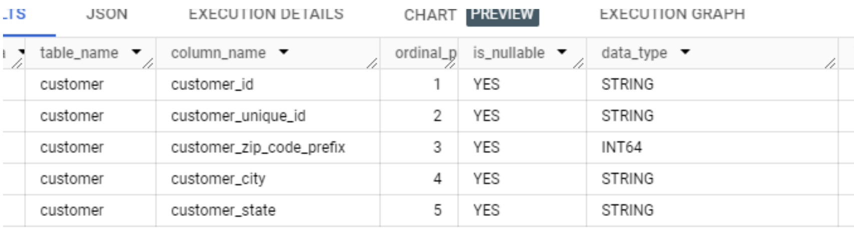
TARGET (Case Study)

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.*

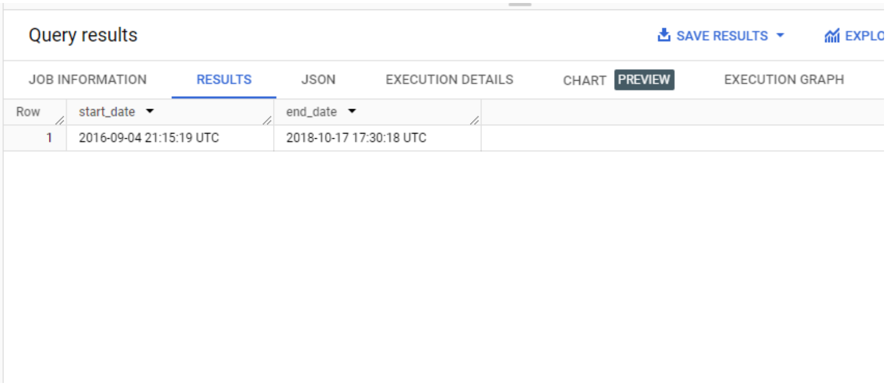
```
SELECT *,
data_type
FROM `scaler-dsml-sql-396813.target.INFORMATION_SCHEMA.COLUMNS`
WHERE
table_name = 'customer'
```



table_name	column_name	ordinal_p	is_nullable	data_type
customer	customer_id	1	YES	STRING
customer	customer_unique_id	2	YES	STRING
customer	customer_zip_code_prefix	3	YES	INT64
customer	customer_city	4	YES	STRING
customer	customer_state	5	YES	STRING

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

```
select
min(order_purchase_timestamp) as start_date ,max(order_purchase_timestamp) as end_date
from `target.orders`
limit 10;
```



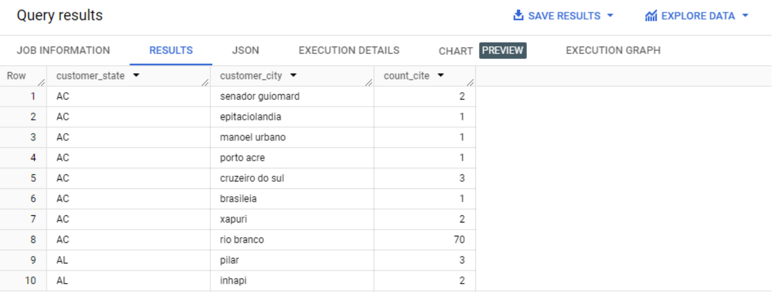
Row	start_date	end_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights:

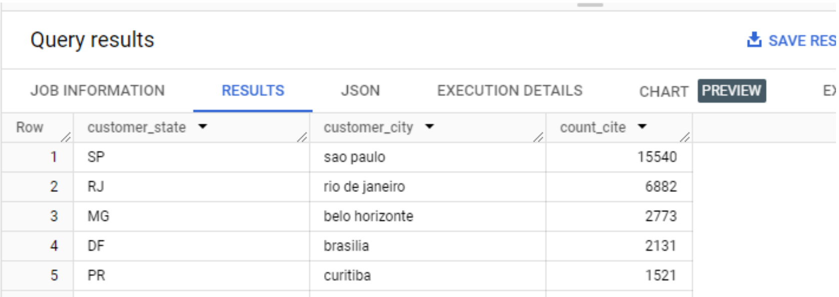
The orders were places between the time rage of sep 2016 to oct 2018

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

```
Select customer_state,
customer_city,
count(customer_id)as count_cite
from `target.customer`
group by customer_city,customer_state
order by customer_state
limit 10;
```



Row	customer_state	customer_city	count_cite
1	AC	senador guiomard	2
2	AC	epitaciolandia	1
3	AC	manoel urbano	1
4	AC	porto acre	1
5	AC	cruzeiro do sul	3
6	AC	brasileia	1
7	AC	xapuri	2
8	AC	rio branco	70
9	AL	pilar	3
10	AL	inhapi	2



Row	customer_state	customer_city	count_cite
1	SP	sao paulo	15540
2	RJ	rio de janeiro	6882
3	MG	belo horizonte	2773
4	DF	brasilia	2131
5	PR	curitiba	1521

Insights:

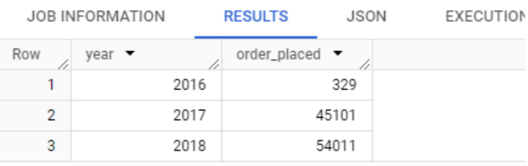
The maximum number of orders were placed in the city of sao paulo

In-depth Exploration:

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

Year-wise analysis:

```
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
COUNT(*) AS order_placed
FROM
`target.orders`
GROUP BY year
ORDER BY year
```



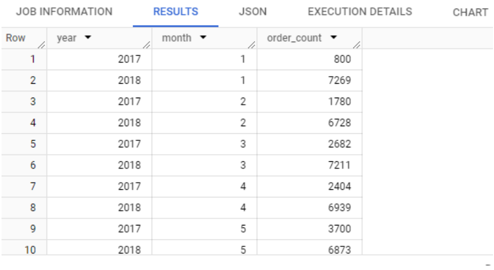
Row	year	order_placed
1	2016	329
2	2017	45101
3	2018	54011

Insights:

Yes, Year-wise analysis also indicates that there is a growing trend when compare to past years.

Month-wise analysis :

```
  SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
  EXTRACT(month FROM order_purchase_timestamp) AS month_each_year,
  COUNT(*) AS order_count
  FROM `target.orders`
  GROUP BY year, month_each_year
ORDER BY month_each_year,year
```



Row	year	month	order_count
1	2017	1	800
2	2018	1	7269
3	2017	2	1780
4	2018	2	6728
5	2017	3	2682
6	2018	3	7211
7	2017	4	2404
8	2018	4	6939
9	2017	5	3700
10	2018	5	6873

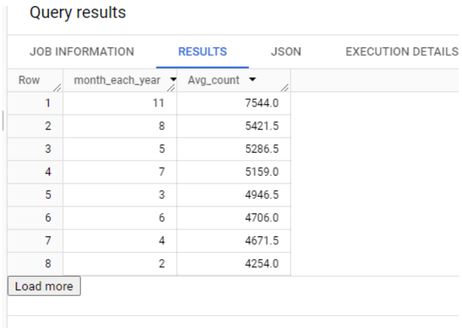
Insights:

Yes, the month-wise analysis indicates a growing trend when compared to the same month in the previous year.

Eg: In the month of 3(March) in 2018 the order count is 7211 and in the year 2017 the order count is 2682 which means there is a growth of 168.98%.

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

```
With cte as (
Select
EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month_each_year,
COUNT(*) AS order_count
FROM
`target.orders`
GROUP BY
year,month_each_year)
SELECT
month_each_year,
AVG(order_count) AS Avg_count
FROM
cte
GROUP BY
month_each_year
ORDER BY
Avg_count desc
```


Query results

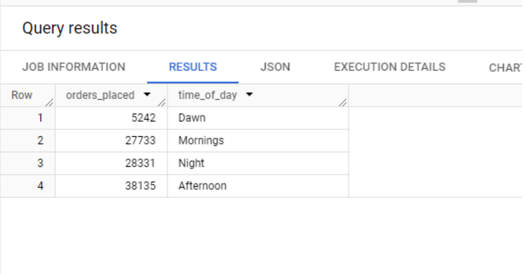
Row	month_each_year	Avg_count
1	11	7544.0
2	8	5421.5
3	5	5286.5
4	7	5159.0
5	3	4946.5
6	6	4706.0
7	4	4671.5
8	2	4254.0

Insights:

In the month of November the orders were placed more when compared to other months

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

```
select
count(order_id) as orders_placed,
case
when EXTRACT(HOUR FROM order_purchase_timestamp) <=6
then 'Dawn'
when EXTRACT(HOUR FROM order_purchase_timestamp) between 7 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 time_of_day
from `target.orders`
group by time_of_day
order by orders_placed;
```



Row	orders_placed	time_of_day
1	5242	Dawn
2	27733	Mornings
3	28331	Night
4	38135	Afternoon

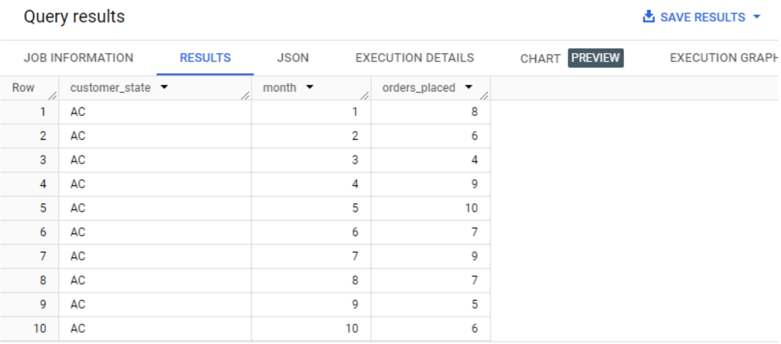
Insights:

Most of the orders were placed in the Afternoon time in the day and least orders were placed in the Dawn time.

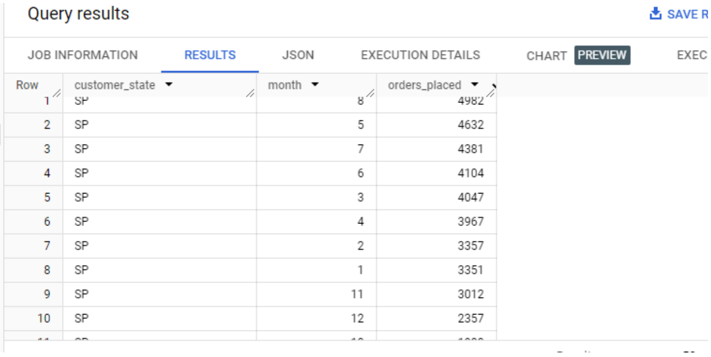
Evolution of E-commerce orders in the Brazil region:

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

```
Select c.customer_state,
extract(month from o.order_purchase_timestamp) as month,
count(o.order_id) as orders_placed
from `target.orders` as o
inner join
`target.customer` as c
on c.customer_id=o.customer_id
group by c.customer_state,month
order by customer_state,month
```



Row	customer_state	month	orders_placed
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6



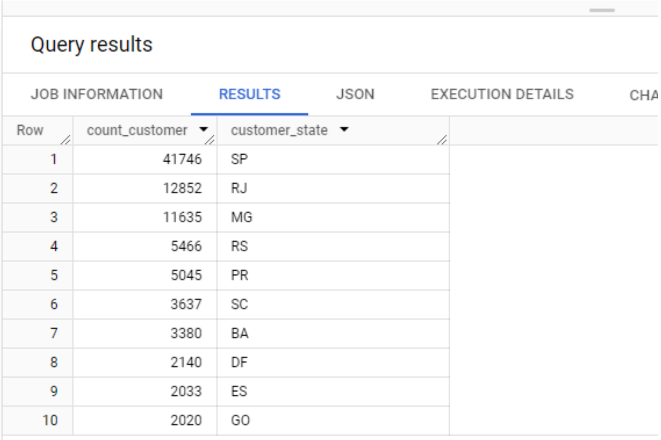
Row	customer_state	month	orders_placed
1	SP	8	4982
2	SP	5	4632
3	SP	7	4381
4	SP	6	4104
5	SP	3	4047
6	SP	4	3967
7	SP	2	3357
8	SP	1	3351
9	SP	11	3012
10	SP	12	2357

Insights:

In the above analysis we can get the insights that the maximum number of orders were placed in the month of august in the state of SP.

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

```
Select
Distinct count(customer_id) as count_customer,
customer_state
from `target.customer`
group by customer_state
order by count_customer desc
```



Row	count_customer	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

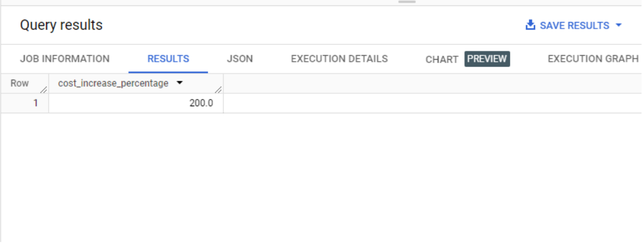
Insights:

The state SP contains the maximum number of customers.

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).*

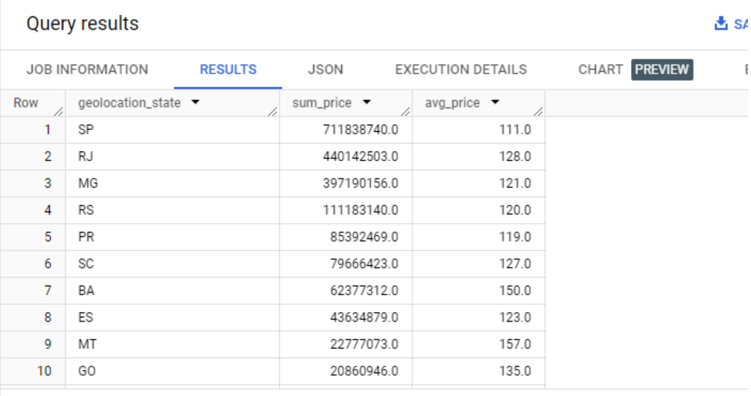
```
with cte as (Select
extract (year from order_purchase_timestamp) as year,
extract (month from order_purchase_timestamp) as month,
sum(p.payment_value) as Total_payment
from `target.orders` as o
join
`target.payments` as p
using(order_id)
group by year,month)

SELECT
100* round((SUM(
  CASE WHEN year = 2018
  THEN Total_payment END) -
  SUM(CASE WHEN year = 2017
  THEN Total_payment END)) /
  SUM(CASE WHEN year = 2017
  THEN Total_payment END)) AS cost_increase_percentage
from cte
where year in (2017,2018) and month between 1 and 7
```



Row	cost_increase_percentage
1	200.0

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

```
Select
g.geolocation_state,
round(sum(ot.price)) as sum_price,
round(avg(ot.price)) as avg_price
from `target.order_items` as ot
join `target.orders` as o
using(order_id)
join `target.customer` as c
using(customer_id)
join `target.geolocation` as g
on c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
group by g.geolocation_state
order by sum_price desc,avg_price
limit 10
```



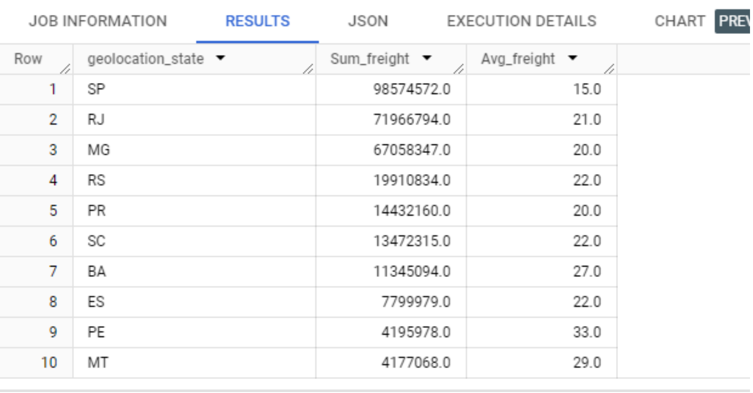
Row	geolocation_state	sum_price	avg_price
1	SP	711838740.0	111.0
2	RJ	440142503.0	128.0
3	MG	397190156.0	121.0
4	RS	111183140.0	120.0
5	PR	85392469.0	119.0
6	SC	79666423.0	127.0
7	BA	62377312.0	150.0
8	ES	43634879.0	123.0
9	MT	22777073.0	157.0
10	GO	20860946.0	135.0

Insights:

You can see states SP have the highest total order values, indicating the states with the most significant sales volume.

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

```
Select
g.geolocation_state,
round(sum(ot.freight_value)) as Sum_freight,
round(avg(ot.freight_value)) as Avg_freight
from `target.order_items` as ot
join `target.orders` as o
using(order_id)
join `target.customer` as c
using(customer_id)
join `target.geolocation` as g
on c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
group by g.geolocation_state
order by sum_freight desc,avg_freight
limit 10 ;
```

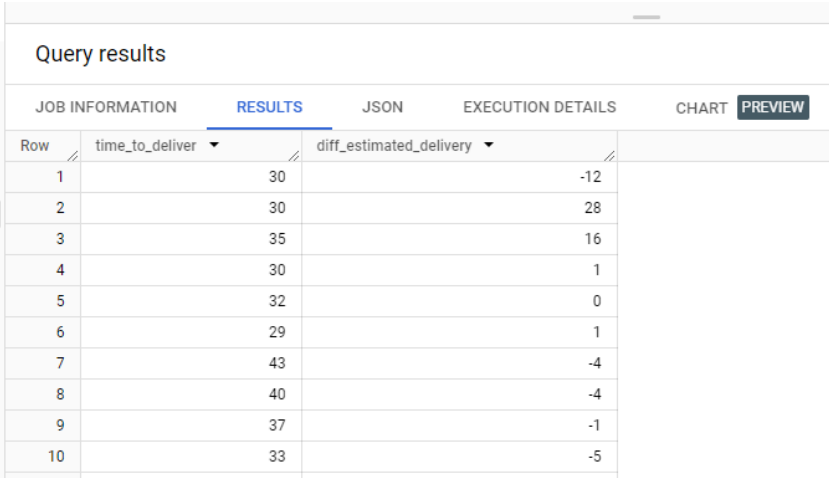


Row	geolocation_state	Sum_freight	Avg_freight
1	SP	98574572.0	15.0
2	RJ	71966794.0	21.0
3	MG	67058347.0	20.0
4	RS	19910834.0	22.0
5	PR	14432160.0	20.0
6	SC	13472315.0	22.0
7	BA	11345094.0	27.0
8	ES	7799979.0	22.0
9	PE	4195978.0	33.0
10	MT	4177068.0	29.0

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.

```
SELECT
  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.orders`
```



Row	time_to_deliver	diff_estimated_delivery
1	30	-12
2	30	28
3	35	16
4	30	1
5	32	0
6	29	1
7	43	-4
8	40	-4
9	37	-1
10	33	-5

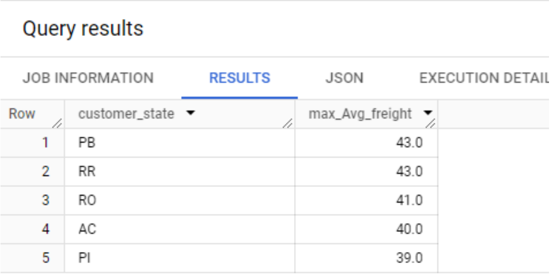
Insights:

Identify orders with long delivery time, indicates the delays or issue in the delivering the order.

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

Highest freight values:

```
Select
c.customer_state,
round(avg(ot.freight_value)) as max_Avg_freight
from `target.order_items` as ot
join `target.orders` as o
using(order_id)
join `target.customer` as c
using(customer_id)
group by c.customer_state
order by max_avg_freight desc
limit 5
```

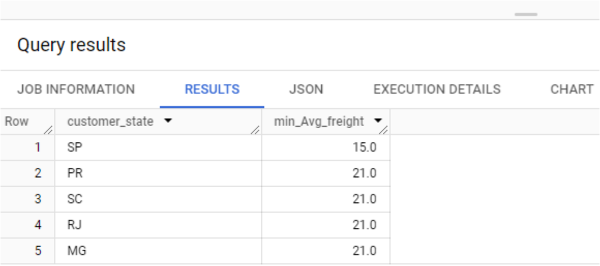


Row	customer_state	max_Avg_freight
1	PB	43.0
2	RR	43.0
3	RO	41.0
4	AC	40.0
5	PI	39.0

Highest freight values:

```
Select
c.customer_state,
round(avg(ot.freight_value)) as min_Avg_freight
from `target.order_items` as ot
join `target.orders` as o
using(order_id)
join `target.customer` as c
using(customer_id)
group by c.customer_state
```

```
order by min_avg_freight
limit 5
```



Row	customer_state	min_Avg_freight
1	SP	15.0
2	PR	21.0
3	SC	21.0
4	RJ	21.0
5	MG	21.0

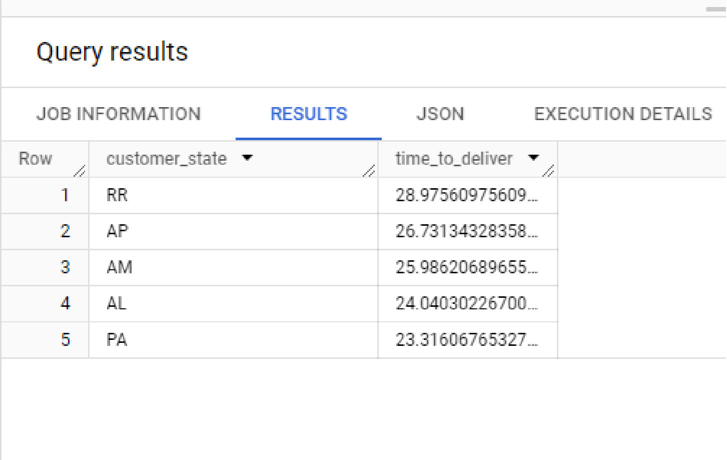
Insights:

States with lower average freight values may benefit from improved logistics and supply chain management to reduce costs and increase competitiveness.

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

Highest delivery time:

```
SELECT c.customer_state,
Avg(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS
time_to_deliver
FROM `target.orders` as o
join `target.customer`as c
using(customer_id)
group by c.customer_state
order by time_to_deliver desc
limit 5
```

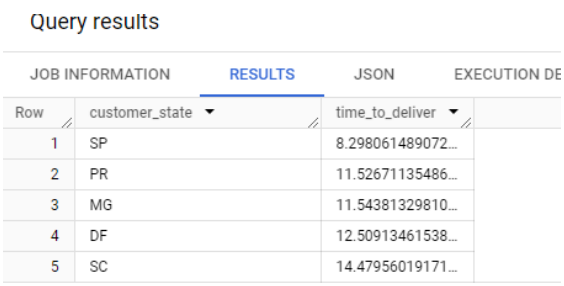


Row	customer_state	time_to_deliver
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...
4	AL	24.04030226700...
5	PA	23.31606765327...

Lowest delivery time:

```
SELECT c.customer_state,
Avg(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS
time_to_deliver
FROM `target.orders` as o
join `target.customer`as c
using(customer_id)
group by c.customer_state
order by time_to_deliver
```

```
limit 5
```

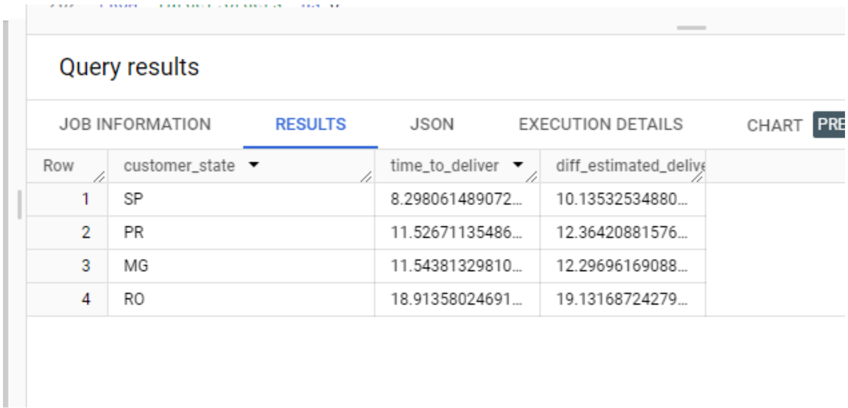


Row	customer_state	time_to_deliver
1	SP	8.298061489072...
2	PR	11.52671135486...
3	MG	11.54381329810...
4	DF	12.50913461538...
5	SC	14.47956019171...

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

Fast delivery:

```
SELECT c.customer_state,
Avg(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS
time_to_deliver,
Avg(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
diff_estimated_delivery
FROM `target.orders` as o
join `target.customer`as c
using(customer_id)
group by c.customer_state
having time_to_deliver < diff_estimated_delivery
order by time_to_deliver ,diff_estimated_delivery DESC;
```



Row	customer_state	time_to_deliver	diff_estimated_delive
1	SP	8.298061489072...	10.13532534880...
2	PR	11.52671135486...	12.36420881576...
3	MG	11.54381329810...	12.29696169088...
4	RO	18.91358024691...	19.13168724279...

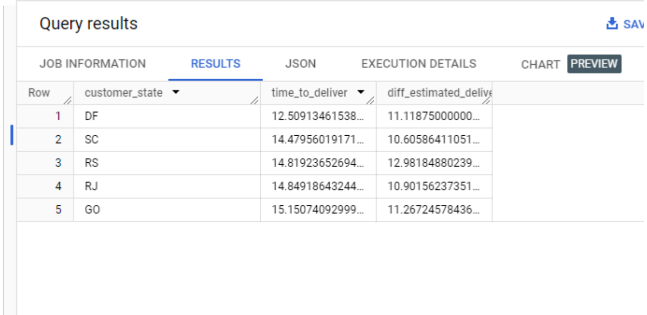
Insights:

Top states where the delivery was fast, means the orders were delivered quickly compare to the estimate time.

Slow delivery:

```
SELECT c.customer_state,
Avg(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS
time_to_deliver,
Avg(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
diff_estimated_delivery
FROM `target.orders` as o
join `target.customer`as c
```

```
using(customer_id)
group by c.customer_state
having time_to_deliver > diff_estimated_delivery
order by time_to_deliver ,diff_estimated_delivery
limit 5;
```

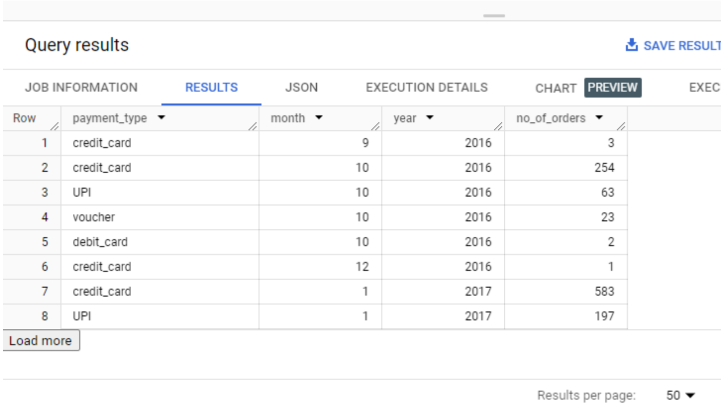


Row	customer_state	time_to_deliver	diff_estimated_delive
1	DF	12.50913461538...	11.11875000000...
2	SC	14.47956019171...	10.60586411051...
3	RS	14.81923652694...	12.98184880239...
4	RJ	14.84918643244...	10.90156237351...
5	GO	15.15074092999...	11.26724578436...

Analysis based on the payments

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

```
Select p.payment_type,
extract (month from o.order_purchase_timestamp) as month,
extract (year from o.order_purchase_timestamp) as year,
count(p.order_id) as no_of_orders
from `target.orders` as o
join
`target.payments` as p
using(order_id)
group by p.payment_type, month,year
order by year,month
```



Row	payment_type	month	year	no_of_orders
1	credit_card	9	2016	3
2	credit_card	10	2016	254
3	UPI	10	2016	63
4	voucher	10	2016	23
5	debit_card	10	2016	2
6	credit_card	12	2016	1
7	credit_card	1	2017	583
8	UPI	1	2017	197

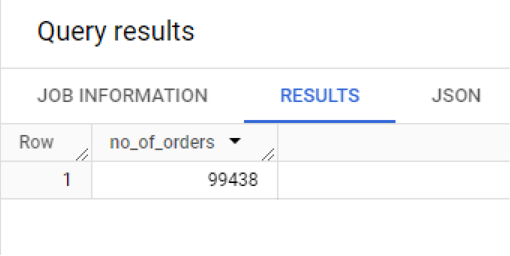
Insights:

Understanding which payment types are most commonly used each month can help businesses tailor their payment processing strategies.

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

```
SELECT
   COUNT(DISTINCT o.order_id) AS no_of_orders
FROM
   `target.orders` o
JOIN
```

```
  `target.payments` p ON o.order_id = p.order_id
WHERE
  p.payment_installments > 0;
```



Row	no_of_orders
1	99438

Insights:

More number of orders were placed in payment installments