Target SQL Case Study – Mohammed Osama Shiraz

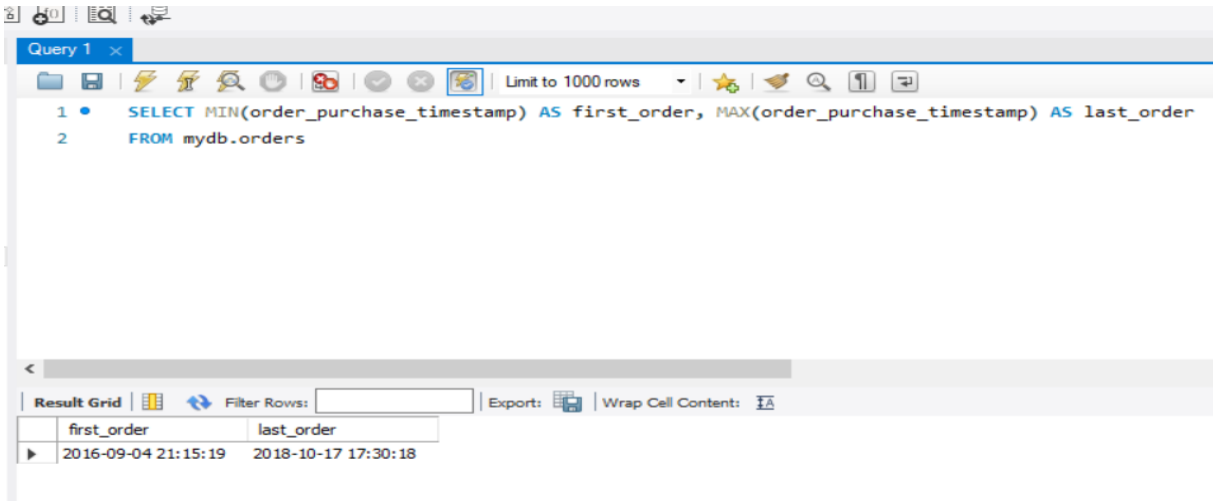
1. Data Explorations

After uploading the CSV datasets into Google Cloud, the customer dataset has been initially analysed.

- Customers Table: There are around 99,441 rows representing each customer who has ordered. The table contains city, state, zip code which has alphanumeric values, integers and string data
- Geolocations Table: It contains around 19,015 unique zip codes; the data types are floating point for latitude and longitude. The name of city and state is also provided as string.
- Order Reviews: Around 99,224 reviews are there in the table whose primary key is review_id. Foreign key is order_id. Each review has a review score from 1-5. And review date having datetime data type.
- Products: There are around 32,951 products, the fields describe the dimensions of the product in integers.
- Order_items: This table shows for each order how many items have been ordered with the corresponding seller_id, order_id, product_id as the foreign keys.
- Payments: In this table we have for each order, the payment type(string), price(float), and number of instalments(int), with around 103886 rows.
- Sellers: We have the information about the sellers. Seller_id(string), zip code(int), city and state(string).
- Orders: We have 99,441 orders with information regarding orders like – order_status(int), customer_id (string), date and time of purchase and delivery.

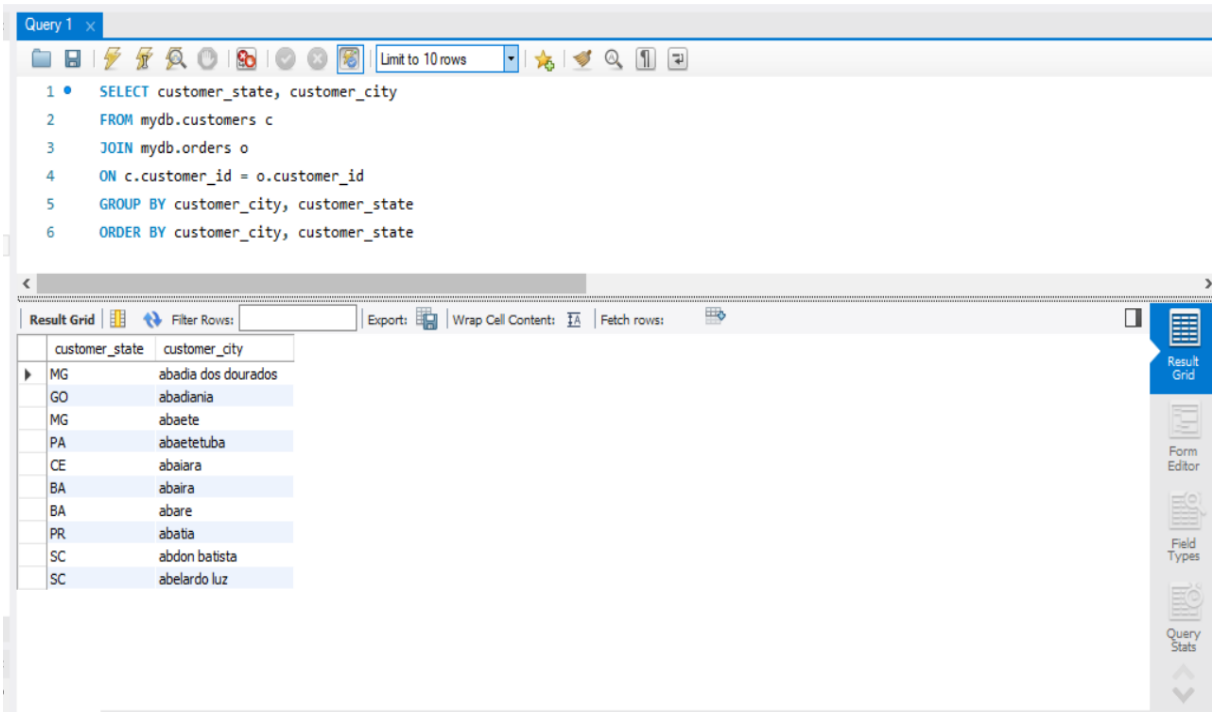
1.1 Time Range

We can see from the orders table that the date range is from 2016-09-04 21:15:19 UTC to 2018-10-17 17:30:18 UTC. around two years.



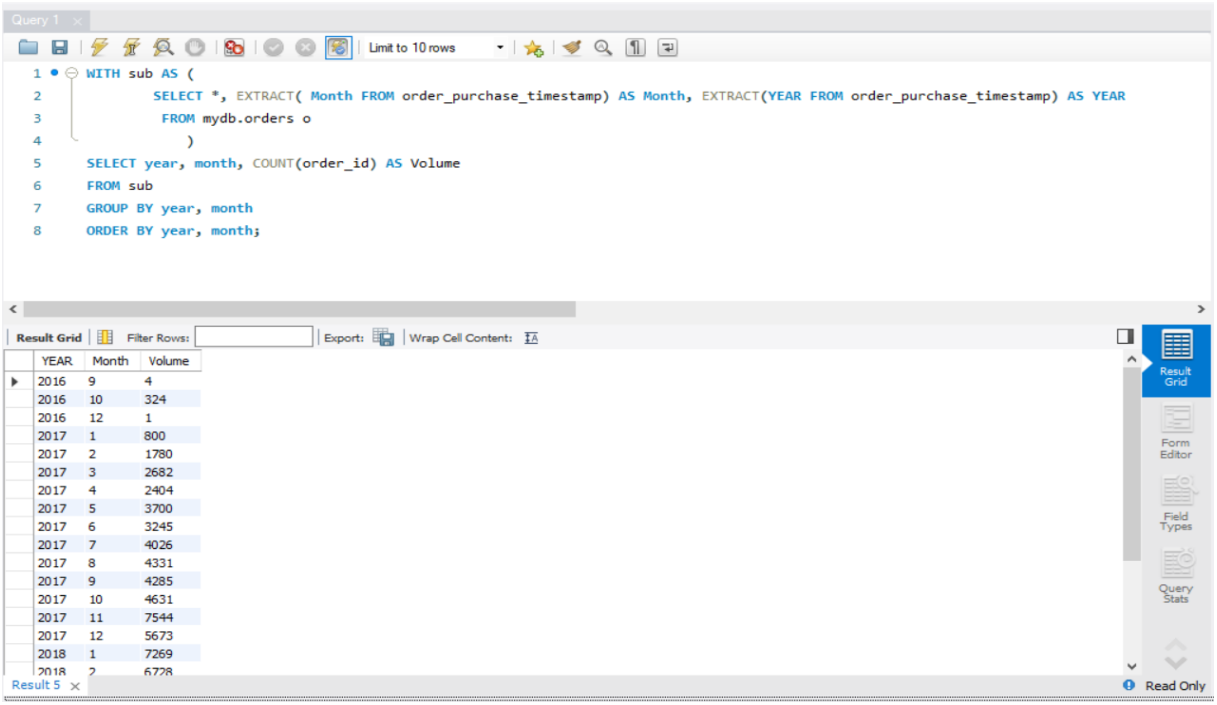
1.2 Cities & States of customers

There are around 4310 cities from where the customers ordered.



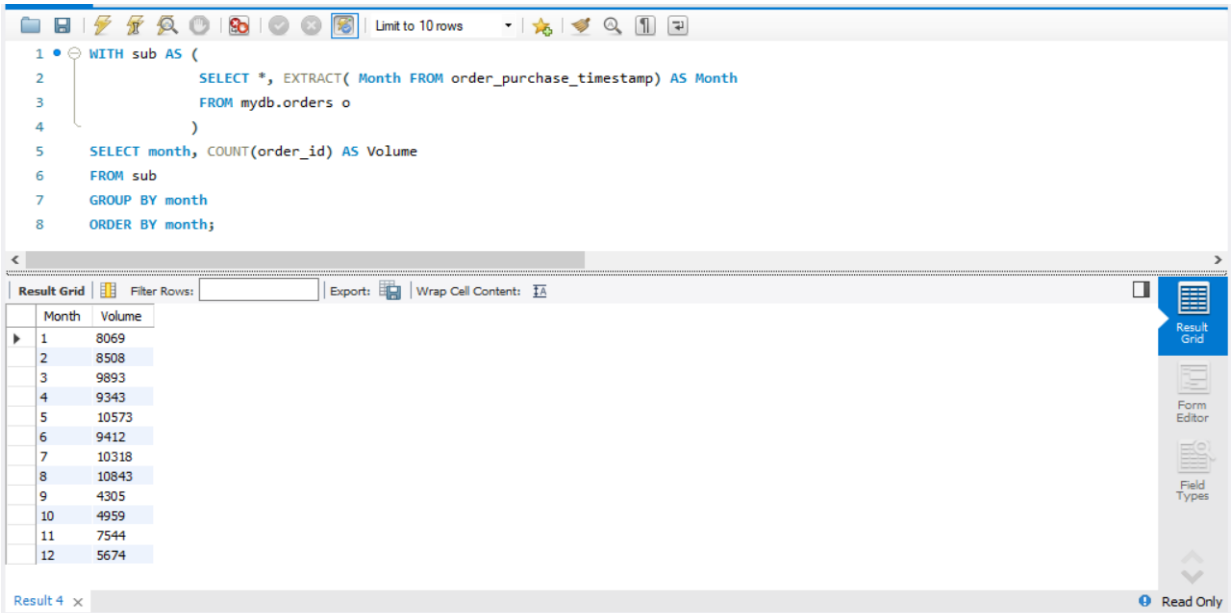
2.1 Trend in the no. of orders

We can observe that there were originally less orders in 2016 in terms of volume (orders). But starting in 2017, the volume of orders grew quickly. Sales reach their high in November 2017 and then decline in 2018.



2.2 Monthly seasonality in terms of the no. of orders

Further, we see that sales peak in the mid-year period during the months of May, July and August.



```
WITH sub AS (
             SELECT *, EXTRACT( Month FROM order_purchase_timestamp) AS Month
             FROM mydb.orders o
            )
SELECT month, COUNT(order_id) AS Volume
FROM sub
GROUP BY month
ORDER BY month;
```

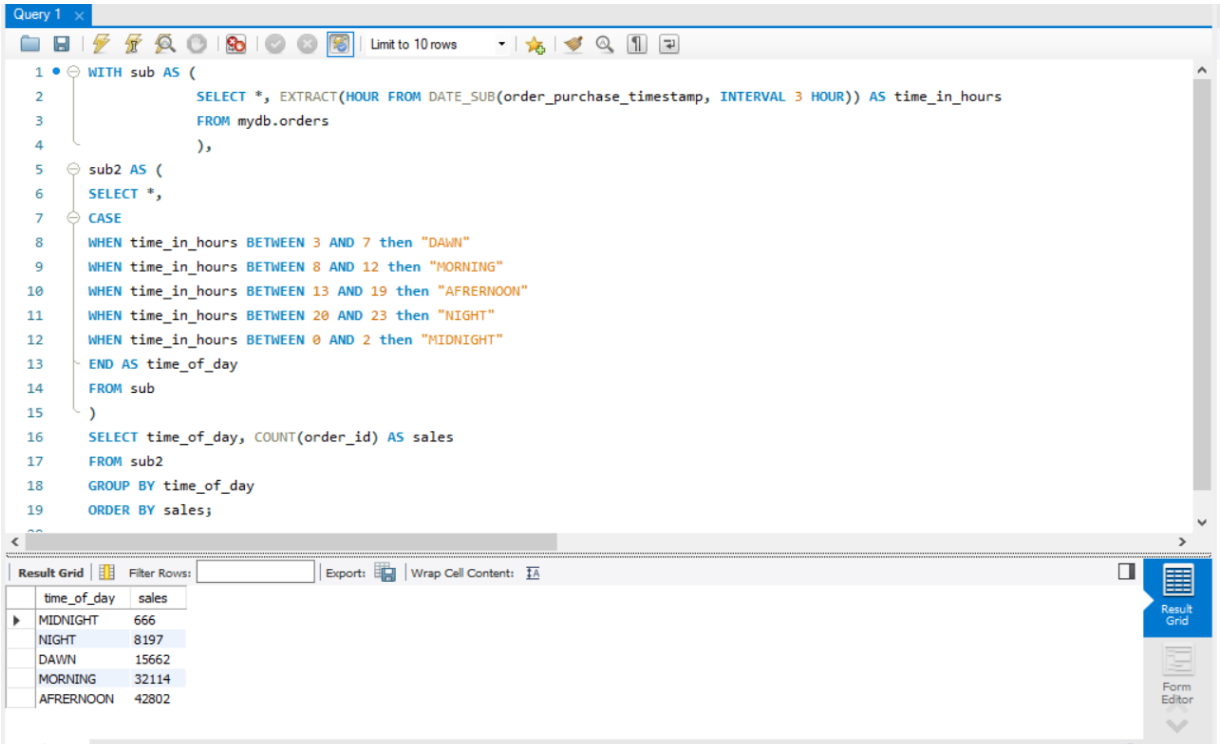
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Month	Volume
1	8069
2	8508
3	9893
4	9343
5	10573
6	9412
7	10318
8	10843
9	4305
10	4959
11	7544
12	5674

Result 4 | Read Only

2.3 Time at which Brazilian customers tend to orders

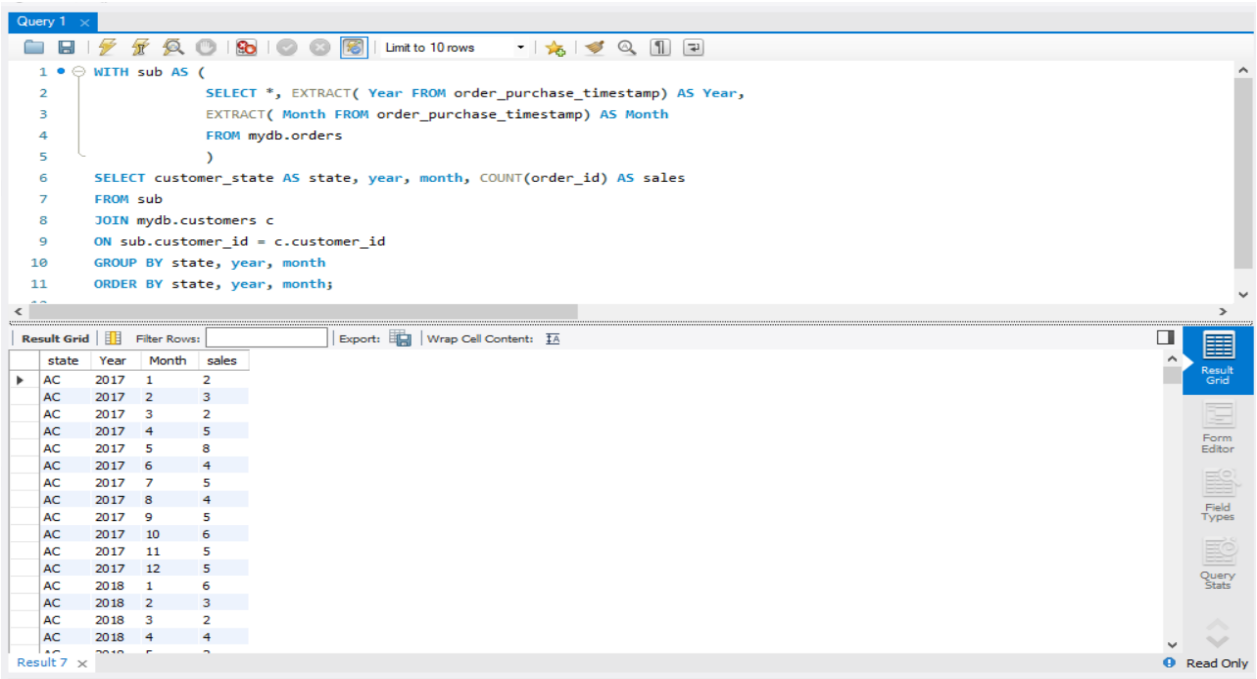
We may categorise the time into dawn, morning, afternoon, and night using the case and when statements to get the time from the order_purchase_timestamp. We can calculate the number of orders, which is then recorded as sales, by grouping by time_of_day. This leads us to the conclusion that Brazilian clients tend to buy more in the afternoon.



```
WITH sub AS (
            SELECT *, EXTRACT(HOUR FROM DATE_SUB(order_purchase_timestamp, INTERVAL 3 HOUR)) AS time_in_hours
            FROM mydb.orders
            ),
sub2 AS (
SELECT *,
CASE
WHEN time_in_hours BETWEEN 3 AND 7 then "DAWN"
WHEN time_in_hours BETWEEN 8 AND 12 then "MORNING"
WHEN time_in_hours BETWEEN 13 AND 19 then "AFRERNOON"
WHEN time_in_hours BETWEEN 20 AND 23 then "NIGHT"
WHEN time_in_hours BETWEEN 0 AND 2 then "MIDNIGHT"
END AS time_of_day
FROM sub
)
SELECT time_of_day, COUNT(order_id) AS sales
FROM sub2
GROUP BY time_of_day
ORDER BY sales;
```

time_of_day	sales
MIDNIGHT	666
NIGHT	8197
DAWN	15662
MORNING	32114
AFRERNOON	42802

3.1 Monthly orders by Brazil States

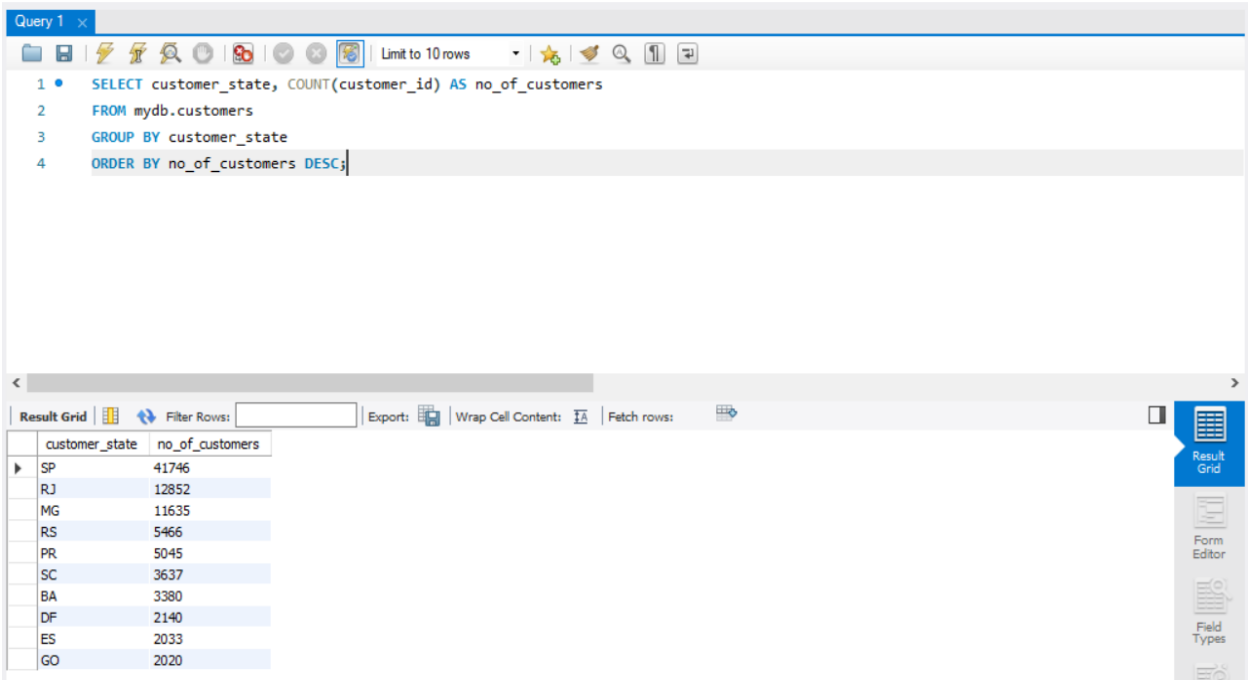


```
WITH sub AS (
            SELECT *, EXTRACT( Year FROM order_purchase_timestamp) AS Year,
            EXTRACT( Month FROM order_purchase_timestamp) AS Month
            FROM mydb.orders
            )
SELECT customer_state AS state, year, month, COUNT(order_id) AS sales
FROM sub
JOIN mydb.customers c
ON sub.customer_id = c.customer_id
GROUP BY state, year, month
ORDER BY state, year, month;
```

state	Year	Month	sales
AC	2017	1	2
AC	2017	2	3
AC	2017	3	2
AC	2017	4	5
AC	2017	5	8
AC	2017	6	4
AC	2017	7	5
AC	2017	8	4
AC	2017	9	5
AC	2017	10	6
AC	2017	11	5
AC	2017	12	5
AC	2018	1	6
AC	2018	2	3
AC	2018	3	2
AC	2018	4	4

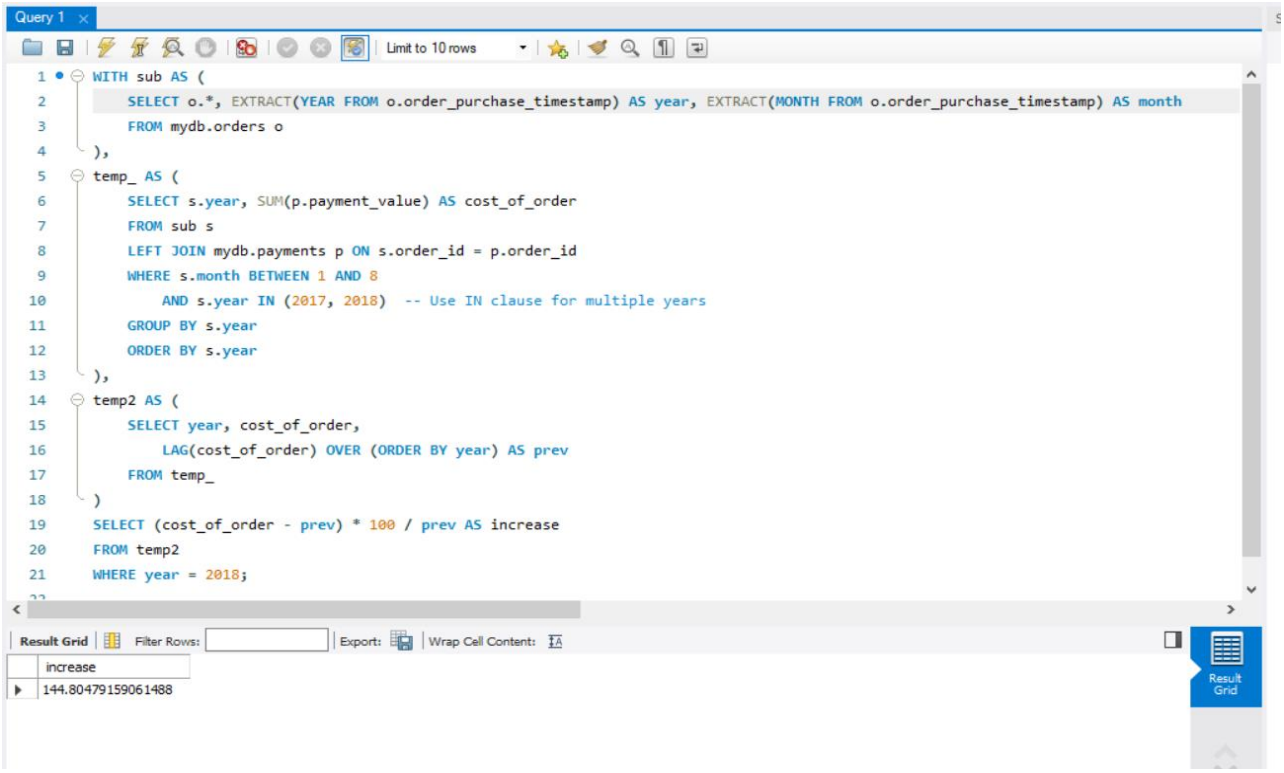
3.2 Customer Distribution of Brazil States

To determine the distribution of customers across different states, you group the "states" column in the "customers" table and tally the count of unique customer IDs in each state group.



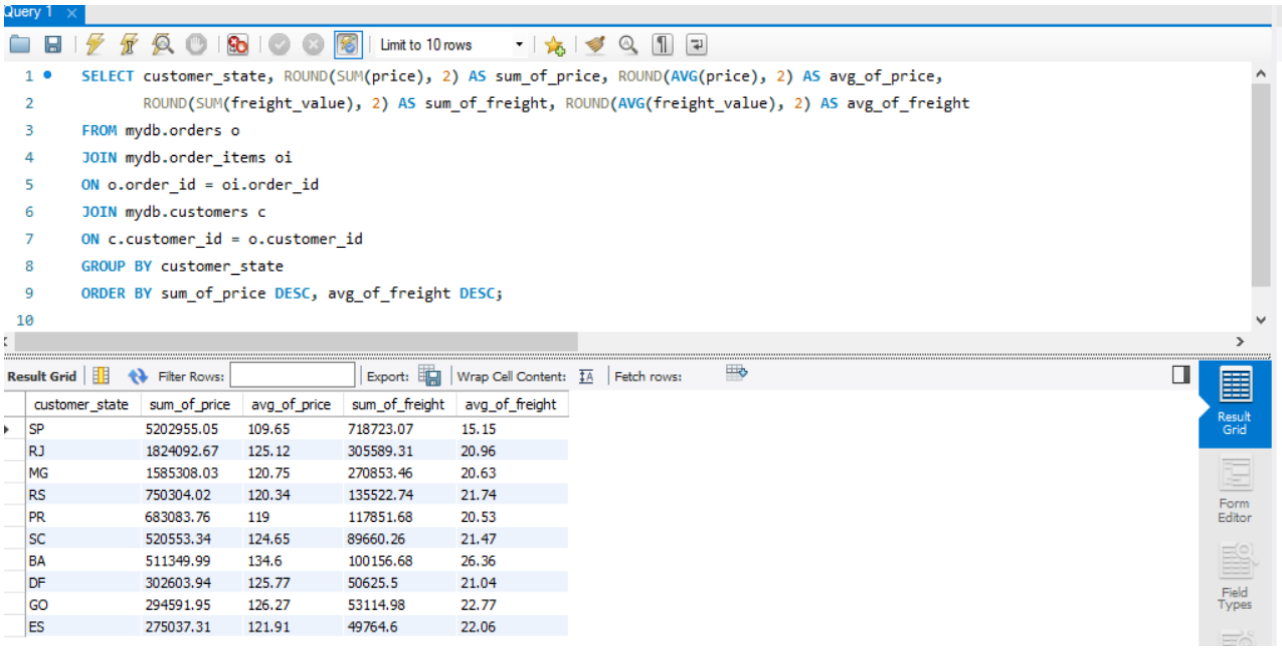
4.1 Percentage increase in the cost of orders

The percentage increase in order costs is computed by joining payment and order data. We consider orders from January to August, group by 2017 and 2018, calculate the cost difference using the lag function, resulting in a 136.97% increase.



4.2 Total and Average value of freight and price.

Sao Paulo state stands out with the highest total price and total freight costs. Notably, Sao Paulo also has the lowest average price and average freight expenses. To boost sales, Target could consider reducing freight charges across all regions, thereby lowering the overall cost to customers.



```
SELECT customer_state, ROUND(SUM(price), 2) AS sum_of_price, ROUND(AVG(price), 2) AS avg_of_price,
       ROUND(SUM(freight_value), 2) AS sum_of_freight, ROUND(AVG(freight_value), 2) AS avg_of_freight
FROM mydb.orders o
JOIN mydb.order_items oi
ON o.order_id = oi.order_id
JOIN mydb.customers c
ON c.customer_id = o.customer_id
GROUP BY customer_state
ORDER BY sum_of_price DESC, avg_of_freight DESC;
```

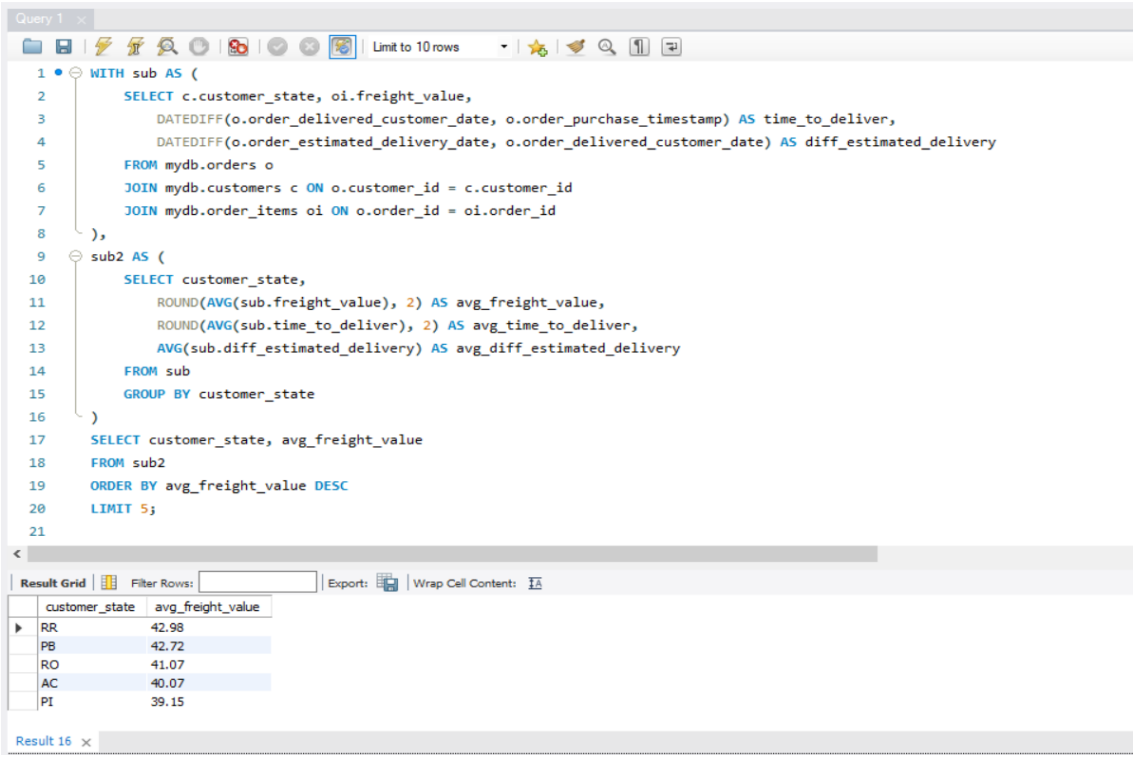
Result Grid

customer_state	sum_of_price	avg_of_price	sum_of_freight	avg_of_freight
SP	5202955.05	109.65	718723.07	15.15
RJ	1824092.67	125.12	305589.31	20.96
MG	1585308.03	120.75	270853.46	20.63
RS	750304.02	120.34	135522.74	21.74
PR	683083.76	119	117851.68	20.53
SC	520553.34	124.65	89660.26	21.47
BA	511349.99	134.6	100156.68	26.36
DF	302603.94	125.77	50625.5	21.04
GO	294591.95	126.27	53114.98	22.77
ES	275037.31	121.91	49764.6	22.06

5.1 Analysis based on sales, freight and delivery time

Target should enhance inventory management and logistics in states with high average freight costs to reduce expenses. Additionally, they should focus on improving delivery times in these states. Notably, states with elevated freight costs also experience longer delivery times. Target should also prioritize refining its delivery estimation algorithm, as it currently deviates significantly, by approximately 40 days, from the actual delivery times in the top 5 states with the fastest deliveries compared to estimated delivery times.

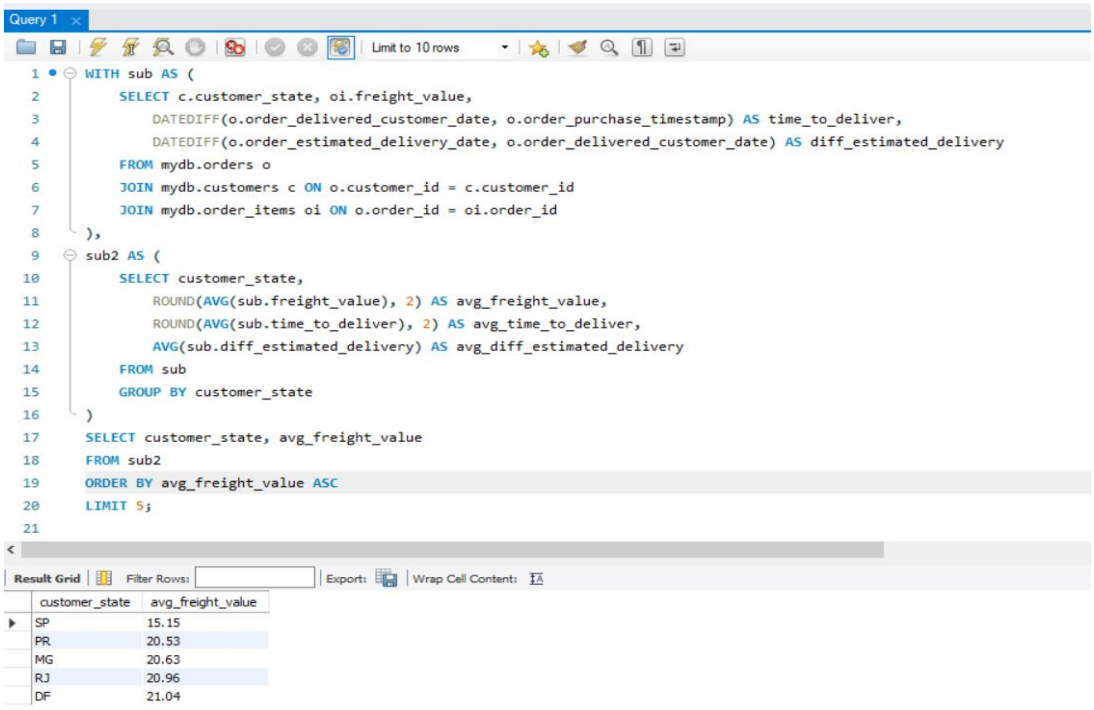
1. States with the highest average freight cost.



```
WITH sub AS (
    SELECT c.customer_state, oi.freight_value,
        DATEDIFF(o.order_delivered_customer_date, o.order_purchase_timestamp) AS time_to_deliver,
        DATEDIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date) AS diff_estimated_delivery
    FROM mydb.orders o
    JOIN mydb.customers c ON o.customer_id = c.customer_id
    JOIN mydb.order_items oi ON o.order_id = oi.order_id
),
sub2 AS (
    SELECT customer_state,
        ROUND(AVG(sub.freight_value), 2) AS avg_freight_value,
        ROUND(AVG(sub.time_to_deliver), 2) AS avg_time_to_deliver,
        AVG(sub.diff_estimated_delivery) AS avg_diff_estimated_delivery
    FROM sub
    GROUP BY customer_state
)
SELECT customer_state, avg_freight_value
FROM sub2
ORDER BY avg_freight_value DESC
LIMIT 5;
```

customer_state	avg_freight_value
RR	42.98
PB	42.72
RO	41.07
AC	40.07
PI	39.15

2. States with the LOWEST average freight cost.



```
WITH sub AS (
    SELECT c.customer_state, oi.freight_value,
        DATEDIFF(o.order_delivered_customer_date, o.order_purchase_timestamp) AS time_to_deliver,
        DATEDIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date) AS diff_estimated_delivery
    FROM mydb.orders o
    JOIN mydb.customers c ON o.customer_id = c.customer_id
    JOIN mydb.order_items oi ON o.order_id = oi.order_id
),
sub2 AS (
    SELECT customer_state,
        ROUND(AVG(sub.freight_value), 2) AS avg_freight_value,
        ROUND(AVG(sub.time_to_deliver), 2) AS avg_time_to_deliver,
        AVG(sub.diff_estimated_delivery) AS avg_diff_estimated_delivery
    FROM sub
    GROUP BY customer_state
)
SELECT customer_state, avg_freight_value
FROM sub2
ORDER BY avg_freight_value ASC
LIMIT 5;
```

customer_state	avg_freight_value
SP	15.15
PR	20.53
MG	20.63
RJ	20.96
DF	21.04

3. States with the highest average time to delivery.

Query 1

```
WITH sub AS (
    SELECT c.customer_state, oi.freight_value,
        DATEDIFF(o.order_delivered_customer_date, o.order_purchase_timestamp) AS time_to_deliver,
        DATEDIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date) AS diff_estimated_delivery
    FROM mydb.orders o
    JOIN mydb.customers c ON o.customer_id = c.customer_id
    JOIN mydb.order_items oi ON o.order_id = oi.order_id
),
sub2 AS (
    SELECT customer_state,
        ROUND(AVG(sub.freight_value), 2) AS avg_freight_value,
        ROUND(AVG(sub.time_to_deliver), 2) AS avg_time_to_deliver,
        AVG(sub.diff_estimated_delivery) AS avg_diff_estimated_delivery
    FROM sub
    GROUP BY customer_state
)
SELECT customer_state, avg_time_to_deliver
FROM sub2
ORDER BY avg_time_to_deliver DESC
LIMIT 5;
```

Result Grid

customer_state	avg_time_to_deliver
AP	28.22
RR	28.17
AM	26.34
AL	24.45
PA	23.70

4. States with the lowest average time to delivery.

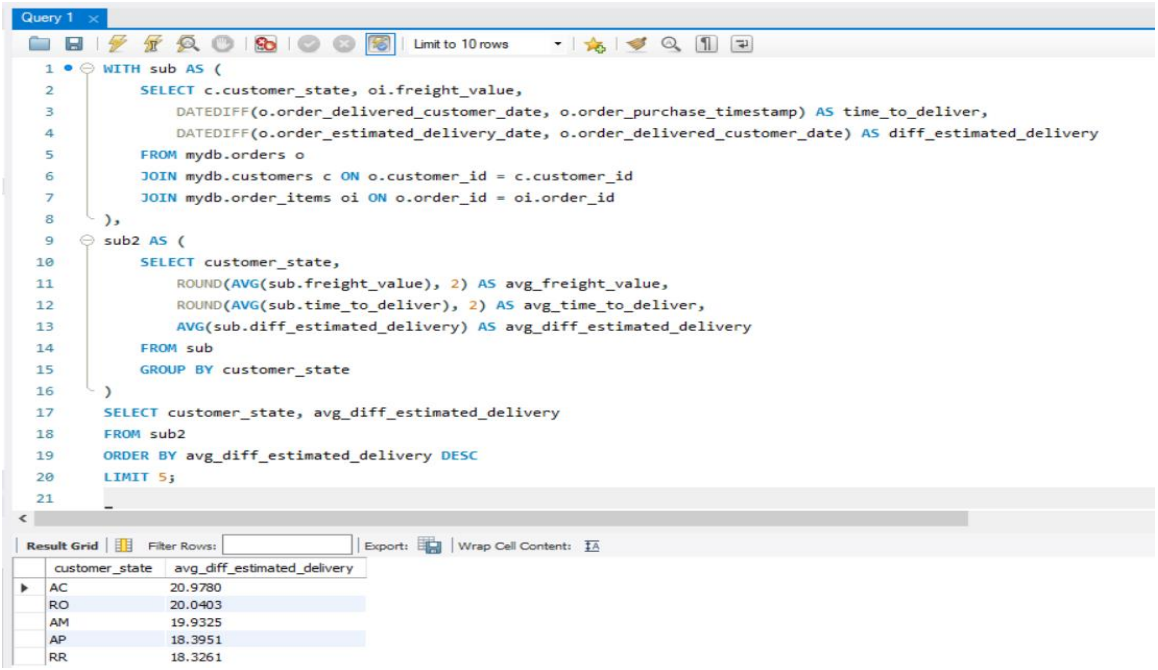
Query 1

```
WITH sub AS (
    SELECT c.customer_state, oi.freight_value,
        DATEDIFF(o.order_delivered_customer_date, o.order_purchase_timestamp) AS time_to_deliver,
        DATEDIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date) AS diff_estimated_delivery
    FROM mydb.orders o
    JOIN mydb.customers c ON o.customer_id = c.customer_id
    JOIN mydb.order_items oi ON o.order_id = oi.order_id
),
sub2 AS (
    SELECT customer_state,
        ROUND(AVG(sub.freight_value), 2) AS avg_freight_value,
        ROUND(AVG(sub.time_to_deliver), 2) AS avg_time_to_deliver,
        AVG(sub.diff_estimated_delivery) AS avg_diff_estimated_delivery
    FROM sub
    GROUP BY customer_state
)
SELECT customer_state, avg_time_to_deliver
FROM sub2
ORDER BY avg_time_to_deliver ASC
LIMIT 5;
```

Result Grid

customer_state	avg_time_to_deliver
SP	8.66
PR	11.89
MG	11.92
DF	12.89
SC	14.95

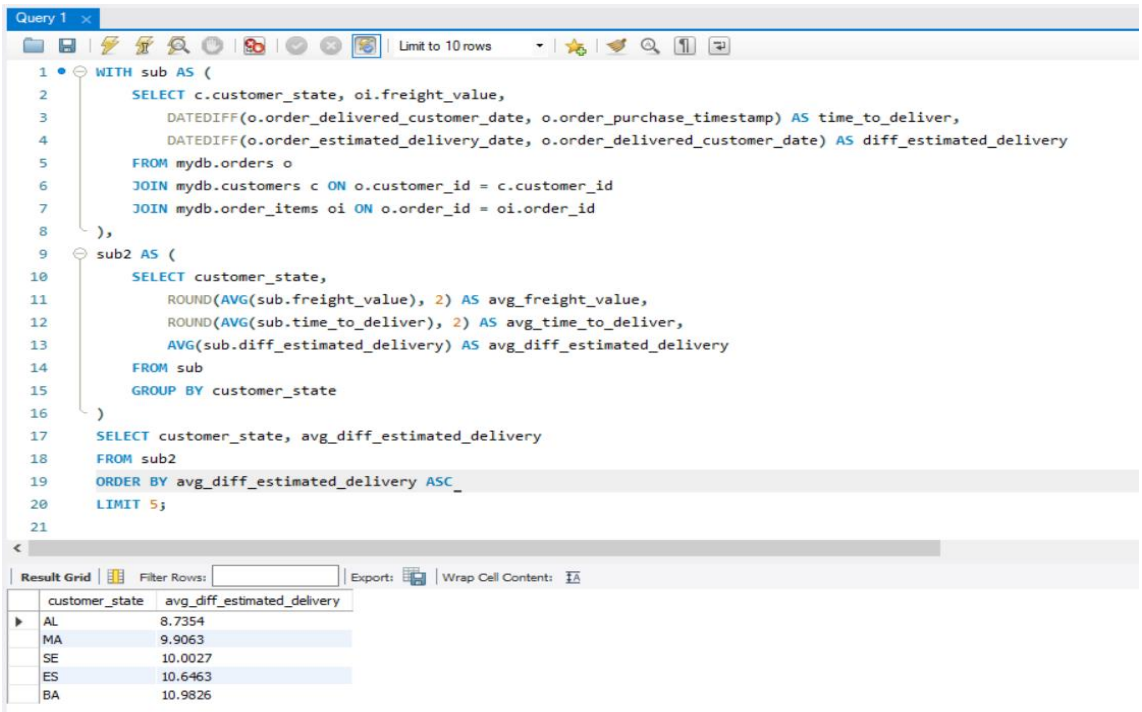
5. States where delivery time is faster than estimated.



```
WITH sub AS (
    SELECT c.customer_state, oi.freight_value,
        DATEDIFF(o.order_delivered_customer_date, o.order_purchase_timestamp) AS time_to_deliver,
        DATEDIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date) AS diff_estimated_delivery
    FROM mydb.orders o
    JOIN mydb.customers c ON o.customer_id = c.customer_id
    JOIN mydb.order_items oi ON o.order_id = oi.order_id
),
sub2 AS (
    SELECT customer_state,
        ROUND(AVG(sub.freight_value), 2) AS avg_freight_value,
        ROUND(AVG(sub.time_to_deliver), 2) AS avg_time_to_deliver,
        AVG(sub.diff_estimated_delivery) AS avg_diff_estimated_delivery
    FROM sub
    GROUP BY customer_state
)
SELECT customer_state, avg_diff_estimated_delivery
FROM sub2
ORDER BY avg_diff_estimated_delivery DESC
LIMIT 5;
```

customer_state	avg_diff_estimated_delivery
AC	20.9780
RO	20.0403
AM	19.9325
AP	18.3951
RR	18.3261

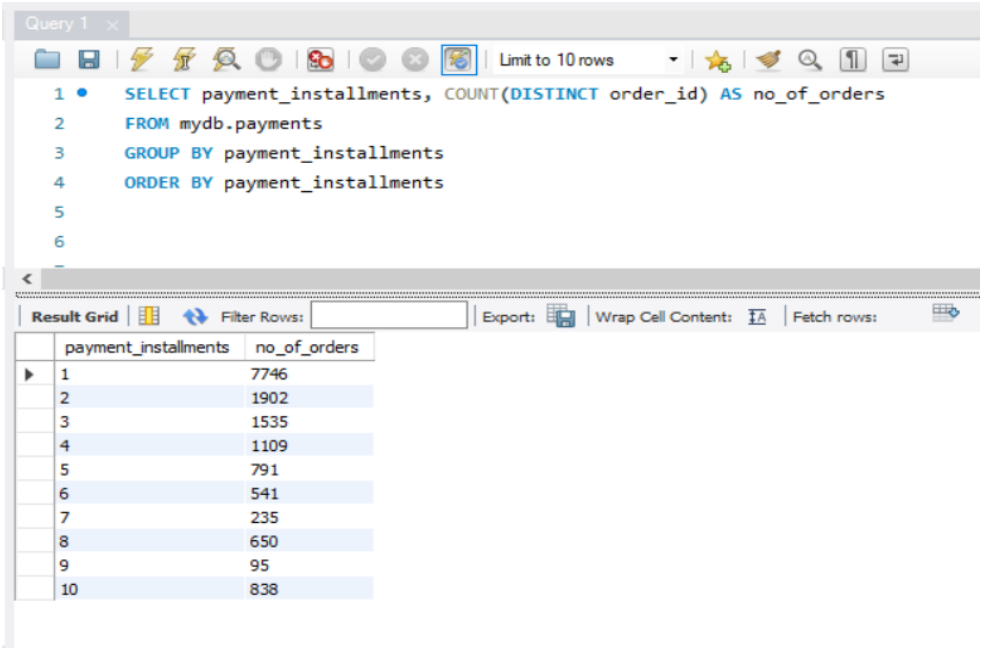
6. States where delivery time is slower than estimated



```
WITH sub AS (
    SELECT c.customer_state, oi.freight_value,
        DATEDIFF(o.order_delivered_customer_date, o.order_purchase_timestamp) AS time_to_deliver,
        DATEDIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date) AS diff_estimated_delivery
    FROM mydb.orders o
    JOIN mydb.customers c ON o.customer_id = c.customer_id
    JOIN mydb.order_items oi ON o.order_id = oi.order_id
),
sub2 AS (
    SELECT customer_state,
        ROUND(AVG(sub.freight_value), 2) AS avg_freight_value,
        ROUND(AVG(sub.time_to_deliver), 2) AS avg_time_to_deliver,
        AVG(sub.diff_estimated_delivery) AS avg_diff_estimated_delivery
    FROM sub
    GROUP BY customer_state
)
SELECT customer_state, avg_diff_estimated_delivery
FROM sub2
ORDER BY avg_diff_estimated_delivery ASC
LIMIT 5;
```

customer_state	avg_diff_estimated_delivery
AL	8.7354
MA	9.9063
SE	10.0027
ES	10.6463
BA	10.9826

6. Analysis of payment types - Order count dependent on number of installments



Recommdations

1. A suggestion could be to advise Target to offer increased discounts and perks to customers who use debit cards since this payment method is currently the least utilized.
2. Enhancements are needed in inventory management and logistics for specific states mentioned earlier. These improvements aim to reduce delivery times and ensure that estimated delivery schedules better match the actual delivery times, preventing customers from receiving orders unexpectedly.
3. Improved logistics will also decrease the average freight costs.
4. To enhance the affordability of larger purchases, Target should consider offering zero-cost EMI options and discounts for transactions with 12 or more installments, as products with higher installment counts tend to have fewer orders due to their typically higher price tags.
5. Since monthly sales are consistently on the rise, it is advisable for Target to concentrate on expanding its presence in the Brazilian market to avoid any potential shortages or delays.
6. The state of Sao Paulo (SP) holds significant importance for Target, as it constitutes the primary source of most orders. Target should prioritize efforts to decrease delivery times within this region.