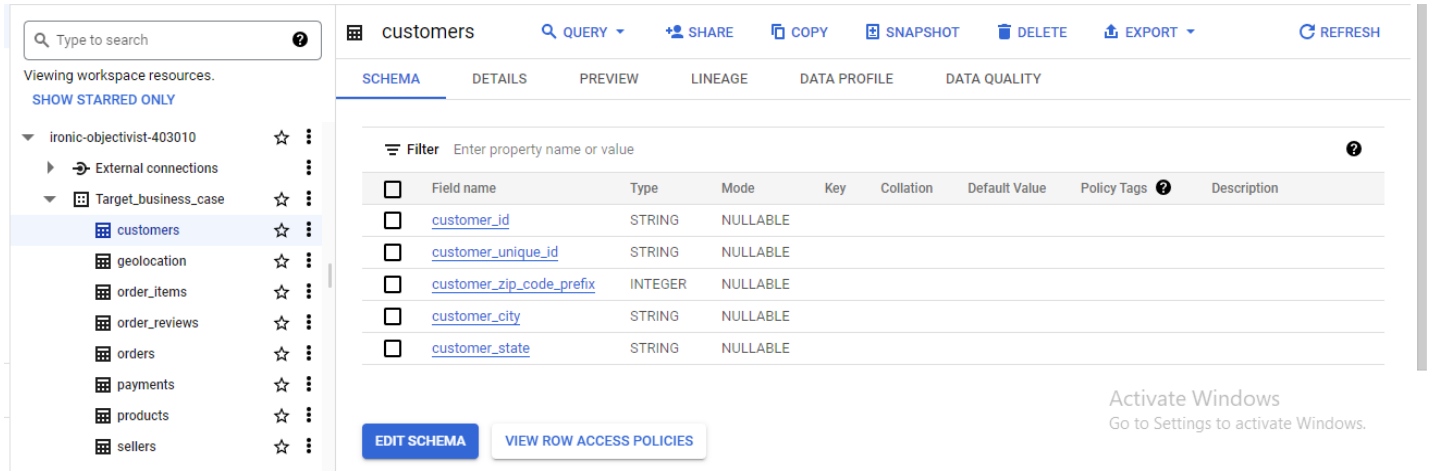
Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

What does 'good' look like?

1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**
    - **i. Data type of all columns in the "customers" table.**

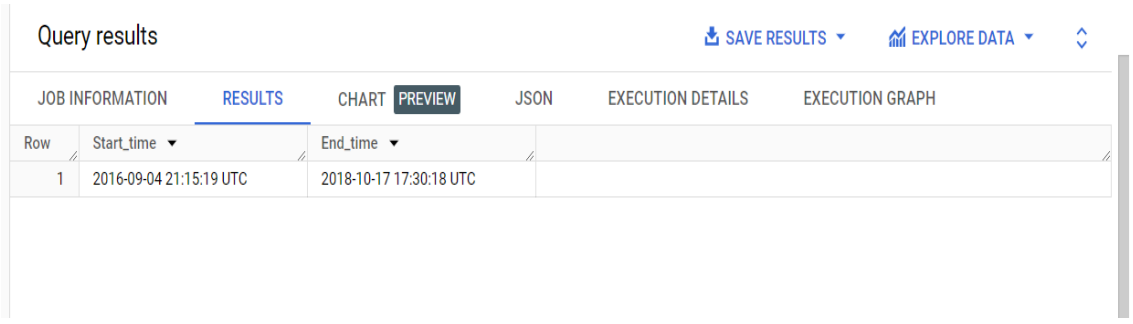
OUTPUT:



- **ii. Get the time range between which the orders were placed.**

QUERY:

```
select
min(order_purchase_timestamp) as Start_time,
max(order_purchase_timestamp) as End_time
from `Target_business_case.orders`
```

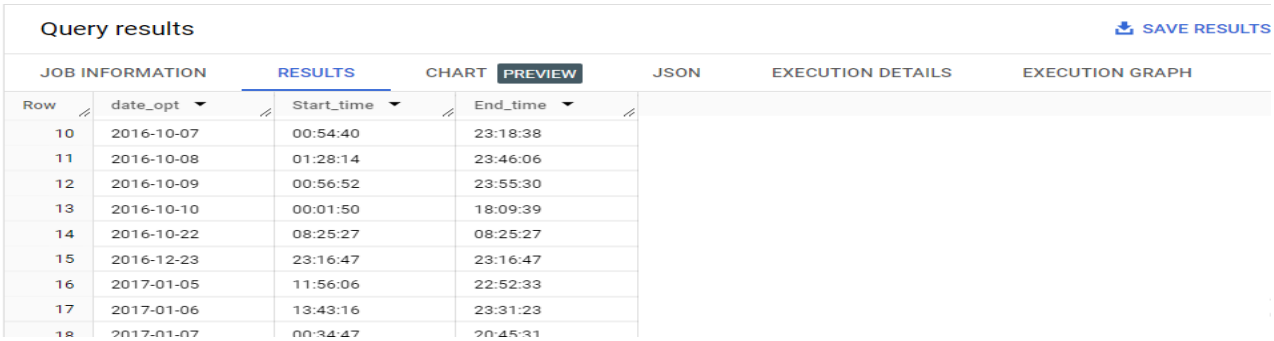


The above query gives the time range for the entire data set i.e., it starts from September 2016 and ends on mid-October 2018.
Now to get the time of order placed for each date we give the following query-

QUERY:

```
select
date_opt,
min(time_opt) as Start_time,
max(time_opt) as End_time
from
(select
order_purchase_timestamp,
extract(date from order_purchase_timestamp) as date_opt,
extract(time from order_purchase_timestamp at time zone "UTC") as time_opt
from `Target_business_case.orders`)
group by date_opt
order by date_opt
```

OUTPUT:



Row	date_opt	Start_time	End_time
10	2016-10-07	00:54:40	23:18:38
11	2016-10-08	01:28:14	23:46:06
12	2016-10-09	00:56:52	23:55:30
13	2016-10-10	00:01:50	18:09:39
14	2016-10-22	08:25:27	08:25:27
15	2016-12-23	23:16:47	23:16:47
16	2017-01-05	11:56:06	22:52:33
17	2017-01-06	13:43:16	23:31:23
18	2017-01-07	00:34:47	20:45:31

INSIGHTS:

When we look at the output and observe the start time and end time, we can observe that on most of the days the order purchase time span of customers is approximately 24 hours.

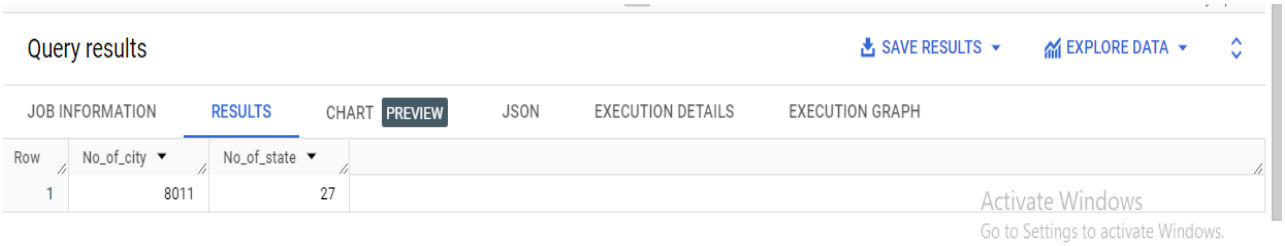
iii. Count the Cities & States of customers who ordered during the given period.

iv.

To get some insight into customers' orders we need to first count the total number of Cities and States in our dataset.

```
SELECT
COUNT (distinct geolocation_city) as Total_no_of_city,
COUNT (distinct geolocation_state) as Total_no_of_state
FROM `Target_business_case.geolocation`
```

OUTPUT:



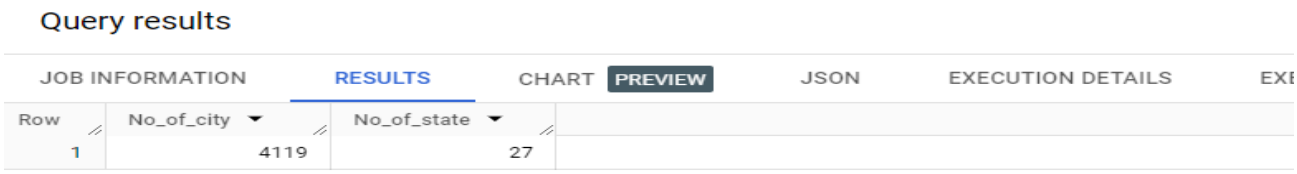
Row	No_of_city	No_of_state
1	8011	27

Now we have to count the number of Cities & States of customers who ordered during the given period.

QUERY:

```
select
count(distinct customer_city) as No_of_city,
count(distinct customer_state) as No_of_state
from `Target_business_case.customers`
```

OUTPUT:



Insight:

The number of cities from customers table is less than the count in geolocations which indicates lesser customer reach in cities.

RECOMMENDATIONS:

For increasing the reach of the company, there are some managerial and marketing steps that needs to be taken.

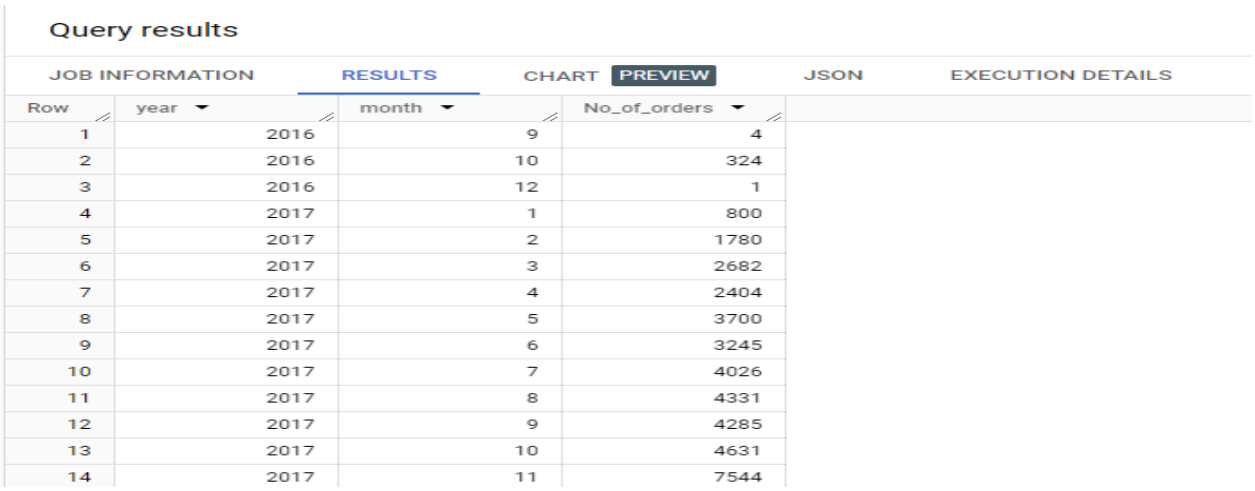
2. **In-depth Exploration:**

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

QUERY:

```
SELECT
year,
month,
COUNT (order_id) as No_of_orders
FROM
(
SELECT
order_id,
order_purchase_timestamp,
Extract (year from order_purchase_timestamp ) as year,
Extract (month from order_purchase_timestamp ) as month
from `Target_business_case.orders`
)
GROUP BY year,month
ORDER BY year, month
```

OUTPUT:



Query results

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

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
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026
11	2017	8	4331
12	2017	9	4285
13	2017	10	4631
14	2017	11	7544

Row year month No_of_orders Load more

12	2017	9	4285
13	2017	10	4631
14	2017	11	7544
15	2017	12	5673
16	2018	1	7269
17	2018	2	6728
18	2018	3	7211
19	2018	4	6939
20	2018	5	6873
21	2018	6	6167
22	2018	7	6292
23	2018	8	6512
24	2018	9	16
25	2018	10	4

The above query has shown the result of month wise no. of orders placed by the customers from SEPT 2016 to OCT 2018 which will be further used for deep insights on orders placed.
To get a insight of yearly sales we will use the following query-

QUERY :

```
SELECT
year,
COUNT(order_id) as No_of_orders
FROM
(
SELECT
order_id,
```

```
order_purchase_timestamp,
Extract(year from order_purchase_timestamp ) as year,
from `Target_business_case.orders`
)
GROUP BY year
ORDER BY year
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW

Row	year	No_of_orders
1	2016	329
2	2017	45101
3	2018	54011

Insight:

There is an overall increase in the no. of orders over the years and approximately **9k** new orders are placed in the year 2018 but this trend is not on continuous spectrum. Winters of 2017 has more no. of orders placed compared to that of 2018.

RECOMMENDATIONS:

Though the number of orders has increased on year-on-year basis, but we can still focus on the months where orders no. has shown a dip and can give special offers and discounts on products for increasing the customer orders.

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

QUERY:

```
SELECT
month,
COUNT (order_id) as No_of_orders
FROM
(SELECT
order_id,
order_purchase_timestamp,
Extract (month from order_purchase_timestamp ) as month
from `Target_business_case.orders`)
group by month
order by month
```

OUTPUT :

Query results

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

Row	month	No_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

Insight:

There is a seasonality present in the patterns of order being placed. The maximum numbers of orders are being placed from the month of May to August. There is dip in the orders after September and again at increases slowly after October.

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

i. 0-6 hrs : Dawn
ii. 7-12 hrs : Mornings
iii. 13-18 hrs : Afternoon
iv. 19-23 hrs : Night

QUERY:

```
SELECT
CASE WHEN (time_hour between 0 and 6) then ("Dawn")
     WHEN (time_hour between 7 and 12) then ("Mornings")
     WHEN (time_hour between 13 and 18) then ("Afternoon")
     else ("Night") end as Shifts,
count (order_id) as no_of_orders
FROM
(
SELECT
order_id,
order_purchase_timestamp,
Extract (hour from order_purchase_timestamp) as time_hour
FROM `Target_business_case.orders`
order by order_purchase_timestamp desc, time_hour
)
group by Shifts
ORDER BY no_of_orders DESC
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION

Row	Shifts	no_of_orders
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

INSIGHTS:

Afternoon has the maximum no of orders followed by Night and least is in the dawn.

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

QUERY:

```
SELECT
c.customer_state,
Extract (year from order_purchase_timestamp ) as year,
Extract(month from o.order_purchase_timestamp) as t_month,
count(o.order_id) as no_of_orders
FROM `Target_business_case.customers` as c
left join `Target_business_case.orders` as o
on c.customer_id= o.customer_id
group by year,c.customer_state, t_month
```

OUTPUT:

Query results

SAVE RESULTS | EXPLORE

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

Row	customer_state	year	t_month	no_of_orders
1	RN	2018	1	46
2	RN	2017	12	30
3	RN	2017	5	17
4	CE	2018	2	88
5	CE	2018	3	98
6	CE	2017	5	62
7	CE	2017	4	43
8	CE	2018	5	74
9	RS	2018	3	418
10	RS	2018	6	305
11	SC	2017	8	159
12	SC	2017	12	193
13	SP	2018	5	3207
14	SP	2018	1	3052

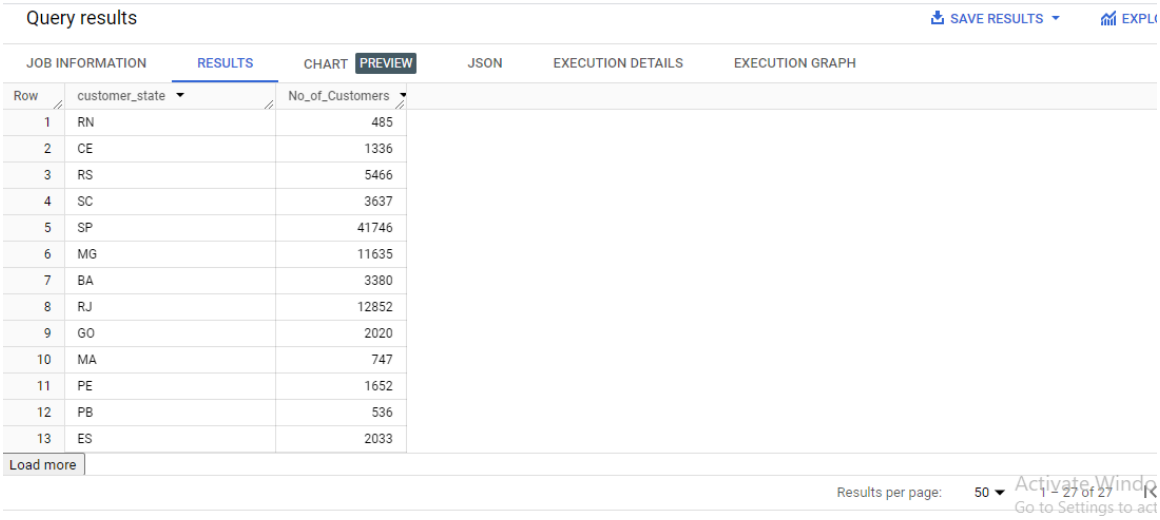
Results per page: 50 1 – 50 of 565

   b. **How are the customers distributed across all the states?**

QUERY:

```
SELECT
customer_state,
Count(customer_id) No_of_Customers
FROM `Target_business_case.customers`
group by customer_state
```

OUTPUT:



Query results

SAVE RESULTS | EXPL

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

Row	customer_state	No_of_Customers
1	RN	485
2	CE	1336
3	RS	5466
4	SC	3637
5	SP	41746
6	MG	11635
7	BA	3380
8	RJ	12852
9	GO	2020
10	MA	747
11	PE	1652
12	PB	536
13	ES	2033

Load more

Results per page: 50 1 – 27 of 27

INSIGHT:

The customer distribution across different states tells us about our business approach at local levels.

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).**
    You can use the "payment_value" column in the payments table to get the cost of orders.

QUERY:

```
SELECT*,
round (((cost_of_orders-prev_year_cost)/prev_year_cost)*100,2) as
percentage_increase
FROM
(
SELECT *,
lag (cost_of_orders,1) over (order by o_year) as prev_year_cost
FROM
(
SELECT
o_year,
sum (payment_value) as cost_of_orders
FROM
(
SELECT
```

```
Extract (year from od.order_purchase_timestamp ) as o_year,
Extract (month from od.order_purchase_timestamp) as t_month,
payment_value
FROM `Target_business_case.orders` as od
left join `Target_business_case.payments` as py
on od.order_id=py.order_id
)
where t_month BETWEEN 01 and 08
Group by o_year
HAVING o_year =2017 OR o_year=2018
)
)
ORDER BY o_year DESC
LIMIT 1
```

OUTPUT:

Query results

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

Row	o_year	cost_of_orders	prev_year_cost	percentage_increase
1	2018	8694733.839999...	3669022.119999...	136.98

INSIGHT:

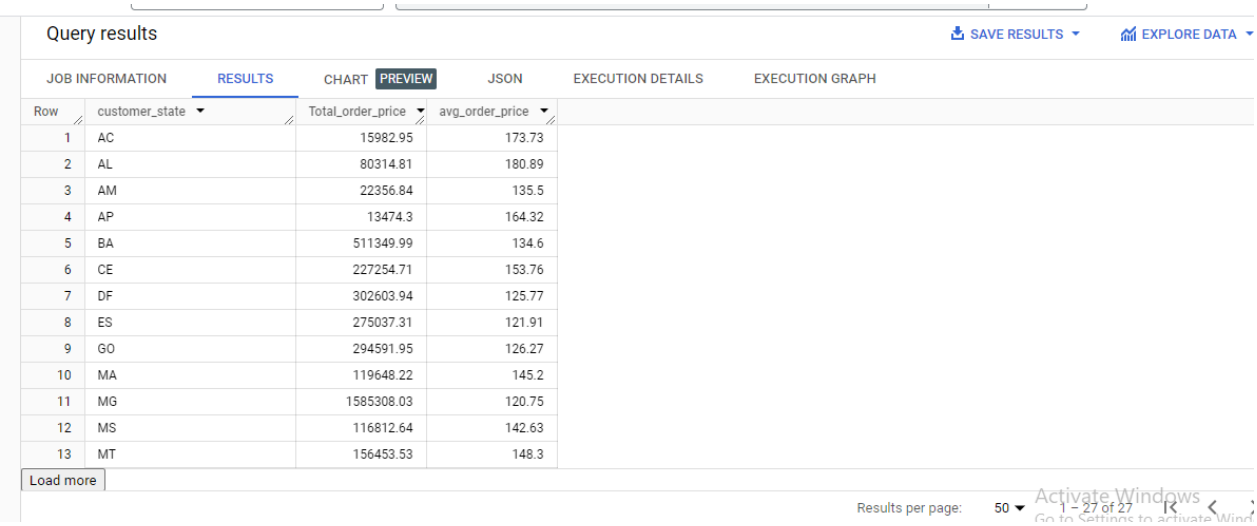
The cost of orders gave more than 100 % growth from the year 2017 to 2018.

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

QUERY:

```
SELECT
c.customer_state,
round (sum (odi.price), 2) as Total_order_price,
round (avg (odi.price), 2) as avg_order_price
FROM `Target_business_case.order_items` odi
join `Target_business_case.orders` o
on odi.order_id=o.order_id
join `Target_business_case.customers` c
on o.customer_id=c.customer_id
group by c.customer_state
order by c.customer_state
```

OUTPUT:



Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	Total_order_price	avg_order_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
11	MG	1585308.03	120.75
12	MS	116812.64	142.63
13	MT	156453.53	148.3

Load more

Results per page: 50 1 – 27 of 27

INSIGHTS:

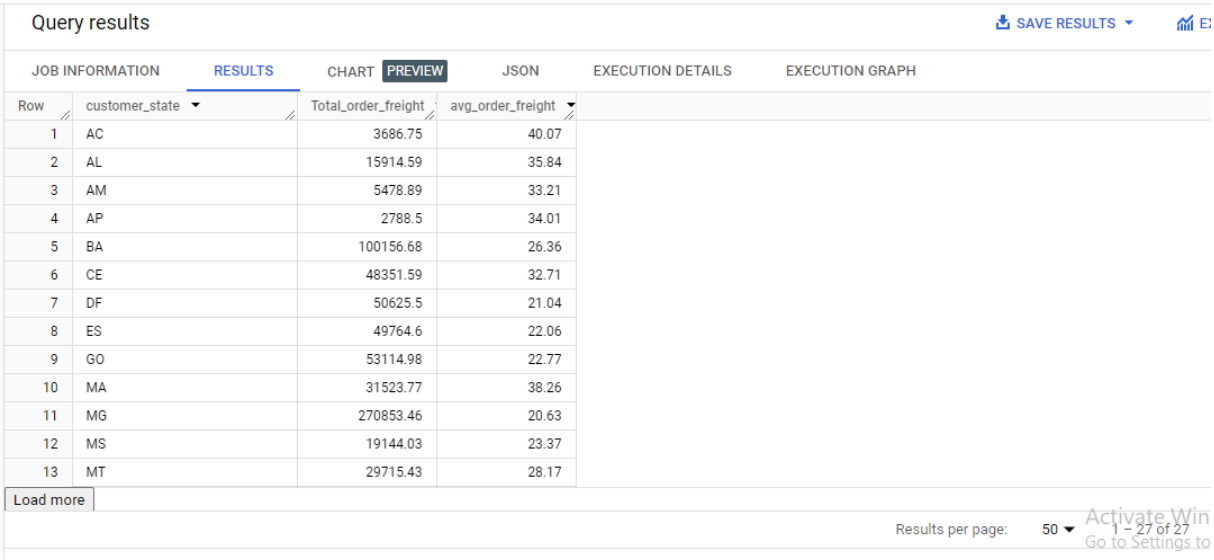
MG has the highest total order price although the average order price is not that high (120.75), which shows that number of orders placed in MG are quite significant.

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

QUERY:

```
SELECT
c.customer_state,
round(sum(odi.freight_value),2) as Total_order_freight,
round(avg(odi.freight_value),2) as avg_order_freight
FROM `Target_business_case.order_items` as odi
join  Target_business_case.orders as o
on odi.order_id=o.order_id
join `Target_business_case.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
order by c.customer_state
```

OUTPUT:



Row	customer_state	Total_order_freight	avg_order_freight
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
11	MG	270853.46	20.63
12	MS	19144.03	23.37
13	MT	29715.43	28.17

INSIGHTS:

The average Order freight price significantly shows that it is one of the major reasons for MG having highest total order price.

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.

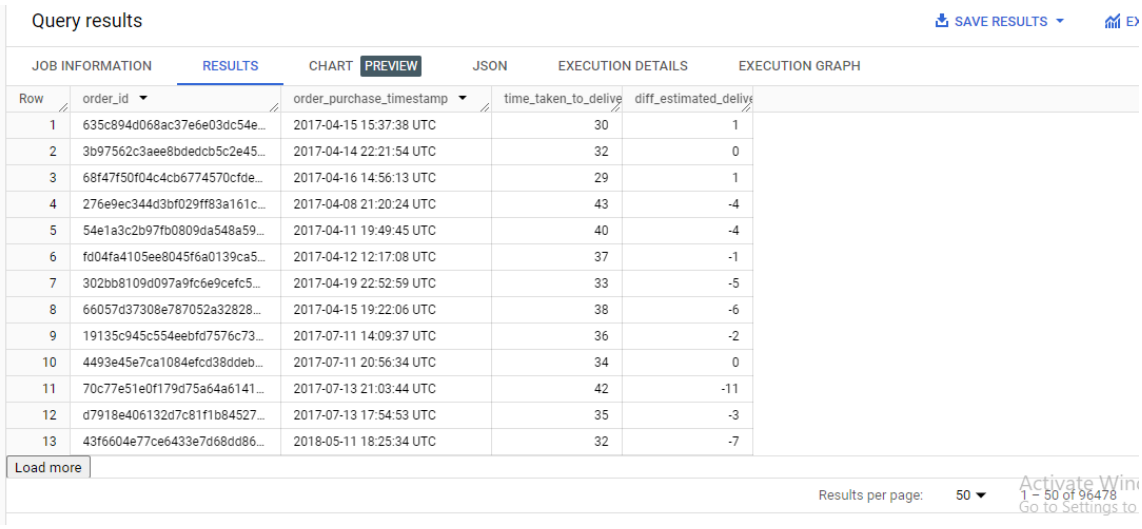
   You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
      i. **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
      ii. **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

QUERY:

```
SELECT
order_id,
order_purchase_timestamp,
time_taken_to_delivery,
diff_estimated_delivery
FROM
(
SELECT
order_id,
order_purchase_timestamp ,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day)
as time_taken_to_delivery,
```

```
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day)
as diff_estimated_delivery
FROM `Target_business_case.orders`
where order_status = 'delivered'
)
```

OUTPUT:



Query results

SAVE RESULTS

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

Row	order_id	order_purchase_timestamp	time_taken_to_delive	diff_estimated_deliv
1	635c894d068ac37e6e03dc54e...	2017-04-15 15:37:38 UTC	30	1
2	3b97562c3aee8bdedcb5c2e45...	2017-04-14 22:21:54 UTC	32	0
3	68f47f50f04c4cb6774570cfde...	2017-04-16 14:56:13 UTC	29	1
4	276e9ec344d3bf029ff83a161c...	2017-04-08 21:20:24 UTC	43	-4
5	54e1a3c2b97fb0809da548a59...	2017-04-11 19:49:45 UTC	40	-4
6	fd04fa4105ee8045f6a0139ca5...	2017-04-12 12:17:08 UTC	37	-1
7	302bb8109d097a9fc6e9cefc5...	2017-04-19 22:52:59 UTC	33	-5
8	66057d37308e787052a32828...	2017-04-15 19:22:06 UTC	38	-6
9	19135c945c554eebfd7576c73...	2017-07-11 14:09:37 UTC	36	-2
10	4493e45e7ca1084efcd38ddeb...	2017-07-11 20:56:34 UTC	34	0
11	70c77e51e0f179d75a64a6141...	2017-07-13 21:03:44 UTC	42	-11
12	d7918e406132d7c81f1b84527...	2017-07-13 17:54:53 UTC	35	-3
13	43f6604e77ce6433e7d68dd86...	2018-05-11 18:25:34 UTC	32	-7

Load more

Results per page: 50 1 – 50 of 96478

INSIGHTS:

The bigger positive value of diff_estimated_delivery column shows fast delivery of the order. Smaller the number, slower the delivery from the estimated time.

RECOMMENDATION:

Target should focus on reducing the delivery time by increasing the diff_estimated_value because less delivery time with no compromise in quality attracts more customers.

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

QUERY:

```
SELECT *
FROM
(
SELECT
c.customer_state,
round(avg(odi.freight_value),2) as avg_order_freight,
'top 5' as avg_order_type
FROM `Target_business_case.order_items` as odi
join `Target_business_case.orders` as o
on odi.order_id=o.order_id
join `Target_business_case.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
order by avg_order_freight desc
limit 5
)
```

OUTPUT:

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

Row	customer_state	avg_order_freight	avg_order_type
1	RR	42.98	top 5
2	PB	42.72	top 5
3	RO	41.07	top 5
4	AC	40.07	top 5
5	PI	39.15	top 5
6	SP	15.15	bottom 5
7	PR	20.53	bottom 5
8	MG	20.63	bottom 5
9	RJ	20.96	bottom 5
10	DF	21.04	bottom 5

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

QUERY:

```
SELECT*
FROM
(SELECT
c.customer_state,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)),0) as
delivery_time,
'top 5' as sorted_by
FROM `Target_business_case.orders` as o
join `Target_business_case.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
order by delivery_time desc
limit 5
)
UNION ALL
(
SELECT
c.customer_state,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)),0) as
delivery_time,
'bottom 5' as sorted_by
FROM `Target_business_case.orders` as o
join `Target_business_case.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
order by delivery_time
limit 5
)
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW | JSON | EXEC

Row	customer_state	delivery_time	sorted_by
1	RR	29.0	top 5
2	AP	27.0	top 5
3	AM	26.0	top 5
4	AL	24.0	top 5
5	PA	23.0	top 5
6	SP	8.0	bottom 5
7	MG	12.0	bottom 5
8	PR	12.0	bottom 5
9	DF	13.0	bottom 5
10	SC	14.0	bottom 5

INSIGHTS:

The delivery time and freight price are correlated in some way and this can be observed by looking at previous two tables.

d. 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.

QUERY:

```
SELECT
c.customer_state,
round(avg(date_diff(order_delivered_customer_date,order_estimated_delivery_date,day)),0) as
Quickness_parameter,
FROM `Target_business_case.orders` as o
join `Target_business_case.customers` as c
on o.customer_id = c.customer_id
group by c.customer_state
Order by Quickness_parameter
LIMIT 5
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW

Row	customer_state	Quickness_paramete
1	AC	-20.0
2	RO	-19.0
3	AM	-19.0
4	AP	-19.0
5	RR	-16.0

INSIGHTS:

RR, AP, AM although being in top 5 in the delivery time yet are among the fastest ones because the delivery is being delivered as promised and even way before.

6. **Analysis based on the payments:**

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

QUERY:

```
SELECT
py.payment_type as Payment_Type,
Extract(year from o.order_purchase_timestamp ) as o_year,
Extract(month from o.order_purchase_timestamp) as t_month,
count(o.order_id) as No_of_orders
FROM `Target_business_case.orders` as o
join `Target_business_case.payments` as py
on o.order_id = py.order_id
group by o_year,t_month, py.payment_type
```

OUTPUT:

Query results

SAV

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

Row	Payment_Type	o_year	t_month	No_of_orders
1	credit_card	2017	11	5897
2	credit_card	2018	3	5691
3	credit_card	2018	1	5520
4	credit_card	2018	5	5497
5	credit_card	2018	4	5455
6	credit_card	2018	2	5253
7	credit_card	2018	8	4985
8	credit_card	2018	6	4813
9	credit_card	2018	7	4755
10	credit_card	2017	12	4377
11	credit_card	2017	10	3524
12	credit_card	2017	8	3284
13	credit_card	2017	9	3283

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

Row	Payment_Type	o_year	t_month	No_of_orders
13	credit_card	2017	9	3283
14	credit_card	2017	7	3086
15	credit_card	2017	5	2853
16	credit_card	2017	6	2463
17	credit_card	2017	3	2016
18	credit_card	2017	4	1846
19	UPI	2018	1	1518
20	UPI	2017	11	1509
21	credit_card	2017	2	1356
22	UPI	2018	3	1352
23	UPI	2018	2	1325
24	UPI	2018	4	1287
25	UPI	2018	5	1263
26	UPI	2018	7	1229

INSIGHTS:
Most of the payments are made via credit card which is followed by UPI.

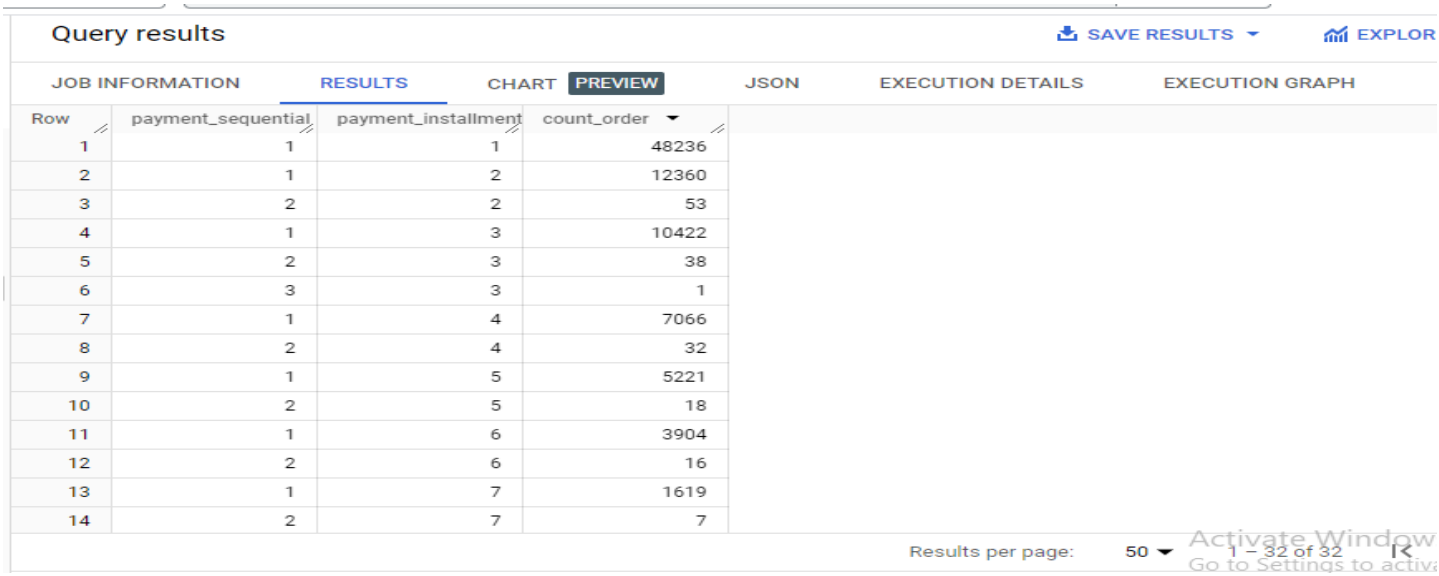
RECOMMENDATIONS:
Target should collab with gateway provider companies to give extra discounts to customers so that customer base as well as customer retention can be increased.

b. Find the no. of orders placed on the basis of the payment installments that have been paid.

QUERY:

```
SELECT
payment_sequential,
payment_installments,
count(order_id) as count_order
FROM `Target_business_case.payments`
WHERE payment_installments >= payment_sequential
GROUP BY payment_sequential, payment_installments
```

OUTPUT:



INSIGHTS:

Majority of the payments are on one time basis but significant amount of orders preferred payments in installments. In instalments quarterly installments are preferred by customers.

OVERALL RECOMMENDATIONS:

- There should have a greater number of cities included in the active customers list in order to have bigger crowd to serve, as the difference was observed in part iii) of the first question
- Giving better offers like more discounts, cashback offers and buy few get 1 free to name a few types offers in the months where there is a nose dip in sales numbers which have been seen in the offseason to attract more customers.
- Having more inventory and better delivery systems to avoid delays in the delivery time to avoid loss of customers over the years.
- Customers with a good purchase history need to be given special privileges in terms of special discounts for their retention over a longer time span.
- The average purchasing power of customers must be kept in mind while having stores with costly products in those states where the sales values and numbers are low.