Project- Target SQL

Submitted by Mr Marshal Harsh Mathew

Data set- Target , Brazil

Synopsis

Data analysis based on the data set from Target to derive valuable insights on the sale, customer behavior etc. Based on the data we are trying to find the trends in different aspects by using Bigquery.

This project is divided into different categories and in each category different queries are used to get insights

Category 1

Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

1.Data type of columns in a table

Query

```
 select
*
from `Bigquery_project_marsh050223`.INFORMATION_SCHEMA.COLUMNS;
```

Result

table_schema	table_name	column_name	ordinal_position
Bigquery_project_marsh050223	customers	customer_id	1
Bigquery_project_marsh050223	customers	customer_unique_id	2
Bigquery_project_marsh050223	customers	customer_zip_code_prefix	3
Bigquery_project_marsh050223	customers	customer_city	4
Bigquery_project_marsh050223	customers	customer_state	5
Bigquery_project_marsh050223	geolocations	geolocation_zip_code_prefix	1
Bigquery_project_marsh050223	geolocations	geolocation_lat	2
Bigquery_project_marsh050223	geolocations	geolocation_lng	3
Bigquery_project_marsh050223	geolocations	geolocation_city	4
Bigquery_project_marsh050223	geolocations	geolocation_state	5
Bigquery_project_marsh050223	order_items	order_id	1
Bigquery_project_marsh050223	order_items	order_item_id	2
Bigquery_project_marsh050223	order_items	product_id	3
Bigquery_project_marsh050223	order_items	seller_id	4
Bigquery_project_marsh050223	order_items	shipping_limit_date	5
Bigquery_project_marsh050223	order_items	price	6
Bigquery_project_marsh050223	order_items	freight_value	7
Bigquery_project_marsh050223	order_reviews	review_id	1
Bigquery_project_marsh050223	order_reviews	order_id	2
Bigquery_project_marsh050223	order_reviews	review_score	3
Bigquery_project_marsh050223	order_reviews	review_comment_title	4
Bigquery_project_marsh050223	order_reviews	review_creation_date	5
Bigquery_project_marsh050223	order_reviews	review_answer_timestamp	6
Bigquery_project_marsh050223	Orders	order_id	1
Bigquery_project_marsh050223	Orders	customer_id	2
Bigquery_project_marsh050223	Orders	order_status	3
Bigquery_project_marsh050223	Orders	order_purchase_timestamp	4
Bigquery_project_marsh050223	Orders	order_approved_at	5
Bigquery_project_marsh050223	Orders	order_delivered_carrier_date	6
Bigquery_project_marsh050223	Orders	order_delivered_customer_date	7
Bigquery_project_marsh050223	Orders	order_estimated_delivery_date	8
Bigquery_project_marsh050223	payments	order_id	1
Bigquery_project_marsh050223	payments	payment_sequential	2
Bigquery_project_marsh050223	payments	payment_type	3
Bigquery_project_marsh050223	payments	payment_installments	4
Bigquery_project_marsh050223	payments	payment_value	5
Bigquery_project_marsh050223	products	product_id	1
Bigquery_project_marsh050223	products	product_category	2
Bigquery_project_marsh050223	products	product_name_length	3
Bigquery_project_marsh050223	products	product_description_length	4
Bigquery_project_marsh050223	products	product_photos_qty	5
Bigquery_project_marsh050223	products	product_weight_g	6
Bigquery_project_marsh050223	products	product_length_cm	7
Bigquery_project_marsh050223	products	product_height_cm	8
Bigquery_project_marsh050223	products	product_width_cm	9

Bigquery_project_marsh050223	Sellers	seller_id	1
Bigquery_project_marsh050223	Sellers	seller_zip_code_prefix	2
Bigquery_project_marsh050223	Sellers	seller_city	3
Bigquery_project_marsh050223	Sellers	seller_state	4

Insight

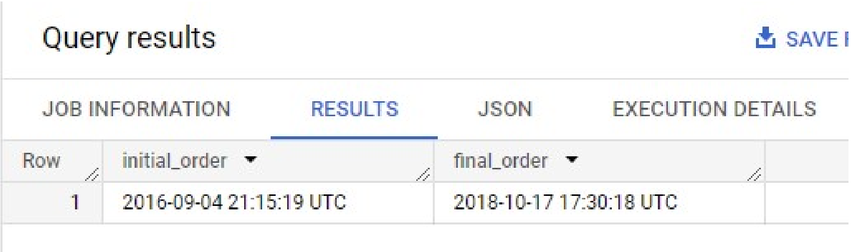
The data is stored in different columns and the primary keys like order_id, customer_id, product_id, seller_id, state_id will help in joining tables to get the insigts

2.Time period of orders

Query

```
select
min(order_purchase_timestamp) as initial_order,
max(order_purchase_timestamp) as final_order
from`Bigquery_project_marsh050223.orders`;
```

Result



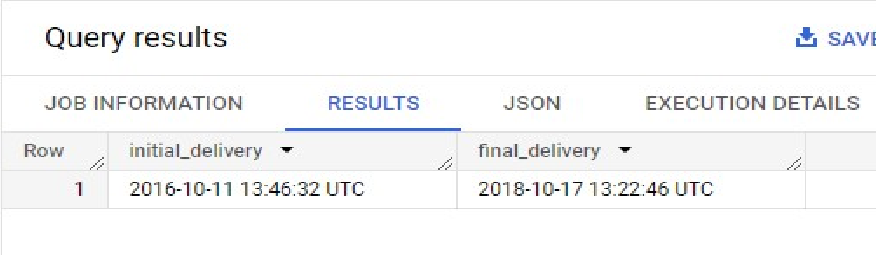
Row	initial_order	final_order
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

3.Time period of order delivery to customer

Query

```
select
min(order_delivered_customer_date) as initial_delivery,
max(order_delivered_customer_date) as final_delivery
from`Bigquery_project_marsh050223.orders`;
```

Result



Row	initial_delivery	final_delivery
1	2016-10-11 13:46:32 UTC	2018-10-17 13:22:46 UTC

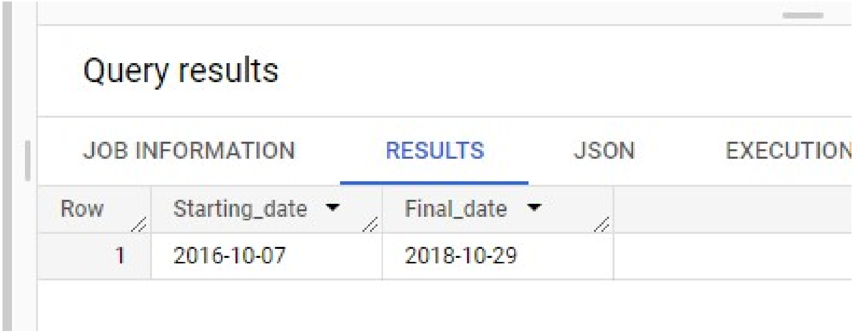
Time period of reviews

The data table was wrong in capturing the date .The year, date and month columns got interchanged which is corrected in the below query

Query

```
select
min(date_act) as Starting_date,
max(date_act) as Final_date
from(
select
date((t.day_new),t.month_new,t.year_new) as date_act
from
(
select
extract(day from review_answer_timestamp)+2000 as day_new,
extract(month from review_answer_timestamp) as month_new,
extract(year from review_answer_timestamp) as year_new
from `Bigquery_project_marsh050223.order_reviews`
order by year_new desc)t
)
```

Result



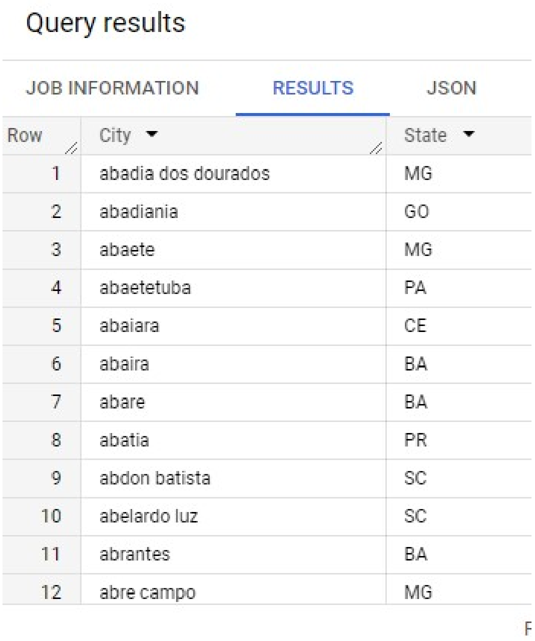
Row	Starting_date	Final_date
1	2016-10-07	2018-10-29

Cities and States of customers ordered during the given period

Query

```
select
customer_city as City,
customer_state as State
from `Bigquery_project_marsh050223.customers`
group by customer_city,customer_state
order by customer_city;
```

Result



Row	City	State
1	abadia dos dourados	MG
2	abadiania	GO
3	abaete	MG
4	abaetetuba	PA
5	abaiara	CE
6	abaira	BA
7	abare	BA
8	abatia	PR
9	abdon batista	SC
10	abelardo luz	SC
11	abrantes	BA
12	abre campo	MG

Insights

Time periods are slights different in tables but the data is mainly from 2016 September to 2018 October.

Category 2

In-depth Exploration:

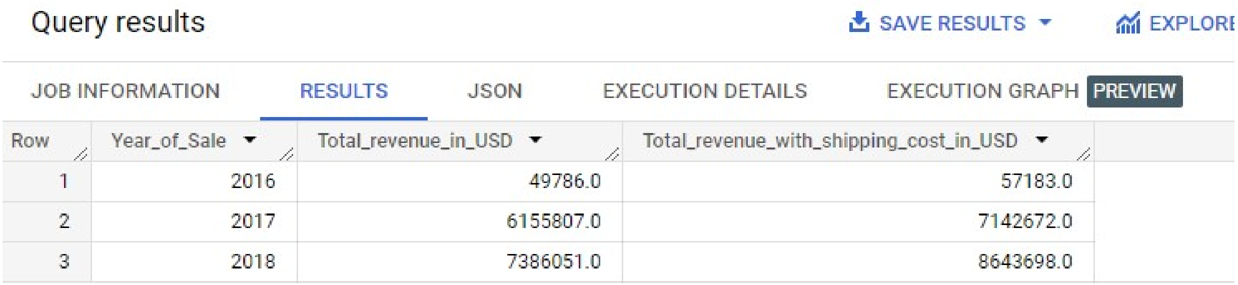
1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

A.The growth of e commerce in the given period of time (year on year)

Query

```
select
distinct t.Year_of_Sale,
round(sum(t.price) over(partition by t.Year_of_Sale)) as Total_revenue_in_USD,
round(sum(t.Price_with_shipping)       over(partition       by       t.Year_of_Sale))       as
Total_revenue_with_shipping_cost_in_USD
from
(select
*,
extract(year from od.order_purchase_timestamp) as Year_of_Sale,
extract(month from od.order_purchase_timestamp) as Month_of_Sale,
ore.price+ore.freight_value as Price_with_shipping
from           `Bigquery_project_marsh050223.orders`          od           left           join
`Bigquery_project_marsh050223.order_items` ore
                                                          on od.order_id=ore.order_id
                                                                                           left
join`Bigquery_project_marsh050223.customers` cr  on od.
                                                          customer_id=cr.customer_id
)t
order by t.Year_of_Sale;
```

Result



Row	Year_of_Sale	Total_revenue_in_USD	Total_revenue_with_shipping_cost_in_USD
1	2016	49786.0	57183.0
2	2017	6155807.0	7142672.0
3	2018	7386051.0	8643698.0

Insight

The sale is increasing year by year as per the given data set. The e commerce is booming as per the data, so more customers will be opting for online sale. We need to forecast the demand and more sellers need to be introduced to the e commerce business.

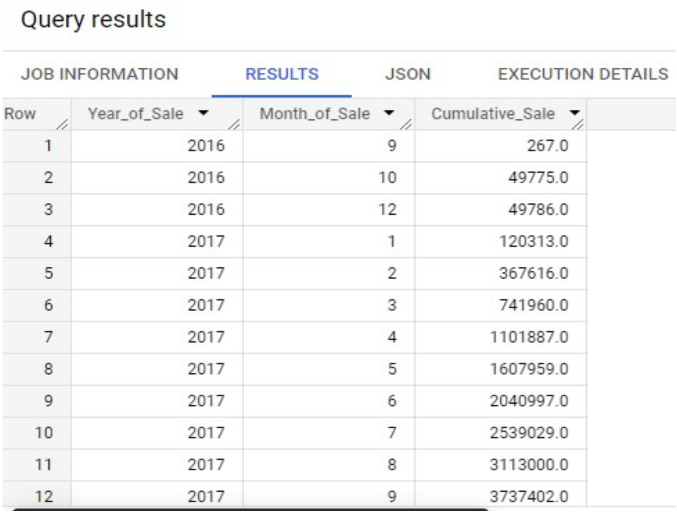
B. Cumulative Sale- MOM

Query

```
select
distinct t.Year_of_Sale,
t.Month_of_Sale,
round(sum(t.price)  over(partition  by  t.Year_of_Sale  order  by  t.Month_of_Sale  range  between
unbounded preceding and current row)) as Cumulative_Sale
from
(select
*,
extract(year from od.order_purchase_timestamp) as Year_of_Sale,
extract(month from od.order_purchase_timestamp) as Month_of_Sale,
ore.price+ore.freight_value as Price_with_shipping,
extract(time from od.order_purchase_timestamp) as Order_time

from          `Bigquery_project_marsh050223.orders`          od          left          join
`Bigquery_project_marsh050223.order_items` ore
                                                          on od.order_id=ore.order_id
                                                                                      left
join`Bigquery_project_marsh050223.customers` cr  on od.
                                                          customer_id=cr.customer_id
)t
order by t.Year_of_Sale,t.Month_of_Sale;
```

Result



Query results

Row	Year_of_Sale	Month_of_Sale	Cumulative_Sale
1	2016	9	267.0
2	2016	10	49775.0
3	2016	12	49786.0
4	2017	1	120313.0
5	2017	2	367616.0
6	2017	3	741960.0
7	2017	4	1101887.0
8	2017	5	1607959.0
9	2017	6	2040997.0
10	2017	7	2539029.0
11	2017	8	3113000.0
12	2017	9	3737402.0

C. Peak months of Sale

Query

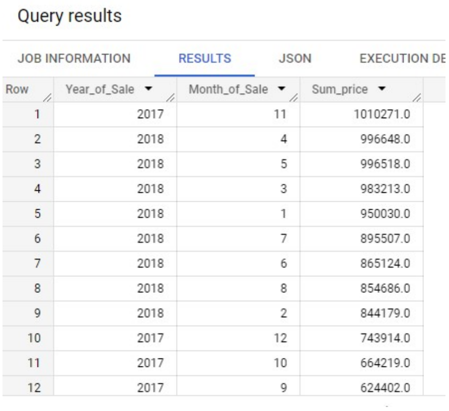
```
select
*
from
(

select
distinct t.Year_of_Sale,
t.Month_of_Sale,
round(sum(t.price) over(partition by t.Year_of_Sale,T.Month_of_Sale)) as Sum_price
from
(select
*,
extract(year from od.order_purchase_timestamp) as Year_of_Sale,
extract(month from od.order_purchase_timestamp) as Month_of_Sale,
ore.price+ore.freight_value as Price_with_shipping,
extract(time from od.order_purchase_timestamp) as Order_time

from `Bigquery_project_marsh050223.orders` od left join
`Bigquery_project_marsh050223.order_items` ore
```

```
                                                              on od.order_id=ore.order_id
                                                                                           left
join`Bigquery_project_marsh050223.customers` cr  on od.
                                                              customer_id=cr.customer_id
)t
)
order by Sum_price desc
```

Result



Query results

Row	Year_of_Sale	Month_of_Sale	Sum_price
1	2017	11	1010271.0
2	2018	4	996648.0
3	2018	5	996518.0
4	2018	3	983213.0
5	2018	1	950030.0
6	2018	7	895507.0
7	2018	6	865124.0
8	2018	8	854686.0
9	2018	2	844179.0
10	2017	12	743914.0
11	2017	10	664219.0
12	2017	9	624402.0

The Peak month is 2017 November, thanks giving festival and starting of summer in Brazil might be the reason of increase in Sale.

Category 3

Evolution of E-commerce orders in the Brazil region:

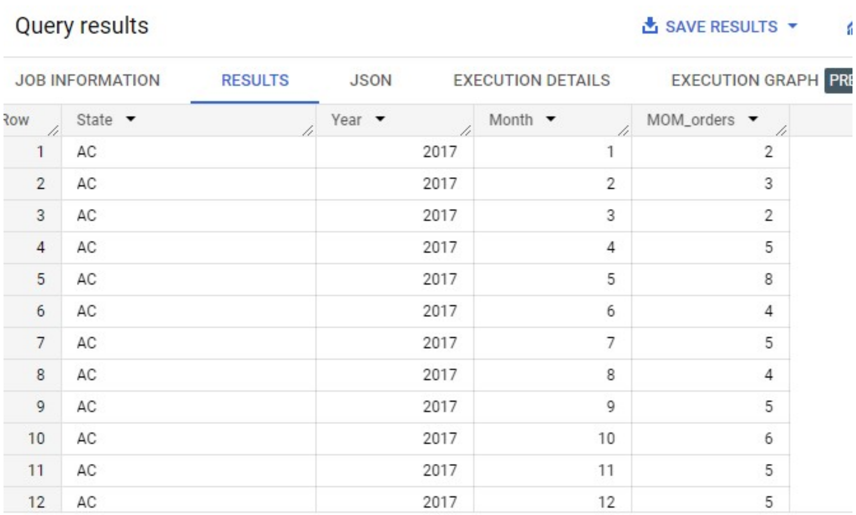
1.Get month on month orders by states

Query

```
 select

distinct t.customer_state as State,
t.Year_of_Sale as Year,
t.Month_of_Sale as Month,
count(distinct  t.order_id)  over(partition  by  t.Year_of_Sale,t.Month_of_Sale,t.customer_state)
as MOM_orders
from
(
 select
od.order_id,
cr.customer_unique_id,
cr.customer_state,
extract(year from od.order_purchase_timestamp) as Year_of_Sale,
extract(month from od.order_purchase_timestamp) as Month_of_Sale,
ore.price+ore.freight_value as Price_with_shipping,
extract(time from od.order_purchase_timestamp) as Order_time,
od.order_purchase_timestamp,
min(od.order_purchase_timestamp)over    (partition    by    cr.customer_unique_id    order    by
od.order_purchase_timestamp  rows  between  unbounded  preceding  and  unbounded  following)  as
First_order_date
 from          `Bigquery_project_marsh050223.orders`          od          left          join
`Bigquery_project_marsh050223.order_items` ore
                                                          on od.order_id=ore.order_id
                                                                                       left
join`Bigquery_project_marsh050223.customers` cr  on od.
                                                          customer_id=cr.customer_id
where lower(od.order_status)!='canceled'
)t
 order by 1,2,3;
```

Result



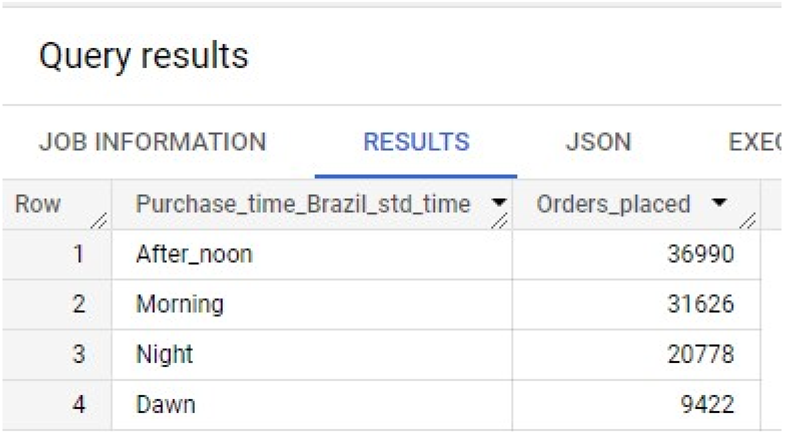
Row	State	Year	Month	MOM_orders
1	AC	2017	1	2
2	AC	2017	2	3
3	AC	2017	3	2
4	AC	2017	4	5
5	AC	2017	5	8
6	AC	2017	6	4
7	AC	2017	7	5
8	AC	2017	8	4
9	AC	2017	9	5
10	AC	2017	10	6
11	AC	2017	11	5
12	AC	2017	12	5

2.What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Query

```
select
t.Purchase_time_Brazil_std_time,
count(distinct t.order_id) as Orders_placed
from(
select
order_id,
order_purchase_timestamp,
case when time_sub(TIME (extract(time from order_purchase_timestamp)), interval 3 HOUR)
between '03:00:00' and '07:00:00' then'Dawn '
when time_sub(TIME (extract(time from order_purchase_timestamp)), interval 3 HOUR) between
'07:00:01' and '11:59:59' then 'Morning'
when time_sub(TIME (extract(time from order_purchase_timestamp)), interval 3 HOUR) between
'12:00:00' and '18:00:00' then 'After_noon'
else 'Night'
end as Purchase_time_Brazil_std_time
from `Bigquery_project_marsh050223.orders`
where lower(order_status)!="canceled"
order by order_purchase_timestamp desc
)t
group by t.Purchase_time_Brazil_std_time
order by count(distinct order_id) desc;
```

Result



Row	Purchase_time_Brazil_std_time	Orders_placed
1	After_noon	36990
2	Morning	31626
3	Night	20778
4	Dawn	9422

Insight

We have calculated the orders based on the Brazilia timing which is 3 hours behind UTC.The customers are placing orders mainly during afternoon and morning. So during these timing more adds need to played on the app/ or other available screens which may push the sale up.

Category 4

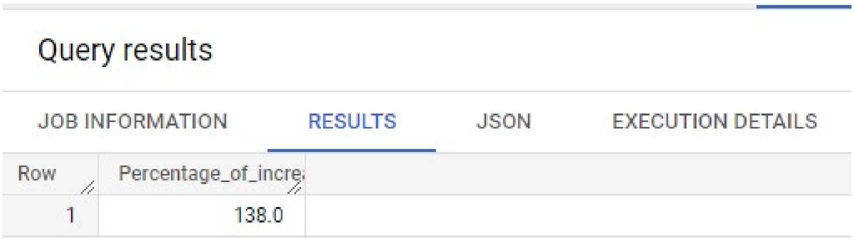
Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

Query

```
SELECT
round(((Next_year-Cost_of_Order) *100/cost_of_order)) as Percentage_of_increase_in_sale
from
(
 select
Year,
Cost_of_Order,
lead(Cost_of_order)over(order by Year) as Next_Year
from
(
select
distinct t.Year,
round(sum(t.payment_value)) as Cost_of_order
from
(
select
p.order_id,
o.order_status,
extract(year from o.order_purchase_timestamp) as Year,
extract(month from o.order_purchase_timestamp) as Month,
p.payment_value
from          `Bigquery_project_marsh050223.payments`          p           left          join
`Bigquery_project_marsh050223.orders` o
                                                      on p.order_id=o.order_id
)t
where t.order_status!='canceled' and t.Year in (2017,2018) and t.Month between 1 and 8
group by t.year
))
limit 1;
```

Result



Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS

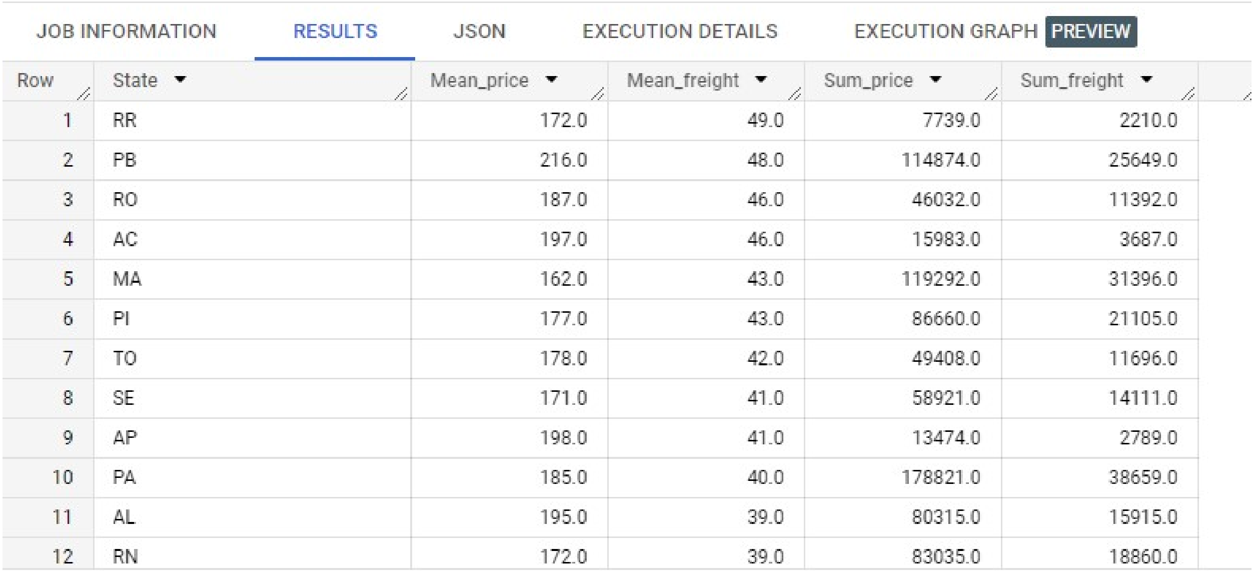
Row	Percentage_of_incre
1	138.0

2. Mean & Sum of price and freight value by customer state

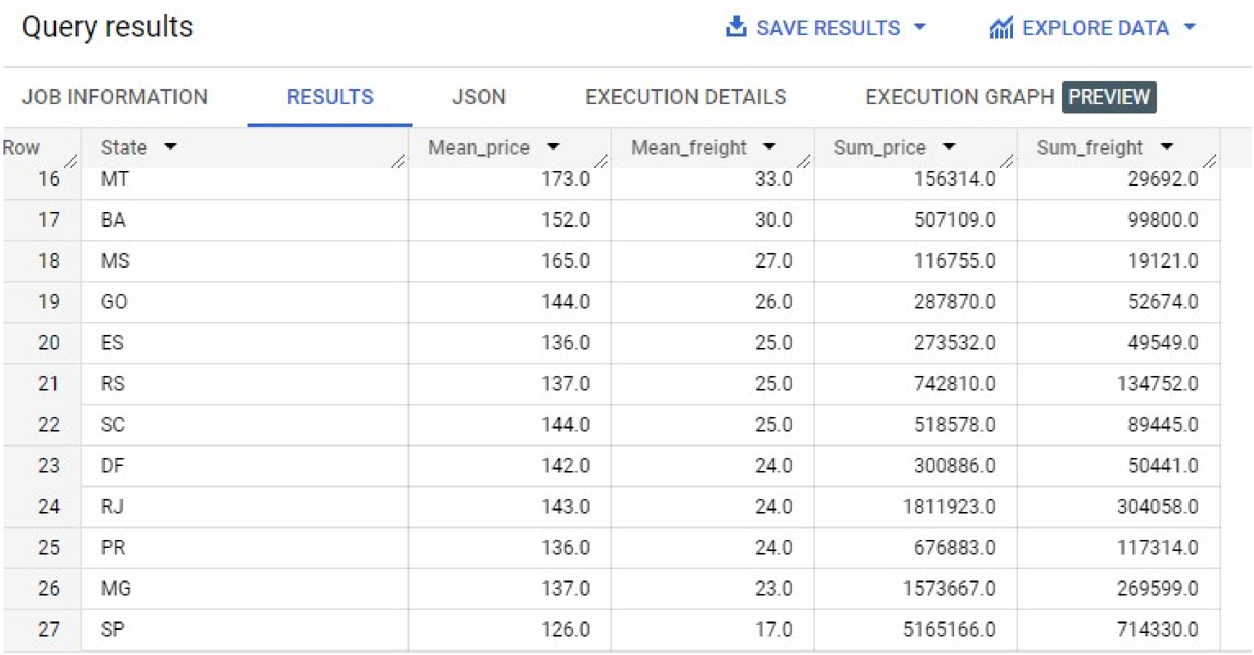
Query

```
select
distinct t.customer_state as State,
round(avg(t.price)over(partition by t.customer_state)) as Mean_price,
round(avg(t.freight_value)over(partition by t.customer_state)) as Mean_freight,
round(sum(t.price)over(partition by t.customer_state)) as Sum_price,
round(sum(t.freight_value)over(partition by t.customer_state)) as Sum_freight,
from
(
select
distinct od.order_id,
cr.customer_state,
sum(ore.freight_value) as freight_value,
sum(ore.price) as price,
from `Bigquery_project_marsh050223.orders` od left join
`Bigquery_project_marsh050223.order_items` ore
                                                          on od.order_id=ore.order_id
                                              left
join`Bigquery_project_marsh050223.customers` cr  on od.
                                                          customer_id=cr.customer_id
where lower(od.order_status)!='canceled'
group by od.order_id, cr.customer_state
)t
order by Mean_freight desc;
```

Result



Row	State	Mean_price	Mean_freight	Sum_price	Sum_freight
1	RR	172.0	49.0	7739.0	2210.0
2	PB	216.0	48.0	114874.0	25649.0
3	RO	187.0	46.0	46032.0	11392.0
4	AC	197.0	46.0	15983.0	3687.0
5	MA	162.0	43.0	119292.0	31396.0
6	PI	177.0	43.0	86660.0	21105.0
7	TO	178.0	42.0	49408.0	11696.0
8	SE	171.0	41.0	58921.0	14111.0
9	AP	198.0	41.0	13474.0	2789.0
10	PA	185.0	40.0	178821.0	38659.0
11	AL	195.0	39.0	80315.0	15915.0
12	RN	172.0	39.0	83035.0	18860.0



Row	State	Mean_price	Mean_freight	Sum_price	Sum_freight
16	MT	173.0	33.0	156314.0	29692.0
17	BA	152.0	30.0	507109.0	99800.0
18	MS	165.0	27.0	116755.0	19121.0
19	GO	144.0	26.0	287870.0	52674.0
20	ES	136.0	25.0	273532.0	49549.0
21	RS	137.0	25.0	742810.0	134752.0
22	SC	144.0	25.0	518578.0	89445.0
23	DF	142.0	24.0	300886.0	50441.0
24	RJ	143.0	24.0	1811923.0	304058.0
25	PR	136.0	24.0	676883.0	117314.0
26	MG	137.0	23.0	1573667.0	269599.0
27	SP	126.0	17.0	5165166.0	714330.0

Insight

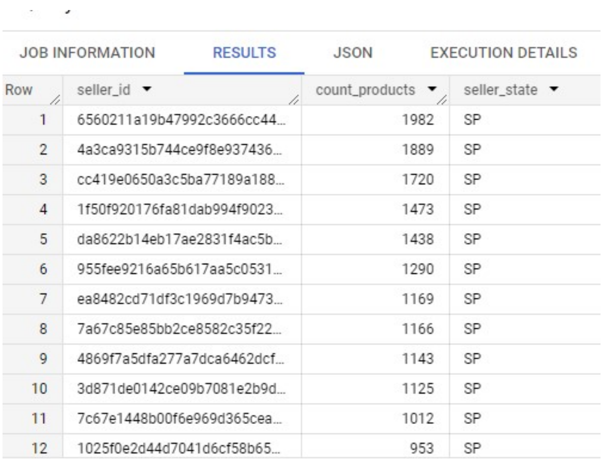
The freight cost is more in RR which is the state Roraima and state PB , which is the state Paraiba which are respectively the northernmost and easternmost states of Brazil.The RR state is full of Amzon rain forest and high terrain so transportation cost is more and the state PB is also very far from the main The least cost of freight is at Sao polo, the financial capital of brazil.

When we look at the seller details based on the orders,

Query

```
select
distinct seller_id,
count_products,
seller_state
from
(
select
count(product_id) over (partition by seller_id) as count_products,
seller_id,
seller_state
from
(
select
distinct o.order_id,
o.product_id,
o.seller_id,
s.seller_state,
from      `Bigquery_project_marsh050223.order_items`      o      inner
join`Bigquery_project_marsh050223.sellers` s
                                          on o.seller_id=s.seller_id
)
)
order by count_products desc
```

Result



Row	seller_id	count_products	seller_state
1	6560211a19b47992c3666cc44...	1982	SP
2	4a3ca9315b744ce9f8e937436...	1889	SP
3	cc419e0650a3c5ba77189a188...	1720	SP
4	1f50f920176fa81dab994f9023...	1473	SP
5	da8622b14eb17ae2831f4ac5b...	1438	SP
6	955fee9216a65b617aa5c0531...	1290	SP
7	ea8482cd71df3c1969d7b9473...	1169	SP
8	7a67c85e85bb2ce8582c35f22...	1166	SP
9	4869f7a5dfa277a7dca6462dcf...	1143	SP
10	3d871de0142ce09b7081e2b9d...	1125	SP
11	7c67e1448b00f6e969d365cea...	1012	SP
12	1025f0e2d44d7041d6cf58b65...	953	SP

Checking the sellers from state RR and PB

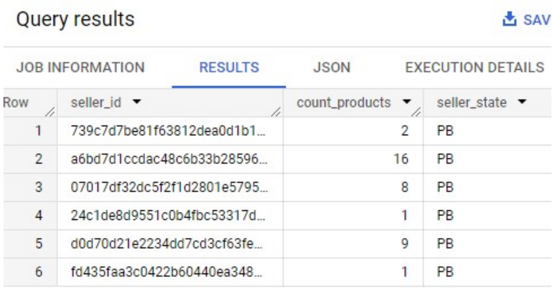
Query

```
select
distinct seller_id,
count_products,
seller_state
from
(
select
count(product_id) over (partition by seller_id) as count_products,
seller_id,
seller_state
from
(
select
distinct o.order_id,
o.product_id,
o.seller_id,
s.seller_state,
from        `Bigquery_project_marsh050223.order_items`        o        inner
join`Bigquery_project_marsh050223.sellers` s
```

```
                                             on o.seller_id=s.seller_id

)
)
where seller_state in ('RR','PB')
```

Result



Row	seller_id	count_products	seller_state
1	739c7d7be81f63812dea0d1b1...	2	PB
2	a6bd7d1ccdac48c6b33b28596...	16	PB
3	07017df32dc5f2f1d2801e5795...	8	PB
4	24c1de8d9551c0b4fbc53317d...	1	PB
5	d0d70d21e2234dd7cd3cf63fe...	9	PB
6	fd435faa3c0422b60440ea348...	1	PB

The top selling sellers are based out of SP , so the average freight cost is lesser. Only six sellers are from PB and there are no sellers from RR which is the reason for the high average freight costs which indicates customer from these states are mainly buying from sellers from other states. So Target should try to find some sellers is RR, PB or nearby states which might bring down the freight cost to these states.

Map of Brazil

Medellín
Venezuela
Bogotá
Guyana
French Guiana
Cali
Colombia
STATE OF RORAIMA
STATE OF AMAPÁ
Quito
Ecuador
Manaus
STATE OF AMAZONAS
STATE OF PARÁ
STATE OF MARANHÃO
STATE OF CEARÁ
STATE OF PARAÍBA
STATE OF PIAUÍ
Brazil
STATE OF ACRE
Peru
STATE OF ALAGOAS
STATE OF RONDÔNIA
STATE OF TOCANTINS
STATE OF SERGIPE
Lima
STATE OF MATO GROSSO
STATE OF BAHIA
Brasília
La Paz
Bolivia
Goiânia
STATE OF MINAS GERAIS
STATE OF ESPÍRITO SANTO
STATE OF MATO GROSSO DO SUL
Belo Horizonte
STATE OF SÃO PAULO
STATE OF RIO DE JANEIRO
Paraguay
Asunción
STATE OF PARANÁ
Chile
STATE OF SANTA CATARINA
STATE OF RIO GRANDE DO SUL
Córdoba
Uruguay
Santiago
Buenos Aires
Argentina
Google

Category 5

Analysis on sales, freight and delivery time

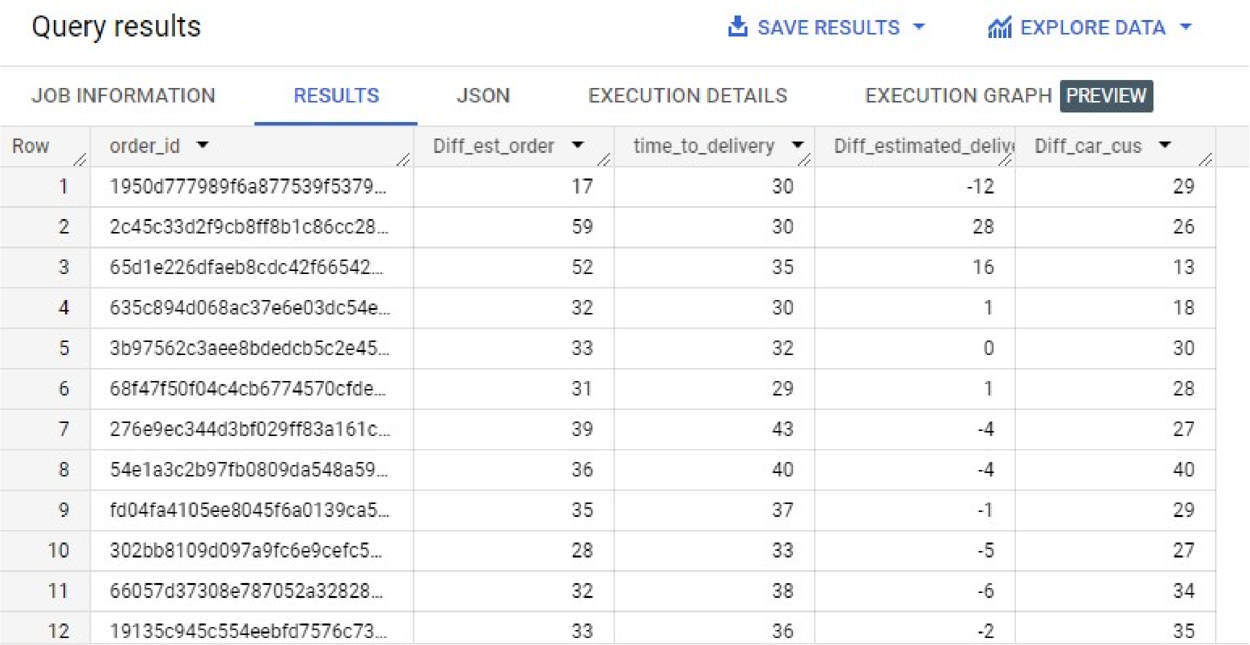
1.Calculate days between purchasing, delivering and estimated delivery

Query

```
select
order_id,
date_diff(order_estimated_delivery_date,order_purchase_timestamp,day) as Diff_est_order,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_delivery,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day)                as
Diff_estimated_delivery,
date_diff(order_delivered_customer_date,order_delivered_carrier_date,day) as Diff_car_cus
```

```
from `Bigquery_project_marsh050223.orders`
```

Result



Row	order_id	Diff_est_order	time_to_delivery	Diff_estimated_deliv	Diff_car_cus
1	1950d777989f6a877539f5379...	17	30	-12	29
2	2c45c33d2f9cb8ff8b1c86cc28...	59	30	28	26
3	65d1e226dfaeb8cdc42f66542...	52	35	16	13
4	635c894d068ac37e6e03dc54e...	32	30	1	18
5	3b97562c3aee8bdedcb5c2e45...	33	32	0	30
6	68f47f50f04c4cb6774570cfde...	31	29	1	28
7	276e9ec344d3bf029ff83a161c...	39	43	-4	27
8	54e1a3c2b97fb0809da548a59...	36	40	-4	40
9	fd04fa4105ee8045f6a0139ca5...	35	37	-1	29
10	302bb8109d097a9fc6e9cefc5...	28	33	-5	27
11	66057d37308e787052a32828...	32	38	-6	34
12	19135c945c554eebfd7576c73...	33	36	-2	35

2.Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

time_to_delivery = order_delivered_customer_date-order_purchase_timestamp

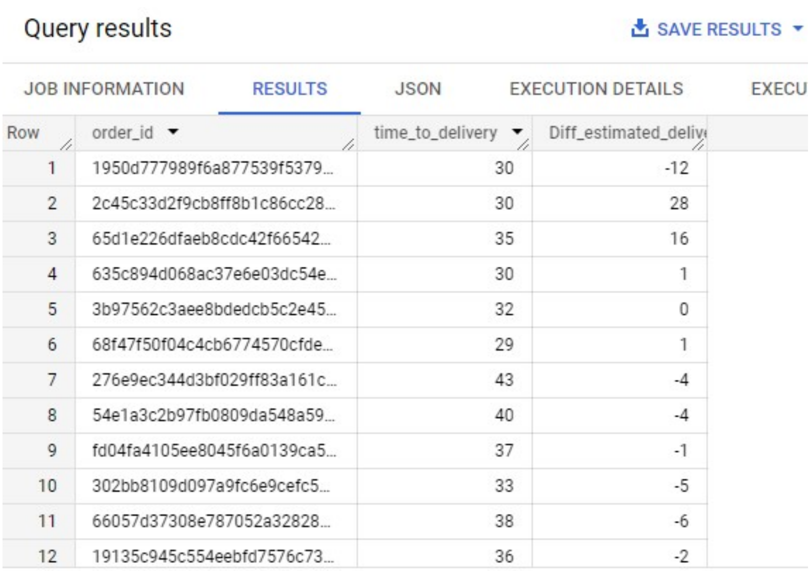
diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

Query

```
select
order_id,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_delivery,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day)            as
Diff_estimated_delivery,

from `Bigquery_project_marsh050223.orders`
```

Result

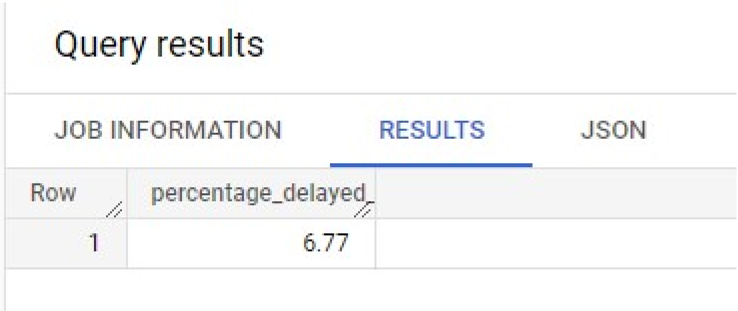


Row	order_id	time_to_delivery	Diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	33	-5
11	66057d37308e787052a32828...	38	-6
12	19135c945c554eebfd7576c73...	36	-2

Query

```
select
round(countif(Diff_estimated_delivery<0)*100/count(order_id),2) as percentage_delayed_orders
from
(
select
order_id,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_delivery,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day)            as
Diff_estimated_delivery,
from `Bigquery_project_marsh050223.orders`
where date_diff(order_estimated_delivery_date,order_delivered_customer_date,day) is not null
)
```

Result



Row	percentage_delayed
1	6.77

Insights

6.77% of the orders got delivered after the estimated time which needs to come down.

3.Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

Query

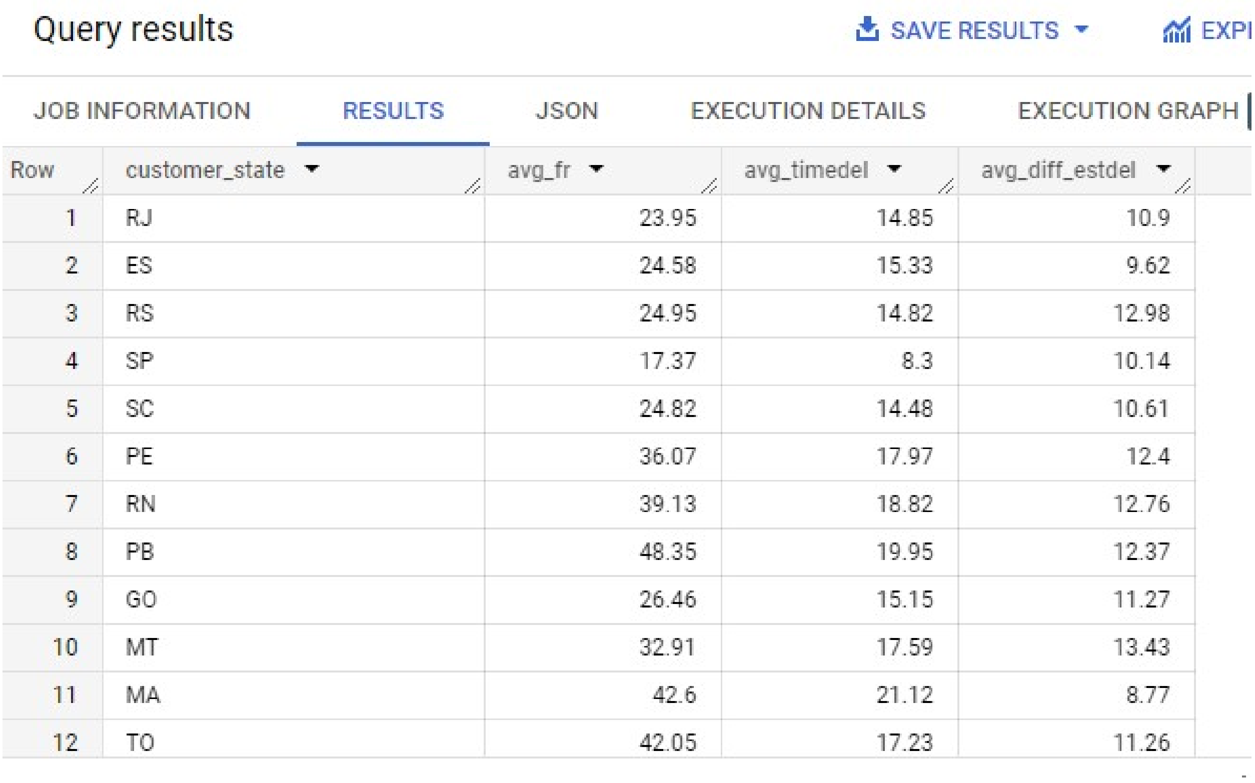
```
select
customer_state,
round(avg(fre_val),2) as avg_fr,
round(avg(time_to_delivery),2) as avg_timedel,
round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)            as
Diff_estimated_delivery,
from             `Bigquery_project_marsh050223.orders`           o           left           join
`Bigquery_project_marsh050223.order_items`  t on
                                                    o.order_id=t.order_id
                                                                                   left   join
`Bigquery_project_marsh050223.customers`c on

                                                    o.customer_id=c.customer_id

))
group by customer_state
```

Result



Query results

Row	customer_state	avg_fr	avg_timedel	avg_diff_estdel
1	RJ	23.95	14.85	10.9
2	ES	24.58	15.33	9.62
3	RS	24.95	14.82	12.98
4	SP	17.37	8.3	10.14
5	SC	24.82	14.48	10.61
6	PE	36.07	17.97	12.4
7	RN	39.13	18.82	12.76
8	PB	48.35	19.95	12.37
9	GO	26.46	15.15	11.27
10	MT	32.91	17.59	13.43
11	MA	42.6	21.12	8.77
12	TO	42.05	17.23	11.26

4. Sort the data to get the following:

1. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

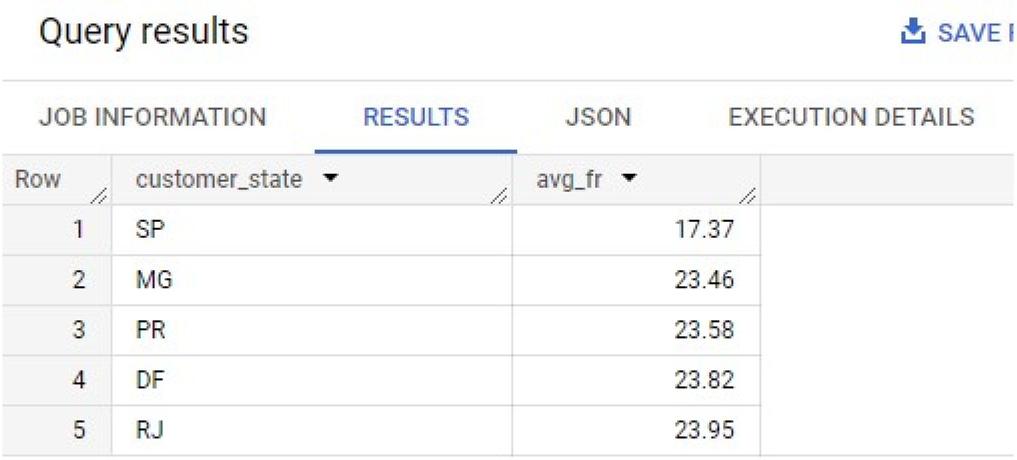
Query- lowest 5 states

```
select

customer_state,
round(avg(fre_val),2) as avg_fr,
#round(avg(time_to_delivery),2) as avg_timedel,
#round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)            as
Diff_estimated_delivery,
from         `Bigquery_project_marsh050223.orders`         o         left          join
`Bigquery_project_marsh050223.order_items` t on
                                                  o.order_id=t.order_id
                                                                                    left   join
`Bigquery_project_marsh050223.customers`c on
                                                  o.customer_id=c.customer_id
))
group by customer_state
order by avg(fre_val)
limit 5;
```

Result



Row	customer_state	avg_fr
1	SP	17.37
2	MG	23.46
3	PR	23.58
4	DF	23.82
5	RJ	23.95

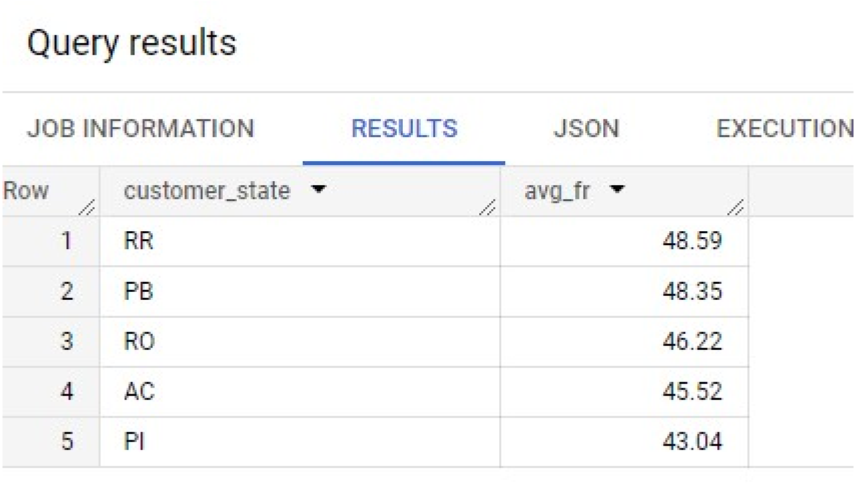
Query – Top 5 States

```
select
customer_state,
round(avg(fre_val),2) as avg_fr,
#round(avg(time_to_delivery),2) as avg_timedel,
#round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day) as
Diff_estimated_delivery,
from `Bigquery_project_marsh050223.orders` o left join
`Bigquery_project_marsh050223.order_items` t on
o.order_id=t.order_id
left join
`Bigquery_project_marsh050223.customers`c on
```

```
                                                  o.customer_id=c.customer_id
))
group by customer_state
order by avg(fre_val) desc
limit 5;
```

Result



Query results

Row	customer_state	avg_fr
1	RR	48.59
2	PB	48.35
3	RO	46.22
4	AC	45.52
5	PI	43.04

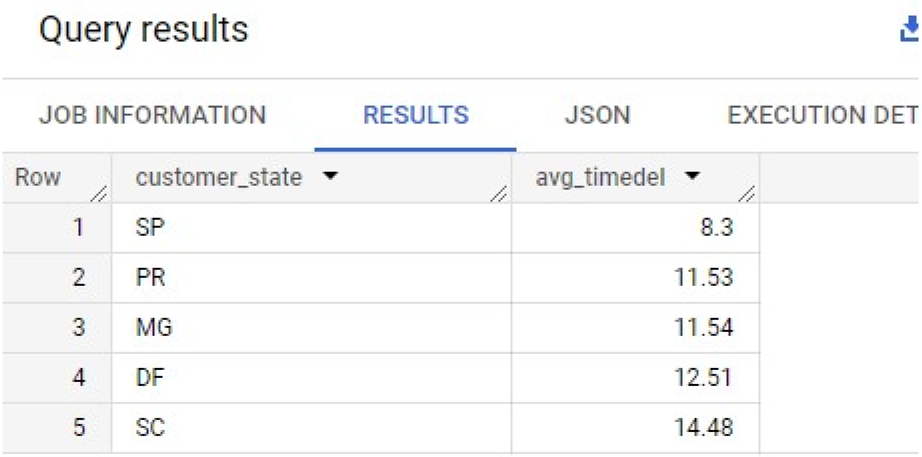
Top 5 states with highest/lowest average time to delivery

Query – lowest five states

```
select
customer_state,
#round(avg(fre_val),2) as avg_fr,
round(avg(time_to_delivery),2) as avg_timedel,
#round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)             as
Diff_estimated_delivery,
from            `Bigquery_project_marsh050223.orders`           o           left           join
`Bigquery_project_marsh050223.order_items`  t on
                                                  o.order_id=t.order_id
                                                                               left   join
`Bigquery_project_marsh050223.customers`c on
                                                  o.customer_id=c.customer_id
))
group by customer_state
order by avg(time_to_delivery)
limit 5;
```

Result



Query results

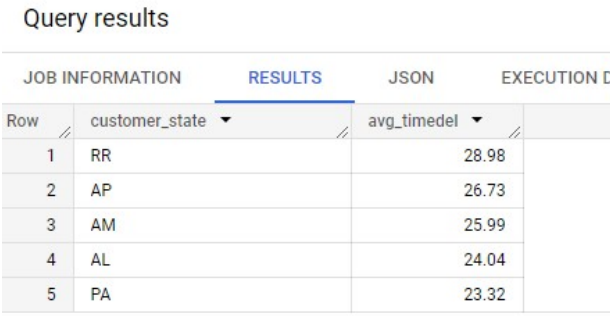
Row	customer_state	avg_timedel
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

Query top five states

```
select
customer_state,
#round(avg(fre_val),2) as avg_fr,
round(avg(time_to_delivery),2) as avg_timedel,
#round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)          as
Diff_estimated_delivery,
from        `Bigquery_project_marsh050223.orders`         o         left         join
`Bigquery_project_marsh050223.order_items` t on
                                                  o.order_id=t.order_id
                                                                                left   join
`Bigquery_project_marsh050223.customers`c on
                                                  o.customer_id=c.customer_id
))
group by customer_state
order by avg(time_to_delivery) desc
limit 5;
```

Result



Row	customer_state	avg_timedel
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

Top 5 states where delivery is really fast/ not so fast compared to estimated date

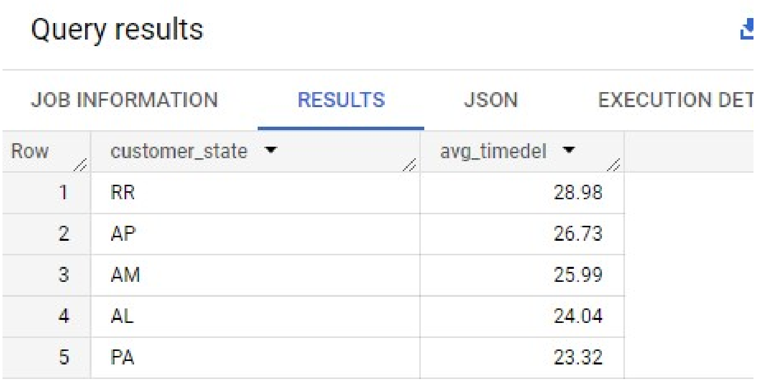
Query top 5 states with fastest delivery compared to estimated delivery

```
select
customer_state,
#round(avg(fre_val),2) as avg_fr,
#round(avg(time_to_delivery),2) as avg_timedel,
round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)           as
Diff_estimated_delivery,
from `Bigquery_project_marsh050223.orders` o left join
`Bigquery_project_marsh050223.order_items` t on
                                                  o.order_id=t.order_id
```

```
                                                                    left   join
`Bigquery_project_marsh050223.customers`c on
                                           o.customer_id=c.customer_id
))
group by customer_state
order by avg(Diff_estimated_delivery) desc
limit 5;
```

Result



Query results

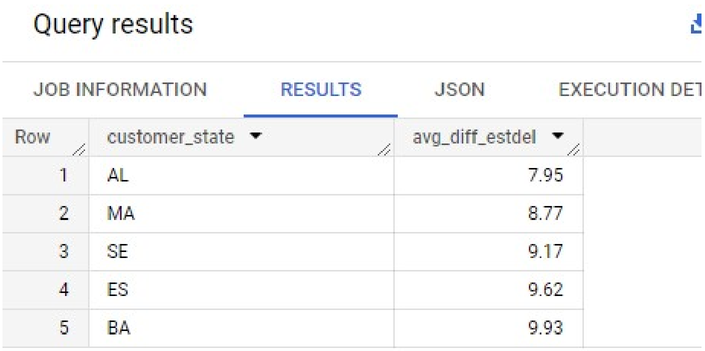
Row	customer_state	avg_timedel
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

Query -five states with lowest delivery time compared to estimated delivery

```
select
customer_state,
#round(avg(fre_val),2) as avg_fr,
#round(avg(time_to_delivery),2) as avg_timedel,
round(avg(Diff_estimated_delivery),2) as avg_diff_estdel,
from
(
select
distinct order_id,
sum(freight_value) over (partition by order_id) as fre_val,
customer_state,
time_to_delivery,
Diff_estimated_delivery
from
(

select
o.order_id,
t.freight_value,
c.customer_state,
date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day) as time_to_delivery,
date_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,day)            as
Diff_estimated_delivery,
from           `Bigquery_project_marsh050223.orders`          o          left          join
`Bigquery_project_marsh050223.order_items` t on
                                                  o.order_id=t.order_id
                                                                              left   join
`Bigquery_project_marsh050223.customers`c on
                                                  o.customer_id=c.customer_id
))
group by customer_state
order by avg(Diff_estimated_delivery)
limit 5;
```

Result



Query results

Row	customer_state	avg_diff_estdel
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

Category 6 Payment Type Analysis

1.Month over Month count of orders for different payment types

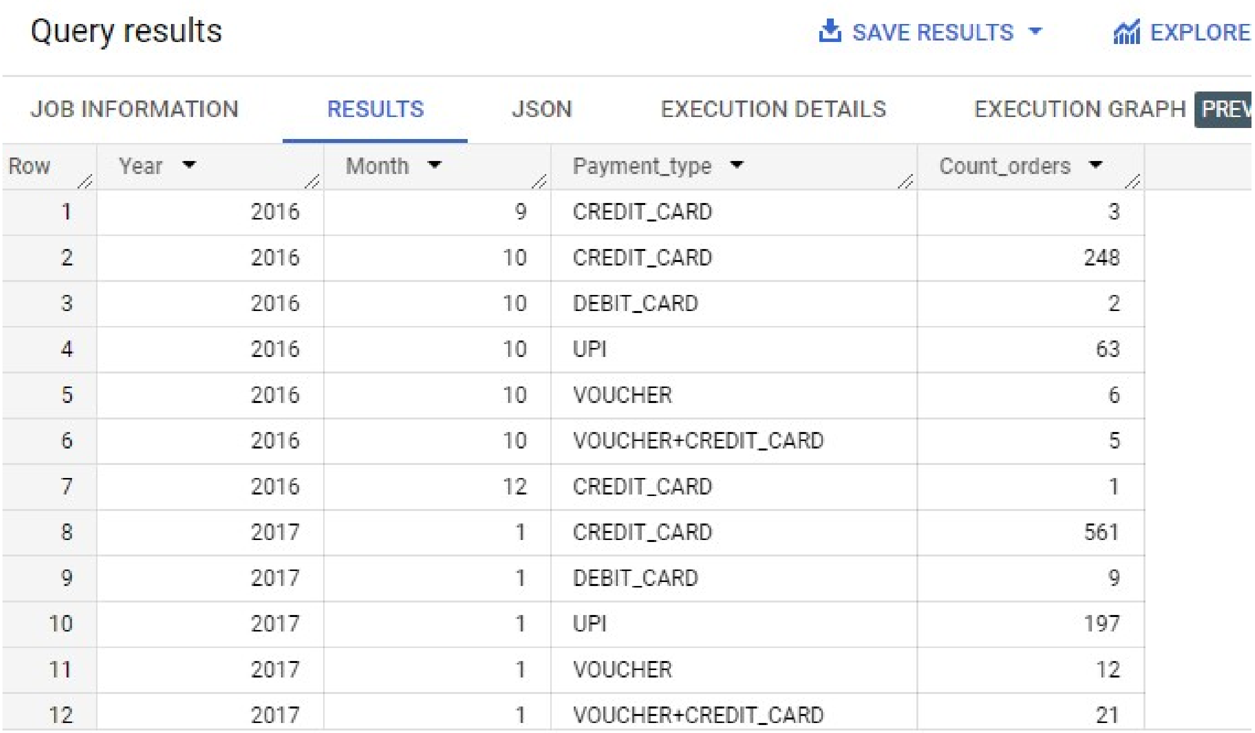
Query:

```
select
Year,
Month,
Payment_type,
count(order_id) as Count_orders,
from
(
select
extract(year from order_purchase_timestamp) as Year,
extract(month from order_purchase_timestamp) as Month,
order_id,
case when UPI>0 and Voucher>0 and credit_card>0 and debit_card>0 then "all"
          when UPI>0 and Voucher>0 and credit_card>0 and debit_card=0 then
'UPI+VOUCHER+CREDIT_CARD'
          when UPI>0 and Voucher>0 and credit_card=0 and debit_card>0 then
'UPI+VOUCHER+CREDIT_CARD'
          when UPI>0 and Voucher=0 and credit_card>0 and debit_card>0 then
'UPI+CREDIT_CARD+DEBIT_CARD'
          when UPI=0 and Voucher>0 and credit_card>0 and debit_card>0 then
'VOUCHER+CREDIT_CARD+DEBIT_CARD'
     when UPI>0 and Voucher>0 and credit_card=0 and debit_card=0 then 'UPI+VOUCHER'
     when UPI>0 and Voucher=0 and credit_card>0 and debit_card=0 then 'UPI+CREDIT_CARD'
     when UPI>0 and Voucher=0 and credit_card=0 and debit_card>0 then 'UPI+DEBIT_CARD'
     when UPI=0 and Voucher>0 and credit_card>0 and debit_card=0 then 'VOUCHER+CREDIT_CARD'
     when UPI=0 and Voucher>0 and credit_card=0 and debit_card>0 then 'VOUCHER+DEBIT_CARD'
     when UPI=0 and Voucher=0 and credit_card>0 and debit_card>0 then 'DEBIT_CARD+CREDIT_CARD'
     when UPI>0 and Voucher=0 and credit_card=0 and debit_card=0 then 'UPI'
     when UPI=0 and Voucher>0 and credit_card=0 and debit_card=0 then 'VOUCHER'
     when UPI=0 and Voucher=0 and credit_card>0 and debit_card=0 then 'CREDIT_CARD'
     when UPI=0 and Voucher=0 and credit_card=0 and debit_card>0 then 'DEBIT_CARD'
     ELSE 'NA'
     END AS PAYMENT_TYPE

     from
     (
select
order_id,
order_purchase_timestamp,
sum(UPI) as UPI,
sum(Voucher) as Voucher,
sum(credit_card) as credit_card,
sum(debit_card) as debit_card,
sum(not_defined) as not_define_new
from(

select
p.order_id,
o.order_purchase_timestamp,
case when lower(p.payment_type)="upi" then 1 else 0 end as UPI,
case when lower(p.payment_type)="voucher" then 1 else 0 end as Voucher,
case when lower(p.payment_type)="credit_card" then 1 else 0 end as credit_card,
case when lower(p.payment_type)="debit_card" then 1 else 0 end as debit_card,
case when lower(p.payment_type)="not_defined" then 1 else 0 end as not_defined,
from `Bigquery_project_marsh050223.payments` p left join `Bigquery_project_marsh050223.orders`
o on p.order_id=o.order_id
)
group by order_id,order_purchase_timestamp
     )
)   group by Year,Month,payment_type
order by Year,Month,payment_type;
```

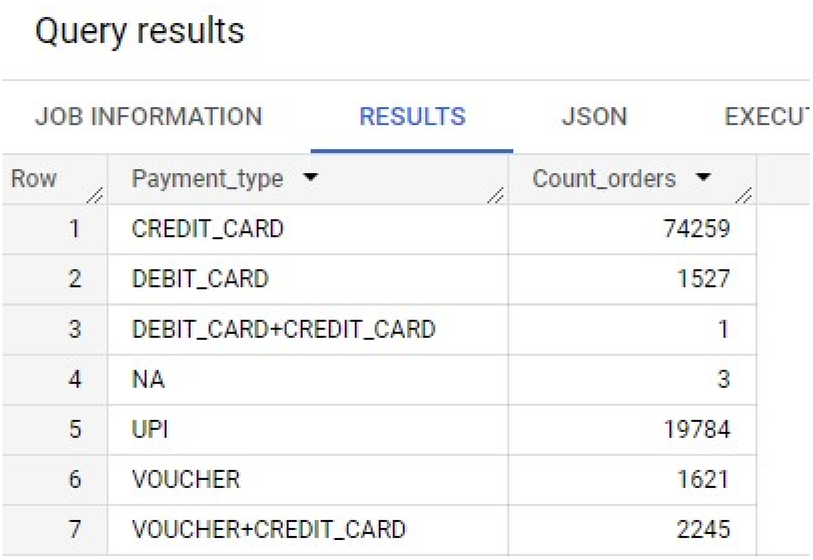
Result



Row	Year	Month	Payment_type	Count_orders
1	2016	9	CREDIT_CARD	3
2	2016	10	CREDIT_CARD	248
3	2016	10	DEBIT_CARD	2
4	2016	10	UPI	63
5	2016	10	VOUCHER	6
6	2016	10	VOUCHER+CREDIT_CARD	5
7	2016	12	CREDIT_CARD	1
8	2017	1	CREDIT_CARD	561
9	2017	1	DEBIT_CARD	9
10	2017	1	UPI	197
11	2017	1	VOUCHER	12
12	2017	1	VOUCHER+CREDIT_CARD	21

Insights:

Credit card is the most popular payment option in the given period of time followed by the UPI. Three cases are there where payment type in not defined and a few customers used multiple payment options together to complete the transaction.



Row	Payment_type	Count_orders
1	CREDIT_CARD	74259
2	DEBIT_CARD	1527
3	DEBIT_CARD+CREDIT_CARD	1
4	NA	3
5	UPI	19784
6	VOUCHER	1621
7	VOUCHER+CREDIT_CARD	2245

2.Count of orders based on the no. of payment installments

Query

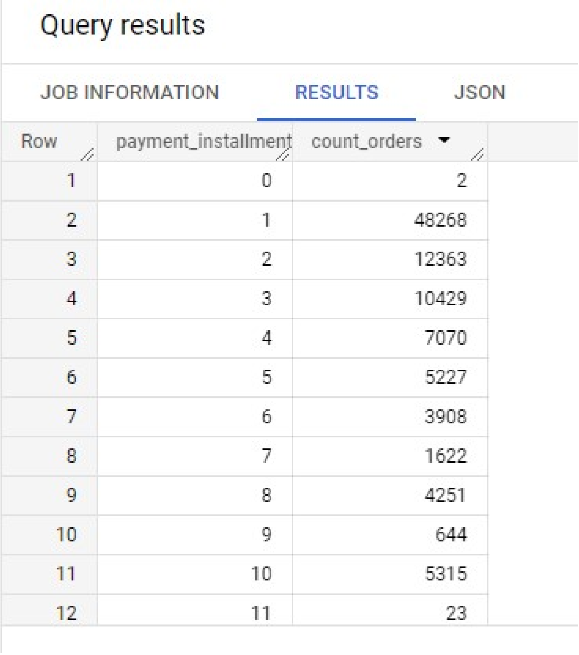
```
select
distinct payment_installments,
count(order_id) as count_orders
from
```

```
(
select
order_id,
payment_installments,
nos,
max_install
from(

select
order_id,
payment_installments,
nos,
max(nos) over(partition by order_id) as max_install
from(
select
order_id,
payment_installments,
row_number()over(partition by order_id) as nos
from `Bigquery_project_marsh050223.payments`

)
)
where nos=max_install
)
group by payment_installments
order by payment_installments
```

Result



Query results

Row	payment_installment	count_orders
1	0	2
2	1	48268
3	2	12363
4	3	10429
5	4	7070
6	5	5227
7	6	3908
8	7	1622
9	8	4251
10	9	644
11	10	5315
12	11	23

Fast moving products in the given time period

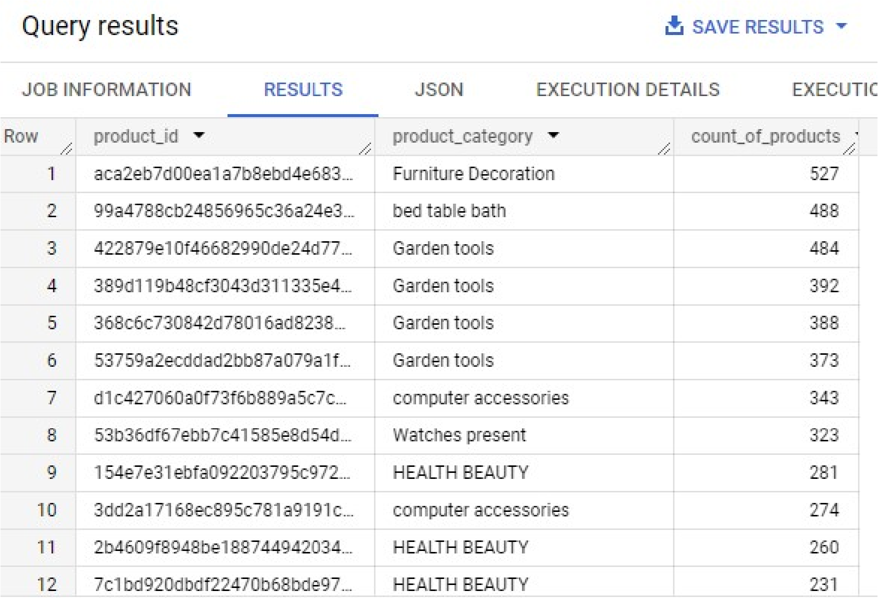
Query

```
select

*
from
(
select
distinct product_id,
product_category,
count(order_item_id) over(partition by product_id) as count_of_products
from
(
SELECT
o.order_id,
o.order_item_id,
o.product_id,
p.product_category,
p.product_name_length
from `Bigquery_project_marsh050223.order_items` o left join `Bigquery_project_marsh050223.products`
p
                                                on o.product_id=p.product_id

)
)
order by count_of_products desc;
```

Result



Query results

Row	product_id	product_category	count_of_products
1	aca2eb7d00ea1a7b8ebd4e683...	Furniture Decoration	527
2	99a4788cb24856965c36a24e3...	bed table bath	488
3	422879e10f46682990de24d77...	Garden tools	484
4	389d119b48cf3043d311335e4...	Garden tools	392
5	368c6c730842d78016ad8238...	Garden tools	388
6	53759a2ecddad2bb87a079a1f...	Garden tools	373
7	d1c427060a0f73f6b889a5c7c...	computer accessories	343
8	53b36df67ebb7c41585e8d54d...	Watches present	323
9	154e7e31ebfa092203795c972...	HEALTH BEAUTY	281
10	3dd2a17168ec895c781a9191c...	computer accessories	274
11	2b4609f8948be188744942034...	HEALTH BEAUTY	260
12	7c1bd920dbdf22470b68bde97...	HEALTH BEAUTY	231

Seller with avg review less than 2

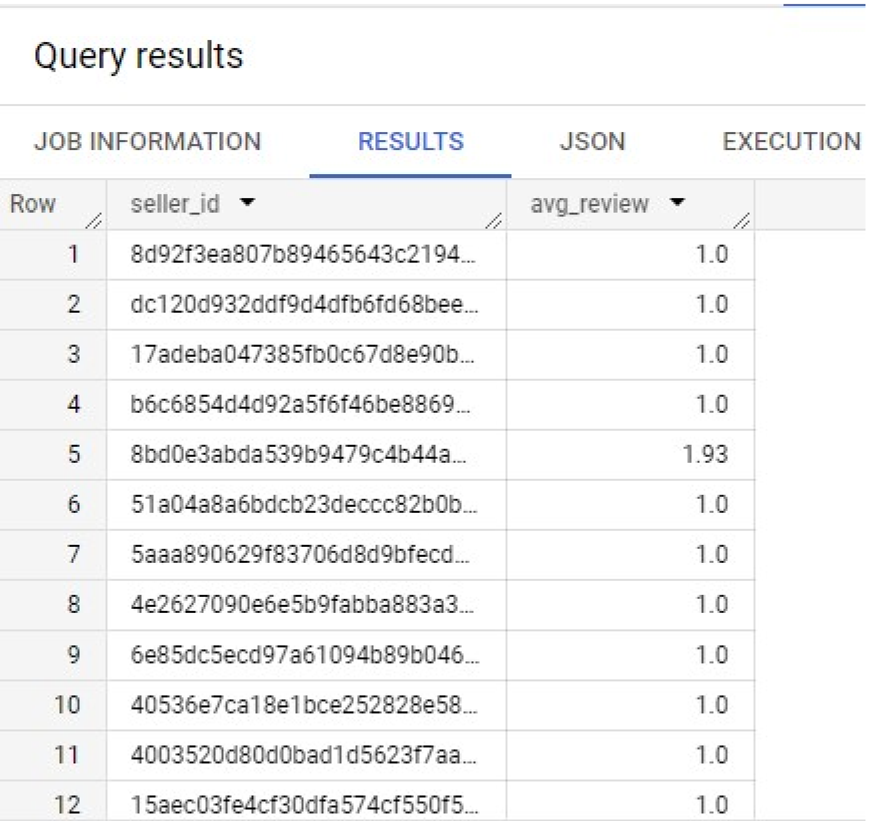
Query

```
select
*
from
(
select
distinct seller_id,
round(avg(review_score)over(partition  by  seller_id),2)  as
avg_review
from
(
select
o.order_id,
ot.product_id,
ov.review_score,
ot.seller_id
from  `Bigquery_project_marsh050223.orders`  o  left  join
`Bigquery_project_marsh050223.order_reviews`      ov      on
o.order_id=ov.order_id
                                                 left join
`Bigquery_project_marsh050223.order_items` ot on

 o.order_id=ot.order_id

where review_score is not null
)
)
where avg_review <2
```

Result



Row	seller_id	avg_review
1	8d92f3ea807b89465643c2194...	1.0
2	dc120d932ddf9d4dfb6fd68bee...	1.0
3	17adeba047385fb0c67d8e90b...	1.0
4	b6c6854d4d92a5f6f46be8869...	1.0
5	8bd0e3abda539b9479c4b44a...	1.93
6	51a04a8a6bdcb23deccc82b0b...	1.0
7	5aaa890629f83706d8d9bfecd...	1.0
8	4e2627090e6e5b9fabba883a3...	1.0
9	6e85dc5ecd97a61094b89b046...	1.0
10	40536e7ca18e1bce252828e58...	1.0
11	4003520d80d0bad1d5623f7aa...	1.0
12	15aec03fe4cf30dfa574cf550f5...	1.0

Suggestions

- E commerce is booming in Brazil from 2017 , so more products need to be introduced to the business.
- More sellers need to introduce in farthest places or nearby places to reduce the freight cost.
- Need to revisit the list of sellers with lower rating and check on the reason , then need to work on improving it.
- Need to improve the delivery of the products with in the estimated time , for which the current percentage is only 93%.
- Customers are mainly using credit card as payment option and also using the EMI option. So need to associate with all major banks to give credit card offers to customers so that more customers will go for sale.
- Need to plan well before the peak months like November by storing more products.