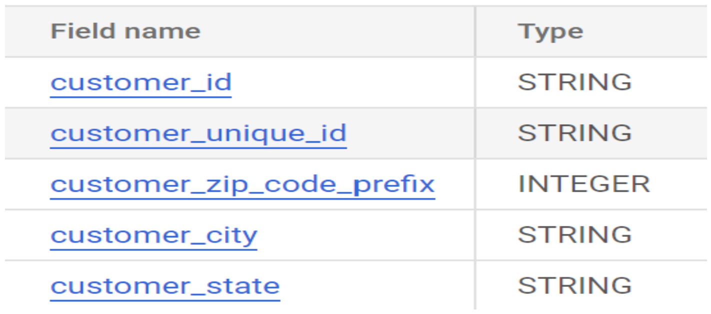
❖ Initial Exploration -

1. Customers table dimensions



Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

Tota distinct customers -

```
select count(distinct customer_id) as total_customers from
`Target.customers`;
```

=> 99441

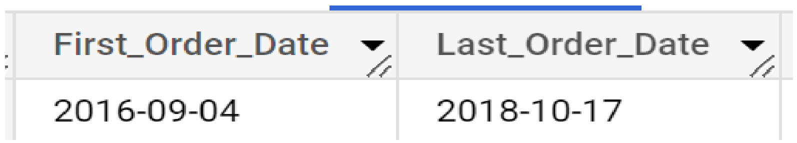
Total distinct account owners -

```
select count(distinct customer_unique_id) as primary_customers from
`Target.customers`;
```

=> 96096

2. Time range between which the orders were placed

```
select min(DATE(order_purchase_timestamp)) as
First_Order_Date,max(DATE(order_purchase_timestamp)) as Last_Order_Date
from `Target.orders`;
```

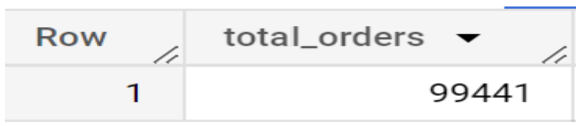


First_Order_Date	Last_Order_Date
2016-09-04	2018-10-17

First Order placed on 2016-09-04 and Last order on 2018-10-17

Total Orders in orders table-

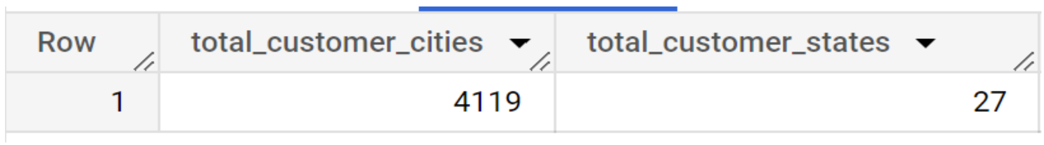
```
select count( distinct order_id) as total_orders from `Target.orders`;
```



Row	total_orders
1	99441

3. Number of Cities and States in our dataset.

```
select count(distinct customer_city) as
total_customer_cities,count(distinct customer_state) as
total_customer_states from `Target.customers` c join `Target.orders` o on
c.customer_id=o.customer_id;
```

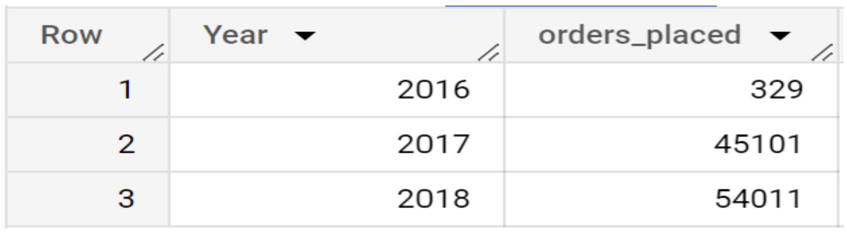


Row	total_customer_cities	total_customer_states
1	4119	27

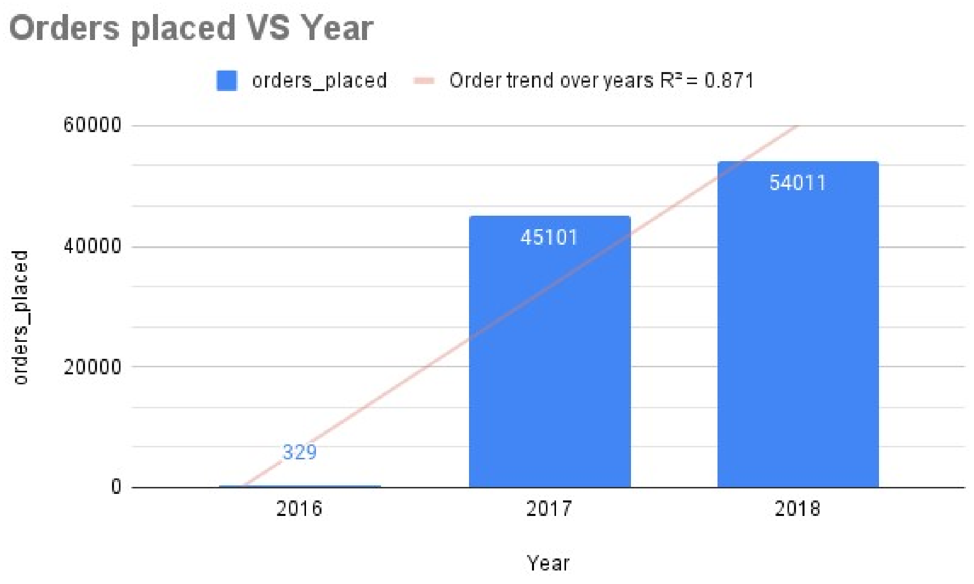
❖ **In-depth Exploration -**

1. Trend in the no. of orders placed over the past years

```
select  Year, count(distinct order_id) as orders_placed from
(select EXTRACT(YEAR from order_purchase_timestamp) as Year, *
from `Target.orders`
) t group by t.Year order by Year;
```



Row	Year	orders_placed
1	2016	329
2	2017	45101
3	2018	54011

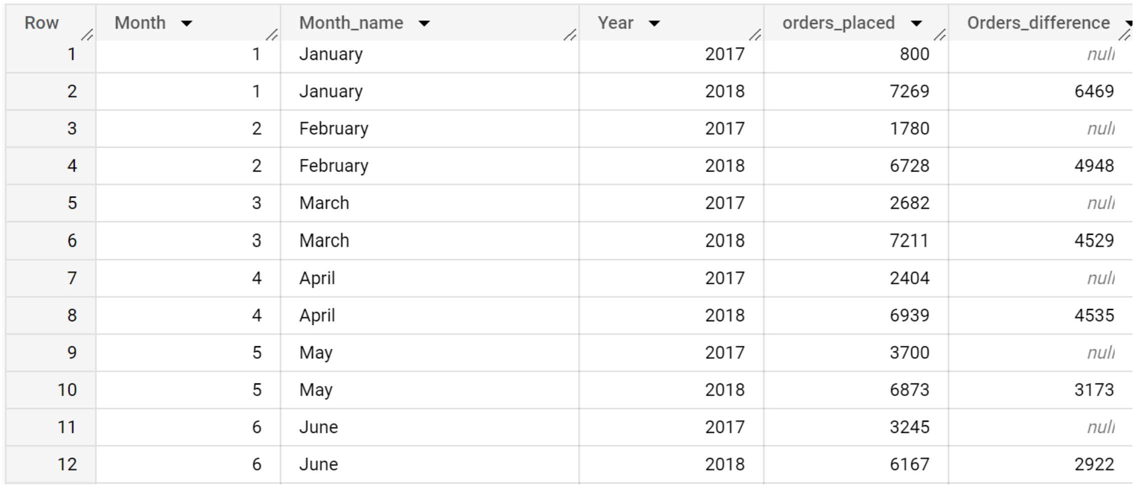


Insights -

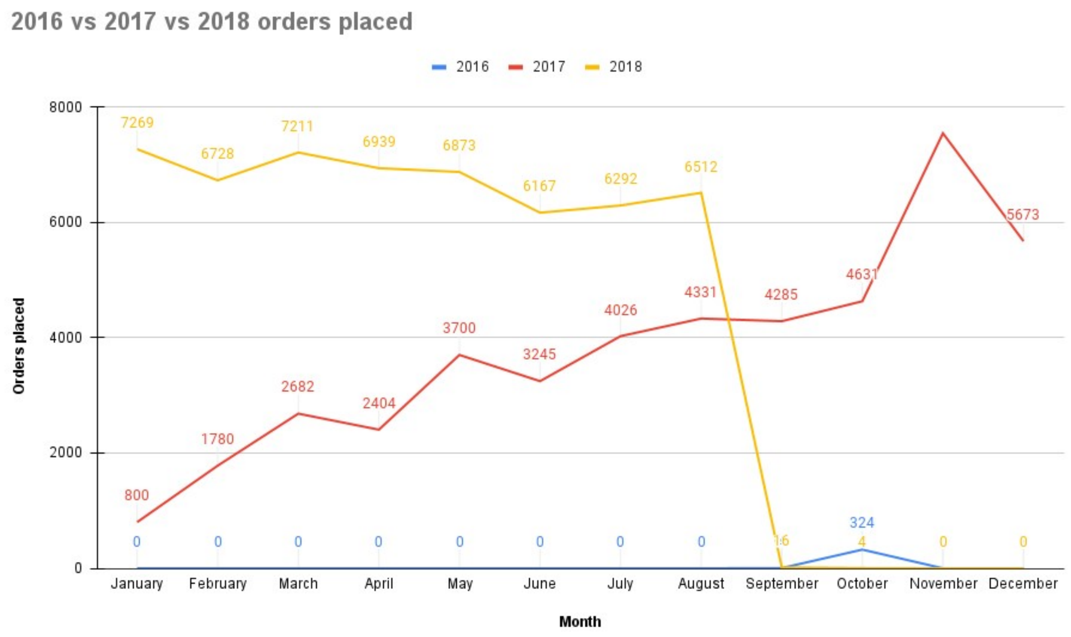
Year-2016 has only 2 months of data, but between Year 2017 and Year 2018(until October), there is an increasing trend of orders.

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

```
select  Month,Month_name,Year, count(distinct order_id) as orders_placed,
(count(distinct order_id)-lag(count(distinct order_id),1) over (partition
by Month order by Year)) as Orders_difference
from
(select EXTRACT(YEAR from order_purchase_timestamp) as Year,EXTRACT(MONTH
from order_purchase_timestamp) as Month,
FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) as Month_name,
*
from Target.orders
) t group by t.Month,t.Month_name,t.Year order by Month,Year;
```



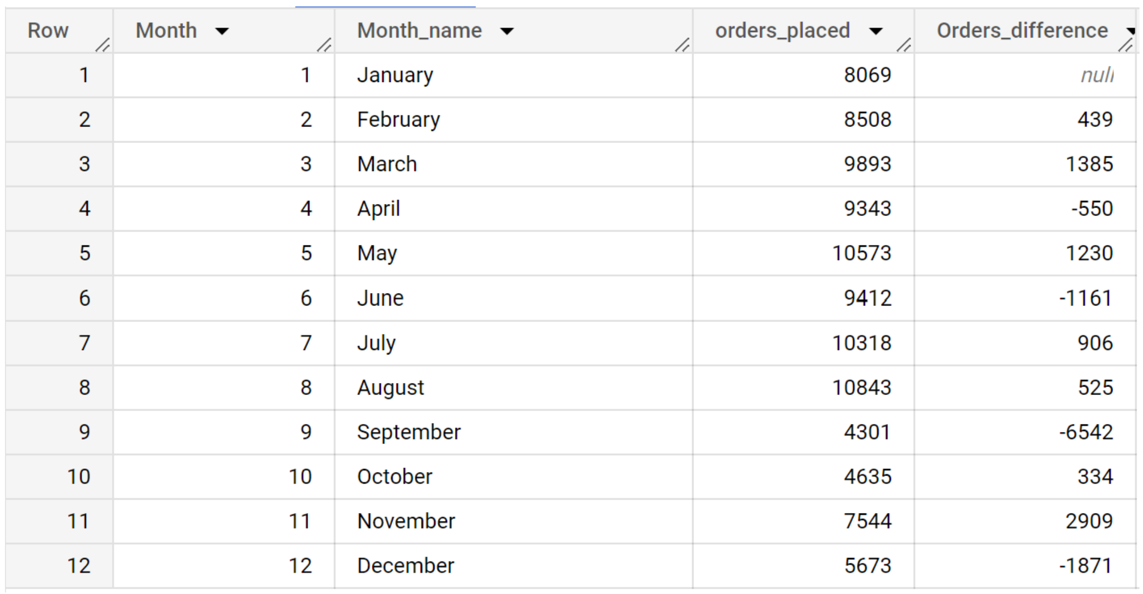
Row	Month	Month_name	Year	orders_placed	Orders_difference
1	1	January	2017	800	*null*
2	1	January	2018	7269	6469
3	2	February	2017	1780	*null*
4	2	February	2018	6728	4948
5	3	March	2017	2682	*null*
6	3	March	2018	7211	4529
7	4	April	2017	2404	*null*
8	4	April	2018	6939	4535
9	5	May	2017	3700	*null*
10	5	May	2018	6873	3173
11	6	June	2017	3245	*null*
12	6	June	2018	6167	2922



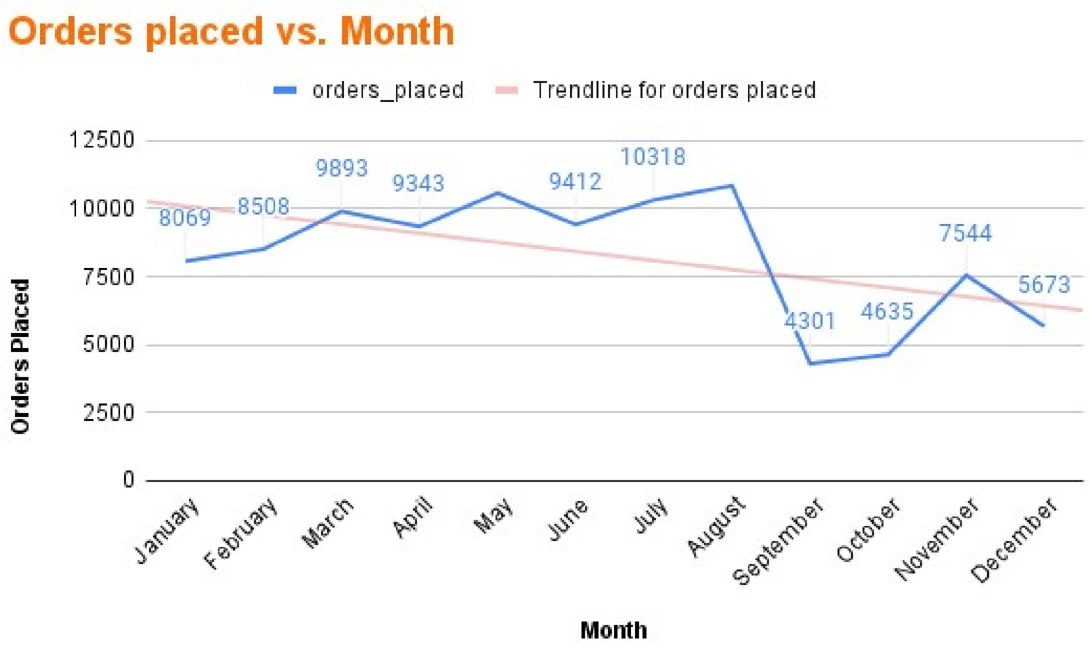
Insights -
Only seasonality observed that order count decreased from May to June Month and again started increasing from July to August. No other common data available between the years 2016,2017 and 2018.
Nov 2017 has a high number of orders(Thanksgiving, Black friday, Cyber Monday etc.) but we don't have other year data to confirm the seasonality.

Since 2016 only has a few months of data(which looks skewed or missing), hence excluding 2016 data in the following query to populate Month vs Number of Orders.

```
select  Month,Month_name, count(distinct order_id) as orders_placed,
(count(distinct order_id)-lag(count(distinct order_id),1) over (order by
Month)) as Orders_difference
from
(select  EXTRACT(MONTH from order_purchase_timestamp) as Month,
FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) as Month_name,
*
from `Target.orders`
where EXTRACT(YEAR from order_purchase_timestamp) != 2016
) t group by t.Month,t.Month_name order by Month;
```



Row	Month	Month_name	orders_placed	Orders_difference
1	1	January	8069	null
2	2	February	8508	439
3	3	March	9893	1385
4	4	April	9343	-550
5	5	May	10573	1230
6	6	June	9412	-1161
7	7	July	10318	906
8	8	August	10843	525
9	9	September	4301	-6542
10	10	October	4635	334
11	11	November	7544	2909
12	12	December	5673	-1871



Insights -

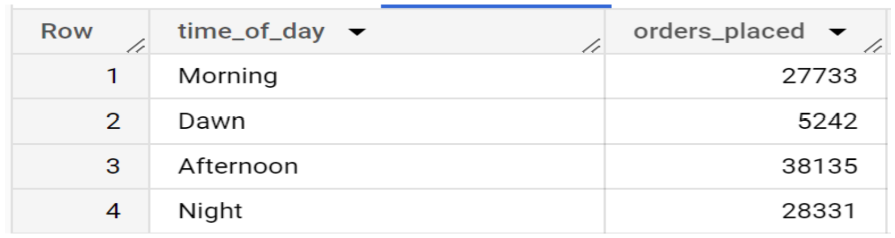
Jan,Mar, May, Nov months have noticed a huge influx of orders compared to the prior months, and Jun, Sep, Dec have a sudden decline in number of orders compared to prior month.

Actionable Input -

Inventory forecasting can be optimized to serve increasing demands in May,Aug,Nov months , while relaxing in Sep, Dec months. Shipping partners can be used efficiently considering demand and drop in orders count.

3. During what time of the day, do the Brazilian customers mostly place their orders?

```
select time_of_day, count(distinct order_id) as orders_placed from
(select case when EXTRACT(HOUR from order_purchase_timestamp) between 0
and 6 then 'Dawn'
             when EXTRACT(HOUR from order_purchase_timestamp) between 7
and 12 then 'Morning'
             when EXTRACT(HOUR from order_purchase_timestamp) between 13
and 18 then 'Afternoon'
             when EXTRACT(HOUR from order_purchase_timestamp) between 19
and 23 then 'Night'
        end as time_of_day,*
from Target.orders) t group by t.time_of_day;
```



Row	time_of_day	orders_placed
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

Insights -

Majority of Brazilians placed orders in Afternoon followed by Night and Morning. Very less number of orders placed during dawn.

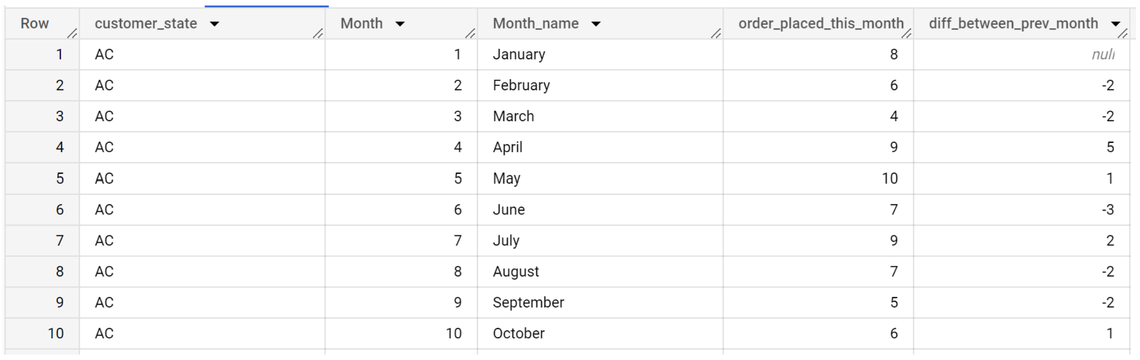
Actionable input -

Make sure Ecommerce portal has zero down time especially in Afternoon and Night period when most of the orders are placed, All site maintenance activities to be planned during Dawn period.

❖ Ecommerce Order journey in Brazil

1. Month on month no. of orders placed in each state

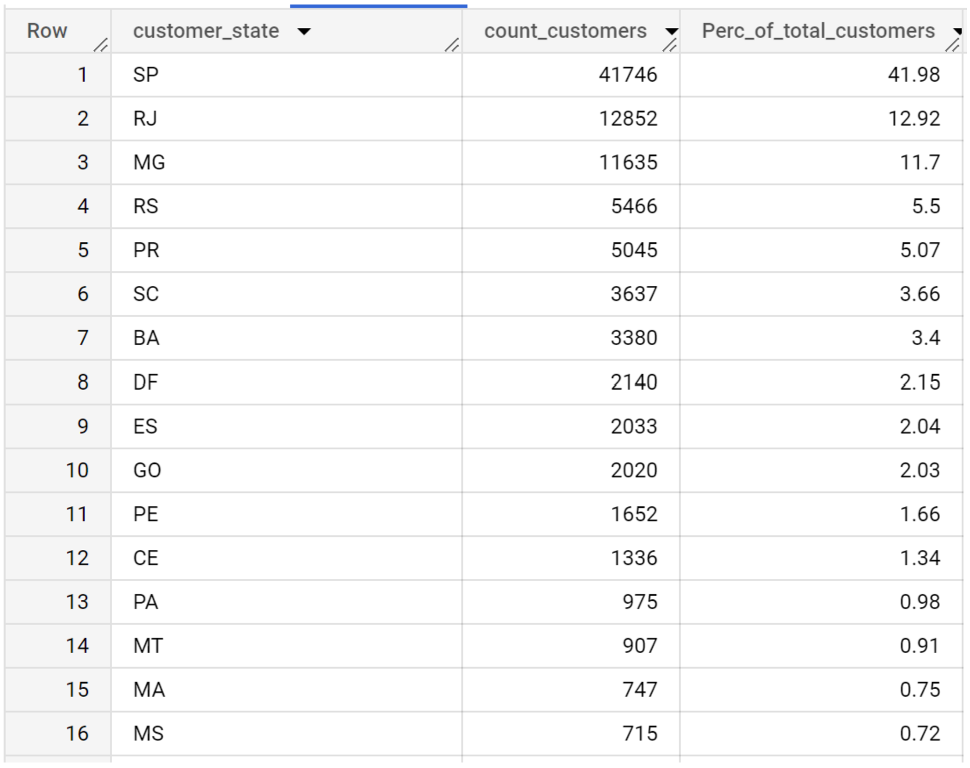
```
select customer_state, Month,Month_name, count(distinct order_id) as
order_placed_this_month,
(count(distinct order_id)-lag(count(distinct order_id),1) over (partition
by customer_state order by Month)) as diff_between_prev_month
from
(select EXTRACT(MONTH from o.order_purchase_timestamp) as Month,
FORMAT_DATETIME("%B", DATETIME(o.order_purchase_timestamp)) as
Month_name,
c.*,o.*
from `Target.orders` o join `Target.customers` c on
o.customer_id=c.customer_id) t
group by customer_state, Month,Month_name
order by 1,2,3;
```



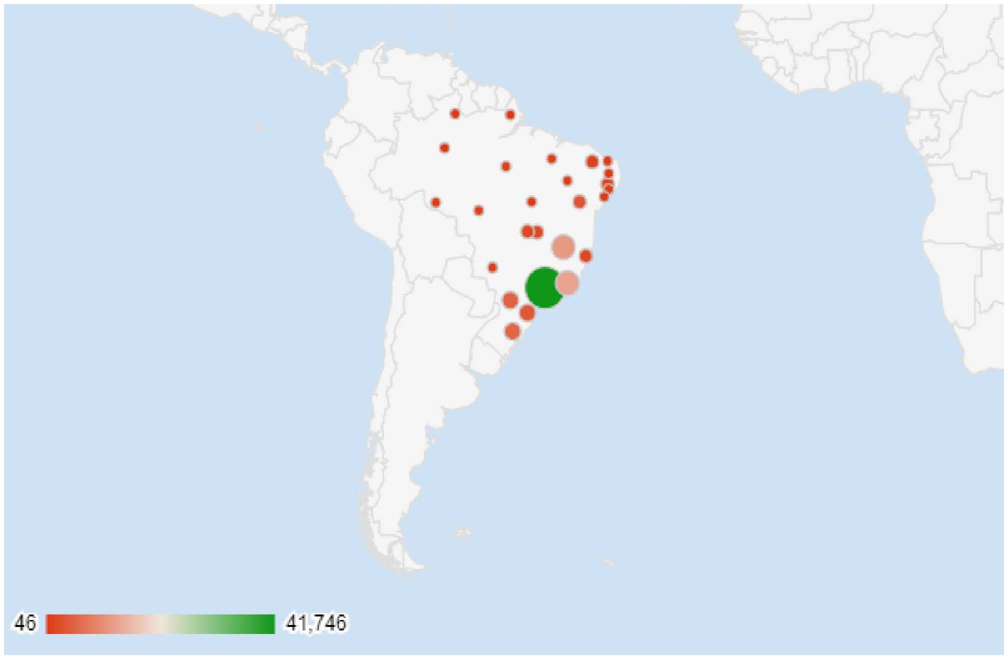
Row	customer_state	Month	Month_name	order_placed_this_month	diff_between_prev_month
1	AC	1	January	8	null
2	AC	2	February	6	-2
3	AC	3	March	4	-2
4	AC	4	April	9	5
5	AC	5	May	10	1
6	AC	6	June	7	-3
7	AC	7	July	9	2
8	AC	8	August	7	-2
9	AC	9	September	5	-2
10	AC	10	October	6	1

2. How are the customers distributed across the states?

```
select customer_state, count_customers, round(count_customers/(select
count(distinct customer_id) from `Target.customers`)*100,2) as
Perc_of_total_customers
from
(select customer_state, count(*) as count_customers
from `Target.customers` group by customer_state order by count_customers
desc)t
order by count_customers desc;
```



Row	customer_state	count_customers	Perc_of_total_customers
1	SP	41746	41.98
2	RJ	12852	12.92
3	MG	11635	11.7
4	RS	5466	5.5
5	PR	5045	5.07
6	SC	3637	3.66
7	BA	3380	3.4
8	DF	2140	2.15
9	ES	2033	2.04
10	GO	2020	2.03
11	PE	1652	1.66
12	CE	1336	1.34
13	PA	975	0.98
14	MT	907	0.91
15	MA	747	0.75
16	MS	715	0.72



Insights -
São Paulo(SP) state has the most customers, more than 3 times of 2nd highest customers of Rio De Janeiro(RJ) and around 42% of total customers in brazil. Top 3 states - SP,RJ,MG contribute around 67% of the customer base.
Actionable Input -
Target should focus on states having low customer count with different marketing strategies to increase site traffic(like promotions, delivery speed) and focus on

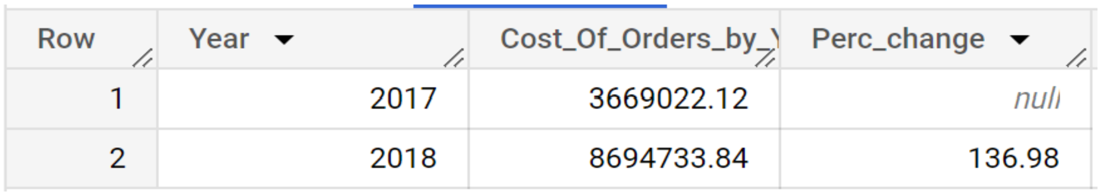
states with high customer count(SP,RJ,MG) to increase inventory, setup of new DC to reduce delivery time and increase customer satisfaction to retain traffic.

❖ Impact on Economy

1. % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only)

```
with Order_value_cte as (
select o.order_id, sum(p.payment_value) as cost_of_orders
from `Target.orders` o join `Target.payments` p on o.order_id=p.order_id
group by o.order_id )

select Year,Cost_Of_Orders_by_Year,
round(((Cost_Of_Orders_by_Year-lag(Cost_Of_Orders_by_Year,1) over(order
by Year))/lag(Cost_Of_Orders_by_Year,1) over(order by Year))*100,2) as
Perc_change
from (
select EXTRACT(YEAR from o1.order_purchase_timestamp) as Year,
round(sum(cost_of_orders),2) as Cost_Of_Orders_by_Year
from `Target.orders` o1 join Order_value_cte o2 on
o1.order_id=o2.order_id
where EXTRACT(MONTH from o1.order_purchase_timestamp) not in (9,10,11,12)
group by EXTRACT(YEAR from o1.order_purchase_timestamp)
) t
order by Year;
```



Row	Year	Cost_Of_Orders_by_Y	Perc_change
1	2017	3669022.12	null
2	2018	8694733.84	136.98

Insights -

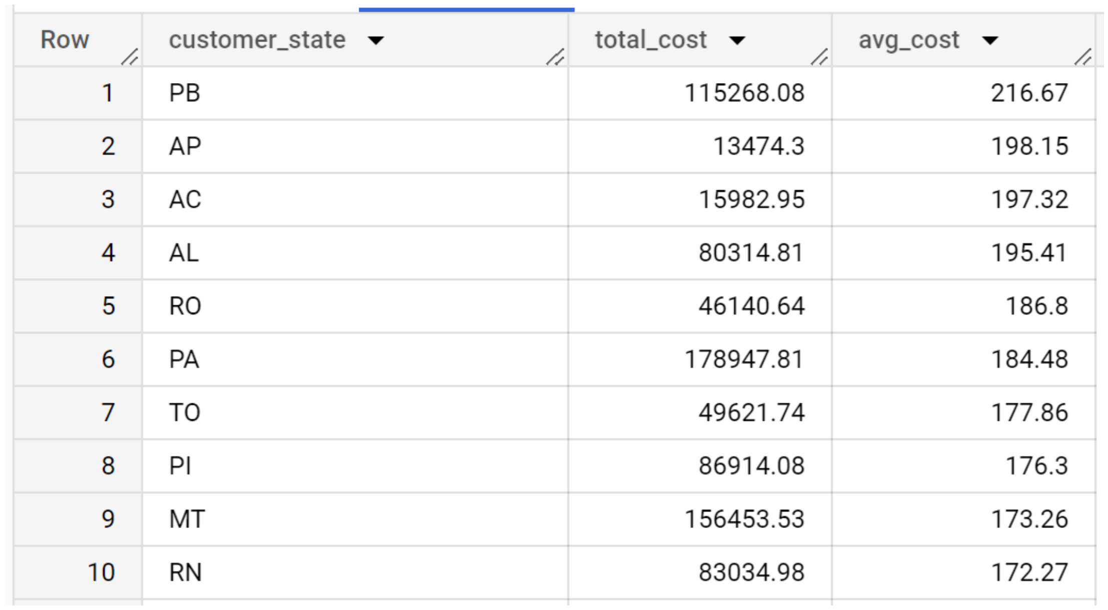
There is 137% increase in cost of orders between 2017 and 2018, which means order count is increasing year over year, customers are loving available products and satisfied with product cost.

Actionable Input -

Target should plan to roll out more products, promotions, stores/distribution centers, and enhance logistics infrastructure to cater doubling growth of orders in coming years and maximize profit by reducing freight cost.

2. Total & Average value of order price for each state

```
select c.customer_state, round(sum(oi.order_value),2) as total_cost,
round(avg(oi.order_value),2) as avg_cost
from `Target.orders` o join `Target.customers` c on
o.customer_id=c.customer_id
join (select order_id, sum(price) as order_value from
`Target.order_items` group by order_id) oi on o.order_id = oi.order_id
group by c.customer_state
order by avg_cost desc
```



Row	customer_state	total_cost	avg_cost
1	PB	115268.08	216.67
2	AP	13474.3	198.15
3	AC	15982.95	197.32
4	AL	80314.81	195.41
5	RO	46140.64	186.8
6	PA	178947.81	184.48
7	TO	49621.74	177.86
8	PI	86914.08	176.3
9	MT	156453.53	173.26
10	RN	83034.98	172.27

Consider item price which is excluding freight cost.

Insights -

Top10 states having the highest average cost shows customer purchase power in these states.

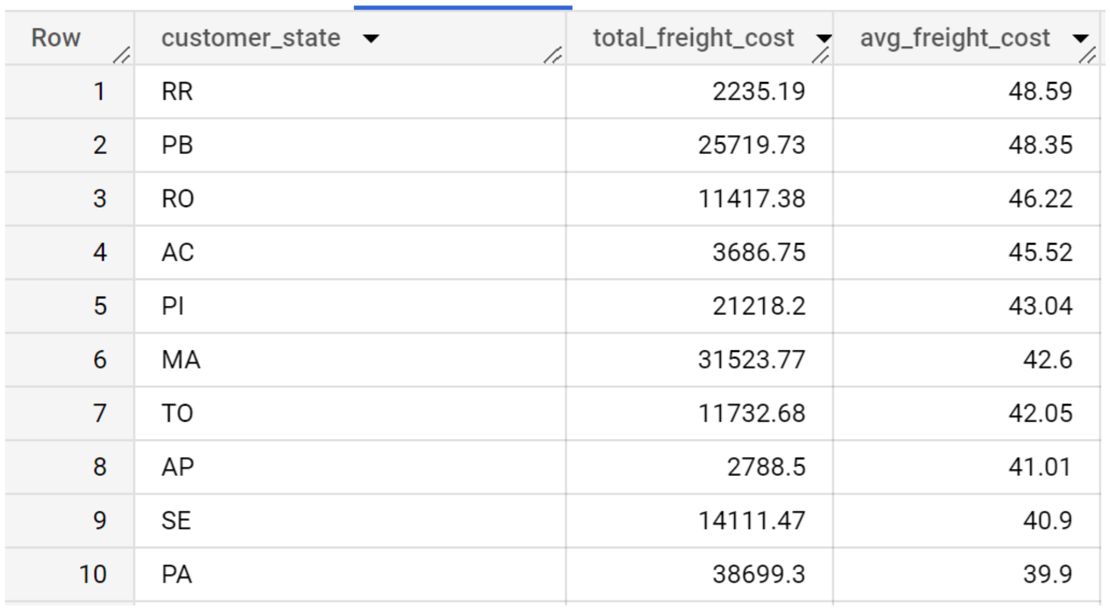
Actionable Input -

Target needs to focus on states with lowest avg cost to offer new product recommendations, Combo product offers, promotions on extra items to increase average spending of customers.

3. Total & Average value of order freight for each state

```
select c.customer_state, round(sum(i.freight_value),2) as
total_freight_cost, round(avg(i.freight_value),2) as avg_freight_cost
from `Target.orders` o join `Target.customers` c on
o.customer_id=c.customer_id
```

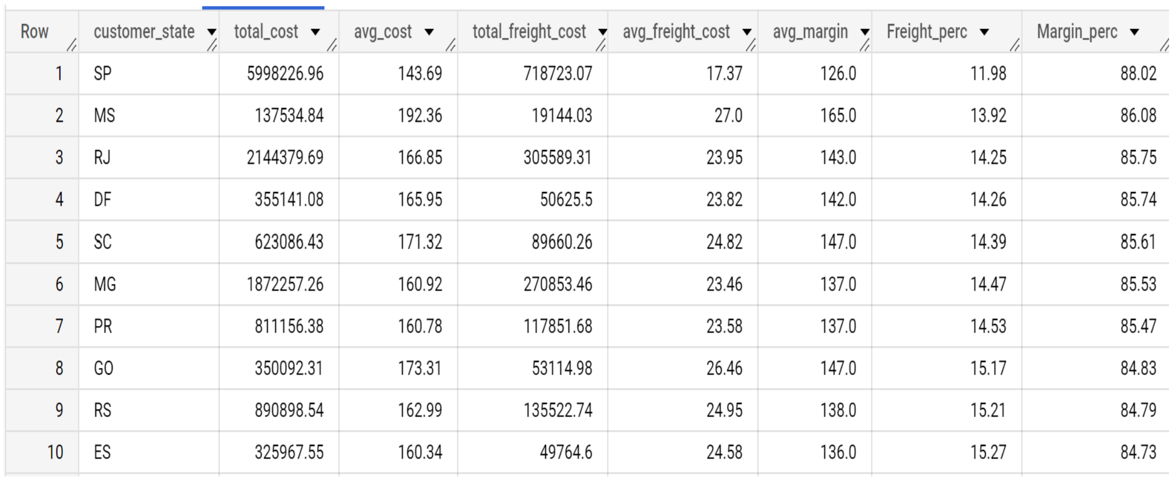
```
join (select order_id, sum(freight_value) as freight_value
from`Target.order_items` group by order_id) i
on o.order_id = i.order_id
group by c.customer_state
order by avg_freight_cost desc
```



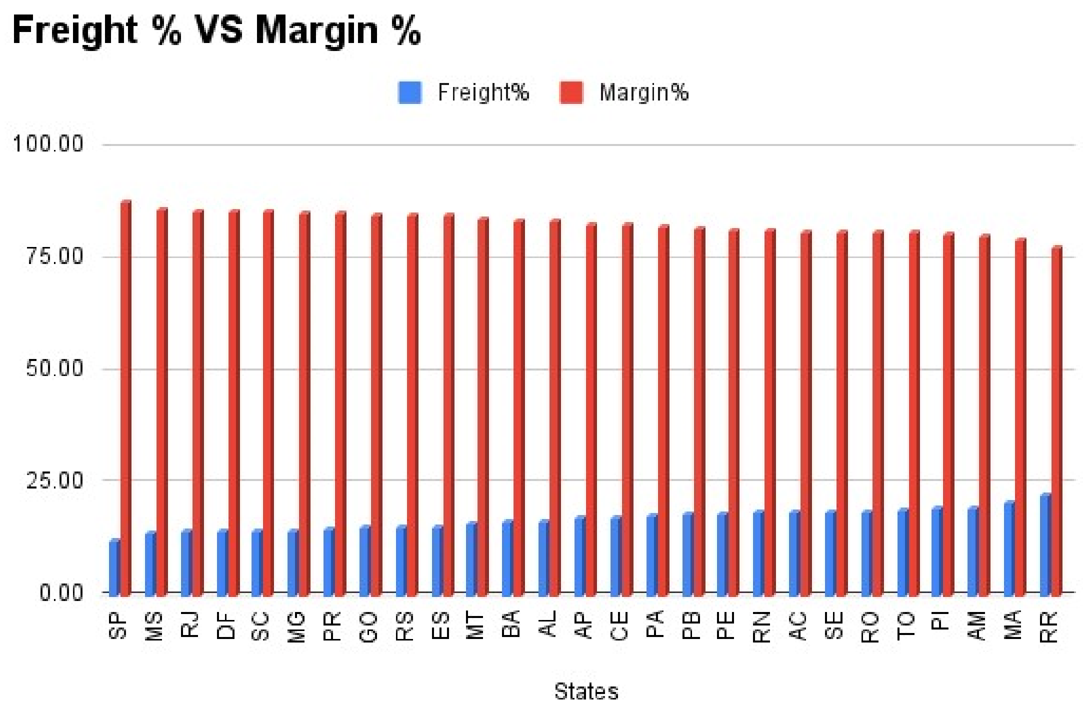
Row	customer_state	total_freight_cost	avg_freight_cost
1	RR	2235.19	48.59
2	PB	25719.73	48.35
3	RO	11417.38	46.22
4	AC	3686.75	45.52
5	PI	21218.2	43.04
6	MA	31523.77	42.6
7	TO	11732.68	42.05
8	AP	2788.5	41.01
9	SE	14111.47	40.9
10	PA	38699.3	39.9

Insights -
By combining results from #2 and #3, it is evident that below states perform very well to keep freight cost low thereby increasing profit margin. States - SP, MS, RJ, DF, SC tops the list in terms of expanding the Margin % per order by reducing Freight cost per order.

Actionable input -
Target needs to revise shipping services, increase the efficiency of delivery and reduce freight cost in states with high freight value.



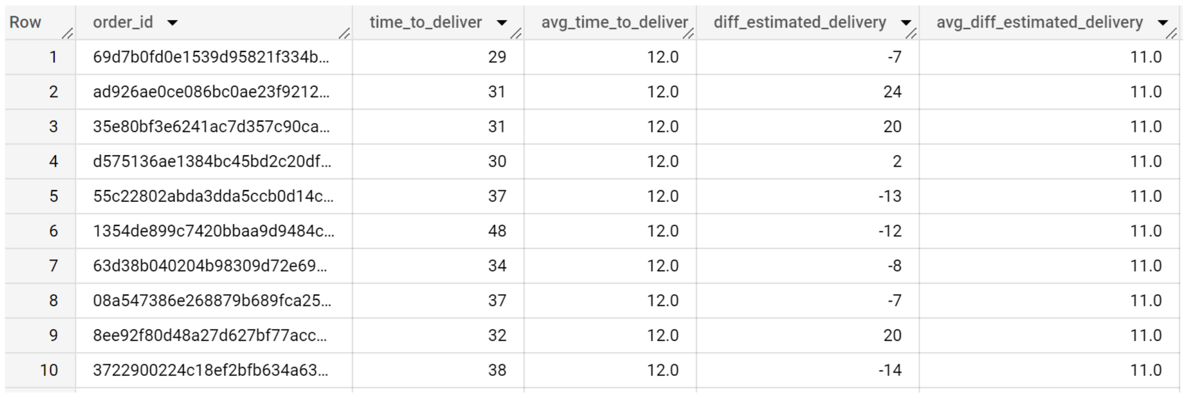
Row	customer_state	total_cost	avg_cost	total_freight_cost	avg_freight_cost	avg_margin	Freight_perc	Margin_perc
1	SP	5998226.96	143.69	718723.07	17.37	126.0	11.98	88.02
2	MS	137534.84	192.36	19144.03	27.0	165.0	13.92	86.08
3	RJ	2144379.69	166.85	305589.31	23.95	143.0	14.25	85.75
4	DF	355141.08	165.95	50625.5	23.82	142.0	14.26	85.74
5	SC	623086.43	171.32	89660.26	24.82	147.0	14.39	85.61
6	MG	1872257.26	160.92	270853.46	23.46	137.0	14.47	85.53
7	PR	811156.38	160.78	117851.68	23.58	137.0	14.53	85.47
8	GO	350092.31	173.31	53114.98	26.46	147.0	15.17	84.83
9	RS	890898.54	162.99	135522.74	24.95	138.0	15.21	84.79
10	ES	325967.55	160.34	49764.6	24.58	136.0	15.27	84.73



❖ **Analysis based on sales, freight and delivery time**

1. Number of days taken to deliver each order from the order's purchase date as delivery time

```
select order_id,time_to_deliver,round(AVG(time_to_deliver) over(),0) as
avg_time_to_deliver,diff_estimated_delivery,
round(AVG(diff_estimated_delivery) over(),0) as
avg_diff_estimated_delivery
from (
select order_id,
TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY
) as time_to_deliver,
TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_dat
e,DAY) as diff_estimated_delivery
from `Target.orders`) t;
```



Row	order_id	time_to_deliver	avg_time_to_deliver	diff_estimated_delivery	avg_diff_estimated_delivery
1	69d7b0fd0e1539d95821f334b...	29	12.0	-7	11.0
2	ad926ae0ce086bc0ae23f9212...	31	12.0	24	11.0
3	35e80bf3e6241ac7d357c90ca...	31	12.0	20	11.0
4	d575136ae1384bc45bd2c20df...	30	12.0	2	11.0
5	55c22802abda3dda5ccb0d14c...	37	12.0	-13	11.0
6	1354de899c7420bbaa9d9484c...	48	12.0	-12	11.0
7	63d38b040204b98309d72e69...	34	12.0	-8	11.0
8	08a547386e268879b689fca25...	37	12.0	-7	11.0
9	8ee92f80d48a27d627bf77acc...	32	12.0	20	11.0
10	3722900224c18ef2bfb634a63...	38	12.0	-14	11.0

Insights -

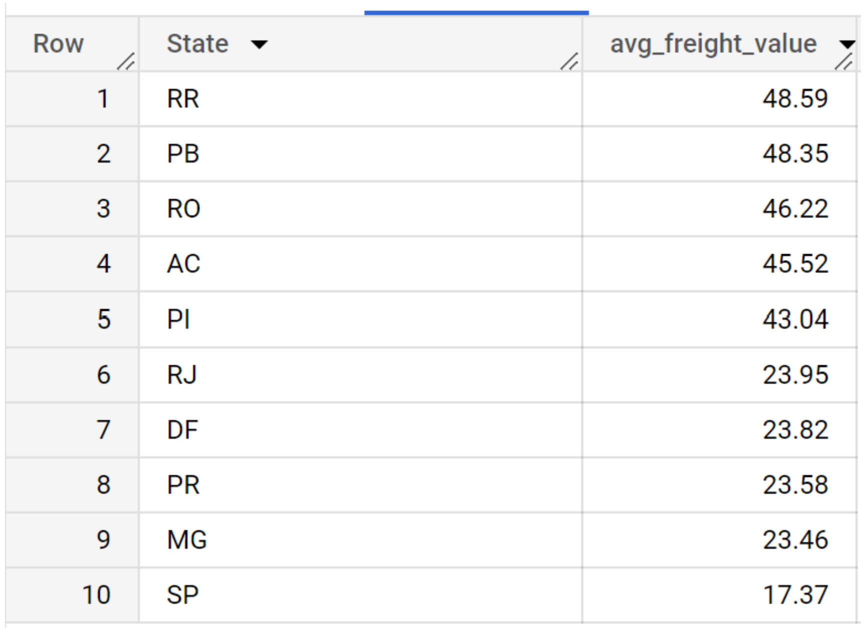
a. Avg delivery time is 12 days and Average difference in estimated delivery is 11 days.
b. 42% of orders took less than average time to deliver.
c. 44% of orders were delivered in less than average difference in estimated delivery and actual delivery time.

Actionable Input -

a. 89941 orders out of 99441 i.e. 90% were delivered on or before estimated delivery date and only 10% order missed estimated delivery time.
   Identify the reason for these 10% orders like distance of customer address location from shipping center, product in-transit days, shipping services in those locations.
b. Add new distribution centers or shipping centers where orders took more than estimated delivery time.
c. Since 90% orders are on or before est. time, re-estimate the delivery time, and find ways to decrease estimated delivery time further.

2. Top 5 states with the highest & lowest average freight value

```
select customer_state as State, avg_freight_value from (
select c.customer_state, round(AVG(freight_value),2) as
avg_freight_value,
dense_rank() over(order by AVG(freight_value) desc) as top_rank,
dense_rank() over(order by AVG(freight_value) asc) as bottom_rank
from
(select order_id, sum(freight_value) as freight_value
from`Target.order_items` group by order_id) oi
join `Target.orders` o on oi.order_id =o.order_id
join `Target.customers` c on o.customer_id = c.customer_id
group by c.customer_state) t
where t.top_rank <=5 or t.bottom_rank <=5
order by t.top_rank;
```



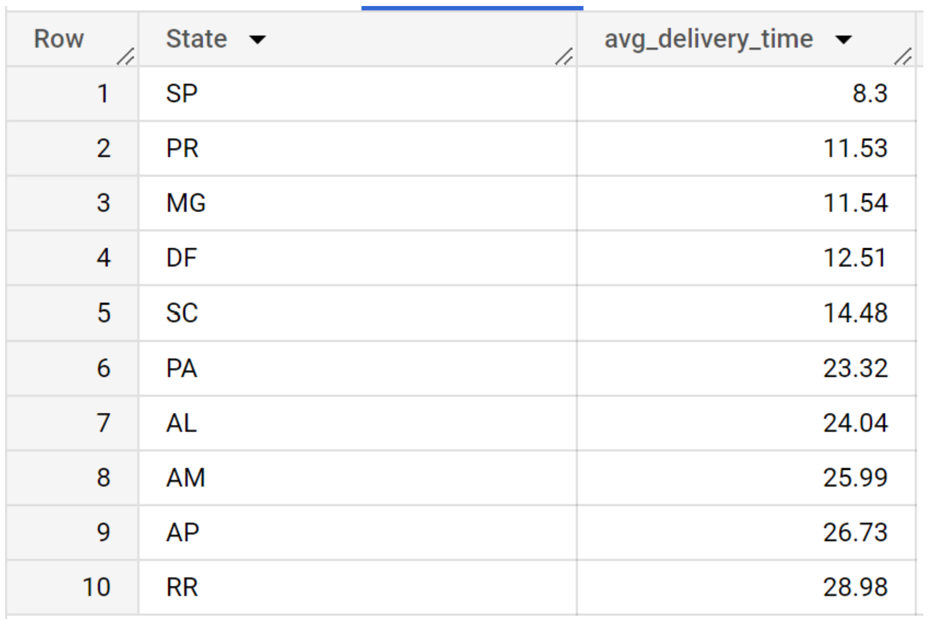
Row	State	avg_freight_value
1	RR	48.59
2	PB	48.35
3	RO	46.22
4	AC	45.52
5	PI	43.04
6	RJ	23.95
7	DF	23.82
8	PR	23.58
9	MG	23.46
10	SP	17.37

Insights -
First 5 rows are top 5 states with highest average freight value
Last 5 rows are bottom 5 states with lowest average freight value
States - SP, PR, MG, RJ, DF are having least freight cost means they have well structured delivery channels, more shipping points.
Actionable input -
Top 5 states in the above result are spending more on freight. Target needs to focus on these states to reduce freight cost by improving the supply chain system.

Target needs to revise shipping services, increase the efficiency of delivery and reduce freight cost in states with high freight value.

3. Top 5 states with the highest & lowest average delivery time

```
select customer_state as State, avg_delivery_time from (
select c.customer_state,
round(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_tim
estamp,DAY)),2) as avg_delivery_time, rank() over(order by
AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp
,DAY)) desc) as high_rank, rank() over(order by
AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp
,DAY)) asc) as low_rank
from `Target.orders` o join `Target.customers` c on
o.customer_id=c.customer_id
group by c.customer_state ) t
where t.high_rank<=5 or t.low_rank<=5
order by t.low_rank;
```



Row	State	avg_delivery_time
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48
6	PA	23.32
7	AL	24.04
8	AM	25.99
9	AP	26.73
10	RR	28.98

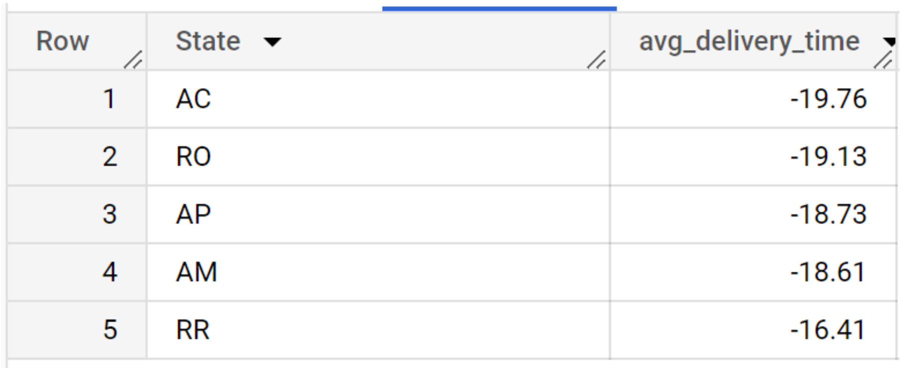
First 5 rows shows lowest average delivery time
Last 5 rows shows highest average delivery time

<u>Insights -</u>
State of SP has an average delivery time less than the national average time of Brazil.

<u>Actionable input</u> -
Target needs to improve delivery infrastructure/services in Top 5 highest avg delivery time states.

4. Top 5 states where the order delivery is really fast as compared to the estimated date of delivery

```
select customer_state as State, avg_delivery_time
from (
select c.customer_state,
round(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_estimated_de
livery_date,DAY)),2) as avg_delivery_time, rank() over(order by
AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_estimated_delivery
_date,DAY))) as rn
from `Target.orders` o join `Target.customers` c on
o.customer_id=c.customer_id
group by c.customer_state) t
where t.rn <=5
order by t.rn;
```



Row	State	avg_delivery_time
1	AC	-19.76
2	RO	-19.13
3	AP	-18.73
4	AM	-18.61
5	RR	-16.41

Insights -

Negative average time shows orders are delivered before estimated delivery time, top 5 States are AC, RO, AP, AM, RR

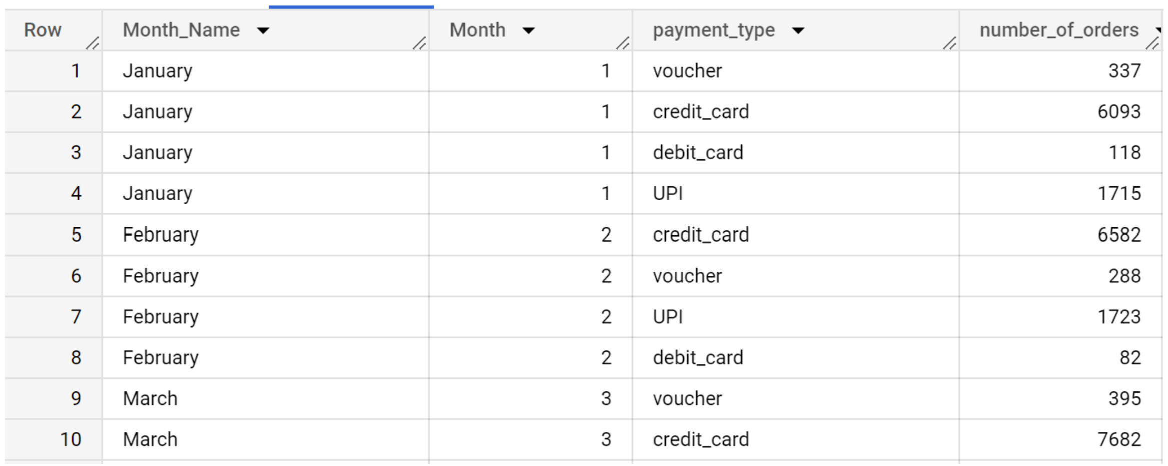
Actionable input -

Estimated delivery dates in these states need to be revised to be more close to actual delivery dates which will result in customers satisfaction with delivery time, more accurate prediction of delivery time and to set new targets to improve delivery services.

❖ Analysis based on the payments

1. Month on month no. of orders placed using different payment types

```
select Month_Name,Month,payment_type,count(distinct order_id) as
number_of_orders
from (
select EXTRACT(MONTH from o.order_purchase_timestamp) as
Month,FORMAT_DATETIME("%B", DATETIME(o.order_purchase_timestamp)) as
Month_name,payment_type,o.order_id
from `Target.payments` p join `Target.orders` o on p.order_id=o.order_id)
t
group by Month_Name,Month,payment_type
order by Month;
```



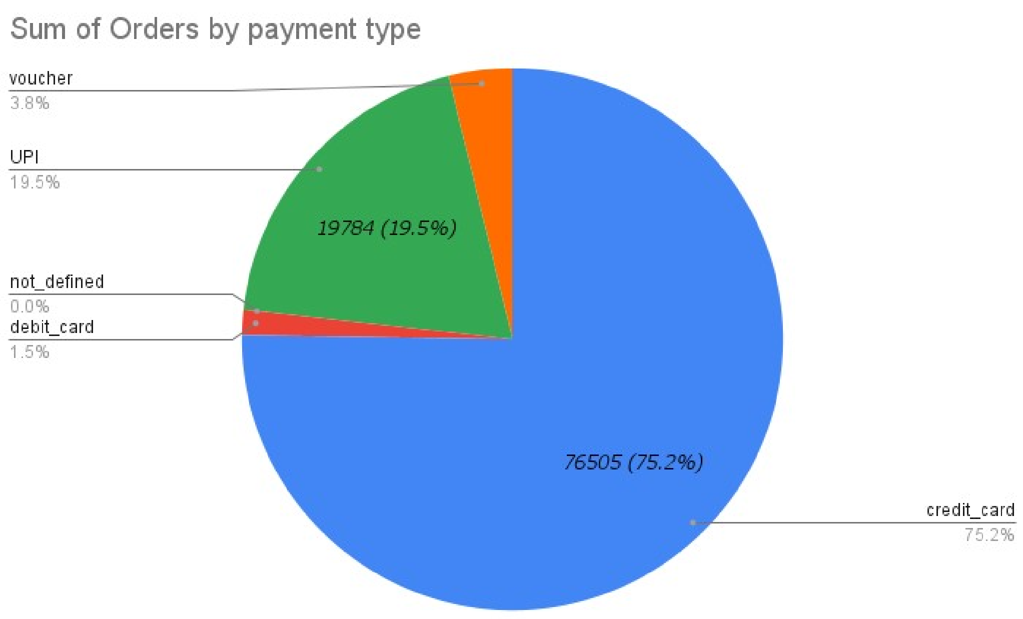
Row	Month_Name	Month	payment_type	number_of_orders
1	January	1	voucher	337
2	January	1	credit_card	6093
3	January	1	debit_card	118
4	January	1	UPI	1715
5	February	2	credit_card	6582
6	February	2	voucher	288
7	February	2	UPI	1723
8	February	2	debit_card	82
9	March	3	voucher	395
10	March	3	credit_card	7682

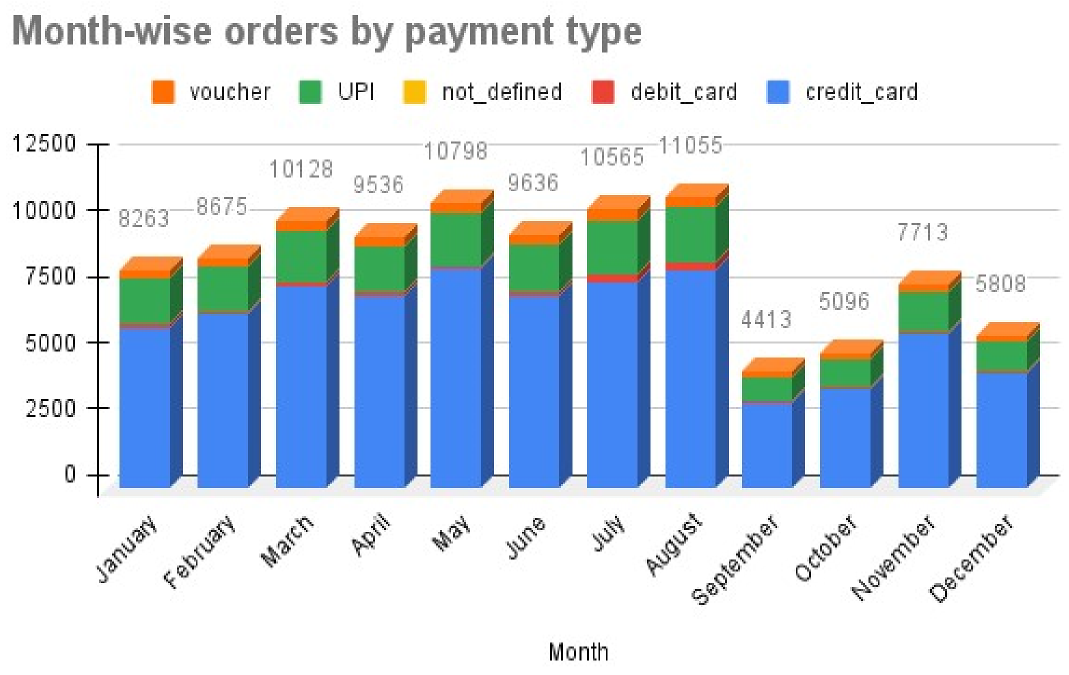
Insights -

Credit cards and UPI are the most popular payment types used by Brazilians during shopping which contribute to 96% of all payments.

Actionable inputs -

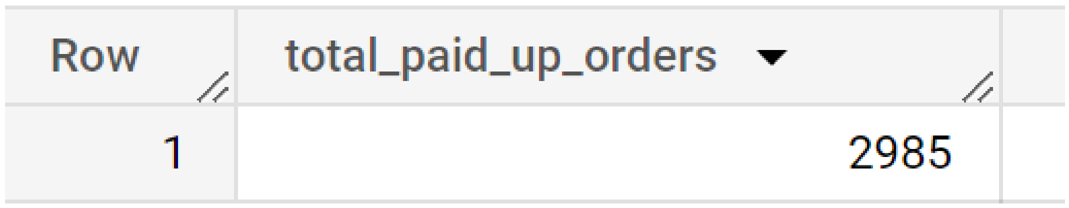
1. Target needs to focus on credit cards and UPI payments issues rigorously and address customer concerns if any in time.
2. Target should check customer concerns over other payment types, and provide more ways for customers to use it.
3. Target should get new deals with Credit cards and UPI service providers to get best value for themselves considering volume of transactions.



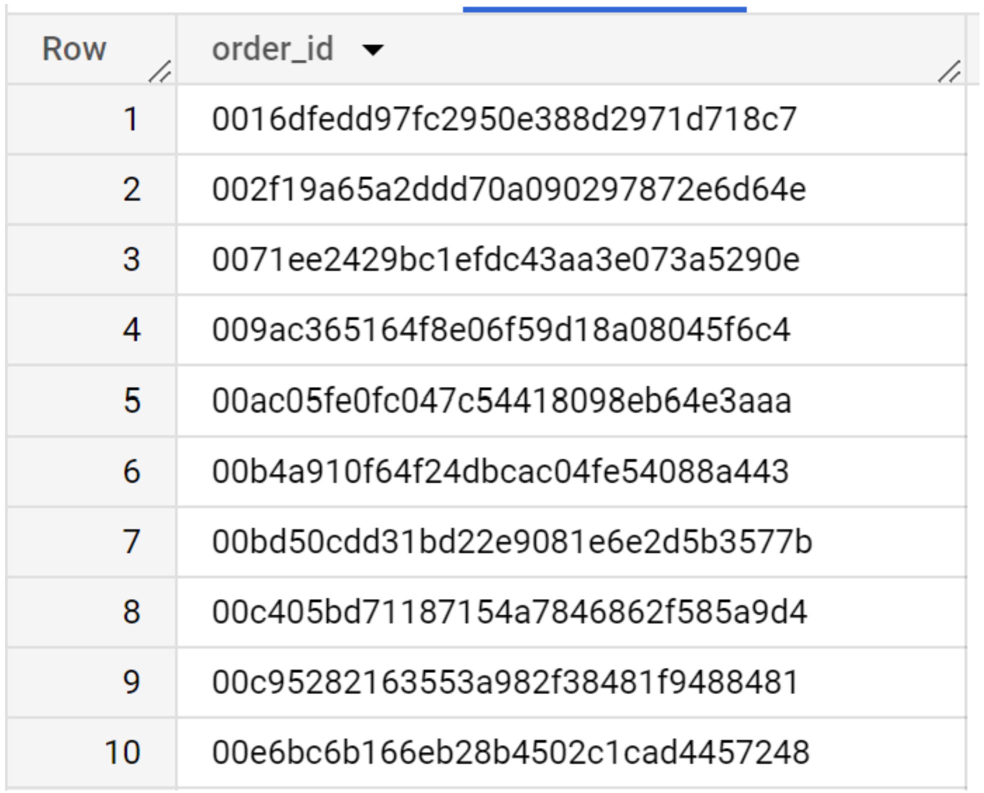


2. Number of orders placed on the basis of the payment installments that have been paid.

```
select t1.order_id from
(select order_id, round(sum(payment_value),2) as total_payments from
`Target.payments` where order_id in (
select distinct order_id from `Target.payments`  where payment_sequential
> 1 )
group by order_id) t1 join
(select order_id, round(sum(price+freight_value),2) as total_order_value
from `Target.order_items` group by order_id) t2
on t1.order_id=t2.order_id
where t1.total_payments = t2.total_order_value
order by t1.order_id
```



Row	total_paid_up_orders
1	2985



Row	order_id
1	0016dfedd97fc2950e388d2971d718c7
2	002f19a65a2ddd70a090297872e6d64e
3	0071ee2429bc1efdc43aa3e073a5290e
4	009ac365164f8e06f59d18a08045f6c4
5	00ac05fe0fc047c54418098eb64e3aaa
6	00b4a910f64f24dbcac04fe54088a443
7	00bd50cdd31bd22e9081e6e2d5b3577b
8	00c405bd71187154a7846862f585a9d4
9	00c95282163553a982f38481f9488481
10	00e6bc6b166eb28b4502c1cad4457248

Insights -

There are a total - 3029 orders placed with payment_sequential > 1 out of which 2985 orders are paid up.