Case study :

Target (SQL)

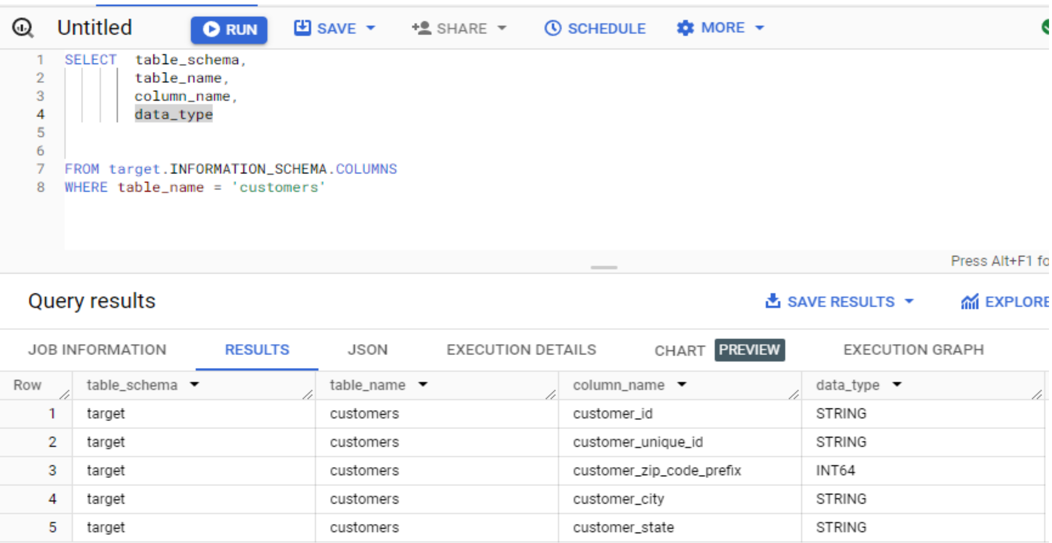
By

Vishnu Vineeth PM

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.

The query to find the data type of the table is given below:



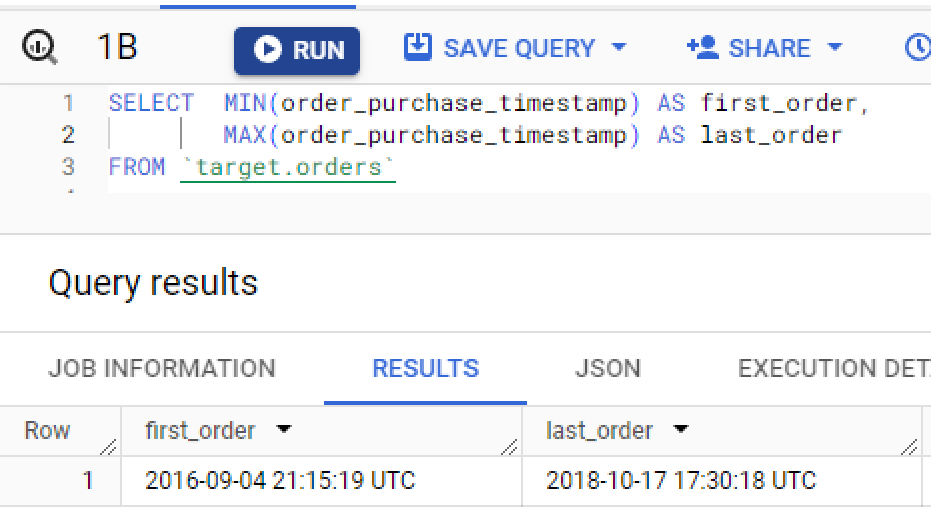
```sql
SELECT  table_schema,
        table_name,
        column_name,
        data_type


FROM target.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers'
```

Row	table_schema	table_name	column_name	data_type
1	target	customers	customer_id	STRING
2	target	customers	customer_unique_id	STRING
3	target	customers	customer_zip_code_prefix	INT64
4	target	customers	customer_city	STRING
5	target	customers	customer_state	STRING

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

The time range of the dataset is from the first day of the order to the last day of it.

The query and the result is :

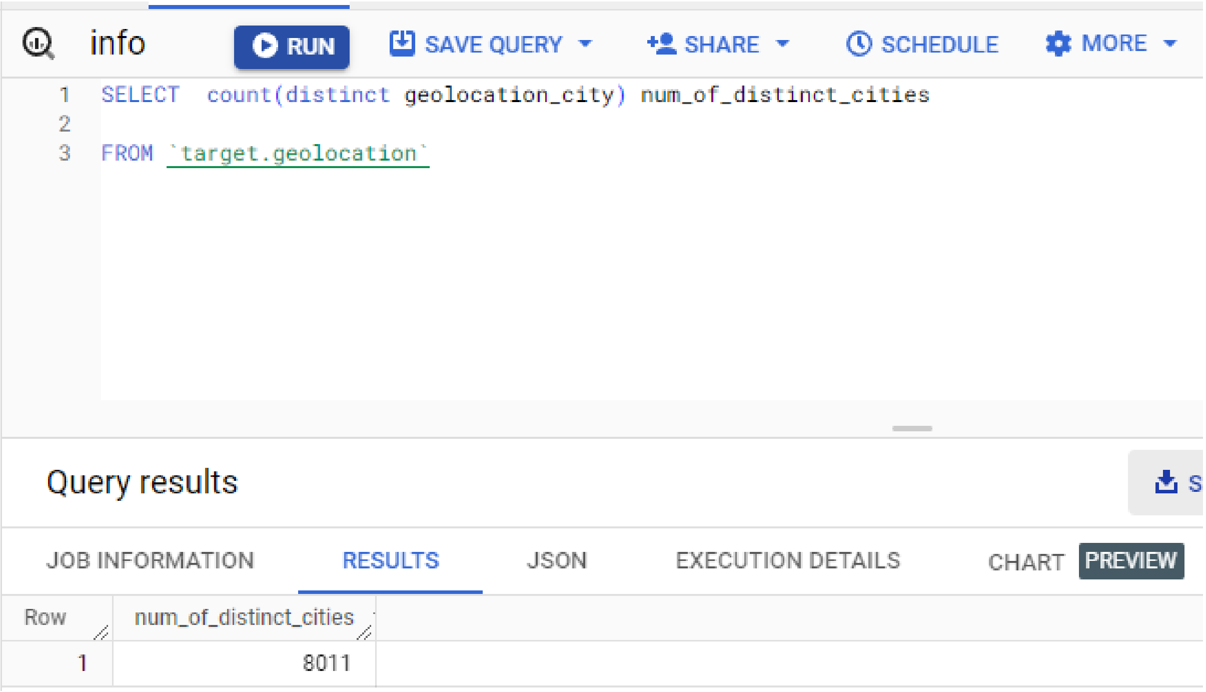


```sql
SELECT  MIN(order_purchase_timestamp) AS first_order,
        MAX(order_purchase_timestamp) AS last_order
FROM `target.orders`
```

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

C. Count the number of Cities and States in our dataset :

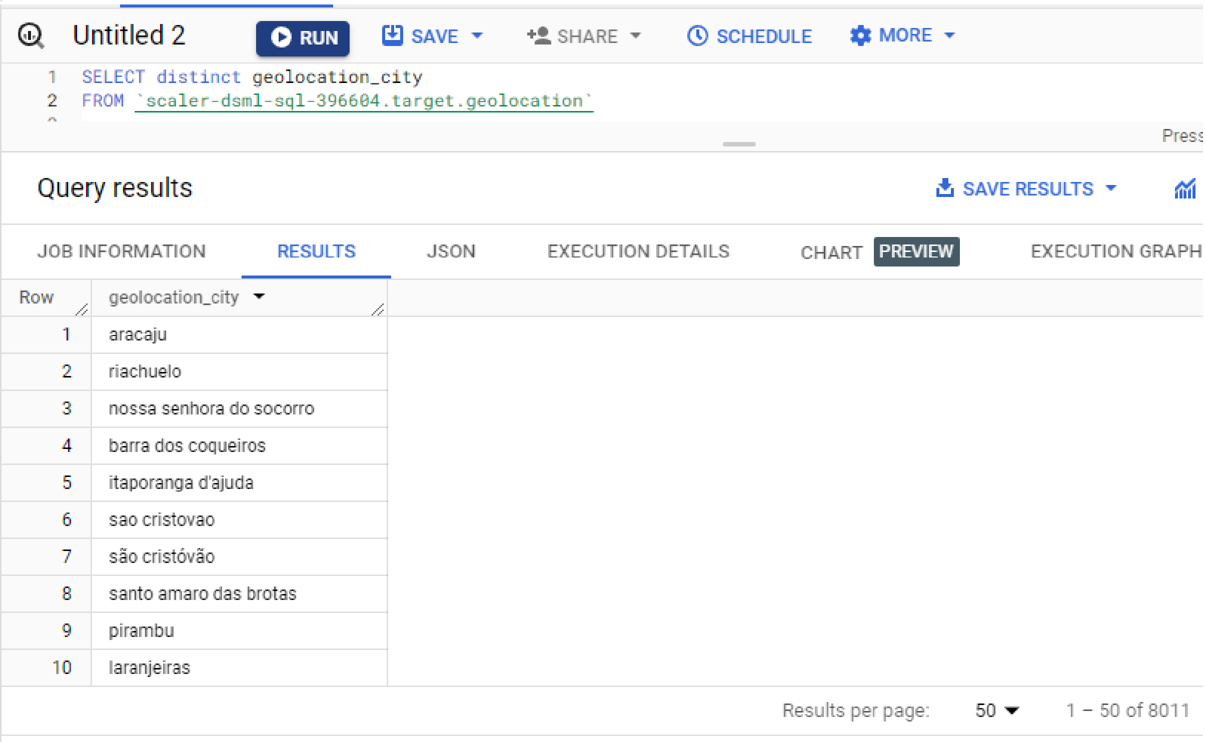
The number of Cities are calculated using the query :



```sql
SELECT  count(distinct geolocation_city) num_of_distinct_cities

FROM `target.geolocation`
```

Query results

Row	num_of_distinct_cities
1	8011



```sql
SELECT distinct geolocation_city
FROM `scaler-dsml-sql-396604.target.geolocation`
```

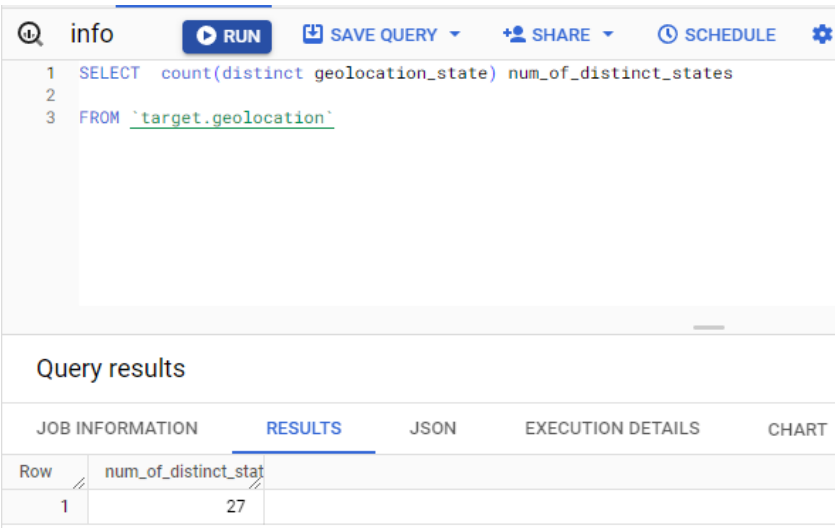
Query results

Row	geolocation_city
1	aracaju
2	riachuelo
3	nossa senhora do socorro
4	barra dos coqueiros
5	itaporanga d'ajuda
6	sao cristovao
7	são cristóvão
8	santo amaro das brotas
9	pirambu
10	laranjeiras

Results per page: 50 1 – 50 of 8011

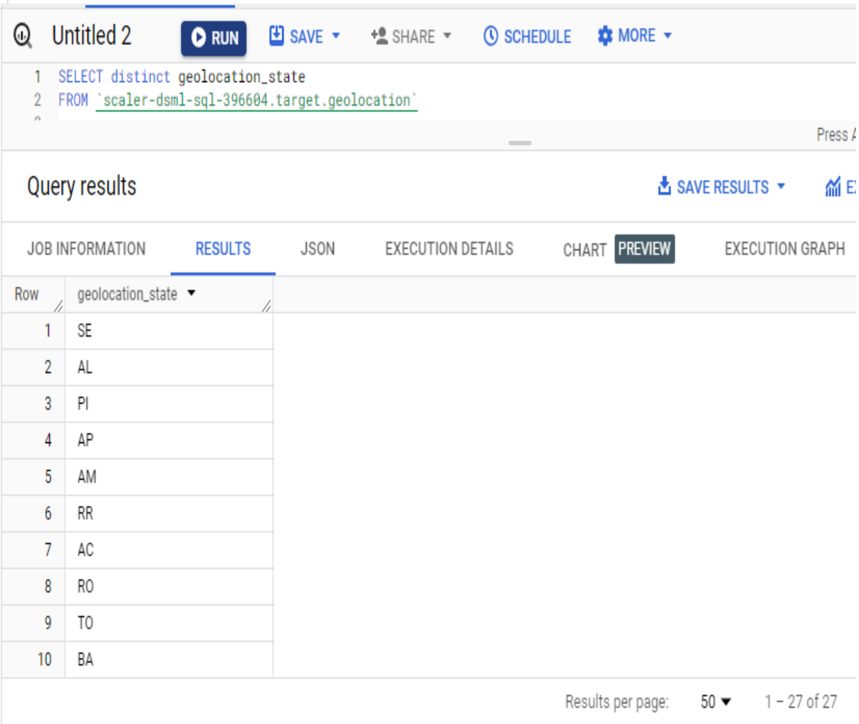
The query gives the result that there are 8011 unique Cities in Brazil .

The number of states are calculated using the query :



```
SELECT  count(distinct geolocation_state) num_of_distinct_states

FROM `target.geolocation`
```

Row	num_of_distinct_stat
1	27



```
SELECT distinct geolocation_state
FROM `scaler-dsml-sql-396604.target.geolocation`
```

Row	geolocation_state
1	SE
2	AL
3	PI
4	AP
5	AM
6	RR
7	AC
8	RO
9	TO
10	BA

Results per page: 50 ▾ 1 – 27 of 27

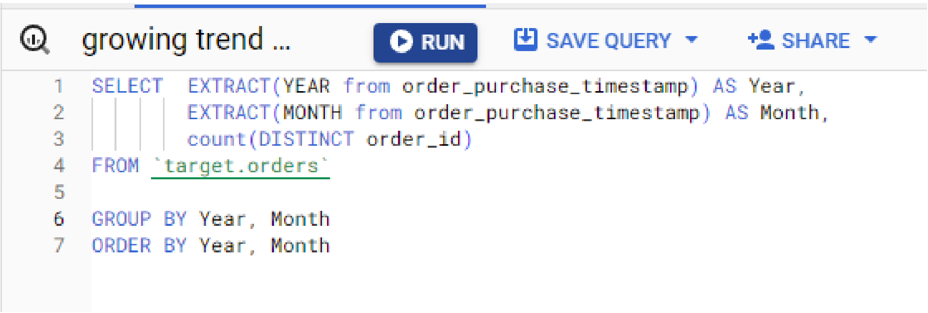
The query gives the result that there are 27 unique States in Brazil.

2) In – depth Explorations :

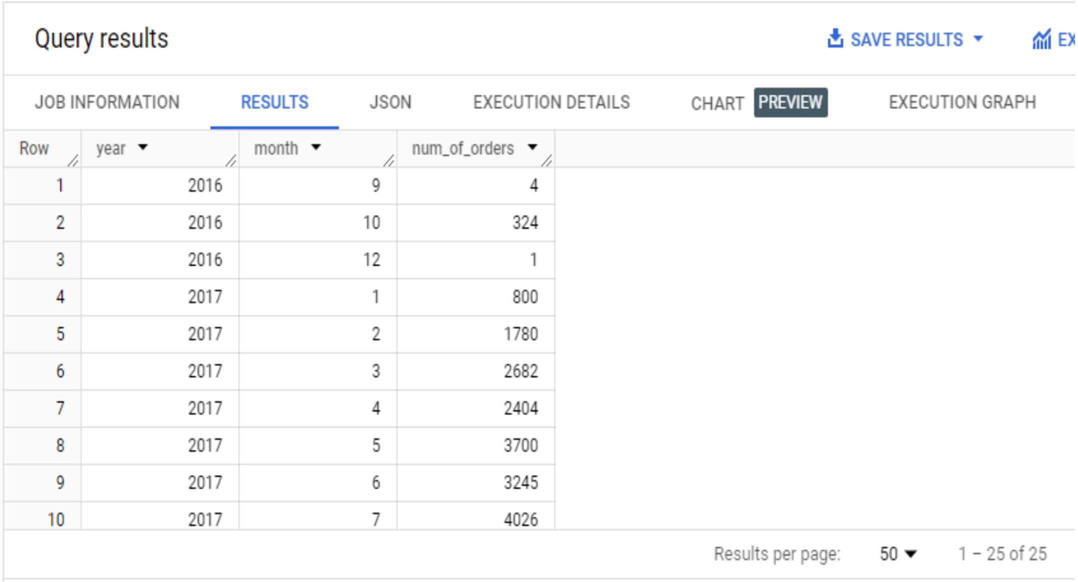
A. Is there a growing trend in the no. of orders placed over the past years?

To find the trend in e-commerce in Brazil, we find the number of orders placed in each month over the past years.

The query and result is given by :



```
SELECT  EXTRACT(YEAR from order_purchase_timestamp) AS Year,
        EXTRACT(MONTH from order_purchase_timestamp) AS Month,
        count(DISTINCT order_id)
FROM `target.orders`

GROUP BY Year, Month
ORDER BY Year, Month
```



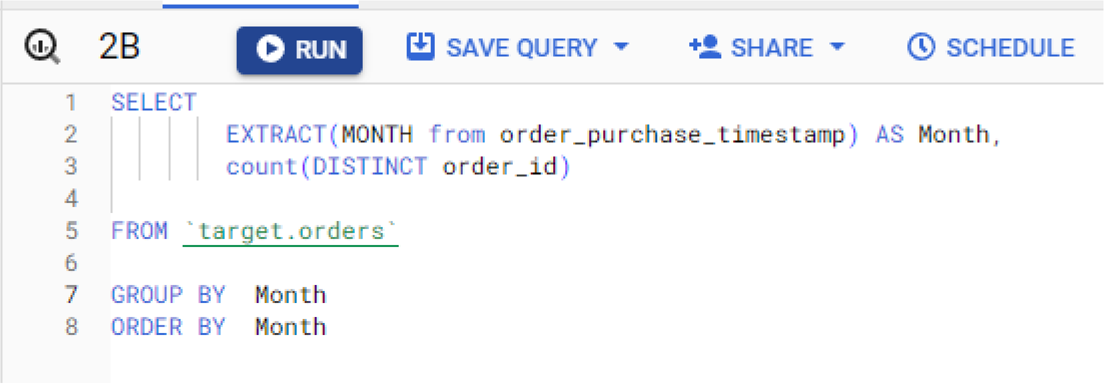
Row	year	month	num_of_orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026

The result shows that there is a growing trend in the e-commerce sector of Brazil.

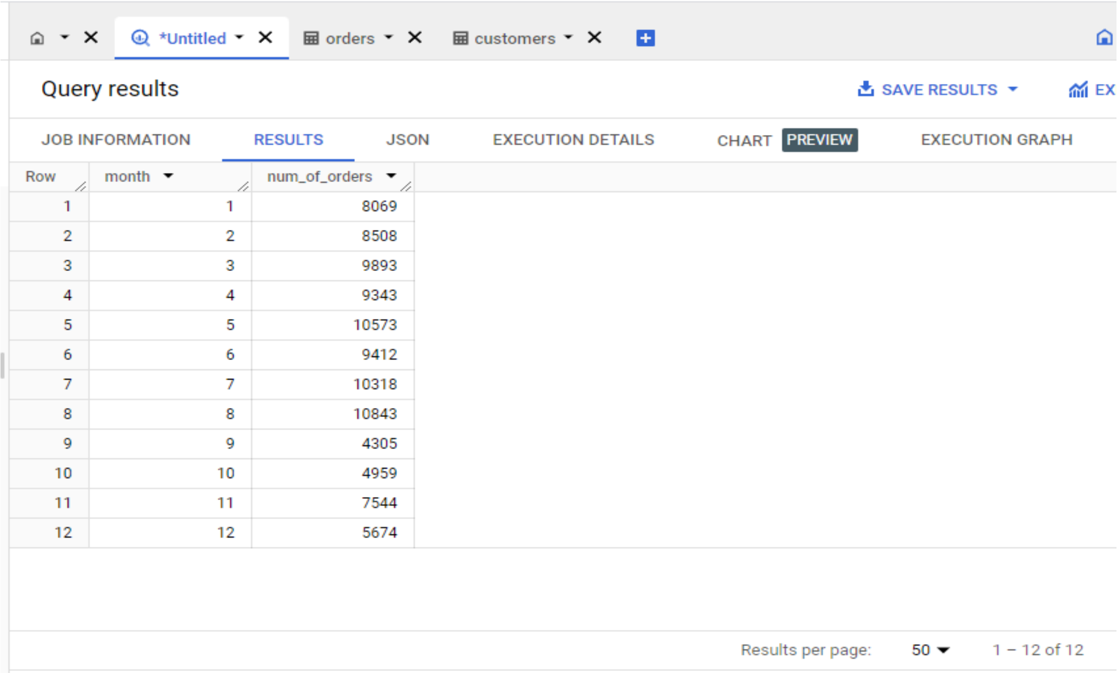
As we can see the number of orders gets increasing over the months.

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

For seasonality we can look at the number of orders placed in each month



```
SELECT
        EXTRACT(MONTH from order_purchase_timestamp) AS Month,
        count(DISTINCT order_id)

FROM `target.orders`

GROUP BY  Month
ORDER BY  Month
```



Query results

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

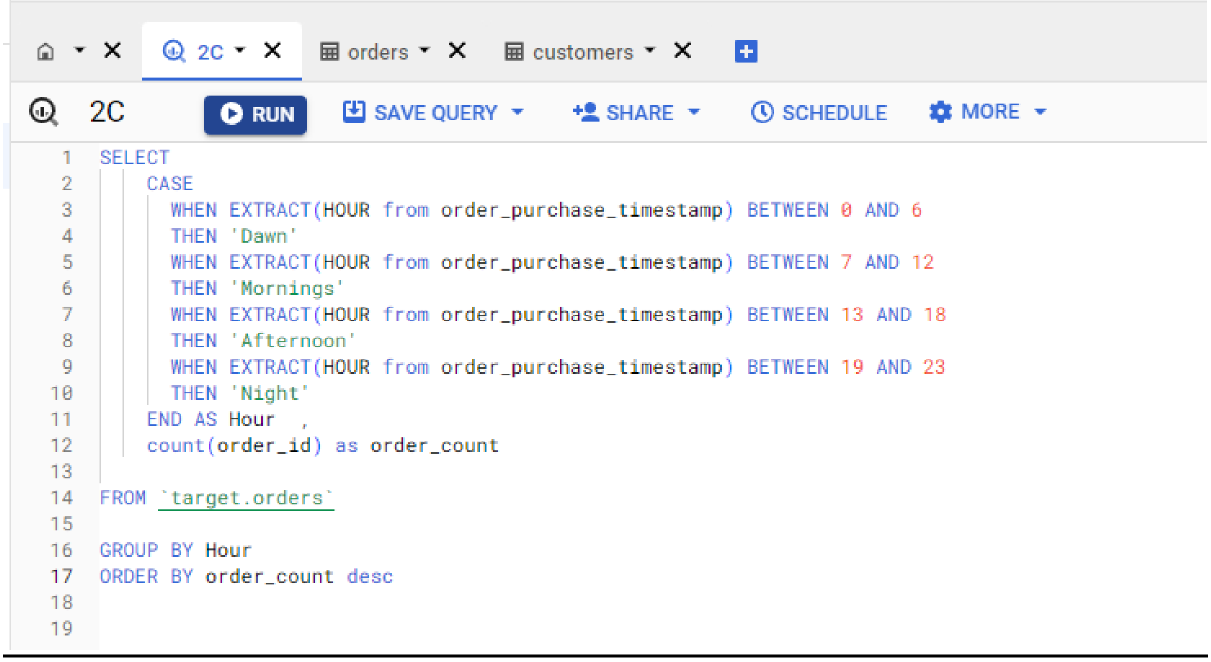
Results per page: 50 1 – 12 of 12

The result shows that in September and October months the number of orders are less and increases by November .

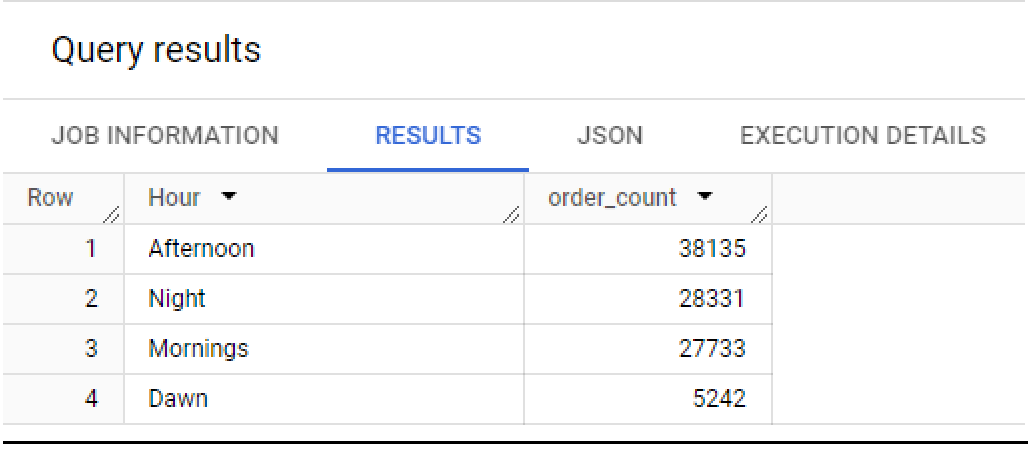
C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

In order to find out when the Brazilian customers mostly place their order we divide 24 hours into 4 intervals and find the number of orders placed in each interval.

Query :



```sql
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 '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 Hour  ,
    count(order_id) as order_count

FROM `target.orders`

GROUP BY Hour
ORDER BY order_count desc


```


Query results

Row	Hour	order_count
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

Insights

- There is a significant increase in the number of orders in November. This is largely due to the popular shopping festival which occurs in November.
- From the above results we can find that most of the orders are placed in the Afternoon which means Brazilians like to purchase products during their leisure time.
- Least number of orders were placed in Dawn which shows that the customers are not really engaged in purchasing items during this time period.

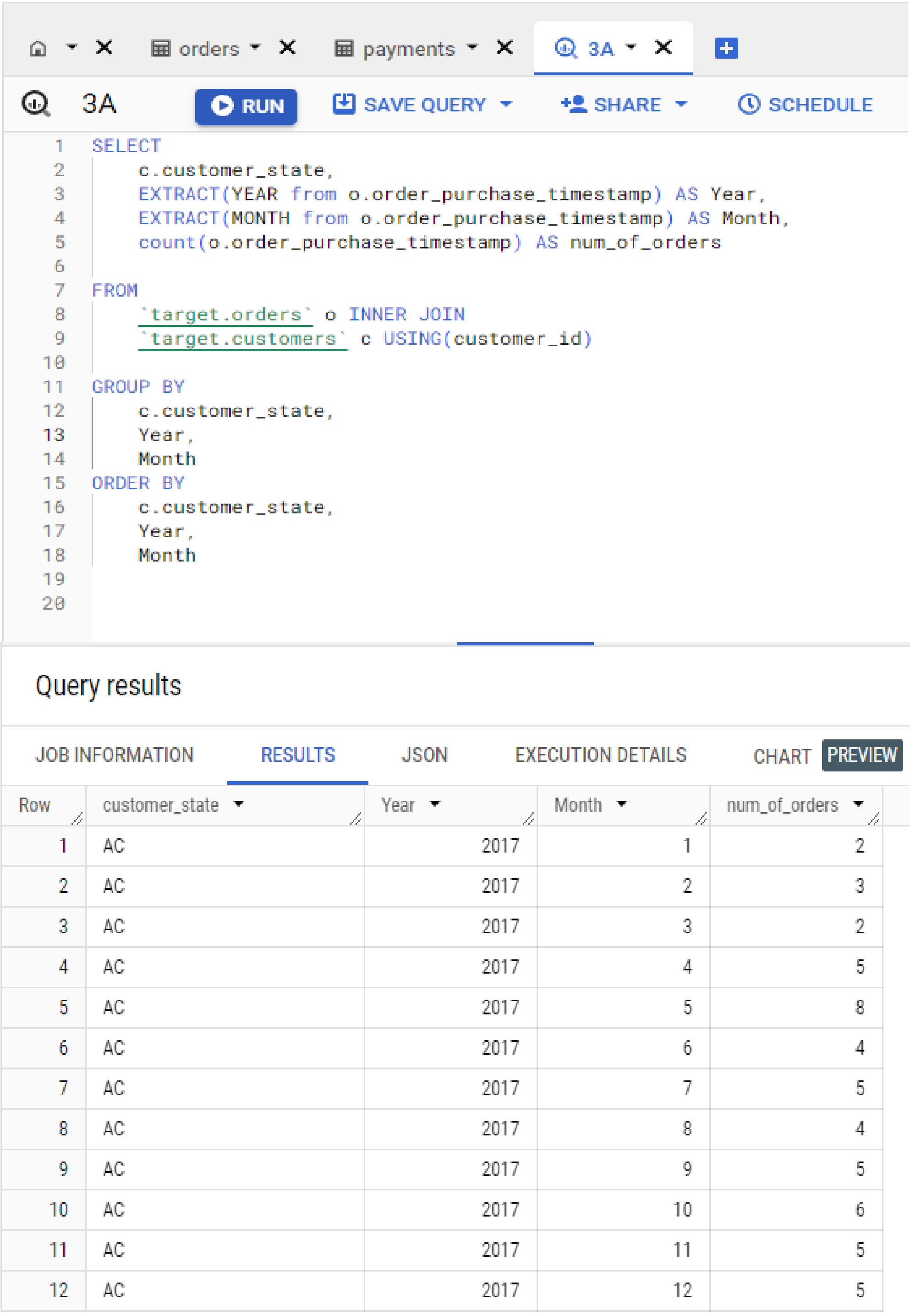
Recommendations

- We have to make sure that the servers should be up and running all the time especially during the time when customers engagement is high.
- In order to increase the purchase rate in September and October we can provide any kind of offers or lucky draws.

3) Evolution of E-commerce orders in the Brazil region:

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

The query and first 10 rows of result is given below :

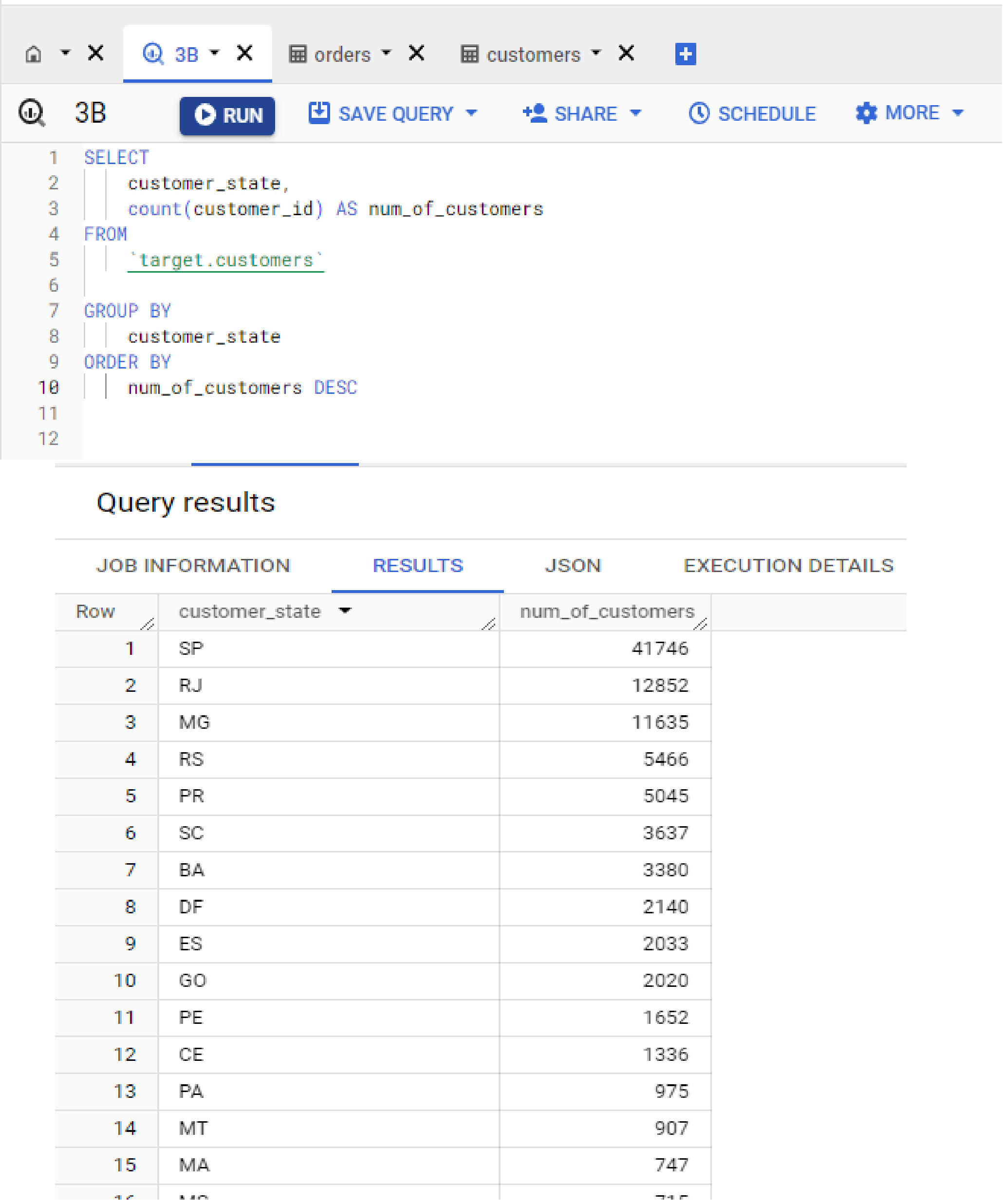


```sql
SELECT
    c.customer_state,
    EXTRACT(YEAR from o.order_purchase_timestamp) AS Year,
    EXTRACT(MONTH from o.order_purchase_timestamp) AS Month,
    count(o.order_purchase_timestamp) AS num_of_orders

FROM
    `target.orders` o INNER JOIN
    `target.customers` c USING(customer_id)

GROUP BY
    c.customer_state,
    Year,
    Month
ORDER BY
    c.customer_state,
    Year,
    Month
```

Row	customer_state	Year	Month	num_of_orders
1	AC	2017	1	2
2	AC	2017	2	3
3	AC	2017	3	2
4	AC	2017	4	5
5	AC	2017	5	8
6	AC	2017	6	4
7	AC	2017	7	5
8	AC	2017	8	4
9	AC	2017	9	5
10	AC	2017	10	6
11	AC	2017	11	5
12	AC	2017	12	5

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

The query and first 15 rows of the result is given below :



```sql
SELECT
    customer_state,
    count(customer_id) AS num_of_customers
FROM
    `target.customers`

GROUP BY
    customer_state
ORDER BY
    num_of_customers DESC
```

Row	customer_state	num_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
11	PE	1652
12	CE	1336
13	PA	975
14	MT	907
15	MA	747

Insights

- The Cities **SP, RJ, MG, RS** and **PR** is found to have more customers.

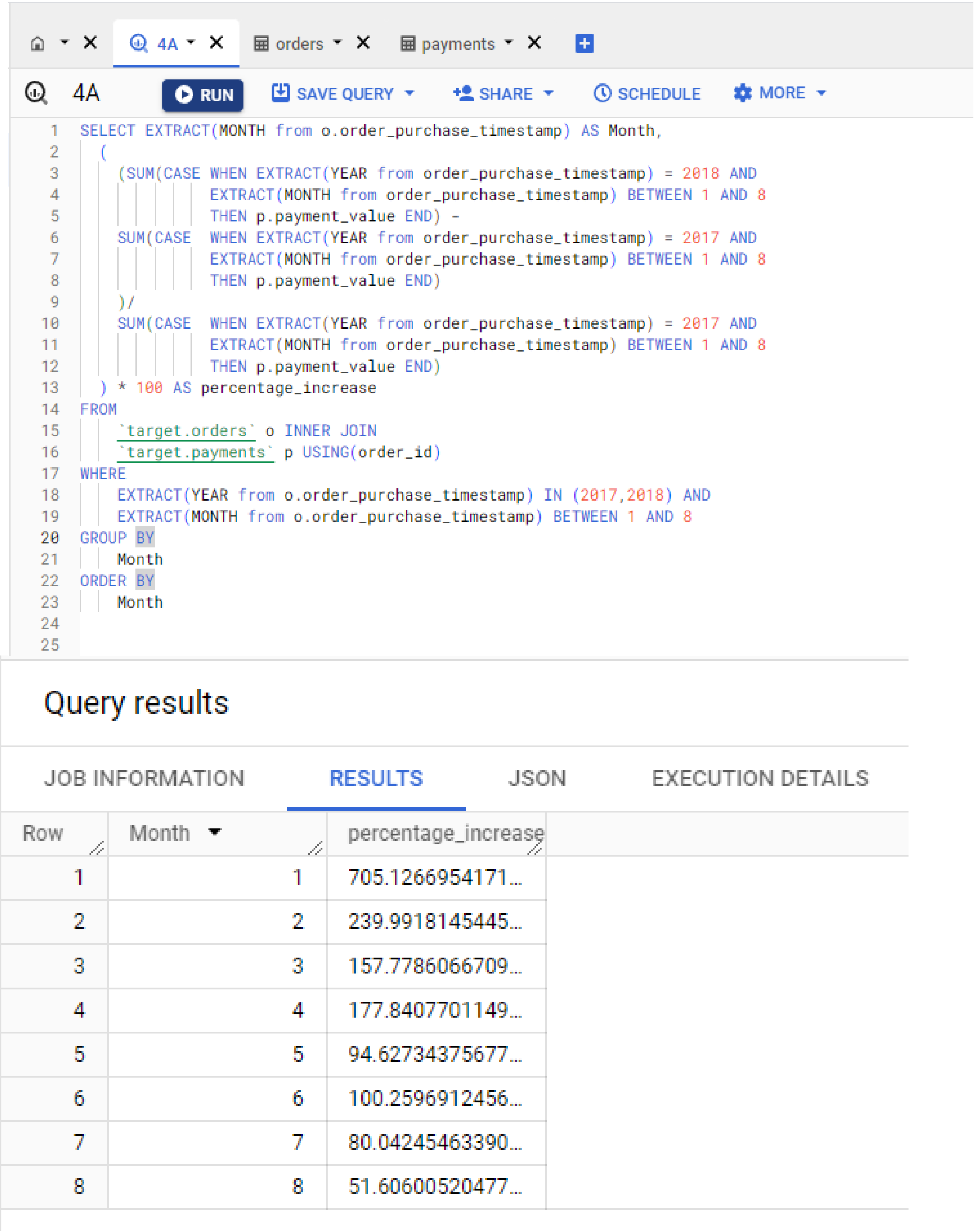
Recommendations

- Including more sellers and products in the e-commerce platform can increase the sales.

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).

The query and result is given below :

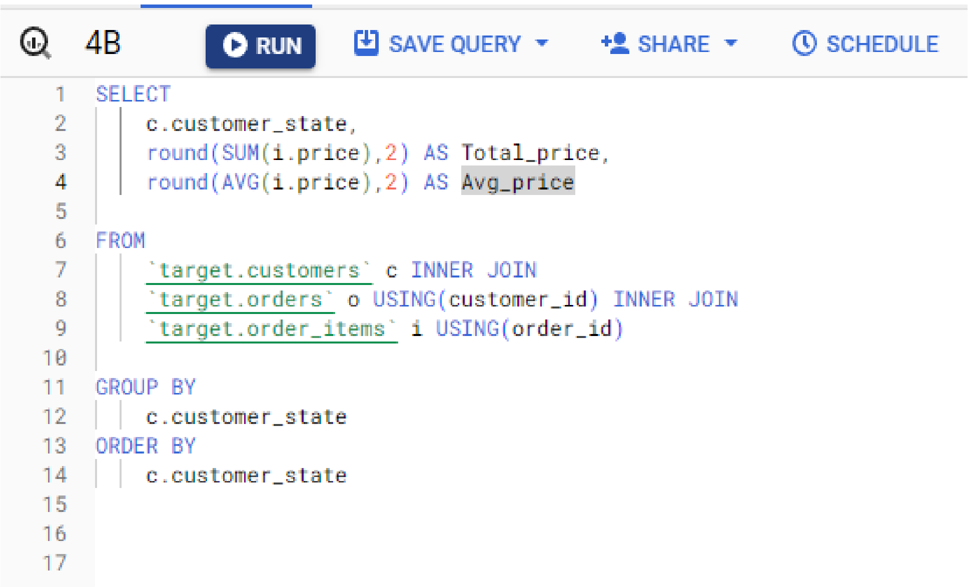


```
SELECT EXTRACT(MONTH from o.order_purchase_timestamp) AS Month,
  (
    (SUM(CASE WHEN EXTRACT(YEAR from order_purchase_timestamp) = 2018 AND
              EXTRACT(MONTH from order_purchase_timestamp) BETWEEN 1 AND 8
              THEN p.payment_value END) -
    SUM(CASE  WHEN EXTRACT(YEAR from order_purchase_timestamp) = 2017 AND
              EXTRACT(MONTH from order_purchase_timestamp) BETWEEN 1 AND 8
              THEN p.payment_value END)
    )/
    SUM(CASE  WHEN EXTRACT(YEAR from order_purchase_timestamp) = 2017 AND
              EXTRACT(MONTH from order_purchase_timestamp) BETWEEN 1 AND 8
              THEN p.payment_value END)
  ) * 100 AS percentage_increase
FROM
    `target.orders` o INNER JOIN
    `target.payments` p USING(order_id)
WHERE
    EXTRACT(YEAR from o.order_purchase_timestamp) IN (2017,2018) AND
    EXTRACT(MONTH from o.order_purchase_timestamp) BETWEEN 1 AND 8
GROUP BY
    Month
ORDER BY
    Month
```

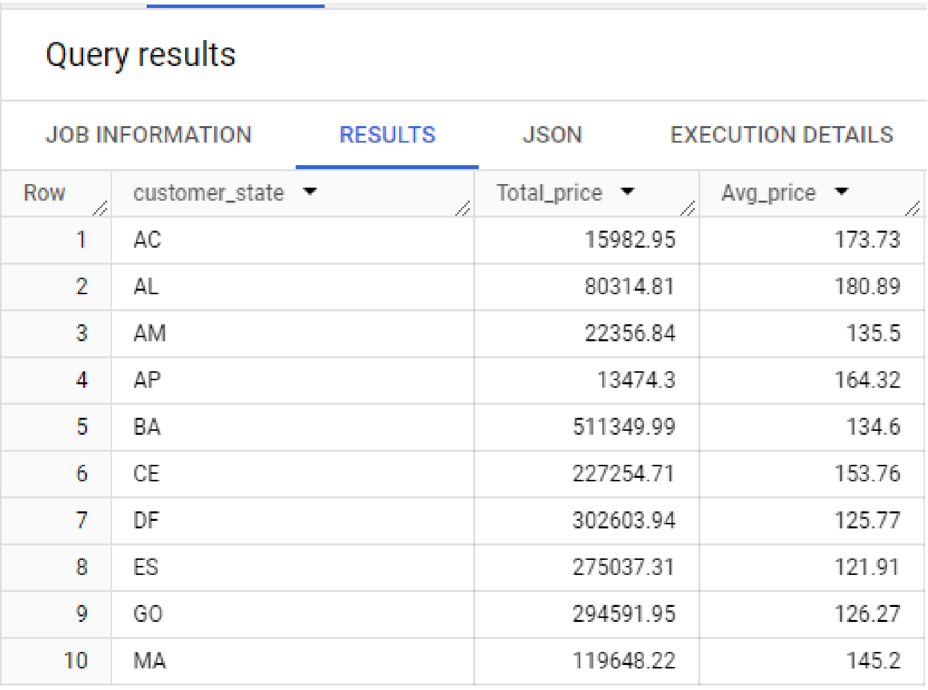
Row	Month	percentage_increase
1	1	705.1266954171...
2	2	239.9918145445...
3	3	157.7786066709...
4	4	177.8407701149...
5	5	94.62734375677...
6	6	100.2596912456...
7	7	80.04245463390...
8	8	51.60600520477...

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

The query and first 10 rows of the result is given below :



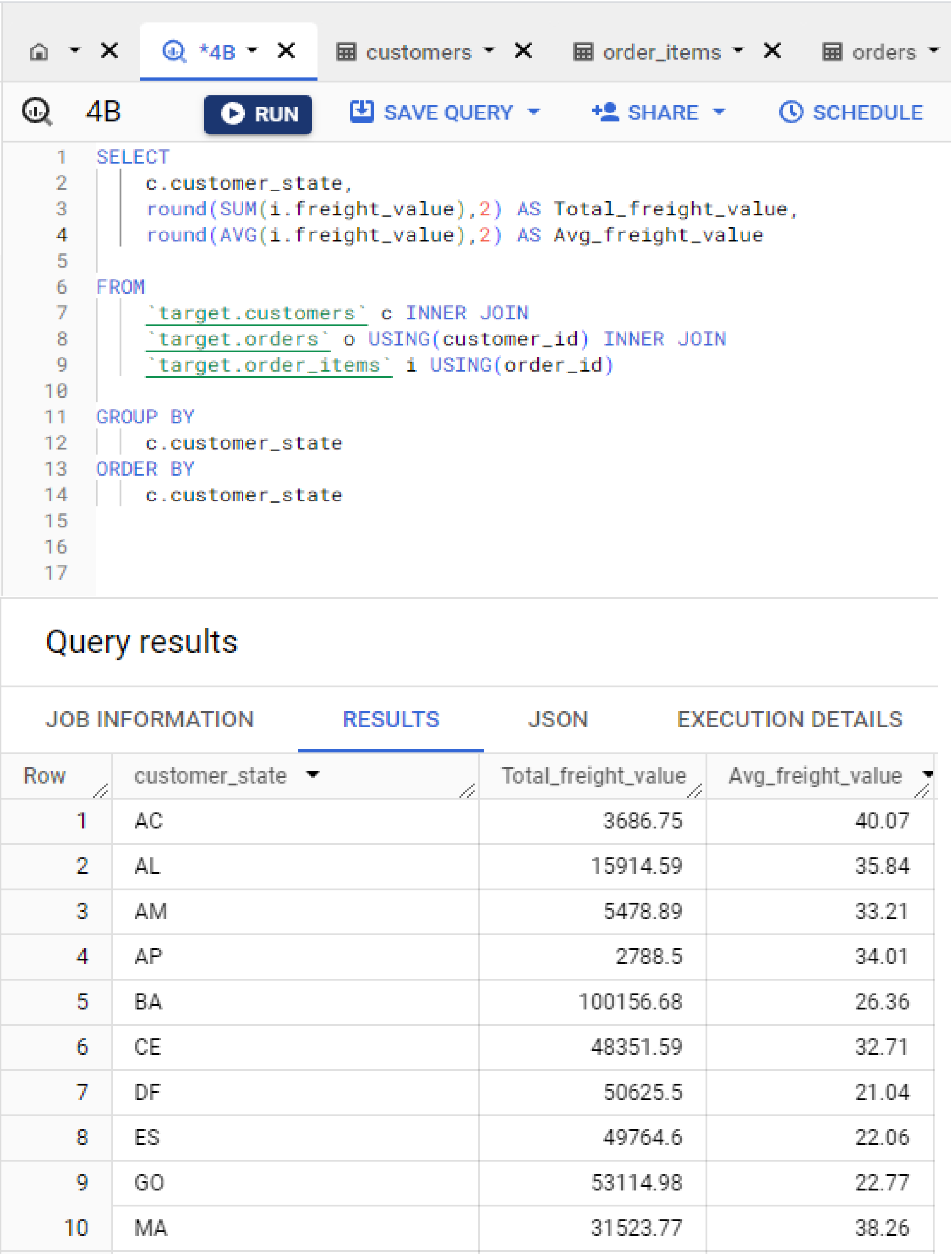
```sql
SELECT
    c.customer_state,
    round(SUM(i.price),2) AS Total_price,
    round(AVG(i.price),2) AS Avg_price

FROM
    `target.customers` c INNER JOIN
    `target.orders` o USING(customer_id) INNER JOIN
    `target.order_items` i USING(order_id)

GROUP BY
    c.customer_state
ORDER BY
    c.customer_state
```



Row	customer_state	Total_price	Avg_price
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2

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

The query and first 10 rows of the result is given below :



```
SELECT
    c.customer_state,
    round(SUM(i.freight_value),2) AS Total_freight_value,
    round(AVG(i.freight_value),2) AS Avg_freight_value

FROM
    `target.customers` c INNER JOIN
    `target.orders` o USING(customer_id) INNER JOIN
    `target.order_items` i USING(order_id)

GROUP BY
    c.customer_state
ORDER BY
    c.customer_state
```

Row	customer_state	Total_freight_value	Avg_freight_value
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26

Insights

- There is a comparative increase in the cost of orders from 2017 to 2018 as the platform grows.
- There is a significant variation in freight values.

Recommendations

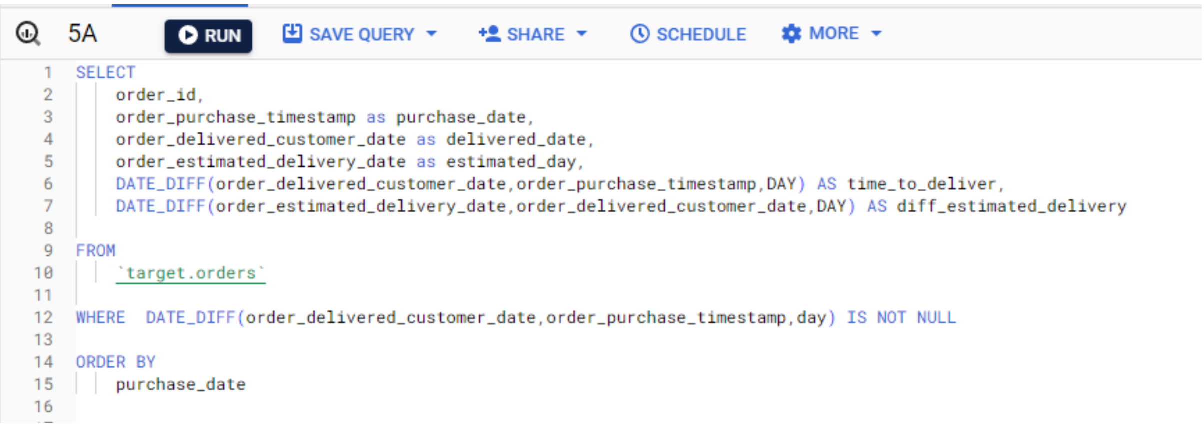
- As the e-commerce platform grows, the freight values can be reduced and profit can be increased. Appropriate strategies should be implemented to maintain competitiveness and cost effectiveness.

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

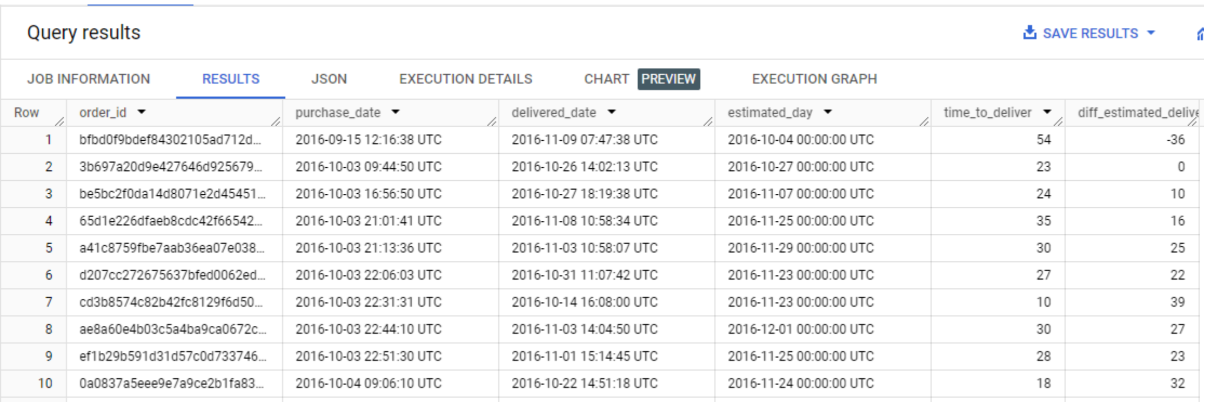
A. Find the no. of days taken to deliver each order from the order's purchase date as delivery time.

Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

Do this in a single query.



```sql
SELECT
    order_id,
    order_purchase_timestamp as purchase_date,
    order_delivered_customer_date as delivered_date,
    order_estimated_delivery_date as estimated_day,
    DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS time_to_deliver,
    DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS diff_estimated_delivery

FROM
    `target.orders`

WHERE  DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,day) IS NOT NULL

ORDER BY
    purchase_date

```



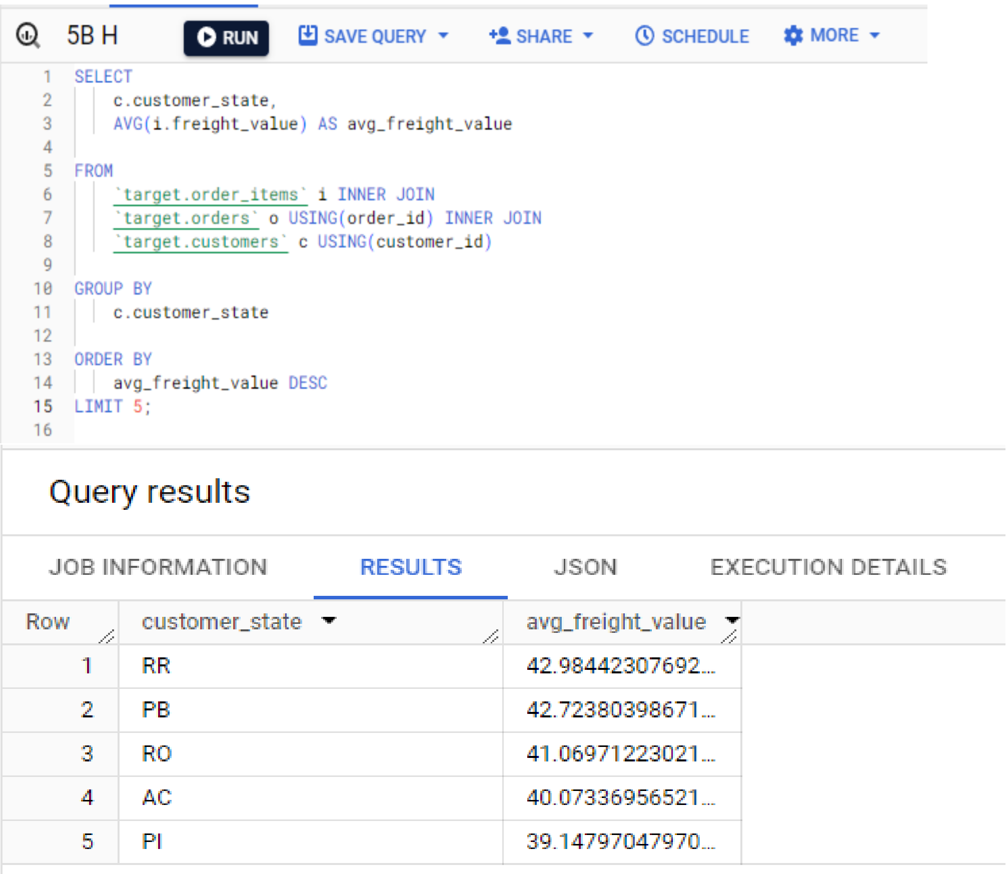
Row	order_id	purchase_date	delivered_date	estimated_day	time_to_deliver	diff_estimated_delive
1	bfbd0f9bdef84302105ad712d...	2016-09-15 12:16:38 UTC	2016-11-09 07:47:38 UTC	2016-10-04 00:00:00 UTC	54	-36
2	3b697a20d9e427646d925679...	2016-10-03 09:44:50 UTC	2016-10-26 14:02:13 UTC	2016-10-27 00:00:00 UTC	23	0
3	be5bc2f0da14d8071e2d45451...	2016-10-03 16:56:50 UTC	2016-10-27 18:19:38 UTC	2016-11-07 00:00:00 UTC	24	10
4	65d1e226dfaeb8cdc42f66542...	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	2016-11-25 00:00:00 UTC	35	16
5	a41c8759fbe7aab36ea07e038...	2016-10-03 21:13:36 UTC	2016-11-03 10:58:07 UTC	2016-11-29 00:00:00 UTC	30	25
6	d207cc272675637bfed0062ed...	2016-10-03 22:06:03 UTC	2016-10-31 11:07:42 UTC	2016-11-23 00:00:00 UTC	27	22
7	cd3b8574c82b42fc8129f6d50...	2016-10-03 22:31:31 UTC	2016-10-14 16:08:00 UTC	2016-11-23 00:00:00 UTC	10	39
8	ae8a60e4b03c5a4ba9ca0672c...	2016-10-03 22:44:10 UTC	2016-11-03 14:04:50 UTC	2016-12-01 00:00:00 UTC	30	27
9	ef1b29b591d31d57c0d733746...	2016-10-03 22:51:30 UTC	2016-11-01 15:14:45 UTC	2016-11-25 00:00:00 UTC	28	23
10	0a0837a5eee9e7a9ce2b1fa83...	2016-10-04 09:06:10 UTC	2016-10-22 14:51:18 UTC	2016-11-24 00:00:00 UTC	18	32

(The negative sign shows that order was delivered late)

The query and the first 10 rows of the result is given above :

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

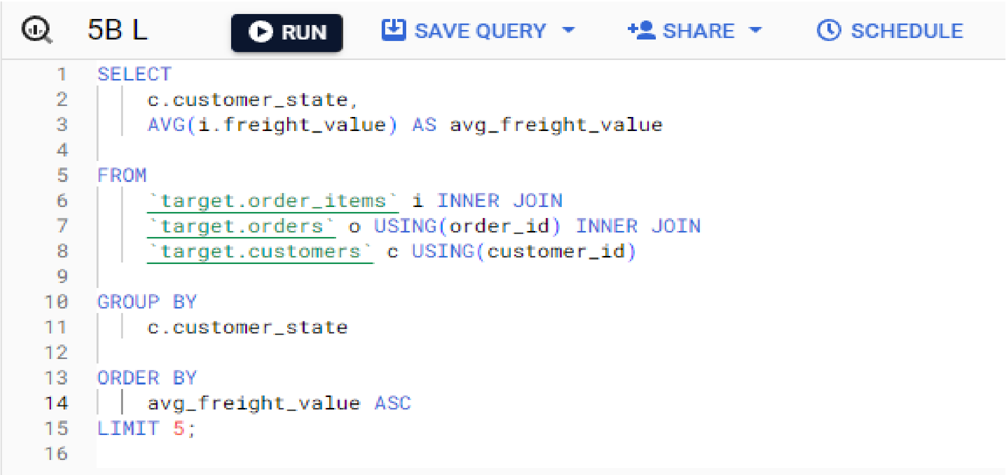
The states with highest freight value :



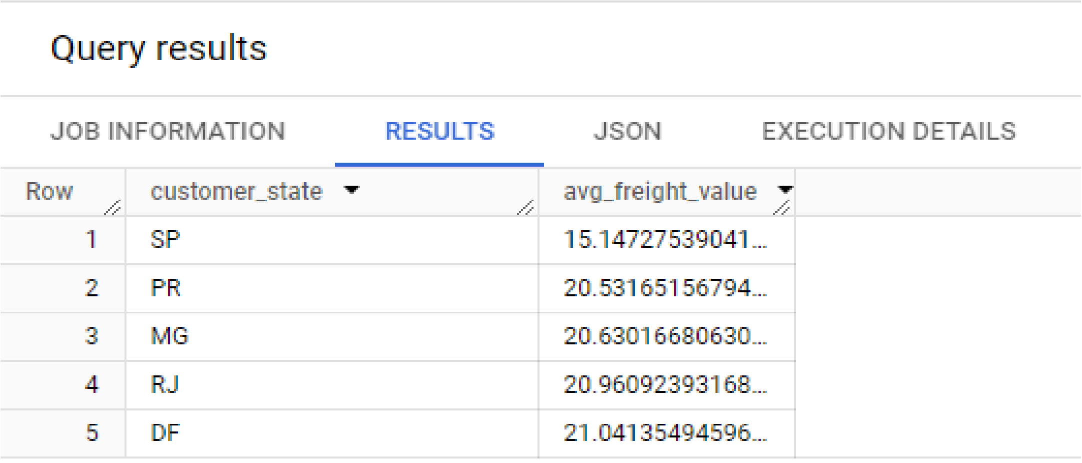
```sql
SELECT
    c.customer_state,
    AVG(i.freight_value) AS avg_freight_value

FROM
    `target.order_items` i INNER JOIN
    `target.orders` o USING(order_id) INNER JOIN
    `target.customers` c USING(customer_id)

GROUP BY
    c.customer_state

ORDER BY
    avg_freight_value DESC
LIMIT 5;
```

Row	customer_state	avg_freight_value
1	RR	42.98442307692…
2	PB	42.72380398671…
3	RO	41.06971223021…
4	AC	40.07336956521…
5	PI	39.14797047970…

The states with lowest freight value :



```sql
SELECT
    c.customer_state,
    AVG(i.freight_value) AS avg_freight_value

FROM
    `target.order_items` i INNER JOIN
    `target.orders` o USING(order_id) INNER JOIN
    `target.customers` c USING(customer_id)

GROUP BY
    c.customer_state

ORDER BY
    avg_freight_value ASC
LIMIT 5;
```

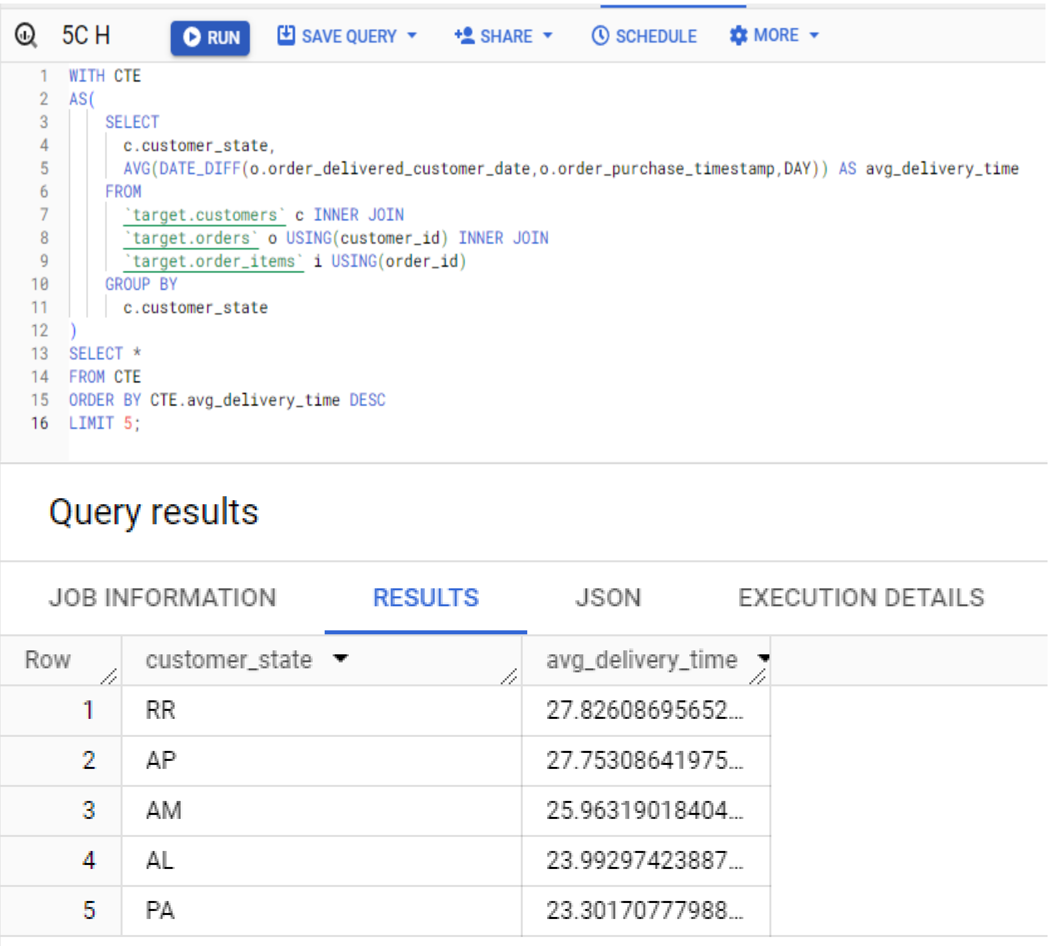


Query results

Row	customer_state	avg_freight_value
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...

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

The states with highest average delivery time :

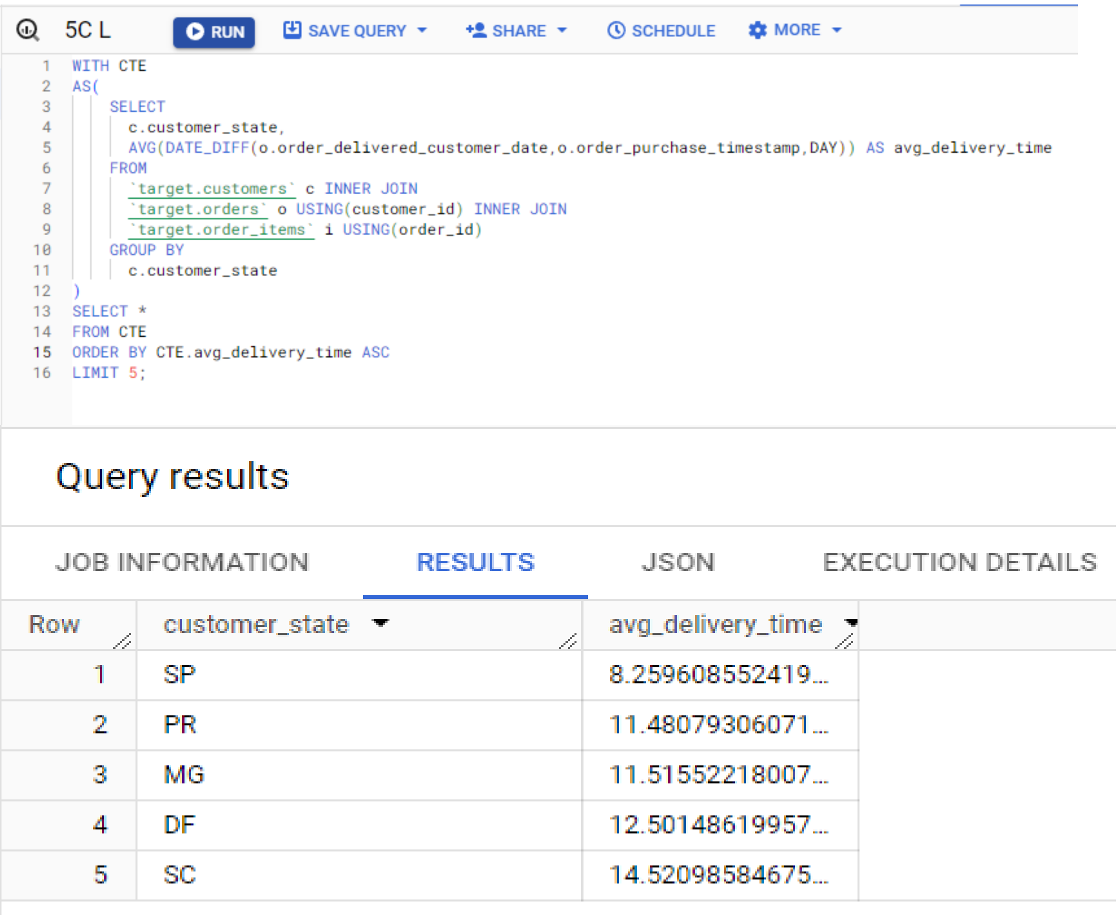


```
WITH CTE
AS(
    SELECT
      c.customer_state,
      AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)) AS avg_delivery_time
    FROM
      `target.customers` c INNER JOIN
      `target.orders` o USING(customer_id) INNER JOIN
      `target.order_items` i USING(order_id)
    GROUP BY
      c.customer_state
)
SELECT *
FROM CTE
ORDER BY CTE.avg_delivery_time DESC
LIMIT 5;
```

Query results

Row	customer_state	avg_delivery_time
1	RR	27.82608695652...
2	AP	27.75308641975...
3	AM	25.96319018404...
4	AL	23.99297423887...
5	PA	23.30170777988...

The states with lowest average delivery time :

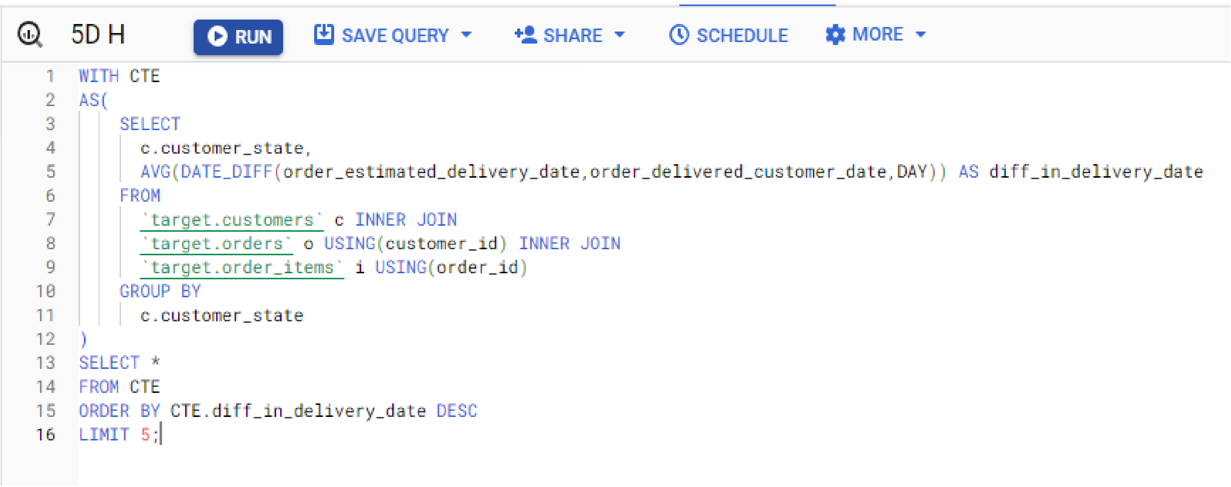


```sql
WITH CTE
AS(
    SELECT
      c.customer_state,
      AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)) AS avg_delivery_time
    FROM
      `target.customers` c INNER JOIN
      `target.orders` o USING(customer_id) INNER JOIN
      `target.order_items` i USING(order_id)
    GROUP BY
      c.customer_state
)
SELECT *
FROM CTE
ORDER BY CTE.avg_delivery_time ASC
LIMIT 5;
```

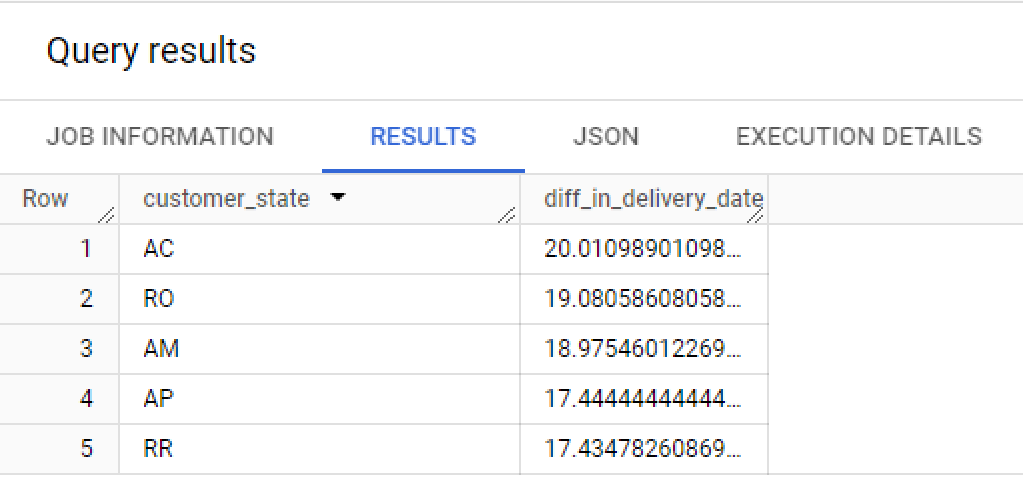
Query results

Row	customer_state	avg_delivery_time
1	SP	8.259608552419...
2	PR	11.48079306071...
3	MG	11.51552218007...
4	DF	12.50148619957...
5	SC	14.52098584675...

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



```sql
WITH CTE
AS(
    SELECT
      c.customer_state,
      AVG(DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)) AS diff_in_delivery_date
    FROM
      `target.customers` c INNER JOIN
      `target.orders` o USING(customer_id) INNER JOIN
      `target.order_items` i USING(order_id)
    GROUP BY
      c.customer_state
)
SELECT *
FROM CTE
ORDER BY CTE.diff_in_delivery_date DESC
LIMIT 5;
```



Row	customer_state	diff_in_delivery_date
1	AC	20.01098901098...
2	RO	19.08058608058...
3	AM	18.97546012269...
4	AP	17.44444444444...
5	RR	17.43478260869...

Insights

- In some states there is a high difference in estimated delivery date and the actual delivered date. Even though the company delivers the products in advance in some situations, it may cause some issue.

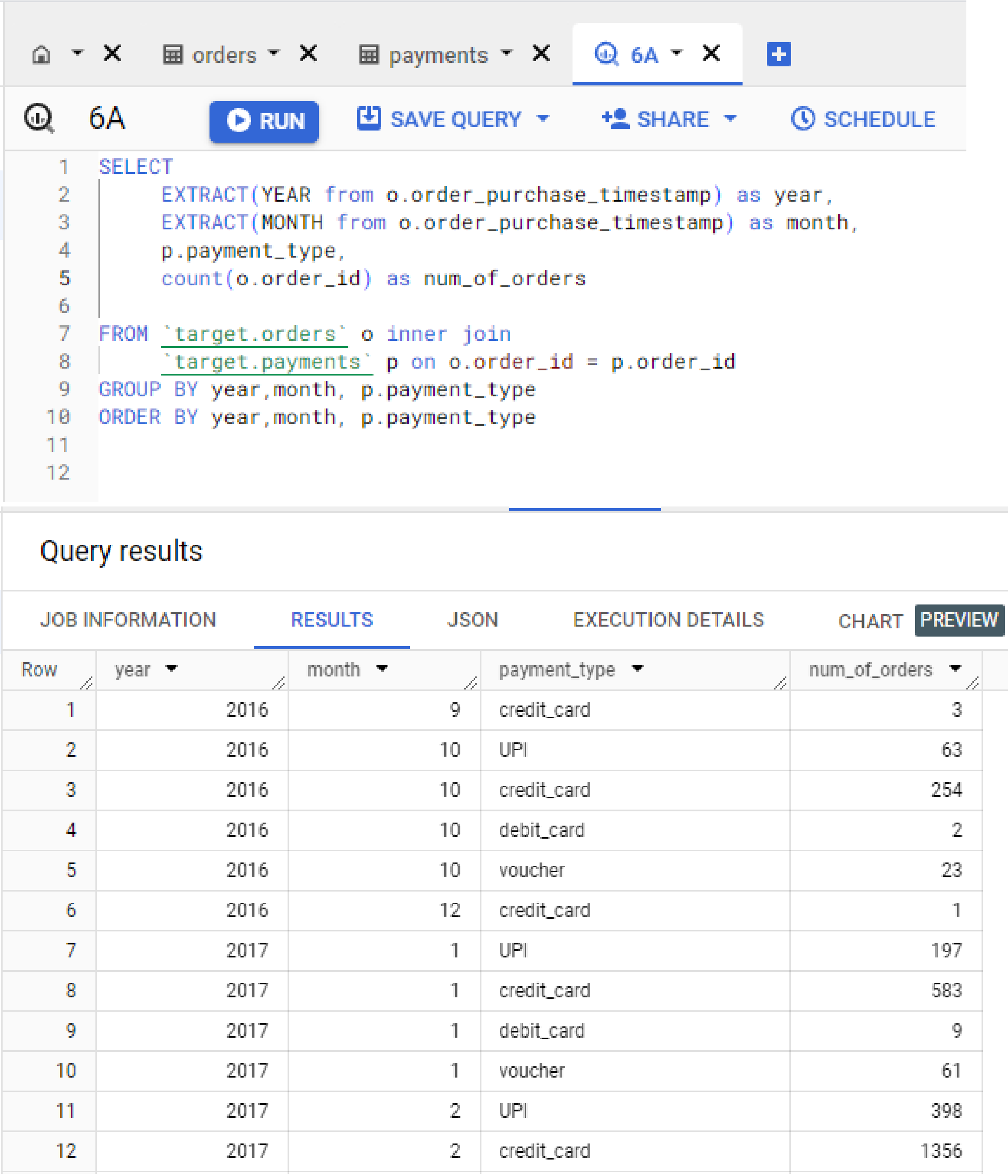
Recommendations

- The difference between the estimated delivery date and the actual delivery date should be minimal or else it may cause customer dissatisfaction.
- By analysing the orders in certain locations and the demand for that order the company can store more estimated products in the nearest warehouses so that they can decrease the cost per item.

6) Analysis based on the payments:

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

The query and the first 10 rows of the result is given below :

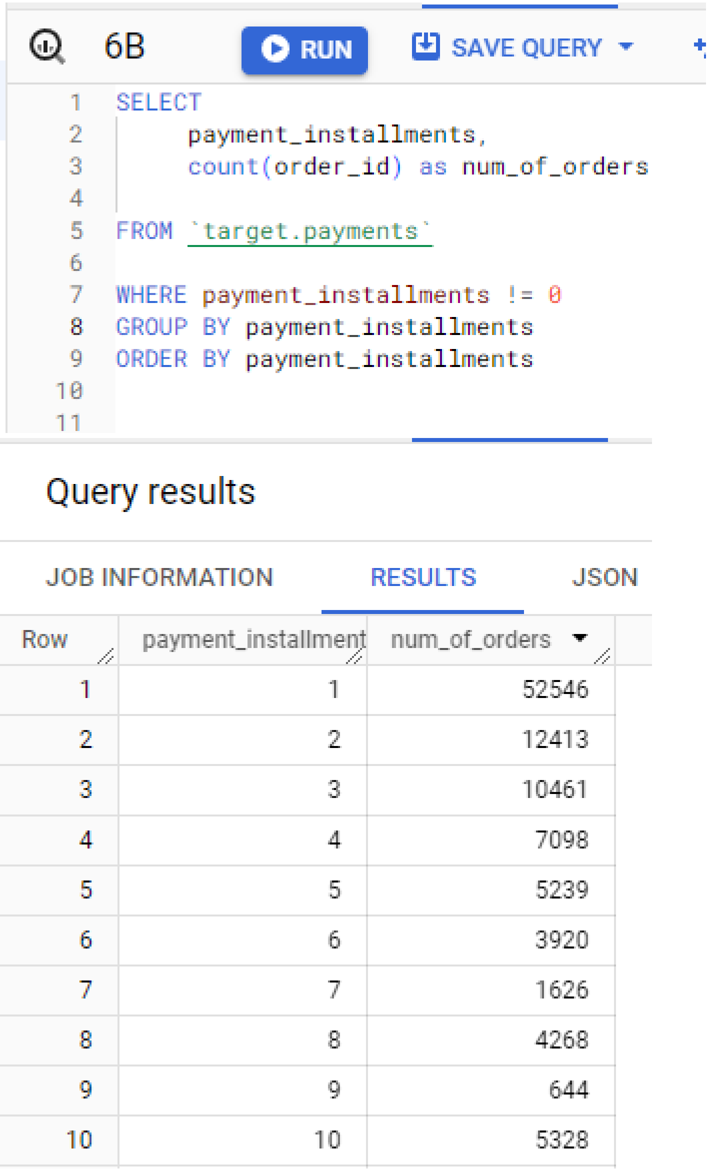


```sql
SELECT
     EXTRACT(YEAR from o.order_purchase_timestamp) as year,
     EXTRACT(MONTH from o.order_purchase_timestamp) as month,
     p.payment_type,
     count(o.order_id) as num_of_orders

FROM `target.orders` o inner join
     `target.payments` p on o.order_id = p.order_id
GROUP BY year,month, p.payment_type
ORDER BY year,month, p.payment_type
```

Row	year	month	payment_type	num_of_orders
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	254
4	2016	10	debit_card	2
5	2016	10	voucher	23
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	583
9	2017	1	debit_card	9
10	2017	1	voucher	61
11	2017	2	UPI	398
12	2017	2	credit_card	1356

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

The query and the first 10 rows of the result is given below :



```sql
SELECT
     payment_installments,
     count(order_id) as num_of_orders

FROM `target.payments`

WHERE payment_installments != 0
GROUP BY payment_installments
ORDER BY payment_installments
```

Row	payment_installment	num_of_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239
6	6	3920
7	7	1626
8	8	4268
9	9	644
10	10	5328

Insights

- It can be seen that customers use credit cards often to pay mostly.
- Also more customers choose EMI options to pay which suggests that the e-commerce platform in Brazil is developing

Recommendations

- A separate EMI payment option can be included since many people opt that method