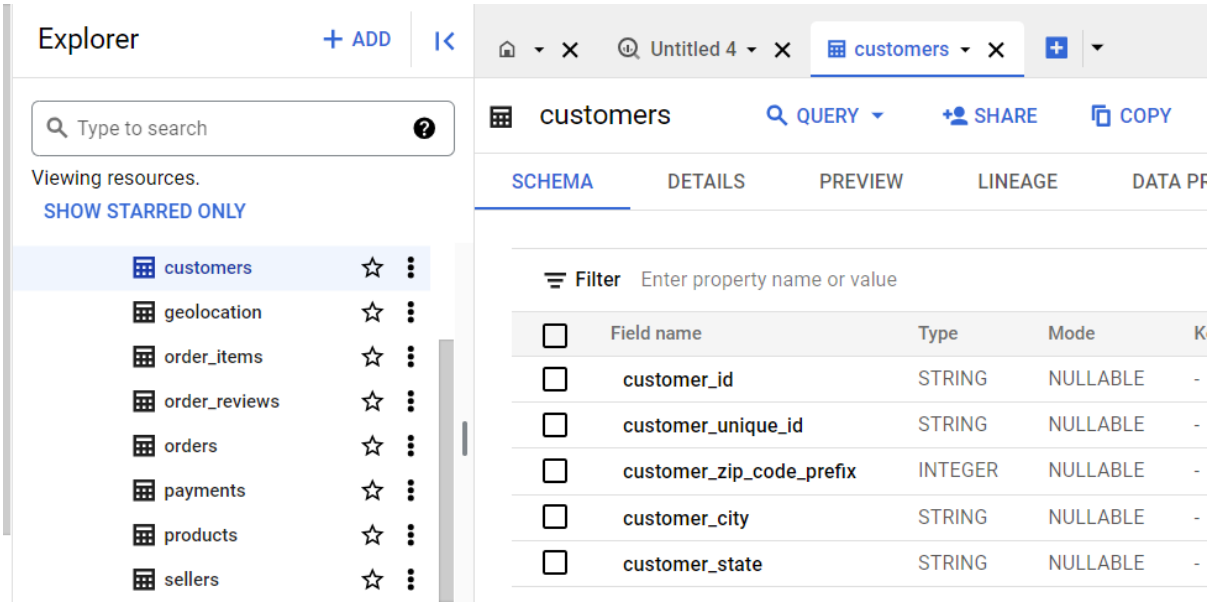
Lohith-Business Case: Target SQL Submission

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**

   Ans:

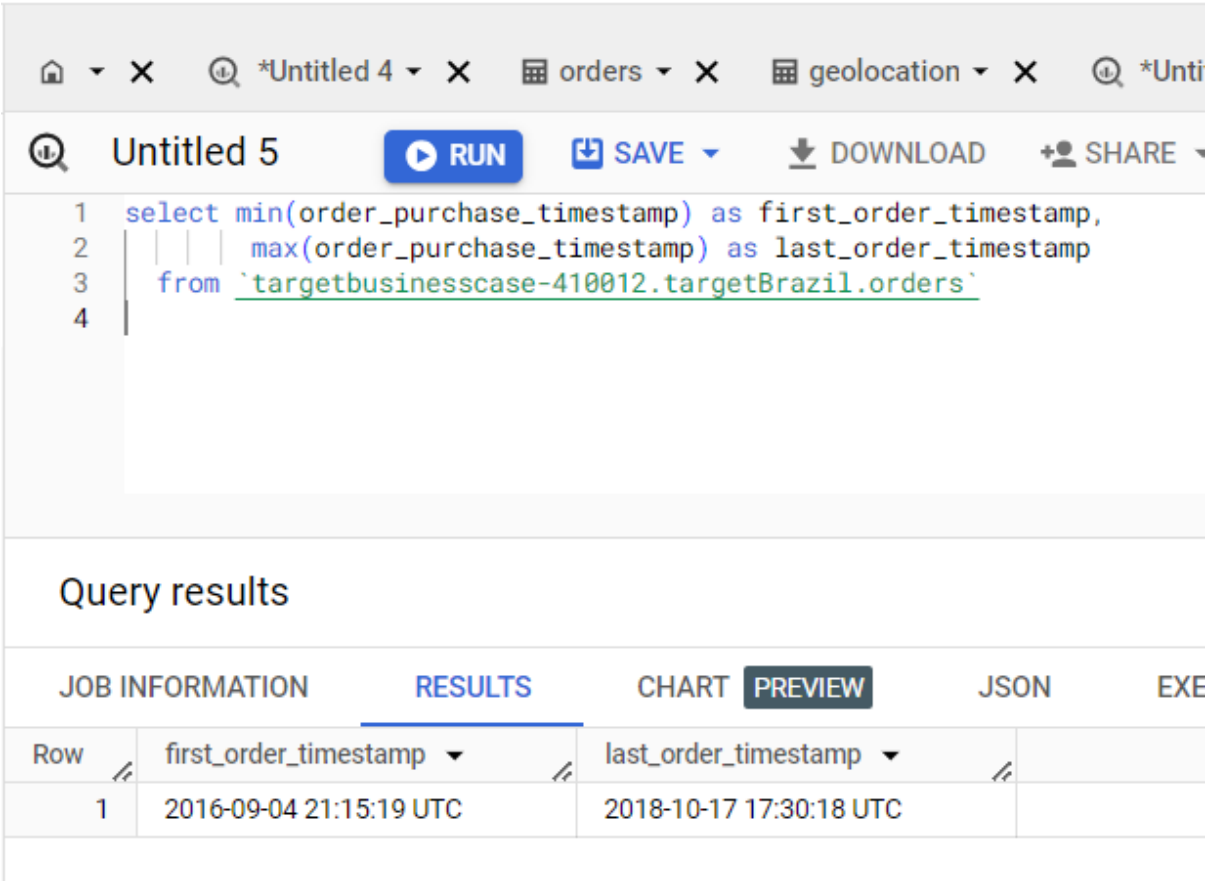


The data types of the all the customers in customer table is in above screen shot
customer_id, customer_unique_id, customer_zip_code_prefix, customer_city, customer_state **- STRING**

customer_state **– INTEGER**

b) Get the time range between which the orders were placed.

Ans:



Query:

```
select min(order_purchase_timestamp) as first_order_timestamp,
       max(order_purchase_timestamp) as last_order_timestamp
  from `targetbusinesscase-410012.targetBrazil.orders`
```

output:

First purchase date: 2016-09-04 21:15:19 UTC

Last Purchase date: 2018-10-17 17:30:18 UTC

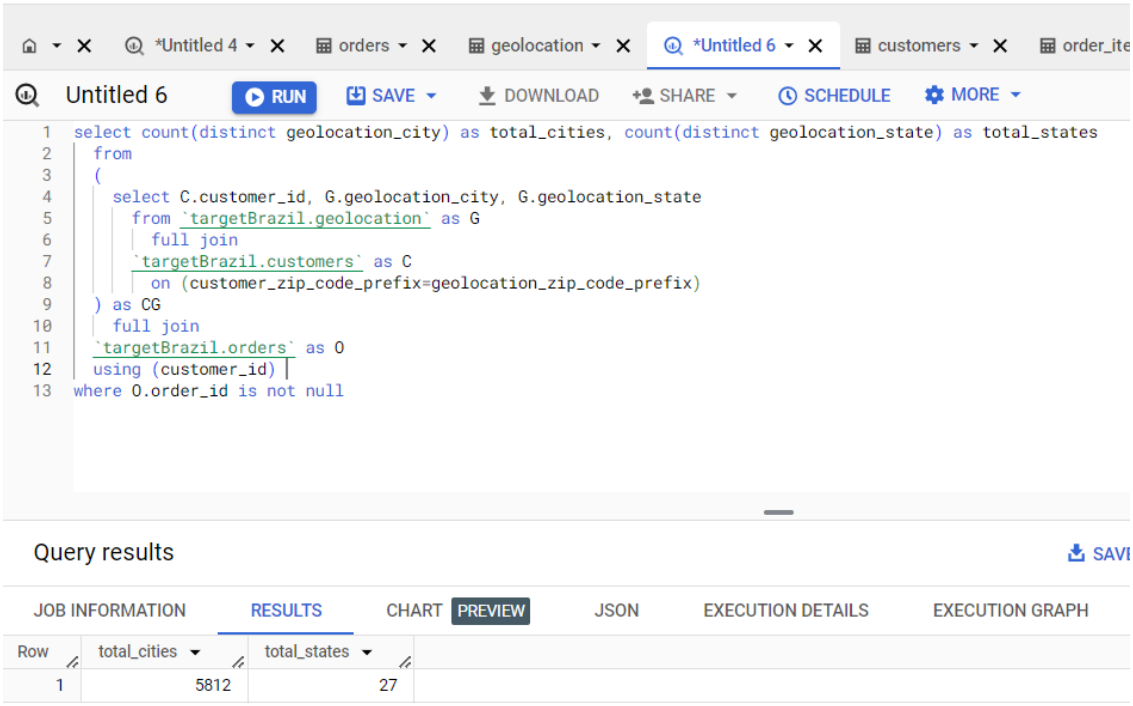
The difference between the dates is 774 days

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

Ans:

Query:

```
select count(distinct geolocation_city) as total_cities,
count(distinct geolocation_state) as total_states
  from
  (
    select C.customer_id, G.geolocation_city,
G.geolocation_state
      from `targetBrazil.geolocation` as G
        full join
      `targetBrazil.customers` as C
        on
(customer_zip_code_prefix=geolocation_zip_code_prefix)
  ) as CG
    full join
  `targetBrazil.orders` as O
  using (customer_id)
where O.order_id is not null
```



Output:

Total cities: 5812

Total States: 27

Explanation:

Since we need the cities and states of the customers who ordered during the given period we need to do following

1. Join the geolocation and customers using the zip_code and considered as the CG table.

2. Now join the CG table and orders table using order_id, Then we get all the orders
3. Count the cities and states and add the condition to check the orders should not be null

==========

2. **In-depth Exploration:**

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

Ans:

Query:

```
select month,year,monthly_count, sum(monthly_count) over (partition by
year) as yearly_count from(
  select count(order_id) as monthly_count, month, year from (
    select order_id, extract(MONTH from order_purchase_timestamp) as
month, extract(year from order_purchase_timestamp) as year
from `targetbusinesscase-410012.targetBrazil.orders`) group by
year,month) order by year, month
```

output:

Row	month	year	monthly_count	yearly_count
1	9	2016	4	329
2	10	2016	324	329
3	12	2016	1	329
4	1	2017	800	45101
5	2	2017	1780	45101
6	3	2017	2682	45101
7	4	2017	2404	45101
8	5	2017	3700	45101
9	6	2017	3245	45101
10	7	2017	4026	45101
11	8	2017	4331	45101
12	9	2017	4285	45101
13	10	2017	4631	45101
14	11	2017	7544	45101
15	12	2017	5673	45101
16	1	2018	7269	54011
17	2	2018	6728	54011
18	3	2018	7211	54011
19	4	2018	6939	54011
20	5	2018	6873	54011
21	6	2018	6167	54011
22	7	2018	6292	54011
23	8	2018	6512	54011
24	9	2018	16	54011
25	10	2018	4	54011

Observation:

From the above table it's clear that there is growing trend when we see on yearly basis

2016 -> Total orders placed are 329
2017 -> Total orders placed are 45101

2018-> Total orders placed are 54011

On monthly basis we can observe that some time the orders are more and sometimes the orders are less but overall we can see a growing trend in orders yearly

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

Ans: No, sales are uncertain, however from the output table of question 1 we can say that the sales are high during September to march months

c) During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
   1. 0-6 hrs : Dawn
   2. 7-12 hrs : Mornings
   3. 13-18 hrs : Afternoon
   4. 19-23 hrs : Night

Ans

Query:

```
select  count(order_id) as order_count,
case
  when hour between 0 and 6 then "Dawn"
  when hour between 7 and 12 then "Mornings"
  when hour between 13 and 18 then "Afternoon"
  when hour between 19 and 23 then "Night"
  else NULL
  end as time_period from
  (
    select order_id,order_purchase_timestamp,
          extract(hour from order_purchase_timestamp) as hour
      from  `targetBrazil.orders`
  )  group by time_period
```

Output:

Untitled 8 RUN SAVE DOWNLOAD

```
1
2  select  count(order_id) as order_count,
3  case
4    when hour between 0 and 6 then "Dawn"
5    when hour between 7 and 12 then "Mornings"
6    when hour between 13 and 18 then "Afternoon"
7    when hour between 19 and 23 then "Night"
8    else NULL
9    end as time_period from
10   (
11     select order_id,order_purchase_timestamp,
12     extract(hour from order_purchase_timestamp) as hour from
13   )  group by time_period
```

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW

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

Observation:

Brazilian customer mostly place their orders during Afternoon

---

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

Ans:

Query:

```
select count(order_id) as state_wise_monthly_count ,
geolocation_state, year , month from
(
  select geolocation_state,order_id, order_purchase_timestamp,
  extract(month from order_purchase_timestamp) as month,
  extract(year from order_purchase_timestamp) as year
  from
  (
    select C.customer_id, G.geolocation_city, G.geolocation_state
      from `targetBrazil.geolocation` as G
        full join
      `targetBrazil.customers` as C
        on (customer_zip_code_prefix=geolocation_zip_code_prefix)
```

```
  ) as CG
    full join
  `targetBrazil.orders` as O
  using (customer_id
)
  where order_id is not null) group by geolocation_state, year, month
  having geolocation_state is not null
  order by geolocation_state, year, month limit 30
```

Output

Row	state_wise_monthly_count	geolocation_state	year	month
1	45	AC	2017	1
2	179	AC	2017	2
3	329	AC	2017	3
4	362	AC	2017	4
5	886	AC	2017	5
6	432	AC	2017	6
7	605	AC	2017	7
8	657	AC	2017	8
9	161	AC	2017	9
10	535	AC	2017	10
11	368	AC	2017	11
12	389	AC	2017	12
13	649	AC	2018	1
14	336	AC	2018	2
15	187	AC	2018	3
16	427	AC	2018	4
17	275	AC	2018	5
18	131	AC	2018	6
19	332	AC	2018	7
20	403	AC	2018	8
21	52	AL	2016	10
22	106	AL	2017	1
23	826	AL	2017	2
24	843	AL	2017	3
25	1158	AL	2017	4
26	1711	AL	2017	5
27	1037	AL	2017	6
28	648	AL	2017	7
29	1281	AL	2017	8
30	1862	AL	2017	9

b) How are the customers distributed across all the states?

Ans:

Query:

```
select  count(distinct customer_id) as state_level_orders,
geolocation_state
  from
  (
    select C.customer_id, G.geolocation_city, G.geolocation_state
      from `targetBrazil.geolocation` as G
        full join
      `targetBrazil.customers` as C
        on (customer_zip_code_prefix=geolocation_zip_code_prefix)
  ) as CG
    full join
  `targetBrazil.orders` as O
  using (customer_id)
group by geolocation_state having geolocation_state is not null
```

output:

Row	state_level_customers	geolocation_state
1	3371	BA
2	11624	MG
3	5034	PR
4	12839	RJ
5	5473	RS
6	41731	SP
7	2011	GO
8	715	MS
9	905	MT
10	534	PB
11	483	RN
12	1332	CE
13	2027	ES
14	972	PA
15	3651	SC
16	412	AL
17	68	AP
18	1648	PE
19	1974	DF
20	349	SE
21	279	TO
22	256	RO
23	492	PI
24	743	MA
25	148	AM
26	120	AC
27	46	RR

---

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.

Ans:

Query:

```
with Order_Purchase_table as


(
    select O.order_id,O.order_purchase_timestamp, P.payment_value  from
    `targetBrazil.orders` as O full join
    `targetBrazil.payments` as P using (order_id)
),

MY_P_table as
(
   select order_id,
  extract (month from order_purchase_timestamp) as month ,
  extract (year from order_purchase_timestamp) as year,
  payment_value from Order_Purchase_table

),

monthly_cost_Jan_to_Aug as
(
    select distinct(month), year,
    sum(payment_value) over (partition by year,month) as monthly_cost
    from MY_P_table where month between 1 and 8 and year <> 2016 order by year,
month

),

_2018 as
(
  select month, monthly_cost as monthly_cost_2018
  from monthly_cost_Jan_to_Aug where year = 2018
),

_2017 as
(
  select month, monthly_cost as monthly_cost_2017
  from monthly_cost_Jan_to_Aug where year = 2017
),

years_2017_2018 as
(
  select month, monthly_cost_2018, monthly_cost_2017
  from _2018 join _2017 using (month)
),
monthly_percent_increse as
(
  select *, round((((monthly_cost_2018-monthly_cost_2017)/monthly_cost_2017) *
100)) as month_percent_increase
  from years_2017_2018
),
```

```
yearly_cost as
(
  select sum(monthly_cost_2018) as yearly_cost_2018,
         sum(monthly_cost_2017) as yearly_cost_2017
         from years_2017_2018
),

percent_increase_yearly as
(
  select *, round((((yearly_cost_2018-yearly_cost_2017)/yearly_cost_2017) * 100))
as yearly_percent_increase
  from yearly_cost
)

select * from  percent_increase_yearly
```

output:

yearly percent increase

Row	yearly_cost_2018	yearly_cost_2017	yearly_percent_increase
1	8694733.84	3669022.1199999996	137.0

Monthly Percent increase

Row	month	monthly_cost_2018	monthly_cost_2017	month_percent_increase
1	1	1115004.18	138488.04	705.0
2	2	992463.34	291908.01	240.0
3	3	1159652.12	449863.6	158.0
4	4	1160785.48	417788.03	178.0
5	5	1153982.15	592918.82	95.0
6	6	1023880.5	511276.38	100.0
7	7	1066540.75	592382.92	80.0
8	8	1022425.32	674396.32	52.0

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

Ans:

Query:

```
with payments_orders as
(
   select  P.payment_value, O.customer_id  from `targetbusinesscase-
410012.targetBrazil.payments` as P
                  full join
```

```
                    `targetbusinesscase-410012.targetBrazil.orders`  as O
                    using(order_id)
),

POC as
(
   select  * from payments_orders as PO
            full join
            `targetbusinesscase-410012.targetBrazil.customers` as C
            using (customer_id)
)

select distinct(customer_state), round(sum(payment_value) over (partition by
customer_state)) as total, round(avg(payment_value) over (partition by
customer_state)) as average from POC order by customer_state
```

output:

Row	customer_state	total	average
1	AC	19681.0	234.0
2	AL	96962.0	227.0
3	AM	27967.0	182.0
4	AP	16263.0	232.0
5	BA	616646.0	171.0
6	CE	279464.0	200.0
7	DF	355141.0	161.0
8	ES	325968.0	155.0
9	GO	350092.0	166.0
10	MA	152523.0	199.0
11	MG	1872257.0	155.0
12	MS	137535.0	187.0
13	MT	187029.0	195.0
14	PA	218296.0	216.0
15	PB	141546.0	248.0
16	PE	324850.0	188.0
17	PI	108524.0	207.0
18	PR	811156.0	154.0
19	RJ	2144380.0	159.0
20	RN	102718.0	197.0
21	RO	60866.0	233.0
22	RR	10065.0	219.0
23	RS	890899.0	157.0
24	SC	623086.0	166.0
25	SE	75246.0	208.0
26	SP	5998227.0	138.0
27	TO	61485.0	204.0

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

Ans:

Query:

```
with order_items_sellers as

(
   select O.freight_value, S.seller_zip_code_prefix
      from `targetbusinesscase-410012.targetBrazil.order_items` as O

         full join

         `targetbusinesscase-410012.targetBrazil.sellers` as S

         using (seller_id)

),

OSL as
(
   select * from  order_items_sellers as OS
      full join
      `targetbusinesscase-410012.targetBrazil.geolocation` as L
      on (OS.seller_zip_code_prefix=geolocation_zip_code_prefix)

)

select distinct(geolocation_state),
      round(sum(freight_value) over (partition by geolocation_state)) as
Total_freight,
      round(avg(freight_value) over (partition by geolocation_state)) as
Average_freight from OSL order by geolocation_state
```

Output:

Row	geolocation_state	Total_freight	Average_freight
1	*null*	5199.0	21.0
2	AC	5386.0	33.0
3	AL	*null*	*null*
4	AM	2209.0	27.0
5	AP	*null*	*null*
6	BA	1939324.0	29.0
7	CE	163716.0	54.0
8	DF	1223547.0	19.0
9	ES	724107.0	29.0
10	GO	694620.0	26.0
11	MA	1201988.0	30.0
12	MG	47130618.0	23.0
13	MS	113814.0	26.0
14	MT	263738.0	32.0
15	PA	*null*	*null*
16	PB	83960.0	35.0
17	PE	265633.0	28.0
18	PI	1773.0	37.0
19	PR	25931024.0	22.0
20	RJ	18429357.0	19.0
21	RN	63552.0	16.0
22	RO	85745.0	50.0
23	RR	*null*	*null*
24	RS	9217561.0	24.0
25	SC	17136727.0	27.0
26	SE	11798.0	29.0
27	SP	198571258.0	18.0
28	TO	*null*	*null*

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:
        1. **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
        2. **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

Ans:

Query:

```
select * from (select order_id,
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 `targetbusinesscase-410012.targetBrazil.orders` where
order_status = "delivered") where  time_to_deliver is not null and
diff_estimated_delivery is not null  limit 10
```

Output:

Row	order_id	time_to_deliver	diff_estimated_delive
1	c158e9806f85a33877bdfd4f60...	23	9
2	b60b53ad0bb7dacacf2989fe2...	12	-5
3	c830f223aae08493ebecb52f2...	12	12
4	a8aa2cd070eeac7e4368cae3d...	7	1
5	813c55ce9b6baa8f879e064fbf...	12	9
6	44558a1547e448b41c48c4087...	1	5
7	036b791897847cdb8e39df794...	6	0
8	1aba60c04110bdd421b250ea3...	21	7
9	0312ecf90786def87f98aa19e0...	7	0
10	635c894d068ac37e6e03dc54e...	30	1

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

Query:

```
with order_items_sellers as
(
   select O.freight_value, S.seller_zip_code_prefix
      from `targetbusinesscase-410012.targetBrazil.order_items` as O

         full join

         `targetbusinesscase-410012.targetBrazil.sellers` as S

         using (seller_id)

),

OSL as
(
   select * from  order_items_sellers as OS
      full join
      `targetbusinesscase-410012.targetBrazil.geolocation` as L
      on (OS.seller_zip_code_prefix=geolocation_zip_code_prefix)

),

total_average_freight as(

select distinct(geolocation_state),
      round(sum(freight_value) over (partition by geolocation_state)) as
Total_freight,
      round(avg(freight_value) over (partition by geolocation_state)) as
Average_freight from OSL order by geolocation_state
),

top5_highest_Average_freight as(
select geolocation_state, Average_freight as Highest_Average_freight from
total_average_freight where geolocation_state is not null order by
Average_freight desc limit 5
),

top5_lowest_Average_freight as(
select geolocation_state, Average_freight as Lowest_Average_freight from
total_average_freight where Average_freight is not null and
geolocation_state is not null order by Average_freight  limit 5
)

select * from
(select ROW_NUMBER() OVER (ORDER BY Highest_Average_freight desc) as
S_NO,Highest_Average_freight,geolocation_state as
high_avg_geolocation_state   from top5_highest_Average_freight) as H full
join

(select ROW_NUMBER() OVER (ORDER BY Lowest_Average_freight) as
S_NO,Lowest_Average_freight,geolocation_state as
low_avg_geolocation_state   from
top5_lowest_Average_freight) as L using(S_NO) order by S_NO
```

output:

S_NO	Highest_Average_fre	high_avg_geolocation_state	Lowest_Average_frei	low_avg_geolocation_state
1	54.0	CE	16.0	RN
2	50.0	RO	18.0	SP
3	37.0	PI	19.0	DF
4	35.0	PB	19.0	RJ
5	33.0	AC	22.0	PR

Some states have no freight charges that are not shown in above table
In Question no condition mentioned if two states has same values, if that is the Case we can use rank

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

Ans:

Query:

```
with state_avg_delv_time
as
(

select customer_state, round(avg(time_to_deliver),2) as
state_level_avg_devivery_time from (select O.order_id,
date_diff(O.order_delivered_customer_date, O.order_purchase_timestamp,
day) as time_to_deliver, C.*
from `targetbusinesscase-410012.targetBrazil.orders` as O full join
`targetbusinesscase-410012.targetBrazil.customers` as C using
(customer_id)) group by customer_state

),

state_avg_delv_high_top5
as
(
   select rank() over (order by state_level_avg_devivery_time desc) as
Rank,
   customer_state as high_avg_customer_state,
state_level_avg_devivery_time as high_state_level_avg_devivery_time
   from state_avg_delv_time order by Rank limit 5
)
,

state_avg_delv_low_top5
as
(
   select rank() over (order by state_level_avg_devivery_time) as
Rank,
   customer_state as low_avg_customer_state,
state_level_avg_devivery_time as low_state_level_avg_devivery_time
   from state_avg_delv_time order by Rank limit 5
)
```

```
select * from state_avg_delv_low_top5 left join
state_avg_delv_high_top5 using (Rank) order by Rank
```

output:

Actual avg delivery

Row	Rank	low_avg_customer_state	low_state_level_avg	high_avg_customer_state	high_state_level_avg_devivery_time
1	1	SP	8.3	RR	28.98
2	2	PR	11.53	AP	26.73
3	3	MG	11.54	AM	25.99
4	4	DF	12.51	AL	24.04
5	5	SC	14.48	PA	23.32

Expected avg delivery

Rank	low_avg_customer_state	low_state_level_avg	high_avg_customer_state	high_state_level_avg
1	SP	18.78	AP	45.87
2	DF	23.95	RR	45.63
3	MG	24.19	AM	44.92
4	PR	24.25	AC	40.72
5	ES	25.22	RO	38.39

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:

```
with delivered_orders as
(
   select * from `targetbusinesscase-410012.targetBrazil.orders` where
order_status="delivered"
),

state_level_actual_avg_devivery_time as
(
select customer_state, round(avg(time_to_deliver),2) as
actual_avg_devivery_time from (select O.order_id,
date_diff(O.order_delivered_customer_date, O.order_purchase_timestamp, day)
as time_to_deliver, C.*
from delivered_orders as O full join `targetbusinesscase-
410012.targetBrazil.customers` as C using
(customer_id)) group by customer_state
),

state_level_estimated_avg_devivery_time as
(
select customer_state, round(avg(time_to_deliver),2) as
estimated_avg_avg_devivery_time from (select O.order_id,
date_diff(O.order_estimated_delivery_date, O.order_purchase_timestamp, day)
as time_to_deliver, C.*
```

```
from delivered_orders as O full join `targetbusinesscase-
410012.targetBrazil.customers` as C using
(customer_id)) group by customer_state
),

state_level_delivery_days_in_advance as
(
select *, (E.estimated_avg_avg_devivery_time - A.actual_avg_devivery_time)
as delivery_days_in_advance
from state_level_estimated_avg_devivery_time as E full join
state_level_actual_avg_devivery_time as A using (customer_state)
)

select * from
(select rank() over (order by delivery_days_in_advance desc) as
Rank,customer_state as top5_fast_delivery_state
from state_level_delivery_days_in_advance  order by Rank asc limit 5) full
join

(select rank() over (order by delivery_days_in_advance asc) as
Rank,customer_state as top5_slow_delivery_state
from state_level_delivery_days_in_advance  order by Rank limit 5) using
(Rank) order by Rank
```

Output:

Row	Rank	top5_fast_delivery_state	top5_slow_delivery_state
1	1	AC	AL
2	2	RO	MA
3	3	AP	SE
4	4	AM	ES
5	5	RR	CE

VI. Analysis based on the payments:

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

Hint: We want you to count the no. of orders placed using different payment methods in each month over the past years.

Ans:

Query:

```
select year, month, payment_type, count(order_id) as order_count from(
select *, extract(month from order_purchase_timestamp) as month, extract(year from
order_purchase_timestamp) as year from `targetbusinesscase-
410012.targetBrazil.orders` full join `targetbusinesscase-
410012.targetBrazil.payments` using (order_id)) group by 1, 2, 3 order by 1,2,3
```

Output:

Row	year	month	payment_type	order_count
1	2016	9	*null*	1
2	2016	9	credit_card	3
3	2016	10	UPI	63
4	2016	10	credit_card	254
5	2016	10	debit_card	2
6	2016	10	voucher	23
7	2016	12	credit_card	1
8	2017	1	UPI	197
9	2017	1	credit_card	583
10	2017	1	debit_card	9
11	2017	1	voucher	61
12	2017	2	UPI	398
13	2017	2	credit_card	1356
14	2017	2	debit_card	13
15	2017	2	voucher	119
16	2017	3	UPI	590
17	2017	3	credit_card	2016
18	2017	3	debit_card	31
19	2017	3	voucher	200
20	2017	4	UPI	496

B. Find the no. of orders placed on the basis of the payment installments that have been paid.
Hint: We want you to count the no. of orders placed based on the no. of payment installments where at least one installment has been successfully paid.

Ans:

Query:

```
select count(order_id) as order_count, payment_installments
from (
   select * from `targetbusinesscase-410012.targetBrazil.orders`
      full join `targetbusinesscase-410012.targetBrazil.payments`
      using (order_id)) group by 2 having payment_installments is not null and
payment_installments <> 0 order by 2
```

Row	order_count	payment_installments
1	52546	1
2	12413	2
3	10461	3
4	7098	4
5	5239	5
6	3920	6
7	1626	7
8	4268	8
9	644	9
10	5328	10
11	23	11
12	133	12
13	16	13
14	15	14
15	74	15
16	5	16
17	8	17
18	27	18
19	17	20
20	3	21
21	1	22
22	1	23
23	18	24