This document contains exploratory analysis and in-depth exploration of the trends of purchase, delivery, items, payments, sellers and products of Target ecommerce business in Brazil. SQL queries have been used to pull data to analyse the impact of the company's business on the economy. Furthermore, analysis of sales, freight and payment methods have been conducted. This is a non-exhaustive study and only provides an overview of the business in the different states of Brazil.

Target Business Case

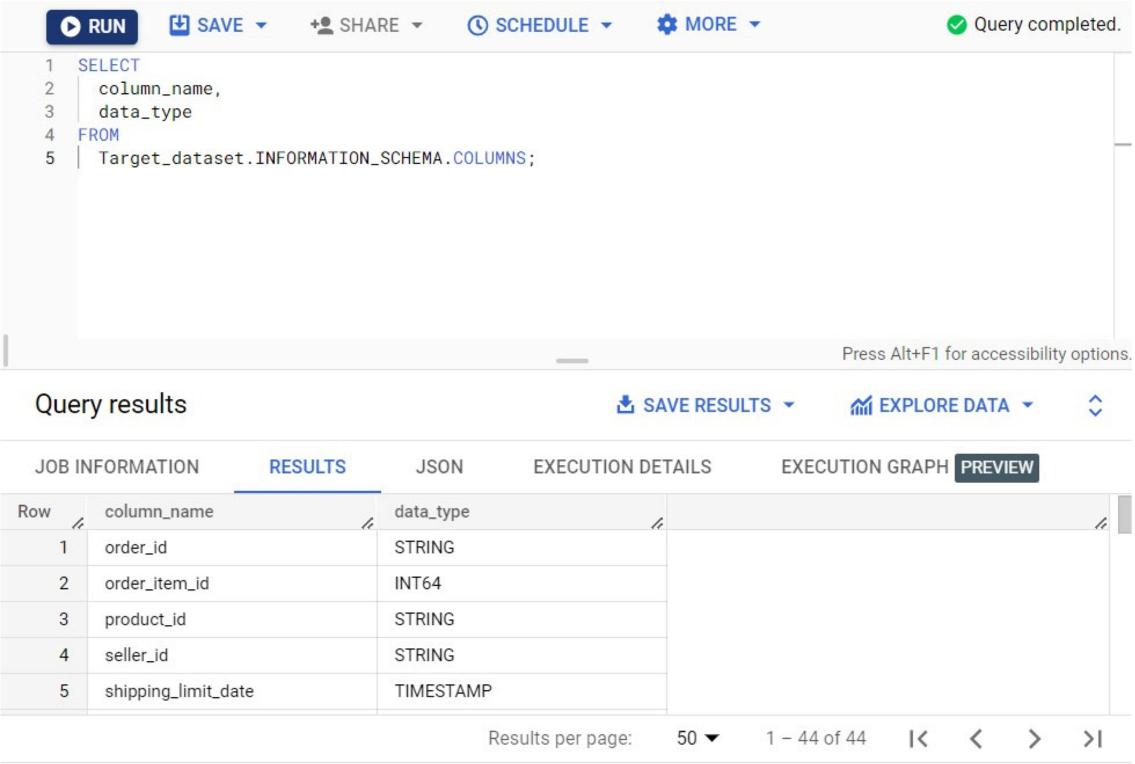
Analysis of Data Using BigQuery

Turya Ganguly

07.03.2023

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

1.1 Data type of columns in a table



```
SELECT
  column_name,
  data_type
FROM
  Target_dataset.INFORMATION_SCHEMA.COLUMNS;
```

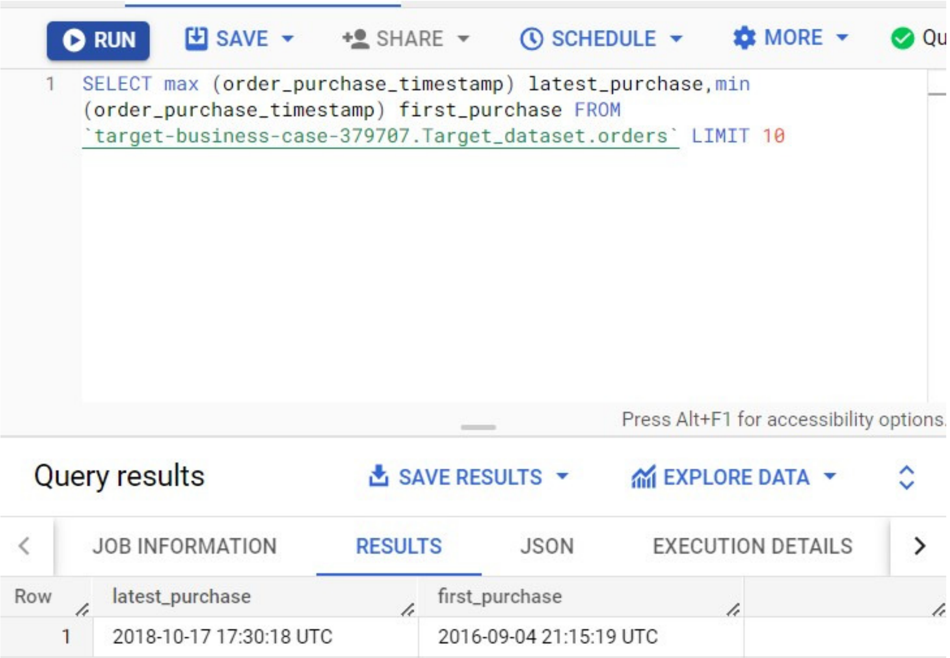
Row	column_name	data_type
1	order_id	STRING
2	order_item_id	INT64
3	product_id	STRING
4	seller_id	STRING
5	shipping_limit_date	TIMESTAMP

geolocation table	
geolocation_zip_code_prefix	Integer
geolocation_lat	Float
geolocation_lng	Float
geolocation_city	String
geolocation_state	String
order_items table	
order_id	String
order_item_id	Integer
product_id	String
seller_id	String
shipping_limit_date	Timestamp
price	Float
freight_value	Float
orders_review table	
review_id	String
order_id	String
review_score	Integer
review_comment_title	String

review_creation_date	Timestamp
review_answer_timestamp	Timestamp
orders table	
order_id	String
customer_id	String
order_status	String
order_purchase_timestamp	Timestamp
order_approved_at	Timestamp
order_delivered_carrier_date	Timestamp
order_delivered_customer_date	Timestamp
order_estimated_delivery_date	Timestamp
payments table	
order_id	String
payment_sequential	Integer
payment_type	String
payment_installments	Integer
payment_value	Float
products table	
product_id	String
product_category	String
product_name_length	Integer
product_description_length	Integer
product_photos_qty	Integer
product_weight_g	Integer
product_length_cm	Integer
product_height_cm	Integer
product_width_cm	Integer
sellers table	
seller_id	String
seller_zip_code_prefix	Integer
seller_city	String
seller_state	String

1.2 Time period for which the data is given

The orders table have been used in this problem. Order_purchase_timestamp provides the exact time at which purchases have been made. Therefore, max and min functions can be used to pull out the latest and the first purchase. The result below shows that the first purchase was made on 4th September 2016 while the date of the latest purchase is 17th October 2018.

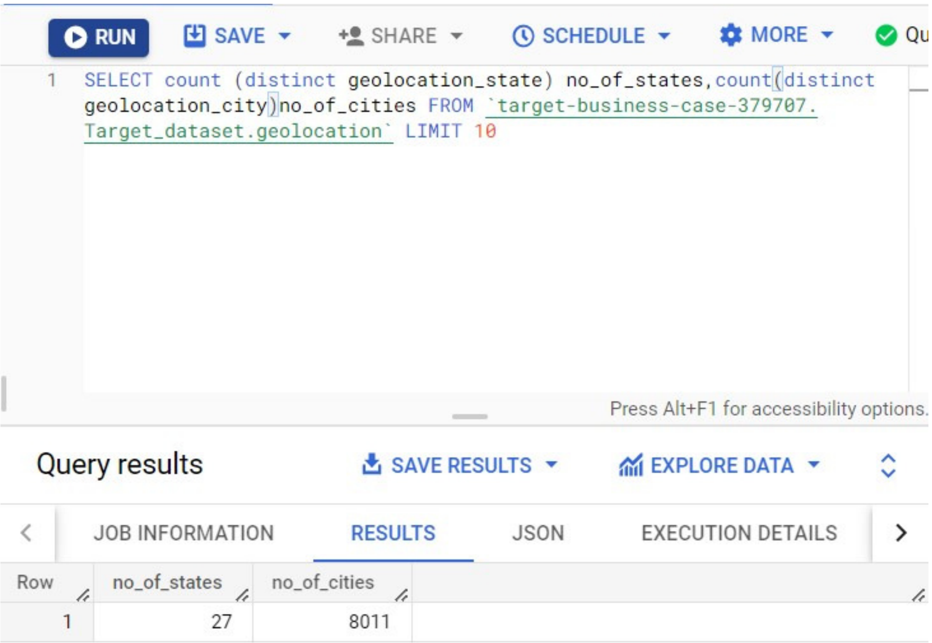


```
SELECT max (order_purchase_timestamp) latest_purchase,min
(order_purchase_timestamp) first_purchase FROM
`target-business-case-379707.Target_dataset.orders` LIMIT 10
```

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

1.3 Cities and States of customers ordered during the given period

This problem simply asks to count the cities and states during the above period. We use distinct to avoid repetitions from the geolocation table and get the correct number of states and cities in the results.



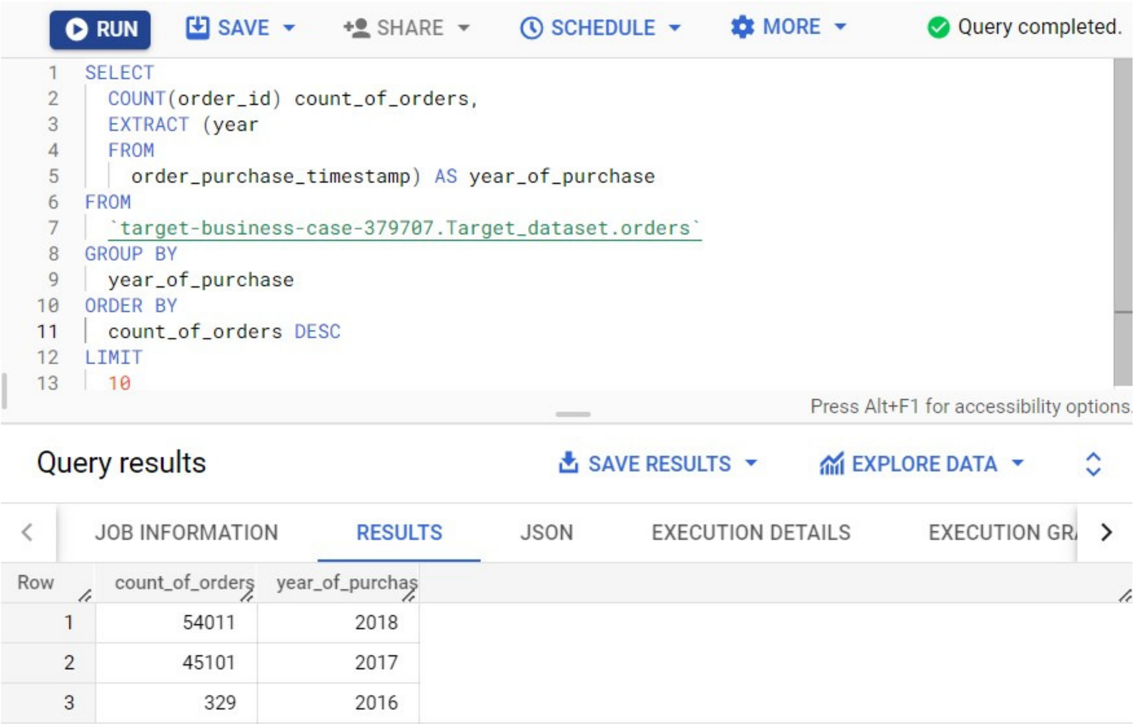
```
SELECT count (distinct geolocation_state) no_of_states,count(distinct
geolocation_city)no_of_cities FROM `target-business-case-379707.
Target_dataset.geolocation` LIMIT 10
```

Row	no_of_states	no_of_cities
1	27	8011

2. In-depth Exploration:

2.1 Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

For this problem, we have to check the increase of orders over time. So, for the first part, we “extract” the year from purchase order timestamp and the count of orders. We can group the query by year of purchase and order by count of orders, either ascending or descending. The increase of orders from 2016 to 2018 shows that there is a need for disruption in the ecommerce section of the business.

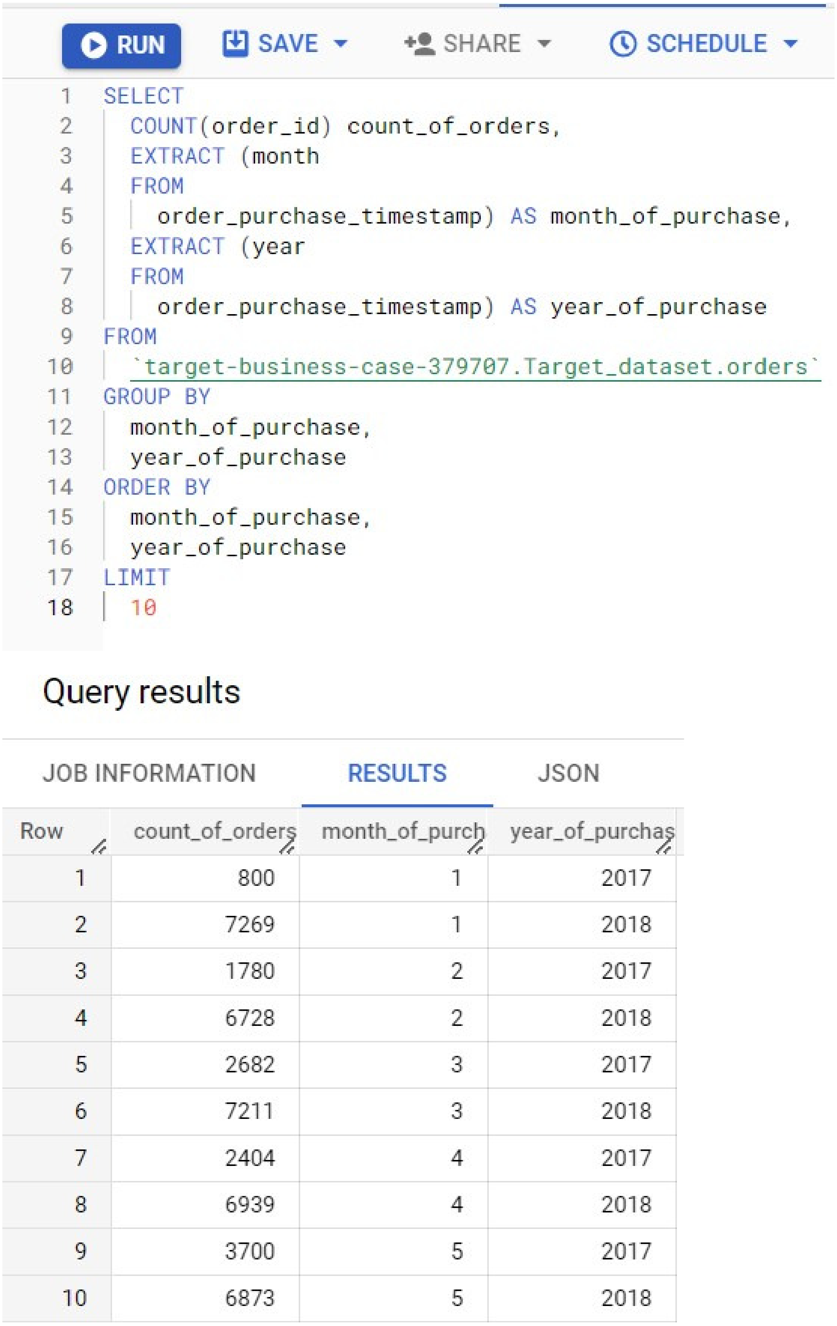


```
SELECT
  COUNT(order_id) count_of_orders,
  EXTRACT (year
  FROM
    order_purchase_timestamp) AS year_of_purchase
FROM
  `target-business-case-379707.Target_dataset.orders`
GROUP BY
  year_of_purchase
ORDER BY
  count_of_orders DESC
LIMIT
  10
```

Row	count_of_orders	year_of_purchas
1	54011	2018
2	45101	2017
3	329	2016

It can be easily seen from the above query that the number of orders placed has increased by a great margin from 2016 to 2017 and there is an 19.75% percent increase in orders from 2017 to 2018 as well. Hence, we can conclude that there is an increasing trend for ecommerce in Brazil.

Now let’s see the seasonality of the same in terms of months of the years. As we see from the query results below, January 2018 has 7269 orders in comparison to 800 orders in January 2017. This is an increase of almost 10%. This pattern is true for every month of 2017 and 2018. The first month of orders that is available from the above data is for September 2016 when there were only 4 orders.



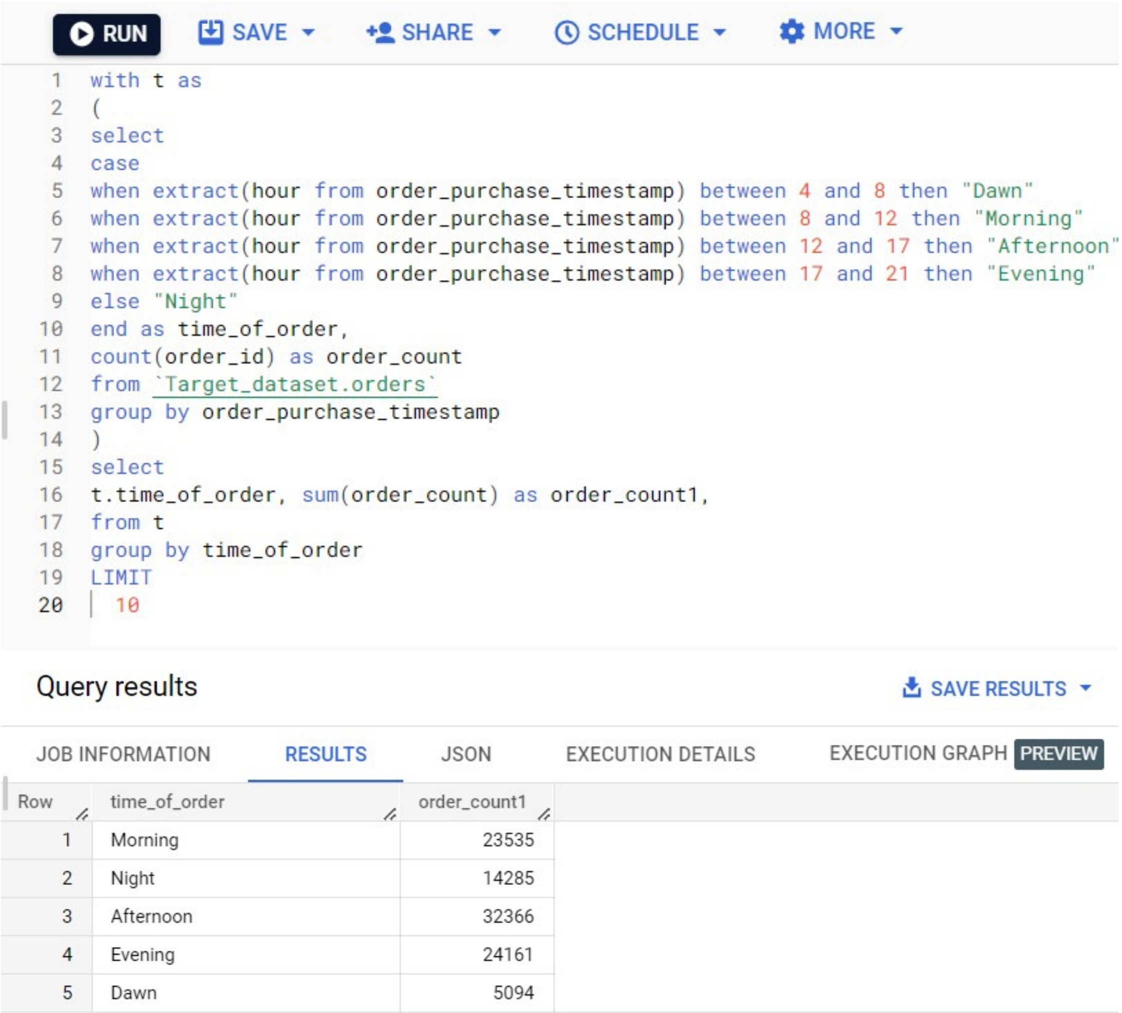
```
SELECT
  COUNT(order_id) count_of_orders,
  EXTRACT (month
  FROM
    order_purchase_timestamp) AS month_of_purchase,
  EXTRACT (year
  FROM
    order_purchase_timestamp) AS year_of_purchase
FROM
  `target-business-case-379707.Target_dataset.orders`
GROUP BY
  month_of_purchase,
  year_of_purchase
ORDER BY
  month_of_purchase,
  year_of_purchase
LIMIT
  10
```

Query results

Row	count_of_orders	month_of_purch	year_of_purchas
1	800	1	2017
2	7269	1	2018
3	1780	2	2017
4	6728	2	2018
5	2682	3	2017
6	7211	3	2018
7	2404	4	2017
8	6939	4	2018
9	3700	5	2017
10	6873	5	2018

2.2 What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

From the below results it is quite evident that Brazilian customers tend to purchase mostly in the morning, followed by night, afternoon and evening in that order. Customers tend to purchase the least in dawn time.



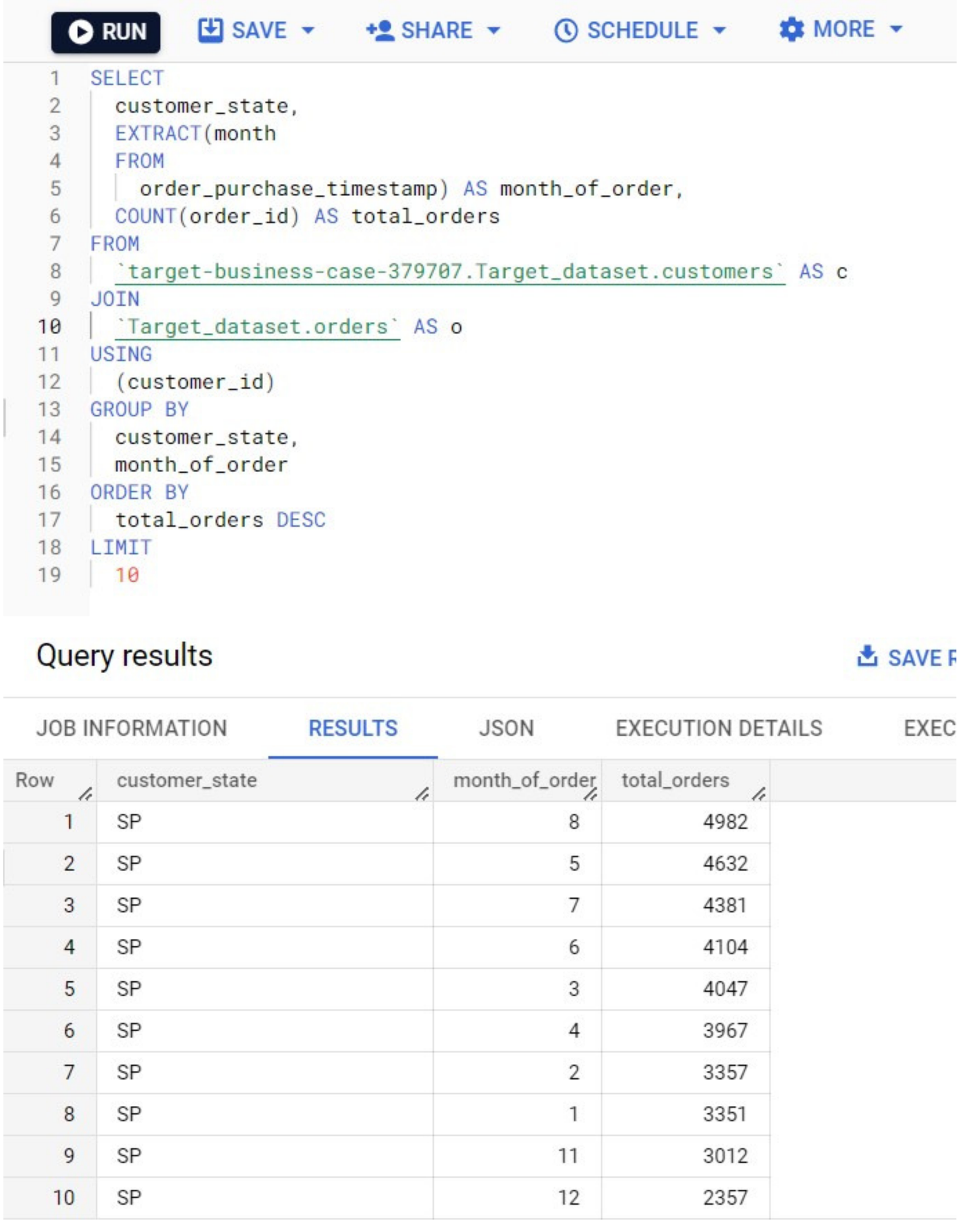
```sql
with t as
(
select
case
when extract(hour from order_purchase_timestamp) between 4 and 8 then "Dawn"
when extract(hour from order_purchase_timestamp) between 8 and 12 then "Morning"
when extract(hour from order_purchase_timestamp) between 12 and 17 then "Afternoon"
when extract(hour from order_purchase_timestamp) between 17 and 21 then "Evening"
else "Night"
end as time_of_order,
count(order_id) as order_count
from `Target_dataset.orders`
group by order_purchase_timestamp
)
select
t.time_of_order, sum(order_count) as order_count1,
from t
group by time_of_order
LIMIT
  10
```

Row	time_of_order	order_count1
1	Morning	23535
2	Night	14285
3	Afternoon	32366
4	Evening	24161
5	Dawn	5094

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

3.1 Get month on month orders by states

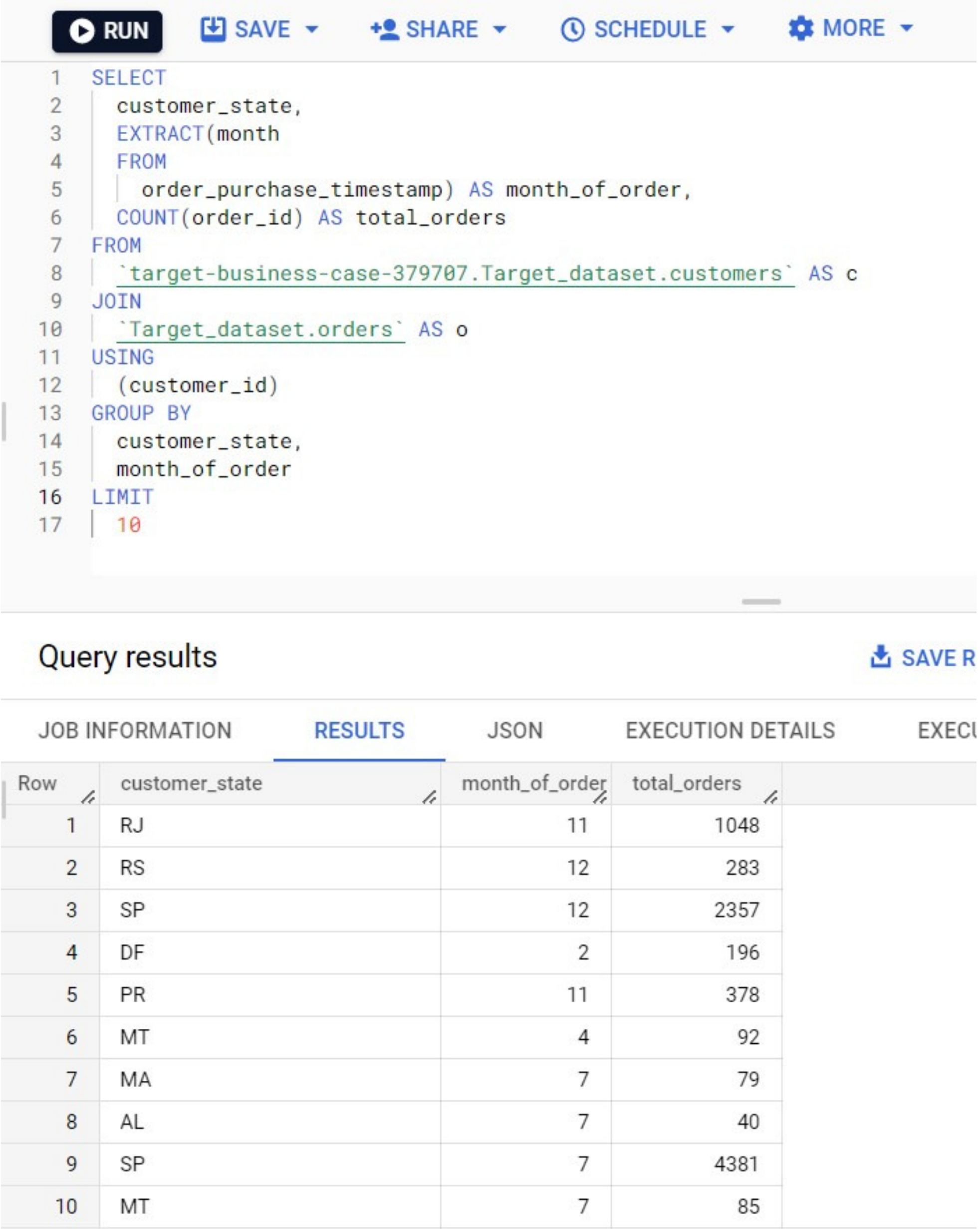
Sao Paulo is the state with the most orders in the country, as evident from the result below when we use order by count (order_id) total orders. If we don't need to count the total orders, we get the results grouped by customer_state and month_of_order. We can alternatively order by month_of_order to check which month had the best sales, or order by customer_state to see which state is the leading purchaser. This hasn't been shown in the results but can be easily achieved by using order by customer_state and/or month_of_order.



```
SELECT
  customer_state,
  EXTRACT(month
  FROM
    order_purchase_timestamp) AS month_of_order,
  COUNT(order_id) AS total_orders
FROM
  `target-business-case-379707.Target_dataset.customers` AS c
JOIN
  `Target_dataset.orders` AS o
USING
  (customer_id)
GROUP BY
  customer_state,
  month_of_order
ORDER BY
  total_orders DESC
LIMIT
  10
```

Query results

Row	customer_state	month_of_order	total_orders
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



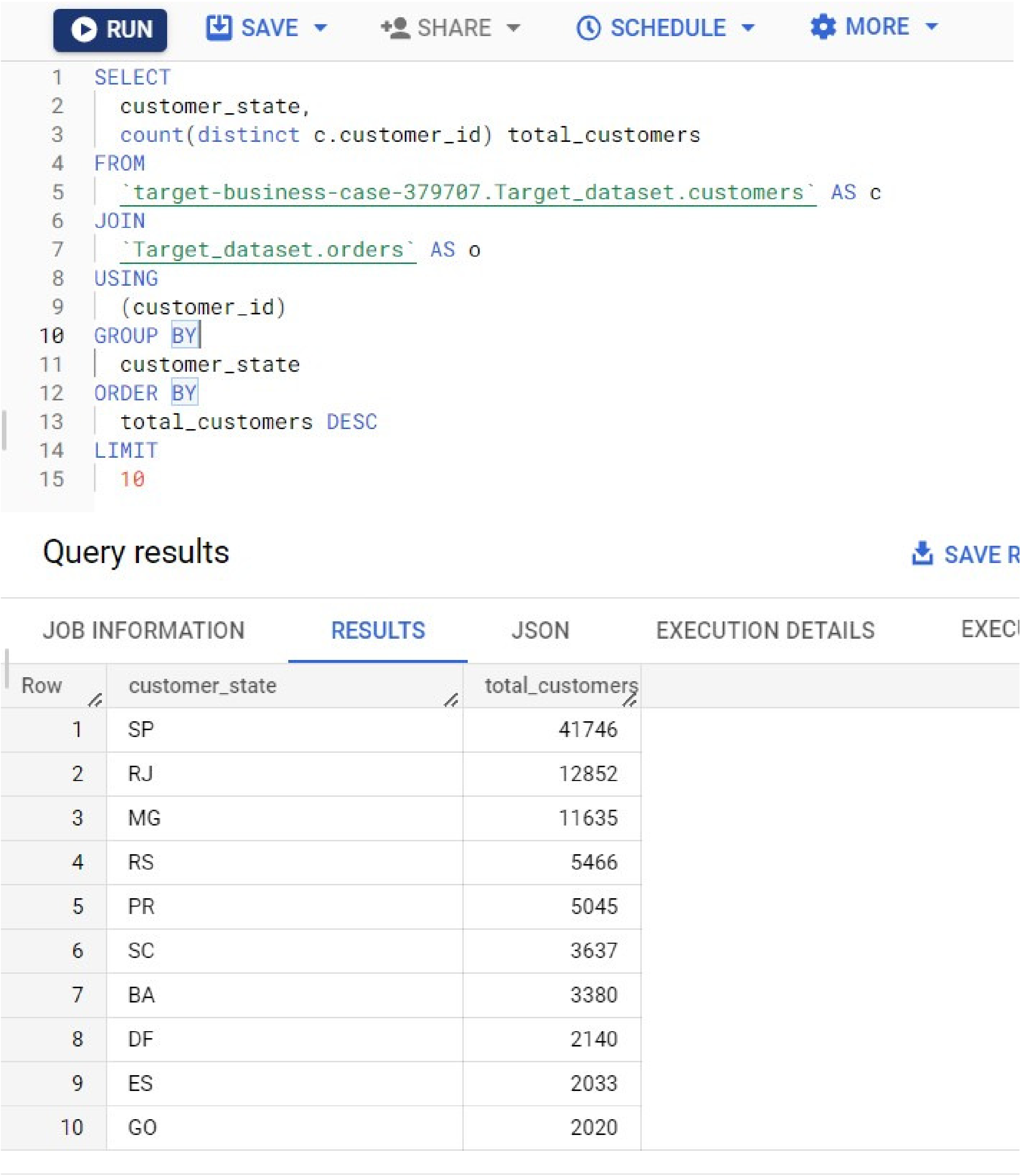
```
SELECT
  customer_state,
  EXTRACT(month
  FROM
    order_purchase_timestamp) AS month_of_order,
  COUNT(order_id) AS total_orders
FROM
  `target-business-case-379707.Target_dataset.customers` AS c
JOIN
  `Target_dataset.orders` AS o
USING
  (customer_id)
GROUP BY
  customer_state,
  month_of_order
LIMIT
  10
```

Query results

Row	customer_state	month_of_order	total_orders
1	RJ	11	1048
2	RS	12	283
3	SP	12	2357
4	DF	2	196
5	PR	11	378
6	MT	4	92
7	MA	7	79
8	AL	7	40
9	SP	7	4381
10	MT	7	85

3.2 Distribution of customers across the states in Brazil

The following table shows the distribution of customers across states in Brazil. The fields selected are the states to which the customers belong and total customers in each of these states. In this regard, it deserves mention that distinct statement has to used to make sure that unique values of customer_id is selected. The query is ordered by total customers in descending order. The result shows the top 10 states by number of unique customers.



```
SELECT
  customer_state,
  count(distinct c.customer_id) total_customers
FROM
  `target-business-case-379707.Target_dataset.customers` AS c
JOIN
  `Target_dataset.orders` AS o
USING
  (customer_id)
GROUP BY
  customer_state
ORDER BY
  total_customers DESC
LIMIT
  10
```

Row	customer_state	total_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

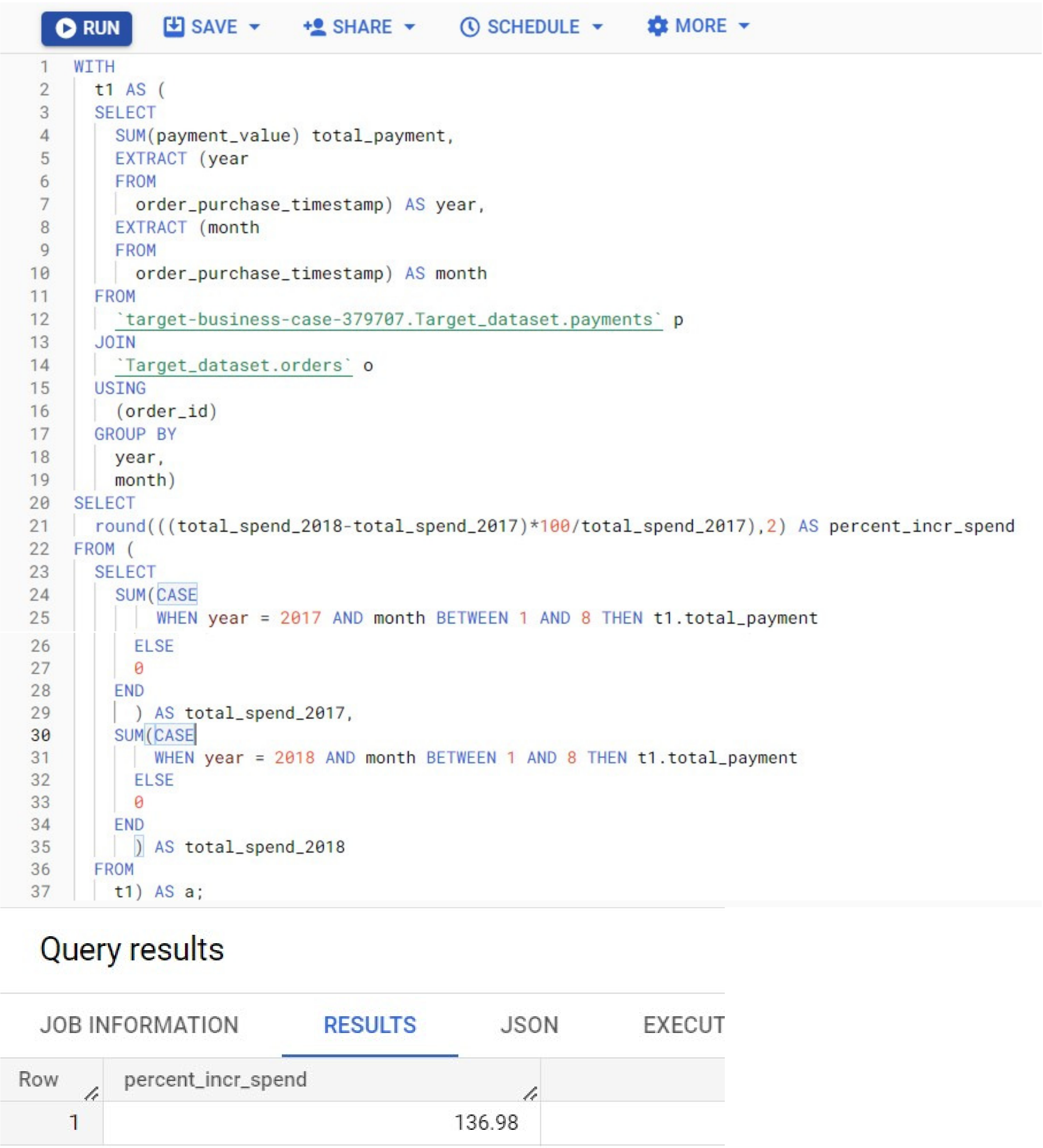
4. Impact on Economy: Analyse the money movement by e-commerce by looking at order prices, freight and others.

4.1 Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only)- You can use “payment_value” column in payments table

The percentage increase in cost of orders from 2017 to 2018 for the months from Jan to Aug is 136.98%.

From what can be understood, the sum of all the orders for the months from January to August in 2017 have to compared to the sum of the same months in 2018. For this purpose, a Common Table

Expression (CTE) has been created where the total_payment variable has been defined, year and month have been extracted. This CTE has been used for the next select statement, where two case statements have been used to get the total spend for 2017 and 2018 separately. Percentage formula has then been used to find the percentage increase of cost of orders from 2017 to 2018.



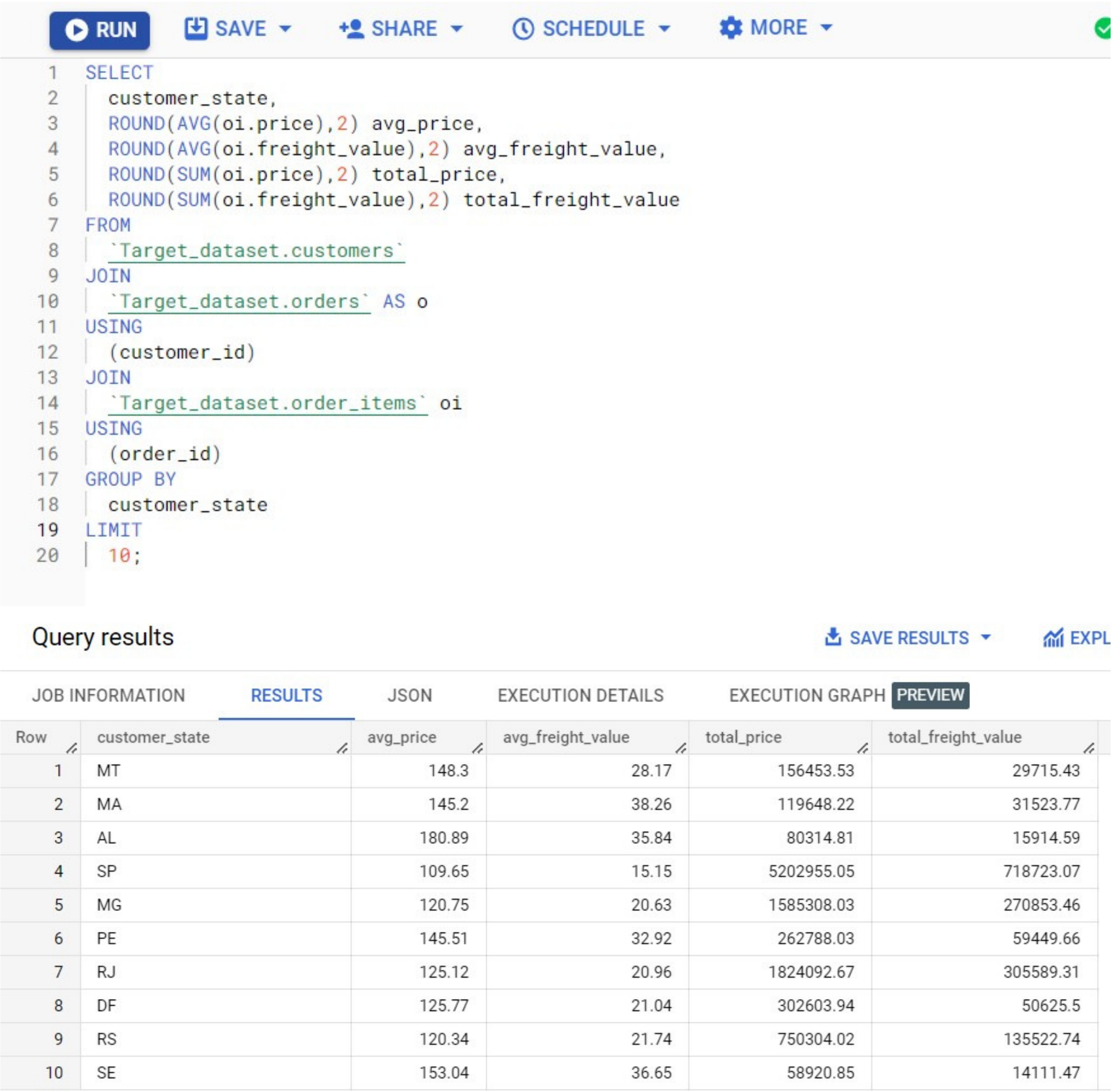
```
WITH
  t1 AS (
  SELECT
    SUM(payment_value) total_payment,
    EXTRACT (year
    FROM
      order_purchase_timestamp) AS year,
    EXTRACT (month
    FROM
      order_purchase_timestamp) AS month
  FROM
    `target-business-case-379707.Target_dataset.payments` p
  JOIN
    `Target_dataset.orders` o
  USING
    (order_id)
  GROUP BY
    year,
    month)
SELECT
  round(((total_spend_2018-total_spend_2017)*100/total_spend_2017),2) AS percent_incr_spend
FROM (
  SELECT
    SUM(CASE
        WHEN year = 2017 AND month BETWEEN 1 AND 8 THEN t1.total_payment
      ELSE
      0
    END
      ) AS total_spend_2017,
    SUM(CASE
        WHEN year = 2018 AND month BETWEEN 1 AND 8 THEN t1.total_payment
      ELSE
      0
    END
      ) AS total_spend_2018
  FROM
    t1) AS a;
```

Query results

Row	percent_incr_spend
1	136.98

4.2 Mean & Sum of price and freight value by customer state

Order_items has been joined with customers and orders table in this problem to get all the relevant fields in this problem. Mean and total price and freight value has been obtained from the order items table. They have been grouped by the states of the customers and 10 results have been shown in the results.



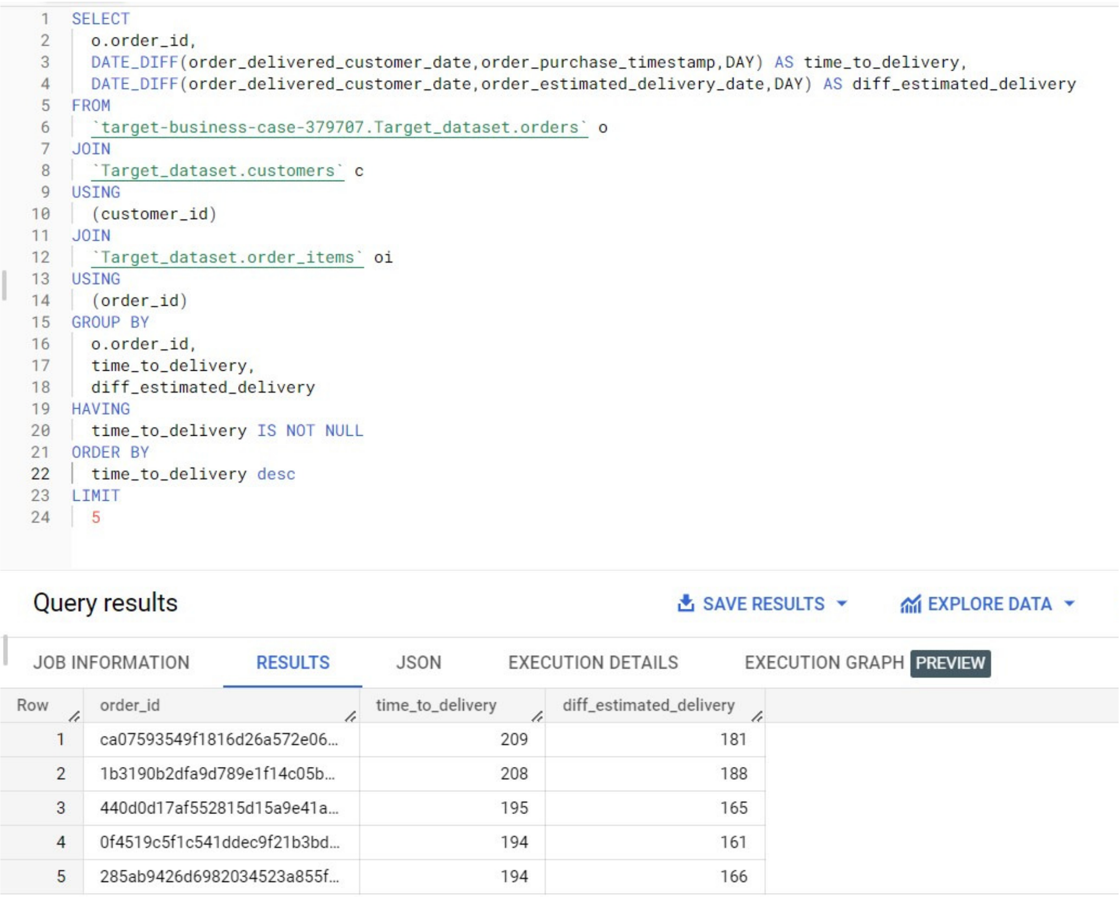
```sql
SELECT
  customer_state,
  ROUND(AVG(oi.price),2) avg_price,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(SUM(oi.price),2) total_price,
  ROUND(SUM(oi.freight_value),2) total_freight_value
FROM
  `Target_dataset.customers`
JOIN
  `Target_dataset.orders` AS o
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  customer_state
LIMIT
  10;
```

Row	customer_state	avg_price	avg_freight_value	total_price	total_freight_value
1	MT	148.3	28.17	156453.53	29715.43
2	MA	145.2	38.26	119648.22	31523.77
3	AL	180.89	35.84	80314.81	15914.59
4	SP	109.65	15.15	5202955.05	718723.07
5	MG	120.75	20.63	1585308.03	270853.46
6	PE	145.51	32.92	262788.03	59449.66
7	RJ	125.12	20.96	1824092.67	305589.31
8	DF	125.77	21.04	302603.94	50625.5
9	RS	120.34	21.74	750304.02	135522.74
10	SE	153.04	36.65	58920.85	14111.47

5. Analysis on sales, freight and delivery time

5.1 Find time_to_delivery & diff_estimated_delivery

Order Id is selected and difference of dates is obtained by DATEDIFF function. The DATEDIFF function helps in finding the difference between purchase of the product and the delivery date. This has been aliased as time to delivery. The difference between estimated delivery of the order and the actual date of delivery is pulled using the same function. The issue with the dataset is that there are a lot of null values. Therefore, we have to use having is not null function to get the non-null values. The query is ordered by time to delivery in descending fashion to get a feel of the data and to check whether the query is getting the right results.

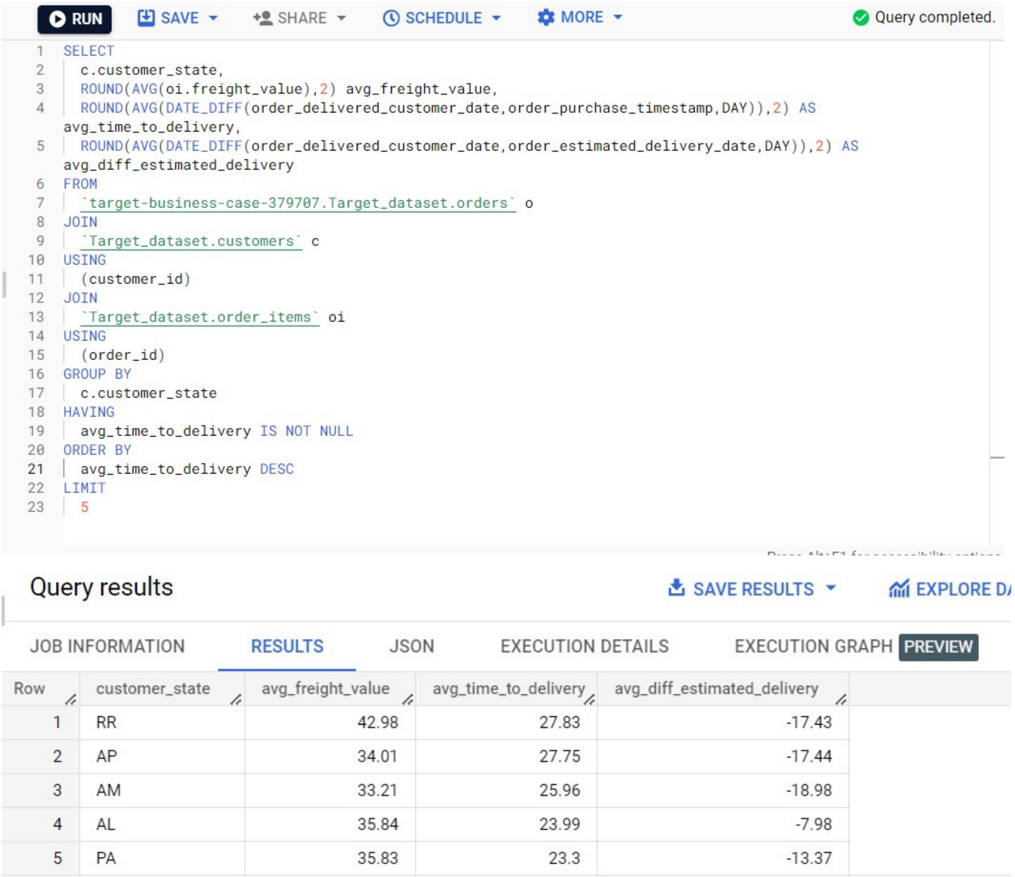


```
SELECT
  o.order_id,
  DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS time_to_delivery,
  DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY) AS diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  o.order_id,
  time_to_delivery,
  diff_estimated_delivery
HAVING
  time_to_delivery IS NOT NULL
ORDER BY
  time_to_delivery desc
LIMIT
  5
```

Row	order_id	time_to_delivery	diff_estimated_delivery
1	ca07593549f1816d26a572e06...	209	181
2	1b3190b2dfa9d789e1f14c05b...	208	188
3	440d0d17af552815d15a9e41a...	195	165
4	0f4519c5f1c541ddec9f21b3bd...	194	161
5	285ab9426d6982034523a855f...	194	166

5.2 Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

In this section of the problem, mean of freight value, time to delivery and difference in delivery estimation has been taken. The data has been ordered by from the highest to the lowest average time of delivery for the first 5 states. The top 5 states are RR, AP, AM, AL and PA.

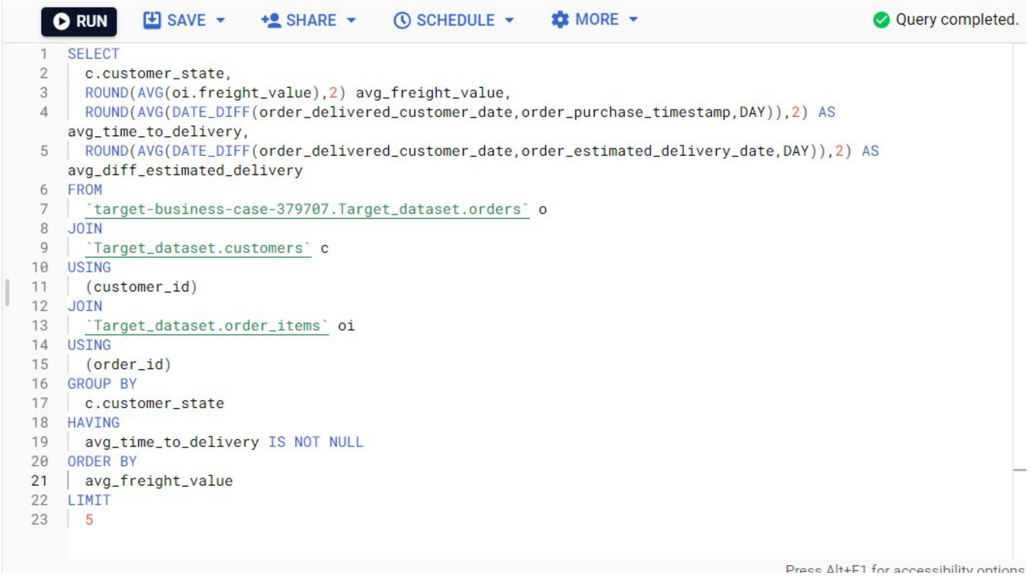


```sql
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_time_to_delivery DESC
LIMIT
  5
```

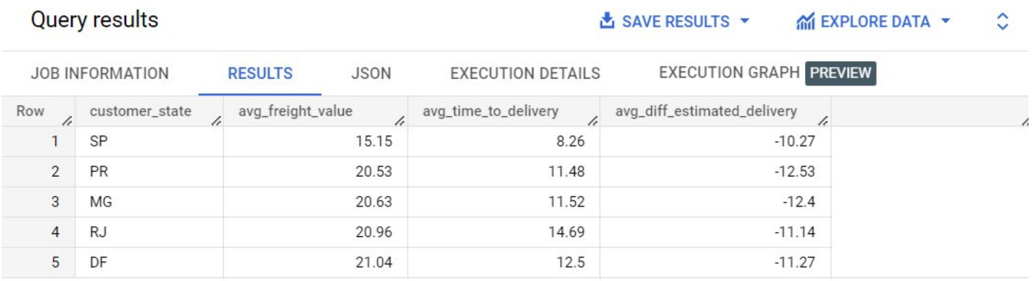
Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	RR	42.98	27.83	-17.43
2	AP	34.01	27.75	-17.44
3	AM	33.21	25.96	-18.98
4	AL	35.84	23.99	-7.98
5	PA	35.83	23.3	-13.37

5.3 Top 5 states with lowest average freight value

Here the only change of the query from the above one is that it is ordered by average freight value in ascending order.



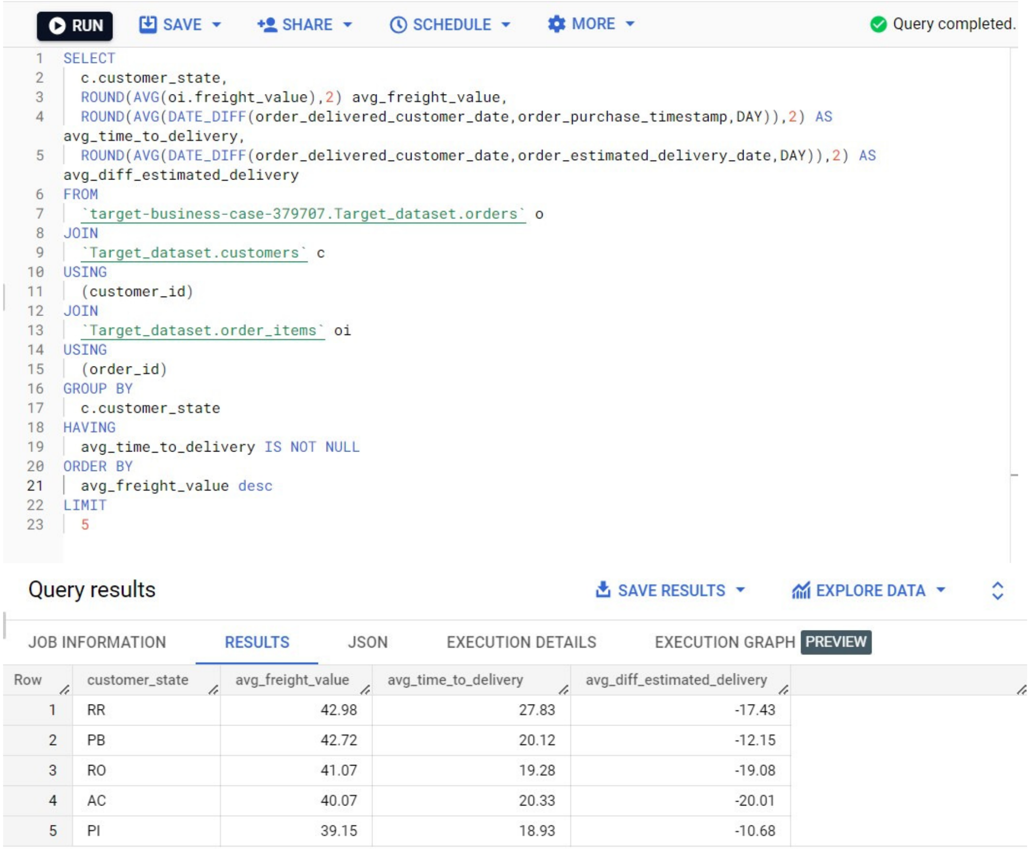
```sql
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_freight_value
LIMIT
  5
```



Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.15	8.26	-10.27
2	PR	20.53	11.48	-12.53
3	MG	20.63	11.52	-12.4
4	RJ	20.96	14.69	-11.14
5	DF	21.04	12.5	-11.27

5.4 Top 5 states with highest average freight value

Here the query has been arranged in descending of average freight value and the top 5 values have been displayed.

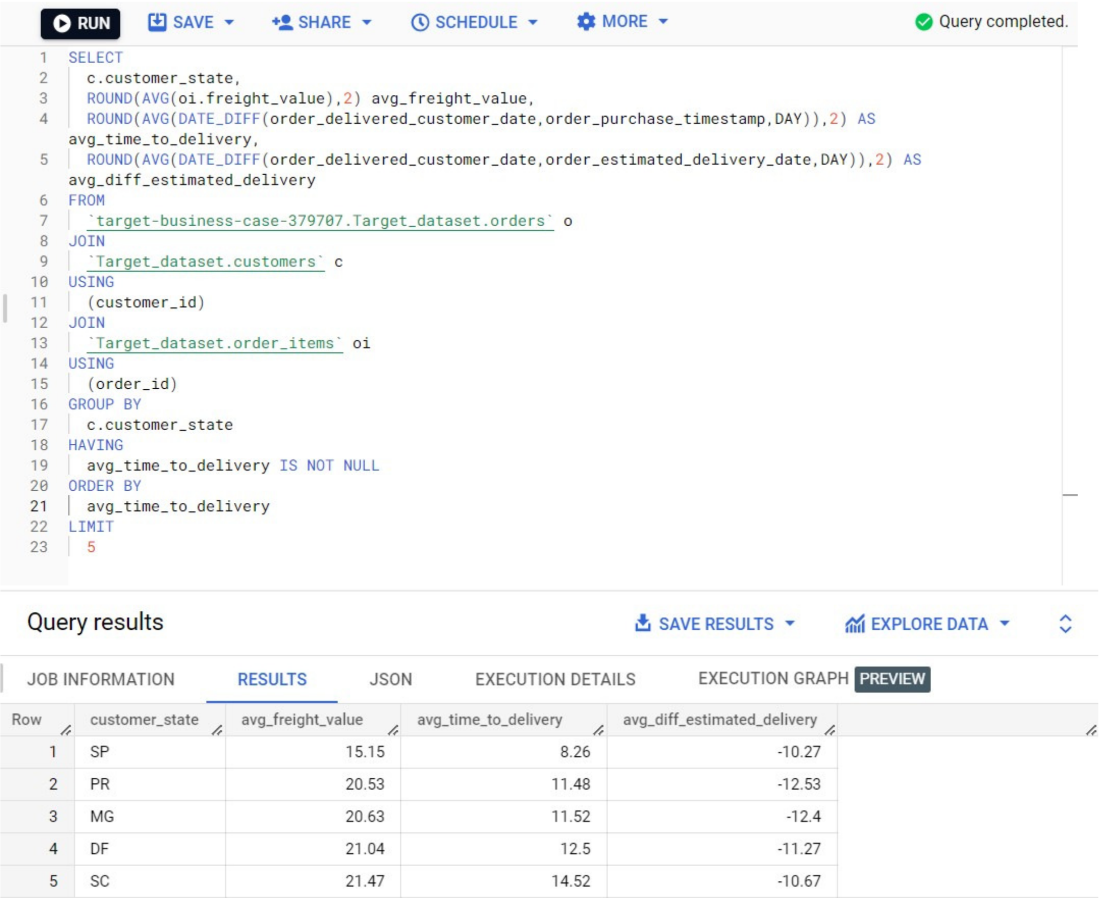


```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS
avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_freight_value desc
LIMIT
  5
```

Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	RR	42.98	27.83	-17.43
2	PB	42.72	20.12	-12.15
3	RO	41.07	19.28	-19.08
4	AC	40.07	20.33	-20.01
5	PI	39.15	18.93	-10.68

5.5 Top 5 states with lowest average time to delivery

In this part of the problem, the query has been arranged in ascending of average time to delivery and the top 5 values have been displayed.

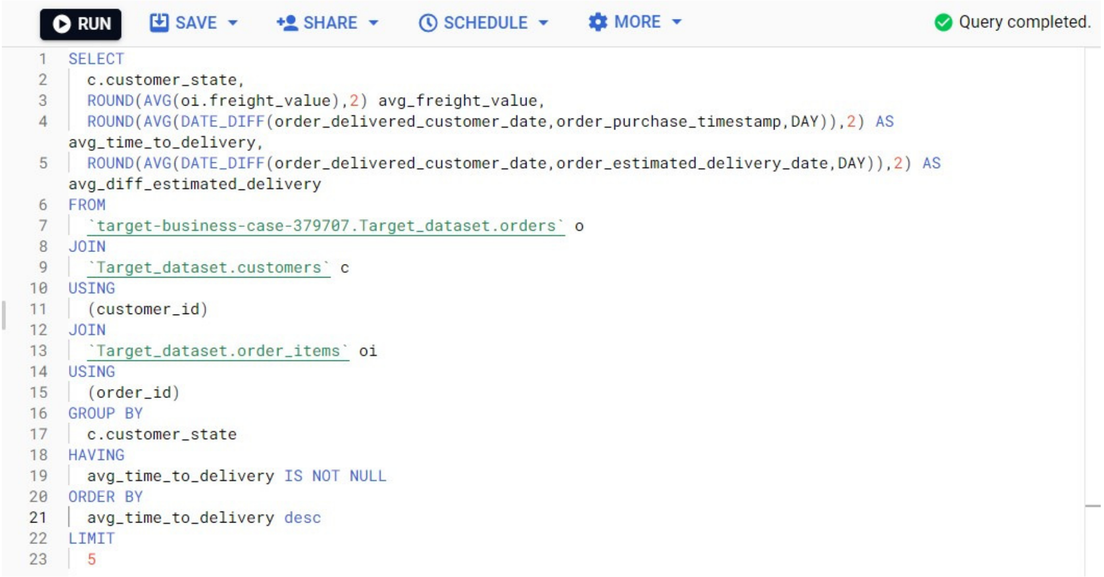


```sql
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_time_to_delivery
LIMIT
  5
```

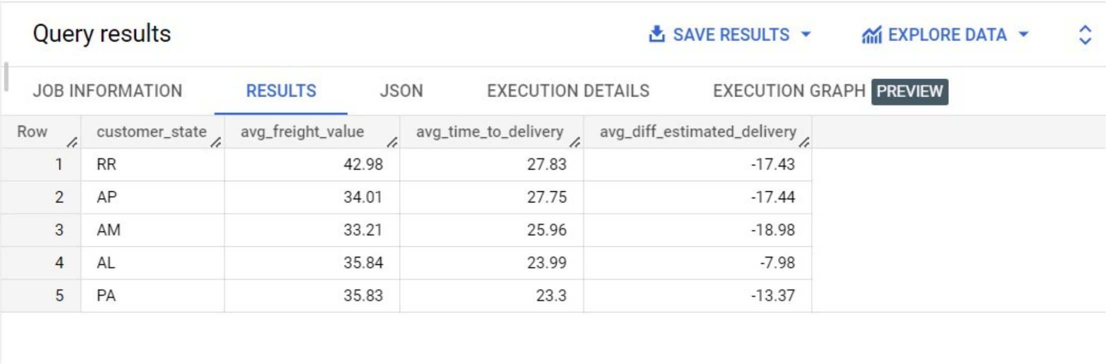
Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.15	8.26	-10.27
2	PR	20.53	11.48	-12.53
3	MG	20.63	11.52	-12.4
4	DF	21.04	12.5	-11.27
5	SC	21.47	14.52	-10.67

5.6 Top 5 states with highest average time to delivery

In this part of the problem, the query has been arranged in descending of average time to delivery and the top 5 values have been displayed.



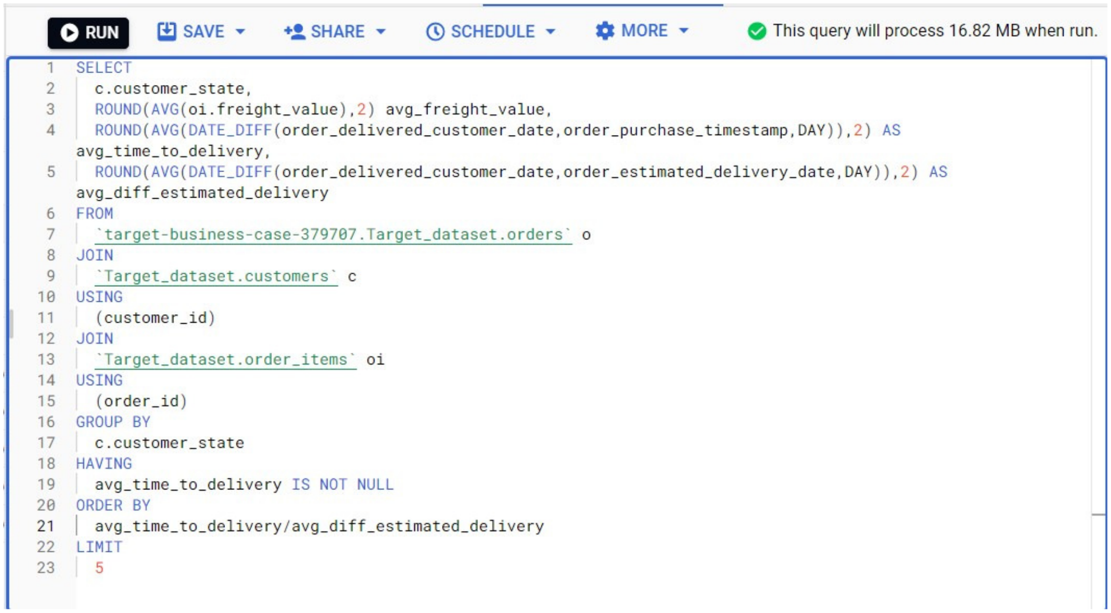
```sql
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_time_to_delivery desc
LIMIT
  5
```



Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	RR	42.98	27.83	-17.43
2	AP	34.01	27.75	-17.44
3	AM	33.21	25.96	-18.98
4	AL	35.84	23.99	-7.98
5	PA	35.83	23.3	-13.37

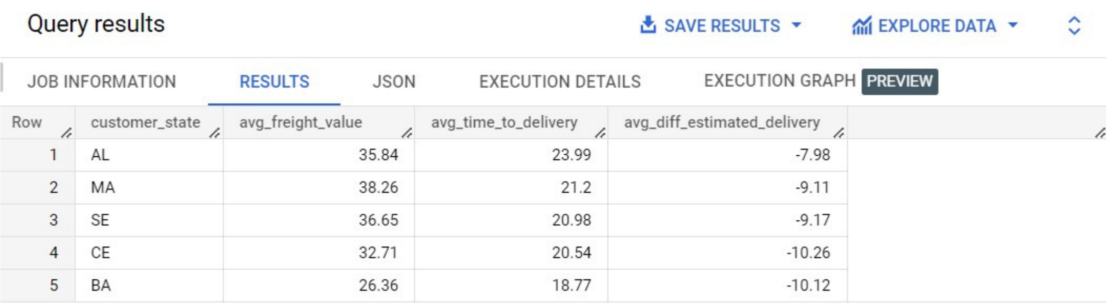
5.7 Top 5 states where delivery is not so fast compared to estimated date

When the ratio of average time to delivery and average difference in estimated delivery and actual delivery is least, the delivery is the slowest.



This query will process 16.82 MB when run.

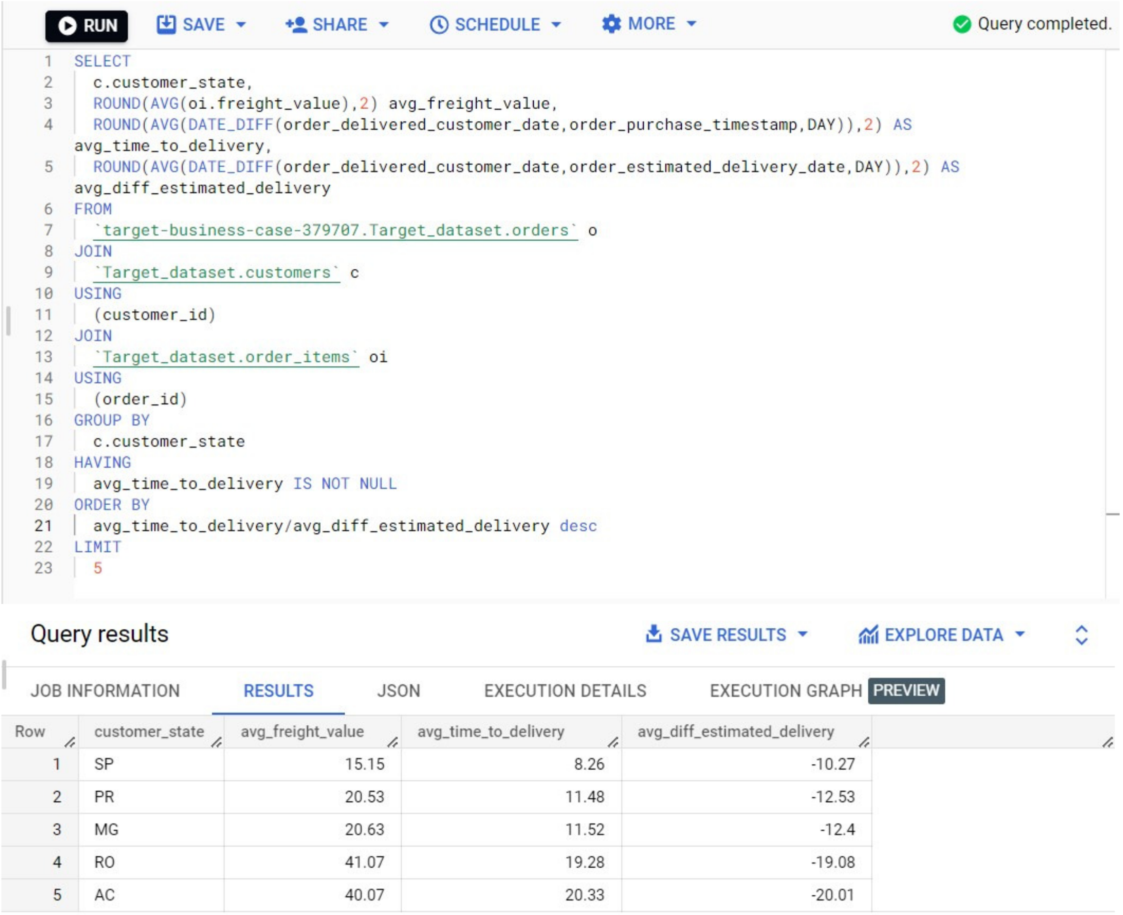
```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS
avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_time_to_delivery/avg_diff_estimated_delivery
LIMIT
  5
```



Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	AL	35.84	23.99	-7.98
2	MA	38.26	21.2	-9.11
3	SE	36.65	20.98	-9.17
4	CE	32.71	20.54	-10.26
5	BA	26.36	18.77	-10.12

5.8 Top 5 states where delivery very fast compared to estimated date

When the ratio of average time to delivery and average difference in estimated delivery and actual delivery is highest, the delivery is the fastest.



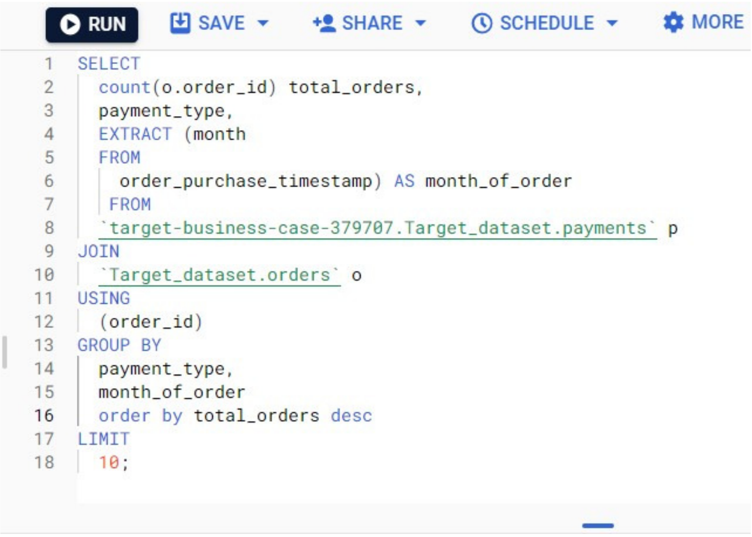
```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value),2) avg_freight_value,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)),2) AS
avg_diff_estimated_delivery
FROM
  `target-business-case-379707.Target_dataset.orders` o
JOIN
  `Target_dataset.customers` c
USING
  (customer_id)
JOIN
  `Target_dataset.order_items` oi
USING
  (order_id)
GROUP BY
  c.customer_state
HAVING
  avg_time_to_delivery IS NOT NULL
ORDER BY
  avg_time_to_delivery/avg_diff_estimated_delivery desc
LIMIT
  5
```

Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.15	8.26	-10.27
2	PR	20.53	11.48	-12.53
3	MG	20.63	11.52	-12.4
4	RO	41.07	19.28	-19.08
5	AC	40.07	20.33	-20.01

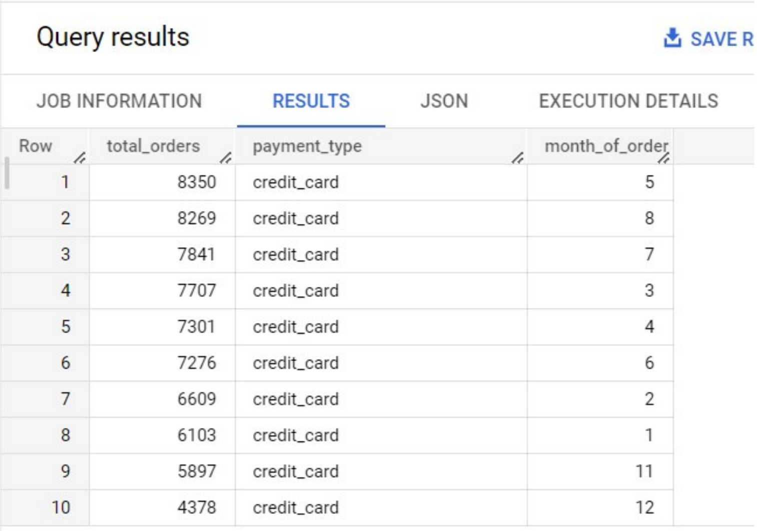
6. Payment type analysis:

6.1 Month over Month count of orders for different payment types

The below results show the count of orders as total_orders, payment_type and month of the order. Orders and payments table have been joined and grouped by payment_type and month_of_order. After ordering the query by total_order in descending order it is observed that the most used / preferred payment option is credit cards.



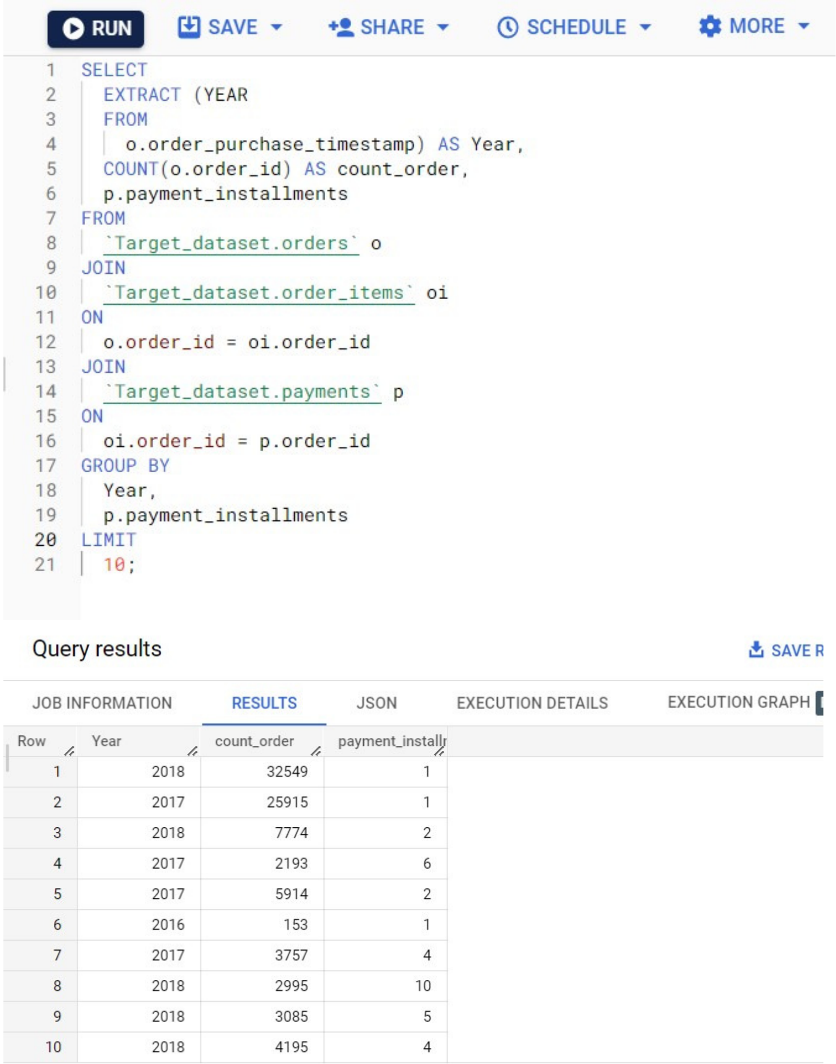
```
SELECT
  count(o.order_id) total_orders,
  payment_type,
  EXTRACT (month
  FROM
    order_purchase_timestamp) AS month_of_order
    FROM
  `target-business-case-379707.Target_dataset.payments` p
JOIN
  `Target_dataset.orders` o
USING
  (order_id)
GROUP BY
  payment_type,
  month_of_order
  order by total_orders desc
LIMIT
  10;
```



Row	total_orders	payment_type	month_of_order
1	8350	credit_card	5
2	8269	credit_card	8
3	7841	credit_card	7
4	7707	credit_card	3
5	7301	credit_card	4
6	7276	credit_card	6
7	6609	credit_card	2
8	6103	credit_card	1
9	5897	credit_card	11
10	4378	credit_card	12

6.2 Count of orders based on the no. of payment installments

The count of orders based on the number of payment installments are provided below. The query hasn't been ordered to show the nature of the data for the limit of 10 rows. Orders, order_items and payments have been joined and data has been grouped by year and payment_installments. From the initial query it seems that payment installments of 1 number seems to be a preferred option in 2018 and 2017, though ordering the query by payment_installments desc might show different results.



```
SELECT
  EXTRACT (YEAR
  FROM
    o.order_purchase_timestamp) AS Year,
  COUNT(o.order_id) AS count_order,
  p.payment_installments
FROM
  `Target_dataset.orders` o
JOIN
  `Target_dataset.order_items` oi
ON
  o.order_id = oi.order_id
JOIN
  `Target_dataset.payments` p
ON
  oi.order_id = p.order_id
GROUP BY
  Year,
  p.payment_installments
LIMIT
  10;
```

Row	Year	count_order	payment_install
1	2018	32549	1
2	2017	25915	1
3	2018	7774	2
4	2017	2193	6
5	2017	5914	2
6	2016	153	1
7	2017	3757	4
8	2018	2995	10
9	2018	3085	5
10	2018	4195	4

7. Actionable insights

- From the increasing number of orders from 2016 to 2017, there is a necessity to recruit more employees to take care of the ecommerce platform. We may need the data for the departments to check the recruitment pattern e.g., whether people have been employed to fulfil the growing sales and if yes, in which departments.
- The number of orders has increased steadily from 2017 to 2018. However, in 2018, the number of orders has been similar in most of the month, ranging from 6000 to 7000. We have to find the cause behind this stagnation. Is there lack of customer support? Is there any issue with the product? Is there price-product parity? Order reviews table can be analysed to see whether customer retention is being done. Price against product tables can be derived as well.
- The time-of-day data shows that Brazilians tend to buy more in the afternoons followed by evenings. But this analysis is definitely skewed, as Brazil has 4 time zones separated by 1 hour. The states and cities need to divided by time zones and then analysis needs to be done. Based

on that we can decide the instruments to improve ecommerce sales in the times that have less traffic to the website / ecommerce platform.

- Some states like MS, MA have very less sales. We have to diagnose the exact issues with these states. Economic conditions, lack of awareness and adoption of technology might be some of the issues. Proper studies need to be undertaken if sales can be increased in tandem with ROI or not. If not, it might be a better option to concentrate more towards the states like RJ and MG that have medium sales and might have the potential to provide more sales.
- One of the main issues with the ecommerce system seems that there is a big difference between estimated delivery date and actual delivery date of the order. This may result in bad customer experience. Also, null values in customer delivery date might indicate lack of follow up for reviews from the customer. This data is extremely important to understand customer experience and ensure customer retention.

8. Recommendations

- Analyse department data to check which department needs recruitment. Also, there is a need to check why there was such a huge spike in order from 2016 to 2017 and then steady growth in 2018.
- Diagnose possible issues with customer support to ensure a seamless experience for customers. Check product and price parity and if there is a requirement for market research for the Brazilian market. The market might seem quite diverse in terms of economic quality of life and social acceptance.
- Most of the customers chose to pay for the product in one go or two installments. It might be the case that there are customers who not opting for the product because of lack of awareness that they can pay with 12 installments. This awareness needs to generated with the help of ads.
- Customer reviews need to be collected. Proactive follow-up by emails and app notifications to review the product and the delivery is imperative, because null values in customer delivery date indicates lack of follow-up.
- Time of day analysis needs to be done for different time zones, because Brazil has four time zones. Brazilian people have late dinners and stay up late. Therefore, there might be an opportunity to increase sales in evening and night time.