Business Case : Target e-commerce in Brazil

Quick walk through:

- Target is a retailer based out of U.S.
- This data is of its e-commerce platform in Brazil.
- With 100,000 orders placed between 2016 to 2018.
- The data is available in tables:
  - customers
  - sellers
  - order_items
  - geolocation
  - payments
  - reviews
  - orders
  - products

1. Exploratory Analysis

1a. Schema of the customers table:

```
select
      column_name,
      data_type
from `Target.INFORMATION_SCHEMA.COLUMNS`
where TABLE_NAME= "customers"
```

Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Inference:

The customers table has five different columns from customer_id to customer_state belonging to string or integer data types.

Further analysis:

Customers table has 5 columns with 99,441 rows.

C Charandeep Reddy
Github : https://github.com/CharanDRC

The customer_id column has 99,441 unique customer_id's which belong to 99,441 customers. Where as the customer_unique_id column has 96,096 rows indicating duplicates which contradicts the column name and makes this column not fit for being a primary key or candidate key.

```
select count(*) as no_of_rows
from Target.customers inner join Target.orders using(customer_id)
```

Row	no_of_rows
1	99441

When we join the customers table with the orders table we got to know that each customer has made only one order.

1b. Time range of orders:

```
select
      min(date(order_purchase_timestamp))as `From`,
      max(date(order_purchase_timestamp))as `To`
from Target.orders
```

Row	From	To
1	2016-09-04	2018-10-17

Inference:

The orders are placed between 2016-09-04 to 2018-10-17 time range.

1c. Count of cities and states of customers who ordered:

```
select
      count(distinct customer_city)as No_of_Cities,
      count(distinct customer_state)as No_of_States
from Target.customers as c inner join Target.orders as o
  on c.customer_id=o.customer_id
```

Row	No_of_Cities	No_of_States
1	4119	27

There are 4,119 cities of 29 states.

Insights:

1. **The customers table has five different columns from customer_id to customer_state belonging to string or integer data types. Where the customer_id column has unique id for each record and each customer has placed an order and only once.**

2. **The orders are placed between 2016-09-04 to 2018-10-17 time range.**

3. **There are 4,119 cities of 29 states to which the customers belong to(who ordered during this time period).**

---

2. In-depth Exploration:

2a. Year on Year analysis on no. of orders placed :

```
select Year,count(order_id)as No_of_Orders
from  (select order_id,extract(year from order_purchase_timestamp) as Year
      from Target.orders) as t
group by Year
order by Year
```

Row	Year	No_of_Orders	Prev_Year_No_of_Orders
1	2016	329	*null*
2	2017	45101	329
3	2018	54011	45101

Inference:

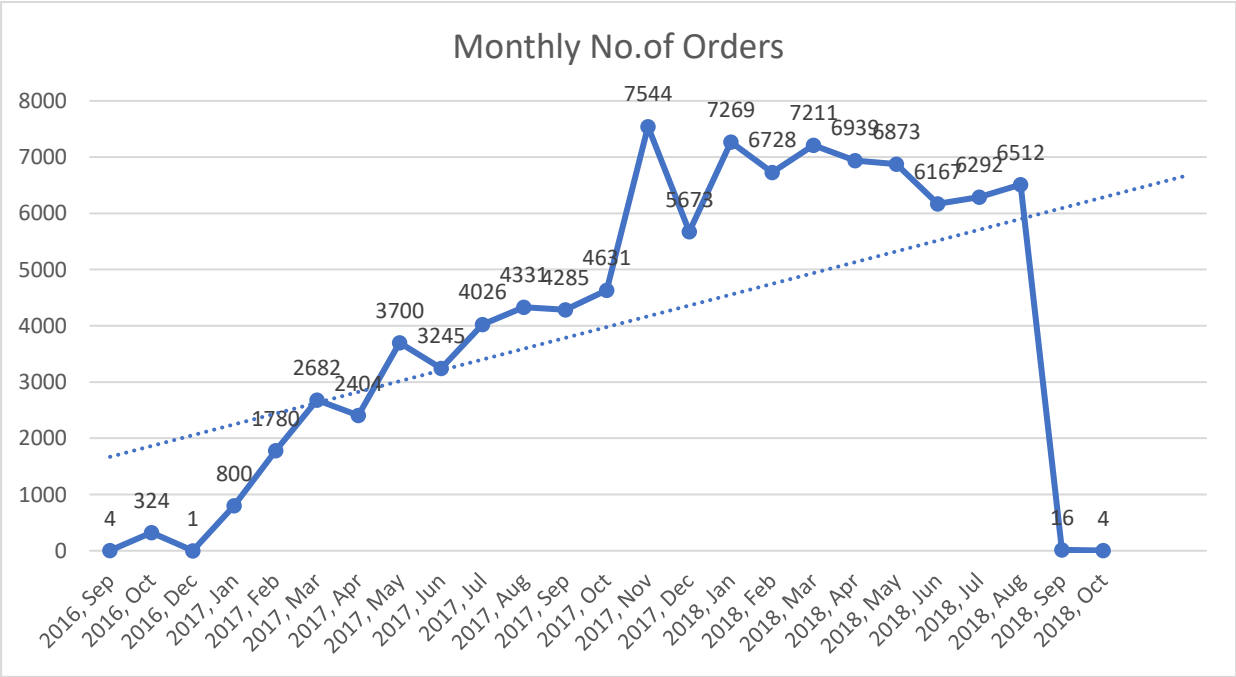
Yes, the no.of orders placed are increasing year on year, which shows an upward trend in the no.of orders places w.r.t years.

Further analysis:

Let's check either this upward trend in no.of orders continues, monthly in a year.

```
select Year,Month,count(order_id)as No_of_Orders
from  (select order_id,
              extract(year from order_purchase_timestamp) as Year,
              extract(month from order_purchase_timestamp) as Month
      from Target.orders) as t
group by Year,Month
order by Year,Month
```

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
15	2017	12	5673
16	2018	1	7269



Inference:

1. The data is missing for the months of Nov-2016, Nov-2018 and Dec-2018.

C Charandeep Reddy
Github : https://github.com/CharanDRC

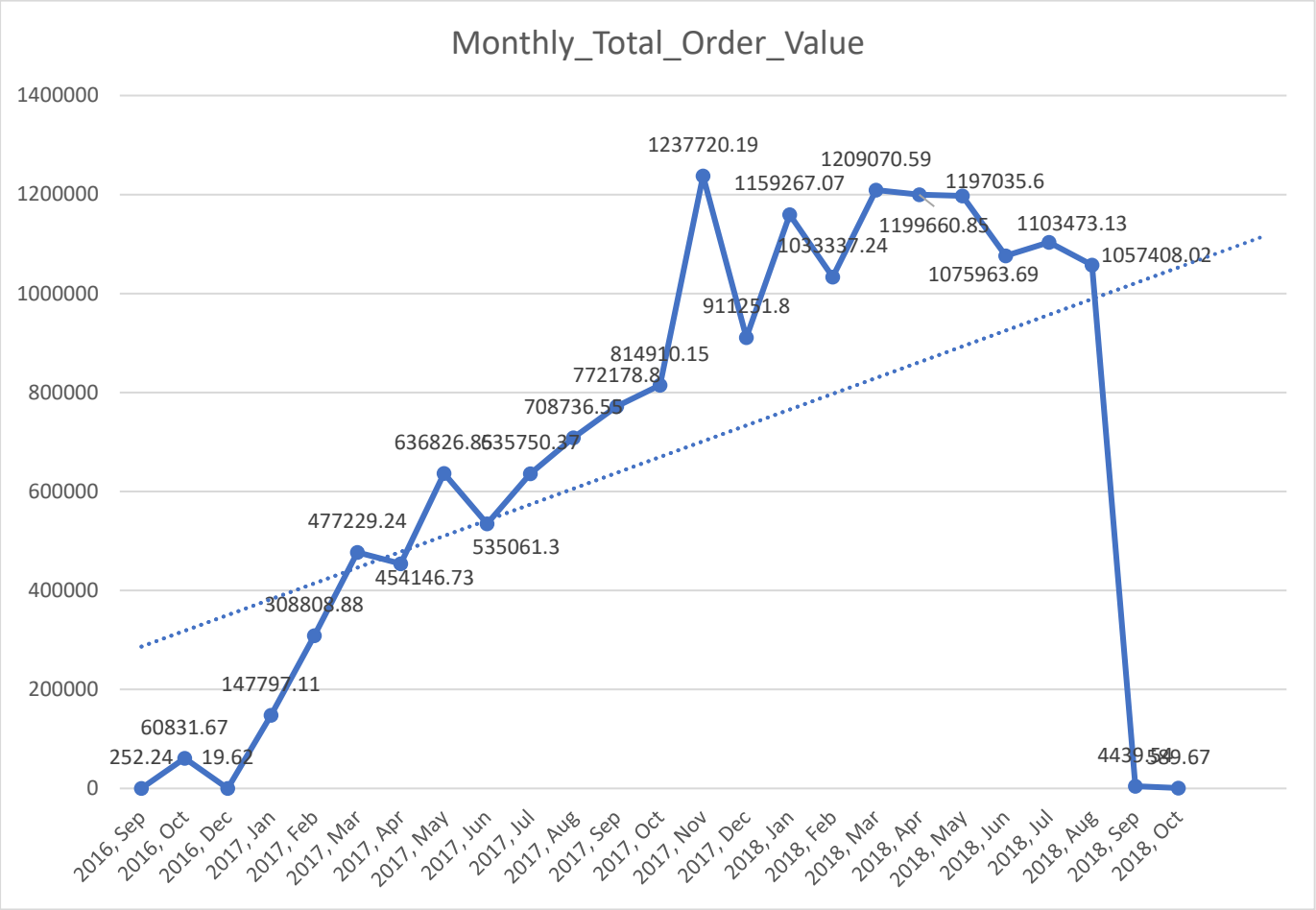
2. And also there are some anomalies in the data, in the years 2016 and 2018 where the no.of orders are quite low. This might be due to error in data entry or some major reason to affect the no.of orders in these months.
3. From the above graph we can note that there is an upward trend(increase in no.of orders) over past years.

Does increase in the no.of orders mean that there is an increase in the sales?

To answer this we need to compare with the revenue with the no.of orders over past years.

```
select Year,Month,round(sum(Total_Order_Value),2)as
Monthly_Total_Order_Value
from  (select o.order_id,
              sum(p.payment_value)over(partition by order_id)as
Total_Order_Value,
              extract(year from o.order_purchase_timestamp) as Year,
              extract(month from o.order_purchase_timestamp) as Month
      from Target.orders o inner join Target.payments p using(order_id)) as
t
group by Year,Month
order by Year,Month
```

Row	Year	Month	Monthly_Total_Order_V
1	2016	9	252.24
2	2016	10	60831.67
3	2016	12	19.62
4	2017	1	147797.11
5	2017	2	308808.88
6	2017	3	477229.24
7	2017	4	454146.73
8	2017	5	636826.85
9	2017	6	535061.3
10	2017	7	635750.37
11	2017	8	708736.55
12	2017	9	772178.8
13	2017	10	814910.15
14	2017	11	1237720.19
15	2017	12	911251.8
16	2018	1	1159267.07



Inference:

As we have seen the no.of orders graph vs years, this total order value strongly graph resembles it. So, we can conclude that there is strong positive co-relation between no.of orders and revenue.

- So we can say that if no.of orders increase then sales increase.

Seasonality: When the data has predictable pattern that repeats itself over time, usually within a year, then we say the there is seasonality.

Ex:- sales of ice-cream usually peaks in summer and dips in winters, sales of Flipkart and Amazon, usually peak during festive sales and remain constant during the rest of the time.

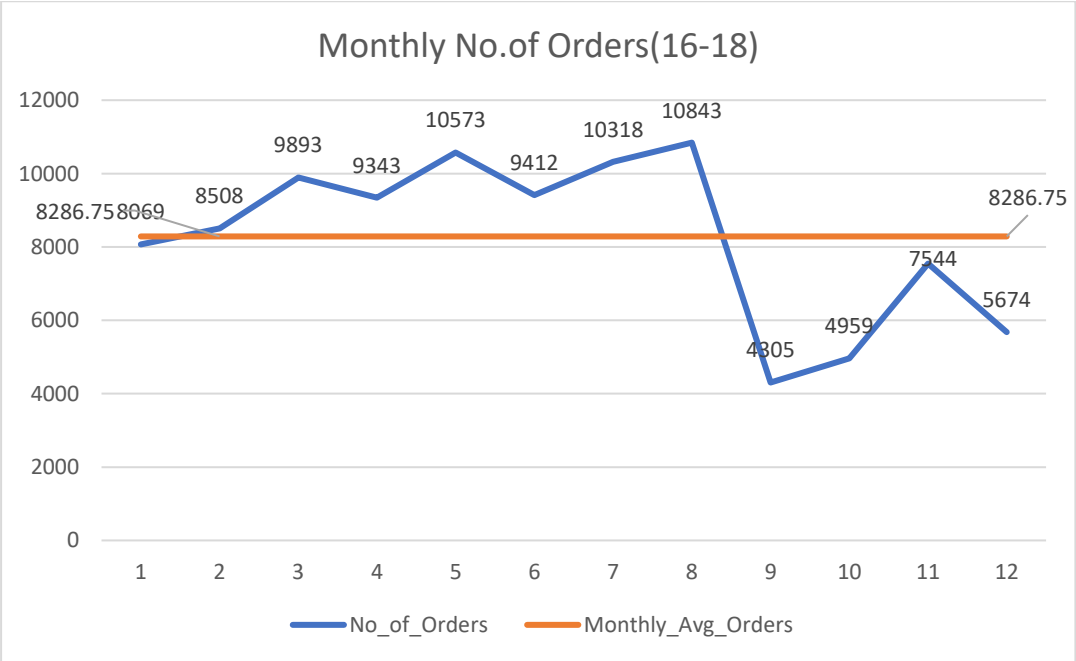
2b. Month on Month analysis of No.of Orders:

```
select month as
Month,No_of_Orders,Monthly_Avg_Orders,if(No_of_Orders>Monthly_Avg_Orders,"Ye
s","No") as Is_greater_than_avg
```

C Charandeep Reddy
Github : https://github.com/CharanDRC

```
from(
  select month,No_of_Orders,round(avg(No_of_Orders)over(),2)as
Monthly_Avg_Orders
  from (
      select extract(month from order_purchase_timestamp)as month,count(*)as
No_of_Orders
      from Target.orders
      group by month)
)
order by month
```

Row	Month	No_of_Orders	Monthly_Avg_Orders	Is_greater_than_avg
1	1	8069	8286.75	No
2	2	8508	8286.75	Yes
3	3	9893	8286.75	Yes
4	4	9343	8286.75	Yes
5	5	10573	8286.75	Yes
6	6	9412	8286.75	Yes
7	7	10318	8286.75	Yes
8	8	10843	8286.75	Yes
9	9	4305	8286.75	No
10	10	4959	8286.75	No
11	11	7544	8286.75	No
12	12	5674	8286.75	No



It requires the grouped table data to be side by side with years so that we can easily compare the month wise patterns in each year and across years and come to some conclusions.

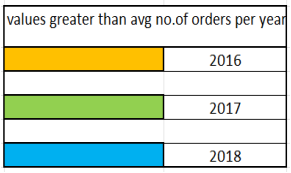
C Charandeep Reddy
Github : https://github.com/CharanDRC

```sql
with
   a as (select  extract(year from order_purchase_timestamp)Year,
        extract(month from order_purchase_timestamp)as month,count(*)as
No_of_Orders
        from Target.orders
        group by Year,month
        having Year=2016),
   b as (select  extract(year from order_purchase_timestamp)Year,
        extract(month from order_purchase_timestamp)as month,count(*)as
No_of_Orders
        from Target.orders
        group by Year,month
        having Year=2017),
   c as (select  extract(year from order_purchase_timestamp)Year,
        extract(month from order_purchase_timestamp)as month,count(*)as
No_of_Orders
        from Target.orders
        group by Year,month
        having Year=2018)
select b.month,a.Year
as`Year_2016`,a.No_of_Orders,round(avg(a.No_of_orders)over(),2)as
Avg_No_of_Orders_2016,
       b.Year as
`Year_2017`,b.No_of_Orders,round(avg(b.No_of_orders)over(),2)as
Avg_No_of_Orders_2017,
       c.Year as
`Year_2018`,c.No_of_Orders,round(avg(c.No_of_orders)over(),2)as
Avg_No_of_Orders_2018
from a full join b using(month) full join c using(month)
order by month
```

Row	month	Year_2016	No_of_Orders	Avg_No_of_Orders_2	Year_2017	No_of_Orders_1	Avg_No_of_Orders_2	Year_2018	No_of_Orders_2	Avg_No_of_Orders_2
1	1	null	null	109.67	2017	800	3758.42	2018	7269	5401.1
2	2	null	null	109.67	2017	1780	3758.42	2018	6728	5401.1
3	3	null	null	109.67	2017	2682	3758.42	2018	7211	5401.1
4	4	null	null	109.67	2017	2404	3758.42	2018	6939	5401.1
5	5	null	null	109.67	2017	3700	3758.42	2018	6873	5401.1
6	6	null	null	109.67	2017	3245	3758.42	2018	6167	5401.1
7	7	null	null	109.67	2017	4026	3758.42	2018	6292	5401.1
8	8	null	null	109.67	2017	4331	3758.42	2018	6512	5401.1
9	9	2016	4	109.67	2017	4285	3758.42	2018	16	5401.1
10	10	2016	324	109.67	2017	4631	3758.42	2018	4	5401.1
11	11	null	null	109.67	2017	7544	3758.42	null	null	5401.1
12	12	2016	1	109.67	2017	5673	3758.42	null	null	5401.1

month	Year_2016	No_of_Orders	Avg_No_of_Orders_2016	Year_2017	No_of_Orders	Avg_No_of_Orders_2017	Year_2018	No_of_Orders	Avg_No_of_Orders_2018
1	-	-	109.67	2017	800	3758.42	2018	7269	5401.1
2	-	-	109.67	2017	1780	3758.42	2018	6728	5401.1
3	-	-	109.67	2017	2682	3758.42	2018	7211	5401.1
4	-	-	109.67	2017	2404	3758.42	2018	6939	5401.1
5	-	-	109.67	2017	3700	3758.42	2018	6873	5401.1
6	-	-	109.67	2017	3245	3758.42	2018	6167	5401.1
7	-	-	109.67	2017	4026	3758.42	2018	6292	5401.1
8	-	-	109.67	2017	4331	3758.42	2018	6512	5401.1
9	2016	4	109.67	2017	4285	3758.42	2018	16	5401.1
10	2016	324	109.67	2017	4631	3758.42	2018	4	5401.1
11	-	-	109.67	2017	7544	3758.42	-	-	5401.1
12	2016	1	109.67	2017	5673	3758.42	-	-	5401.1

C Charandeep Reddy
Github : https://github.com/CharanDRC

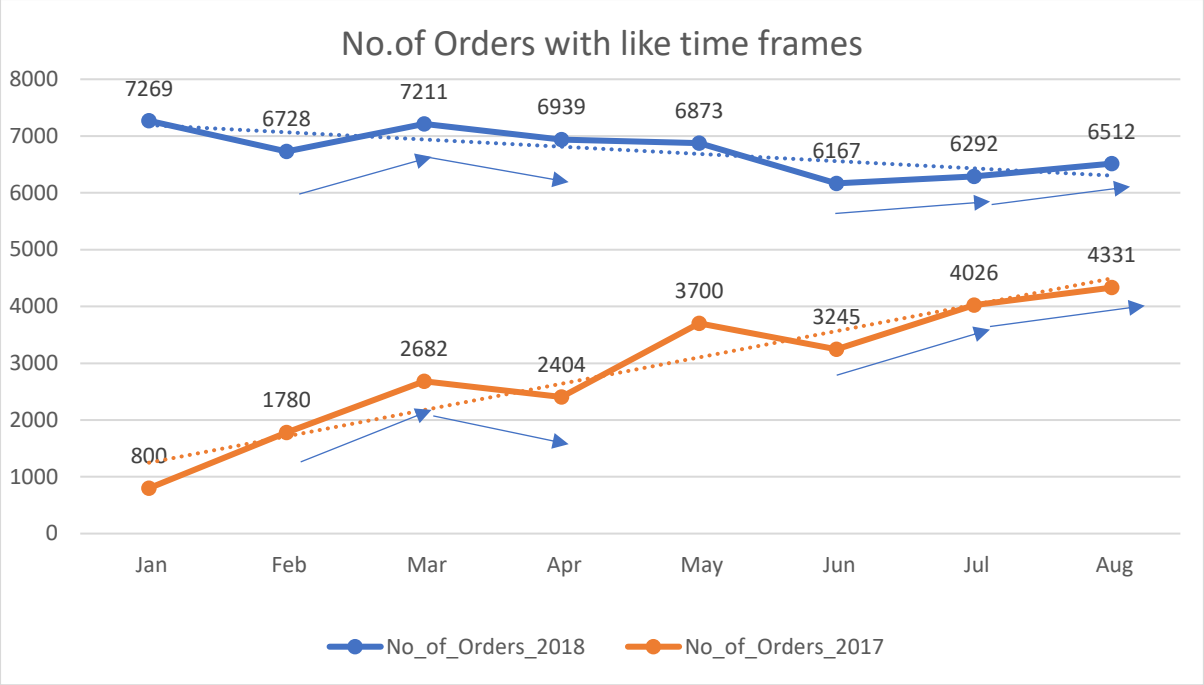


Inference from the above table:

- Missing data and anomalies in the years 2016 and 2018 as mentioned before, makes the month on month analysis difficult, to deal with this we consider like time frames and compare the no.of orders.
- You can see the coloured cells where the no.of orders are above the yearly avg value of no.of orders which points out that the orders count have improved from 2017 july.
- There is an clear increase in the no.of orders(sales) in the months of Jul and Aug which are common in both 2017 and 2018 years. We need to find out which factor is influencing this seasonality.

Comparing like time frames to check monthly seasonality:

We have data like time frame data for the years from 2017 Jan-2017 Aug to 2018 Jan-2018Aug.



Common Trend Lines

Inference from above graph:

- You can view the trend lines common in the both equal time frames of no.of orders for the years 2017 and 2018
- The months of March, June and August have seen the increase in no.of orders(Sales).

Inference:

Despite the availability of complete data, it is challenging to make solid conclusions regarding seasonality. However, from the analysis and visualization, we can observe some seasonality in the e-commerce orders. The count of orders generally increases from March to August with fluctuations in between.

Increase in orders in February and March is probably due to the Carnival season in Brazil. And in the month of August is due to the Festival de Cachaça.
It is important to note that these conclusions are not solid and require data and further analysis to support seasonality in the no.of orders being placed.

2c. Time with highest orders in a day:

```
select Time_of_Day,count(*)as No_of_orders
from
  (select Hour,
        (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"
          else "Night"
        end)as Time_of_Day
```

C Charandeep Reddy
Github : https://github.com/CharanDRC

```
  from
    (select cast(extract(hour FROM order_purchase_timestamp)as int64) as
Hour
    from Target.orders))
group by Time_of_Day
order by length(Time_of_Day) asc
```

Row	Time_of_Day	No_of_orders
1	Dawn	5242
2	Night	28331
3	Mornings	27733
4	Afternoon	38135

Inference:

Highest number of orders are placed during afternoons in a day.

Insights:

1. Upward trend in no.of orders over past years.
2. Seasonality in no.of orders placed during March, July and August which coincides with festivals in Brazil.
3. Most orders the places during Afternoon of the day.

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

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

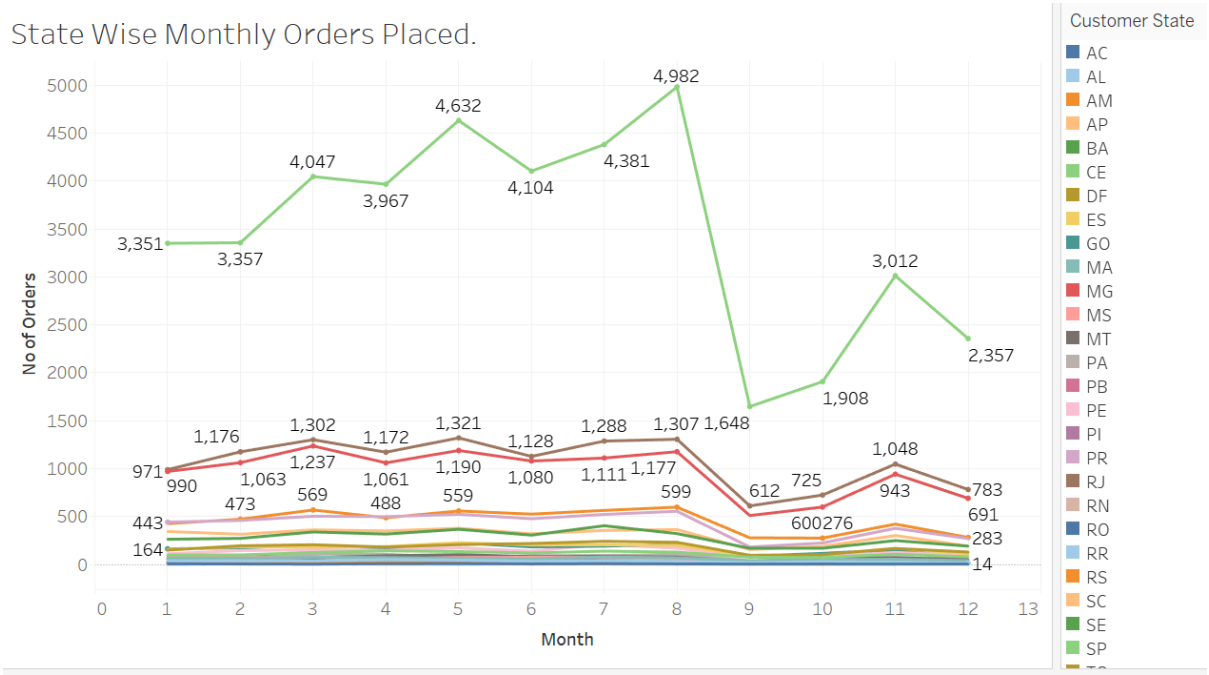
2. How are the customers distributed across all the states?

3a. Month on Month analysis for the no. of orders placed in each state

```
select customer_state,extract(month from order_purchase_timestamp)as
month,count(*)as No_of_Orders
from Target.orders inner join Target.customers using(customer_id)
group by customer_state,month
order by customer_state,month
```

C Charandeep Reddy
Github : https://github.com/CharanDRC

Row	customer_state	month	No_of_Orders
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6
11	AC	11	5
12	AC	12	5
13	AL	1	39
14	AL	2	39
15	AL	3	40
16	AL	4	51



Inference from above:

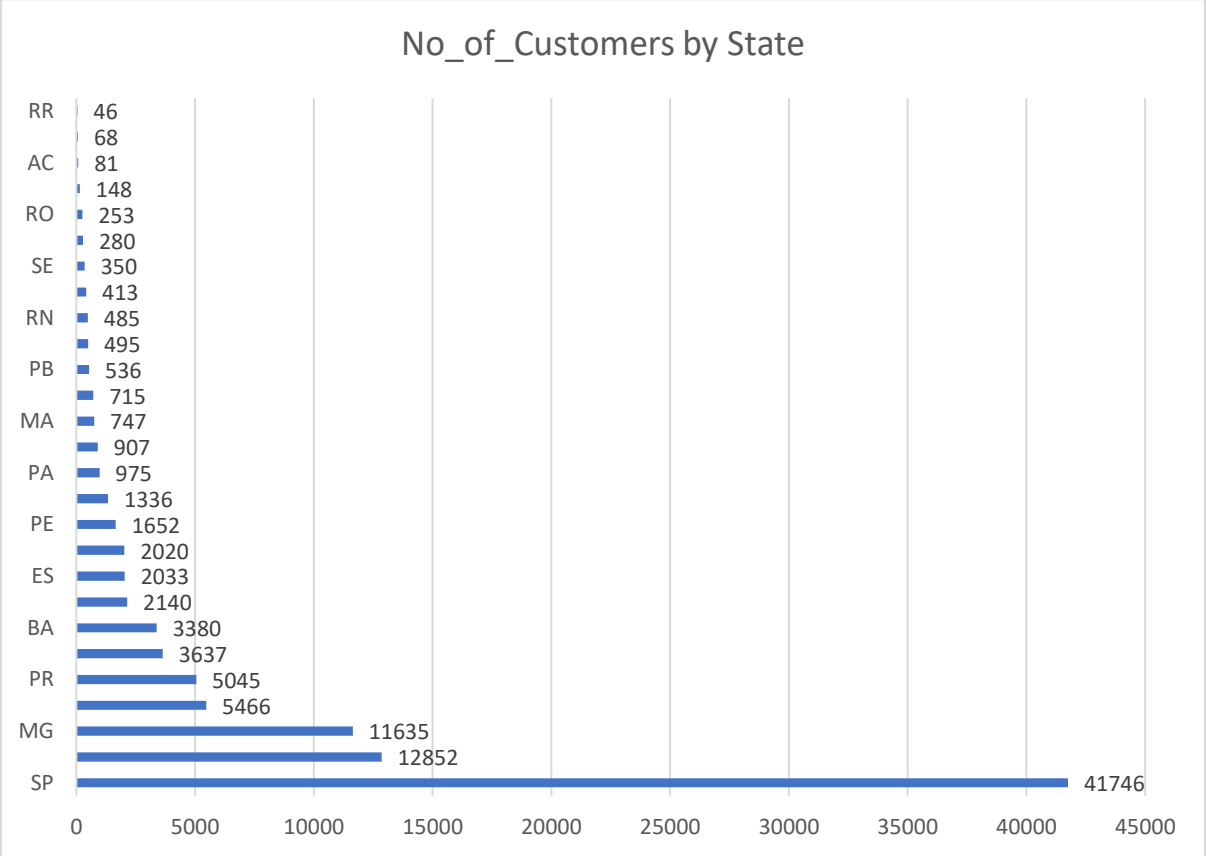
- We can see seasonal trend in orders placed in the top 10 states.
- And also there are few states which contribute more to the orders being placed.
- There are few states with

3b. Customers distributed across all the states

```
select customer_state, count(*)as No_of_Customers
from Target.orders inner join Target.customers using(customer_id)
group by customer_state
order by No_of_Customers desc,customer_state
```

C Charandeep Reddy
Github : https://github.com/CharanDRC

Row	customer_state	No_of_Customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020



Inference from above:

The Sao Paulo(sp) state has the highest no.of customers with 41k, which is greater than the no.of customers combined for the next top 8 states.

Insights:
We can see the same seasonality in orders placed in some states.

C Charandeep Reddy
Github : https://github.com/CharanDRC

The Sao Paulo(sp) state has the highest no.of customers with 41k, which is greater than the no.of customers combined for the next top 8 states.
These top 10 states have many urban centers so the our customers are mostly urban population of Brazil.

There is a relation between the customers distribution and the no.of orders placed from each state. States with high no.of customers have placed more no.of orders compared to other states ex: sp.

5.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.
b. Calculate the Total & Average value of order price for each state.
c. Calculate the Total & Average value of order freight for each state.

4a.

```
select Year,
       Total_cost_of_orders,
       round(((Total_cost_of_orders-
prev_year_total_cost)/prev_year_total_cost)*100,2)as `%_diff_compared_to_prev_year`
from
(
  select *,
         lag(Total_cost_of_orders,1)over(order by Year asc)as prev_year_total_cost
  from
    (
    select  extract(year from order_purchase_timestamp)as Year ,
            round(sum(payment_value),3)as Total_cost_of_orders
    from Target.orders o inner join Target.payments p using(order_id)
    where extract(month from order_purchase_timestamp) between 1 and 8
    group by Year
    having Year=2017 or Year=2018
    )
)
order by Year asc
```

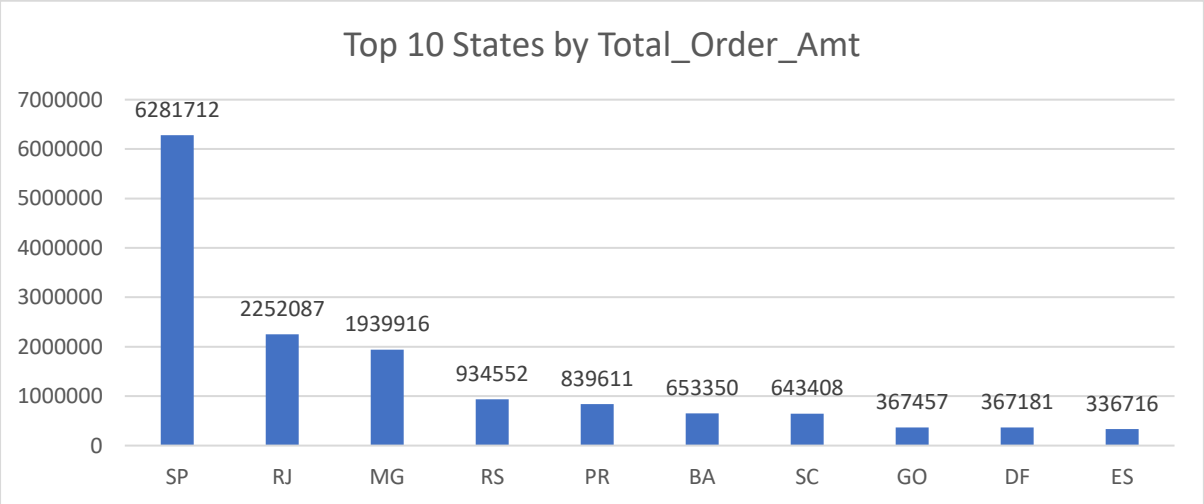
Row	Year	Total_cost_of_orders	%_diff_compared_to_prev_year
1	2017	3669022.12	null
2	2018	8694733.84	136.98

Insights:
There is an growth of 136.98% in the orders being placed in 2018 compared to that of 2017.(Jan-Aug)

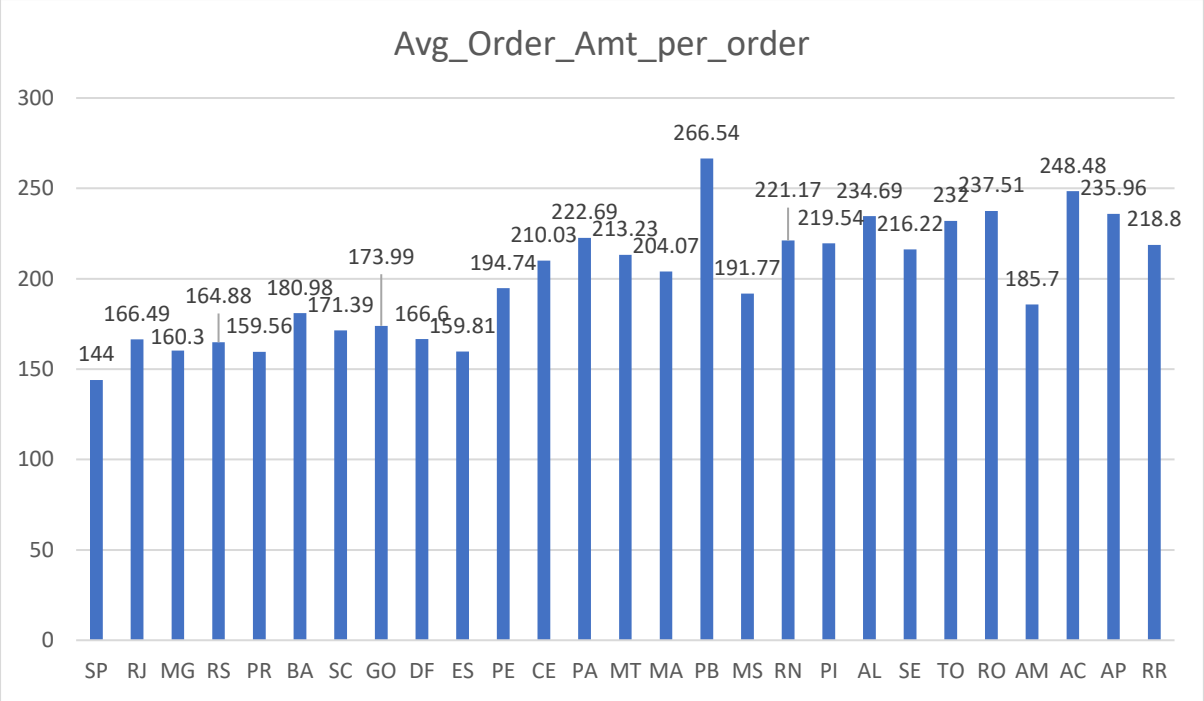
4b.

```sql
create view Target.Total_Payments as
  (select order_id,sum(payment_value)over(partition by order_id) as Total_Amt
  from Target.payments)

select
  customer_state,
  round(sum(Total_Amt),2) as Total_Order_Amt,
  round(avg(Total_Amt),2)as Avg_Order_Amt_per_order
from
  Target.Total_Payments p inner join Target.orders o
    using(order_id)
  inner join Target.customers c
    using(customer_id)
group by customer_state
order by  Total_Order_Amt desc
```

Row	customer_state	Total_Order_Amt	Avg_Order_Amt_per_order
1	SP	6281712.17	144.0
2	RJ	2252086.98	166.49
3	MG	1939915.56	160.3
4	RS	934551.87	164.88
5	PR	839611.15	159.56
6	BA	653350.05	180.98
7	SC	643407.92	171.39
8	GO	367457.13	173.99
9	DF	367180.57	166.6
10	ES	336715.81	159.81
11	PE	336516.38	194.74
12	CE	293626.88	210.03
13	PA	225136.73	222.69
14	MT	204275.05	213.23



C Charandeep Reddy
Github : https://github.com/CharanDRC

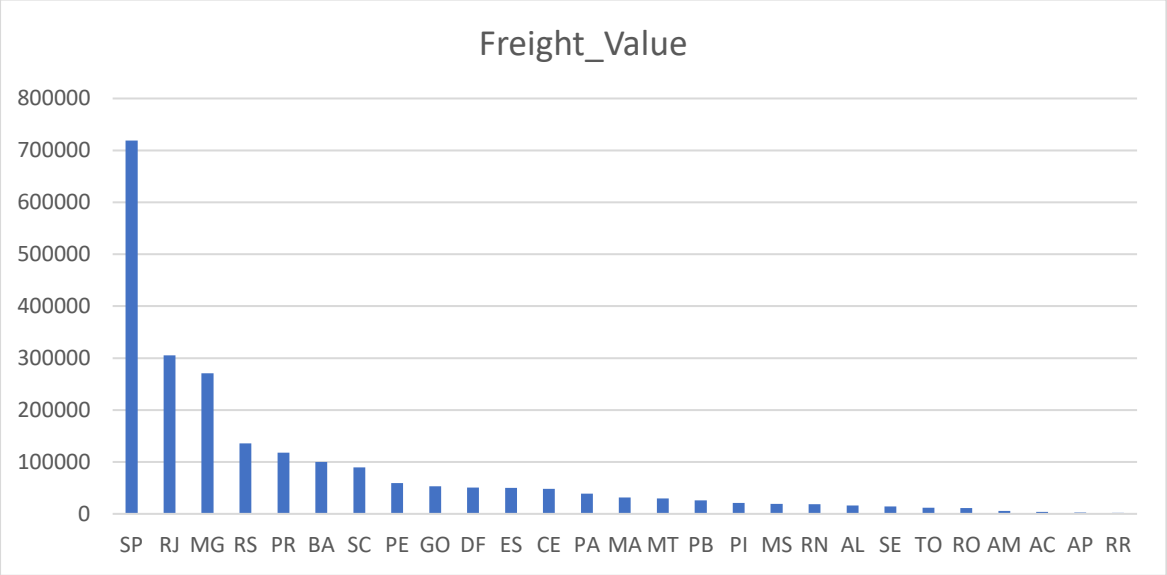


4c.

```
with

a as  (select
         order_id,
         round(sum(freight_value),2)as total_freight_value
       from Target.order_items
       group by order_id),

b as (select*
       from Target.orders),

c as (select*
       from Target.customers)

select customer_state,
       round(sum(total_freight_value),2)as Freight_Value,
       round(avg(total_freight_value),2)as Avg_Freight_value
from a inner join b using(order_id) inner join c using(customer_id)
group by customer_state
order by Freight_Value desc
```

C Charandeep Reddy
Github : https://github.com/CharanDRC

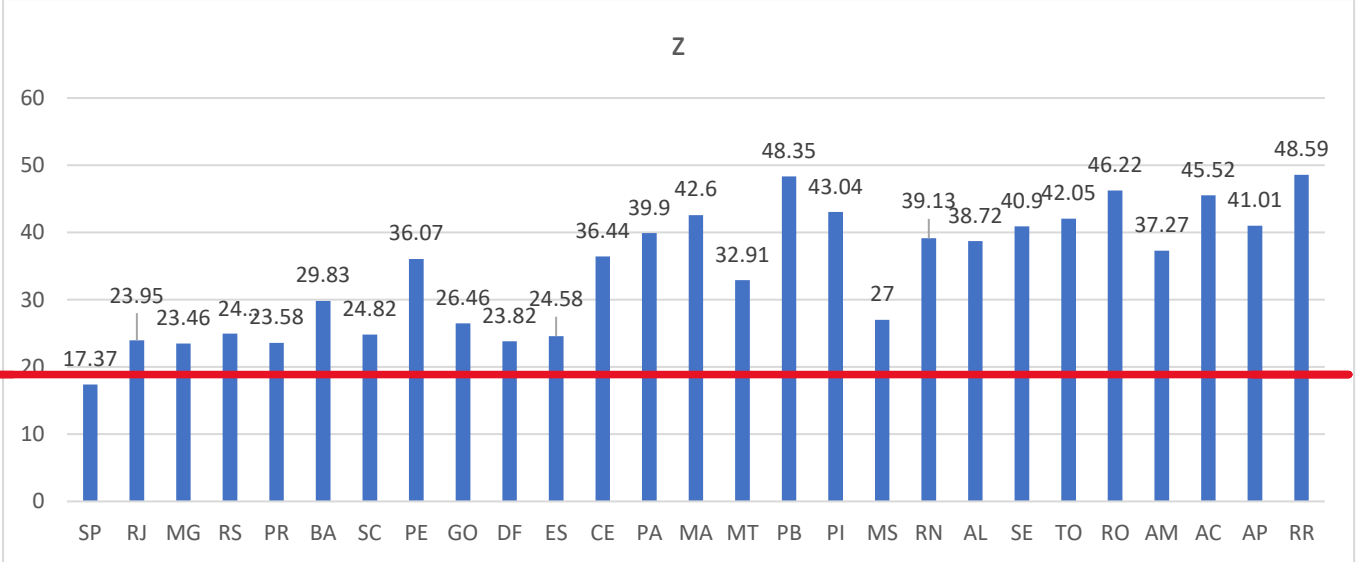
Row	customer_state	Freight_Value	Avg_Freight_value
1	SP	718723.07	17.37
2	RJ	305589.31	23.95
3	MG	270853.46	23.46
4	RS	135522.74	24.95
5	PR	117851.68	23.58
6	BA	100156.68	29.83
7	SC	89660.26	24.82
8	PE	59449.66	36.07
9	GO	53114.98	26.46
10	DF	50625.5	23.82
11	ES	49764.6	24.58
12	CE	48351.59	36.44
13	PA	38699.3	39.9
14	MA	31523.77	42.6



Inference from above graph:

States like 'SP', 'RJ' and 'MG'…. being the states with huge customer base and highest no.of orders being placed comprises the huge portion of freight value which we can know that more orders means much delivery and transportation which rise the freight value.

C Charandeep Reddy
Github : https://github.com/CharanDRC



Inference from above graph:

The states 'RR', 'PB', 'RO' are the top three states with high average freight value.

All the are states above the avg freight value of all states(19.9) except SP.

Insights:

The states with huge customer base and highest no.of orders being placed comprises the huge portion of freight value.

The average freight value of the top orders states is near to the

The states with huge customer base and highest no.of orders being placed comprises the huge portion of freight value.

While São Paulo (SP) has the highest total price value and total freight value, it surprisingly has the lowest average price value and average freight value among all states. On the other hand, the state of Paraíba (PB) has the highest average price value and average freight value.

Understanding the impact on the economy requires a comprehensive analysis of cost trends and price and freight values. By leveraging SQL queries and examining state-wise patterns, businesses can gain valuable insights into the economic landscape, identify potential opportunities for growth, and make data-driven decisions to optimize pricing strategies, enhance logistics, and drive overall economic impact.

2. **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_delivered_customer_date - order_estimated_delivery_date
b. Find out the top 5 states with the highest & lowest average freight value.
c. Find out the top 5 states with the highest & lowest average delivery time.
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.

5.a

```
select *,
  date_diff(order_delivered_customer_date, order_purchase_timestamp,day)as
time_to_deliver ,
  date_diff(order_delivered_customer_date, order_estimated_delivery_date,day) as
diff_estimated_delivery
from Target.orders
```

Row	der_delivered_carrier_date	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliver	diff_estimated_delive
1	)18-02-20 19:57:13 UTC	2018-03-21 22:03:51 UTC	2018-03-09 00:00:00 UTC	30	12
2	)16-10-14 10:40:50 UTC	2016-11-09 14:53:50 UTC	2016-12-08 00:00:00 UTC	30	-28
3	)16-10-25 12:14:28 UTC	2016-11-08 10:58:34 UTC	2016-11-25 00:00:00 UTC	35	-16
4	)17-04-27 16:06:59 UTC	2017-05-16 14:49:55 UTC	2017-05-18 00:00:00 UTC	30	-1
5	)17-04-17 09:08:52 UTC	2017-05-17 10:52:15 UTC	2017-05-18 00:00:00 UTC	32	0
6	)17-04-17 10:44:19 UTC	2017-05-16 09:07:47 UTC	2017-05-18 00:00:00 UTC	29	-1
7	)17-04-25 10:53:00 UTC	2017-05-22 14:11:31 UTC	2017-05-18 00:00:00 UTC	43	4
8	)17-04-12 14:47:39 UTC	2017-05-22 16:18:42 UTC	2017-05-18 00:00:00 UTC	40	4
9	)17-04-19 14:19:04 UTC	2017-05-19 13:44:52 UTC	2017-05-18 00:00:00 UTC	37	1
10	)17-04-26 09:43:45 UTC	2017-05-23 14:19:48 UTC	2017-05-18 00:00:00 UTC	33	5
11	)17-04-19 13:25:07 UTC	2017-05-24 08:11:57 UTC	2017-05-18 00:00:00 UTC	38	6
12	)17-07-12 18:24:53 UTC	2017-08-16 20:19:32 UTC	2017-08-14 00:00:00 UTC	36	2
13	)17-07-14 20:49:34 UTC	2017-08-14 21:37:08 UTC	2017-08-14 00:00:00 UTC	34	0

5.b

```
with

a as  (select
         order_id,
         round(sum(freight_value),2)as total_freight_value
       from Target.order_items
       group by order_id),
```

```
b as (select*
      from Target.orders),

c as (select*
      from Target.customers)

select customer_state,
      round(avg(total_freight_value),2)as Avg_Freight_value
from a inner join b using(order_id) inner join c using(customer_id)
group by customer_state
order by Avg_Freight_value desc
limit 5
```

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

```
with

a as  (select
        order_id,
        round(sum(freight_value),2)as total_freight_value
      from Target.order_items
      group by order_id),

b as (select*
      from Target.orders),

c as (select*
      from Target.customers)

select customer_state,
      round(avg(total_freight_value),2)as Avg_Freight_value
from a inner join b using(order_id) inner join c using(customer_id)
group by customer_state
order by Avg_Freight_value asc
limit 5
```

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

5c.

```
with
```

```
a as(select *,
      date_diff(order_delivered_customer_date, order_purchase_timestamp,day)as
time_to_deliver ,
      date_diff(order_delivered_customer_date, order_estimated_delivery_date,day)
as diff_estimated_delivery
    from Target.orders),
b as(select *
     from Target.customers)

select customer_state,round(avg(time_to_deliver),2) as Avg_Delivery_Time
from a inner join b using(customer_id)
group by customer_state
order by Avg_Delivery_Time desc
limit 5
```

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

```
with
a as(select *,
      date_diff(order_delivered_customer_date, order_purchase_timestamp,day)as
time_to_deliver ,
      date_diff(order_delivered_customer_date, order_estimated_delivery_date,day)
as diff_estimated_delivery
    from Target.orders),
b as(select *
     from Target.customers)

select customer_state,round(avg(time_to_deliver),2) as Avg_Delivery_Time
from a inner join b using(customer_id)
group by customer_state
order by Avg_Delivery_Time asc
limit 5
```

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

5d.

```
with
a as(select *,
      date_diff(order_delivered_customer_date, order_estimated_delivery_date,day)
as diff_estimated_delivery
```

```
    from Target.orders),
b as(select *
     from Target.customers)

select customer_state,
      round(avg(diff_estimated_delivery),2) as `Avg_Diff_Actual vs Estimated`
from a inner join b using(customer_id)
group by customer_state
order by `Avg_Diff_Actual vs Estimated` asc
limit 5
```

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

```
with
a as(select *,
      date_diff(order_delivered_customer_date, order_estimated_delivery_date,day)
as diff_estimated_delivery
    from Target.orders),
b as(select *
     from Target.customers)

select customer_state,
      round(avg(diff_estimated_delivery),2) as `Avg_Diff_Actual vs Estimated`
from a inner join b using(customer_id)
group by customer_state
order by `Avg_Diff_Actual vs Estimated` desc
limit 5
```

Row	customer_state	Avg_Diff_Actual vs Estimated
1	AL	-7.95
2	MA	-8.77
3	SE	-9.17
4	ES	-9.62
5	BA	-9.93

Insights:

Recommendations:

C Charandeep Reddy
Github : https://github.com/CharanDRC

6.Analysis based on the payments:

a. Find the month on month no. of orders placed using different payment types.
b. Find the no. of orders placed on the basis of the payment installments that have been paid.

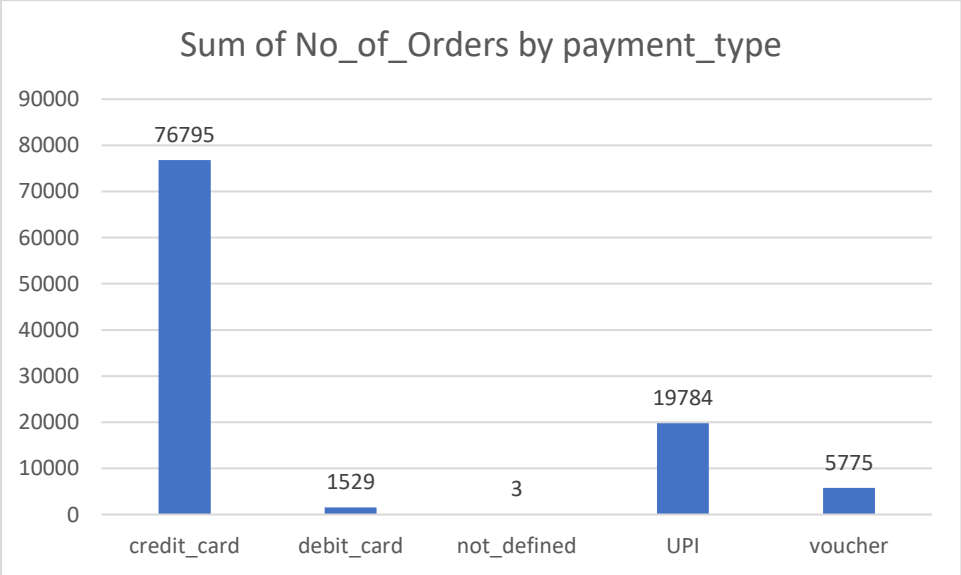
6a. Monthly analysis over no. of orders placed using different payment methods.

```
select distinct payment_type
from Target.payments
```

Row	payment_type
1	credit_card
2	voucher
3	not_defined
4	debit_card
5	UPI

Inference:

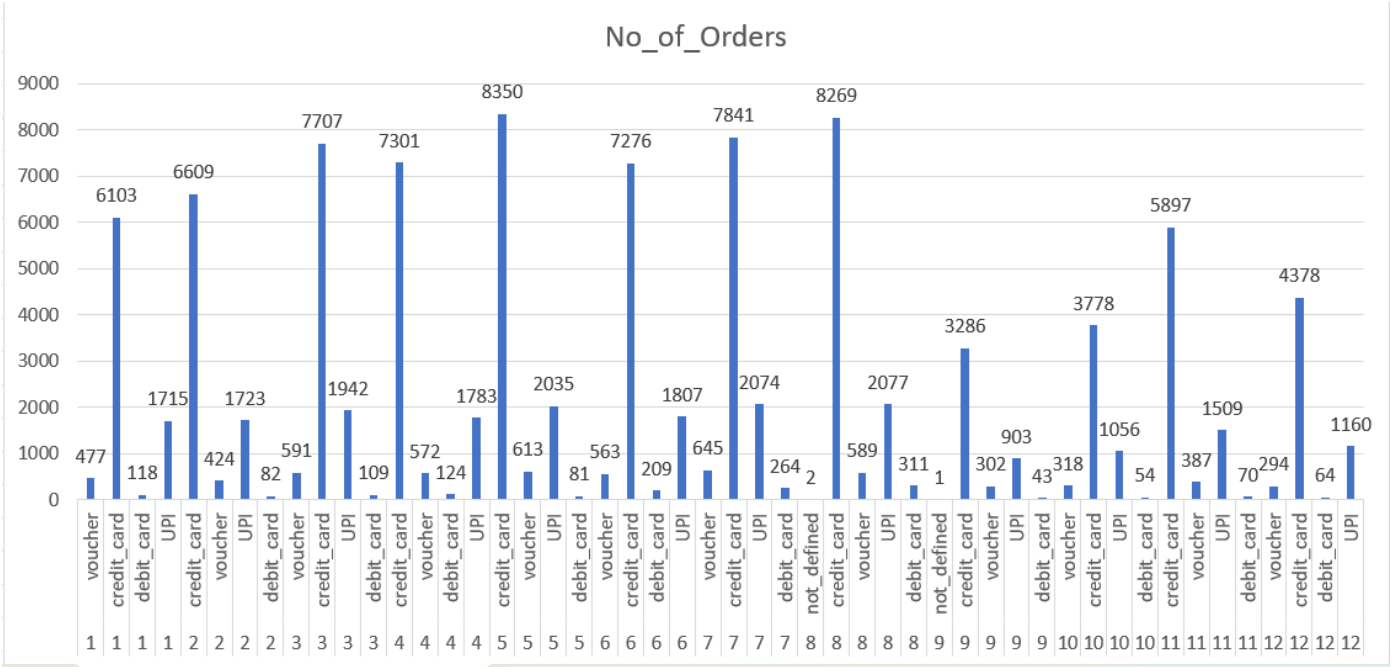
- There are five different payment types.




```
select Month,payment_type,count(*)as No_of_Orders
from(
  select payment_type,extract(month from o.order_purchase_timestamp)as Month
  from Target.payments p inner join Target.orders o using(order_id)
```

```
)
group by Month,payment_type
order by Month
```

Row	Month	payment_type	No_of_Orders
1	1	voucher	477
2	1	credit_card	6103
3	1	debit_card	118
4	1	UPI	1715
5	2	credit_card	6609
6	2	voucher	424
7	2	UPI	1723
8	2	debit_card	82
9	3	voucher	591
10	3	credit_card	7707
11	3	UPI	1942
12	3	debit_card	109
13	4	credit_card	7301
14	4	voucher	572
15	4	debit_card	124
16	4	UPI	1783



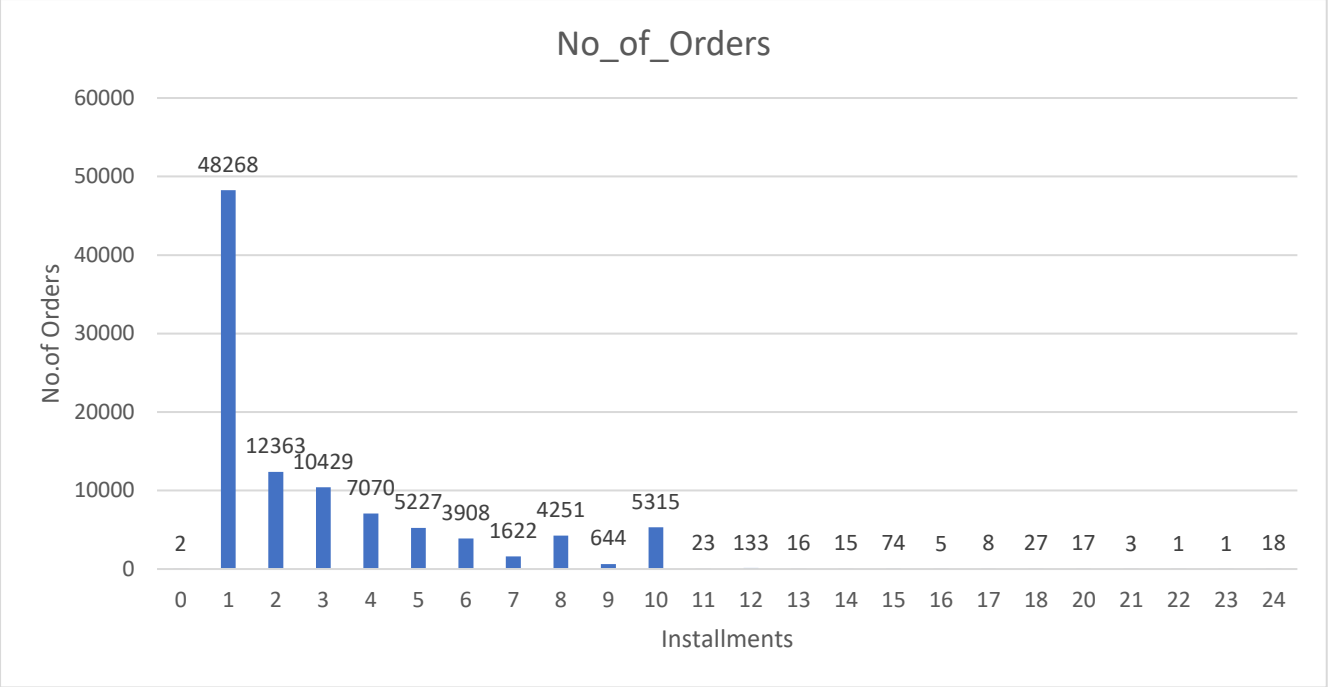
C Charandeep Reddy
Github : https://github.com/CharanDRC

6b. Orders vs Payment Installments

```
select Payment_Installments, count(*)as No_of_Orders
from(
  select distinct order_id,max(payment_installments)over(partition by
order_id)as Payment_Installments
  from Target.payments
)
group by Payment_Installments
order by Payment_Installments
```

Row	Payment_Installment	No_of_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
13	12	133
14	13	16
15	14	15
16	15	74

C Charandeep Reddy
Github : https://github.com/CharanDRC



Insights:

Credit card transactions are the most popular payment method, followed by UPI. Debit card transactions are the least preferred option. Notably, credit card transactions are rapidly increasing compared to other payment methods, possibly due to benefits like “buy now, pay later” options or cashback received using credit cards.

The analysis reveals that the majority of orders (maximum count) have only one payment installment. The highest number of installments is 24, which is associated with 18 orders.

In conclusion, the analysis provides valuable insights into payment types and installment preferences. It highlights the popularity of credit card transactions, the increasing trend of credit card usage, and the prevalence of single-payment installment orders. These insights can help businesses align their payment strategies and improve customer satisfaction.

C Charandeep Reddy
Github : https://github.com/CharanDRC

Unveiling Hidden Gems: Actionable Insights for Brazilian E-commerce Success

Big Opportunity Beyond SP: While São Paulo reigns supreme, a treasure trove of potential lies in other Brazilian states. Spreading the love with targeted efforts can significantly expand your customer base.

Seasonal Surprises: Brazilians love to celebrate! Align your marketing and sales with festive seasons to maximize sales and keep customers happy. It's a win-win!

Speedy Deliveries, Satisfied Customers: Long delivery times can leave a bad taste. Invest in efficient logistics and reliable shipping to keep your customers coming back for more.

Loyalty Pays: São Paulo and Rio de Janeiro are already on board, but there's room for more! Implement loyalty programs, personalized marketing, and exceptional customer service to turn buyers into lifelong fans.

Know Your Audience, Grow Your Sales: Understanding customer demographics unlocks the key to product and marketing personalization. Speak directly to their needs and watch your sales soar.

Beat the Off-Season Blues: Don't let September and October blues impact your bottom line. Strategic discounts and promotions can turn these months into opportunities.

Economic Impact: A Key Ingredient: While not included in this data, economic conditions play a big role. Stay informed to adapt your strategy and ensure long-term success.

Recommendations:

- **Upgrade Your Delivery Game:** Optimize warehouses, refine routes, and choose reliable partners to speed up deliveries and keep customers smiling.
- **Turn One-Timers into Loyal Fans:** Reward repeat business with loyalty programs and personalized offers. Referral programs can also spread the love.
- **Price Right, Win Big:** Analyze pricing and delivery fees to stay competitive. Don't be afraid to adjust when necessary.
- **Tech Up Your Business:** Chatbots, website improvements, and personalized recommendations based on buying habits will take your e-commerce experience to the next level.
- **Partner Up for Success:** Collaborate with sellers to expand your product range and cater to diverse customer preferences.
- **The Power of Influence:** Social media and influencers are huge in Brazil. Leverage their reach to promote your products and build brand awareness.
- **Customer Service is King:** Offer exceptional customer service through chat support and prompt responses to inquiries. A happy customer is a loyal customer!
- **Keep Your Eye on the Competition:** Stay updated on competitor moves and adapt your strategy accordingly. Be it pricing, product offerings, or customer service, always strive to be the best.

C Charandeep Reddy
Github : https://github.com/CharanDRC

By following these actionable insights and recommendations, you can unlock the full potential of the Brazilian e-commerce market.

Done by :-

Name: C Charandeep Reddy

Github: https://github.com/CharanDRC

Linkedin: https://www.linkedin.com/in/c-charandeep-reddy-294855275/