Business Case Study

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

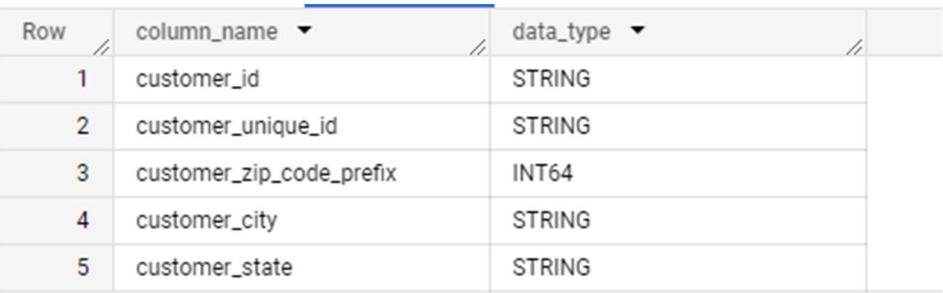
1.1 Data type of all columns in the "customers" table.

Sol:

QUERY: The below shows the query and output of the customer's table. The result consists of the column's name and its datatype.

```
SELECT
    column_name,data_type
  FROM
    `target-case-study-394414.case_study`. INFORMATION_SCHEMA.COLUMNS
  WHERE
    table_name = 'customers';
```

Result: The result contains the information of the column's name and its datatype.



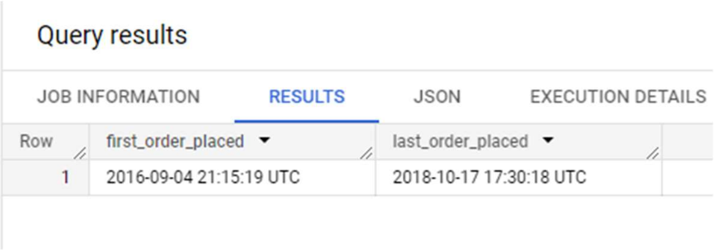
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

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

QUERY: The below query tells us about the timestamp at which the first and last orders were placed in the given time period.

```
select
min(order_purchase_timestamp) as first_order_placed,
max(order_purchase_timestamp) as last_order_placed
from `case_study.orders`
```

RESULT: The below result consists of the timestamp at which the first and last orders are placed in the provided time period.



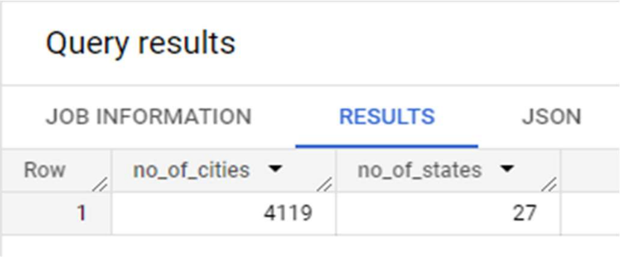
Row	first_order_placed	last_order_placed
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

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

QUERY:

```
select
count(distinct customer_city) as no_of_cities,
count(distinct customer_state) as no_of_states
from `case_study.customers`
```

RESULT:



Row	no_of_cities	no_of_states
1	4119	27

Insights and suggestions:

1. The business is expanded into 4119 cities and 27 states.
2. Still there are many more cities and states in Russia to capture the business.

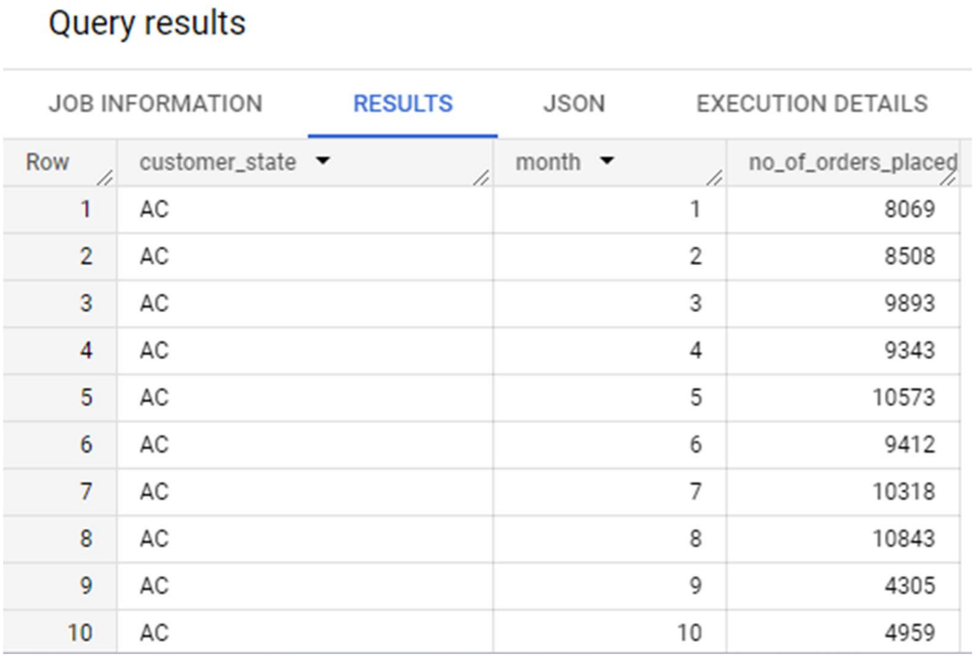
III. Evolution of E-commerce orders in the Brazil region.

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

QUERY

```
select distinct customer_state,
EXTRACT(month from order_purchase_timestamp) as month,
count(o.customer_id) over(partition by EXTRACT(month from
order_purchase_timestamp)) as no_of_orders_placed_in_each_month
from `case_study.orders` as o inner join `case_study.customers` as c on
o.customer_id = c.customer_id
order by customer_state, month
```

RESULT



Query results

Row	customer_state	month	no_of_orders_placed
1	AC	1	8069
2	AC	2	8508
3	AC	3	9893
4	AC	4	9343
5	AC	5	10573
6	AC	6	9412
7	AC	7	10318
8	AC	8	10843
9	AC	9	4305
10	AC	10	4959

Insights and suggestions:

1. The result shows the state in the month of May, June, and August have a greater number of orders.
2. We have to find out why the orders in the other months are lesser in number, analyze the people's needs and incorporate the products in our products.

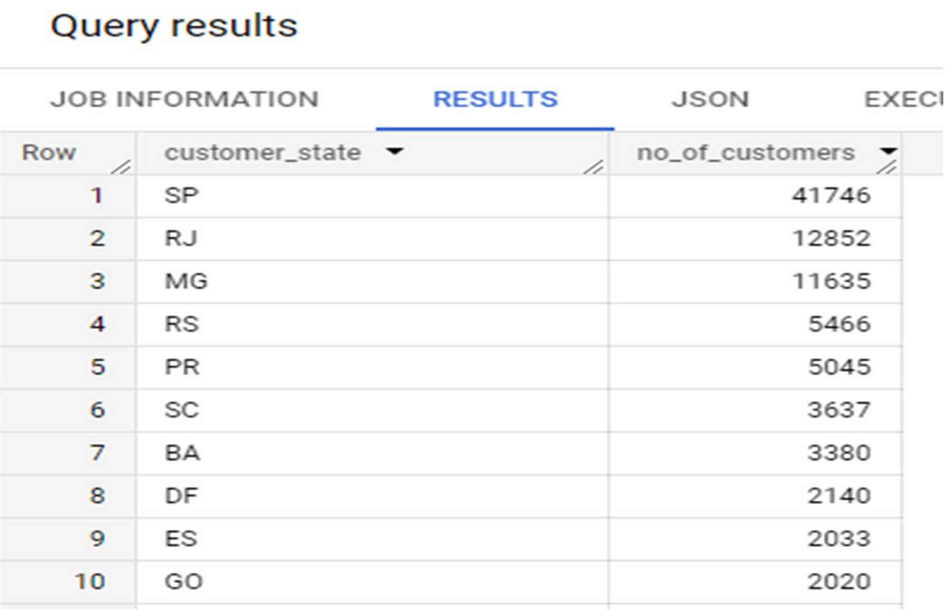
2. How are the customers distributed across all the states?

QUERY:

```
select

customer_state,
count(*) as no_of_customers
from `case_study.customers`
group by customer_state
order by no_of_customers desc
```

RESULT:



Row	customer_state	no_of_customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Insights and suggestions:

1. We could see from the above output the state SP, RJ, and MG have a higher number of customers.
2. We need to analyse the people in the other states and why the orders are in fewer numbers.
3. Get to know their requirements and update the list of products.
4. If most people are not aware of online shopping. Let’s run a campaign and introduce our apps and procedure to order.

6. Analysis based on the payments:

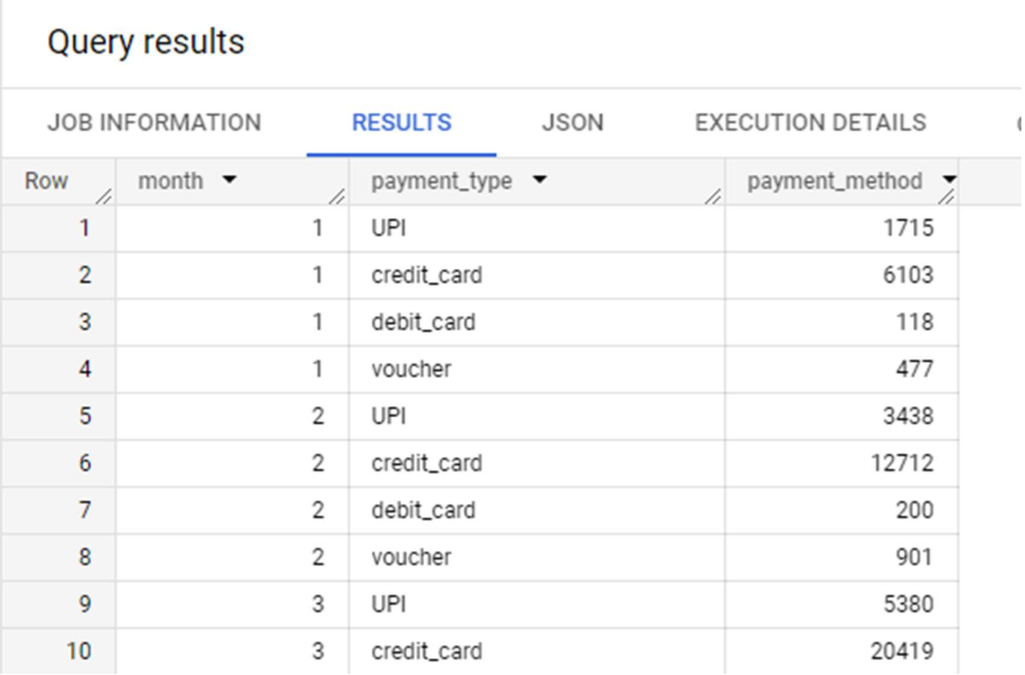
6.1 Find the month-on-month no. of orders placed using different payment types.

QUERY:

```
select distinct Extract(month from order_purchase_timestamp) as
 month,payment_type,

count(payment_type) over(partition by payment_type order by Extract(month from
order_purchase_timestamp)) as payment_method
from `case_study.payments` p inner join `case_study.orders` o on p.order_id =
o.order_id
order by month
```

RESULT:



Row	month	payment_type	payment_method
1	1	UPI	1715
2	1	credit_card	6103
3	1	debit_card	118
4	1	voucher	477
5	2	UPI	3438
6	2	credit_card	12712
7	2	debit_card	200
8	2	voucher	901
9	3	UPI	5380
10	3	credit_card	20419

Insights and suggestions:

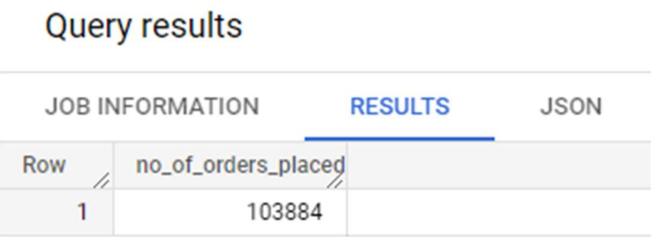
1. We could see that most payment methods are online. Then that's a good point.
2. We can also build a hassle-free online payment system. Adding subscriptions, coupons, and more offers

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

QUERY:

```
SELECT
count(payment_installments) as no_of_orders_placed
FROM `case_study.payments`
where payment_installments >= 1
```

RESULT:



Row	no_of_orders_placed
1	103884

Outcomes:

1. We could see people are comfortable paying in instalments. If we provide EMI options with fewer interest rates. We can increase the number of orders.

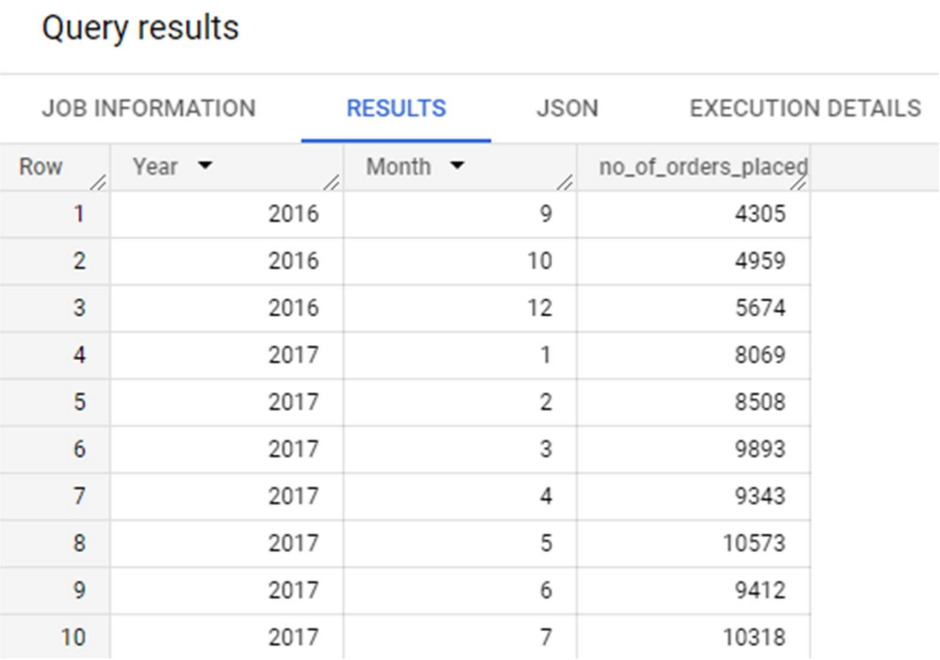
2. In-depth Exploration:

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

QUERY:

```
SELECT
distinct EXTRACT(year from order_purchase_timestamp) as Year,
EXTRACT(month from order_purchase_timestamp) as Month,
count(*) over(partition by EXTRACT(month from order_purchase_timestamp))  as
no_of_orders_placed_every_month
from `case_study.orders`
ORDER BY Year,Month
```

RESULTS:



Row	Year	Month	no_of_orders_placed
1	2016	9	4305
2	2016	10	4959
3	2016	12	5674
4	2017	1	8069
5	2017	2	8508
6	2017	3	9893
7	2017	4	9343
8	2017	5	10573
9	2017	6	9412
10	2017	7	10318

Outcomes:

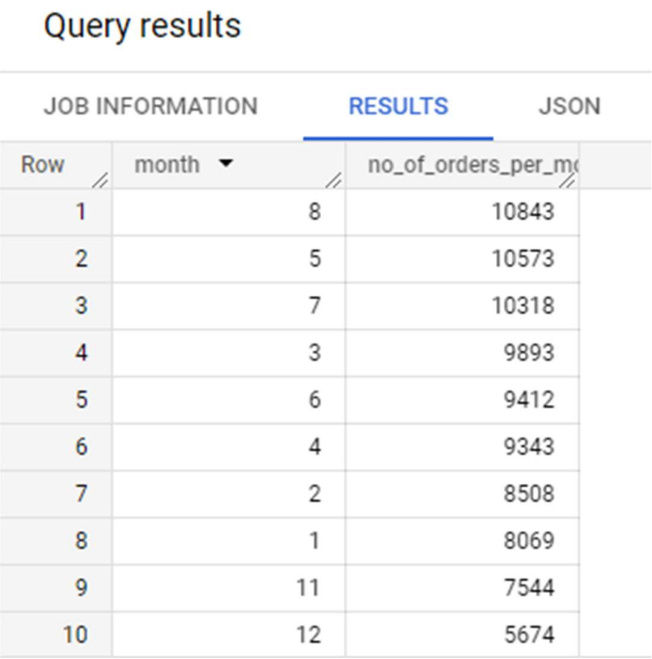
1. Yes, there is a significant increase in the number of orders placed year by year.

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

QUERY:

```
SELECT
EXTRACT(month from order_purchase_timestamp) as month,
count(*) as no_of_orders_per_month
FROM `case_study.orders`
group by month
order by no_of_orders_per_month desc
```

RESULT:



Query results

Row	month	no_of_orders_per_m
1	8	10843
2	5	10573
3	7	10318
4	3	9893
5	6	9412
6	4	9343
7	2	8508
8	1	8069
9	11	7544
10	12	5674

Outcomes:

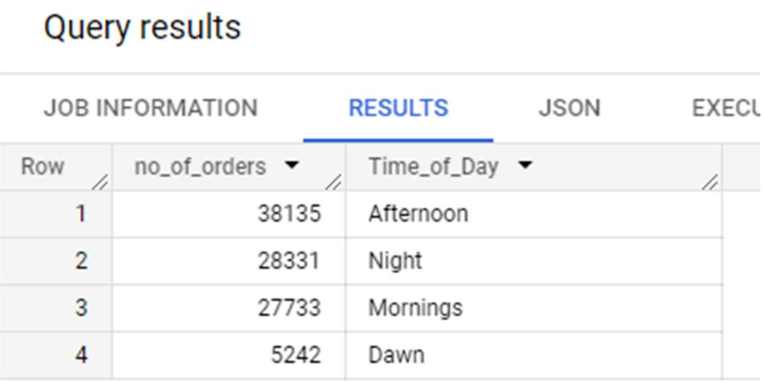
1. We could see in the month of August, May, and July have a higher no of orders.
2. Later on there is a decrease in the number of orders, need to analyse why and know the products they are in need of. Also incorporate seasonal offers.

3. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
   1. 0-6 hrs : Dawn
   2. 7-12 hrs : Mornings
   3. 13-18 hrs : Afternoon
   4. 19-23 hrs : Night

QUERY:

```
select count(EXTRACT(HOUR FROM order_purchase_timestamp)) as no_of_orders,
case
  when EXTRACT(HOUR FROM order_purchase_timestamp) between 0 and 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 `case_study.orders`
group by Time_of_Day
order by no_of_orders desc
```

RESULT:



Row	no_of_orders	Time_of_Day
1	38135	Afternoon
2	28331	Night
3	27733	Mornings
4	5242	Dawn

Insights:

1. Brazilian customers mostly place their orders during the Afternoons.

4. Impact on the Economy:

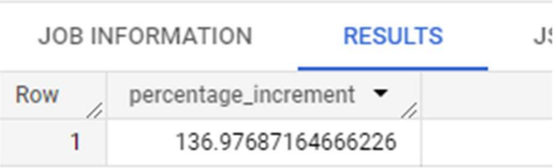
Analyse the money movement by e-commerce by looking at order prices, freight, and others.

4.1. Get the % increase in the cost of orders from the 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.

QUERY:

```
select ((max(total_payment) - min(total_payment))*100/min(total_payment)) as
percentage_increment
from (
select
extract(year from order_purchase_timestamp) as year,
sum(payment_value) as total_payment
from `case_study.payments` p inner join `case_study.orders` o on p.order_id =
o.order_id
where  (extract(year from order_purchase_timestamp) = 2018 and extract(month from
order_purchase_timestamp) between 1 and 8) or (extract(year from
order_purchase_timestamp) = 2017 and extract(month from order_purchase_timestamp)
between 1 and 8)
group by year
) as tbl
```

RESULT:



Row	percentage_increment
1	136.97687164666226

Insights:

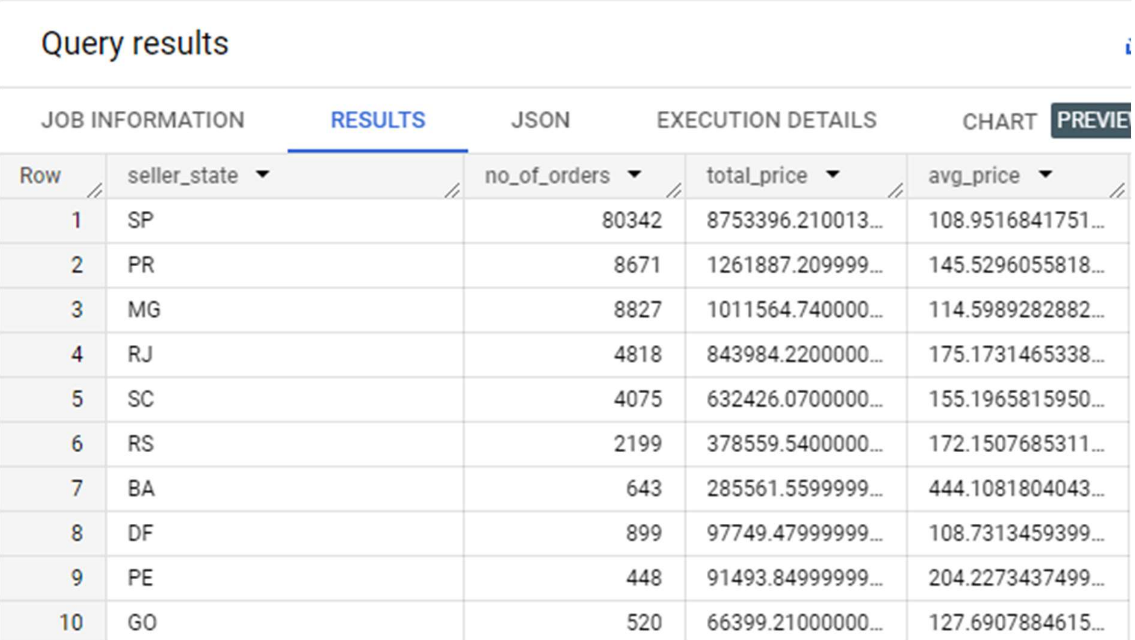
1. There is a 136.97% increment in the year 2018 when compared to the year 2017.

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

QUERY:

```
select
seller_state,
count(order_id) as no_of_orders,
sum(price) as total_price,
avg(price) as avg_price
from `case_study.sellers` s inner join `case_study.order_items` oi on s.seller_id =
oi.seller_id
group by seller_state
order by total_price desc
```

RESULT:



Row	seller_state	no_of_orders	total_price	avg_price
1	SP	80342	8753396.210013...	108.9516841751...
2	PR	8671	1261887.209999...	145.5296055818...
3	MG	8827	1011564.740000...	114.5989282882...
4	RJ	4818	843984.2200000...	175.1731465338...
5	SC	4075	632426.0700000...	155.1965815950...
6	RS	2199	378559.5400000...	172.1507685311...
7	BA	643	285561.5599999...	444.1081804043...
8	DF	899	97749.47999999...	108.7313459399...
9	PE	448	91493.84999999...	204.2273437499...
10	GO	520	66399.21000000...	127.6907884615...

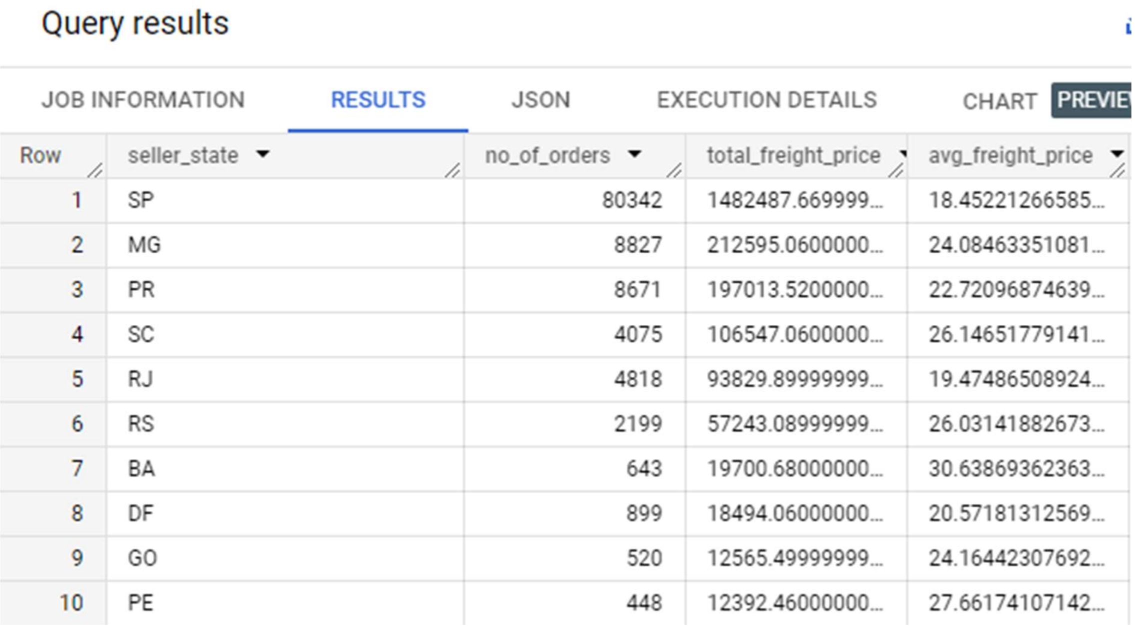
Outcome: The total and average prices of the orders state wise shows state SP has higher no of orders.

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

QUERY:

```
select
seller_state,
count(order_id) as no_of_orders,
sum(freight_value) as total_freight_price,
avg(freight_value) as avg_freight_price
from `case_study.sellers` s inner join `case_study.order_items` oi on s.seller_id =
oi.seller_id
group by seller_state
order by total_freight_price desc
```

RESULT:



Query results

Row	seller_state	no_of_orders	total_freight_price	avg_freight_price
1	SP	80342	1482487.669999...	18.45221266585...
2	MG	8827	212595.0600000...	24.08463351081...
3	PR	8671	197013.5200000...	22.72096874639...
4	SC	4075	106547.0600000...	26.14651779141...
5	RJ	4818	93829.89999999...	19.47486508924...
6	RS	2199	57243.08999999...	26.03141882673...
7	BA	643	19700.68000000...	30.63869362363...
8	DF	899	18494.06000000...	20.57181312569...
9	GO	520	12565.49999999...	24.16442307692...
10	PE	448	12392.46000000...	27.66174107142...

Outcomes:

1. The total and average freight prices of the orders state-wise show state SP has a higher no of orders.

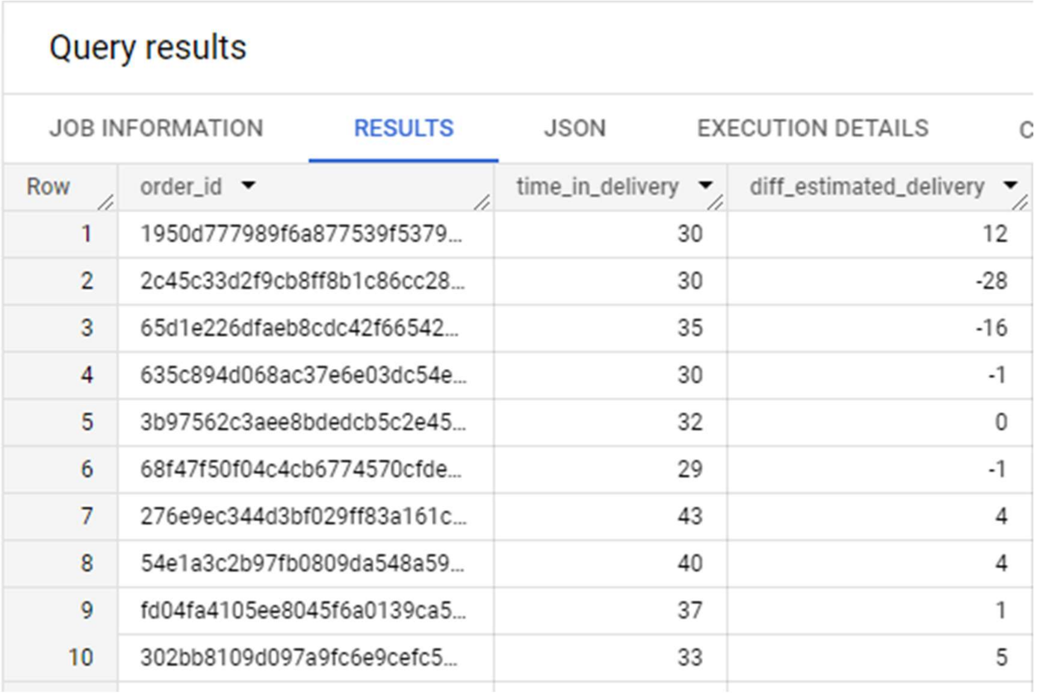
5. **Analysis based on sales, freight, and delivery time.**

   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.

QUERY:

```
select
order_id,
date_diff(order_delivered_customer_date, order_purchase_timestamp,day)
time_in_delivery,
date_diff(order_delivered_customer_date, order_estimated_delivery_date,day)
diff_estimated_delivery,
from `case_study.orders`
where order_delivered_customer_date is not null
```

RESULT:



Query results

Row	order_id	time_in_delivery	diff_estimated_delivery
1	1950d777989f6a877539f5379…	30	12
2	2c45c33d2f9cb8ff8b1c86cc28…	30	-28
3	65d1e226dfaeb8cdc42f66542…	35	-16
4	635c894d068ac37e6e03dc54e…	30	-1
5	3b97562c3aee8bdedcb5c2e45…	32	0
6	68f47f50f04c4cb6774570cfde…	29	-1
7	276e9ec344d3bf029ff83a161c…	43	4
8	54e1a3c2b97fb0809da548a59…	40	4
9	fd04fa4105ee8045f6a0139ca5…	37	1
10	302bb8109d097a9fc6e9cefc5…	33	5

Outcomes:

1. Negative values in the diff_estimated_delivery indicates the orders are delivered in fewer days than the estimated delivery date.

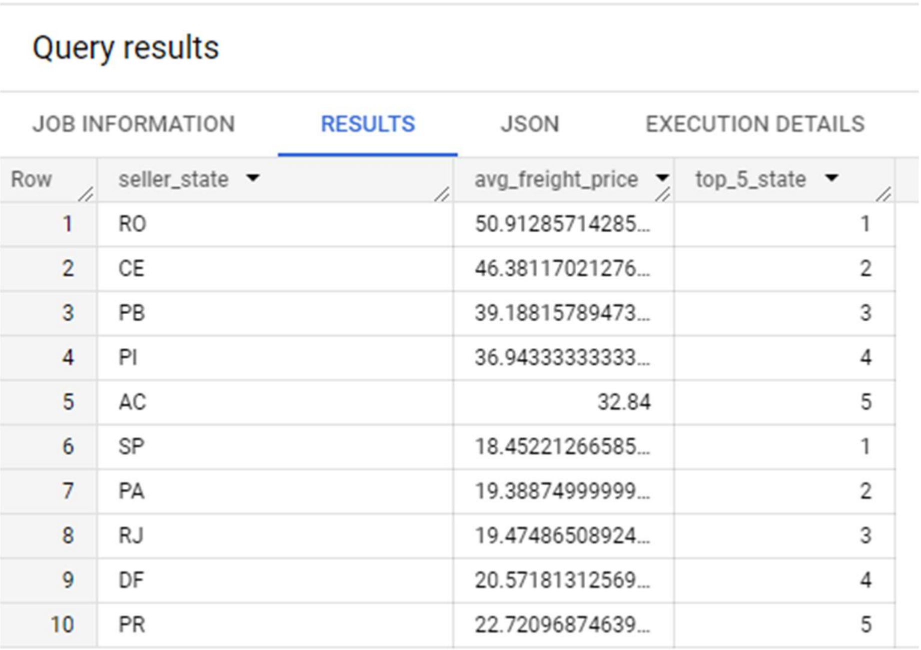
2. Likewise, the difference is days should go on a minimum, and steps to achieve should plan and work.

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

QUERY:

```
(
select
*,
dense_rank() over(order by avg_freight_price) as top_5_state
from (
select
distinct seller_state,
avg(freight_value) over(partition by seller_state) as avg_freight_price,
from `case_study.sellers` s inner join `case_study.order_items` oi on s.seller_id =
oi.seller_id
) tbl
order by avg_freight_price
limit 5
)
union all
(
select
*,
dense_rank() over(order by avg_freight_price desc) as top_5_state
from (
select
distinct seller_state,
avg(freight_value) over(partition by seller_state) as avg_freight_price,
from `case_study.sellers` s inner join `case_study.order_items` oi on s.seller_id =
oi.seller_id
) tbl
order by avg_freight_price desc
limit 5
)
```

RESULT:



Row	seller_state	avg_freight_price	top_5_state
1	RO	50.91285714285...	1
2	CE	46.38117021276...	2
3	PB	39.18815789473...	3
4	PI	36.94333333333...	4
5	AC	32.84	5
6	SP	18.45221266585...	1
7	PA	19.38874999999...	2
8	RJ	19.47486508924...	3
9	DF	20.57181312569...	4
10	PR	22.72096874639...	5

Insights and suggestions:

1. The states RO, CE, PB, PI, and AC are having highest freight values on average. The freight values can be minimized.
   a. by choosing the right transport system.
   b. Avoiding wastages used in product packages.
   c. Analyse data because the freight value is high in number and take necessary actions Etc…,

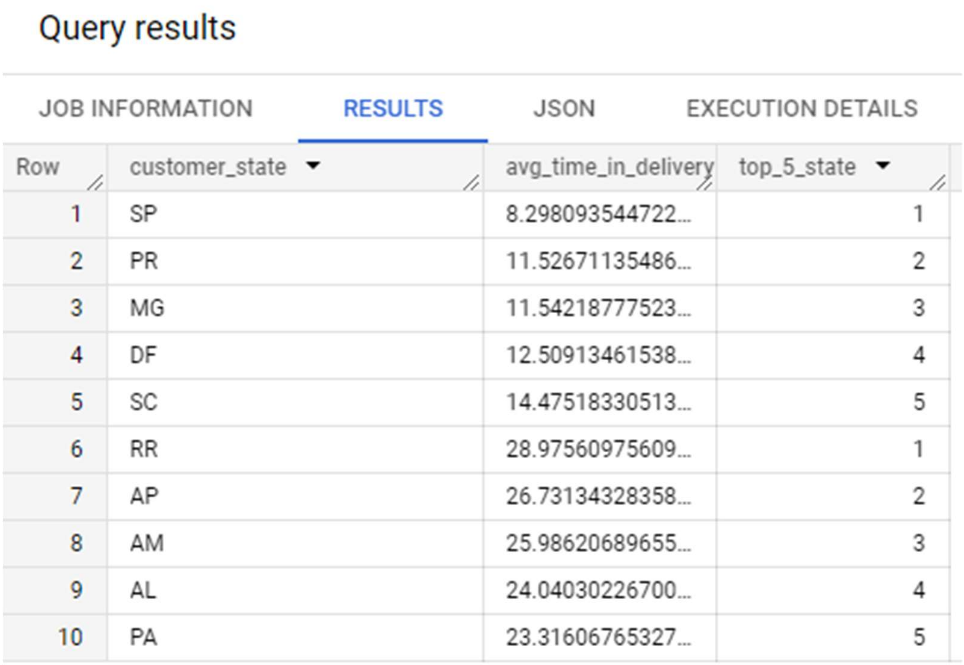
2.The states SP, PA, RJ, DF, and PR are having less freight values. Analyse data on why these states have fewer values. If any fruitful results are found, apply those to the states having higher freight values.

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

QUERY:

```
(
select *,
dense_rank() over(order by avg_time_in_delivery) as top_5_state
from
(
select
customer_state,
avg(date_diff(order_delivered_customer_date, order_purchase_timestamp,day))
avg_time_in_delivery,
from `case_study.orders` o inner join `case_study.customers` c on o.customer_id =
c.customer_id
where order_delivered_customer_date is not null and order_status = 'delivered'
group by customer_state
) tbl
order by avg_time_in_delivery
limit 5
)
union all
(
select *,
dense_rank() over(order by avg_time_in_delivery desc) as top_5_state
from
(
select
customer_state,
avg(date_diff(order_delivered_customer_date, order_purchase_timestamp,day))
avg_time_in_delivery,
from `case_study.orders` o inner join `case_study.customers` c on o.customer_id =
c.customer_id
where order_delivered_customer_date is not null and order_status = 'delivered'
group by customer_state
) tbl
order by avg_time_in_delivery desc
limit 5
)
```

RESULT:



Row	customer_state	avg_time_in_delivery	top_5_state
1	SP	8.298093544722...	1
2	PR	11.52671135486...	2
3	MG	11.54218777523...	3
4	DF	12.50913461538...	4
5	SC	14.47518330513...	5
6	RR	28.97560975609...	1
7	AP	26.73134328358...	2
8	AM	25.98620689655...	3
9	AL	24.04030226700...	4
10	PA	23.31606765327...	5

Insights and suggestions:

1.The states SP, PR, MG, DF, and SC stood as the top 5 states. The delivery partners in these states are working well in delivering the orders they have less average time they took for delivery. Still, if we need to have less delivery time. We must look over the possibilities and do the needful.

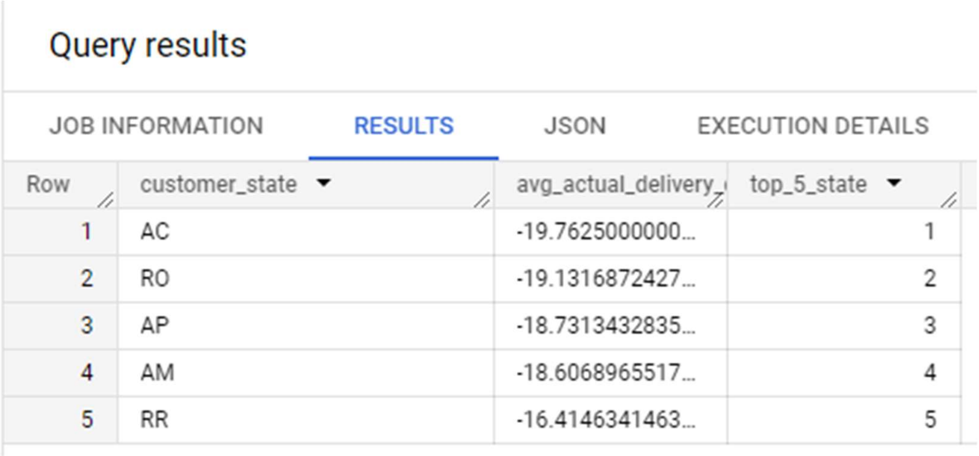
2.The other states like RR, AP, AM, AL, and PA stood as the lowest state where average delivery time is in high numbers. We can troubleshoot this issue by assigning more delivery agents. So that can delivery time is minimized. And establishing warehouses within short distances can also be a remedy to an issue. so that ordered products are differentiated and can be moved sooner.

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 dates to figure out how fast the delivery was for each state.

QUERY:

```
select *,
dense_rank() over(order by avg_actual_delivery_date) as top_5_state
from(
select
customer_state,
avg(date_diff(order_delivered_customer_date, order_estimated_delivery_date,day))
avg_actual_delivery_date,
from `case_study.orders` o inner join `case_study.customers` c on o.customer_id =
c.customer_id
group by customer_state
)
order by avg_actual_delivery_date
limit 5
```

RESULT:



Row	customer_state	avg_actual_delivery_	top_5_state
1	AC	-19.7625000000...	1
2	RO	-19.1316872427...	2
3	AP	-18.7313432835...	3
4	AM	-18.6068965517...	4
5	RR	-16.4146341463...	5

OUTCOMES:

1.In the above top 5 states, the customers are receiving their ordered products than the expected delivery rate. This shows the delivery partners or agencies are performing at their best.
2.The negative values in the above result, indicate the average no of days per state the order is delivered than the expected delivery rate