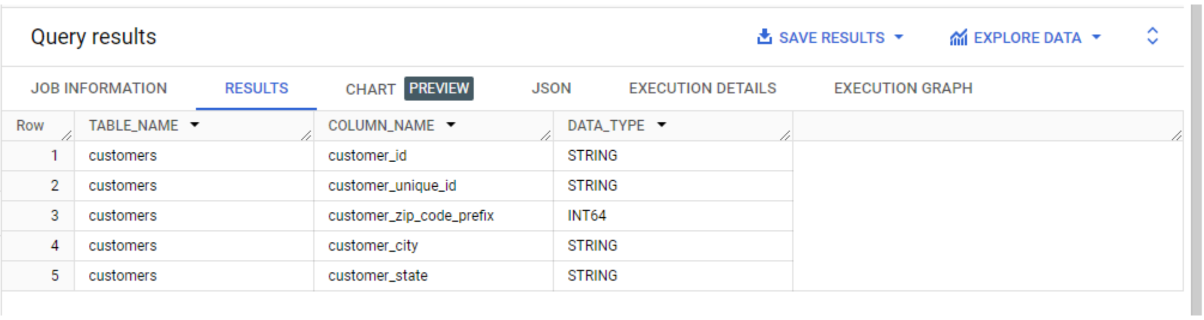
1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

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

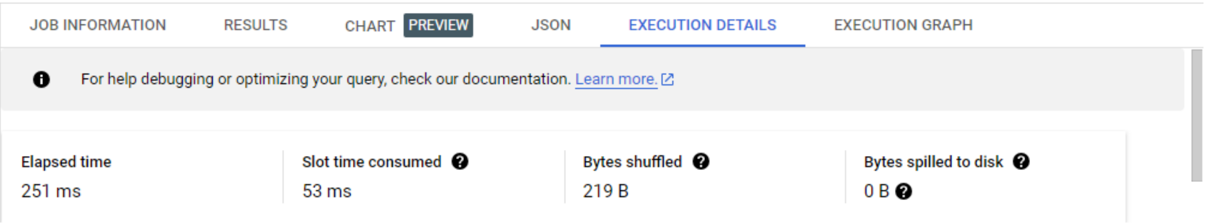
Query:

```
SELECT
  TABLE_NAME,
  COLUMN_NAME,
  DATA_TYPE
FROM able-scope-402205.casestudy_target.INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME='customers';
```

```
  Query Execution details : Elapsed time : 251ms
                            Slot time consumed: 53ms
```



Row	TABLE_NAME	COLUMN_NAME	DATA_TYPE
1	customers	customer_id	STRING
2	customers	customer_unique_id	STRING
3	customers	customer_zip_code_prefix	INT64
4	customers	customer_city	STRING
5	customers	customer_state	STRING



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time 251 ms

Slot time consumed 53 ms

Bytes shuffled 219 B

Bytes spilled to disk 0 B

Insights:-

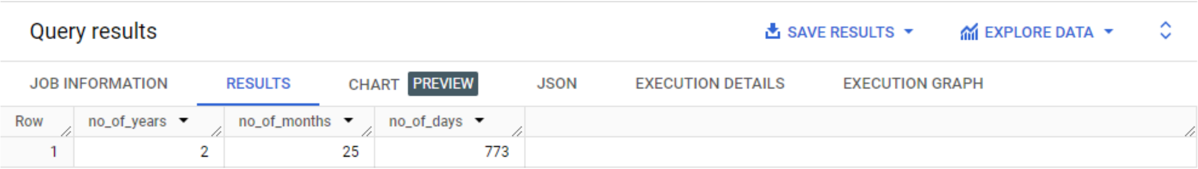
1. we can see the columns in customers table are of type string(VARCHAR) except customer_zip_code_prefix which of type integer. This implies most of the columns are having string as a datatype.

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

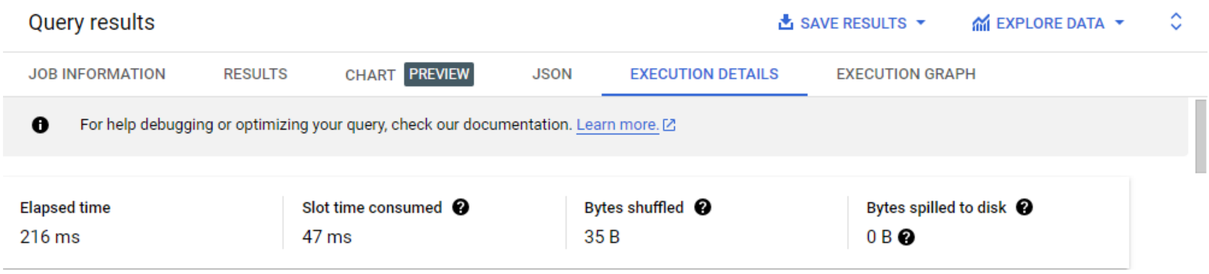
Query:

```
with time_range as (
SELECT EXTRACT(DATE from order_purchase_timestamp) as date_order
FROM casestudy_target.orders
)

select
DATE_DIFF(max(date_order), min(date_order), year) as no_of_years,
DATE_DIFF(max(date_order), min(date_order), month) as no_of_months,
DATE_DIFF(max(date_order), min(date_order), day) as no_of_days
from time_range;
```

```
Query Execution details : Elapsed time : 216ms
                          Slot time consumed: 47ms
```



Row	no_of_years	no_of_months	no_of_days
1	2	25	773



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
216 ms	47 ms	35 B	0 B

Insights:-

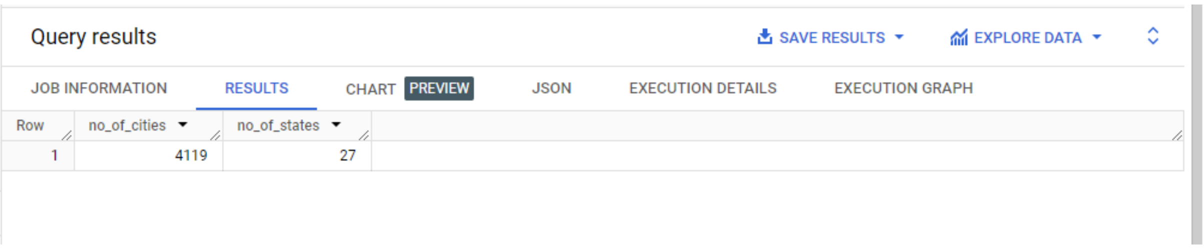
1. Thus the **total time range** in this case study is **2years, 25months, 773days.**
2. We can use this data to analyse the trends and monthly or yearly seasonality

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

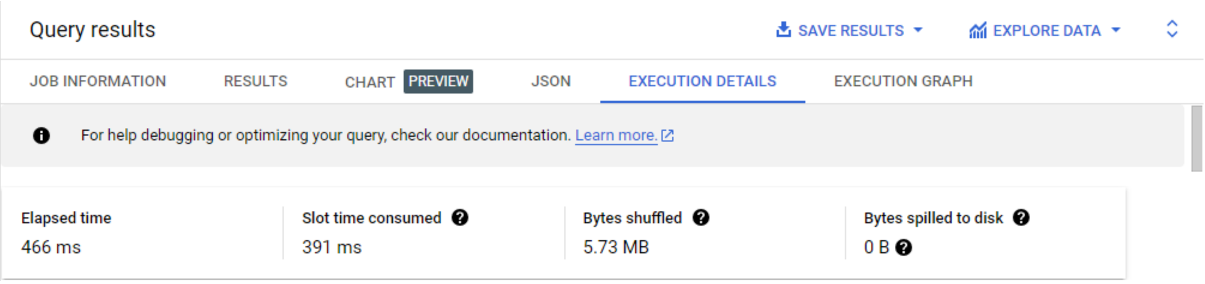
Query:

```
select
  count(distinct c.customer_city) as no_of_cities,
  count(distinct c.customer_state) as no_of_states
from casestudy_target.orders o inner join
casestudy_target.customers c
on c.customer_id=o.customer_id
```

```
Query Execution details : Elapsed time : 466ms
                          Slot time consumed: 391ms
```



Row	no_of_cities	no_of_states
1	4119	27



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time
466 ms

Slot time consumed
391 ms

Bytes shuffled
5.73 MB

Bytes spilled to disk
0 B

Insights:

1. The case study provided has 27 states and 4119 cities. From this data we can analyze how the customer's are distributed in different states and cities .
2. Understanding the different geographical distributions can help to concentrate our customers in these states and cities, figuring our hotspots.
3. And it can help to know how far the company has spread across the world.

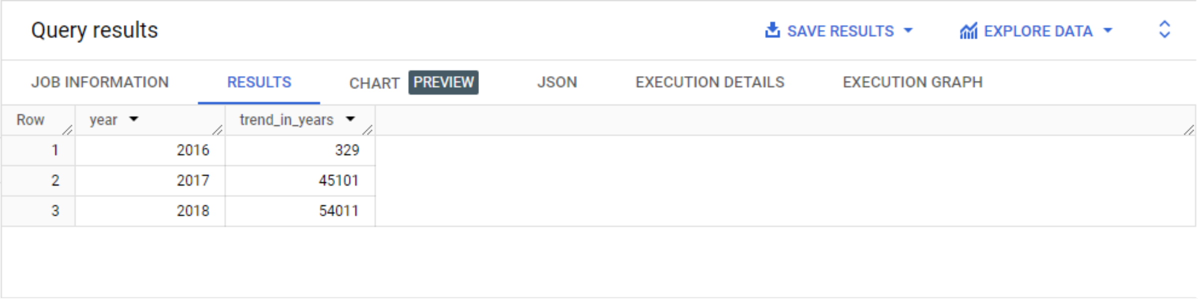
2. In-depth Exploration:

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

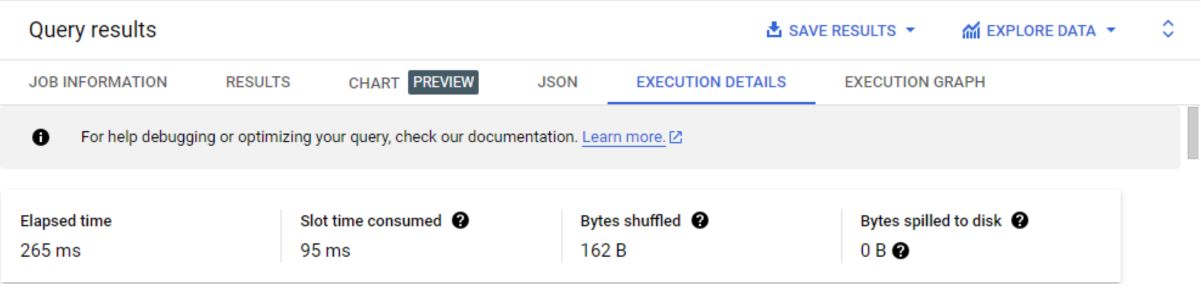
Query:

```
select
  EXTRACT(year from order_purchase_timestamp) as year,
  count(*) as trend_in_years
from casestudy_target.orders
group by year
order by year

Query Execution details : Elapsed time : 265 ms
```

```
                                Slot time consumed: 95 ms
```



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



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time
265 ms

Slot time consumed
95 ms

Bytes shuffled
162 B

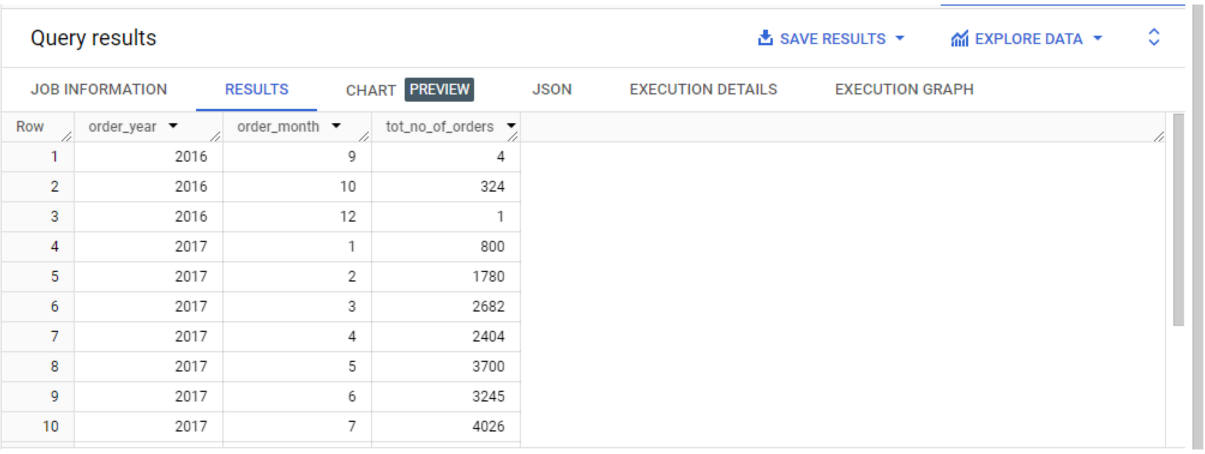
Bytes spilled to disk
0 B

Insights:-

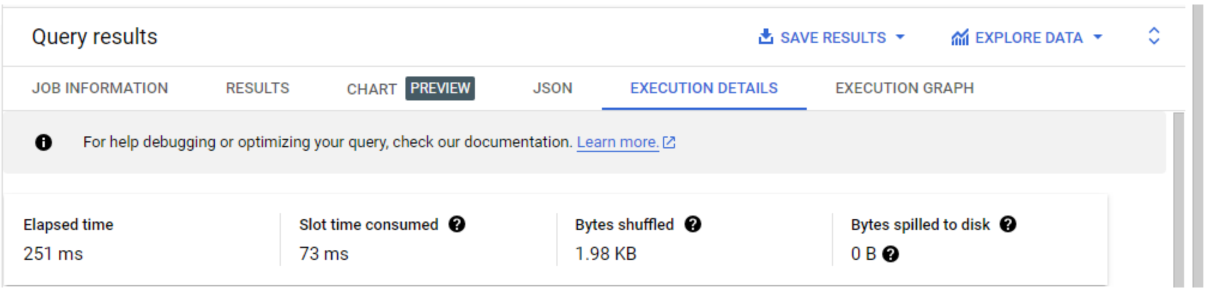
1. There has been a significant trend in the number of ordered when compared to 2016 and 2017 . And from 2017 to 2018 there has been a favorable trend can be seen .

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

```
with info as(
select *,
EXTRACT(year from order_purchase_timestamp) as order_year,
EXTRACT(month from order_purchase_timestamp) as order_month
 from casestudy_target.orders
)

select order_year,order_month,count(order_id) as tot_no_of_orders
from info
group by order_year,order_month
order by order_year,order_month

Query Execution details : Elapsed time : 251 ms
                          Slot time consumed: 73 ms
```



Row	order_year	order_month	tot_no_of_orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
251 ms	73 ms	1.98 KB	0 B

Insights:-

1. We see a seasonal trend for November and December there was increase in orders during this period this might be because of black Friday and new year celebrations.
2. And we can also see from March 2017 to October 2017 and January 2018 to August 2018 can be categorised as a steady period because the no.of orders seems to be relatively stable.
3. And from August 2018 it's noticeable that the data shows extremely low order numbers .This might be because of significant change in business operations.

Recommendations:-

1. Prepare targeted promotions and offers for peaks seasons like year end holidays, black Friday, and major festivals.
2. Gather more accurate data over time for better decision making.

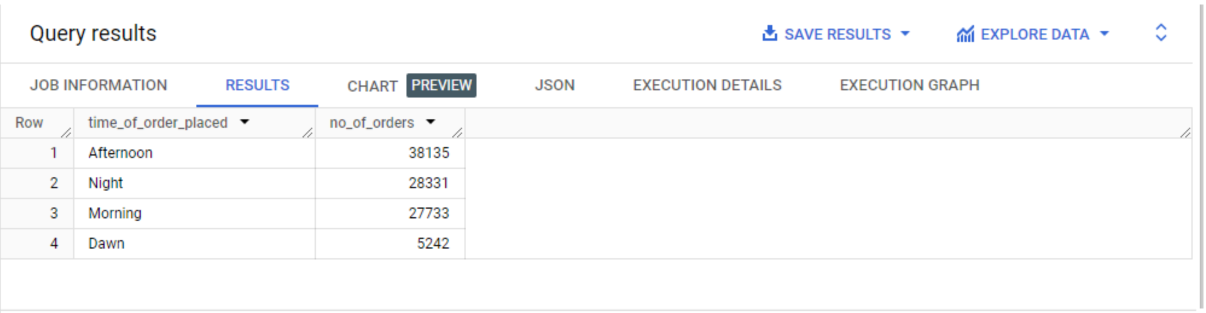
c. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

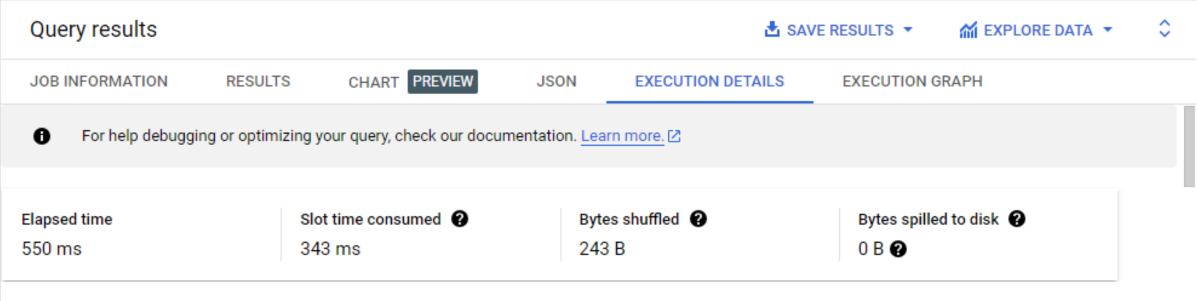
Query:

```
select

  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 'Morning'
    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_order_placed,
  count(*) as no_of_orders
from casestudy_target.orders
group by time_of_order_placed
order by no_of_orders desc;
```

```
Query Execution details : Elapsed time : 550ms
                          Slot time consumed: 343ms
```



Row	time_of_order_placed	no_of_orders
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242



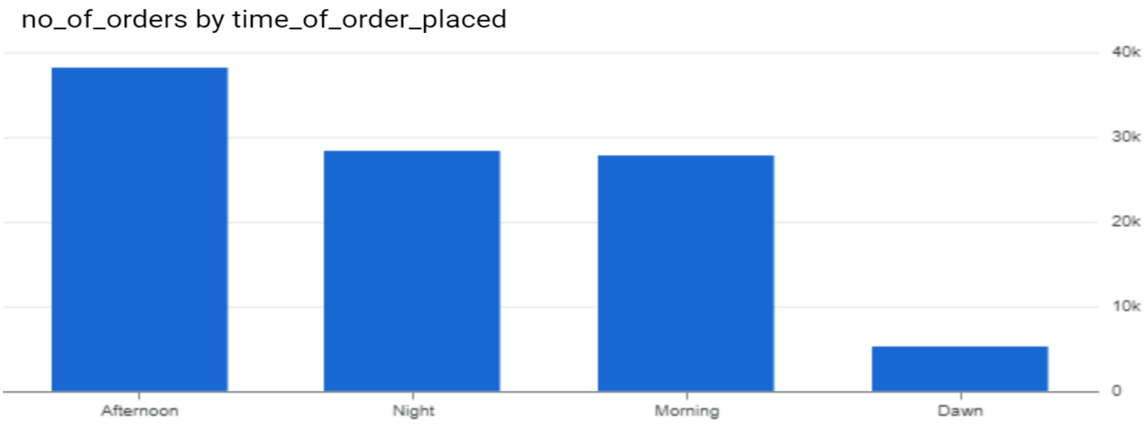
For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time
550 ms

Slot time consumed
343 ms

Bytes shuffled
243 B

Bytes spilled to disk
0 B



Insights:-

1. Analysing the data based on the time of the day when orders are placed provides valuable insights into customer behaviour and preferences through out the day.
2. The highest number of orders placed is afternoon ,indicating a peak in customer activity this time. Night and morning orders are slightly lower than the afternoon, signifying consistent customer engagement during these periods. And orders placed I dawn are significantly low.

Recommendations:-

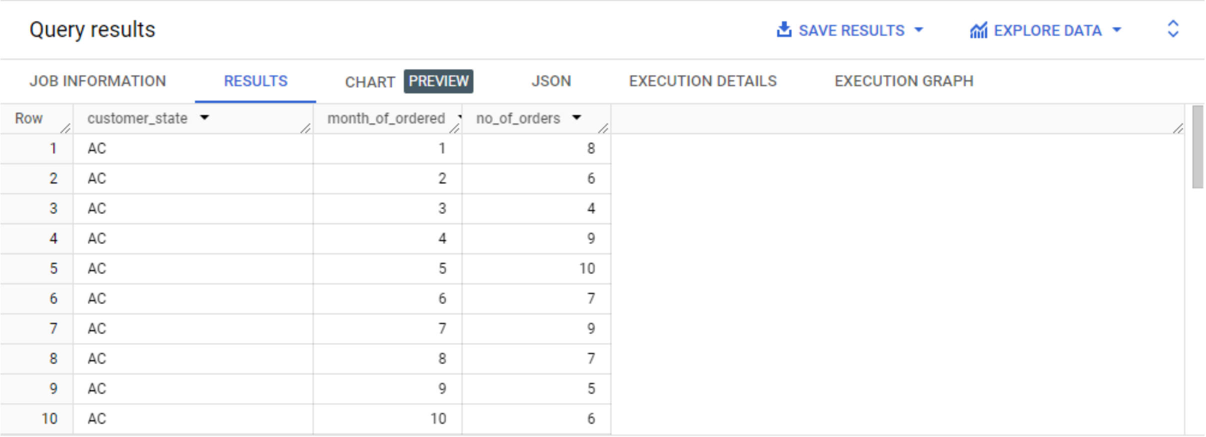
1. usage of the peak in afternoon orders by marketing campaigns and promotions during this time. Consider flash sales, special offers.
2. Although number of orders placed in night and morning are comparatively low but we can still improve this by creating some social media posts advertisements .

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

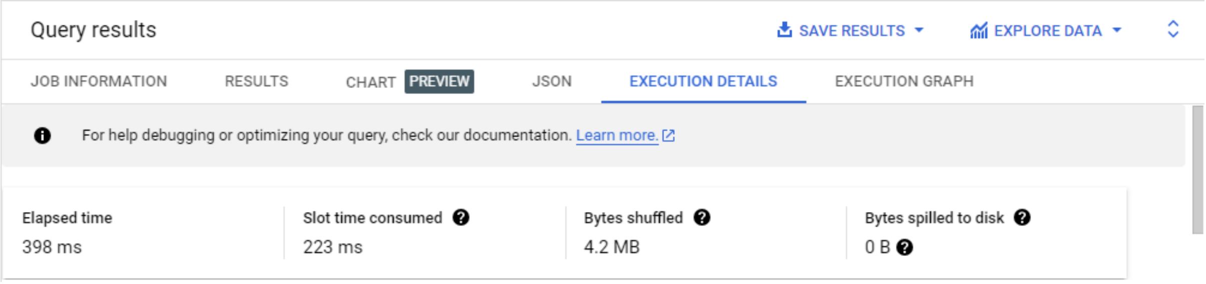
A. Get the month on month no. of orders placed in each state.

Query:

```
select

  c.customer_state,
  EXTRACT(month from o.order_purchase_timestamp) as
month_of_ordered,
  count(*) as no_of_orders
from casestudy_target.orders o join
      casestudy_target.customers c
  on o.customer_id = c.customer_id
group by c.customer_state,month_of_ordered
order by c.customer_state,month_of_ordered

Query Execution details : Elapsed time : 550 ms
                          Slot time consumed: 343 ms
```



Row	customer_state	month_of_ordered	no_of_orders
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



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
398 ms	223 ms	4.2 MB	0 B

Insights:-

1. From the results we can see monthly ordered count for each state in brazil. From this data we can spot trends, patterns and seasonality in terms of number of orders for each state in brazil. In this data we can find that for every month the state named SP has highest number of orders.
2. The data shows some states are relatively consistent in terms of order volume.while some states are higher and lower.

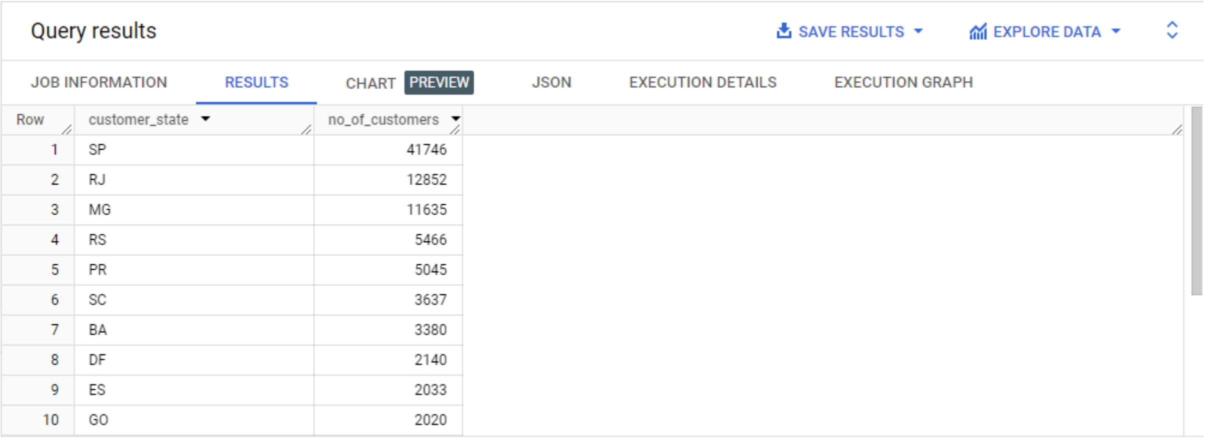
Recommendations:-

1. marketing specific to each state's purchasing behaviour. implementing promotions or offers based on the observed trends in each state to maximize customers.
2. Gather more extensive data and insights for each state to comprehend local trends, This can help in refining strategies for each region.

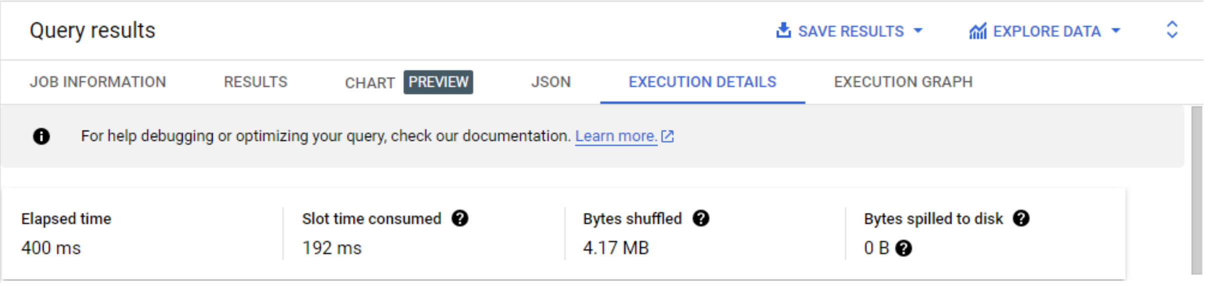
B. How are the customers distributed across all the states?

Query:

```
select
  customer_state,
  count(distinct customer_id) as no_of_customers
from casestudy_target.customers
group by customer_state
order by no_of_customers desc;
```



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



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
400 ms	192 ms	4.17 MB	0 B

Insights:-

1. By analysing the query results we can come know that which states have the most customers and which states have comparatively fewer customers.
2. Here the state named SP have highest number of customers and state named RR has fewest customers.

Recommendations:-

1. We can improve the fewer customers states like RR by promotions and discount over these areas.

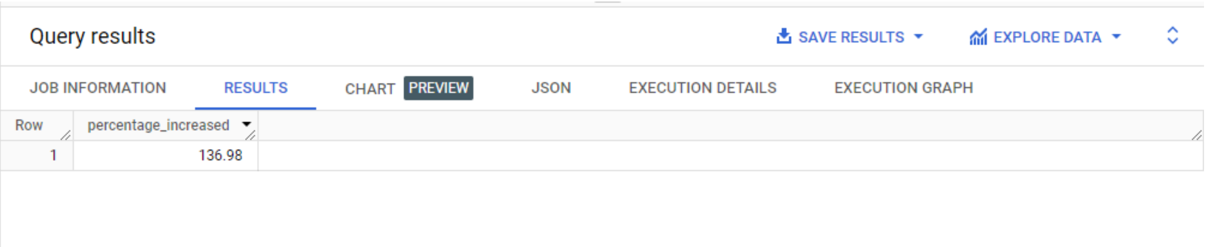
4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

A. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
You can use the "payment_value" column in the payments table to get the cost of orders.

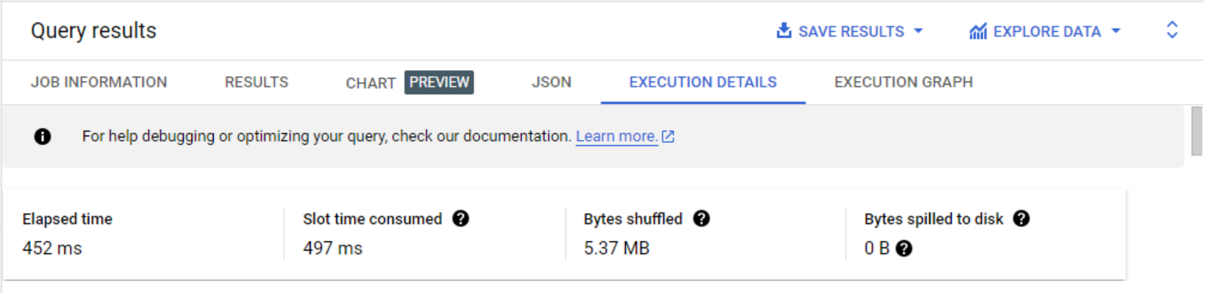
Query:

```
with orders_in_2017 as (
    select
        SUM(p.payment_value) as total_pay_2017
    from
        casestudy_target.orders o
    inner join
        casestudy_target.payments p
        on o.order_id = p.order_id
    where
        EXTRACT(year from o.order_purchase_timestamp) = 2017
        and EXTRACT(month from o.order_purchase_timestamp) BETWEEN 1
and 8
),
orders_in_2018 as (
    select
        SUM(p.payment_value) as total_pay_2018
    from
        casestudy_target.orders o
    inner join
        casestudy_target.payments p
        on o.order_id = p.order_id
    where
        EXTRACT(year from o.order_purchase_timestamp) = 2018
        and EXTRACT(month from o.order_purchase_timestamp) BETWEEN 1
and 8
)
select
    ROUND(((total_pay_2018 - total_pay_2017) / total_pay_2017) *
100, 2) AS percentage_increased
from
    orders_in_2017, orders_in_2018;
```

```
Query Execution details : Elapsed time : 452ms
                      Slot time consumed: 497ms
```



Row	percentage_increased
1	136.98



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time
452 ms

Slot time consumed
497 ms

Bytes shuffled
5.37 MB

Bytes spilled to disk
0 B

Insights:-

1. From the query results we can observe that growth rate increased approximately 137% from 2017 to 2018. For this query used the months between January to August in the years 2017 and 2018.
2. Limiting the months in the year allows us for more targeted information.

Recommendations:-

1. By calculating the percentage increase, you can identify if there's a positive or negative trend in spending during these specific months across the two years.
2. This allows us to observe some seasonal trends.

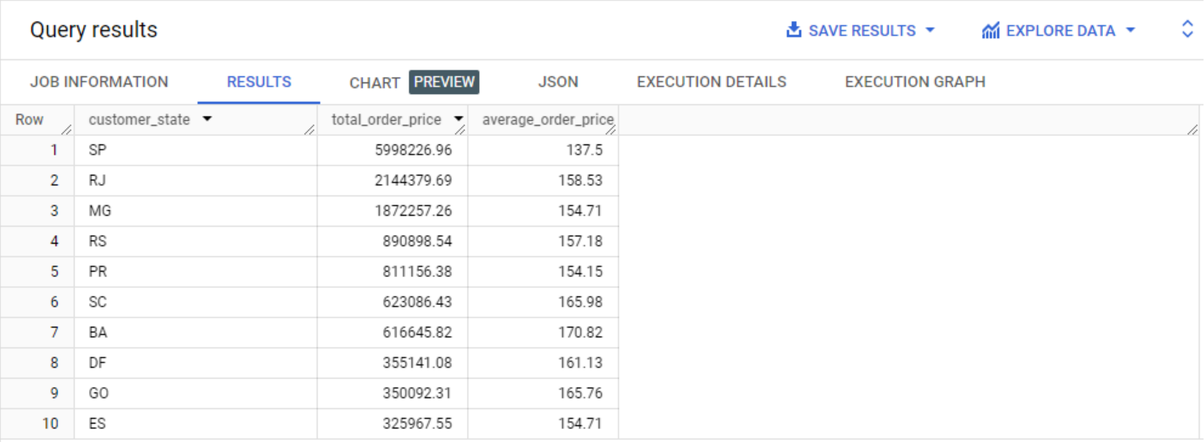
B. Calculate the Total & Average value of order price for each state.

Query:

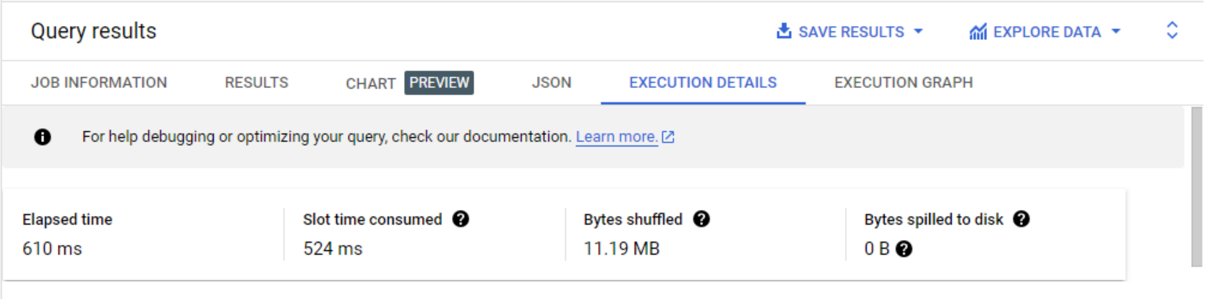
```
select

  customer_state,
  ROUND(sum(p.payment_value),2) as total_order_price,
  ROUND(avg(p.payment_value),2) as average_order_price
from casestudy_target.payments p inner join
  casestudy_target.orders o
  on p.order_id=o.order_id
inner join casestudy_target.customers c
  on o.customer_id=c.customer_id
group by customer_state
order by total_order_price desc, average_order_price desc;
```

Query Execution details : Elapsed time : 610 ms

Slot time consumed: 524 ms



Row	customer_state	total_order_price	average_order_price
1	SP	5998226.96	137.5
2	RJ	2144379.69	158.53
3	MG	1872257.26	154.71
4	RS	890898.54	157.18
5	PR	811156.38	154.15
6	SC	623086.43	165.98
7	BA	616645.82	170.82
8	DF	355141.08	161.13
9	GO	350092.31	165.76
10	ES	325967.55	154.71



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
610 ms	524 ms	11.19 MB	0 B

Insights:-

1. The query results showcases total order prices and average order prices for different states, reflecting varying levels of consumer spending in each region.
2. From the results we can see that the state named SP is have highest total order price and average total order price.
3. This helps to identify high value markets.

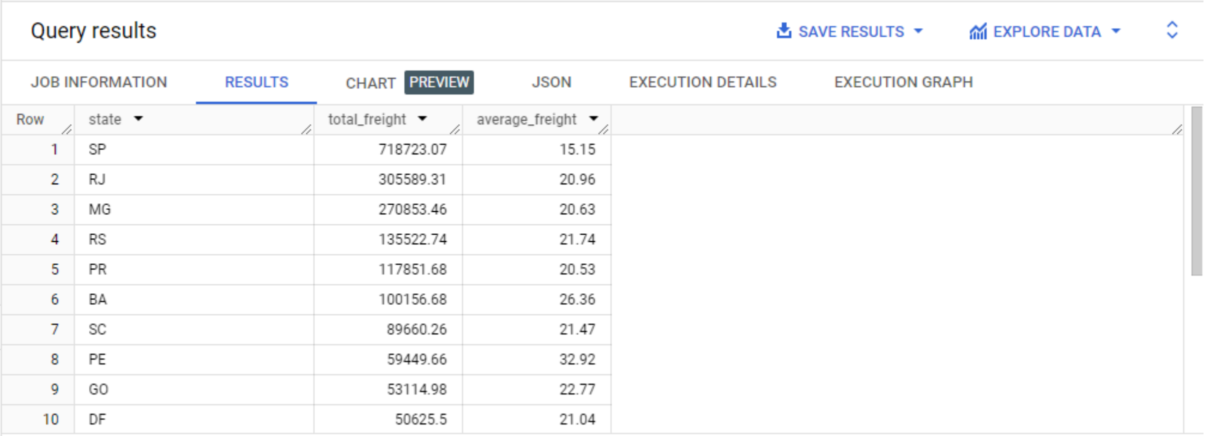
Recommendations:-

1. Examine the states with higher and lower average order prices to understand spending habits, preferences, or potential factors influencing purchase decisions in each region.

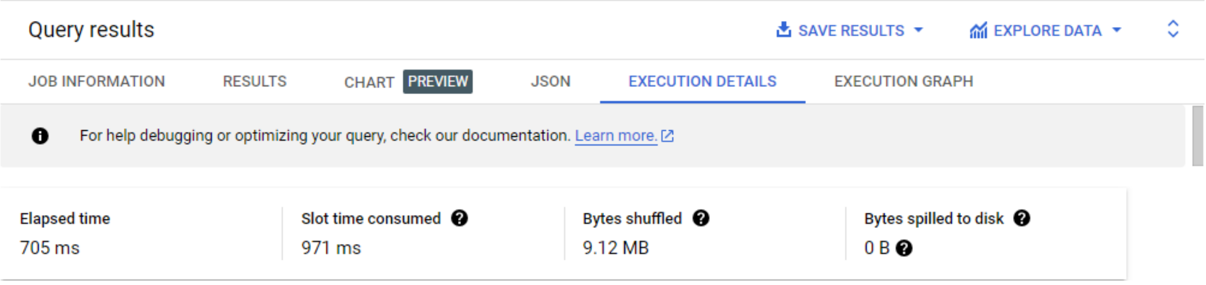
C. Calculate the Total & Average value of order freight for each state.

```
select
  c.customer_state as state,
  ROUND(sum(oi.freight_value),2) as total_freight,
  ROUND(avg(oi.freight_value),2) as average_freight
from casestudy_target.orders o inner join
  casestudy_target.order_items oi
  on o.order_id=oi.order_id inner join
  casestudy_target.customers c
  on c.customer_id = o.customer_id
group by state
order by total_freight desc,average_freight desc;
```

```
Query Execution details : Elapsed time : 705 ms
                                Slot time consumed: 971 ms
```



Row	state	total_freight	average_freight
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
705 ms	971 ms	9.12 MB	0 B

Insights:-

1. The query results shows total and average order freight costs for different states, indicating variations in shipping expenses across regions.
2. The state which is having highest total freight value is state named SP.

Recommendations:-

1. The states which are having high freight value for them we have to give free shipping charges in order to increase the number of orders.

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

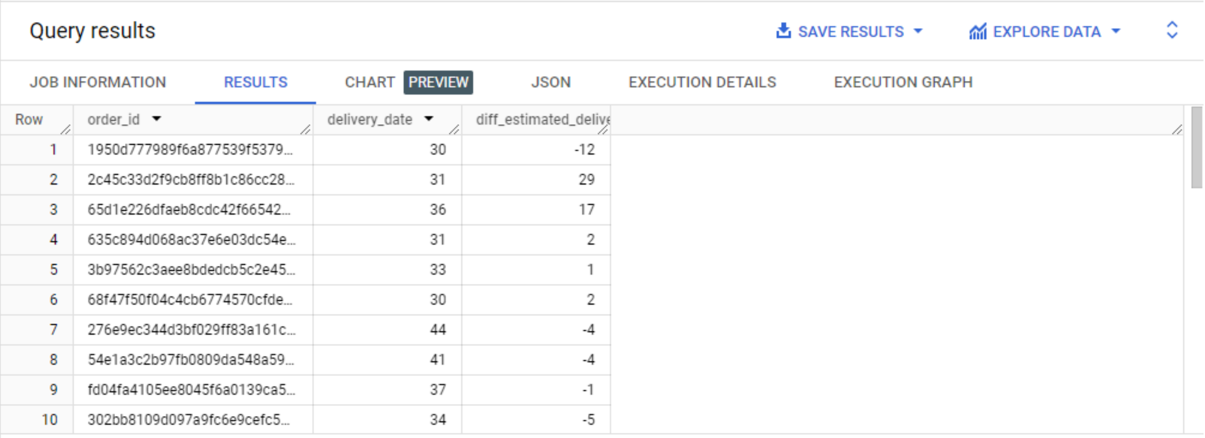
A. Find the no. of days taken to deliver each order from the order’s purchase date as delivery time.
   Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
   Do this in a single query.

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

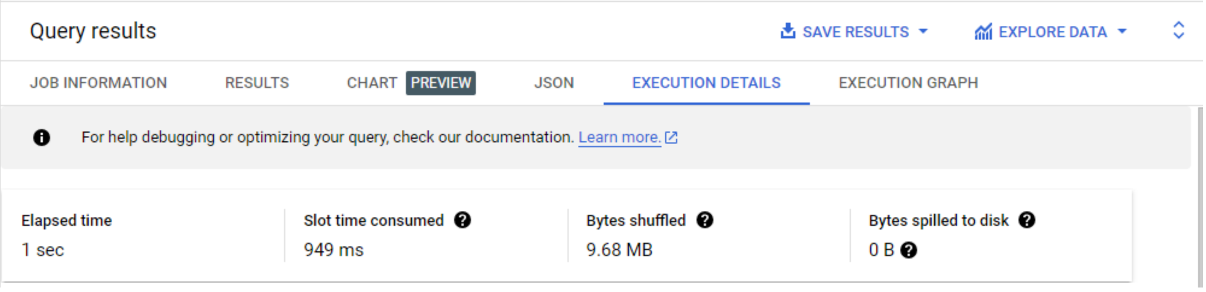
Query:

```
select
  order_id,
  DATE_DIFF(DATE(order_delivered_customer_date),DATE(order_purchase_
timestamp),day) as delivery_date,
  DATE_DIFF(DATE(order_estimated_delivery_date),DATE(order_delivered
_customer_date),day) as diff_estimated_delivery
from casestudy_target.orders
```

```
Query Execution details : Elapsed time : 1 s
                          Slot time consumed: 949 ms
```



Row	order_id	delivery_date	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	31	29
3	65d1e226dfaeb8cdc42f66542...	36	17
4	635c894d068ac37e6e03dc54e...	31	2
5	3b97562c3aee8bdedcb5c2e45...	33	1
6	68f47f50f04c4cb6774570cfde...	30	2
7	276e9ec344d3bf029ff83a161c...	44	-4
8	54e1a3c2b97fb0809da548a59...	41	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	34	-5



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
1 sec	949 ms	9.68 MB	0 B

Insights:-

1. The dataset shows the difference between the estimated delivery date and the actual delivery date for various orders.
2. Different orders take different times may be because of varying order sizes.

Recommendations:-

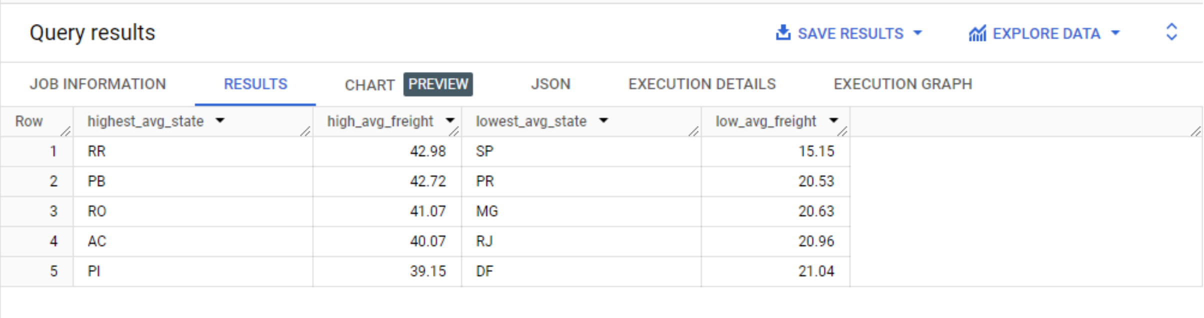
1. List out the reasons behind the late deliveries by contacting delivery partners .

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

Query:

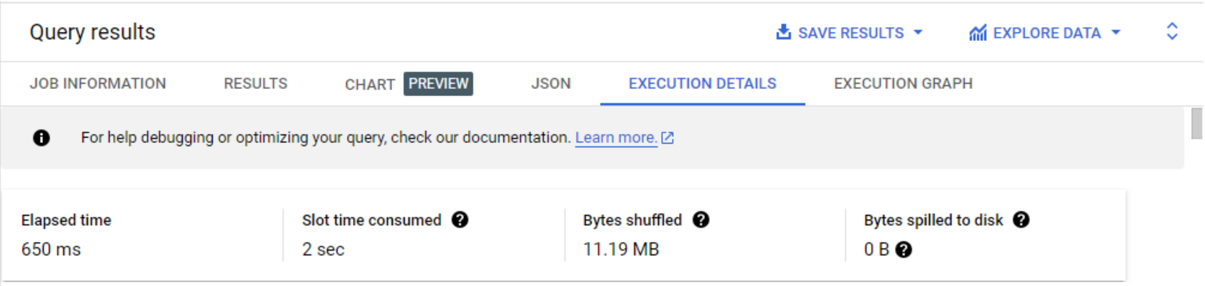
```
with high as(
  select
    c.customer_state,
    ROUND(avg(oi.freight_value),2) as average_freight_value,
    row_number() over(order by (ROUND(avg(oi.freight_value),2))
desc) as rw
  from casestudy_target.orders o inner join
  casestudy_target.order_items  oi
  on o.order_id=oi.order_id inner join
  casestudy_target.customers c
  on o.customer_id=c.customer_id
  group by c.customer_state
  order by average_freight_value desc
  limit 5
), low as(
  select
    c.customer_state,
    ROUND(avg(oi.freight_value),2) as average_freight_value,
    row_number() over(order by (ROUND(avg(oi.freight_value),2))) as
rw2
  from casestudy_target.orders o inner join
  casestudy_target.order_items  oi
  on o.order_id=oi.order_id inner join
  casestudy_target.customers c
  on o.customer_id=c.customer_id
  group by c.customer_state
  order by average_freight_value
  limit 5
)

select
  high.customer_state as highest_avg_state,
  high.average_freight_value as high_avg_freight,
  low.customer_state as lowest_avg_state,
  low.average_freight_value as low_avg_freight
from high inner join low on high.rw = low.rw2
```

```
Query Execution details : Elapsed time : 650 ms
                          Slot time consumed: 2 s
```



Query results

Row	highest_avg_state	high_avg_freight	lowest_avg_state	low_avg_freight
1	RR	42.98	SP	15.15
2	PB	42.72	PR	20.53
3	RO	41.07	MG	20.63
4	AC	40.07	RJ	20.96
5	PI	39.15	DF	21.04



Query results

For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
650 ms	2 sec	11.19 MB	0 B

Insights:-

1. The query results shows the states with highest average freight value and lowest average freight value.
2. The state named RR have highest average freight value and state named SP have lowest average freight value.

Recommendations:-

1. Negotiate the shipping charges in the states having highest freight values. This might lead to reduce the shipping expenses.

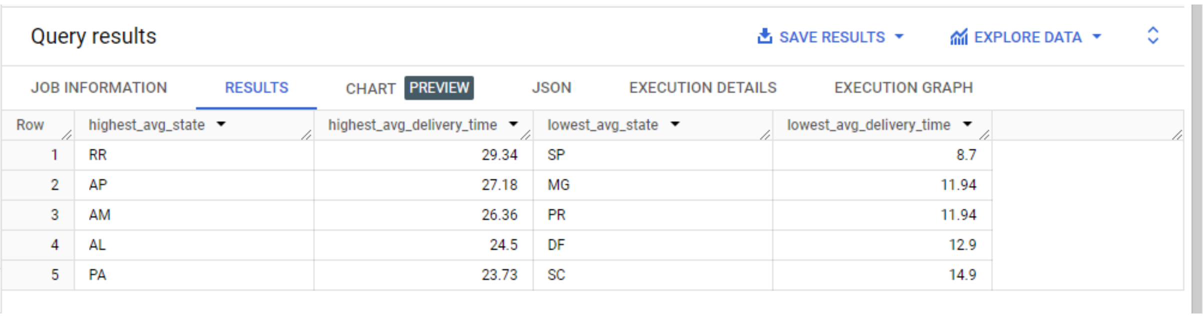
C. Find out the top 5 states with the highest & lowest average delivery time

Query:

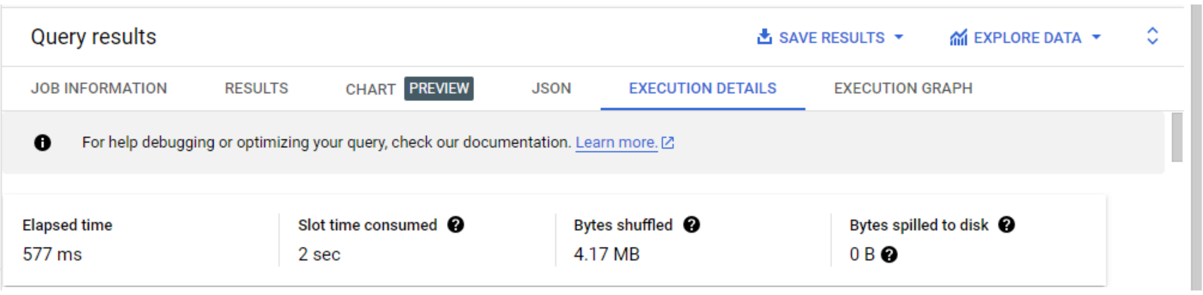
```
with high_avg as (
select
  c.customer_state,
  ROUND(avg(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o.o
rder_purchase_timestamp),day)),2) as delivery_time,
  row_number() over(order by
ROUND(avg(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o.ord
er_purchase_timestamp),
  day)),2) desc) as rw
from casestudy_target.orders o join casestudy_target.customers c
 on c.customer_id=o.customer_id
 where o.order_status='delivered' and
o.order_delivered_customer_date is not null
 group by c.customer_state
 order by delivery_time desc
 limit 5
),low_avg as(
  select
    c.customer_state,
    ROUND(avg(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o
.order_purchase_timestamp),day)),2) as delivery_time,
    row_number() over(order by
ROUND(avg(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o.ord
er_purchase_timestamp),     day)),2)) as rw2
 from casestudy_target.orders o join casestudy_target.customers c
 on c.customer_id=o.customer_id
 where o.order_status='delivered' and
o.order_delivered_customer_date is not null
```

```
 group by c.customer_state
 order by delivery_time
 limit 5
)

select
  h.customer_state as highest_avg_state,
  h.delivery_time as highest_avg_delivery_time,
  l.customer_state as lowest_avg_state,
  l.delivery_time as lowest_avg_delivery_time
 from high_avg h join low_avg l on h.rw=l.rw2
```

```
Query Execution details : Elapsed time : 577 ms
                          Slot time consumed: 2 s
```



Row	highest_avg_state	highest_avg_delivery_time	lowest_avg_state	lowest_avg_delivery_time
1	RR	29.34	SP	8.7
2	AP	27.18	MG	11.94
3	AM	26.36	PR	11.94
4	AL	24.5	DF	12.9
5	PA	23.73	SC	14.9



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
577 ms	2 sec	4.17 MB	0 B

Insights:-

1. The dataset shows the states having highest average delivery time and staes having lowest average delivery time.
2. The state named RR have highest average delivery time and state named SP have lowest average delivery time.

Recommendations:-

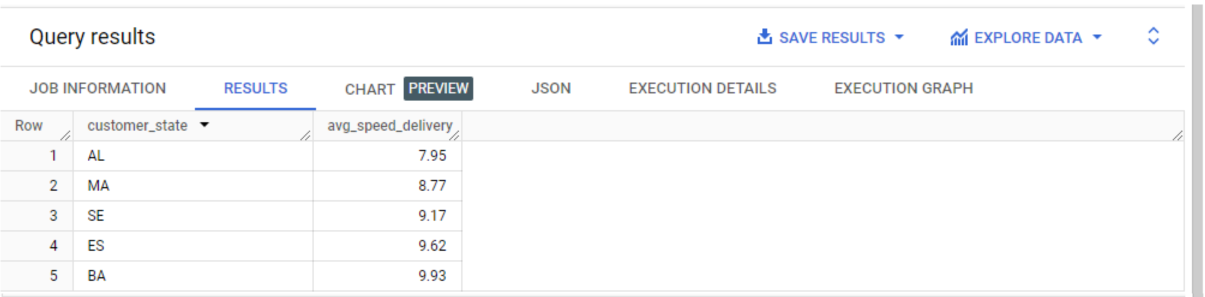
1. List out the reasons behind the late deliveries by contacting delivery partners.

D. 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.

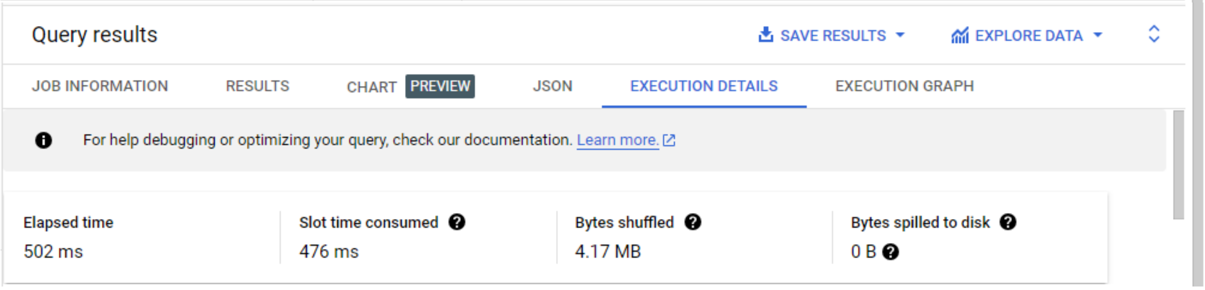
Query:

```
select
  c.customer_state,
  round(avg(DATE_DIFF(o.order_estimated_delivery_date,o.order_delive
red_customer_date,day)),2) as avg_speed_delivery
from casestudy_target.orders o join
     casestudy_target.customers c
     on c.customer_id=o.customer_id
where order_status='delivered' and order_delivered_customer_date is
not null
group by c.customer_state
order by avg_speed_delivery
limit 5;
```

```
Query Execution details : Elapsed time : 502 ms
                          Slot time consumed: 476 ms
```

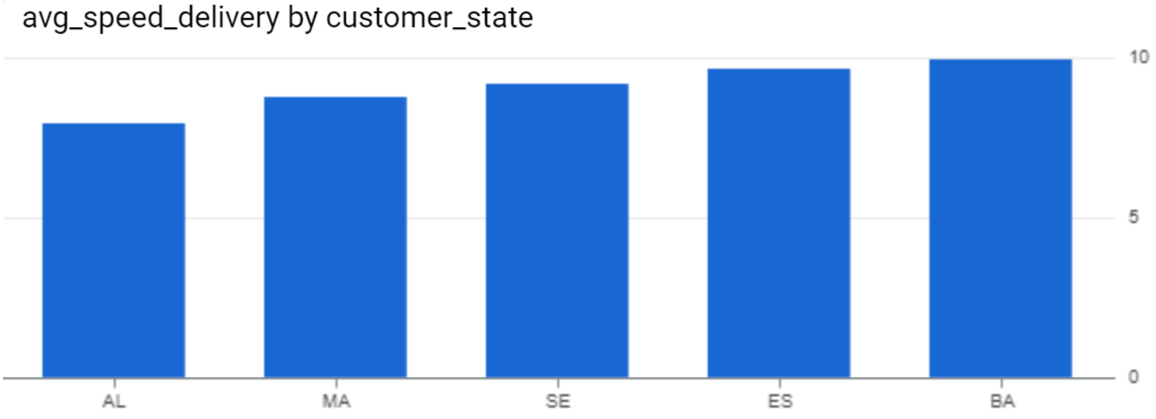


Row	customer_state	avg_speed_delivery
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
502 ms	476 ms	4.17 MB	0 B



Insights:-

1. The query outputs the top five states were order delivery time is really fast as compared to the estimated delivery
2. The state named AL has speed delivery when compared to others states.

Recommendations:-

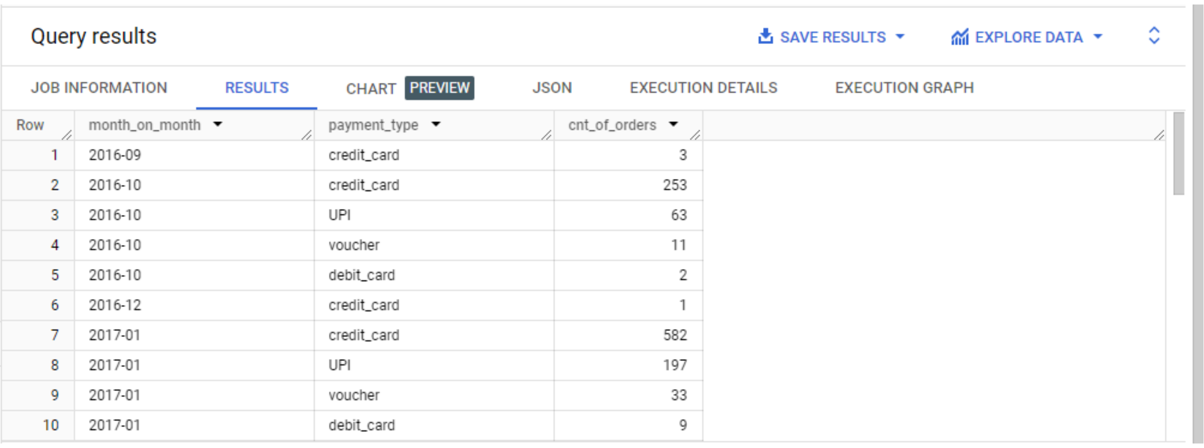
1. In these states the marketing strategies like one-day delivery should be implemented in order to attract buyers and increase number of orders.

6. Analysis based on the payments

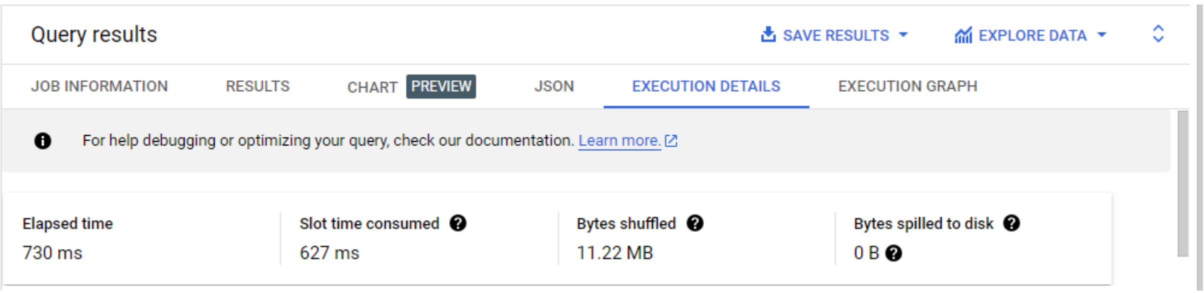
A. Find the month on month no. of orders placed using different payment types.

```
select
  FORMAT_DATE('%Y-%m',o.order_purchase_timestamp) as month_on_month,
  p.payment_type,
  count(distinct o.order_id) as cnt_of_orders
from casestudy_target.orders o  join
casestudy_target.payments p
on p.order_id=o.order_id
group by month_on_month,p.payment_type
order by month_on_month;
```

Query Execution details : Elapsed time : 730 ms
Slot time consumed: 627 ms

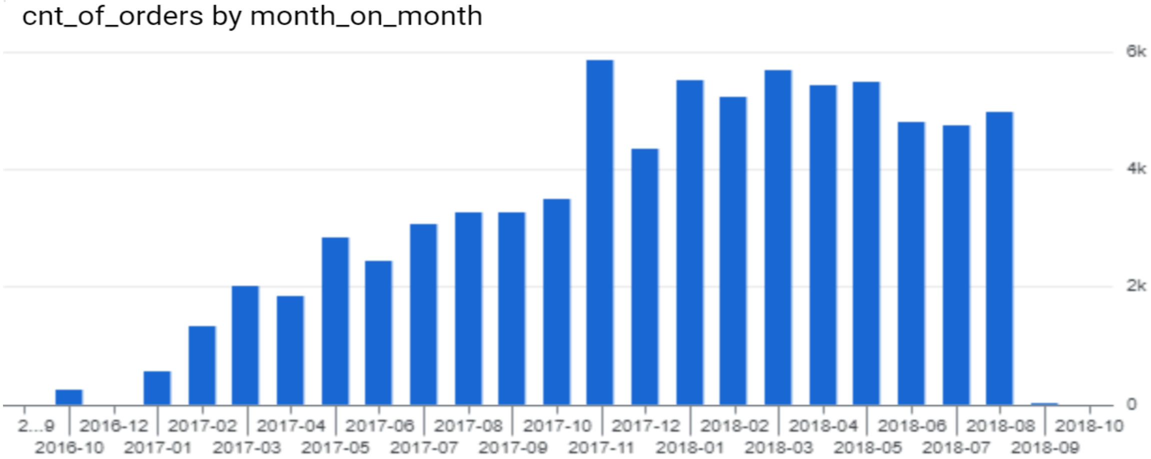


Row	month_on_month	payment_type	cnt_of_orders
1	2016-09	credit_card	3
2	2016-10	credit_card	253
3	2016-10	UPI	63
4	2016-10	voucher	11
5	2016-10	debit_card	2
6	2016-12	credit_card	1
7	2017-01	credit_card	582
8	2017-01	UPI	197
9	2017-01	voucher	33
10	2017-01	debit_card	9



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
730 ms	627 ms	11.22 MB	0 B



Insights:-

1. The query here results the count of orders for different payment types (credit card, UPI, voucher, debit card) across various months.
2. It gives us which type of payment mode is more prefred by customers.

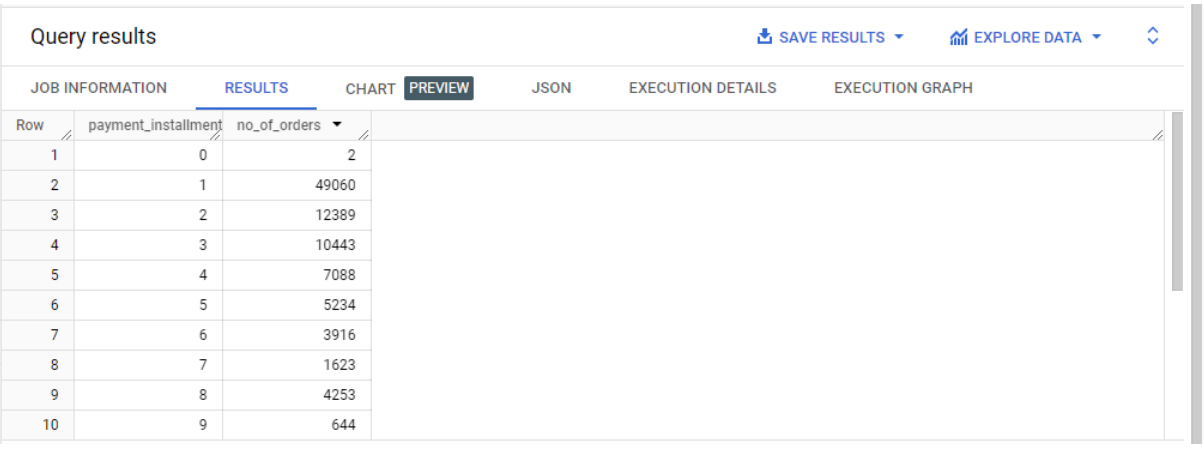
Recommendations:-

1. Analyze the fluctuations in payment types across months to understand seasonal trends or events that influence payment preferences
2. Adding more types of payments favors more customers.

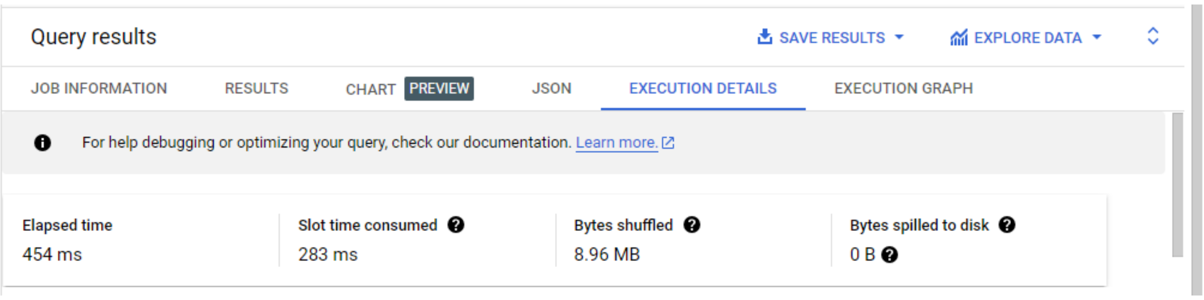
B. Find the no. of orders placed on the basis of the payment installments that have been paid.

```
select
  p.payment_installments,
  count(distinct o.order_id) as no_of_orders
 from `casestudy_target.orders` o join `casestudy_target.payments` p
 on p.order_id = o.order_id
 group by p.payment_installments
 order by p.payment_installments
```

```
Query Execution details : Elapsed time : 454 ms
                         Slot time consumed: 283 ms
```



Row	payment_installment	no_of_orders
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644



For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
454 ms	283 ms	8.96 MB	0 B

Insights:-

1. The query output shows us the number of orders made with various installment plans.
2. Understanding the count of payment instalments helps us in offering payment plans.
3. Separate customers based on their instalments count this helps in targeted marketing.

Recommendations:-

1. Telling the customers the advantages of installment plans
2. Provide clear information on terms, fees, and payment schedules in order to build customer confidence.