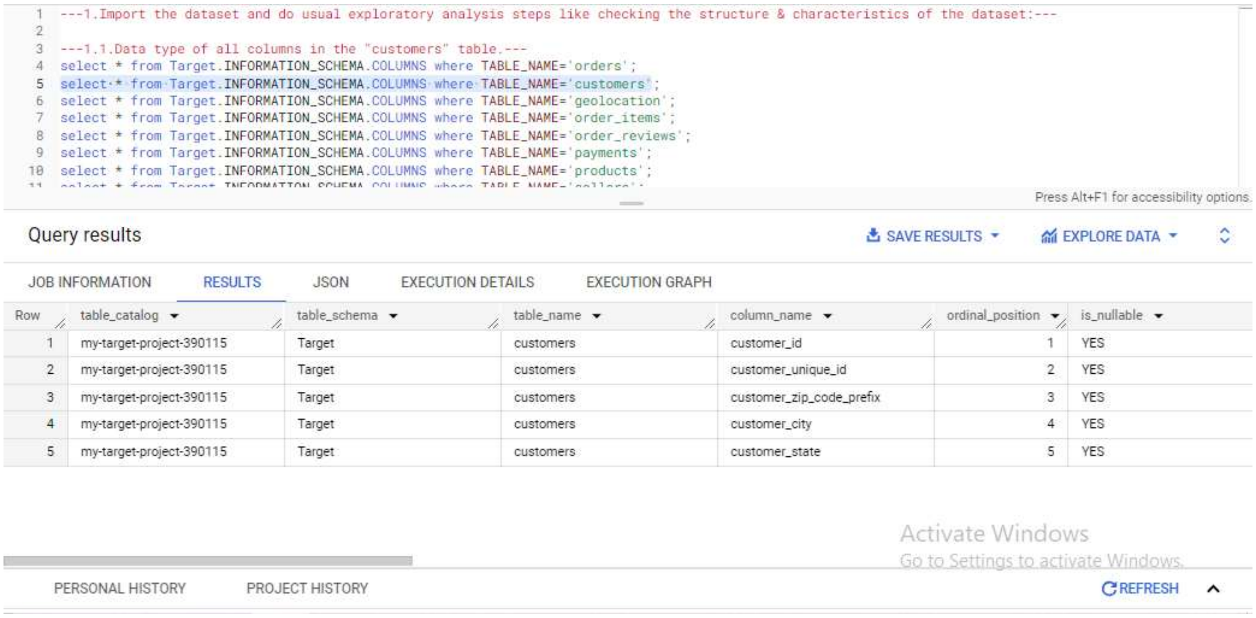
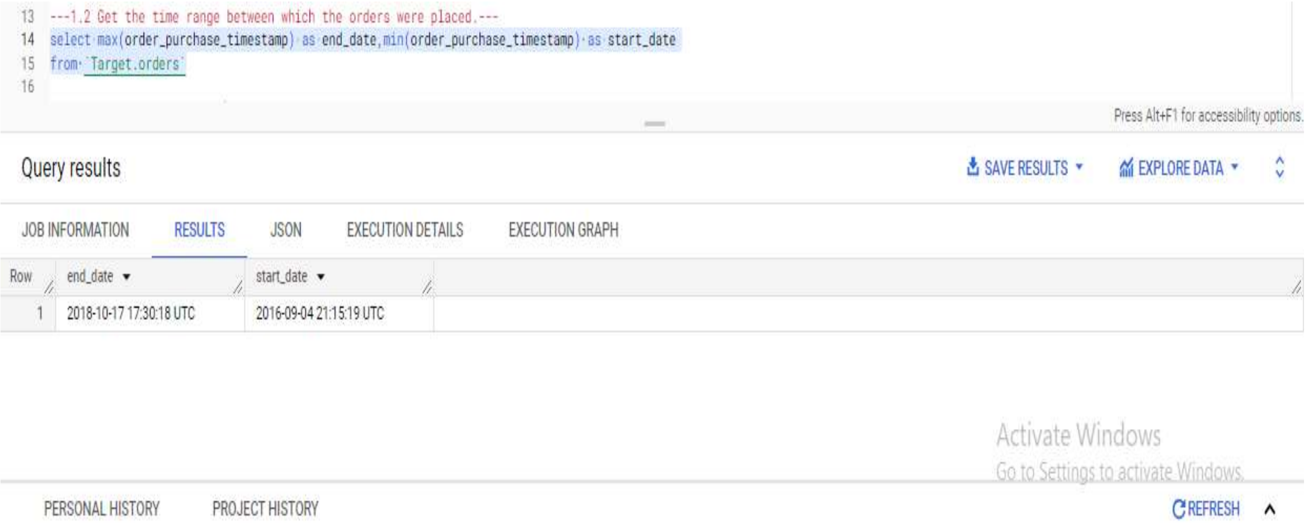


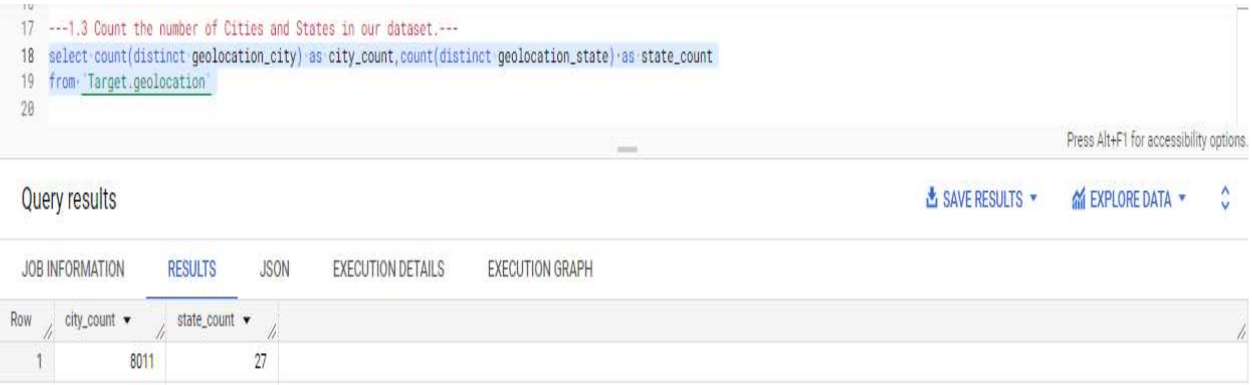
```sql
---1.Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:---

---1.1.Data type of all columns in the "customers" table.---
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='orders';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='customers';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='geolocation';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='order_items';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='order_reviews';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='payments';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='products';
select * from Target.INFORMATION_SCHEMA.COLUMNS where TABLE_NAME='sellers';
```

Row	table_catalog	table_schema	table_name	column_name	ordinal_position	is_nullable
1	my-target-project-390115	Target	customers	customer_id	1	YES
2	my-target-project-390115	Target	customers	customer_unique_id	2	YES
3	my-target-project-390115	Target	customers	customer_zip_code_prefix	3	YES
4	my-target-project-390115	Target	customers	customer_city	4	YES
5	my-target-project-390115	Target	customers	customer_state	5	YES



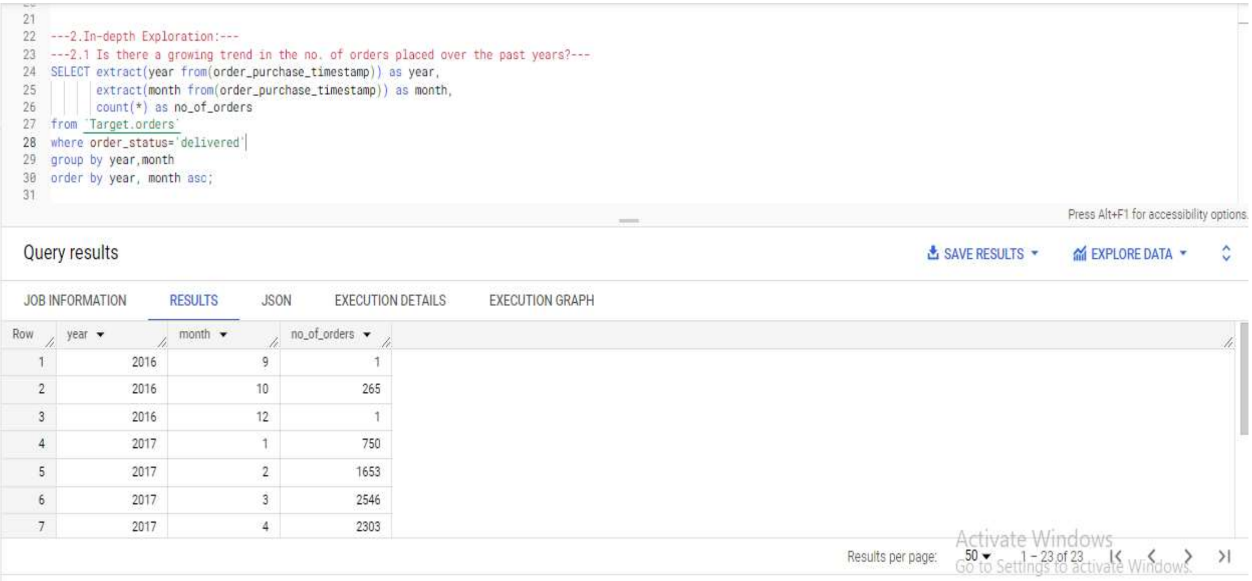
```sql
---1.2 Get the time range between which the orders were placed.---
select max(order_purchase_timestamp) as end_date,min(order_purchase_timestamp) as start_date
from `Target.orders`
```

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



```
---1.3 Count the number of Cities and States in our dataset.---
select count(distinct geolocation_city) as city_count,count(distinct geolocation_state) as state_count
from `Target.geolocation`
```

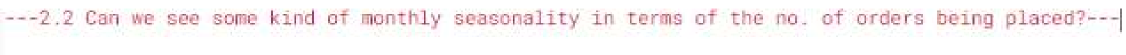
Row	city_count	state_count
1	8011	27



```
---2.In-depth Exploration:---
---2.1 Is there a growing trend in the no. of orders placed over the past years?---
SELECT extract(year from(order_purchase_timestamp)) as year,
       extract(month from(order_purchase_timestamp)) as month,
       count(*) as no_of_orders
from `Target.orders`
where order_status='delivered'
group by year,month
order by year, month asc;
```

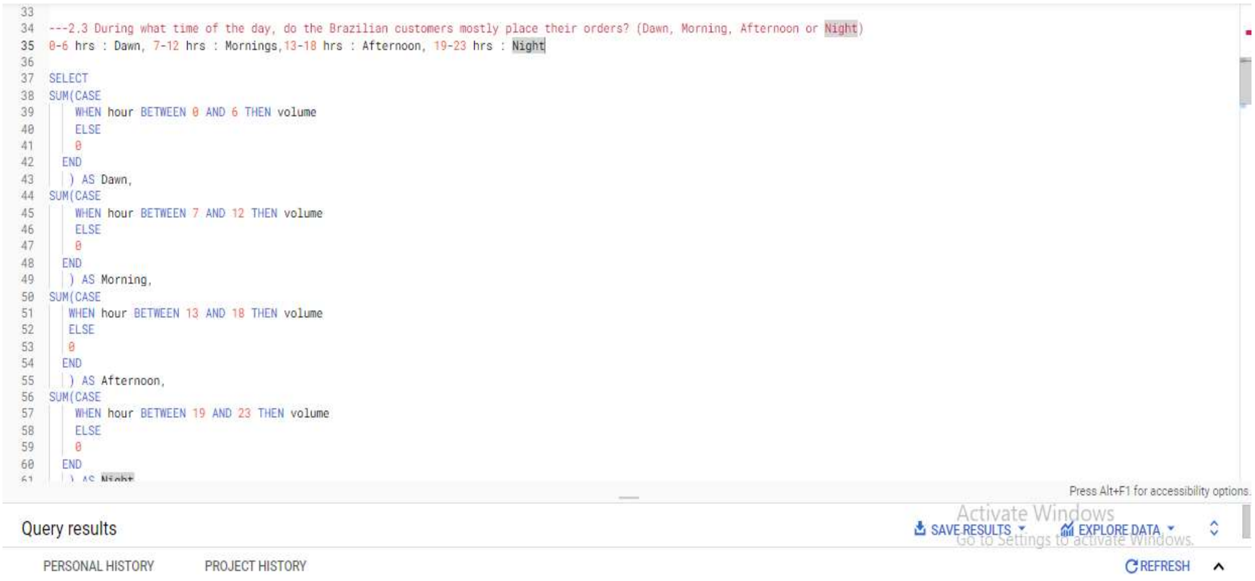
Row	year	month	no_of_orders
1	2016	9	1
2	2016	10	265
3	2016	12	1
4	2017	1	750
5	2017	2	1653
6	2017	3	2546
7	2017	4	2303

“YES”, there is a growing trend in the number of orders as seeing by the month on month numbers.

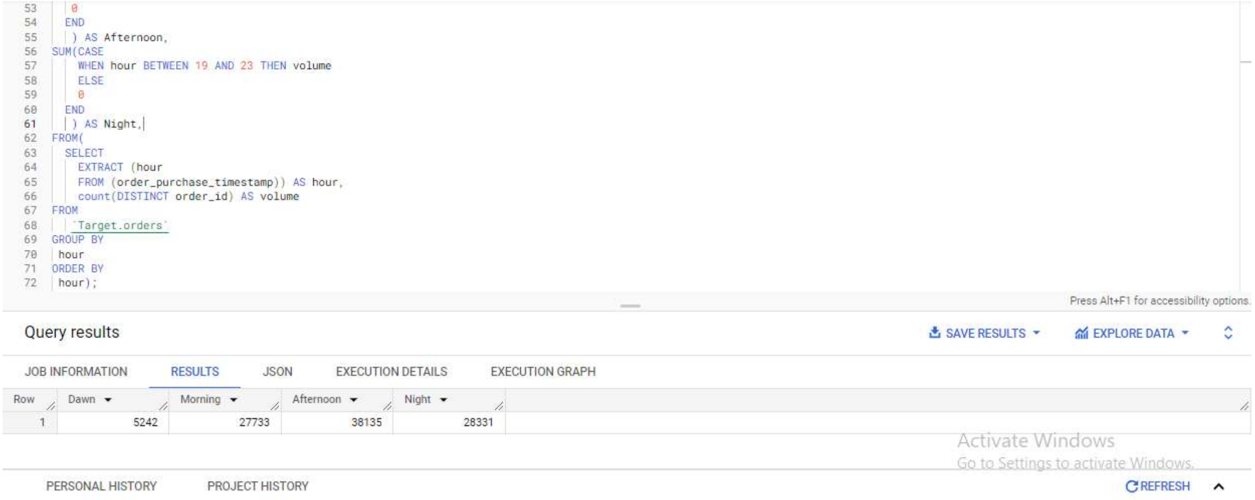


```
---2.2 Can we see some kind of monthly seasonality in terms of the no. of orders being placed?---
```

- “YES”, It is observed that the winter season has the maximum sales(Oct.,Nov.,Dec.).



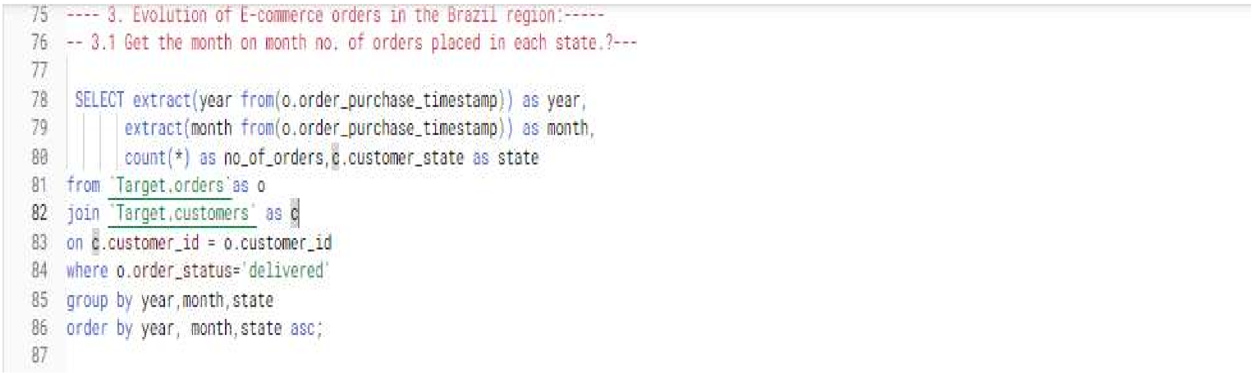
```
---2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
0-6 hrs : Dawn, 7-12 hrs : Mornings,13-18 hrs : Afternoon, 19-23 hrs : Night

SELECT
SUM(CASE
    WHEN hour BETWEEN 0 AND 6 THEN volume
    ELSE
    0
  END
  ) AS Dawn,
SUM(CASE
    WHEN hour BETWEEN 7 AND 12 THEN volume
    ELSE
    0
  END
  ) AS Morning,
SUM(CASE
   WHEN hour BETWEEN 13 AND 18 THEN volume
   ELSE
   0
  END
  ) AS Afternoon,
SUM(CASE
    WHEN hour BETWEEN 19 AND 23 THEN volume
    ELSE
    0
  END
  ) AS Night
```



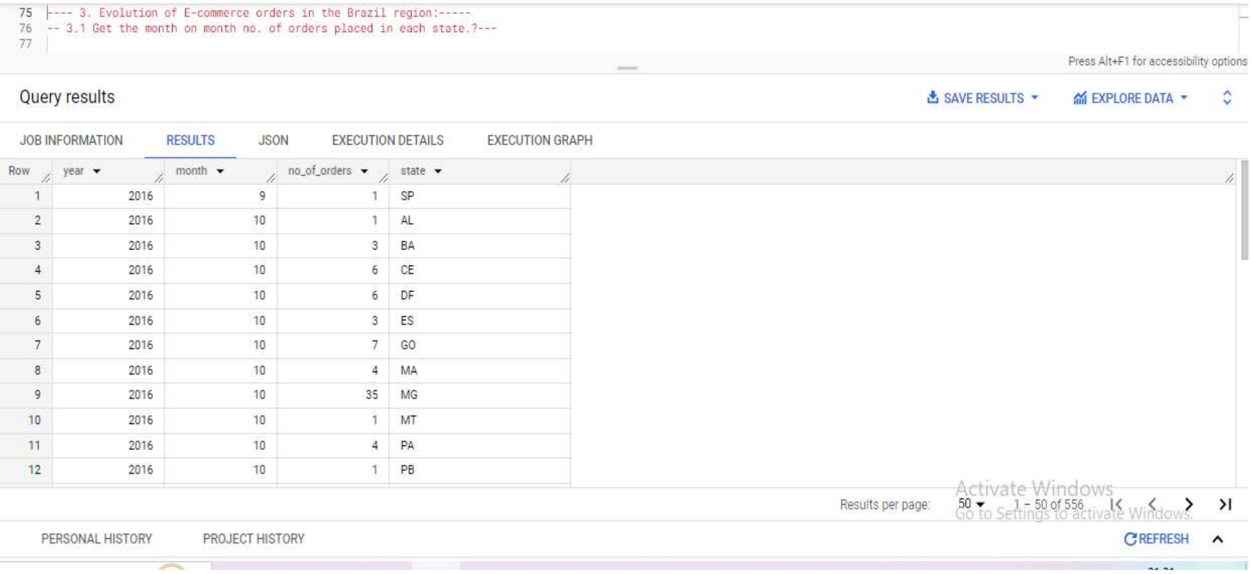
```
   0
  END
  ) AS Afternoon,
SUM(CASE
    WHEN hour BETWEEN 19 AND 23 THEN volume
    ELSE
    0
  END
  ) AS Night,
FROM(
  SELECT
    EXTRACT (hour
    FROM (order_purchase_timestamp)) AS hour,
    count(DISTINCT order_id) AS volume
FROM
  `Target.orders`
GROUP BY
 hour
ORDER BY
 hour);
```

Row	Dawn	Morning	Afternoon	Night
1	5242	27733	38135	28331

The maximum number of transactions have happened in the afternoon (38135).

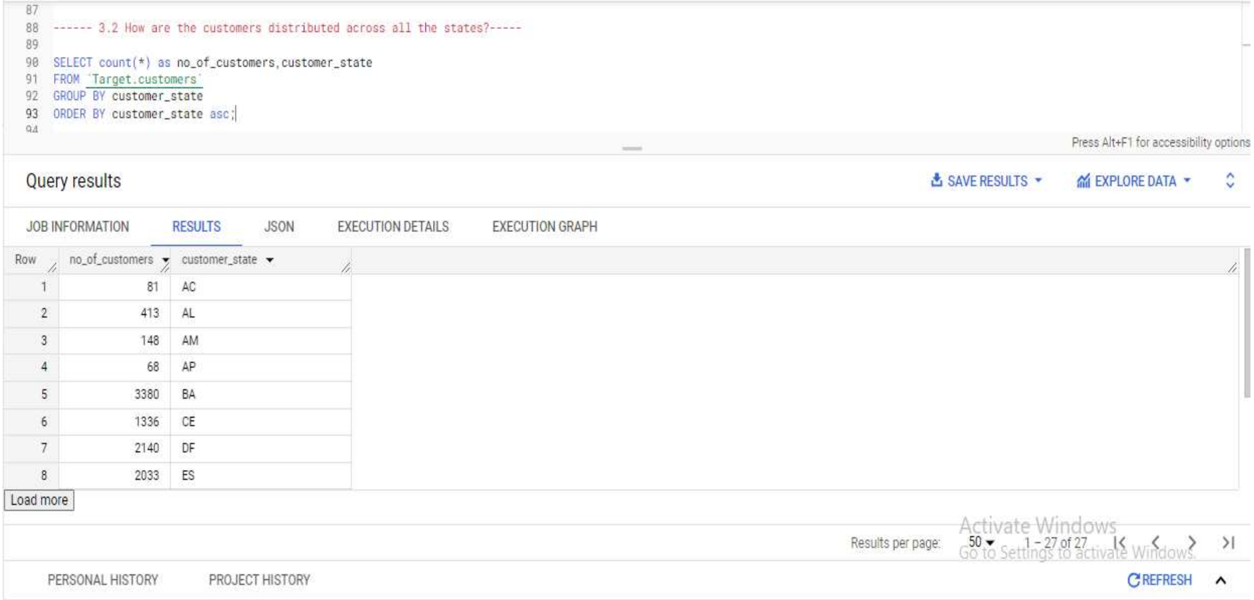


```
---- 3. Evolution of E-commerce orders in the Brazil region:-----
-- 3.1 Get the month on month no. of orders placed in each state.?---

 SELECT extract(year from(o.order_purchase_timestamp)) as year,
        extract(month from(o.order_purchase_timestamp)) as month,
        count(*) as no_of_orders,c.customer_state as state
from `Target.orders`as o
join `Target.customers` as c
on c.customer_id = o.customer_id
where o.order_status='delivered'
group by year,month,state
order by year, month,state asc;
```



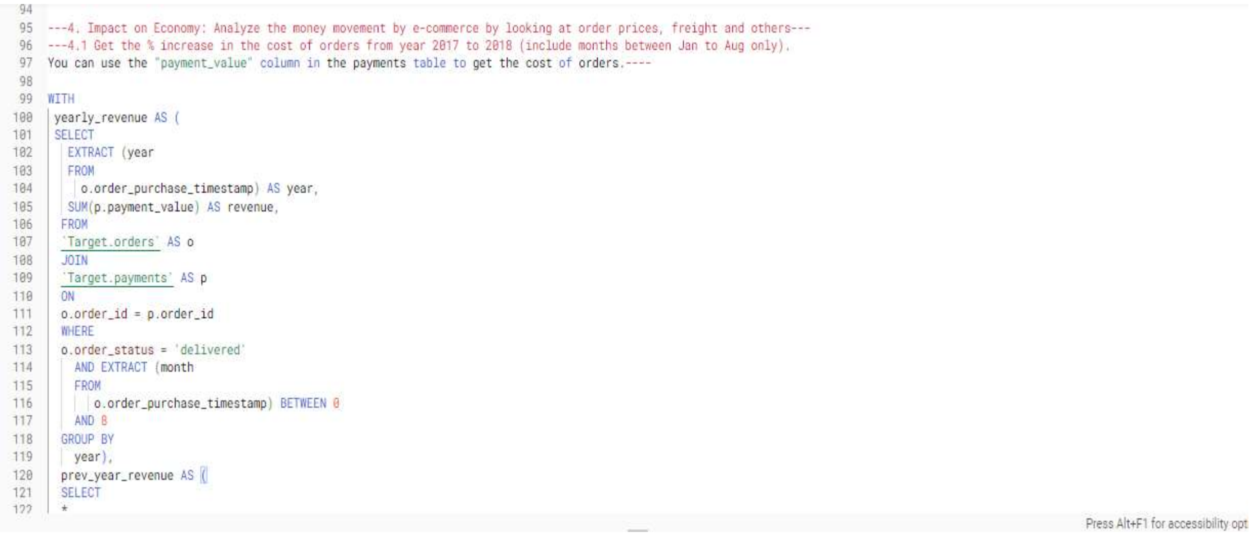
```
---- 3. Evolution of E-commerce orders in the Brazil region:-----
-- 3.1 Get the month on month no. of orders placed in each state.?---
```

Row	year	month	no_of_orders	state
1	2016	9	1	SP
2	2016	10	1	AL
3	2016	10	3	BA
4	2016	10	6	CE
5	2016	10	6	DF
6	2016	10	3	ES
7	2016	10	7	GO
8	2016	10	4	MA
9	2016	10	35	MG
10	2016	10	1	MT
11	2016	10	4	PA
12	2016	10	1	PB

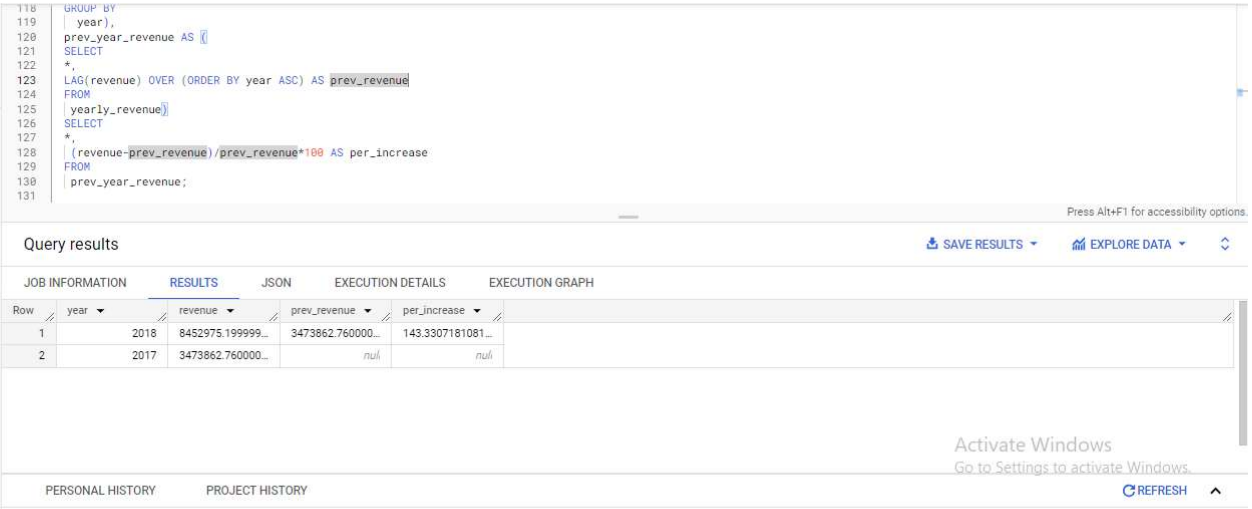


```
------ 3.2 How are the customers distributed across all the states?-----

SELECT count(*) as no_of_customers,customer_state
FROM `Target.customers`
GROUP BY customer_state
ORDER BY customer_state asc;
```

Row	no_of_customers	customer_state
1	81	AC
2	413	AL
3	148	AM
4	68	AP
5	3380	BA
6	1336	CE
7	2140	DF
8	2033	ES

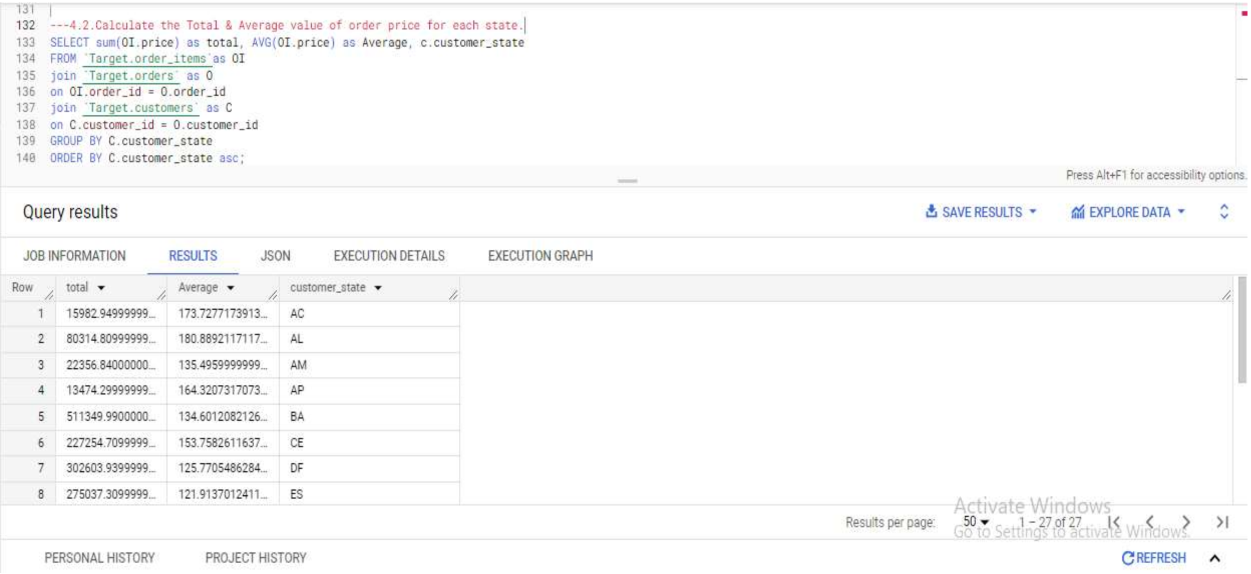


```
---4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others---
---4.1 Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
You can use the "payment_value" column in the payments table to get the cost of orders.----

WITH
 yearly_revenue AS (
 SELECT
   EXTRACT (year
   FROM
     o.order_purchase_timestamp) AS year,
   SUM(p.payment_value) AS revenue,
  FROM
  `Target.orders` AS o
  JOIN
  `Target.payments` AS p
  ON
  o.order_id = p.order_id
  WHERE
  o.order_status = 'delivered'
    AND EXTRACT (month
    FROM
      o.order_purchase_timestamp) BETWEEN 0
    AND 8
  GROUP BY
    year),
  prev_year_revenue AS (
  SELECT
  *
```



```
  GROUP BY
    year),
  prev_year_revenue AS (
  SELECT
  *,
  LAG(revenue) OVER (ORDER BY year ASC) AS prev_revenue
  FROM
   yearly_revenue)
  SELECT
  *,
   (revenue-prev_revenue)/prev_revenue*100 AS per_increase
  FROM
   prev_year_revenue;
```

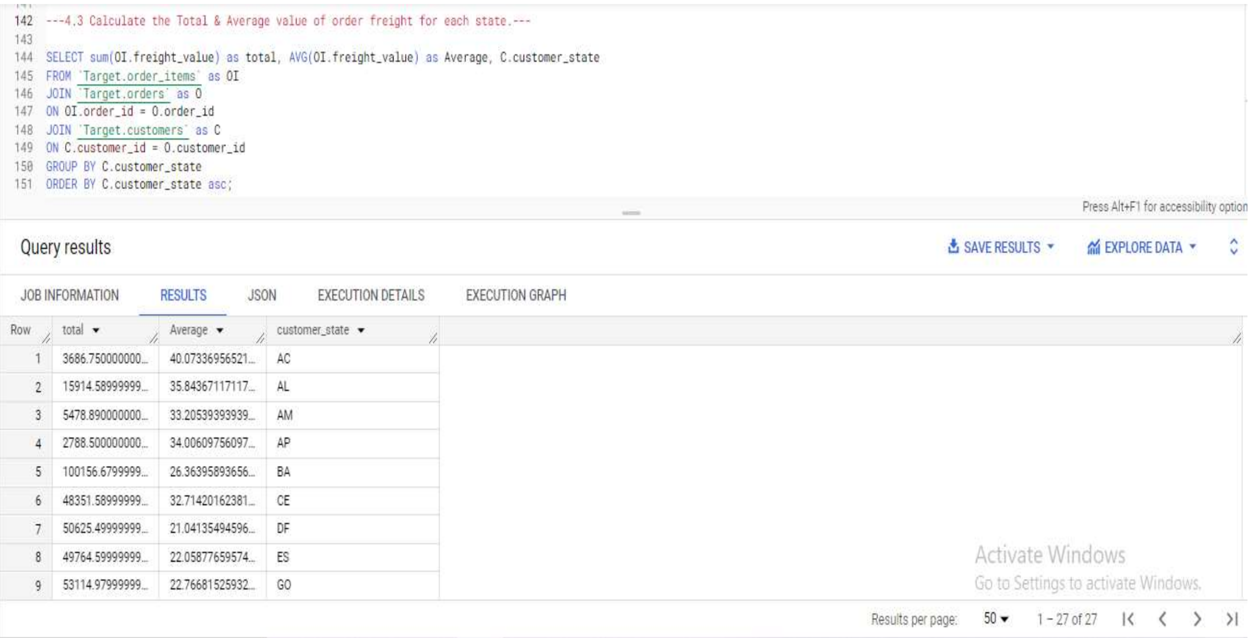
Row	year	revenue	prev_revenue	per_increase
1	2018	8452975.199999...	3473862.760000...	143.3307181081...
2	2017	3473862.760000...	null	null



```sql
---4.2.Calculate the Total & Average value of order price for each state.
SELECT sum(OI.price) as total, AVG(OI.price) as Average, c.customer_state
FROM `Target.order_items`as OI
join `Target.orders` as O
on OI.order_id = O.order_id
join `Target.customers` as C
on C.customer_id = O.customer_id
GROUP BY C.customer_state
ORDER BY C.customer_state asc;
```

Query results

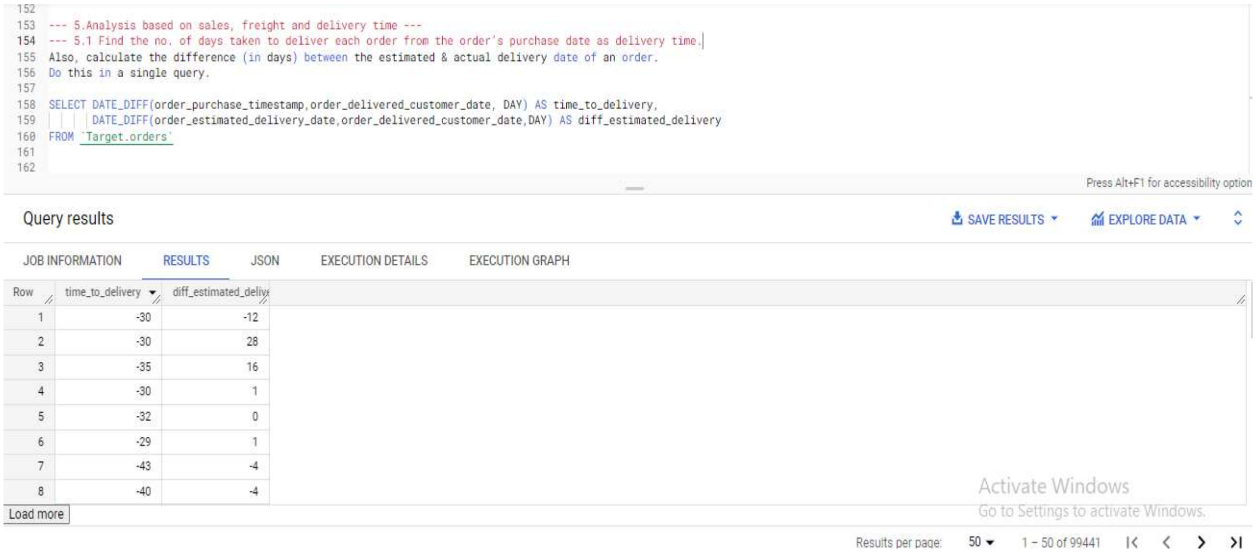
Row	total	Average	customer_state
1	15982.94999999...	173.7277173913...	AC
2	80314.80999999...	180.8892117117...	AL
3	22356.84000000...	135.4959999999...	AM
4	13474.29999999...	164.3207317073...	AP
5	511349.9900000...	134.6012082126...	BA
6	227254.7099999...	153.7582611637...	CE
7	302603.9399999...	125.7705486284...	DF
8	275037.3099999...	121.9137012411...	ES



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

SELECT sum(OI.freight_value) as total, AVG(OI.freight_value) as Average, C.customer_state
FROM `Target.order_items` as OI
JOIN `Target.orders` as O
ON OI.order_id = O.order_id
JOIN `Target.customers` as C
ON C.customer_id = O.customer_id
GROUP BY C.customer_state
ORDER BY C.customer_state asc;
```

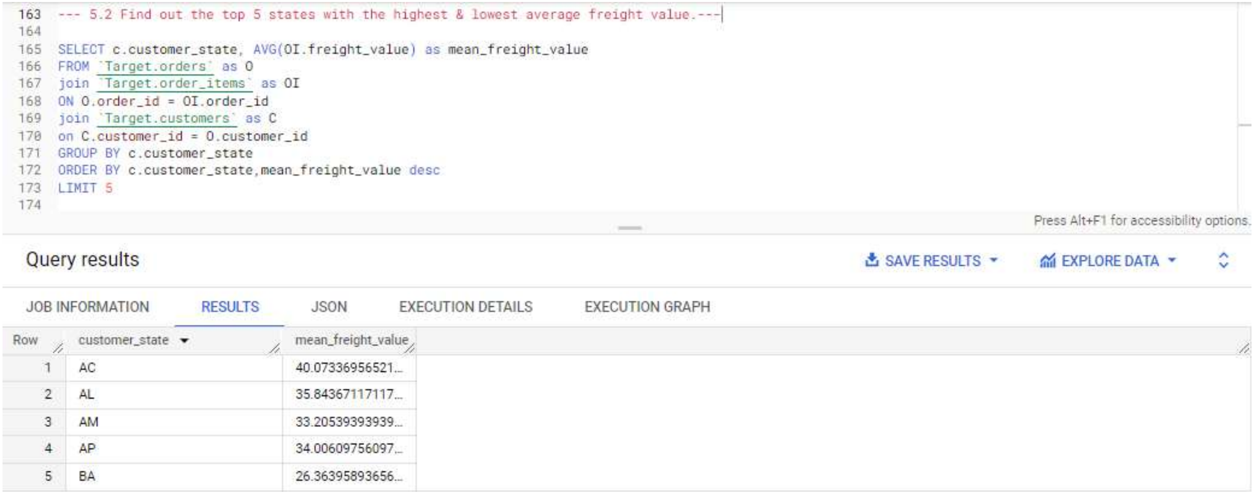
Query results

Row	total	Average	customer_state
1	3686.750000000...	40.07336956521...	AC
2	15914.58999999...	35.84367117117...	AL
3	5478.890000000...	33.20539393939...	AM
4	2788.500000000...	34.00609756097...	AP
5	100156.6799999...	26.36395893656...	BA
6	48351.58999999...	32.71420162381...	CE
7	50625.49999999...	21.04135494596...	DF
8	49764.59999999...	22.05877659574...	ES
9	53114.97999999...	22.76681525932...	GO



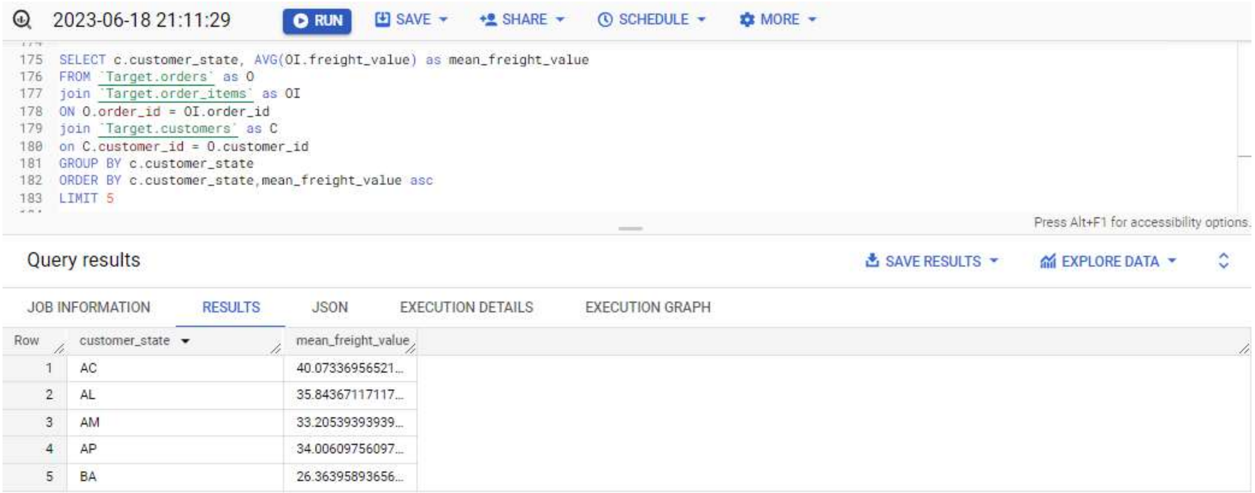
```sql
--- 5.Analysis based on sales, freight and delivery time ---
--- 5.1 Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
Do this in a single query.

SELECT DATE_DIFF(order_purchase_timestamp,order_delivered_customer_date, DAY) AS time_to_delivery,
       DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS diff_estimated_delivery
FROM `Target.orders`
```

Row	time_to_delivery	diff_estimated_deliv
1	-30	-12
2	-30	28
3	-35	16
4	-30	1
5	-32	0
6	-29	1
7	-43	-4
8	-40	-4



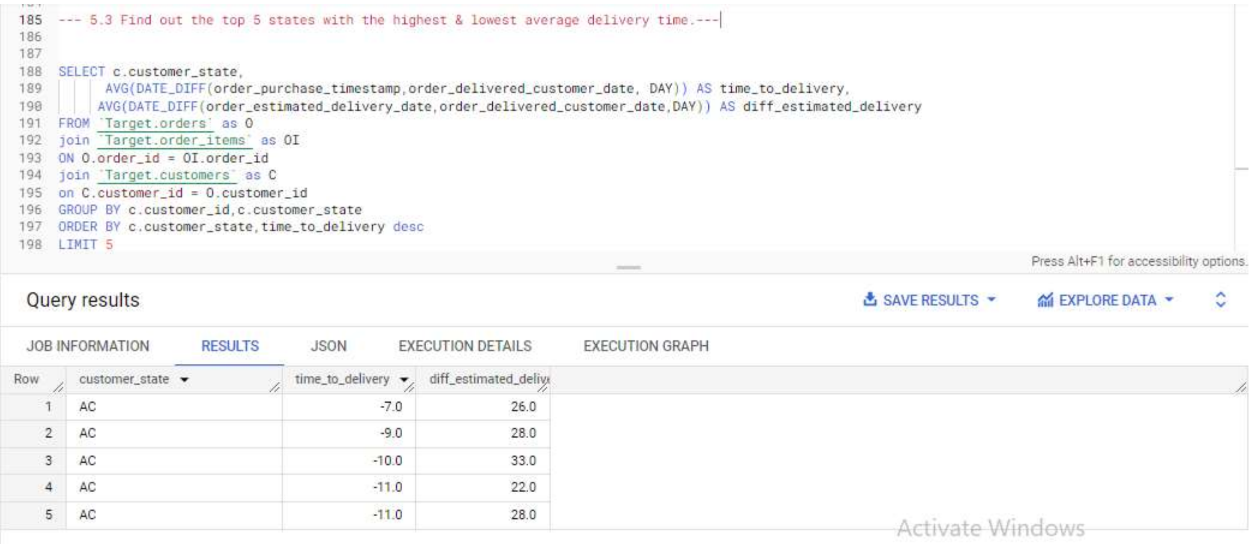
```sql
--- 5.2 Find out the top 5 states with the highest & lowest average freight value.---

SELECT c.customer_state, AVG(OI.freight_value) as mean_freight_value
FROM `Target.orders` as O
join `Target.order_items` as OI
ON O.order_id = OI.order_id
join `Target.customers` as C
on C.customer_id = O.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state,mean_freight_value desc
LIMIT 5
```

Row	customer_state	mean_freight_value
1	AC	40.07336956521...
2	AL	35.84367117117...
3	AM	33.20539393939...
4	AP	34.00609756097...
5	BA	26.36395893656...



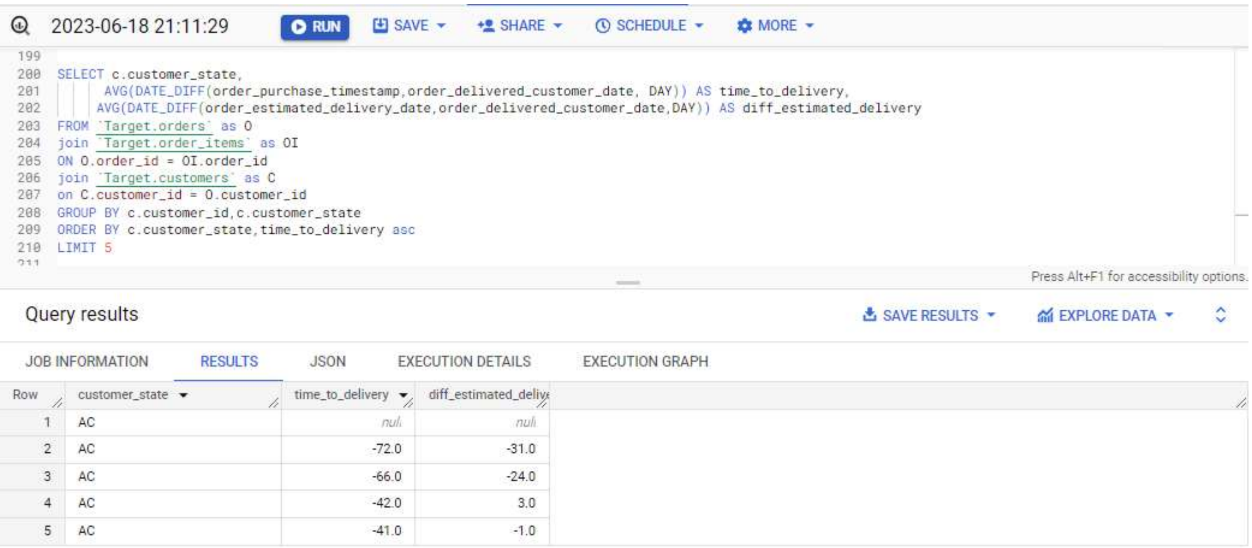
```sql
SELECT c.customer_state, AVG(OI.freight_value) as mean_freight_value
FROM `Target.orders` as O
join `Target.order_items` as OI
ON O.order_id = OI.order_id
join `Target.customers` as C
on C.customer_id = O.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state,mean_freight_value asc
LIMIT 5
```

Row	customer_state	mean_freight_value
1	AC	40.07336956521...
2	AL	35.84367117117...
3	AM	33.20539393939...
4	AP	34.00609756097...
5	BA	26.36395893656...



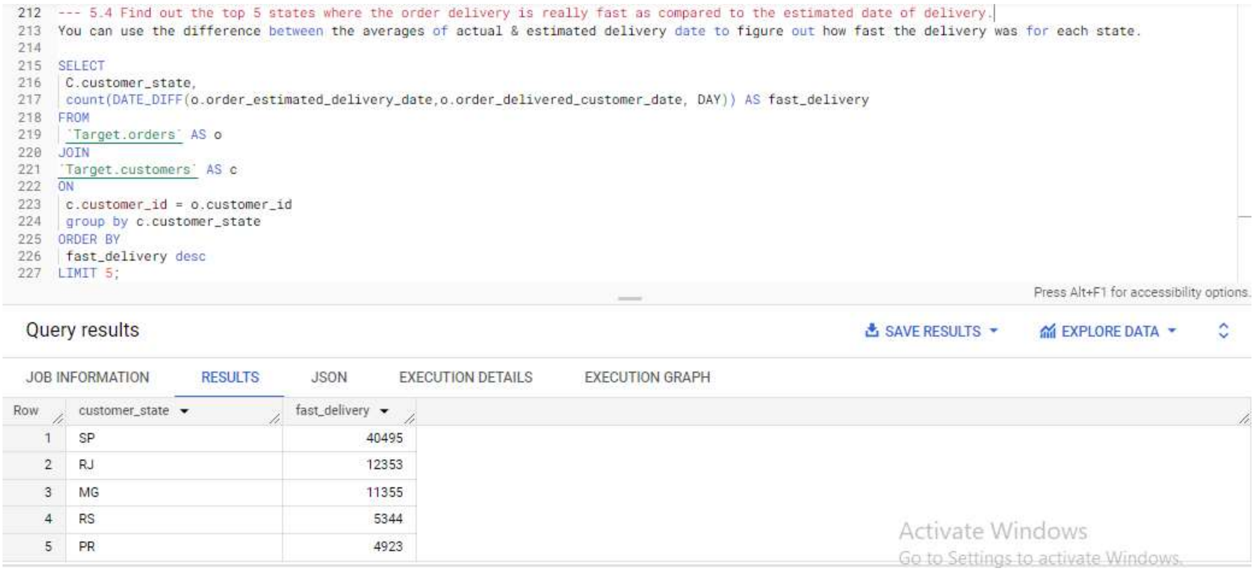
```sql
--- 5.3 Find out the top 5 states with the highest & lowest average delivery time.---


SELECT c.customer_state,
       AVG(DATE_DIFF(order_purchase_timestamp,order_delivered_customer_date, DAY)) AS time_to_delivery,
     AVG(DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)) AS diff_estimated_delivery
FROM `Target.orders` as O
join `Target.order_items` as OI
ON O.order_id = OI.order_id
join `Target.customers` as C
on C.customer_id = O.customer_id
GROUP BY c.customer_id,c.customer_state
ORDER BY c.customer_state,time_to_delivery desc
LIMIT 5
```

Row	customer_state	time_to_delivery	diff_estimated_deliv
1	AC	-7.0	26.0
2	AC	-9.0	28.0
3	AC	-10.0	33.0
4	AC	-11.0	22.0
5	AC	-11.0	28.0



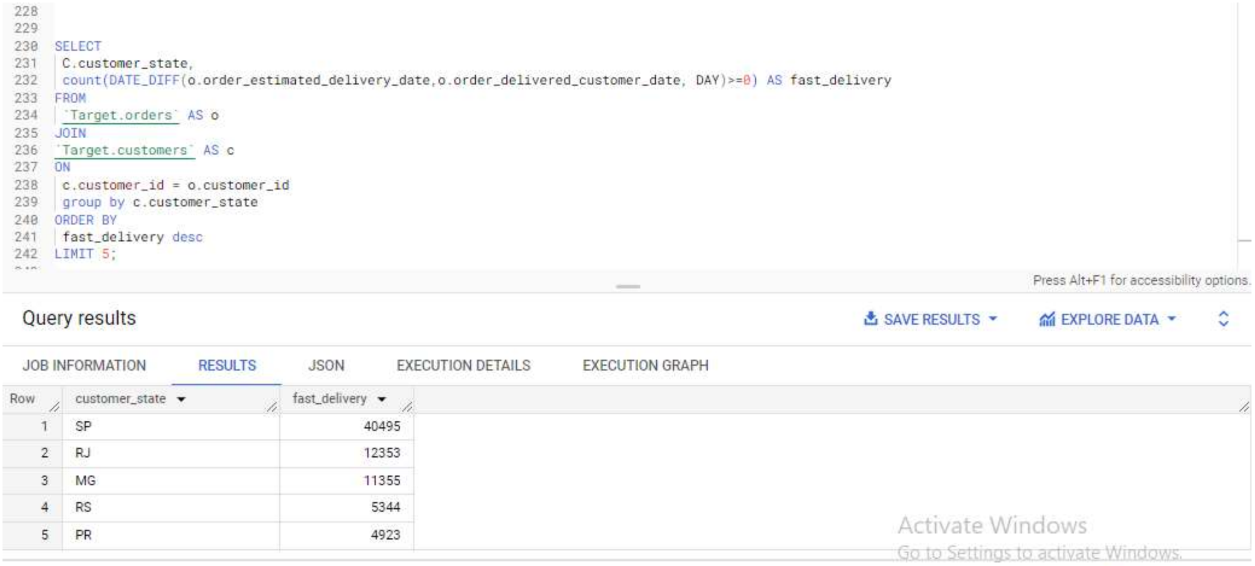
```sql
SELECT c.customer_state,
       AVG(DATE_DIFF(order_purchase_timestamp,order_delivered_customer_date, DAY)) AS time_to_delivery,
     AVG(DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)) AS diff_estimated_delivery
FROM `Target.orders` as O
join `Target.order_items` as OI
ON O.order_id = OI.order_id
join `Target.customers` as C
on C.customer_id = O.customer_id
GROUP BY c.customer_id,c.customer_state
ORDER BY c.customer_state,time_to_delivery asc
LIMIT 5
```

Row	customer_state	time_to_delivery	diff_estimated_deliv
1	AC	nul	nul
2	AC	-72.0	-31.0
3	AC	-66.0	-24.0
4	AC	-42.0	3.0
5	AC	-41.0	-1.0



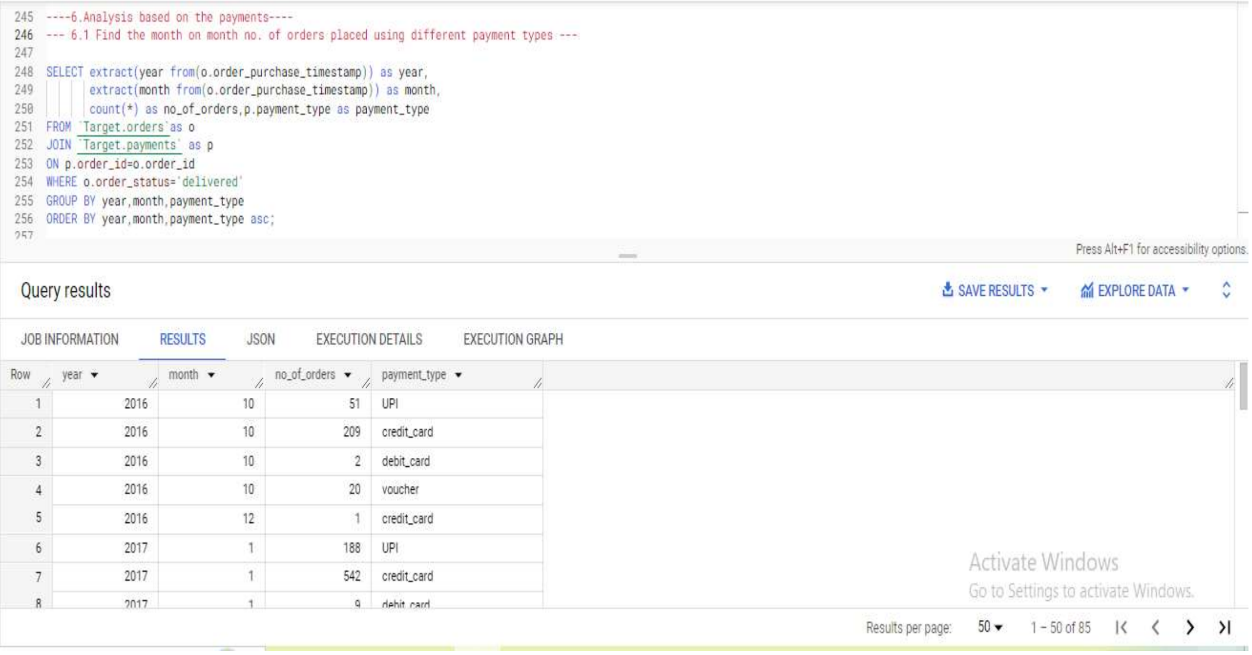
```
--- 5.4 Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

SELECT
 C.customer_state,
 count(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)) AS fast_delivery
FROM
 `Target.orders` AS o
JOIN
`Target.customers` AS c
ON
 c.customer_id = o.customer_id
 group by c.customer_state
ORDER BY
 fast_delivery desc
LIMIT 5;
```

Row	customer_state	fast_delivery
1	SP	40495
2	RJ	12353
3	MG	11355
4	RS	5344
5	PR	4923



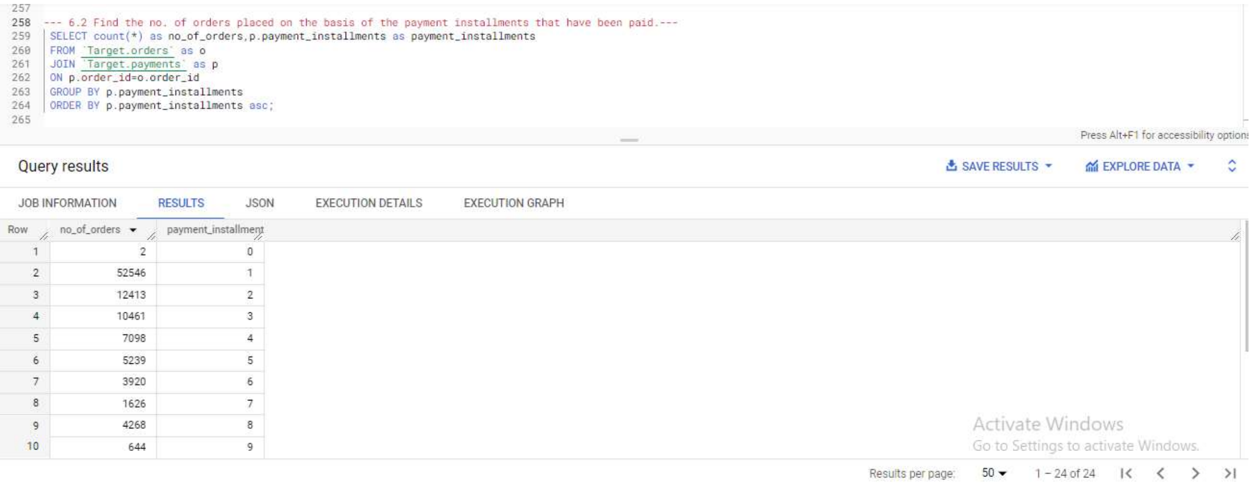
```
SELECT
 C.customer_state,
 count(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)>=0) AS fast_delivery
FROM
 `Target.orders` AS o
JOIN
`Target.customers` AS c
ON
 c.customer_id = o.customer_id
 group by c.customer_state
ORDER BY
 fast_delivery desc
LIMIT 5;
```

Row	customer_state	fast_delivery
1	SP	40495
2	RJ	12353
3	MG	11355
4	RS	5344
5	PR	4923



```
----6.Analysis based on the payments----
--- 6.1 Find the month on month no. of orders placed using different payment types ---

SELECT extract(year from(o.order_purchase_timestamp)) as year,
       extract(month from(o.order_purchase_timestamp)) as month,
       count(*) as no_of_orders,p.payment_type as payment_type
FROM `Target.orders`as o
JOIN `Target.payments` as p
ON p.order_id=o.order_id
WHERE o.order_status='delivered'
GROUP BY year,month,payment_type
ORDER BY year,month,payment_type asc;
```

Row	year	month	no_of_orders	payment_type
1	2016	10	51	UPI
2	2016	10	209	credit_card
3	2016	10	2	debit_card
4	2016	10	20	voucher
5	2016	12	1	credit_card
6	2017	1	188	UPI
7	2017	1	542	credit_card
8	2017	1	9	debit_card



```
--- 6.2 Find the no. of orders placed on the basis of the payment installments that have been paid.---
SELECT count(*) as no_of_orders,p.payment_installments as payment_installments
FROM `Target.orders` as o
JOIN `Target.payments` as p
ON p.order_id=o.order_id
GROUP BY p.payment_installments
ORDER BY p.payment_installments asc;
```

Row	no_of_orders	payment_installment
1	2	0
2	52546	1
3	12413	2
4	10461	3
5	7098	4
6	5239	5
7	3920	6
8	1626	7
9	4268	8
10	644	9