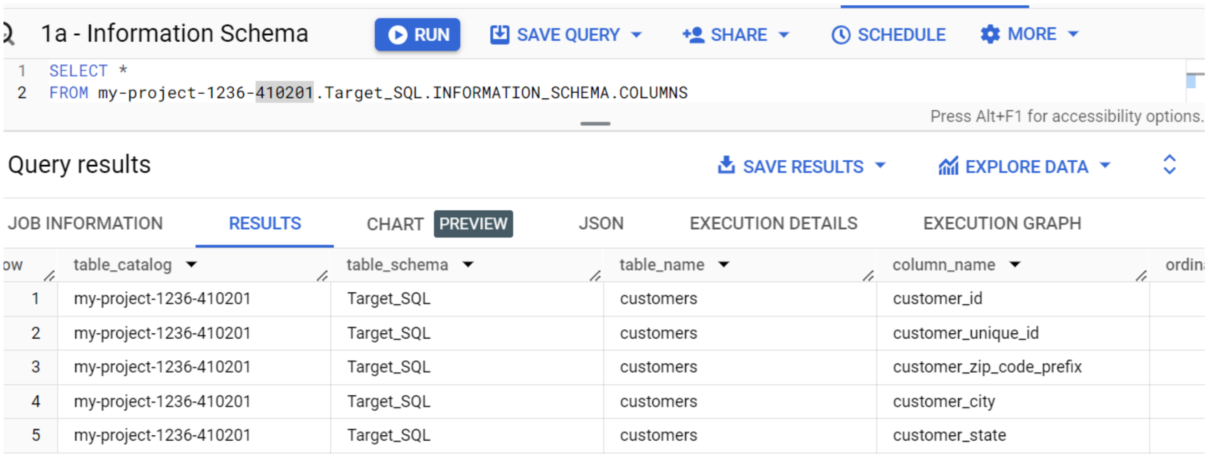
Business Case study: TARGET SQL

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.**

```
SELECT *
FROM My-project-1236-410201.Target_SQL.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers'
```



1a - Information Schema

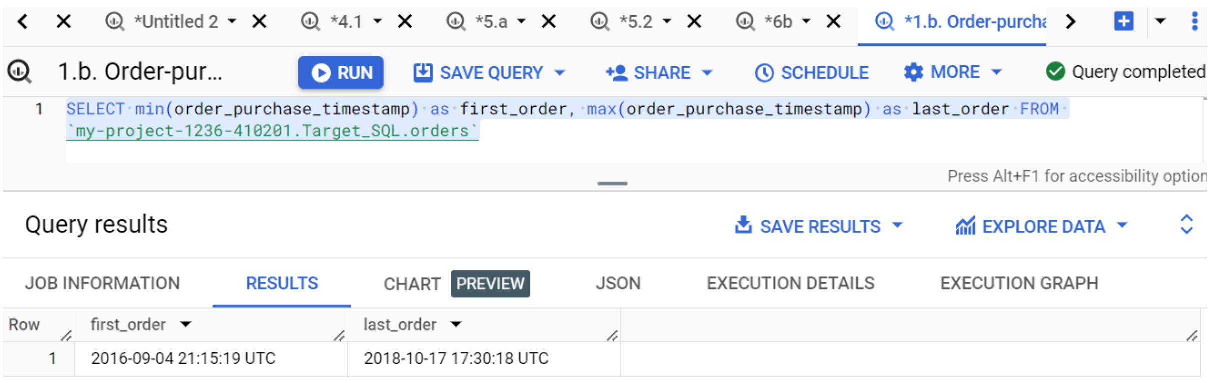
```
SELECT *
FROM my-project-1236-410201.Target_SQL.INFORMATION_SCHEMA.COLUMNS
```

Query results

ow	table_catalog	table_schema	table_name	column_name	ordin
1	my-project-1236-410201	Target_SQL	customers	customer_id	
2	my-project-1236-410201	Target_SQL	customers	customer_unique_id	
3	my-project-1236-410201	Target_SQL	customers	customer_zip_code_prefix	
4	my-project-1236-410201	Target_SQL	customers	customer_city	
5	my-project-1236-410201	Target_SQL	customers	customer_state	

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

```
SELECT
    min(order_purchase_timestamp) as first_order,
    max(order_purchase_timestamp) as last_order
FROM `My-project-1236-410201.Target_SQL.orders`
```



1.b. Order-pur...

```
SELECT min(order_purchase_timestamp) as first_order, max(order_purchase_timestamp) as last_order FROM
`my-project-1236-410201.Target_SQL.orders`
```

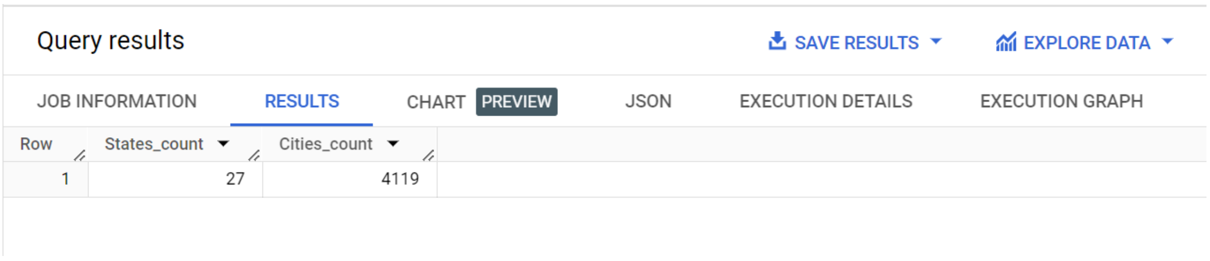
Query completed

Query results

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

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

```
SELECT
    count(distinct C.customer_state) as States_count,
    count(distinct C.customer_city) as Cities_count
FROM my-project-1236-410201.Target_SQL.orders O
INNER JOIN my-project-1236-410201.Target_SQL.customers C ON C.customer_id =
O.customer_id
```


Query results

SAVE RESULTS | EXPLORE DATA

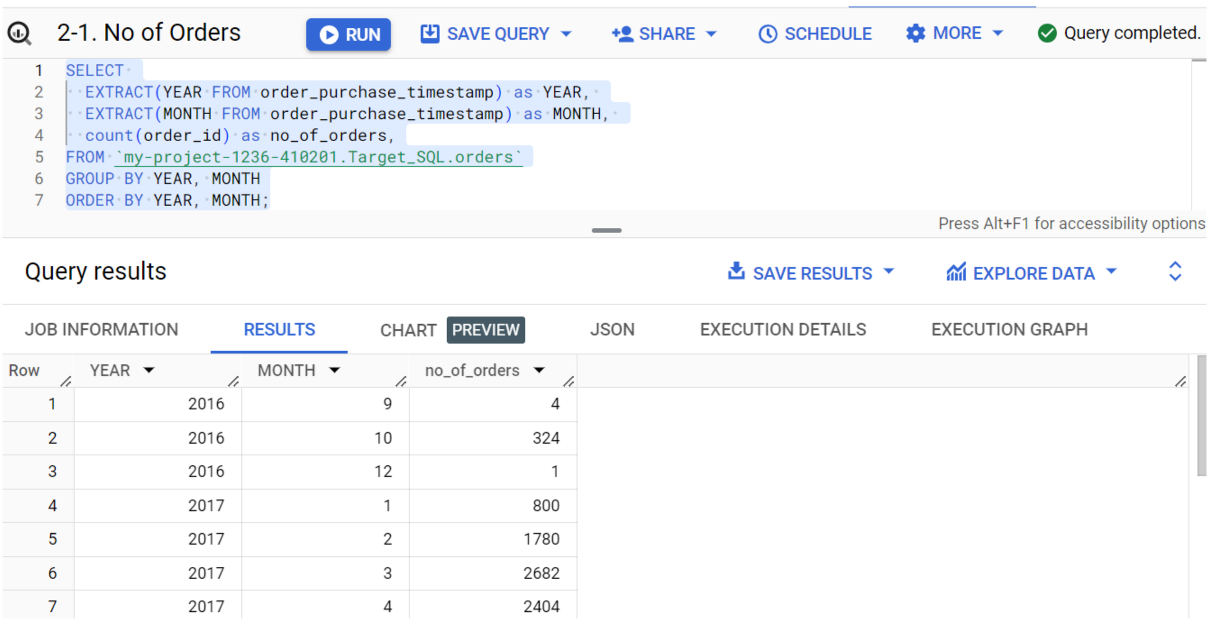
JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	States_count	Cities_count
1	27	4119

2. In-depth Exploration

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

```
SELECT
    EXTRACT(YEAR FROM order_purchase_timestamp) as YEAR,
    EXTRACT(MONTH FROM order_purchase_timestamp) as MONTH,
    count(order_id),
FROM `My-project-1236-410201.Target_SQL.orders`
GROUP BY YEAR, MONTH
ORDER BY YEAR, MONTH;
```


2-1. No of Orders

RUN | SAVE QUERY | SHARE | SCHEDULE | MORE | Query completed.

```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) as YEAR,
  EXTRACT(MONTH FROM order_purchase_timestamp) as MONTH,
  count(order_id) as no_of_orders,
FROM `my-project-1236-410201.Target_SQL.orders`
GROUP BY YEAR, MONTH
ORDER BY YEAR, MONTH;
```

Press Alt+F1 for accessibility options

Query results

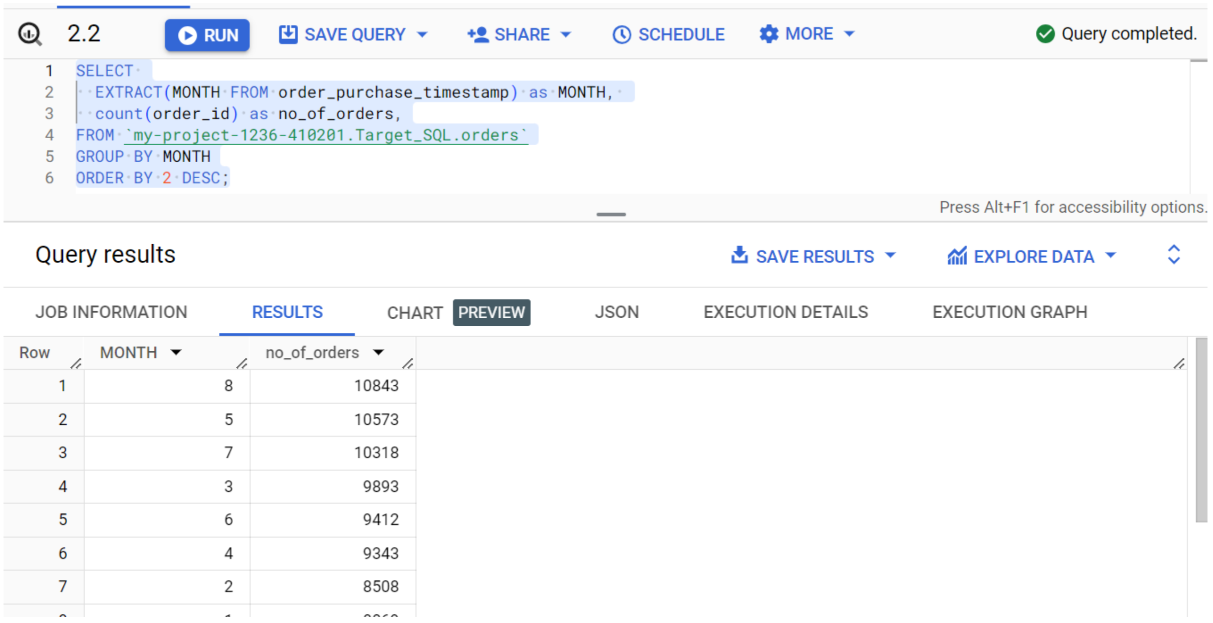
SAVE RESULTS | EXPLORE DATA

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	YEAR	MONTH	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

B. Can we see some kind of monthly seasonality in terms of the no. of orders being placed? Maximum orders are placed in May, July, August and March months. Minimum orders are placed in last four months of the year – Sep, Oct, Nov and Dec.

```
SELECT
    EXTRACT(MONTH FROM order_purchase_timestamp) as MONTH,
    count(order_id) as no_of_orders,
FROM `My-project-1236-410201.Target_SQL.orders`
GROUP BY MONTH
ORDER BY 2 DESC;
```

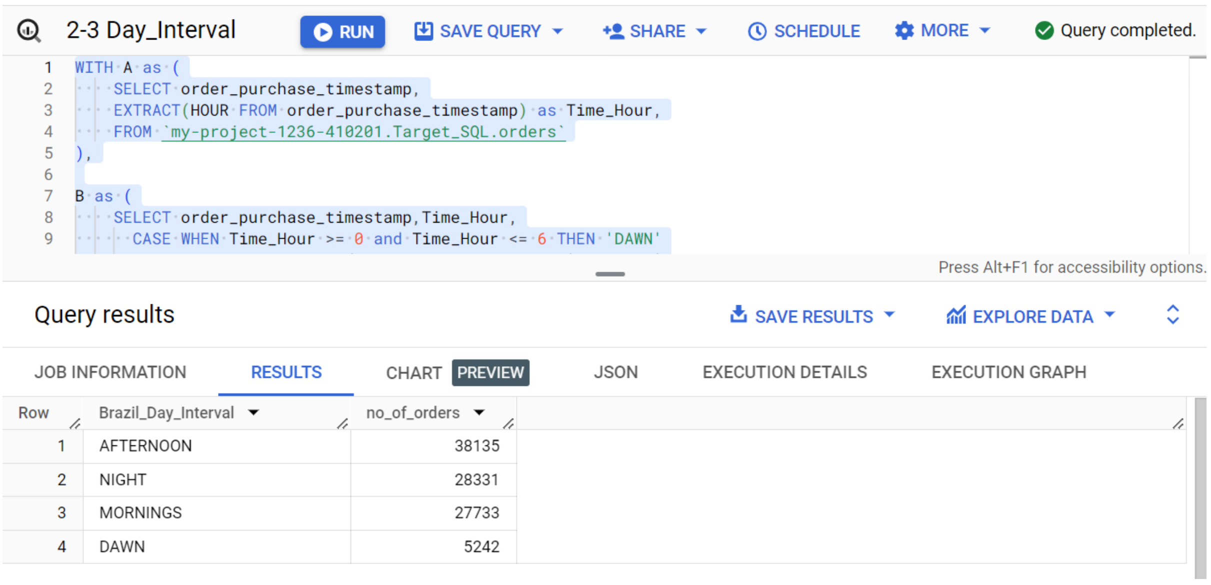


```
SELECT
  EXTRACT(MONTH FROM order_purchase_timestamp) as MONTH,
  count(order_id) as no_of_orders,
FROM `my-project-1236-410201.Target_SQL.orders`
GROUP BY MONTH
ORDER BY 2 DESC;
```

Row	MONTH	no_of_orders
1	8	10843
2	5	10573
3	7	10318
4	3	9893
5	6	9412
6	4	9343
7	2	8508

c. **During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night) –** Maximum orders are placed during Afternoon and night.

```
WITH A as (
    SELECT order_purchase_timestamp,
           EXTRACT(HOUR FROM order_purchase_timestamp) as Time_Hour,
     FROM `My-project-1236-410201.Target_SQL.orders`
),

B as (
    SELECT order_purchase_timestamp,Time_Hour,
           CASE WHEN Time_Hour >= 0 and Time_Hour <= 6 THEN 'DAWN'
           WHEN Time_Hour >= 7 and Time_Hour <= 12 THEN 'MORNINGS'
           WHEN Time_Hour >= 13 and Time_Hour <= 18 THEN 'AFTERNOON'
           ELSE 'NIGHT' END as Brazil_Day_Interval
   FROM A
)

SELECT
  Brazil_Day_Interval,
  count(Time_Hour) as no_of_orders
FROM B
GROUP BY 1
ORDER BY 2 DESC
```


2-3 Day_Interval

```
WITH A as (
    SELECT order_purchase_timestamp,
    EXTRACT(HOUR FROM order_purchase_timestamp) as Time_Hour,
    FROM `my-project-1236-410201.Target_SQL.orders`
),

B as (
    SELECT order_purchase_timestamp,Time_Hour,
      CASE WHEN Time_Hour >= 0 and Time_Hour <= 6 THEN 'DAWN'
```

Query completed.

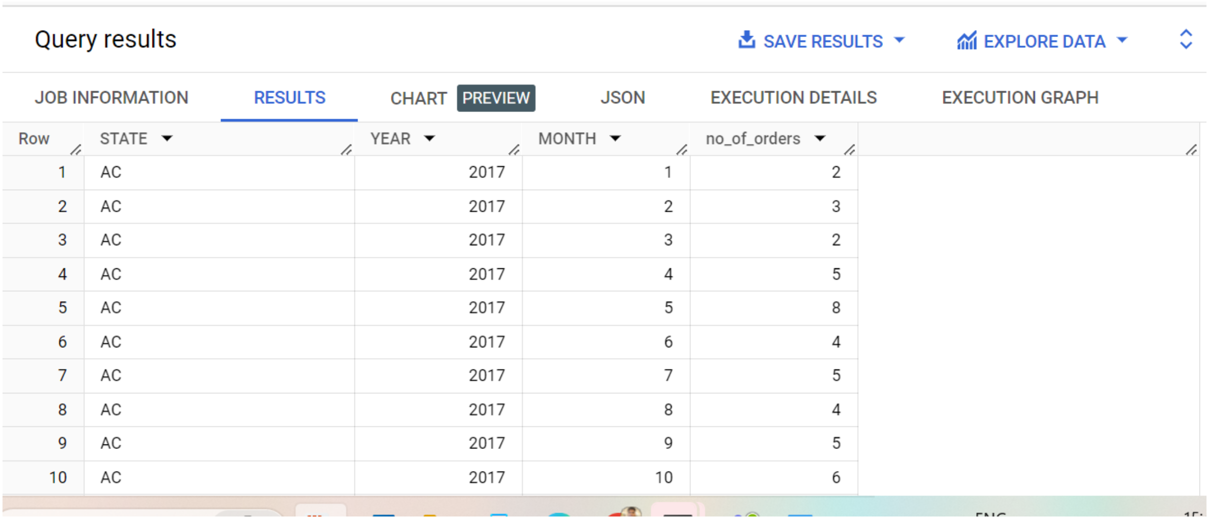
Query results

Row	Brazil_Day_Interval	no_of_orders
1	AFTERNOON	38135
2	NIGHT	28331
3	MORNINGS	27733
4	DAWN	5242

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

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

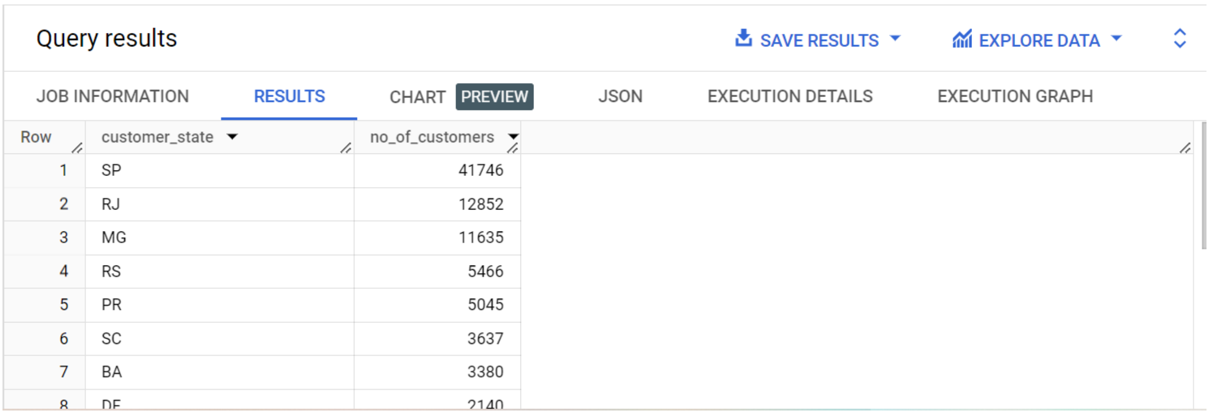
```
SELECT
    B.customer_state as STATE,
    EXTRACT(YEAR FROM A.order_purchase_timestamp) as YEAR,
    EXTRACT(MONTH FROM A.order_purchase_timestamp) as MONTH,
    count(A.order_purchase_timestamp) as no_of_orders
FROM `My-project-1236-410201.Target_SQL.orders` A
LEFT JOIN My-project-1236-410201.Target_SQL.customers B ON B.customer_id =
A.customer_id
GROUP BY STATE,YEAR,MONTH
ORDER BY STATE,YEAR,MONTH;
```


Query results

Row	STATE	YEAR	MONTH	no_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

B. How are the customers distributed across all the states? States SP has maximum customers and RR as minimum customers.

```
SELECT customer_state,
       count(customer_id) as no_of_customers
FROM `My-project-1236-410201.Target_SQL.customers`
GROUP BY 1
ORDER BY 2 DESC
```



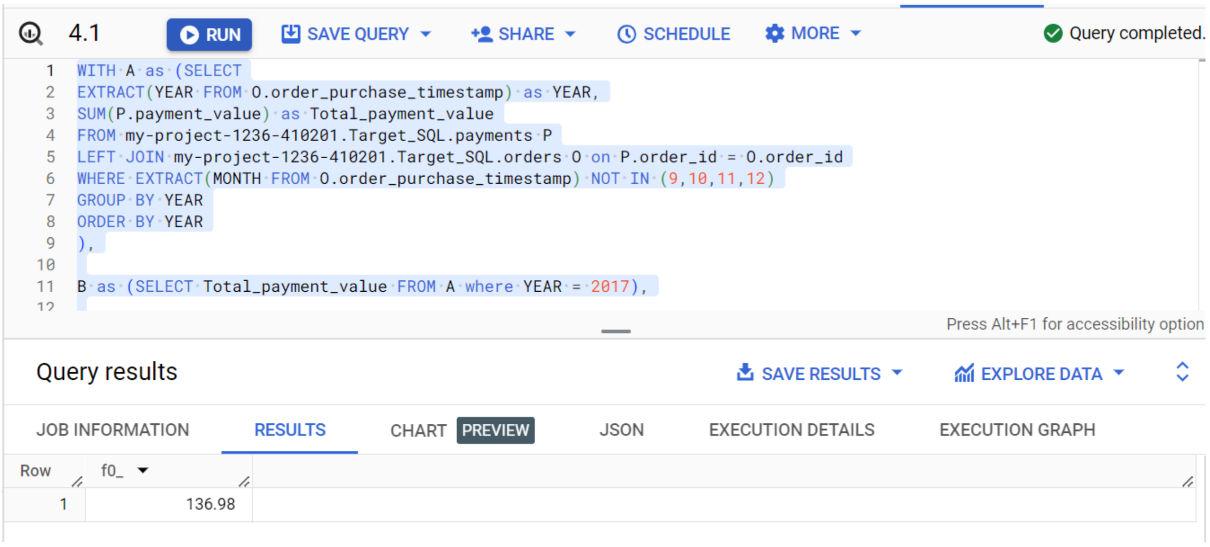
Query results

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

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).** 136.98%

```
WITH A as (SELECT
EXTRACT(YEAR FROM O.order_purchase_timestamp) as YEAR,
SUM(P.payment_value) as Total_payment_value
FROM My-project-1236-410201.Target_SQL.payments P
LEFT JOIN My-project-1236-410201.Target_SQL.orders O on P.order_id = O.order_id
WHERE EXTRACT(MONTH FROM O.order_purchase_timestamp) NOT IN (9,10,11,12)
GROUP BY YEAR
ORDER BY YEAR
),

B as (SELECT Total_payment_value FROM A where YEAR = 2017),

C as (SELECT Total_payment_value FROM A where YEAR = 2018)

SELECT ROUND(((C.Total_payment_value -
B.Total_payment_value)/B.Total_payment_value) * 100,2) FROM B,C;
```



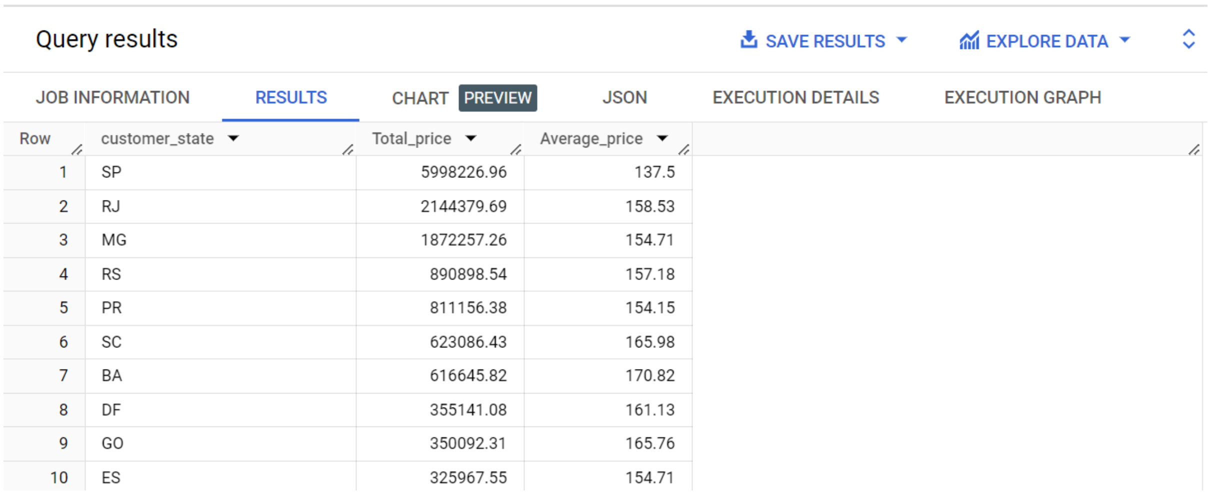
```
WITH A as (SELECT
EXTRACT(YEAR FROM O.order_purchase_timestamp) as YEAR,
SUM(P.payment_value) as Total_payment_value
FROM my-project-1236-410201.Target_SQL.payments P
LEFT JOIN my-project-1236-410201.Target_SQL.orders O on P.order_id = O.order_id
WHERE EXTRACT(MONTH FROM O.order_purchase_timestamp) NOT IN (9,10,11,12)
GROUP BY YEAR
ORDER BY YEAR
),

B as (SELECT Total_payment_value FROM A where YEAR = 2017),
```

Query results

Row	f0_
1	136.98

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

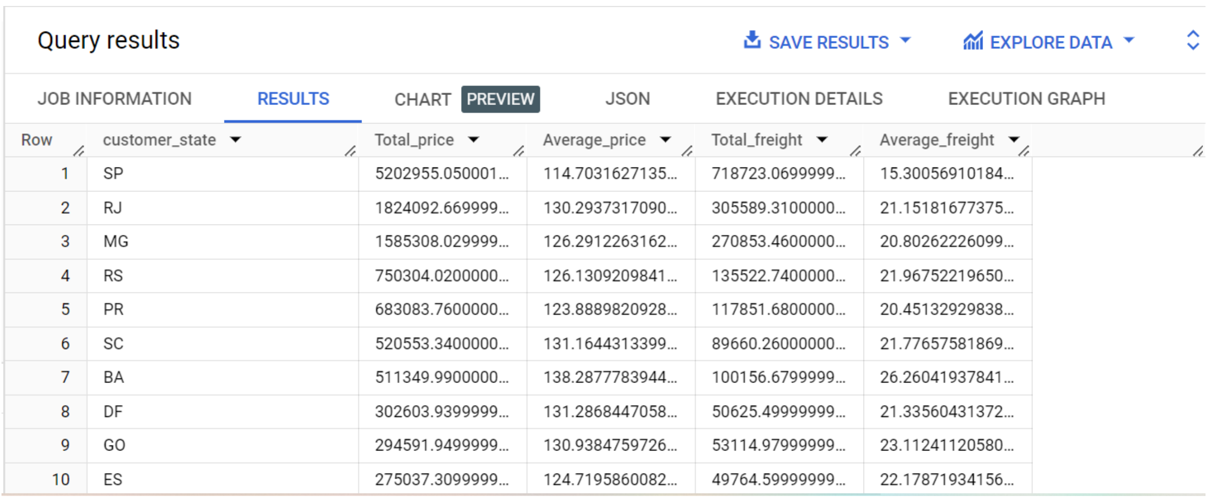
```
SELECT C.customer_state,
    Round(SUM(P.payment_value),2) as Total_price,
    ROUND(AVG(P.payment_value),2) as Average_price
FROM My-project-1236-410201.Target_SQL.customers as C
LEFT JOIN My-project-1236-410201.Target_SQL.orders O on C.customer_id =
O.customer_id
LEFT JOIN My-project-1236-410201.Target_SQL.payments P on O.order_id =
P.order_id
GROUP BY C.customer_state
ORDER BY 2 DESC,3 DESC
```



Row	customer_state	Total_price	Average_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

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

```
WITH A as (
    SELECT
            C.customer_state,
            O.order_id,
            SUM(OT.price) as Total_price,
            AVG(OT.price) as AVG_price,
            AVG(OT.freight_value) as AVG_freight
    FROM My-project-1236-410201.Target_SQL.customers C
    LEFT JOIN My-project-1236-410201.Target_SQL.orders O on C.customer_id =
    O.customer_id
    LEFT JOIN `My-project-1236-410201.Target_SQL.order_items` OT on O.order_id =
    OT.order_id
   GROUP BY 1,2
   ORDER BY 3,4
)

SELECT customer_state,
SUM(Total_price) as Total_price,
AVG(AVG_price) as Average_price,
SUM(Total_freight) as Total_freight,
AVG(AVG_freight) as Average_freight
FROM A
GROUP BY 1
Order by 2 DESC
```

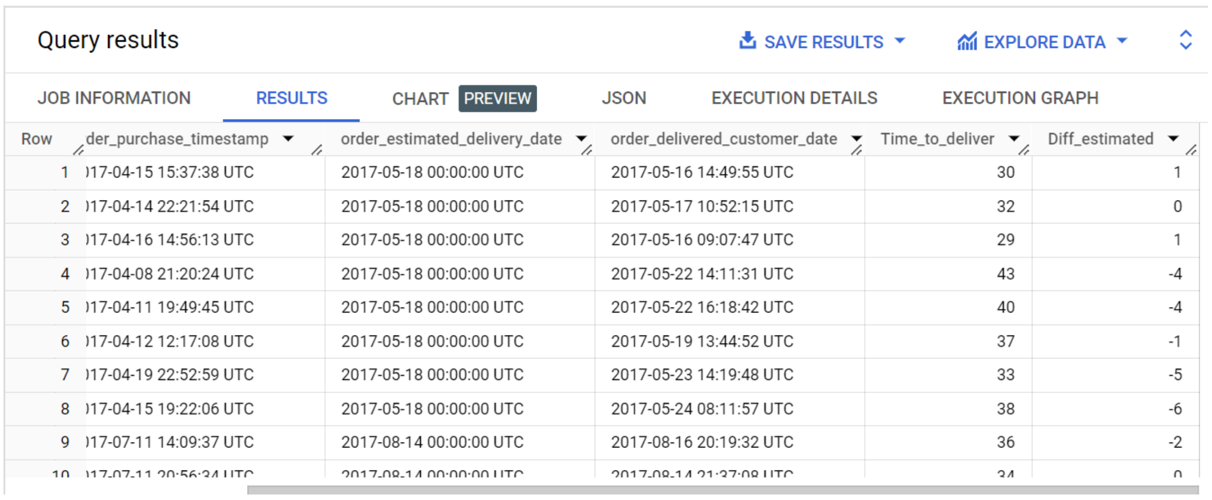


Row	customer_state	Total_price	Average_price	Total_freight	Average_freight
1	SP	5202955.050001...	114.7031627135...	718723.0699999...	15.30056910184...
2	RJ	1824092.669999...	130.2937317090...	305589.3100000...	21.15181677375...
3	MG	1585308.029999...	126.2912263162...	270853.4600000...	20.80262226099...
4	RS	750304.0200000...	126.1309209841...	135522.7400000...	21.96752219650...
5	PR	683083.7600000...	123.8889820928...	117851.6800000...	20.45132929838...
6	SC	520553.3400000...	131.1644313399...	89660.26000000...	21.77657581869...
7	BA	511349.9900000...	138.2877783944...	100156.6799999...	26.26041937841...
8	DF	302603.9399999...	131.2868447058...	50625.49999999...	21.33560431372...
9	GO	294591.9499999...	130.9384759726...	53114.97999999...	23.11241120580...
10	ES	275037.3099999...	124.7195860082...	49764.59999999...	22.17871934156...

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.

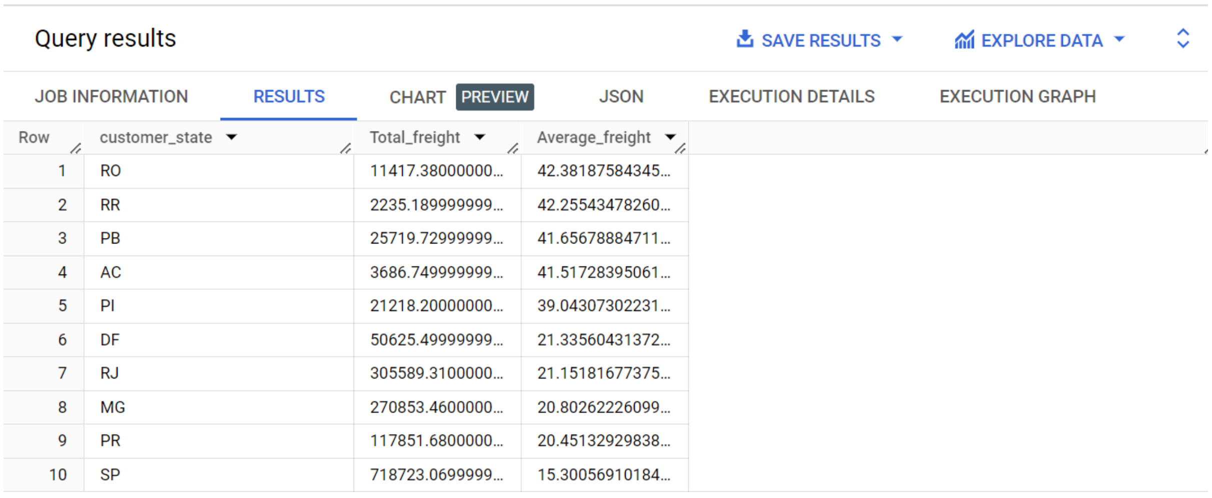
```
SELECT order_id,
order_purchase_timestamp,
order_estimated_delivery_date,
order_delivered_customer_date,
DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) as
Time_to_deliver,
DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)
as Diff_estimated
FROM `My-project-1236-410201.Target_SQL.orders`
WHERE order_status = 'delivered';
```



Row	der_purchase_timestamp	order_estimated_delivery_date	order_delivered_customer_date	Time_to_deliver	Diff_estimated
1	)17-04-15 15:37:38 UTC	2017-05-18 00:00:00 UTC	2017-05-16 14:49:55 UTC	30	1
2	)17-04-14 22:21:54 UTC	2017-05-18 00:00:00 UTC	2017-05-17 10:52:15 UTC	32	0
3	)17-04-16 14:56:13 UTC	2017-05-18 00:00:00 UTC	2017-05-16 09:07:47 UTC	29	1
4	)17-04-08 21:20:24 UTC	2017-05-18 00:00:00 UTC	2017-05-22 14:11:31 UTC	43	-4
5	)17-04-11 19:49:45 UTC	2017-05-18 00:00:00 UTC	2017-05-22 16:18:42 UTC	40	-4
6	)17-04-12 12:17:08 UTC	2017-05-18 00:00:00 UTC	2017-05-19 13:44:52 UTC	37	-1
7	)17-04-19 22:52:59 UTC	2017-05-18 00:00:00 UTC	2017-05-23 14:19:48 UTC	33	-5
8	)17-04-15 19:22:06 UTC	2017-05-18 00:00:00 UTC	2017-05-24 08:11:57 UTC	38	-6
9	)17-07-11 14:09:37 UTC	2017-08-14 00:00:00 UTC	2017-08-16 20:19:32 UTC	36	-2
10	)17-07-11 20:56:34 UTC	2017-08-14 00:00:00 UTC	2017-08-14 21:37:08 UTC	34	0

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

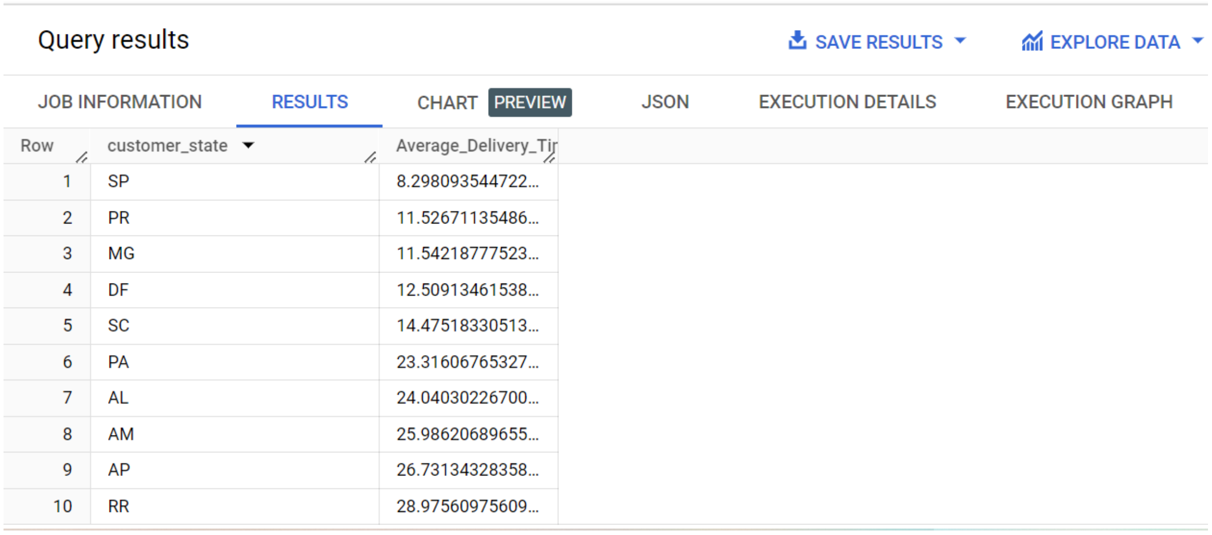
```
WITH A as (
    SELECT C.customer_state,
    O.order_id,
    SUM(OT.freight_value) as Total_freight,
    AVG(OT.freight_value) as AVG_freight
    FROM My-project-1236-410201.Target_SQL.customers C
    LEFT JOIN My-project-1236-410201.Target_SQL.orders O on C.customer_id =
O.customer_id
    LEFT JOIN `My-project-1236-410201.Target_SQL.order_items` OT on O.order_id =
OT.order_id
    GROUP BY 1,2
   ),

TOP_5 as (
      SELECT customer_state,
      SUM(Total_freight) as Total_freight,
      AVG(AVG_freight) as Average_freight
FROM A
GROUP BY 1
Order by 3 DESC
LIMIT 5
),

BOTTOM_5 as (
      SELECT customer_state,
      SUM(Total_freight) as Total_freight,
      AVG(AVG_freight) as Average_freight
FROM A
GROUP BY 1
Order by 3 ASC
LIMIT 5
)

SELECT * FROM TOP_5
UNION ALL
SELECT * FROM BOTTOM_5
ORDER BY Average_freight DESC;
```



Row	customer_state	Total_freight	Average_freight
1	RO	11417.38000000...	42.38187584345...
2	RR	2235.189999999...	42.25543478260...
3	PB	25719.72999999...	41.65678884711...
4	AC	3686.749999999...	41.51728395061...
5	PI	21218.20000000...	39.04307302231...
6	DF	50625.49999999...	21.33560431372...
7	RJ	305589.3100000...	21.15181677375...
8	MG	270853.4600000...	20.80262226099...
9	PR	117851.6800000...	20.45132929838...
10	SP	718723.0699999...	15.30056910184...

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

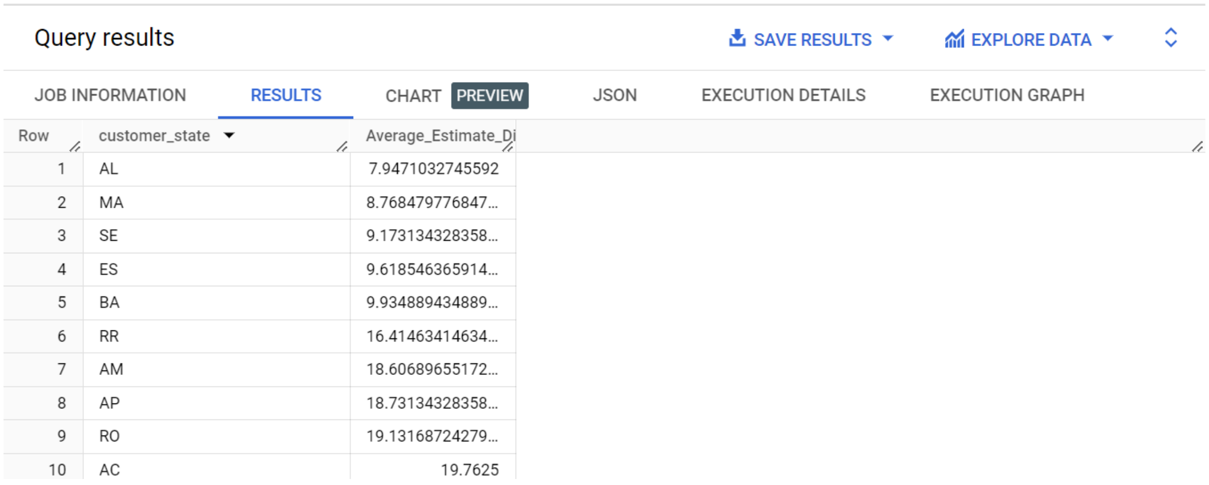
```
WITH A as (SELECT C.customer_state, O.order_id,
O.order_purchase_timestamp,
O.order_estimated_delivery_date,
O.order_delivered_customer_date,
DATE_DIFF(O.order_delivered_customer_date,O.order_purchase_timestamp,DAY) as
Time_to_deliver,
DATE_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date,DAY) as
Diff_estimated
FROM My-project-1236-410201.Target_SQL.customers C
LEFT JOIN My-project-1236-410201.Target_SQL.orders O ON C.customer_id = O.customer_id
WHERE order_status = 'delivered'),

BOTTOM_5 as (SELECT customer_state,
AVG(Time_to_deliver) as Average_Delivery_Time,
FROM A
GROUP BY 1
ORDER BY 2
LIMIT 5
),

TOP_5 as (SELECT customer_state,
AVG(Time_to_deliver) as Average_Delivery_Time,
FROM A
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5
)

SELECT customer_state, Average_Delivery_Time FROM BOTTOM_5
UNION ALL
SELECT customer_state, Average_Delivery_Time FROM TOP_5
ORDER BY 2;
```



Row	customer_state	Average_Delivery_Tir
1	SP	8.298093544722...
2	PR	11.52671135486...
3	MG	11.54218777523...
4	DF	12.50913461538...
5	SC	14.47518330513...
6	PA	23.31606765327...
7	AL	24.04030226700...
8	AM	25.98620689655...
9	AP	26.73134328358...
10	RR	28.97560975609...

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

```
WITH A as (
       SELECT
              C.customer_state,
              O.order_id,
              O.order_purchase_timestamp,
              O.order_estimated_delivery_date,
              O.order_delivered_customer_date,
              DATE_DIFF(O.order_delivered_customer_date,O.order_purchase_timestamp,DAY) as
       Time_to_deliver,
              DATE_DIFF(O.order_estimated_delivery_date,O.order_delivered_customer_date,DA
       Y) as Diff_estimated
       FROM My-project-1236-410201.Target_SQL.customers C
       LEFT JOIN My-project-1236-410201.Target_SQL.orders O ON C.customer_id =
   O.customer_id
       WHERE order_status = 'delivered'),

   BOTTOM_5 as (
       SELECT customer_state,
              AVG(A.Diff_estimated) as Average_Estimate_Diff
       FROM A
       GROUP BY 1
       ORDER BY 2
       LIMIT 5
   ),

   TOP_5 as (
       SELECT customer_state,
              AVG(A.Diff_estimated) as Average_Estimate_Diff
       FROM A
       GROUP BY 1
       ORDER BY 2 DESC
       LIMIT 5
   )

   SELECT customer_state, Average_Estimate_Diff FROM BOTTOM_5
   UNION ALL
   SELECT customer_state, Average_Estimate_Diff FROM TOP_5
   order by 2;
```



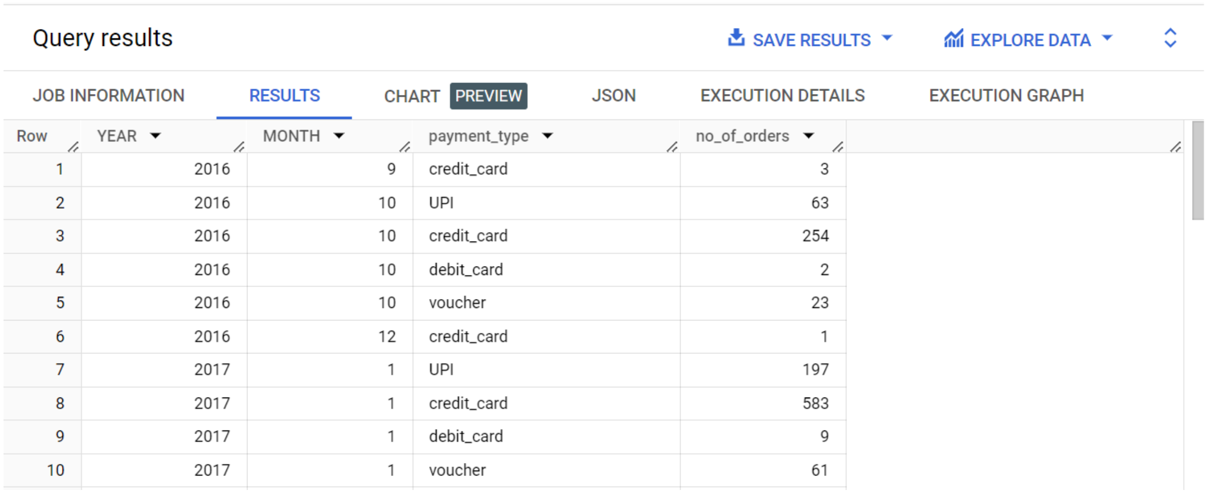
Row	customer_state	Average_Estimate_Di
1	AL	7.9471032745592
2	MA	8.768479776847...
3	SE	9.173134328358...
4	ES	9.618546365914...
5	BA	9.934889434889...
6	RR	16.41463414634...
7	AM	18.60689655172...
8	AP	18.73134328358...
9	RO	19.13168724279...
10	AC	19.7625

6. Analysis based on the payments:

A. 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,
        P.payment_type,
        count(P.order_id) as no_of_orders
```

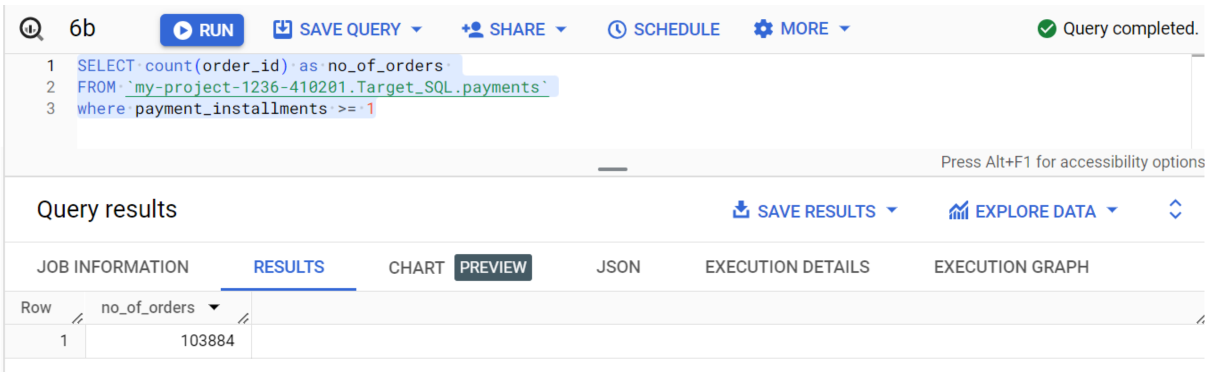
```
FROM `My-project-1236-410201.Target_SQL.orders` O
RIGHT JOIN My-project-1236-410201.Target_SQL.payments P ON P.order_id = O.order_id
GROUP BY 1,2,3
ORDER BY 1,2,3
```



Row	YEAR	MONTH	payment_type	no_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

B. **Find the no. of orders placed on the basis of the payment installments that have been paid. 103884**

```
SELECT count(order_id) as no_of_orders
FROM `My-project-1236-410201.Target_SQL.payments`
WHERE payment_installments >= 1
```



```
SELECT count(order_id) as no_of_orders
FROM `my-project-1236-410201.Target_SQL.payments`
where payment_installments >= 1
```

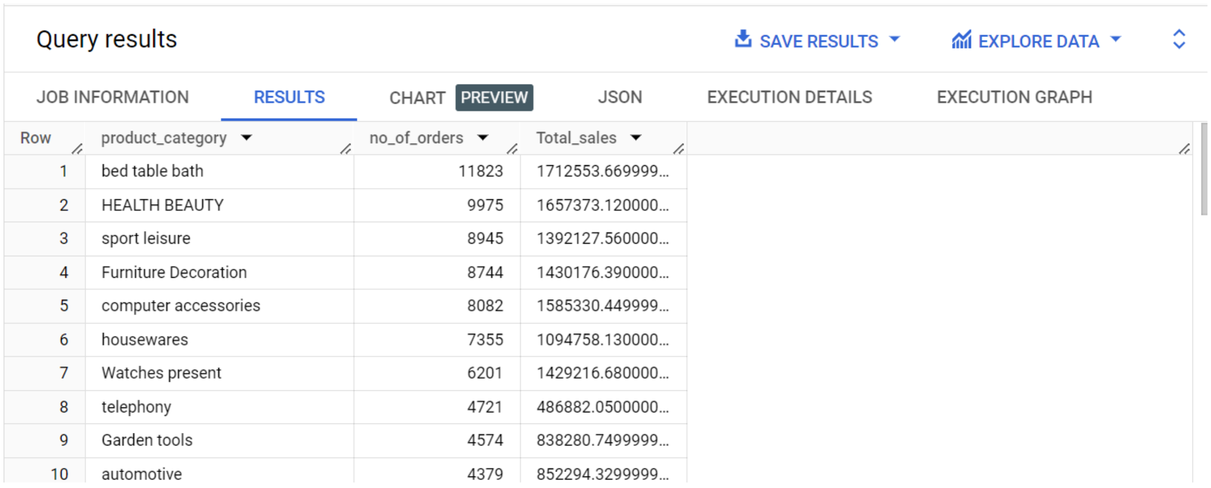
Row	no_of_orders
1	103884

Additional Insights :

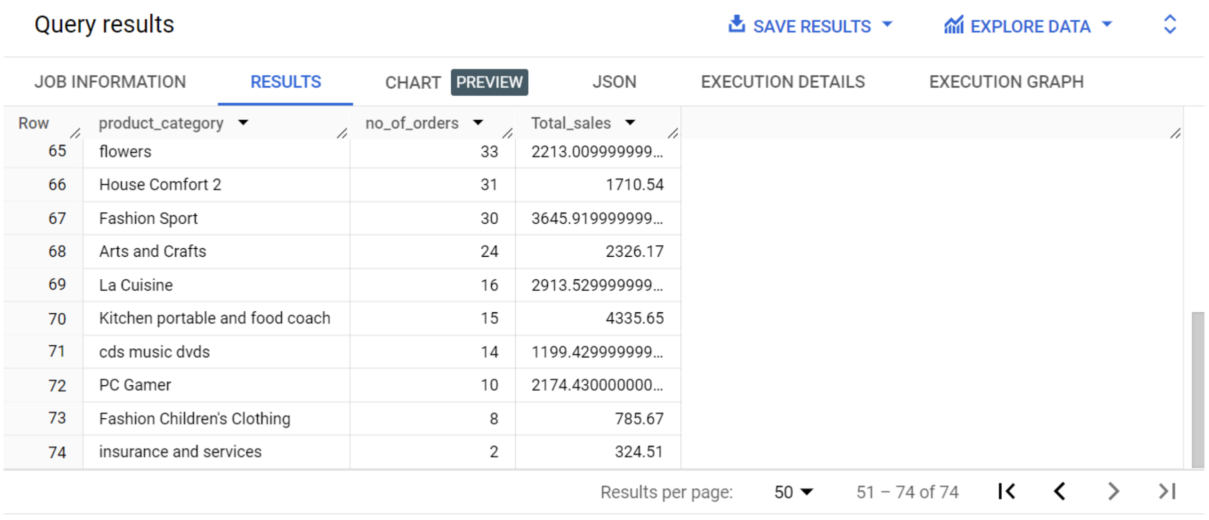
1. **Identify the Product categories having maximum and minimum orders along with total Payment Value**

```
WITH A as (
  SELECT P.product_id, P.product_category, OT.order_id, payment_value FROM `my-project-
1236-410201.Target_SQL.products` P
LEFT JOIN my-project-1236-410201.Target_SQL.order_items OT ON P.product_id = OT.product_id
LEFT JOIN my-project-1236-410201.Target_SQL.payments PAY ON OT.order_id = PAY.order_id
)
```

```
SELECT product_category, count(order_id) as no_of_orders, SUM(payment_value) as Total_sales
FROM A
GROUP BY 1
ORDER BY 2 DESC,3 DESC
```



Row	product_category	no_of_orders	Total_sales
1	bed table bath	11823	1712553.669999...
2	HEALTH BEAUTY	9975	1657373.120000...
3	sport leisure	8945	1392127.560000...
4	Furniture Decoration	8744	1430176.390000...
5	computer accessories	8082	1585330.449999...
6	housewares	7355	1094758.130000...
7	Watches present	6201	1429216.680000...
8	telephony	4721	486882.0500000...
9	Garden tools	4574	838280.7499999...
10	automotive	4379	852294.3299999...



Row	product_category	no_of_orders	Total_sales
65	flowers	33	2213.009999999...
66	House Comfort 2	31	1710.54
67	Fashion Sport	30	3645.919999999...
68	Arts and Crafts	24	2326.17
69	La Cuisine	16	2913.529999999...
70	Kitchen portable and food coach	15	4335.65
71	cds music dvds	14	1199.429999999...
72	PC Gamer	10	2174.430000000...
73	Fashion Children's Clothing	8	785.67
74	insurance and services	2	324.51