AC	19680.62	234.29
2	AL	96962.06	227.08
3	AM	27966.93	181.6
4	AP	16262.8	232.33
5	BA	616645.82	170.82
6	CE	279464.03	199.9
7	DF	355141.08	161.13
8	ES	325967.55	154.71
9	GO	350092.31	165.76
10	MA	152523.02	198.86
11	MG	1872257.26	154.71
12	MS	137534.84	186.87
13	MT	187029.29	195.23

Results per page: 50

Insights : The average price in AC is the highest.

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

```
Select customer_state,round(sum(freight_value),1) as sum_freight_value,
round(avg(freight_value),1) as avg_freight_value
from `Project_Target.customers`c
join `Project_Target.orders`o
on c.customer_id = o.customer_id
join `Project_Target.orderitems`od
on od.order_id = o.order_id
group by customer_state
order by customer_state
```



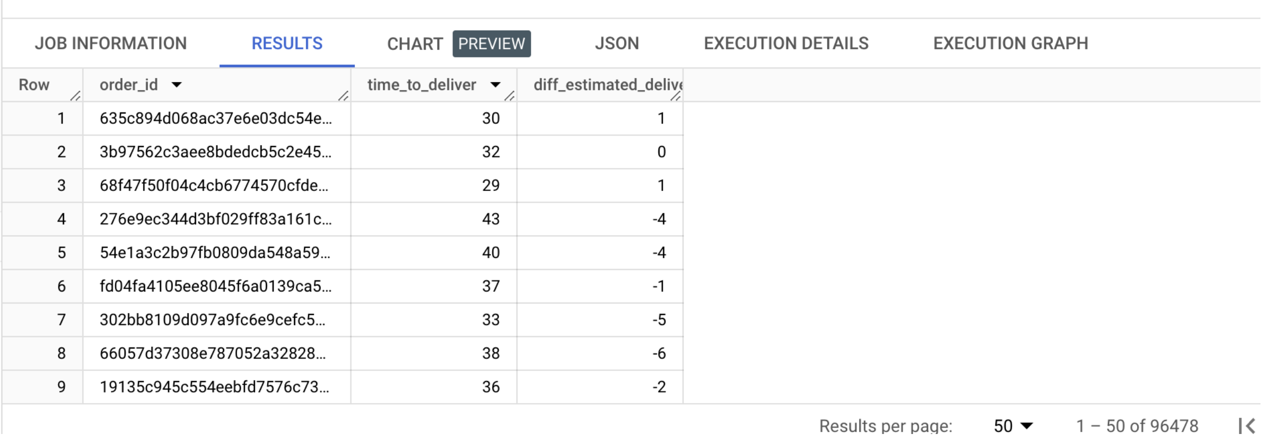
Row	customer_state	sum_freight_value	avg_freight_value
1	AC	3686.7	40.1
2	AL	15914.6	35.8
3	AM	5478.9	33.2
4	AP	2788.5	34.0
5	BA	100156.7	26.4

Insights: The output presents the sum and the average of the price rate at which a product is delivered from one point to another, grouped by states.

Analysis based on sales, freight and delivery time.

A. Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

```
Select order_id, 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 `Project_Target.orders`
where order_status = 'delivered'
```



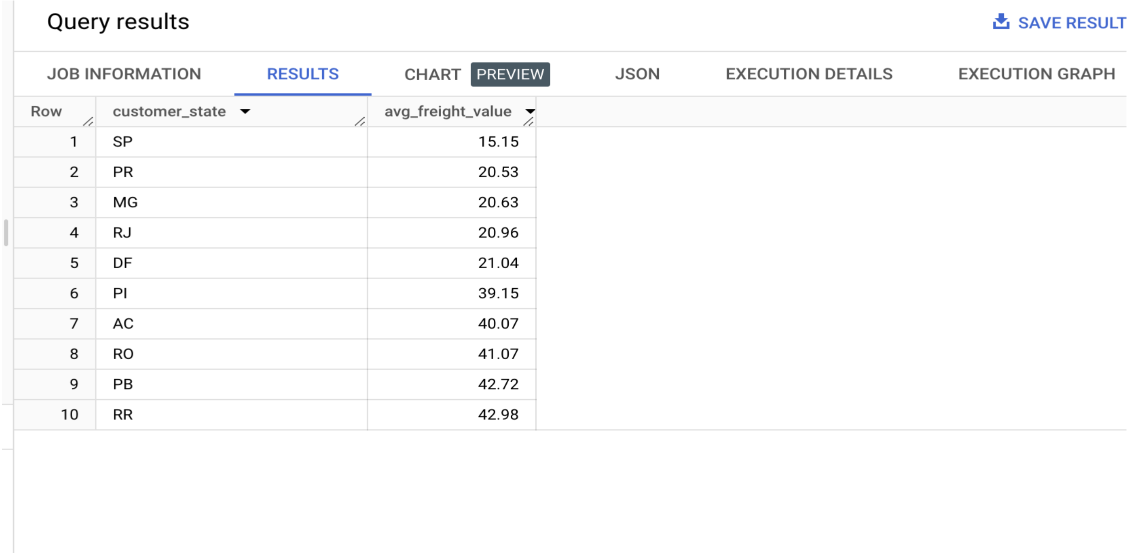
Row	order_id	time_to_deliver	diff_estimated_delive
1	635c894d068ac37e6e03dc54e...	30	1
2	3b97562c3aee8bdedcb5c2e45...	32	0
3	68f47f50f04c4cb6774570cfde...	29	1
4	276e9ec344d3bf029ff83a161c...	43	-4
5	54e1a3c2b97fb0809da548a59...	40	-4
6	fd04fa4105ee8045f6a0139ca5...	37	-1
7	302bb8109d097a9fc6e9cefc5...	33	-5
8	66057d37308e787052a32828...	38	-6
9	19135c945c554eebfd7576c73...	36	-2

Assumption : only those orders which have been delivered, are taken into consideration

Insights : The column time to deliver shows the time taken to deliver an order, if we work onto the aspects of delivery time and decrease the days, it can help, on better feedback and more happy customers.
The column diff_estimated_delivery presents the data on the estimated & actual delivery date of an order,for the order deliveries in negatives, those areas can be worked upon more on, delivery time, if the order is delivered before estimated time, this results in acquiring more customers and leading to better feedback.

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

```
Select a.customer_state,round(avg_freight_value,2) as avg_freight_value
from
(Select  c.customer_state,avg(od.freight_value) as avg_freight_value, dense_rank()
over(order by avg(od.freight_value)desc) as Heighest_avg,
dense_rank() over(order by avg(od.freight_value)asc) as lowest_avg
from `Project_Target.customers` c
join `Project_Target.orders` o
on c.customer_id = o.customer_id
join `Project_Target.orderitems` od
on o.order_id = od.order_id
group by c.customer_state)a
where Heighest_avg <= 5 or lowest_avg <=5
order by avg_freight_value
```

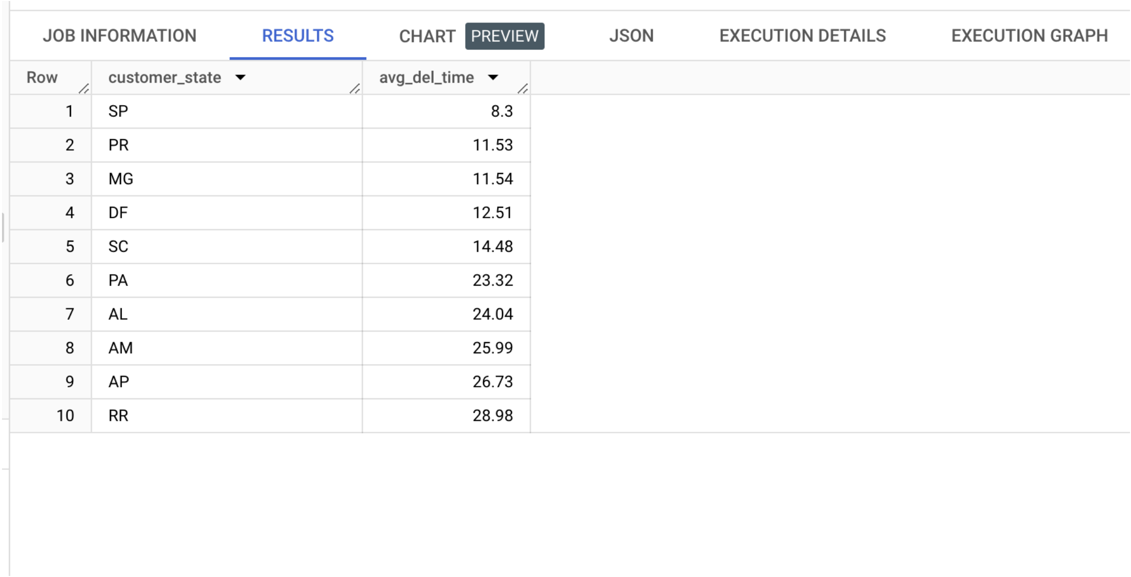


Row	customer_state	avg_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04
6	PI	39.15
7	AC	40.07
8	RO	41.07
9	PB	42.72
10	RR	42.98

Insights : The top rows (1-5) present the lowest average freight values, while the rows(6-10) show the highest average freight value.

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

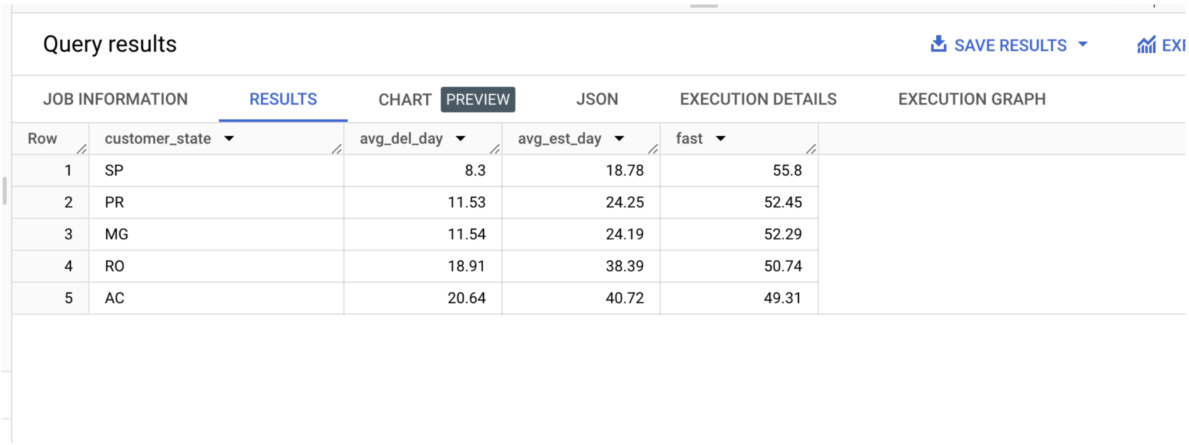
```
SELECT customer_state, t.avg_del_time
from
(Select customer_state,
round(avg(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,day)),2)
as avg_del_time,
dense_rank() over(order by
round(avg(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,day)),2)
desc) as high,
dense_rank() over(order by
round(avg(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,day)),2)
) as low
from `Project_Target.orders` o
join `Project_Target.customers` c
on o.customer_id = c.customer_id
where order_status = 'delivered'
group by customer_state)t
where high<= 5 or low <= 5
order by avg_del_time
```



Row	customer_state	avg_del_time
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48
6	PA	23.32
7	AL	24.04
8	AM	25.99
9	AP	26.73
10	RR	28.98

D. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
Select *,
round(((avg_est_date-avg_del_day)/avg_est_day)*100,2) as fast
from
(
Select c.customer_state,
round(avg(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,day)),2)
as avg_del_day,
round(avg(Date_DIFF(o.order_estimated_delivery_date,o.order_purchase_timestamp,day)),2)
as avg_est_day
from `Project_Target.orders` o
join `Project_Target.customers` c
on o.customer_id = c.customer_id
where order_status = 'delivered'
group by c.customer_state)t
order by fast desc
limit 5
```



Row	customer_state	avg_del_day	avg_est_day	fast
1	SP	8.3	18.78	55.8
2	PR	11.53	24.25	52.45
3	MG	11.54	24.19	52.29
4	RO	18.91	38.39	50.74
5	AC	20.64	40.72	49.31

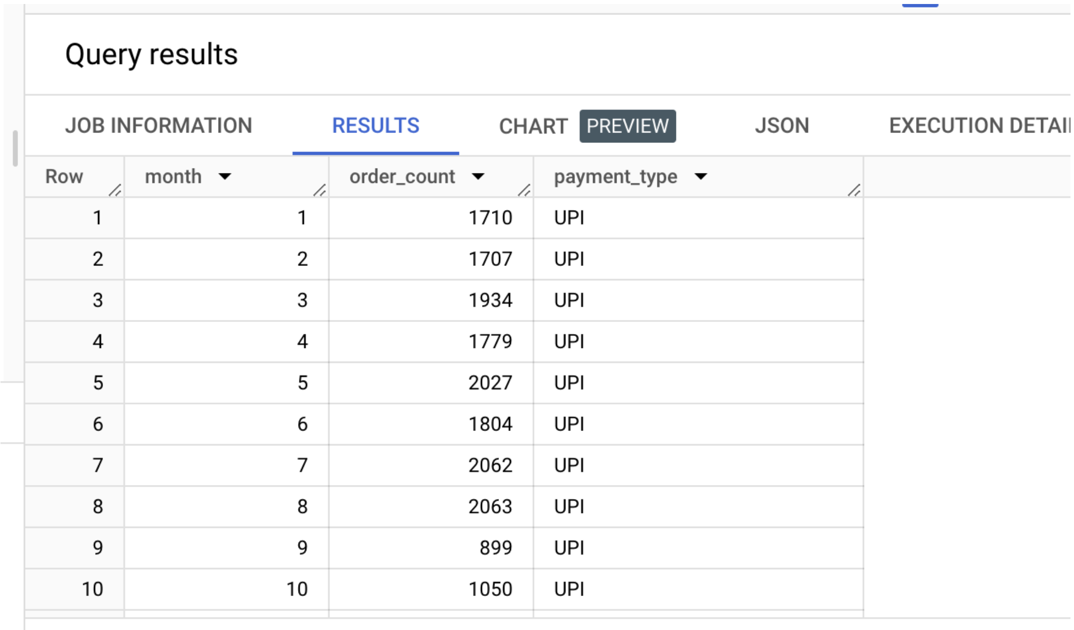
Insights : The table shows the fastest delivery time in the top 5 states.

VI Analysis based on the payments:

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

```
Select t.month,t.order_count,t.payment_type
from
(Select extract(MONTH from o.order_purchase_timestamp) as month,count(o.order_id) as
order_count,payment_type
from `Project_Target.orders`o
join `Project_Target.payments ` p
on o.order_id = p.order_id
where order_status != 'canceled'
group by month,payment_type
order by month) t

order by payment_type,month
```

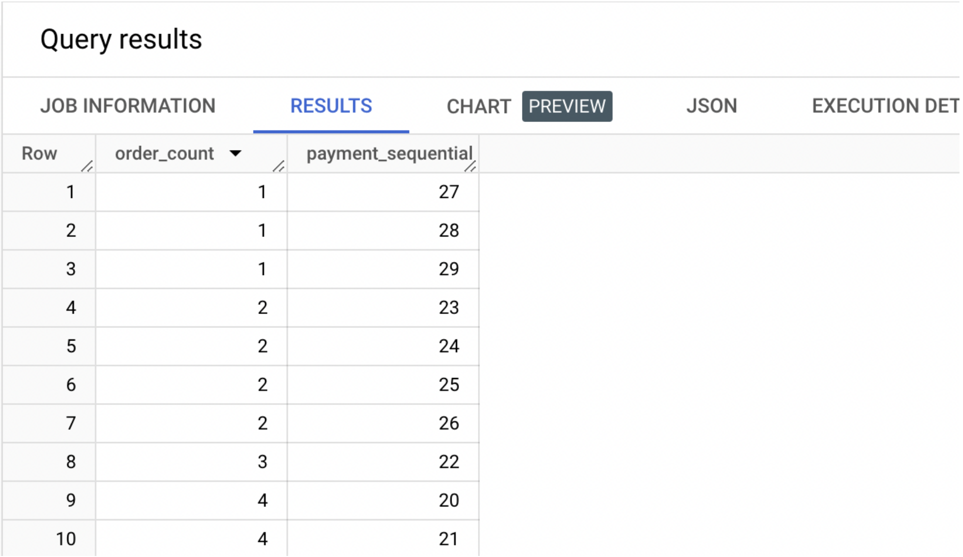


Row	month	order_count	payment_type
1	1	1710	UPI
2	2	1707	UPI
3	3	1934	UPI
4	4	1779	UPI
5	5	2027	UPI
6	6	1804	UPI
7	7	2062	UPI
8	8	2063	UPI
9	9	899	UPI
10	10	1050	UPI

Insights : The data shows the different types of payment method used by the customers month on month and how many orders have they placed using the payment method.

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

```
Select count(o.order_id) as order_count,payment_sequential
from `Project_Target.orders`o
join `Project_Target.payments `p
on o.order_id = p.order_id
where payment_sequential >= 1
group by 2
order by order_count,payment_sequential
```



Row	order_count	payment_sequential
1	1	27
2	1	28
3	1	29
4	2	23
5	2	24
6	2	25
7	2	26
8	3	22
9	4	20
10	4	21

Insights : The data shows , the number of orders bought on EMIs and where at least 1 installment by customer has been paid