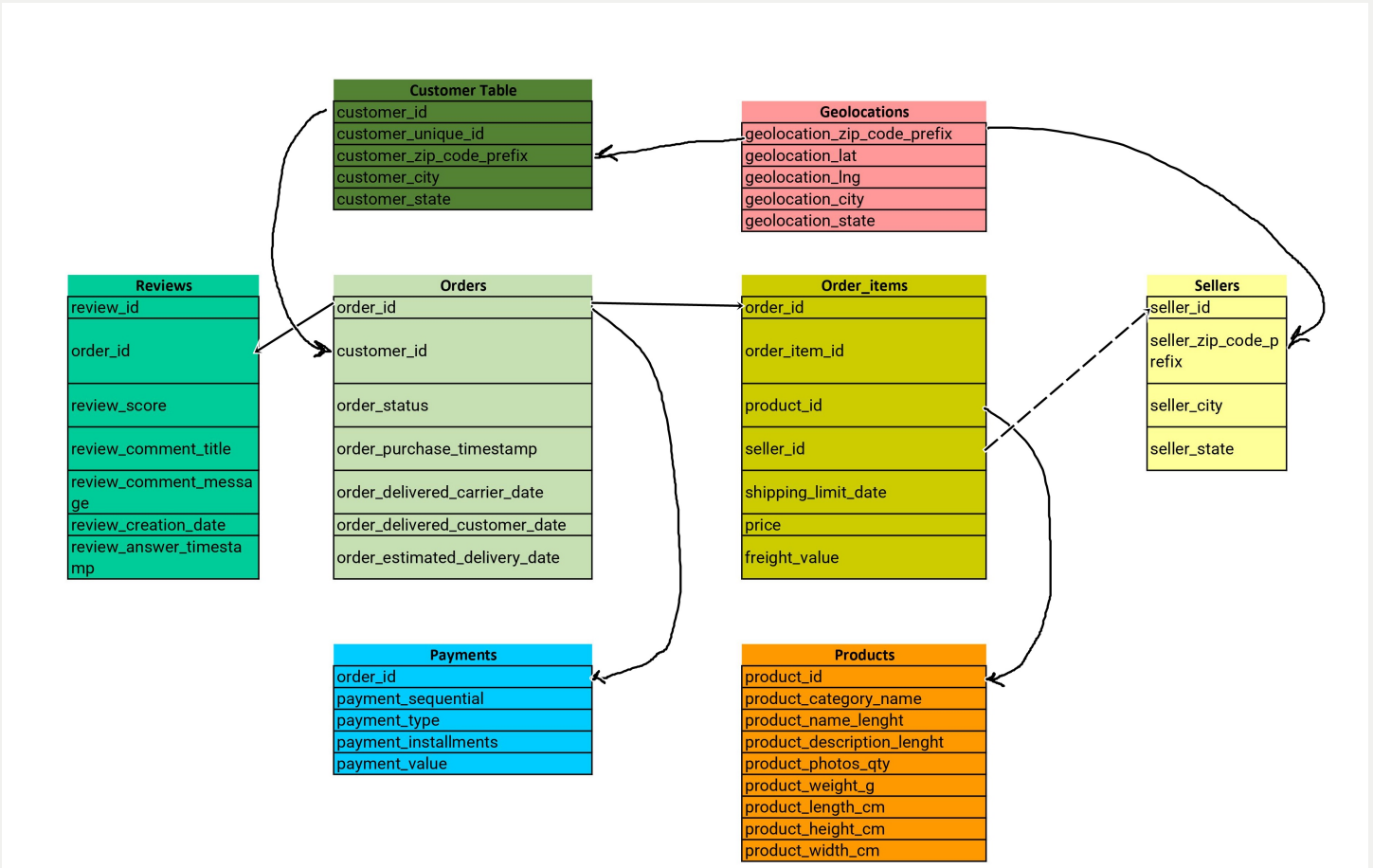
Brazilian Retailer Gaint EDA : Using SQL

For better viewing experience, Click on this link: -

Brazilian Retailer Gaint EDA : Using SQL

Database Schema: -



▼ 1. Basic exploratory analysis steps like checking the structure and characteristics of the given dataset.

▼ a. Data type of columns in a table

```
SELECT
  table_name,
  column_name,
  data_type
  FROM
dsml-targetproject.ecommerce_dataset.INFORMATION_SCHEMA.COLUMNS
```

Output: -

Row	table_name	column_name	data_type
1	order_items	order_id	STRING
2	order_items	order_item_id	INT64
3	order_items	product_id	STRING
4	order_items	seller_id	STRING
5	order_items	shipping_limit_date	TIMESTAMP
6	order_items	price	FLOAT64
7	order_items	freight_value	FLOAT64
8	sellers	seller_id	STRING
9	sellers	seller_zip_code_prefix	INT64
10	sellers	seller_city	STRING
11	sellers	seller_state	STRING

▼ b. Time period for which the data is given

```
SELECT
  MIN(Extract (date from (order_purchase_timestamp))) AS first_date_of_dataset,
  MAX(Extract (date from (order_purchase_timestamp))) AS Last_date_of_dataset,
  Concat(
        ROUND(
          DATE_DIFF(
                    MAX(Extract (date from (order_purchase_timestamp))),
                    MIN(Extract (date from (order_purchase_timestamp))),
                     day)/365,2),' Years')
        AS DURATION
FROM
    `ecommerce_dataset.orders`
```

Outpu: -

Row	first_date_of_dataset	Last_date_of_dataset	DURATION
1	2016-09-04	2018-10-17	2.12 Years

- Considering the most relevant data time related, which is from the orders table, the total duration between the first_order_date and the last_order_date has been presented.

▼ c. Cities and States covered in the datasets

▼ Basic info of Customer and Seller’s states and cities individually

- Total number of different cities and states in which **customers** are registered.

```
SELECT
  Count(Distinct customer_city) as Total_cities,
  Count(Distinct customer_state) as Total_states
FROM
  `dsml-targetproject.ecommerce_dataset.customers`
```

  - Output: -

Row	Total_cities	Total_states
1	4119	27

- Total number of different cities and states in which **sellers** are registered.

```
SELECT
  Count(Distinct seller_city) as Total_cities,
  Count(Distinct seller_state) as Total_states
FROM
  `ecommerce_dataset.sellers`
```

- Output: -

Row	Total_cities	Total_states
1	611	23

▼ 2. In-depth Exploration:

▼ Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?

```
WITH  cte AS
(

         SELECT
            EXTRACT(Month
            FROM
              O.order_purchase_timestamp) AS Months,
            EXTRACT(Year
            FROM
              O.order_purchase_timestamp) AS Years,
            ROUND(SUM(i.price) OVER(PARTITION BY EXTRACT(Year FROM O.order_purchase_timestamp),
                EXTRACT(Month
                FROM
                  O.order_purchase_timestamp))) AS Total_monthly_order_value,
              ROUND(AVG(i.price) OVER(PARTITION BY EXTRACT(Year FROM O.order_purchase_timestamp),
                EXTRACT(Month
                FROM
                  O.order_purchase_timestamp))) AS Average_monthly_order_value,

              Count(i.price) OVER(PARTITION BY EXTRACT(Year FROM O.order_purchase_timestamp),
                EXTRACT(Month
                FROM
                  O.order_purchase_timestamp)) AS Total_monthly_orders
           FROM
             `dsml-targetproject.ecommerce_dataset.orders` AS O
           JOIN
             `dsml-targetproject.ecommerce_dataset.order_items` AS i
           ON
             i.order_id=o.order_id

)


SELECT
  DISTINCT *
FROM
  CTE
ORDER BY
  years,
  months
```

Output: -

Row	Months	Years	Total_monthly_order_value	Average_monthly_order_value	Total_monthly_orders
1	9	2016	267.0	45.0	6
2	10	2016	49508.0	136.0	363
3	12	2016	11.0	11.0	1
4	1	2017	120313.0	126.0	955
5	2	2017	247303.0	127.0	1951
6	3	2017	374344.0	125.0	3000
7	4	2017	359927.0	134.0	2684
8	5	2017	506071.0	122.0	4136
9	6	2017	433039.0	121.0	3583
10	7	2017	498031.0	110.0	4519

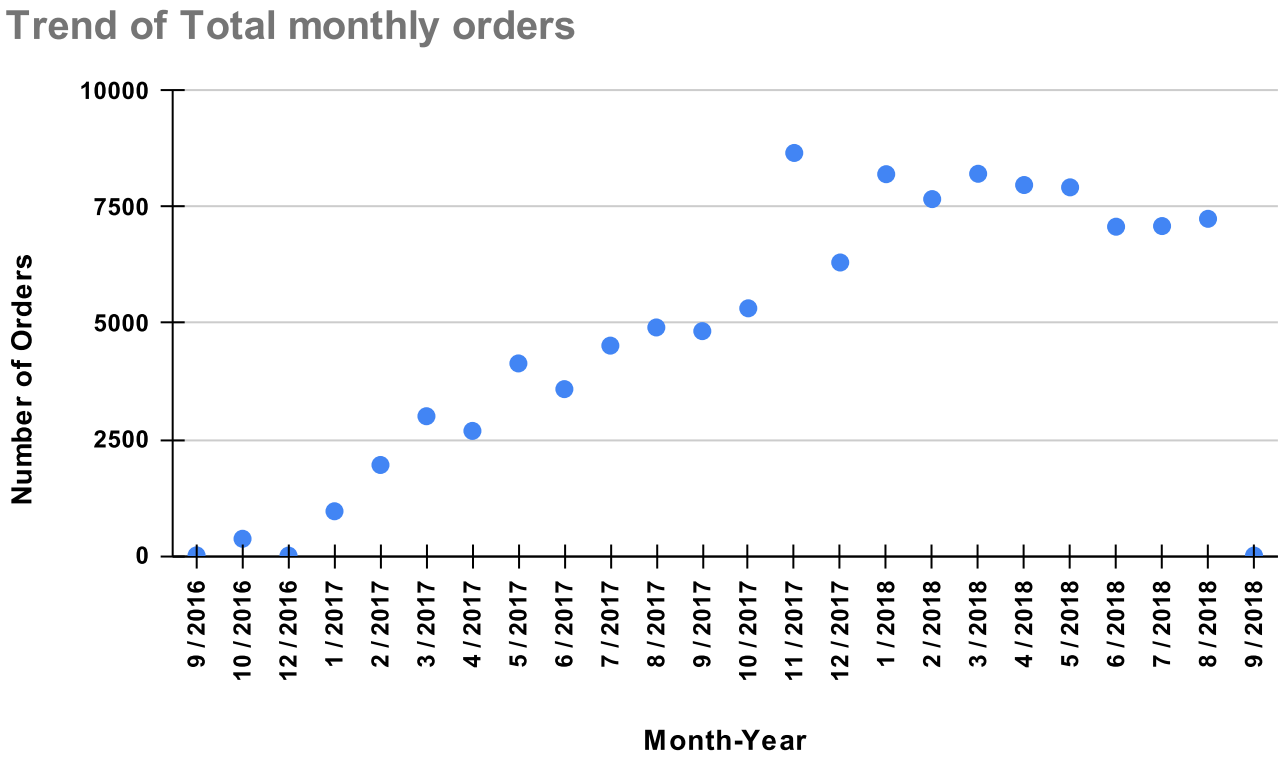
▼ Can we see some seasonality with peaks at specific months?

```
WITH  cte AS
(

         SELECT
            EXTRACT(Month
            FROM
              O.order_purchase_timestamp) AS Months,
            EXTRACT(Year
            FROM
              O.order_purchase_timestamp) AS Years,
            Count(i.price) OVER(PARTITION BY EXTRACT(Year FROM O.order_purchase_timestamp), EXTRACT(Month FROM
                  O.order_purchase_timestamp)) AS Total_monthly_orders
          FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS O
          JOIN
            `dsml-targetproject.ecommerce_dataset.order_items` AS i
          ON
            i.order_id=o.order_id

)


SELECT
  DISTINCT *
FROM
  cte
ORDER BY
  years,
  months
```

Output: -

Row	Months	Years	Total_month...
1	9	2016	6
2	10	2016	363
3	12	2016	1
4	1	2017	955
5	2	2017	1951
6	3	2017	3000
7	4	2017	2684
8	5	2017	4136
9	6	2017	3583

- Total number of orders per month



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

```
WITH  cte AS
(
    SELECT

      order_id,
      Case WHEN Extract(Hour FROM order_purchase_timestamp) between 0 and 5 Then 'Dawn       (0:00 AM to 5:59:59 AM)'
           When Extract(Hour FROM order_purchase_timestamp) between 6 and 11 Then 'Morning      (6:00 AM to 11;59:59 AM)'
           When Extract(Hour FROM order_purchase_timestamp) between 12 and 17 Then 'Afternoon (12:00 PM to 5:59:59 PM)'
           When Extract(Hour FROM order_purchase_timestamp) between 18 and 23 Then 'Night        (6:00 PM to 11:59:59 PM)'
        End as time_of_day
          FROM
            `dsml-targetproject.ecommerce_dataset.orders`


)


SELECT
  time_of_day, Count(order_id) as total_orders
FROM
  CTE
group by
  time_of_day
Order by
  Total_orders
```

- Note: - Considered no time is evening and dawn from 0 to 6
- Output: -

time_of_day	total_orders
Dawn (0:00 AM to 5:59:59 AM)	4740
Morning (6:00 AM to 11;59:59 AM)	22240
Night (6:00 PM to 11:59:59 PM)	34100
Afternoon (12:00 PM to 5:59:59 PM)	38361

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

▼ 1. Get month on month orders by region, states.

```
WITH  cte AS
(

         SELECT
            c.customer_state as state,
            c.customer_city as city,
            EXTRACT(Year FROM O.order_purchase_timestamp) AS Years,
            EXTRACT(Month FROM O.order_purchase_timestamp) AS Months,

            Count(o.order_id) OVER(PARTITION BY  c.customer_state, c.customer_city,
                                   EXTRACT(Year FROM O.order_purchase_timestamp),
                                   EXTRACT(Month FROM O.order_purchase_timestamp)) AS Total_monthly_orders

          FROM
            `dsml-targetproject.ecommerce_dataset.orders` as o
            JOIn `dsml-targetproject.ecommerce_dataset.customers` as c ON c.customer_id=o.customer_id
)


SELECT
  DISTINCT
 *
FROM
  CTE
order by state, city, years, months
```

- Output: -

Row	state	city	Years	Months	Total_monthly_orders
1	AC	brasileia	2017	2	1
2	AC	cruzeiro do sul	2017	12	2
3	AC	cruzeiro do sul	2018	5	1
4	AC	epitaciolandia	2017	10	1
5	AC	manoel urbano	2017	9	1
6	AC	porto acre	2017	4	1
7	AC	rio branco	2017	1	2
8	AC	rio branco	2017	2	2
9	AC	rio branco	2017	3	2

	Years		
State	2016	2017	2018
SP	5	70	88
RJ	4	46	47
MG	4	35	38
PR	3	28	31
RS	3	26	28
BA	1	23	22
SC	2	22	25
ES	1	22	24
PE	2	19	19
GO	2	18	16
PA	1	17	14
DF	1	14	7
MS		13	16
CE	2	15	14
RN	2	13	13
PI	1	12	11
MA	1	12	10
AL	1	12	11
PB	1	13	11
MT	2	11	15
SE	1	11	7
AM		7	7
TO		7	9
RO		7	8
AC		6	5
AP		4	5
RR	1	3	4

▼ 2. How are customers distributed in Brazil?

▼ Total no of distinct states, cities and zipcodes in which customers are present.

```
SELECT
  count(DISTINCT customer_state) as Total_customer_states,
  count(DISTINCT customer_city) as Total_customer_city,
  count(DISTINCT customer_zip_code_prefix) as Total_customer_zipcodes
FROM
  `dsml-targetproject.ecommerce_dataset.customers`
```

- Output: -

Row	Total_customer_states	Total_customer_city	Total_customer_zipcodes
1	27	4119	14994

▼ Total number of customers present in different states.

```
SELECT
  customer_state,
  count(customer_id) as Total_customer
```

```
FROM
  `dsml-targetproject.ecommerce_dataset.customers`
GROUP BY
  customer_state
Order by
  Total_customer desc
```

- Output: -

customer_state	Total_custo...
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033

▼ Total number of customers present in different cities.

```
SELECT
  customer_city,
  count(customer_id) as Total_customer

FROM
  `dsml-targetproject.ecommerce_dataset.customers`
GROUP BY
  customer_city
Order by
  Total_customer desc
```

- Output: -

Row	customer_city	Total_customer
1	sao paulo	15540
2	rio de janeiro	6882
3	belo horizonte	2773
4	brasilia	2131
5	curitiba	1521
6	campinas	1444
7	porto alegre	1379
8	salvador	1245
9	guarulhos	1189

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

▼ Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only)

▼ % difference in each month of 2017 and 2018.

```
With cte as
(
    SELECT
          EXTRACT(Year FROM O.order_purchase_timestamp) AS Years,
          EXTRACT(Month FROM O.order_purchase_timestamp) AS Months,
          Sum(i.price) OVER(PARTITION BY EXTRACT(Year FROM O.order_purchase_timestamp), EXTRACT(Month FROM O.order_purchase
_timestamp)) as Cost_of_monthly_orders
    FROM
                `dsml-targetproject.ecommerce_dataset.orders` AS O
              JOIN
                `dsml-targetproject.ecommerce_dataset.order_items` AS i
              ON
                i.order_id=o.order_id
    Where
          EXTRACT(Month FROM O.order_purchase_timestamp)between 1 and 8
          And
          Extract(Year from O.order_purchase_timestamp) between 2017 and 2018
)

SELECT
      months,
      round(max(CASE When years=2017 Then Cost_of_monthly_orders else NULL END),2) as monthly_order_cost_2017,
      round(max(CASE When years=2018 Then Cost_of_monthly_orders else NULL END),2) as monthly_order_cost_2018,
      concat(round((
        (
          round(max(CASE When years=2018 Then Cost_of_monthly_orders else NULL END),2) -
          round(max(CASE When years=2017 Then Cost_of_monthly_orders else NULL END),2)
        )/round(max(CASE When years=2017 Then Cost_of_monthly_orders else NULL END),2)
      )*100.0,2)," %") as percent_increase
FROM
    cte
Group by
  months
Order by
  months
```

- Output: -

months	monthly_order_cost_2017	monthly_order_cost_2018	percent_increase
1	120312.87	950030.36	689.63 %
2	247303.02	844178.71	241.35 %
3	374344.3	983213.44	162.65 %
4	359927.23	996647.75	176.9 %
5	506071.14	996517.68	96.91 %
6	433038.6	865124.31	99.78 %
7	498031.48	895507.22	79.81 %
8	573971.68	854686.33	48.91 %

▼ % difference of total sales from (Jan to Aug) for 2017 and 2018.

```
With cte as
(
    SELECT

          EXTRACT(Year FROM O.order_purchase_timestamp) AS Years,
          sum(i.price) as Total_cost_of_orders
    FROM
                `dsml-targetproject.ecommerce_dataset.orders` AS O
              JOIN
                `dsml-targetproject.ecommerce_dataset.order_items` AS i
              ON
                i.order_id=o.order_id
    Where
          EXTRACT(Month FROM O.order_purchase_timestamp)between 1 and 8
          And
          Extract(Year from O.order_purchase_timestamp) between 2017 and 2018
    Group by
         EXTRACT(Year FROM O.order_purchase_timestamp)
)


SELECT
```

```
       round(Max(case When Years=2017 Then Total_cost_of_orders else NULL ENd),2) as Total_order_cost_2017,
       round(Max(case When Years=2018 Then Total_cost_of_orders else NULL ENd),2) as Total_order_cost_2018,
       concat(round((
        (
          round(Max(case When Years=2018 Then Total_cost_of_orders else NULL ENd) ,2) -
          round(max(CASE When years=2017 Then Total_cost_of_orders else NULL END),2)
        )/round(max(CASE When years=2017 Then Total_cost_of_orders else NULL END),2)
      )*100.0,2)," %") as percent_increase

FROM
    cte
```

- Output: -

Total_order_cost_2017	Total_order_cost_2018	percent_increase
3113000.32	7385905.8	137.26 %

▼ Mean & Sum of price and freight value by a customer state

```
SELECT
           c.customer_state as State,
           c.customer_city as City,
           round(Sum(i.price),2) as Total_order_value_per_area,
           round(Sum(i.freight_value),2) as Total_shipping_cost_per_area,
           round(avg(i.price),2) as Mean_order_value_per_area,
           round(avg(i.freight_value),2) as Mean_freight_value_per_area

     FROM
           `dsml-targetproject.ecommerce_dataset.orders` AS O
         JOIN
           `dsml-targetproject.ecommerce_dataset.order_items` AS i ON i.order_id=o.order_id
         JOIN `dsml-targetproject.ecommerce_dataset.customers`as c on c.customer_id=O.customer_id
     GROUP BY
       c.customer_state,c.customer_city
     Order by
       total_order_value_per_area,Total_shipping_cost_per_area, Mean_order_value_per_area,Mean_freight_value_per_area;
```

- Output: -

State	City	Total_order_value_per_area	Total_shipping_cost_per_area	Mean_order_value_per_area	Mean_freight_value_per_area
RS	polo petroquimico de triunfo	5.6	15.1	5.6	15.1
PR	sabaudia	5.9	14.52	5.9	14.52
MG	santo antonio do rio abaixo	6.0	18.23	6.0	18.23
PA	senador jose porfirio	6.0	25.63	6.0	25.63
MG	jenipapo de minas	7.48	15.1	7.48	15.1
RJ	santa maria	8.0	16.11	8.0	16.11
BA	erico cardoso	8.09	16.79	8.09	16.79
PR	tamboara	8.99	15.1	8.99	15.1
PA	medicilandia	8.99	34.15	8.99	34.15

▼ 5. Analysis of sales, freight and delivery time

▼ Days between purchasing, delivering and estimated delivery.

- 2. Create columns:

1. time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
2. diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
SELECT
            c.customer_state,
            c.customer_city,
            c.customer_zip_code_prefix,
            date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day) as Time_of_estimated_delivery,
            date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day) as Time_of_actual_delivery,
            (
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ) as Differemce


      FROM
```

```
      `dsml-targetproject.ecommerce_dataset.orders` AS o
JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
WHERE
  lower(o.order_status)='delivered'
Order by
  c.customer_state,
  c.customer_city,
  c.customer_zip_code_prefix
```

- Output: - Here, negative difference means, the order has reached to the customer earlier than the estimated delivery date.

customer_state	customer_city	customer_zi...	Time_of_estimated_delivery	Time_of_actual_delivery	Differemce
AC	brasileia	69932	41	30	-11
AC	cruzeiro do sul	69980	54	36	-18
AC	cruzeiro do sul	69980	64	21	-43
AC	cruzeiro do sul	69980	53	14	-39
AC	epitaciolandia	69934	44	13	-31
AC	manoel urbano	69950	46	12	-34
AC	porto acre	69927	43	29	-14
AC	rio branco	69900	40	41	1
AC	rio branco	69900	41	72	31

▼ Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_estimated
_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_actual_de
livery_time,
            round(Avg(
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state,
        Avg_shipping_cost,
        Avg_estimated_delivery_time,
        Avg_actual_delivery_time
FROM
  cte
```

- Output: -

customer_state	Avg_shipping_cost	Avg_estimated_delivery_time	Avg_actual_delivery_time	Avg_Difference
RR	43.09	46.5	28.17	-18.33
PB	43.09	33.58	20.55	-13.04
RO	41.33	39.7	19.66	-20.04
AC	40.05	41.66	20.68	-20.98
PI	39.12	30.85	19.32	-11.53
MA	38.49	31.5	21.59	-9.91
TO	37.44	29.74	17.4	-12.34
SE	36.57	31.42	21.42	-10.0
AL	35.87	33.18	24.45	-8.74

▼ Sort the data to get the following:

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

▼ Top 5 states with highest frieght value.

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_
estimated_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_
actual_delivery_time,
            round(Avg(
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state, Avg_shipping_cost
FROM
  cte
Order by
  cte.Avg_shipping_cost desc
  LIMIT 5
```

- Output: -

state	Avg_shipping_cost
RR	43.09
PB	43.09
RO	41.33
AC	40.05
PI	39.12

▼ Top 5 states with Lowest frieght value.

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_
estimated_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_
actual_delivery_time,
            round(Avg(
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
```

```
        state


)


SELECT
        state, Avg_shipping_cost
FROM
  cte
Order by
  cte.Avg_shipping_cost asc
  LIMIT 5
```

- Output: -

state	Avg_shipping_cost
SP	15.12
PR	20.47
MG	20.63
RJ	20.91
DF	21.07

▼ Top 5 states with highest/lowest average time to delivery

▼ Top 5 state with highest average time to delivery

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_
estimated_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_
actual_delivery_time,
            round(Avg(
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state, Avg_actual_delivery_time
FROM
  cte
Order by
  cte.Avg_actual_delivery_time desc
  LIMIT 5
```

- Output: -

state	Avg_actual_delivery_time
AP	28.22
RR	28.17
AM	26.34
AL	24.45
PA	23.7

▼ Top 5 state with lowest average time to delivery i.e., Fastest delivery.

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_
estimated_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_
actual_delivery_time,
            round(Avg(
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state, Avg_actual_delivery_time
FROM
  cte
Order by
  cte.Avg_actual_delivery_time asc
  LIMIT 5
```

- Output: -

state	Avg_actual_delivery_time
SP	8.66
PR	11.89
MG	11.92
DF	12.89
SC	14.95

▼ Top 5 states where delivery is speedy/ not so fast compared to the estimated date

▼ Top 5 states where delivery really FAST compared to estimated delivery time

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_
estimated_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_
actual_delivery_time,
            round(Avg(
```

```
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state,
        Avg_estimated_delivery_time,
        Avg_actual_delivery_time,
        Avg_Difference
FROM
  cte
Order by
  cte.Avg_Difference asc
  LIMIT 5
```

- Output: -

state	Avg_estimated_delivery_time	Avg_actual_delivery_time	Avg_Difference
AC	41.66	20.68	-20.98
RO	39.7	19.66	-20.04
AM	46.27	26.34	-19.93
AP	46.62	28.22	-18.4
RR	46.5	28.17	-18.33

▼ Top 5 states where delivery really slow compared to estimated delivery time

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)),2) as Avg_
estimated_delivery_time,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_
actual_delivery_time,
            round(Avg(
              date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)-
              date_diff(Date(o.order_estimated_delivery_date),Date(o.order_purchase_timestamp),day)
            ),2) as Avg_Difference


      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state,
        Avg_estimated_delivery_time,
        Avg_actual_delivery_time,
        Avg_Difference
FROM
  cte
Order by
  cte.Avg_Difference desc
  LIMIT 5
```

- Output: -

state	Avg_estimated_delivery_time	Avg_actual_delivery_time	Avg_Difference
AL	33.18	24.45	-8.74
MA	31.5	21.59	-9.91
SE	31.42	21.42	-10.0
ES	26.23	15.59	-10.65
BA	30.18	19.19	-10.98

▼ No of orders. shipping cost and delivery time over different states.

```
WITH cte as
(
      SELECT
            c.customer_state as state,
            count(o.order_id) as no_of_orders,
            round(avg(i.freight_value),2) as Avg_shipping_cost,
            round(Avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as Avg_actual_de
livery_time,

      FROM
            `dsml-targetproject.ecommerce_dataset.orders` AS o
      JOIN `dsml-targetproject.ecommerce_dataset.customers` as c ON o.customer_id=c.customer_id
      JOIn `dsml-targetproject.ecommerce_dataset.order_items` as i ON o.order_id=i.order_id
      WHERE
        lower(o.order_status)='delivered'
      group by
        state


)


SELECT
        state,
        no_of_orders,
        Avg_shipping_cost,

        Avg_actual_delivery_time
FROM
  cte

ORDER by
  no_of_orders desc
```

- Output: -

state	no_of_orders	Avg_shipping_cost	Avg_actual_delivery_time
SP	46448	15.12	8.66
RJ	14143	20.91	15.07
MG	12916	20.63	11.92
RS	6134	21.61	15.13
PR	5649	20.47	11.89
SC	4097	21.51	14.95
BA	3683	26.49	19.19
DF	2355	21.07	12.89
GO	2277	22.56	15.34
ES	2225	22.03	15.59
PE	1746	32.69	18.22
CE	1426	32.73	20.92

▼ 6. Payment type analysis

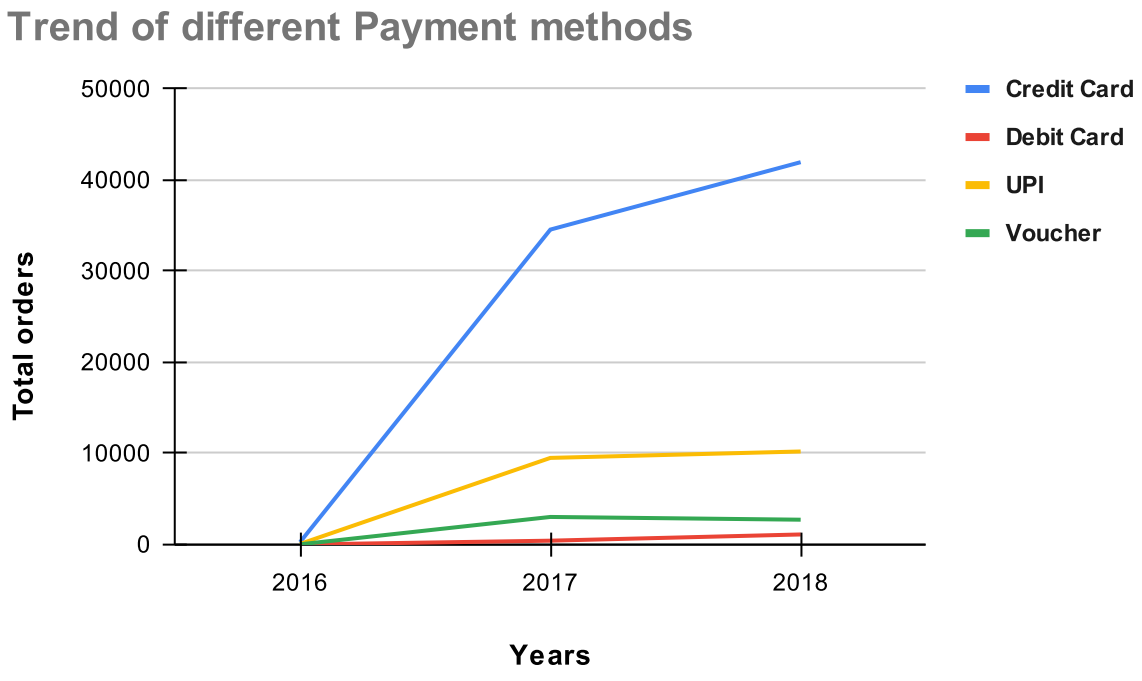
▼ Month over Month count of orders for different payment types.

```
With cte as
(
      SELECT
          EXTRACT(Year FROM O.order_purchase_timestamp) AS Years,
          EXTRACT(Month FROM O.order_purchase_timestamp) AS Months,
          payment_type,
          count(p.order_id) as No_of_orders
      FROM
        `ecommerce_dataset.payments` as p
        JOIN `ecommerce_dataset.orders` as o on o.order_id=p.order_id

      GROUP bY
        EXTRACT(Year FROM O.order_purchase_timestamp),
        EXTRACT(Month FROM O.order_purchase_timestamp),
        payment_type

)

SELECT
      *
FROM
    cte
order by
years,months,no_of_orders
```

- Output: -

Years	Months	payment_type	No_of_orders
2016	9	credit_card	3
2016	10	debit_card	2
2016	10	voucher	23
2016	10	UPI	63
2016	10	credit_card	254
2016	12	credit_card	1
2017	1	debit_card	9
2017	1	voucher	61
2017	1	UPI	197
2017	1	credit_card	583
2017	2	debit_card	13
2017	2	voucher	119
2017	2	UPI	398
2017	2	credit_card	1356

- Insights: -
  - Overall credit card payments are increasing over the years as



▼ EMI vs. Non-EMI Orders

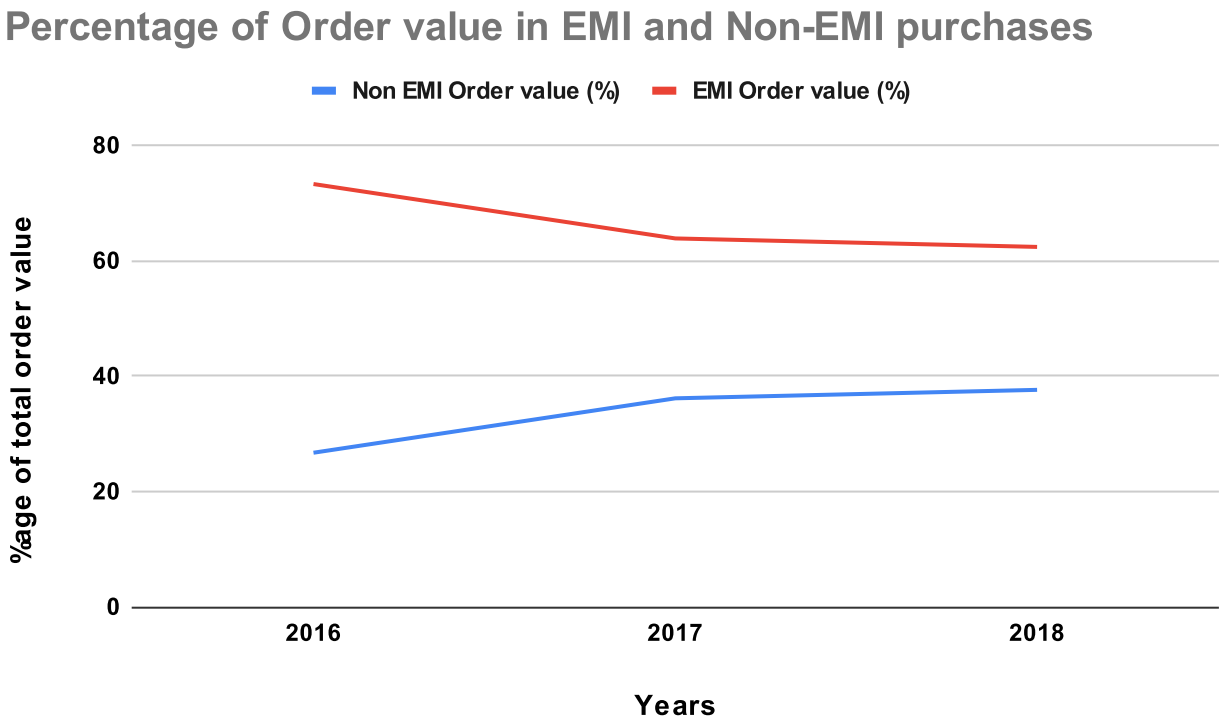
▼ EMI and Non-EMI order value split.

```
With cte as
(
      SELECT
                EXTRACT(Year FROM O.order_purchase_timestamp) AS Years,

                round(sum(Case WHen p.payment_installments<2 Then p.payment_value END),2) as Non_EMI_order_value,
                round(sum(Case WHen p.payment_installments>=2 Then p.payment_value END),2) as EMI_order_value

            FROM
              `ecommerce_dataset.payments` as p
              JOIN `ecommerce_dataset.orders` as o on o.order_id=p.order_id
            Group by
              EXTRACT(Year FROM O.order_purchase_timestamp)
)

SELECT
  years,
    concat(
        round(
        (Non_EMI_order_value/(EMI_order_value+Non_EMI_order_value)*100.0),2
        )," %"
      ) as Per100_Non_EMI_order_value,
    concat(
        round(
        (EMI_order_value/(EMI_order_value+Non_EMI_order_value)*100.0),2
        )," %"
      ) as Per100_EMI_order_value
FROM
    cte
ORDER BY
  years
```

- Output: -

years	Per100_Non_EMI_order_value	Per100_EMI_order_value
2016	26.71 %	73.29 %
2017	36.14 %	63.86 %
2018	37.61 %	62.39 %

- Percenatage of Order value in EMI and Non-EMI purchases



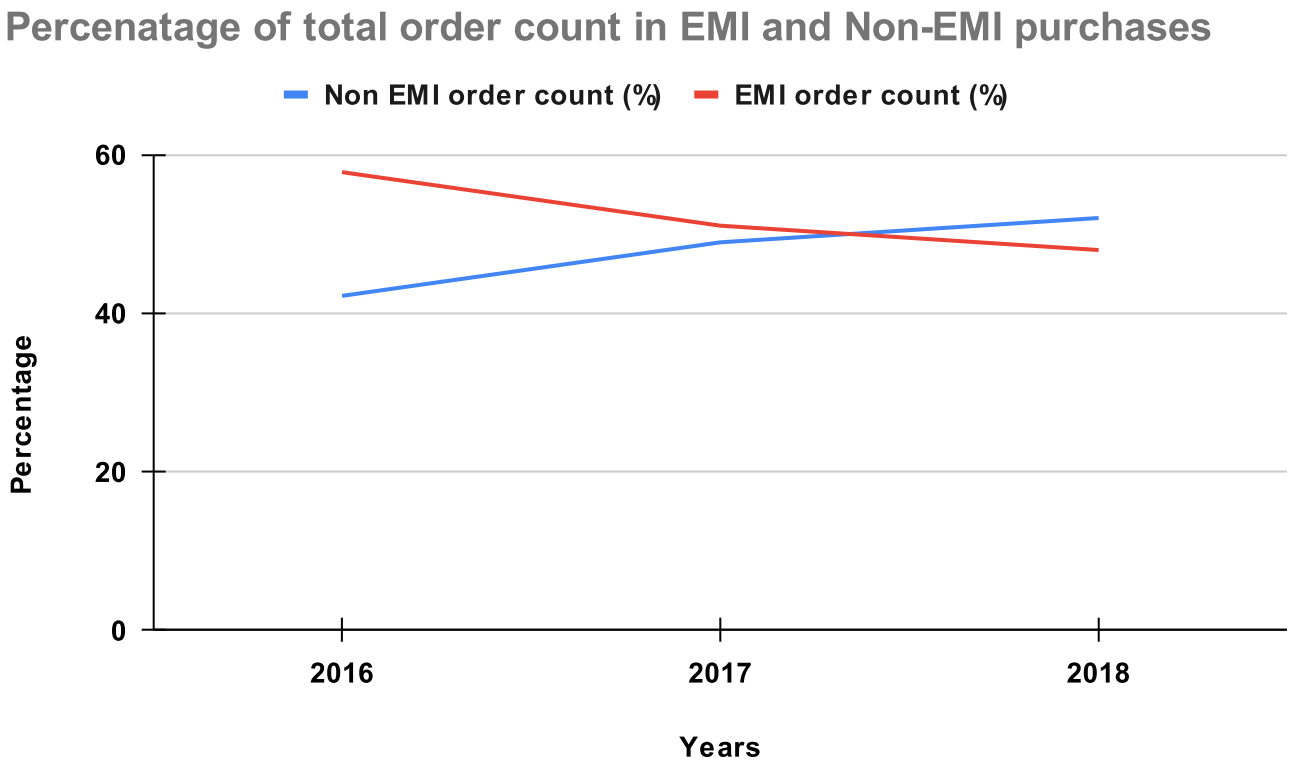
▼ EMI and Non-EMI order count split.

```
With cte as
(
      SELECT
                EXTRACT(Year FROM O.order_purchase_timestamp) AS Years,

                round(count(Case WHen p.payment_installments<2 Then p.order_id END),2) as Non_EMI_order_count,
                round(count(Case WHen p.payment_installments>=2 Then p.order_id END),2) as EMI_order_count

            FROM
              `ecommerce_dataset.payments` as p
              JOIN `ecommerce_dataset.orders` as o on o.order_id=p.order_id
            Group by
              EXTRACT(Year FROM O.order_purchase_timestamp)
)

SELECT
  years,
    concat(
        round(
        (Non_EMI_order_count/(EMI_order_count+Non_EMI_order_count)*100.0),2
        )," %"
      ) as Per100_Non_EMI_order_count,
    concat(
        round(
        (EMI_order_count/(EMI_order_count+Non_EMI_order_count)*100.0),2
        )," %"
      ) as Per100_EMI_order_count
FROM
    cte
ORDER BY
  years
```

- Output: -

years	Per100_Non_EMI_order_count	Per100_EMI_order_count
2016	42.2 %	57.8 %
2017	48.95 %	51.05 %
2018	52.02 %	47.98 %

- Percenatage of total order count in EMI and Non-EMI purchases.



▼ Distribution of payment instalments and count of orders

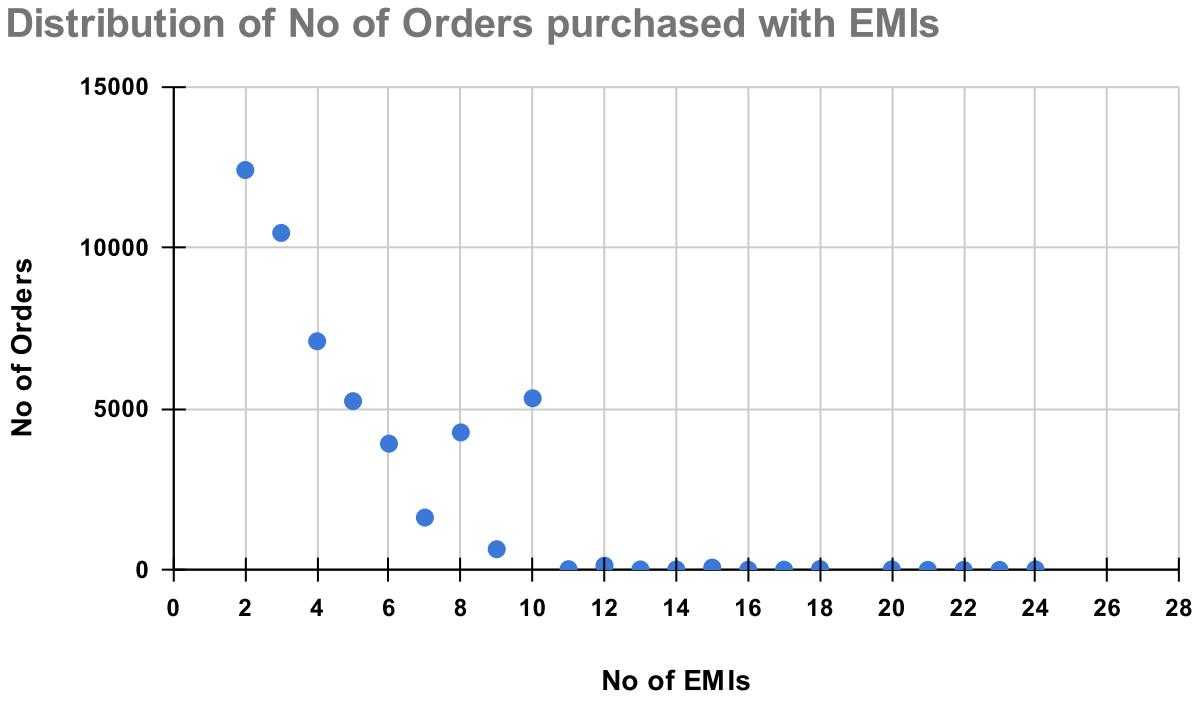
▼ Payment_installments vs. No of orders

```
SELECT
        payment_installments,
        payment_type,
        count(order_id) as No_of_orders
    FROM
        `ecommerce_dataset.payments`
    WHERE
      payment_installments>=2
-- To filter out the EMIs order from all the orders
    GROUP by
      payment_installments,payment_type
    ORDER BY
      no_of_orders desc
```

- Output: -

payment_installments	payment_type	No_of_orders
2	credit_card	12413
3	credit_card	10461
4	credit_card	7098
10	credit_card	5328
5	credit_card	5239
8	credit_card	4268
6	credit_card	3920
7	credit_card	1626
9	credit_card	644
12	credit_card	133
15	credit_card	74
18	credit_card	27

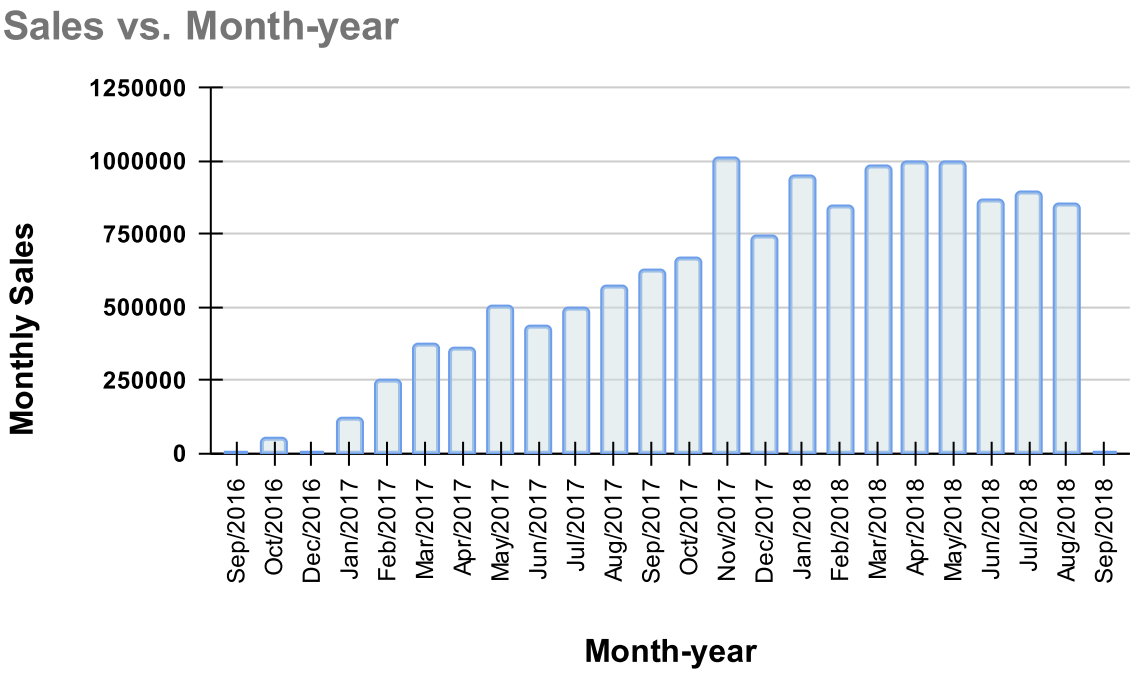
- Distribution of Number of orders purchased with different EMI plans.



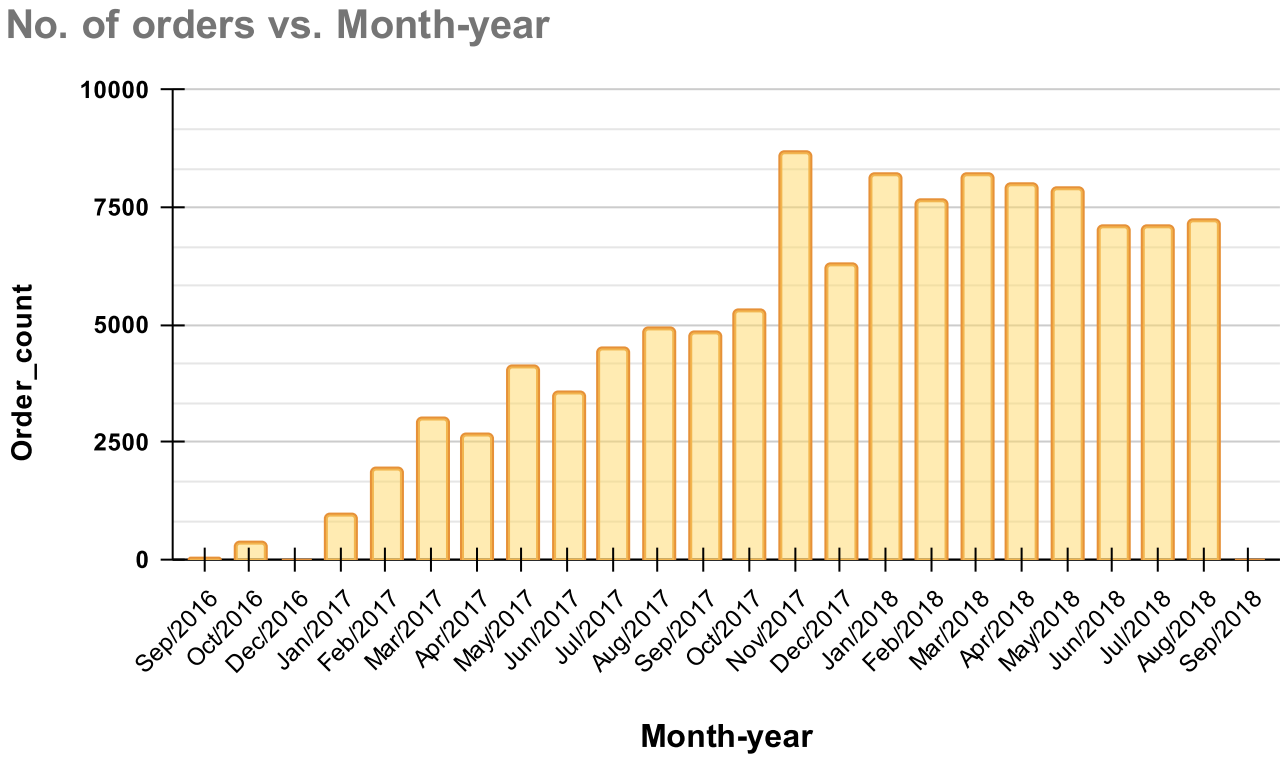
▼ Insights: -

▼ Order trend analysis

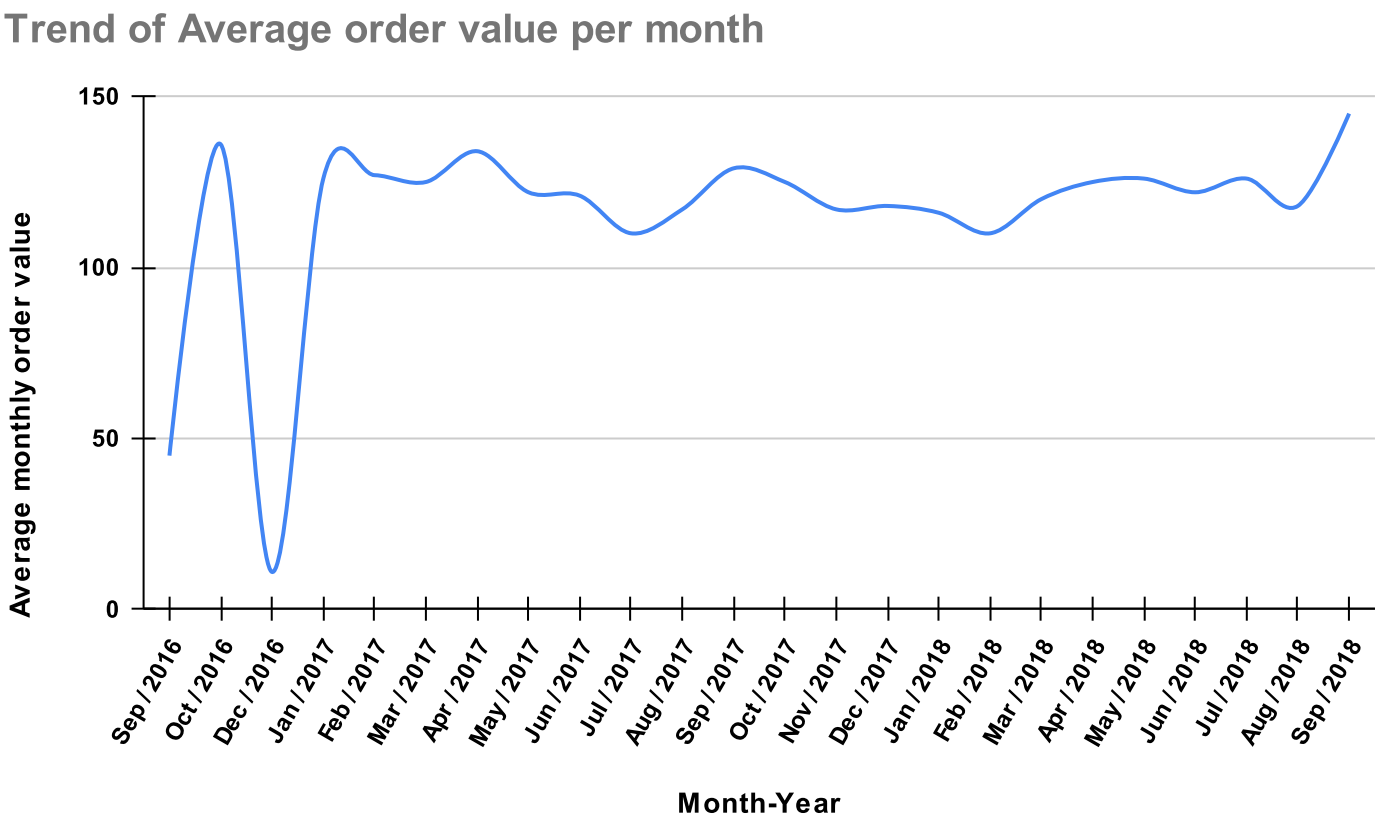
- Monthly sales over the entire duration of 2.12 Years



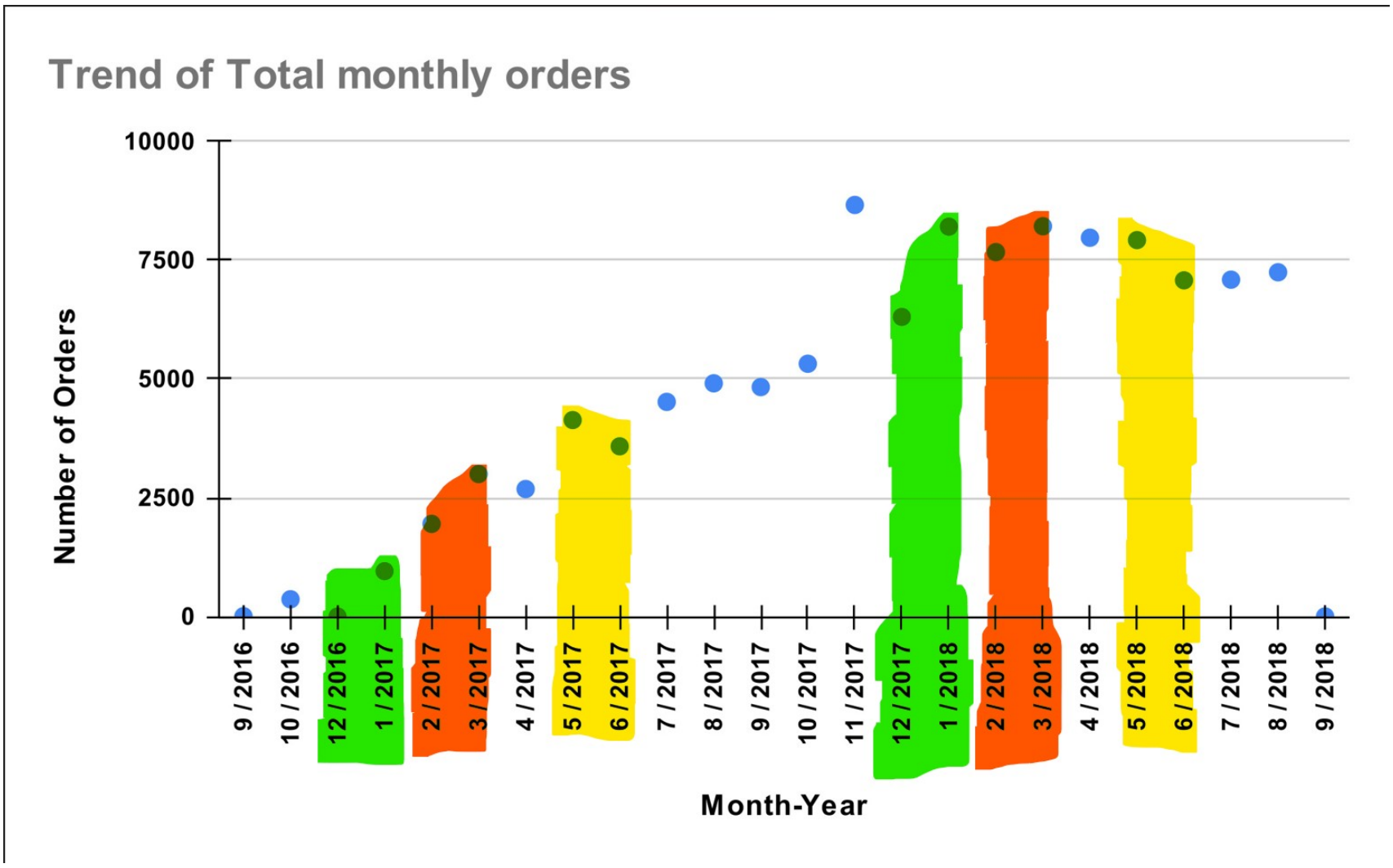
- Total number of orders per month over the entire duration of 2.12 Years



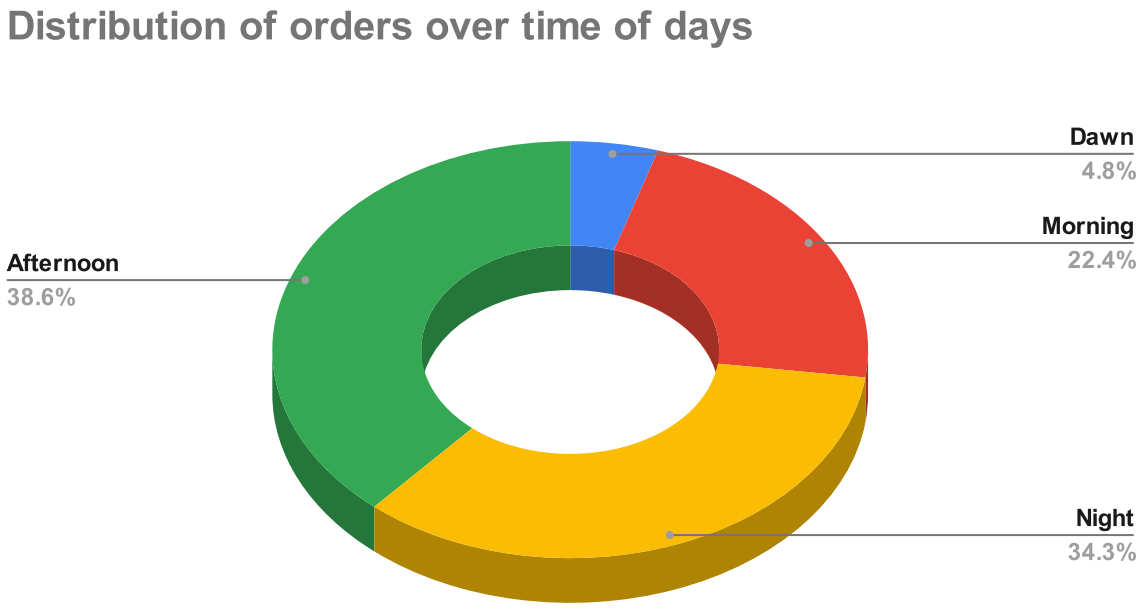
- Trend of Average order value per month



- Seasonality with peaks at specific months



- Distribution of number of orders within different time period of a day.



▼ Summary

- Monthly order value and order numbers are stagnating and no growth has been shown since the peak of November 2017.
- Between December and January, February and March: There is a seasonality and orders tend to increase during these period blocks.
- Similarly, there is a trend of order number decline observed from May to June.
- December-January is the month when order number increases sharply and this may be due to Christmas and New Year’s Eve.
- Customers tend to buy more during the Afternoon (12:00 PM to 6:00 PM) and at night (6:00 PM to 0:00 AM). Most orders are from the afternoon time period.

▼ Location (state & city) based analysis

- Heighest and lowest no of orders from different states.

State	Years 2016	2017	2018
SP	5	70	88
RJ	4	46	47
MG	4	35	38
PR	3	28	31
RS	3	26	28
BA	1	23	22
SC	2	22	25
ES	1	22	24
PE	2	19	19
GO	2	18	16
PA	1	17	14
DF	1	14	7
MS		13	16
CE	2	15	14
RN	2	13	13
PI	1	12	11
MA	1	12	10
AL	1	12	11
PB	1	13	11
MT	2	11	15
SE	1	11	7
AM		7	7
TO		7	9
RO		7	8
AC		6	5
AP		4	5
RR	1	3	4

- Top 10 cities with highest number of orders

City	State	Years 2016	2017	2018
sorocaba	SP	1	11	9
sao paulo	SP	1	12	9
rio de janeiro	RJ	1	11	9
belo horizonte	MG	1	11	9
porto alegre	RS	1	11	8
niteroi	RJ	1	12	8
sao jose dos campos	SP	2	11	7
sao bernardo do ca..	SP	1	12	7
curitiba	PR	2	11	7
brasilia	DF	1	12	7

- Top cities with most registered customers.

Row	customer_city	Total_customer
1	sao paulo	15540
2	rio de janeiro	6882
3	belo horizonte	2773
4	brasilia	2131
5	curitiba	1521
6	campinas	1444
7	porto alegre	1379
8	salvador	1245
9	guarulhos	1189

- Top states where most customers are registered.

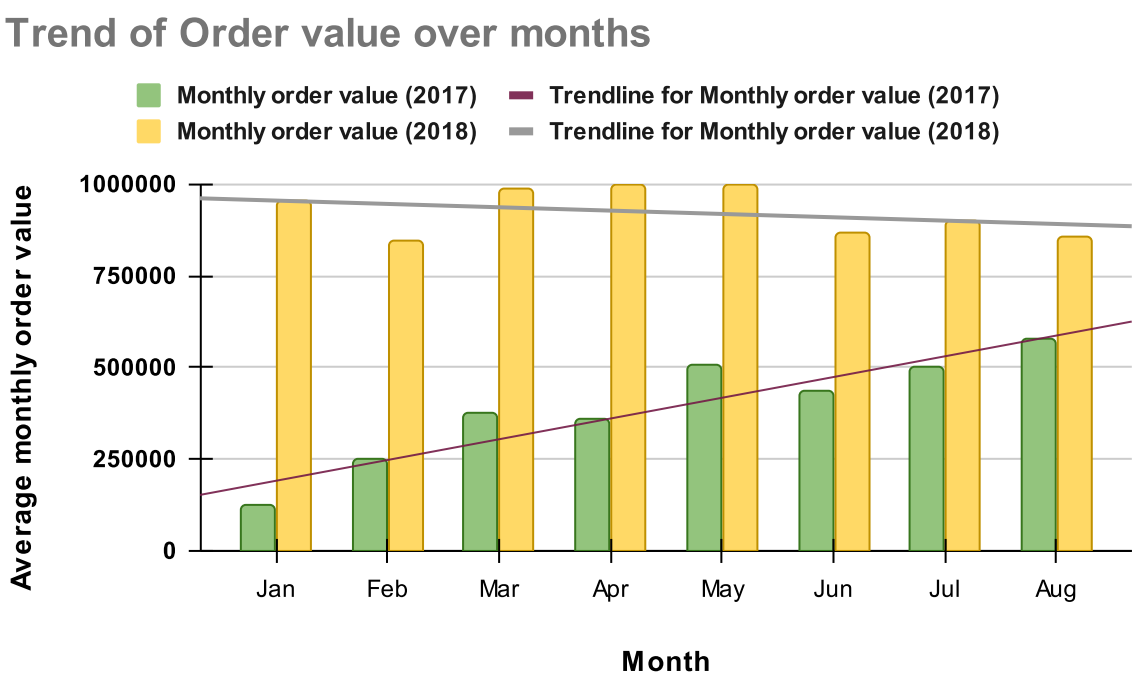
customer_state	Total_custo...
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033

▼ Summary

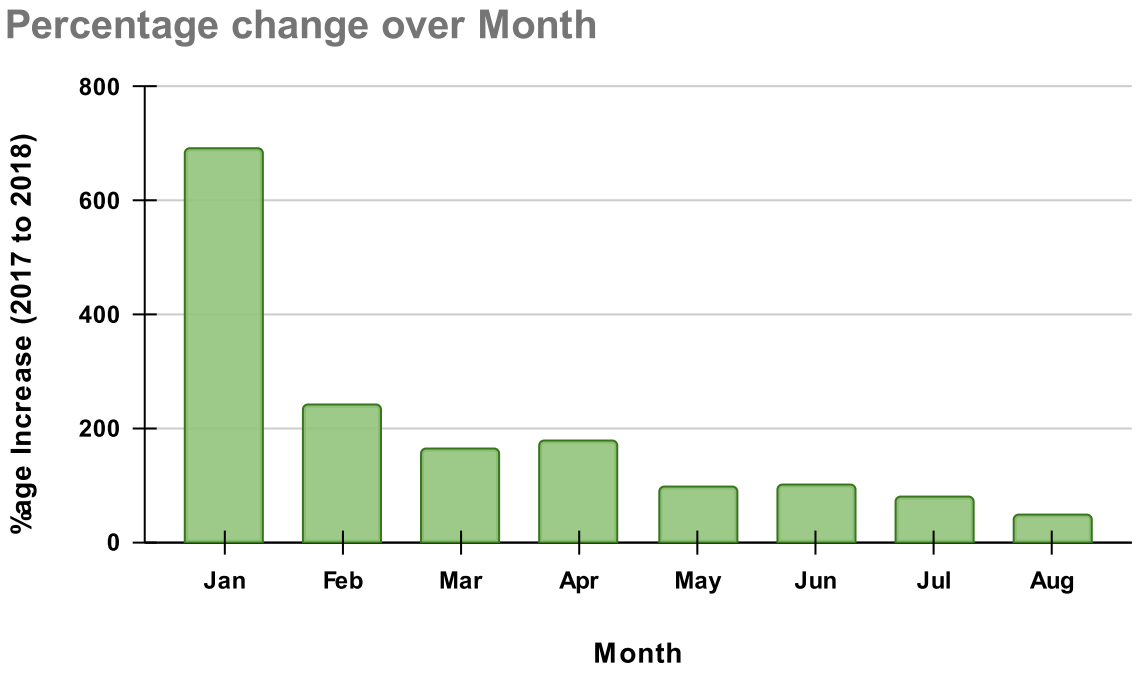
- **SP** is the state with the highest number of orders every year. Similarly, **RR** is the state with the lowest number of orders each year.
- Sao Paulo and Rio de Janeiro are the major cities where customers are present.
- SP, RJ, and MG are the top cities from where customers are registered.

▼ Yearly analysis

- Trend of order value over Months of 3017-2018



- Percentage change over Months of 2017-2018



▼ Summary: -

- The rate at which the order value (month-on-month) is changing is decreasing.
- January month had the biggest jump in terms of order value (month-on-month)
- Since then the percentage change in order value month on month has been decreasing.
- **This all means there is little to no growth in 2018.**

▼ Delivery time and freight value analysis

▼ Top 5 Highest freight value state: RR → PB→ RO → AC → PI

state	Avg_shipping_cost
RR	43.09
PB	43.09
RO	41.33
AC	40.05
PI	39.12

▼ Top 5 Lowest freight value state: SP → PR → MG → RJ → DF

state	Avg_shipping_cost
SP	15.12
PR	20.47
MG	20.63
RJ	20.91
DF	21.07

▼ Top 5 states with the highest average delivery time (Slowest Delivery): AP→RR→ AM→ AL→ PA

state	Avg_actual_delivery_time
AP	28.22
RR	28.17
AM	26.34
AL	24.45
PA	23.7

▼ Top 5 states with the lowest average delivery time (Fastest Delivery): SP→ PR→ MG→ DF → SG

state	Avg_actual_delivery_time
SP	8.66
PR	11.89
MG	11.92
DF	12.89
SC	14.95

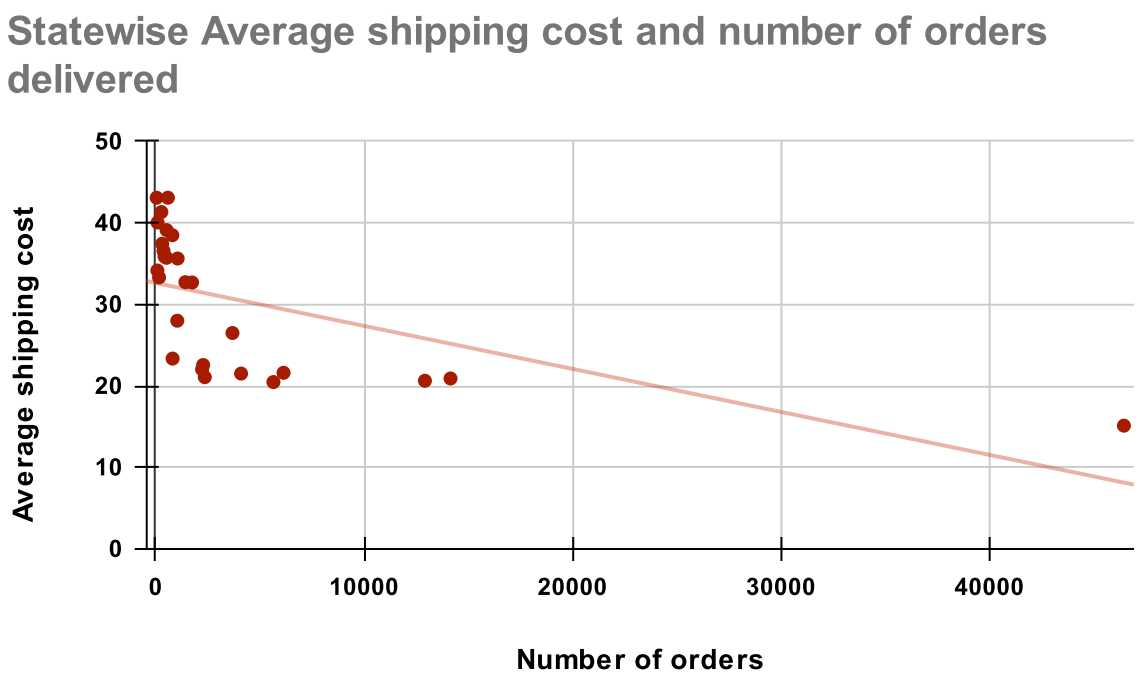
▼ Top 5 states where delivery really FAST compared to estimated delivery time: AC→RO→AM→AP→RR

state	Avg_estimated_delivery_time	Avg_actual_delivery_time	Avg_Difference
AC	41.66	20.68	-20.98
RO	39.7	19.66	-20.04
AM	46.27	26.34	-19.93
AP	46.62	28.22	-18.4
RR	46.5	28.17	-18.33

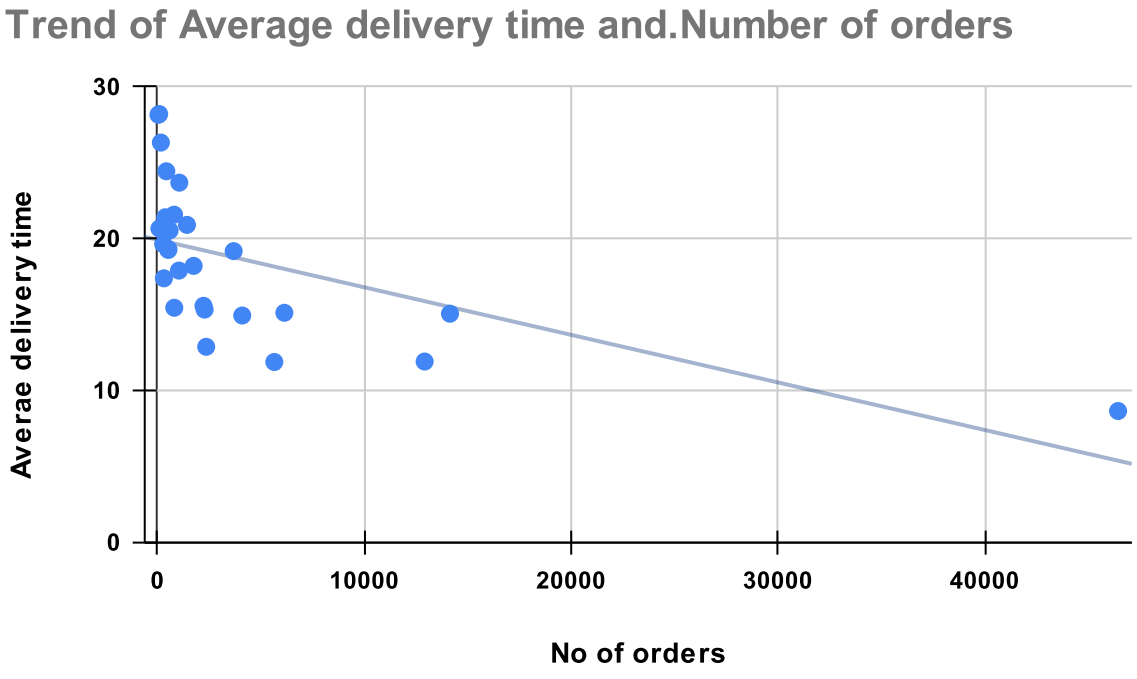
▼ Top 5 states where delivery is really slow compared to estimated delivery time: AL→MA→SE→ES→BA

state	Avg_estimated_delivery_time	Avg_actual_delivery_time	Avg_Difference
AL	33.18	24.45	-8.74
MA	31.5	21.59	-9.91
SE	31.42	21.42	-10.0
ES	26.23	15.59	-10.65
BA	30.18	19.19	-10.98

- Trend of Average shipping cost with the number of orders delivered state-wise



- Trend of Average delivery time with the number of orders delivered state-wise

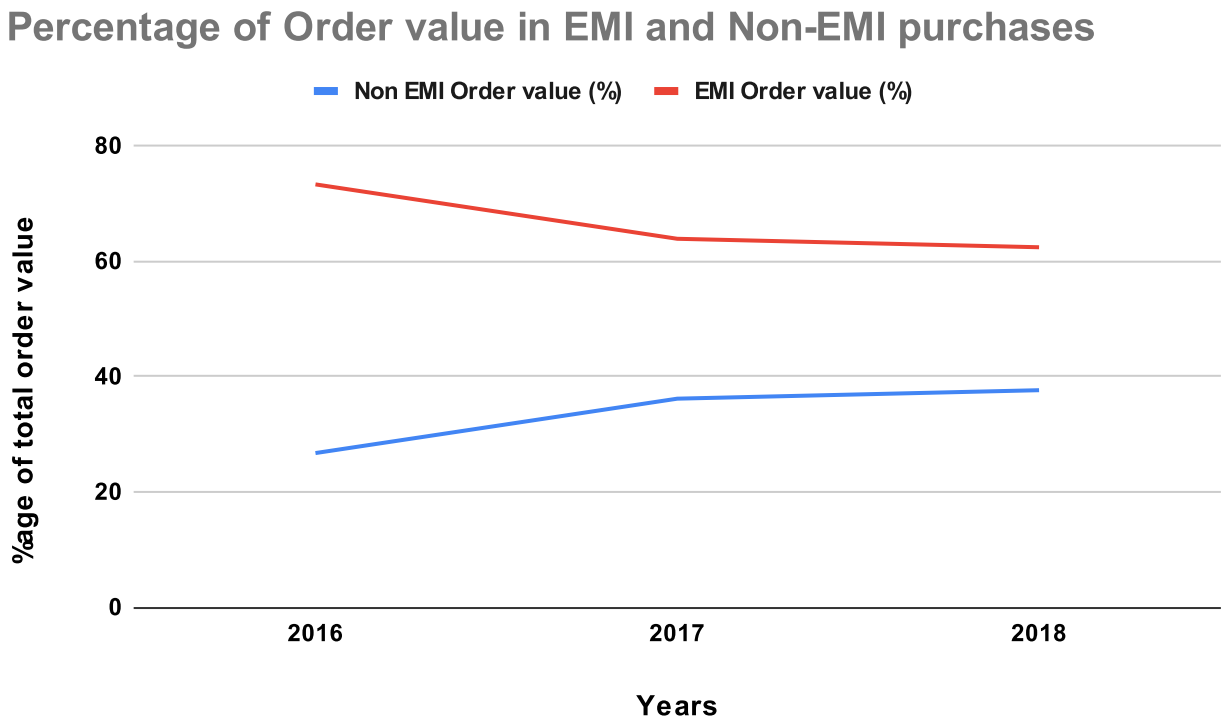


▼ Summary: -

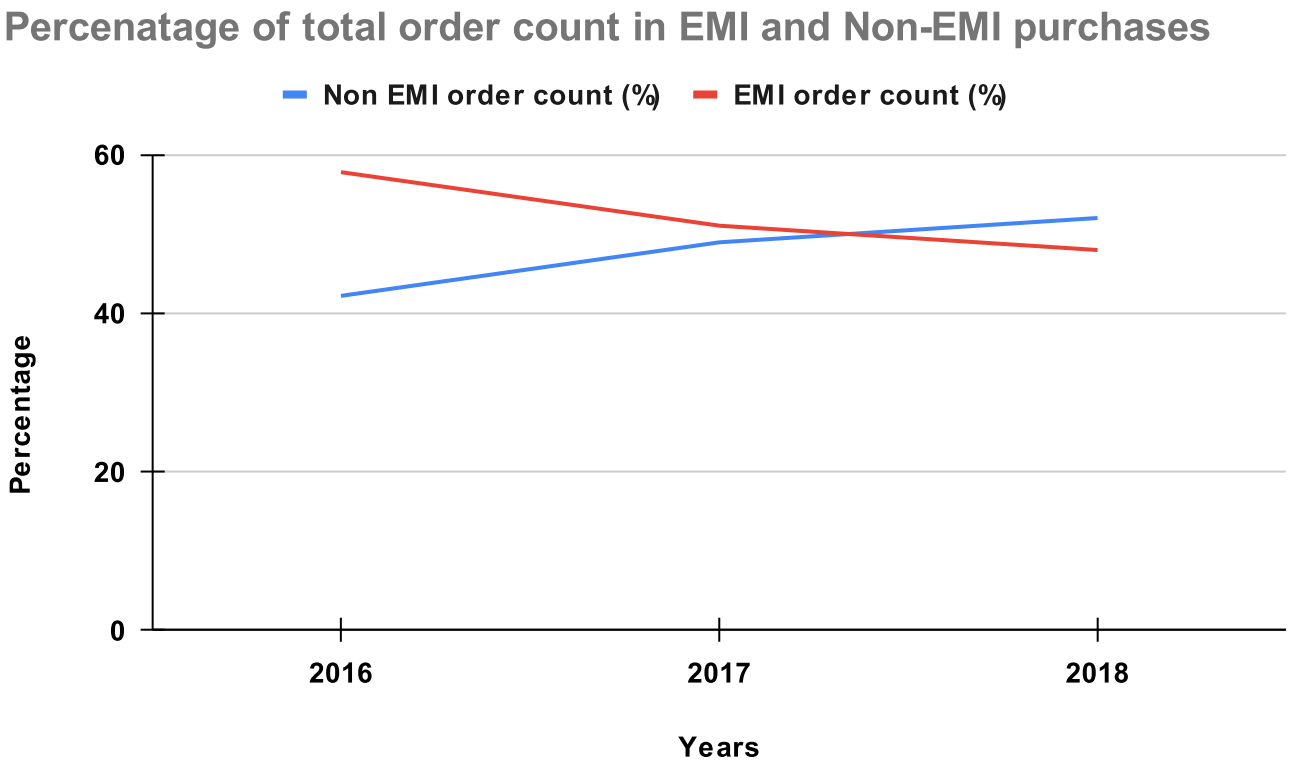
- Delivery time and Shipping cost are correlated to the number of orders received.
- As the delivery time and shipping cost increases the number of orders from that particular area (state) decreases.
- It means, there is an inversly propotional relation of number of orders delivered and Delivery time & Shipping cost.
- **It means customers prefers Low delivery cost and faster delivery time.**

▼ Payment analysis:

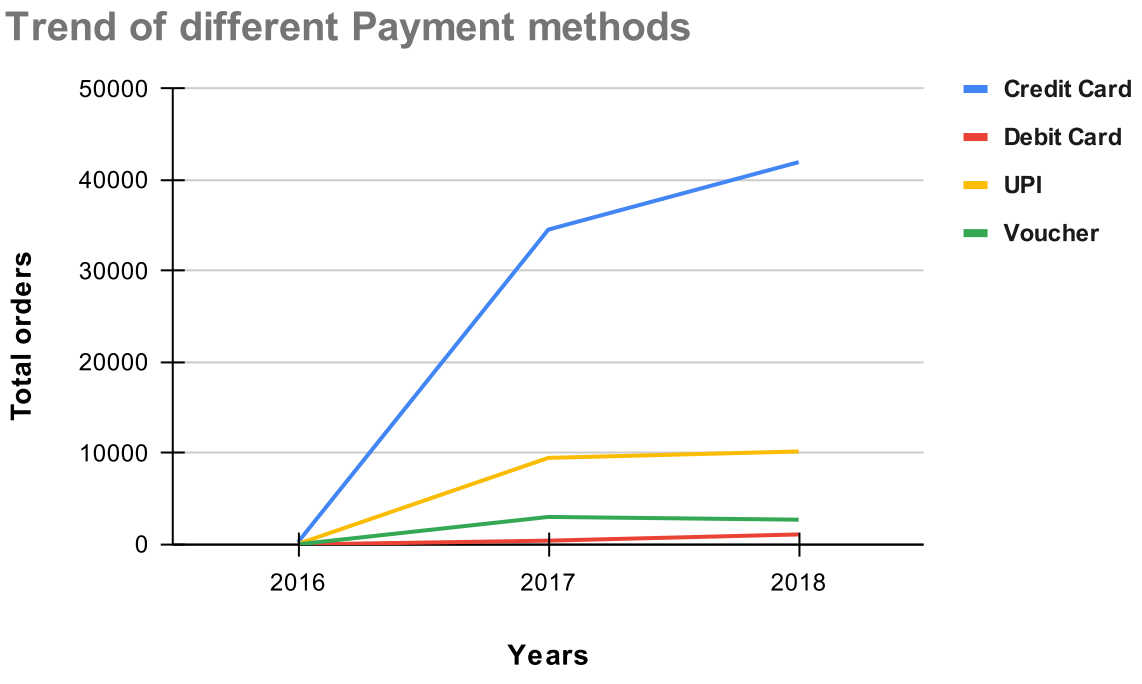
- **Percentage of order value in EMI and Non-EMI purchases**



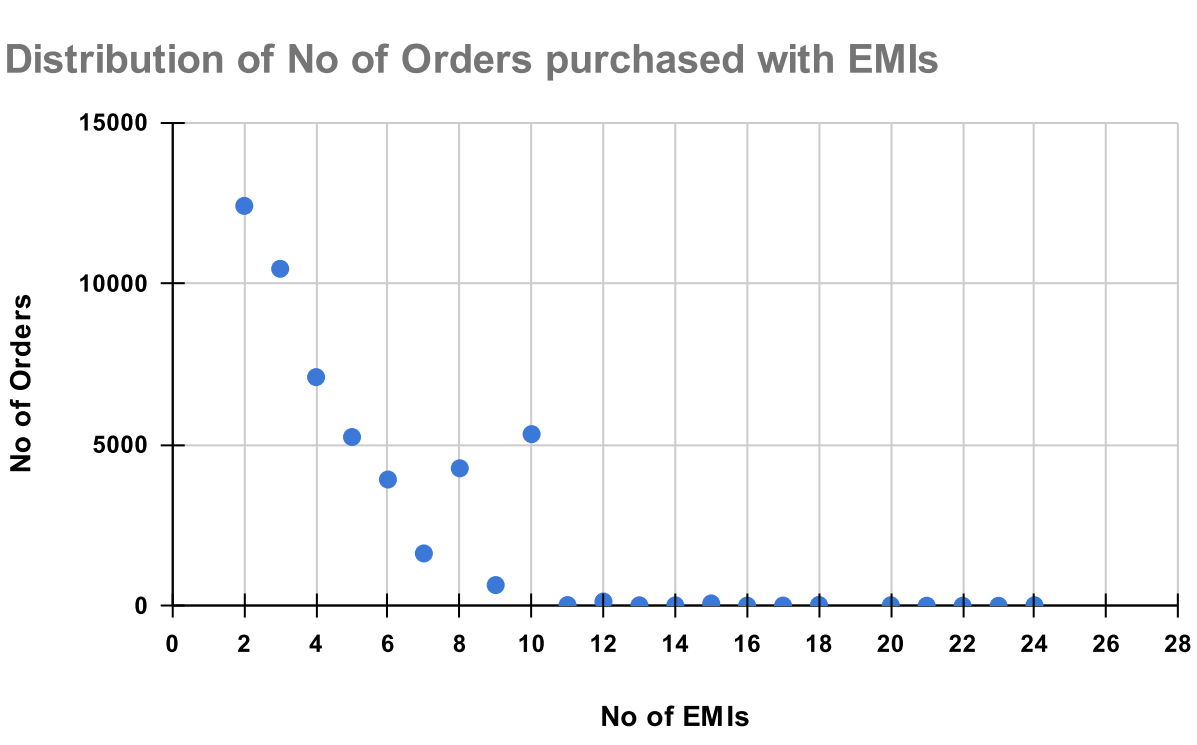
- **Percentage of total order count in EMI and Non-EMI purchases**



- **Trend of different payment methods over the years.**



- **Distribution of number of orders purchased with different EMIs.**



▼ Summary: -

- The order value and order count of EMI orders have decreased but still, the total order value of EMI orders is greater than the non-EMI orders but the order count of EMI orders has decreased and become less than the non-order count.
- It means, people are purchasing high ticket-sized products with EMI and usually prefer no EMI for low-value products.
- From a month-on-month analysis of payment methods, we observed that most of the orders are placed with credit card payments.
- The data shows that EMIs for >2 months are done with credit cards only.
- **Customers prefer 2-10 months EMIs the most.**

▼ Recommendation(s): -

- We can start a trial for premium membership in which customer will get free delivery and then further analyse the data based on the trends that we would receive. (This might resolve the issue with states with low order count).
- Delivery time can be improved by adding more sellers in the states where sellers are not present (Further analysis to be done).
- We can provide some offers like No cost EMIs (by parterning with Banks and product companies) to customers for high ticket-sized products on different credit card EMIs plan between a tenure of 2 - 10 months during afternoon and night time as the user activity increases during that time.