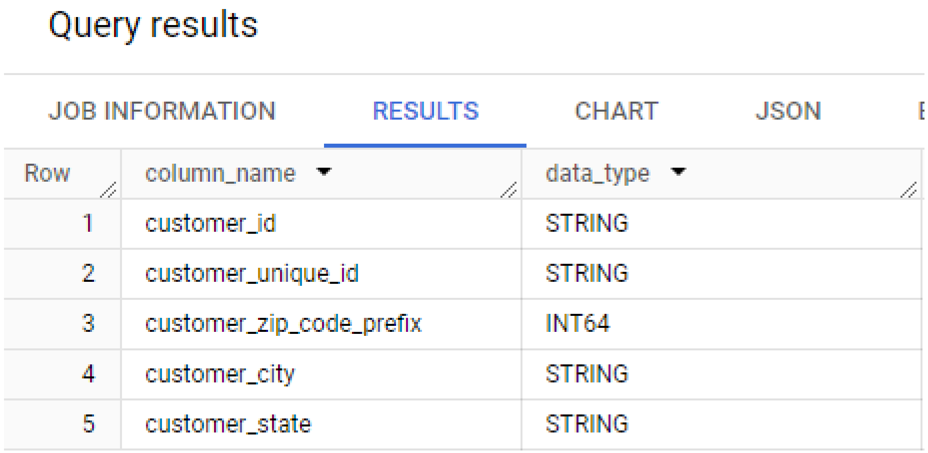
Exploratory Analysis

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

Q1 A.Data type of all columns in the “customers” table.

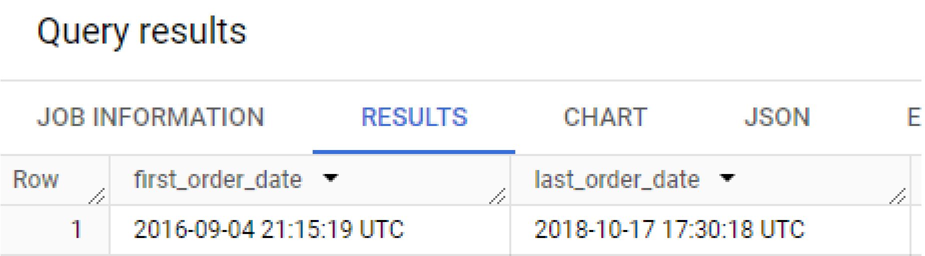
```
SELECT column_name , data_type
FROM target-project- 431704.shop_co.INFORMATION_SCHEMA.COLUMNS
where table_name = 'customers'
```



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

Q1 B. Get the time range between which the orders were placed.

```
    select min(order_purchase_timestamp) as first_order_date ,
max(order_purchase_timestamp) as last_order_date
from `target-project-431704.shop_co.orders`
```

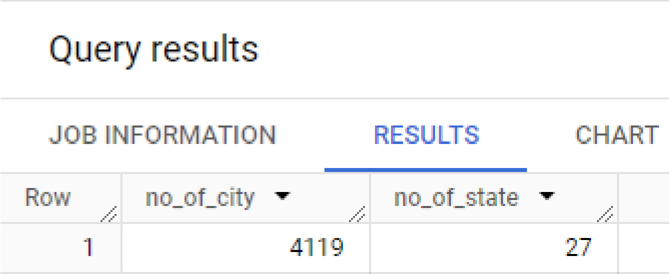


Row	first_order_date	last_order_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insight – From the above table we can see that the first order was placed on 2016 and the last order was placed on 2018.

Q1 C. Count the Cities & States of customers who ordered during the given period.

```
select count(distinct(customer_city )) as no_of_city ,
count(distinct(customer_state)) as
        no_of_state
from `shop_co.customers`
```



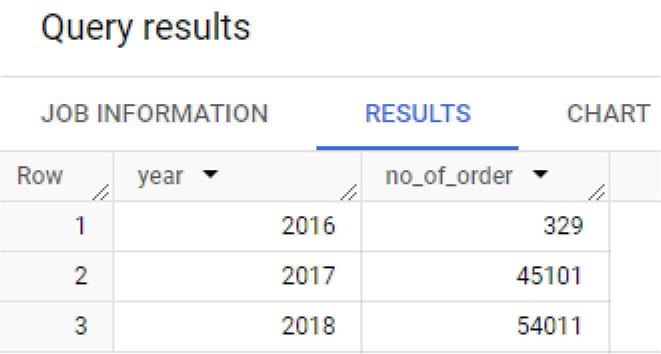
Row	no_of_city	no_of_state
1	4119	27

Insight – from the above table we can see that customer are from 27 different state and 4119 different cities. Here we can see that this is a Global market opportunity and we can get Feedback and Communication to understand their requirements.

Q2 In-depth Exploration:

Q2 A. Is there a growing trend in the no. of orders placed over the past years?

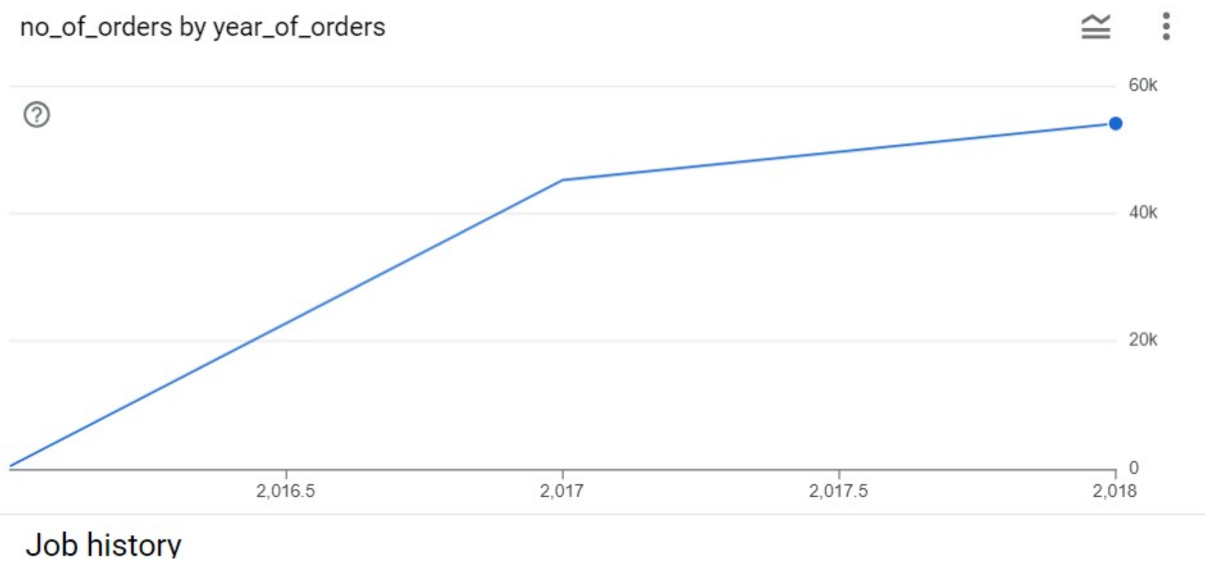
```
 select extract(year FROM order_purchase_timestamp) as year ,count(order_id)
as no_of_order

from `target-project-431704.shop_co.orders`
group by year
order by year asc
```



Row	year	no_of_order
1	2016	329
2	2017	45101
3	2018	54011

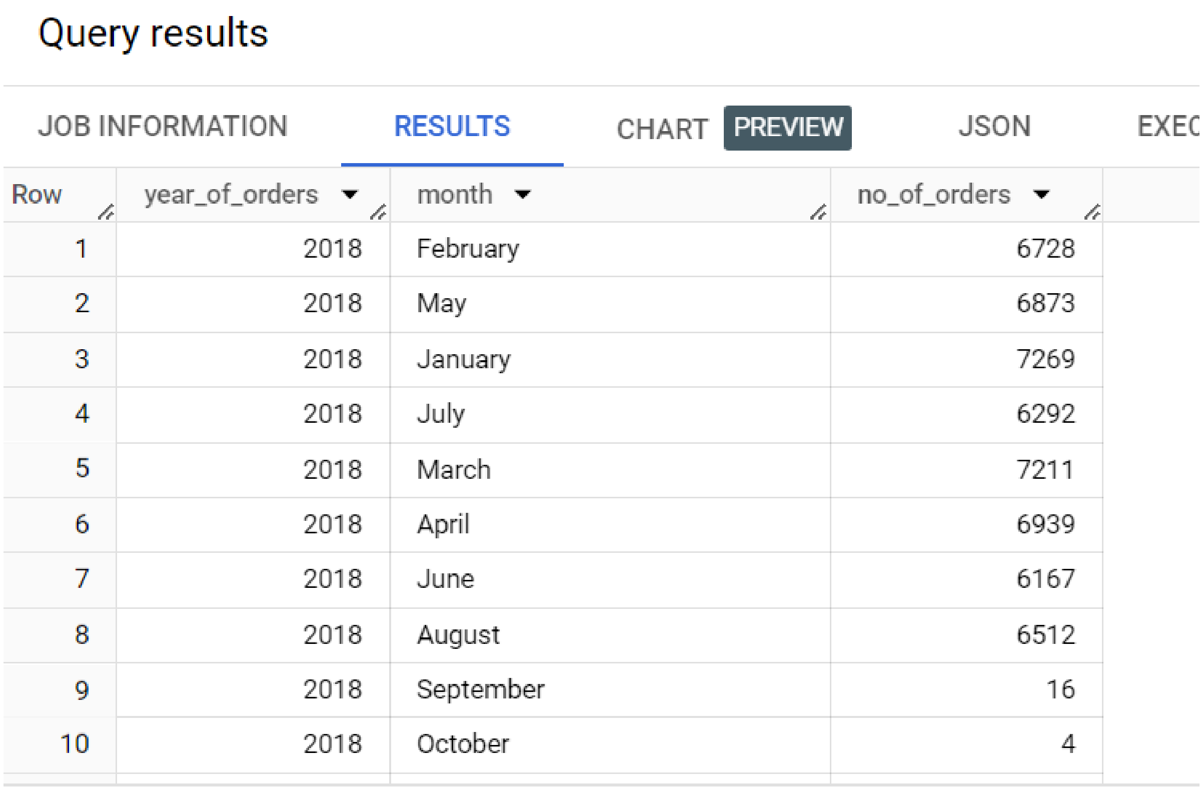
Insight – we can see that there is a increasing growth in the number of order on the basis of year.

So this is a **‘strategic growth’**.



Q2 B. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

```
select extract(year FROM order_purchase_timestamp  ) as
year_of_orders ,format_timestamp('%B',order_purchase_timestamp) as month ,
count(order_id) as no_of_orders
from shop_co.orders
group by year_of_orders,month
order by year_of_orders desc
```

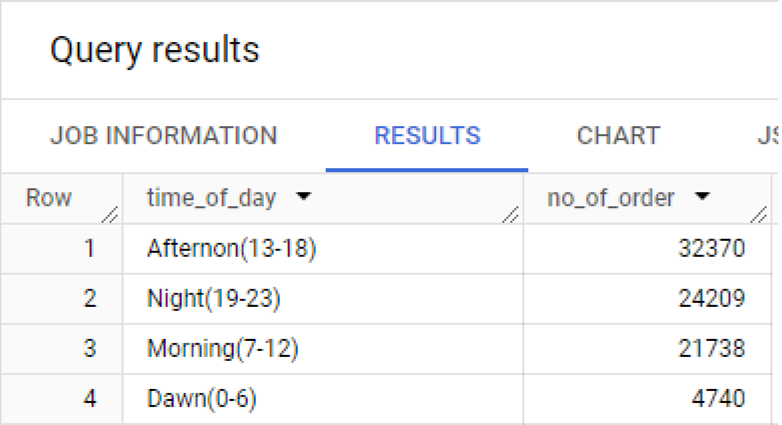


Row	year_of_orders	month	no_of_orders
1	2018	February	6728
2	2018	May	6873
3	2018	January	7269
4	2018	July	6292
5	2018	March	7211
6	2018	April	6939
7	2018	June	6167
8	2018	August	6512
9	2018	September	16
10	2018	October	4

Insight – from the data we see that there is an increment in number of order then there is a sudden drop in number of order afterward it again increase. This pattern is known as ‘**cyclic pattern’**

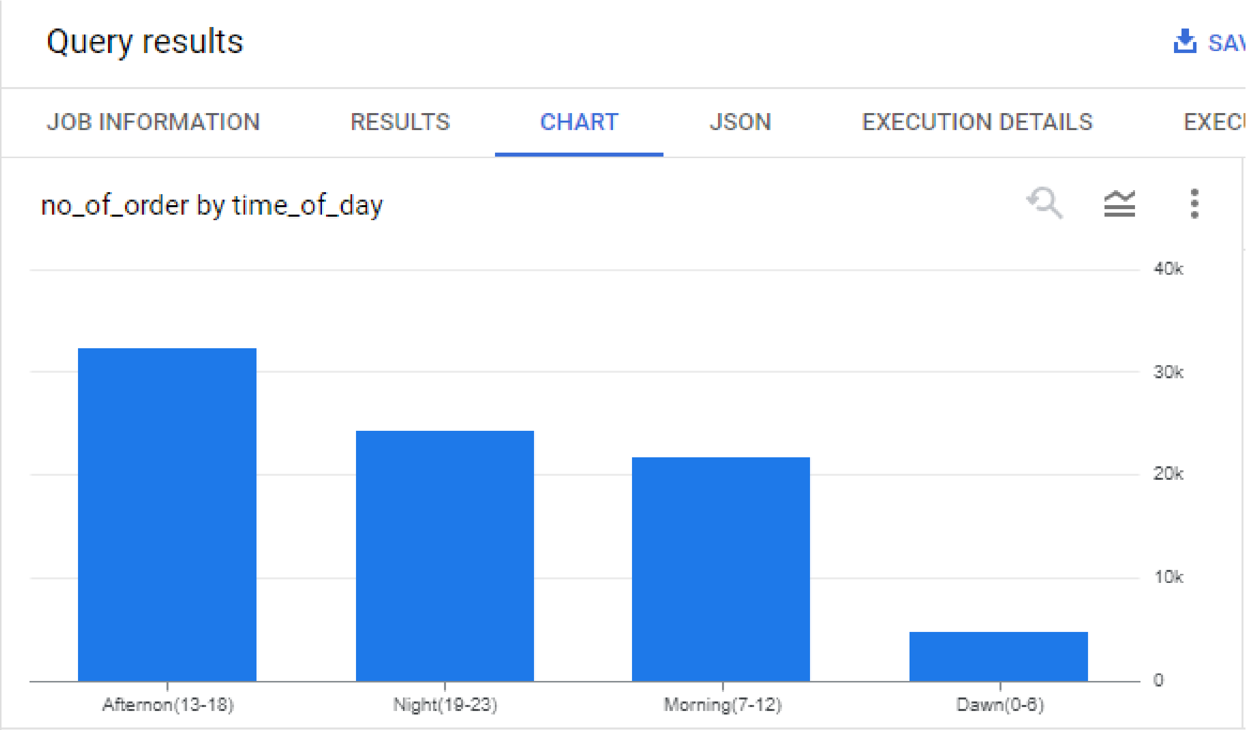
Q2 C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

```
select time_of_day , count(customer_id) as no_of_order
from (
select customer_id ,order_id ,
        case
        when extract(time from order_purchase_timestamp )
        between '00:00:00' and '06:00:00' then 'Dawn(0-6)'

        when extract(time from order_purchase_timestamp )
        between '07:00:00' and '12:00:00' then 'Morning(7-12)'

        when extract(time from order_purchase_timestamp)
        between '13:00:00' and '18:00:00' then 'Afternon(13-18)'
        when extract(time from order_purchase_timestamp)
        between '19:00:00' and '23:00:00' then 'Night(19-23)' end   as time_of_day
from shop_co.orders ) a
where time_of_day is not null
group by time_of_day
order by no_of_order desc
```



Row	time_of_day	no_of_order
1	Afternon(13-18)	32370
2	Night(19-23)	24209
3	Morning(7-12)	21738
4	Dawn(0-6)	4740

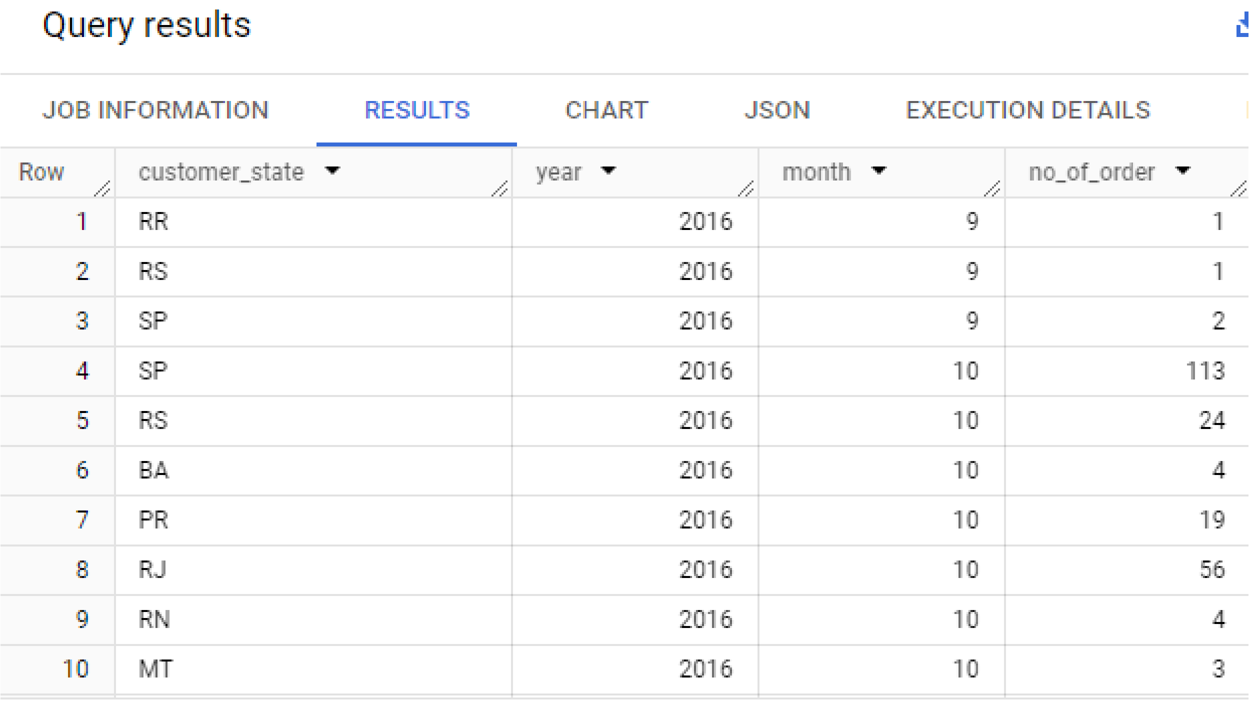
Insight - From the above data we can see that the maximum number of order placed during afternoon.It may be due to common time when people are available or due to lunch break



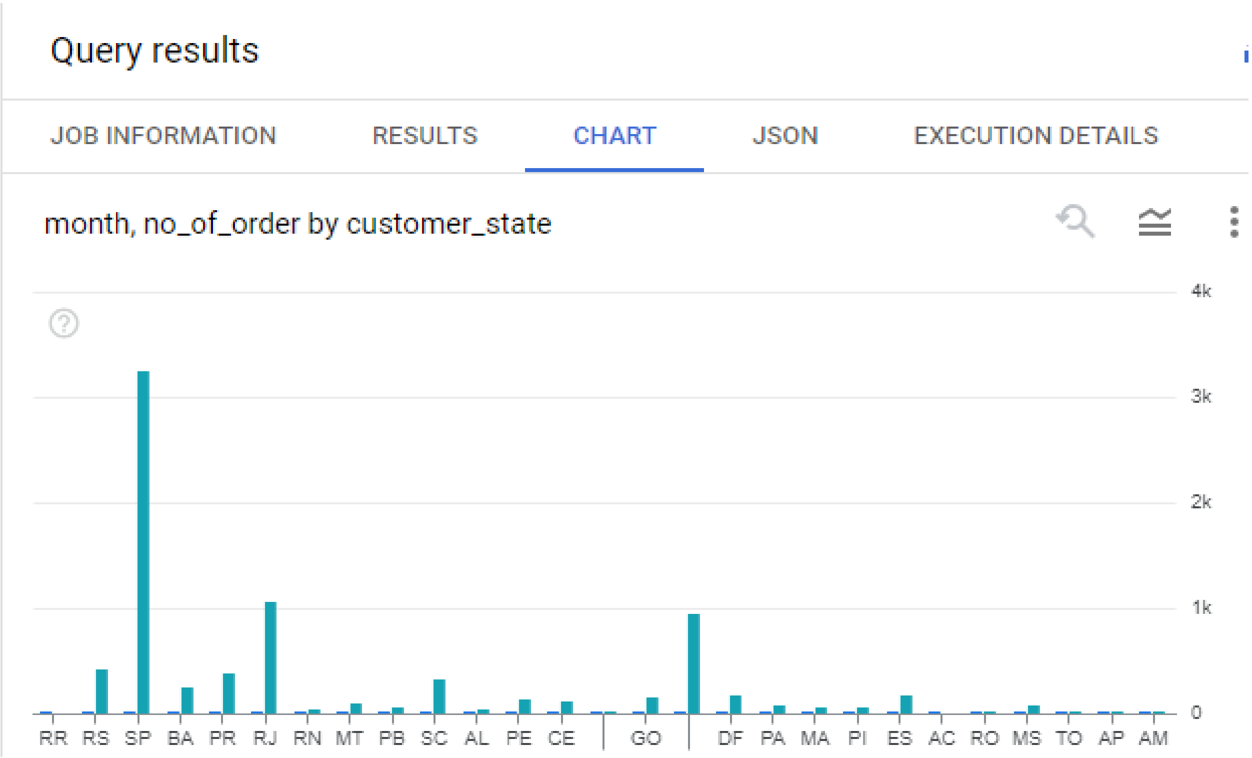
Q3. Evolution of E-commerce orders in region

Q3 A. Get the month on month no. of orders placed in each state.

```
select customer_state ,
extract(year from order_purchase_timestamp) as year ,
extract(month from order_purchase_timestamp) as month ,
count(a.customer_id) as no_of_order
from shop_co.customers a
inner join shop_co.orders b
on a.customer_id = b.customer_id
group by customer_state, year ,month
order by year , month
```



Row	customer_state	year	month	no_of_order
1	RR	2016	9	1
2	RS	2016	9	1
3	SP	2016	9	2
4	SP	2016	10	113
5	RS	2016	10	24
6	BA	2016	10	4
7	PR	2016	10	19
8	RJ	2016	10	56
9	RN	2016	10	4
10	MT	2016	10	3

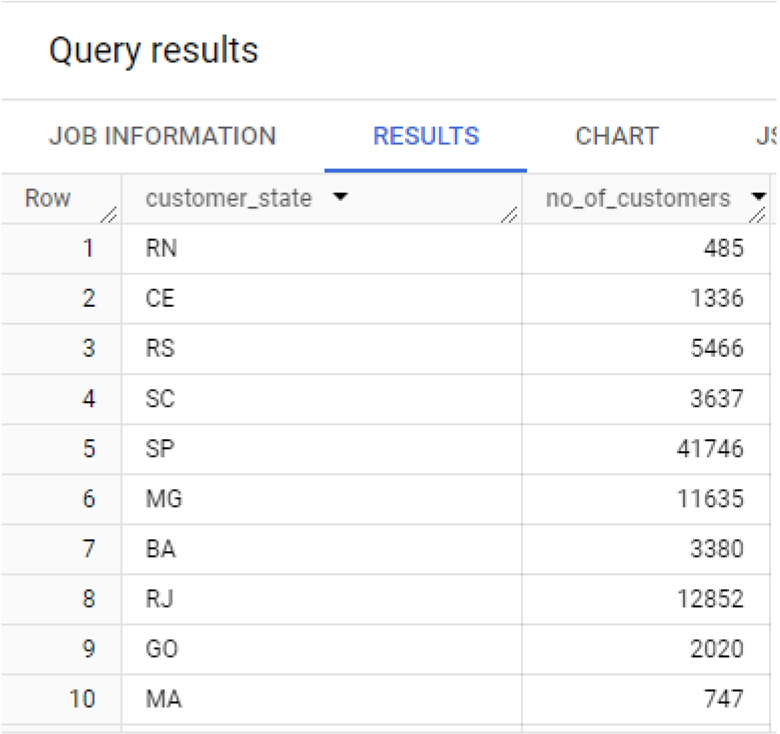


Query results

month, no_of_order by customer_state

Q3 B. How are the customers distributed across all the states?

```
select customer_state , count(distinct(customer_id)) as no_of_customers
from
`shop_co.customers`
group by customer_state
```



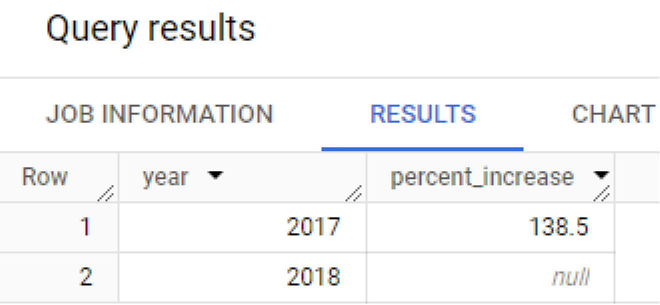
Query results

Row	customer_state	no_of_customers
1	RN	485
2	CE	1336
3	RS	5466
4	SC	3637
5	SP	41746
6	MG	11635
7	BA	3380
8	RJ	12852
9	GO	2020
10	MA	747

Q4 . Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

Q4 A. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

```
select year,
round( 100* (   ( lead(Total_financial_payment) over (order by year)-
    Total_financial_payment)   / Total_financial_payment ),1 ) as
percent_increase
from (
select
 extract(year from o.order_purchase_timestamp) as year,
 sum(p.payment_value) as Total_financial_payment
 from shop_co.orders o
 inner join shop_co.payments p
 on o.order_id = p.order_id
where o.order_purchase_timestamp between '2017-01-01' and '2017-08-31'
or o.order_purchase_timestamp between '2018-01-01' and '2018-08-31'
 group by year
 order by year asc)
```



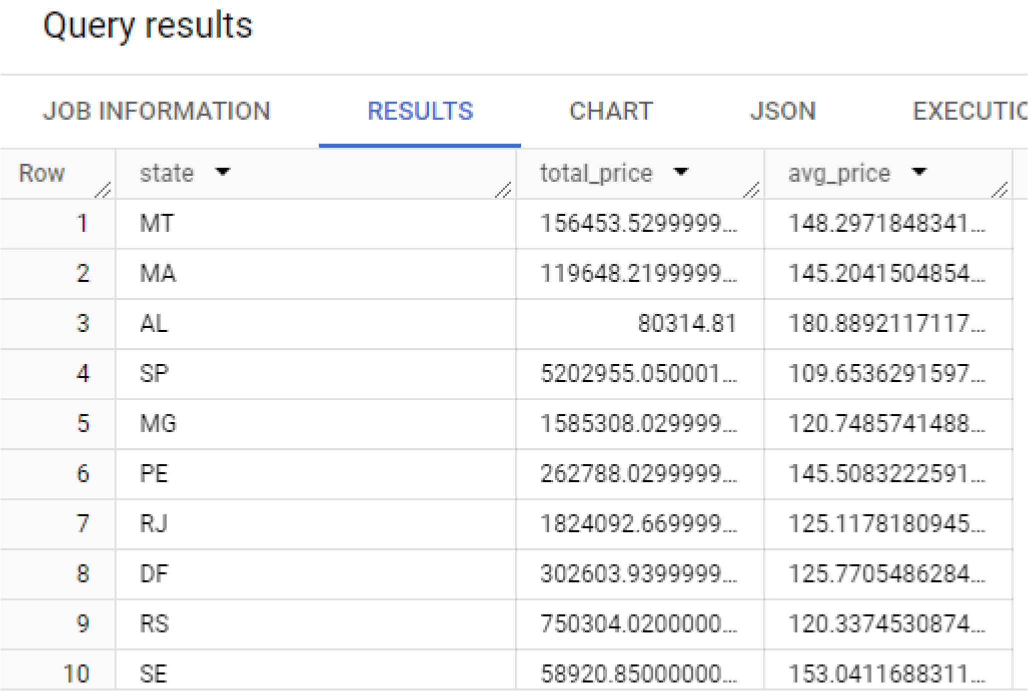
Row	year	percent_increase
1	2017	138.5
2	2018	null

Insight – there is an increment of 138.5% , the profit margin has increased means business growth and improved sales performance.

Q4 B. Calculate the Total & Average value of order price for each state.

```
select customer_state as state ,sum(price) as total_price , avg(price) as
avg_price
```

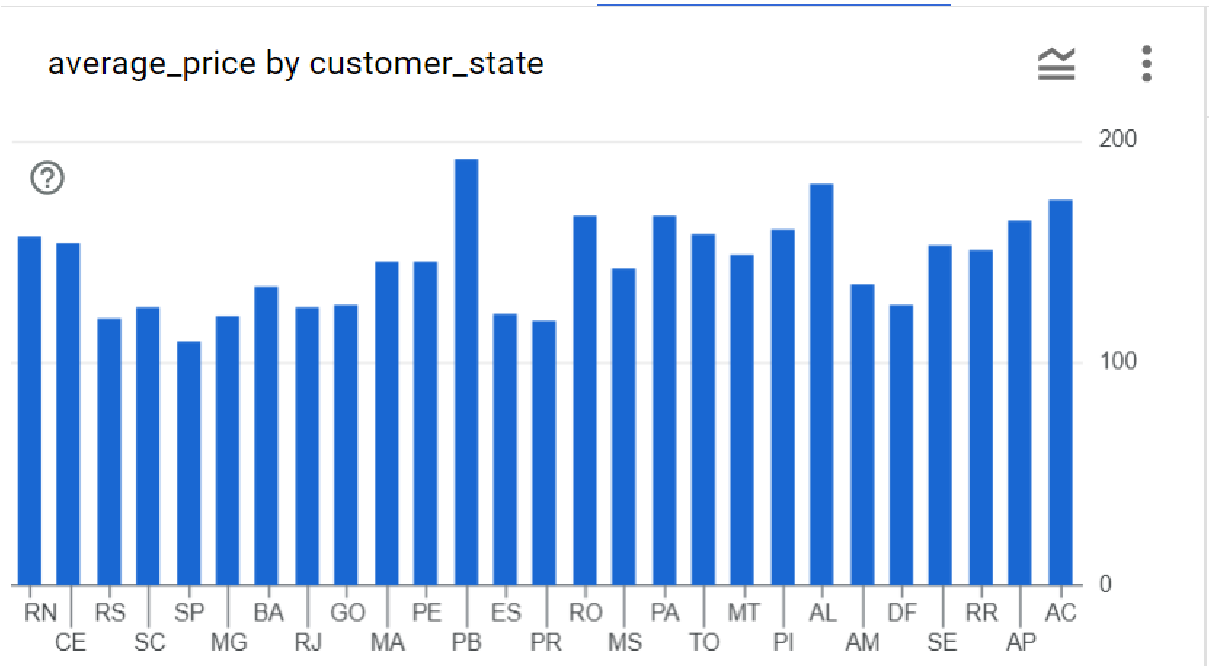
```
from `shop_co.customers` a
join shop_co.orders b
on a.customer_id = b.customer_id
join shop_co.order_items c
on b.order_id = c.order_id
group by state
```

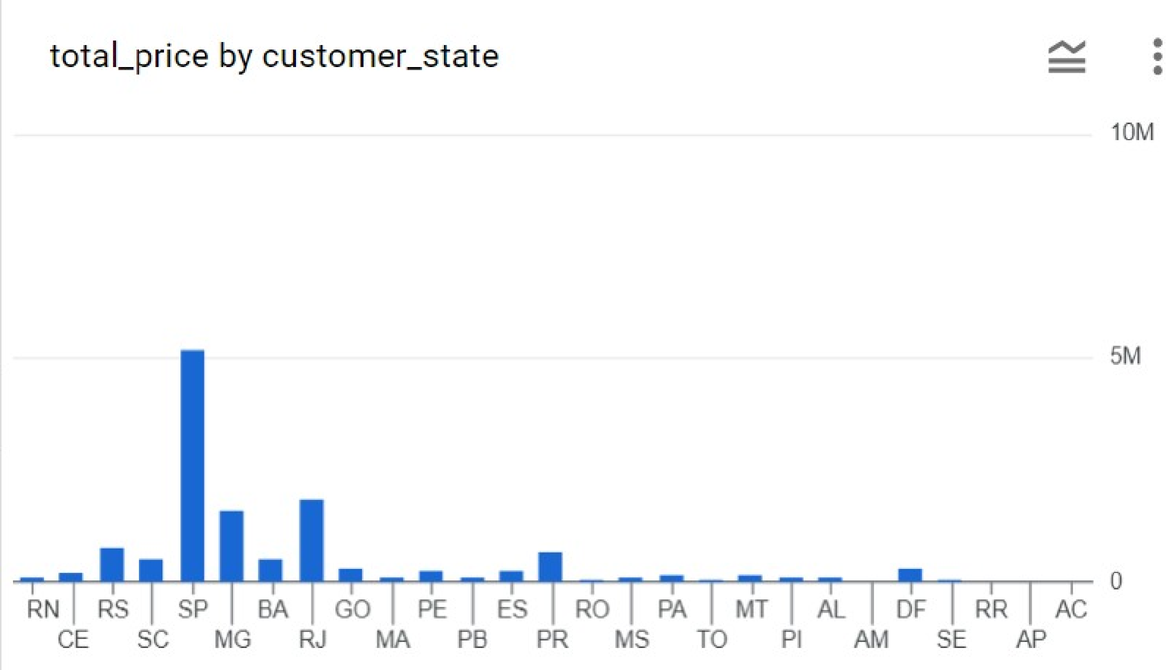


Query results

Row	state	total_price	avg_price
1	MT	156453.5299999...	148.2971848341...
2	MA	119648.2199999...	145.2041504854...
3	AL	80314.81	180.8892117117...
4	SP	5202955.050001...	109.6536291597...
5	MG	1585308.029999...	120.7485741488...
6	PE	262788.0299999...	145.5083222591...
7	RJ	1824092.669999...	125.1178180945...
8	DF	302603.9399999...	125.7705486284...
9	RS	750304.0200000...	120.3374530874...
10	SE	58920.85000000...	153.0411688311...

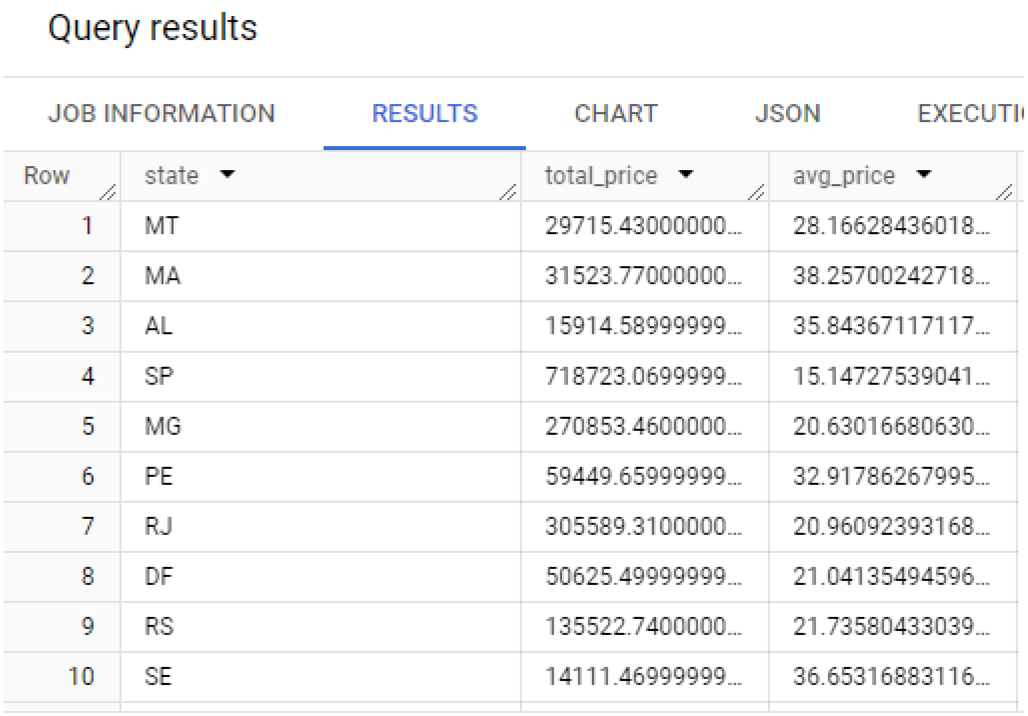
Insight – the average is above 100 and we got 27 state with their average and sum





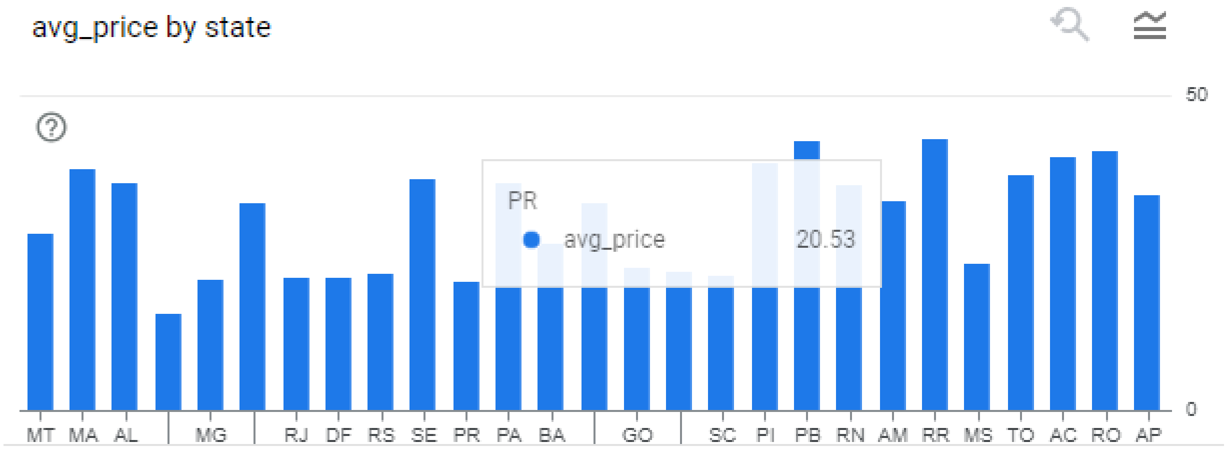
Q4 C. Calculate the Total & Average value of order freight for each state.

```
select customer_state as state ,sum(freight_value) as total_price ,
 avg(freight_value) as avg_price
from `shop_co.customers` a
join shop_co.orders b
on a.customer_id = b.customer_id
join shop_co.order_items c
on b.order_id = c.order_id
group by state
```

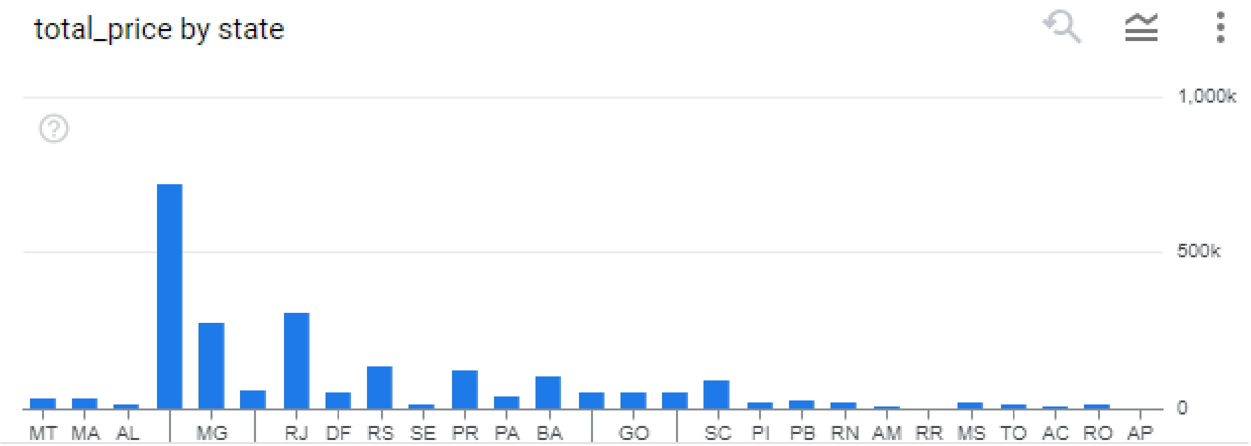


Query results

Row	state	total_price	avg_price
1	MT	29715.43000000…	28.16628436018…
2	MA	31523.77000000…	38.25700242718…
3	AL	15914.58999999…	35.84367117117…
4	SP	718723.0699999…	15.14727539041…
5	MG	270853.4600000…	20.63016680630…
6	PE	59449.65999999…	32.91786267995…
7	RJ	305589.3100000…	20.96092393168…
8	DF	50625.49999999…	21.04135494596…
9	RS	135522.7400000…	21.73580433039…
10	SE	14111.46999999…	36.65316883116…



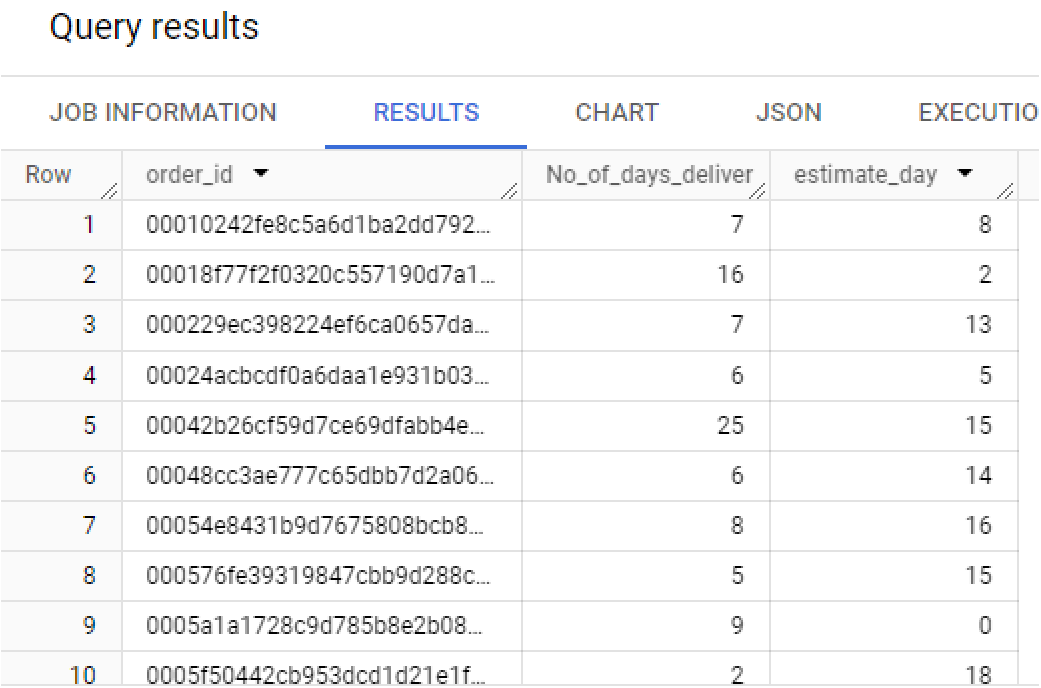
avg_price by state



Q5. Analysis based on sales, freight and delivery time.

Q5 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.

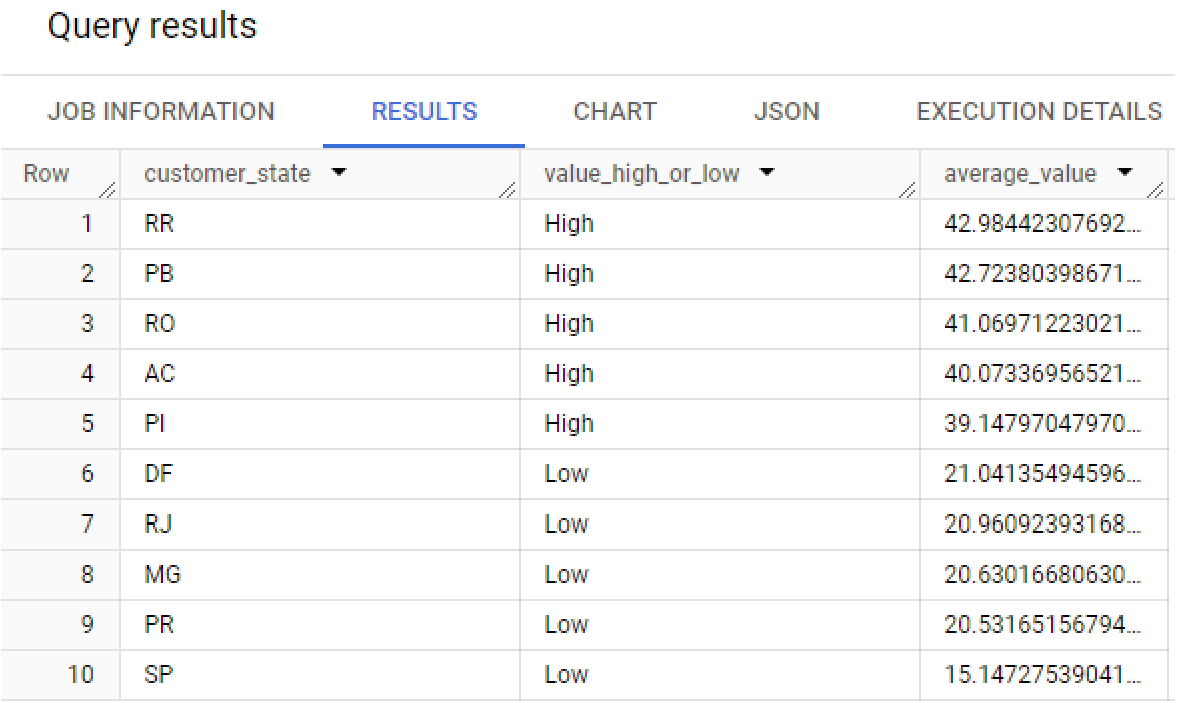
```
select order_id , timestamp_diff( order_delivered_customer_date ,
      order_purchase_timestamp , day ) as No_of_days_deliver,
      timestamp_diff(order_estimated_delivery_date,
      order_delivered_customer_date ,day) as estimate_day
from shop_co.orders
where order_delivered_customer_date is not null and
     order_estimated_delivery_date is not null
order by order_id
```



Row	order_id	No_of_days_deliver	estimate_day
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

Q5 B. Find out the top 5 states with the highest & lowest average freight value

```
( select customer_state, 'High' as value_high_or_low , avg(freight_value) as
average_value
from shop_co.customers a
join shop_co.orders b
on a.customer_id = b.customer_id
join shop_co.order_items c
on b.order_id = c.order_id
group by customer_state
order by average_value desc
limit 5 )

union all

(select customer_state, 'Low' as value_high_or_low , avg(freight_value) as
average_value
from shop_co.customers a
join shop_co.orders b
on a.customer_id = b.customer_id
join shop_co.order_items c
on b.order_id = c.order_id
group by customer_state
order by average_value asc
limit 5 )
order by average_value desc
```

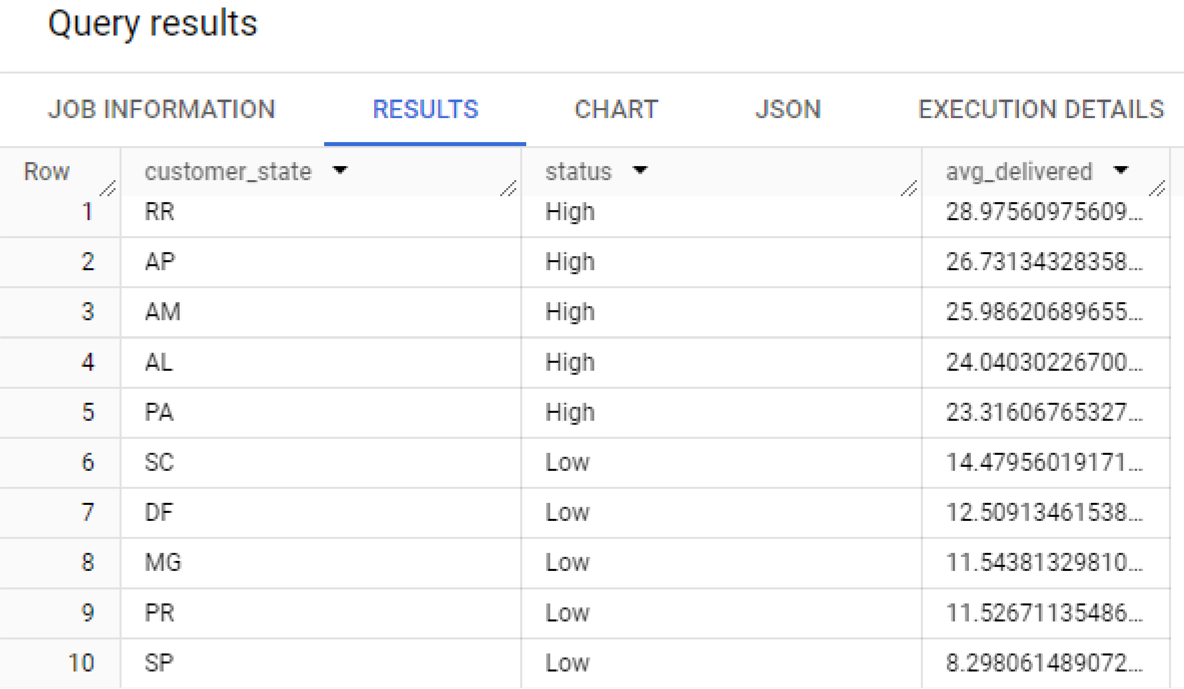


Row	customer_state	value_high_or_low	average_value
1	RR	High	42.98442307692...
2	PB	High	42.72380398671...
3	RO	High	41.06971223021...
4	AC	High	40.07336956521...
5	PI	High	39.14797047970...
6	DF	Low	21.04135494596...
7	RJ	Low	20.96092393168...
8	MG	Low	20.63016680630...
9	PR	Low	20.53165156794...
10	SP	Low	15.14727539041...

Insight – here 5 state have high average value and 5 state with low average value

Q5 C. Find out the top 5 states with the highest & lowest average delivery time

```
(select  customer_state ,'High' as status ,avg(timestamp_diff(
order_delivered_customer_date ,
 order_purchase_timestamp , day ) ) as avg_delivered
from shop_co.orders a
join shop_co.customers b
on a.customer_id = b.customer_id
group by customer_state
order by avg_delivered desc
limit 5)

union all
(
select  customer_state ,'Low' as status ,avg(timestamp_diff(
order_delivered_customer_date ,
 order_purchase_timestamp , day ) ) as avg_delivered
from shop_co.orders a
join shop_co.customers b
on a.customer_id = b.customer_id
group by customer_state
order by avg_delivered asc
limit 5 )
order by avg_delivered desc
```

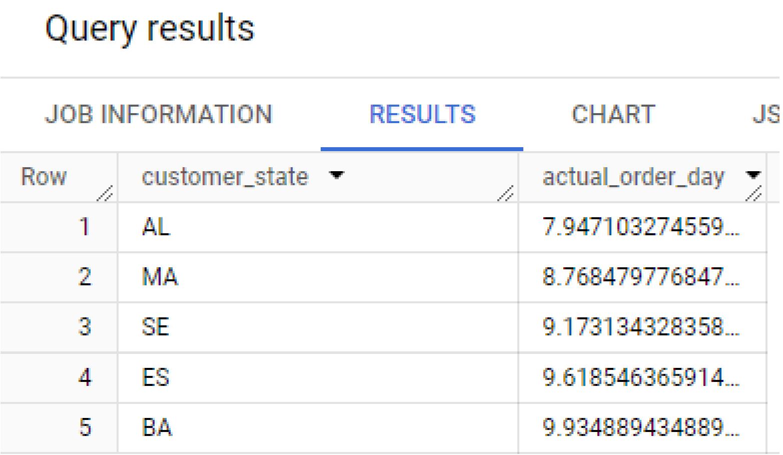


Row	customer_state	status	avg_delivered
1	RR	High	28.97560975609...
2	AP	High	26.73134328358...
3	AM	High	25.98620689655...
4	AL	High	24.04030226700...
5	PA	High	23.31606765327...
6	SC	Low	14.47956019171...
7	DF	Low	12.50913461538...
8	MG	Low	11.54381329810...
9	PR	Low	11.52671135486...
10	SP	Low	8.298061489072...

Insight - Here we have top 5 highest and lowest average on the basis of states.so can see that there is Logistics Challenges and Operational Efficiency.

Q5 D.Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
select customer_state , avg(timestamp_diff(order_estimated_delivery_date ,
order_delivered_customer_date,day) ) as actual_order_day
from shop_co.orders a
join shop_co.customers b
on a.customer_id = b.customer_id
group by customer_state
order by actual_order_day asc
limit 5
```

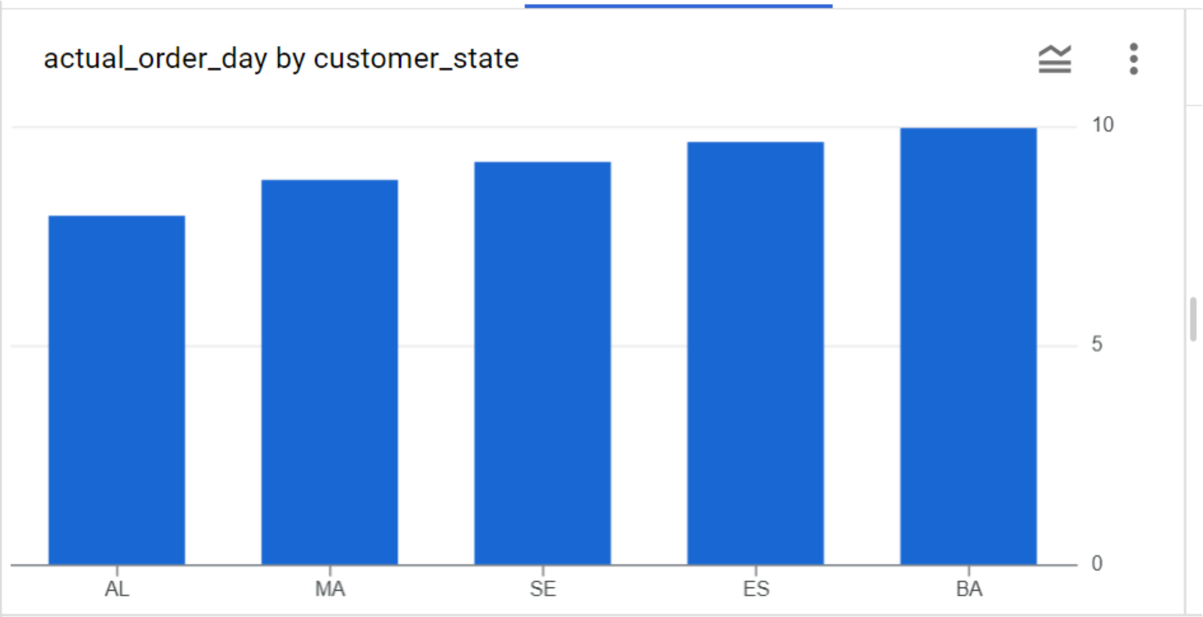


Query results

Row	customer_state	actual_order_day
1	AL	7.947103274559...
2	MA	8.768479776847...
3	SE	9.173134328358...
4	ES	9.618546365914...
5	BA	9.934889434889...

Insight – these are the top 5 states where the order delivery is really fast means that it have a good

Operational Efficiency and supply chain optimization


actual_order_day by customer_state
AL
MA
SE
ES
BA