

## Context

Target is one of the world's most recognized brands and one of America's leading retailers. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver.

This business case has information of 100k orders from 2016 to 2018 made at Target in Brazil. Its features allows viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers.

Dataset: <https://drive.google.com/drive/folders/1TGEc66YKbD443nslRi1bWgVd238gJCnb>

Tools Used: Mysql and Tableau

Data is available in 8 csv files:

1. customers.csv
2. geolocation.csv
3. order\_items.csv
4. payments.csv
5. reviews.csv
6. orders.csv
7. products.csv
8. sellers.csv

Each feature or columns of different CSV files are described below:

The customers.csv contain following features:

Features	Description
customer_id	Id of the consumer who made the purchase.
customer_unique_id	Unique Id of the consumer.
customer_zip_code_prefix	Zip Code of the location of the consumer.

customer_city	Name of the City from where order is made.
customer_state	State Code from where order is made(Ex- sao paulo-SP).

The sellers.csv contains following features:

Features	Description
seller_id	Unique Id of the seller registered
seller_zip_code_prefix	Zip Code of the location of the seller.
seller_city	Name of the City of the seller.
seller_state	State Code (Ex- sao paulo-SP)

The order\_items.csv contain following features:

Features	Description
order_id	A unique id of order made by the consumers.
order_item_id	A Unique id given to each item ordered in the order.
product_id	A unique id given to each product available on the site.
seller_id	Unique Id of the seller registered in Target.
shipping_limit_date	The date before which shipping of the ordered product must be completed.
price	Actual price of the products ordered .
freight_value	Price rate at which a product is delivered from one point to another.

The geolocations.csv contain following features:

Features	Description
geolocation_zip_code_prefix	first 5 digits of zip code
geolocation_lat	latitude
geolocation_lng	longitude
geolocation_city	city name
geolocation_state	state

The payments.csv contain following features:

Features	Description
order_id	A unique id of order made by the consumers.
payment_sequential	sequences of the payments made in case of EMI.
payment_type	mode of payment used.(Ex-Credit Card)
payment_installments	number of installments in case of EMI purchase.
payment_value	Total amount paid for the purchase order.

The orders.csv contain following features:

Features	Description
order_id	A unique id of order made by the consumers.
customer_id	Id of the consumer who made the purchase.

order_status	status of the order made i.e delivered, shipped etc.
order_purchase_timestamp	Timestamp of the purchase.
order_delivered_carrier_date	delivery date at which carrier made the delivery.
order_delivered_customer_date	date at which customer got the product.
order_estimated_delivery_date	estimated delivery date of the products.

The reviews.csv contain following features:

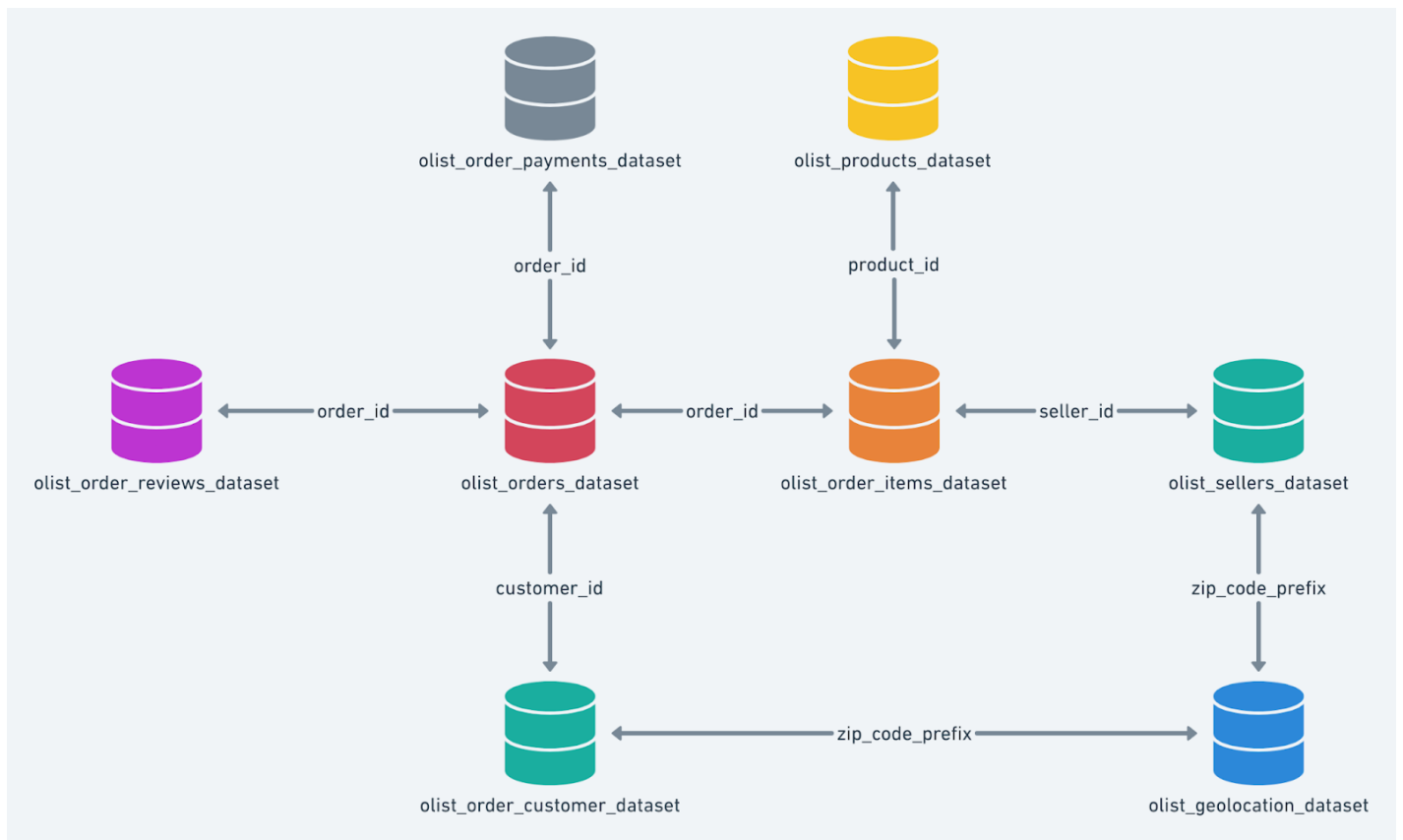
Features	Description
review_id	Id of the review given on the product ordered by the order id.
order_id	A unique id of order made by the consumers.
review_score	review score given by the customer for each order on the scale of 1–5.
review_comment_title	Title of the review
review_comment_message	Review comments posted by the consumer for each order.
review_creation_date	Timestamp of the review when it is created.
review_answer_timestamp	Timestamp of the review answered.

The products.csv contain following features:

Features	Description
product_id	A unique identifier for the proposed project.

product_category_name	Name of the product category
product_name_lenght	length of the string which specifies the name given to the products ordered.
product_description_lenght	length of the description written for each product ordered on the site.
product_photos_qty	Number of photos of each product ordered available on the shopping portal.
product_weight_g	Weight of the products ordered in grams.
product_length_cm	Length of the products ordered in centimeters.
product_height_cm	Height of the products ordered in centimeters.
product_width_cm	width of the product ordered in centimeters.

High level overview of relationship between datasets:



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

1. Data type of columns in a table

Result:


1: EXPLAIN target\_db.customers;

Result Grid							
		Filter Rows:		Search		Export:	
	Field	Type	Null	Key	Default	Extra	
▶	customer_id	text	YES		NULL		
▶	customer_unique_id	text	YES		NULL		
▶	customer_zip_code_prefix	text	YES		NULL		
▶	customer_city	text	YES		NULL		
▶	customer_state	text	YES		NULL		

2: EXPLAIN target\_db.order\_items;

Result Grid							Filter Rows:	Q Search	Export
	Field	Type	Null	Key	Default	Extra			
▶	order_id	text	YES		NULL				
▢	order_item_id	int	YES		NULL				
▢	product_id	text	YES		NULL				
▢	seller_id	text	YES		NULL				
▢	shipping_limit_date	text	YES		NULL				
▢	price	double	YES		NULL				
▢	freight_value	double	YES		NULL				

3: EXPLAIN target\_db.orders;

Result Grid							Filter Rows:	Q Search	Export: 
	Field	Type	Null	Key	Default	Extra			
▶	order_id	text	YES		NULL				
▢	customer_id	text	YES		NULL				
▢	order_status	text	YES		NULL				
▢	order_purchase_timestamp	text	YES		NULL				
▢	order_approved_at	text	YES		NULL				
▢	order_delivered_carrier_date	text	YES		NULL				
▢	order_delivered_customer_date	text	YES		NULL				
▢	order_estimated_delivery_date	text	YES		NULL				

4: EXPLAIN target\_db.payments;

Result Grid							Filter Rows:	Q Search	Export:
	Field	Type	Null	Key	Default	Extra			
▶	order_id	text	YES		NULL				
▢	payment_sequential	int	YES		NULL				
▢	payment_type	text	YES		NULL				
▢	payment_installments	int	YES		NULL				
▢	payment_value	double	YES		NULL				

5: EXPLAIN target\_db.products;

Result Grid						
		Filter Rows:		Search		Export:
	Field	Type	Null	Key	Default	Extra
▶	product_id	text	YES		NULL	
▶	product category	text	YES		NULL	
▶	product_name_length	int	YES		NULL	
▶	product_description_length	int	YES		NULL	
▶	product_photos_qty	int	YES		NULL	
▶	product_weight_g	int	YES		NULL	
▶	product_length_cm	int	YES		NULL	
▶	product_height_cm	int	YES		NULL	
▶	product_width_cm	int	YES		NULL	

6: EXPLAIN target\_db.sellers;

Result Grid						
		Filter Rows:		Search		Export:
	Field	Type	Null	Key	Default	Extra
▶	seller_id	text	YES		NULL	
▶	seller_zip_code_prefix	text	YES		NULL	
▶	seller_city	text	YES		NULL	
▶	seller_state	text	YES		NULL	

2. Time period for which the data is given:

Result:

```
SELECT
    MAX(order_purchase_timestamp) AS Latest_date,
    MIN(order_purchase_timestamp) AS First_date
FROM
    orders;
```

	Latest_date	First_date
▶	2018-10-17 17:30:18	2016-09-04 21:15:19



Note: We can see that the time period of the data is from 9th Sept 2016 to 17th October 2018.

### 3. Cities and States covered in the dataset

```
select distinct(customer_city) from customers;
```

	customer_city	
▶	franca	
●	sao bernardo do campo	
	sao paulo	
●	mogi das cruces	
	campinas	
●	jaragua do sul	
	timoteo	
●	curitiba	
	belo horizonte	
●	montes claros	
	rio de janeiro	
●	lencois paulista	
	caxias do sul	
●	piracicaba	
	guarulhos	

```
select distinct(customer_state) from customers;
```

	customer_state	
▶	SP	
	SC	
	MG	
	PR	
	RJ	
	RS	
	PA	
	GO	
	ES	
	BA	
	MA	
	MS	
	CE	
	DF	
	RN	
	PE	

---

## 2. In-depth Exploration:

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

Result: : By running the following query and observing the data we can conclude that there is a growing trend in orders. Initially the trend was very low then the number of orders increased and fell drastically.

We can notice that data is **Left skewed**. Which means it has negative skewness.

We can observe the orders were high in the month of Oct, Nov, Dec, Jan feb and march. Which seems to be the Winter season. So We can see that people made purchases because of the Christmas season and chose to get delivery at home during winters.

```
SELECT  
  
    YEAR(order_purchase_timestamp) AS year,  
  
    MONTH(order_purchase_timestamp) AS month,  
  
    COUNT(order_id) AS month_wise_orders  
  
FROM  
  
    orders  
  
GROUP BY year , month  
  
ORDER BY year , month;
```

	year	month	COUNT(order_id)
▶	2016	9	4
	2016	10	324
	2016	12	1
	2017	1	800
	2017	2	1780
	2017	3	2682
	2017	4	2404
	2017	5	3700
	2017	6	3245
	2017	7	4026
	2017	8	4331
	2017	9	4285
	2017	10	4631
	2017	11	7544
	2017	12	5673
	2018	1	7269

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

Note: We can observe that most of the orders are placed during Afternoon.

SELECT

time\_stamp, COUNT(time\_stamp) AS no\_of\_orders

FROM

(SELECT

HOUR(order\_purchase\_timestamp) AS buy\_duration,

CASE

WHEN HOUR(order\_purchase\_timestamp) BETWEEN 8 AND 12 THEN 'Morning'

WHEN HOUR(order\_purchase\_timestamp) BETWEEN 13 AND 17 THEN 'Afternoon'

WHEN HOUR(order\_purchase\_timestamp) BETWEEN 18 AND 21 THEN 'Evening'

WHEN HOUR(order\_purchase\_timestamp) BETWEEN 22 AND 4 THEN 'Night'

```

        ELSE 'Dawn'

    END AS time_stamp

FROM

    customers c

    JOIN orders o ON o.customer_id = c.customer_id) x

GROUP BY time_stamp

```

	time_stamp	no_of_orders	
►	Afternoon	32366	
	Morning	26502	
	Evening	24161	
	Dawn	16412	

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

1. Get month on month orders by region, states

```

SELECT

    YEAR(o.order_purchase_timestamp) AS year,

    MONTH(o.order_purchase_timestamp) AS month,

    c.customer_state,

    COUNT(o.order_id) AS Statewise_orders_in_month

FROM

    customers c



```

LEFT JOIN

orders o ON c.customer\_id = o.customer\_id

GROUP BY year , month,c.customer\_state

ORDER BY year , month,c.customer\_state;

Result Grid   Filter Rows: <input type="text" value="Search"/> Export:				
	year	month	customer_state	Statewise_orders_in_month
▶	2016	9	RR	1
▶	2016	9	RS	1
▶	2016	9	SP	2
▶	2016	10	AL	2
▶	2016	10	BA	4
▶	2016	10	CE	8
▶	2016	10	DF	6
▶	2016	10	ES	4
▶	2016	10	GO	9
▶	2016	10	MA	4
▶	2016	10	MG	40
▶	2016	10	MT	3
▶	2016	10	PA	4
▶	2016	10	PB	1
▶	2016	10	PE	7
▶	2016	10	PI	1

## 2. How are customers distributed in Brazil

SELECT

customer\_state,

customer\_city,

COUNT(customer\_id) AS cust\_count

FROM

customers

GROUP BY customer\_state , customer\_city

ORDER BY cust\_count DESC;

	customer_state	customer_city	cust_count	
	MG	belo horizonte	2773	
	DF	brasilia	2131	
	PR	curitiba	1521	
	SP	campinas	1444	
	RS	porto alegre	1379	
	BA	salvador	1245	
	SP	guarulhos	1189	
	SP	sao bernardo do campo	938	
	RJ	niteroi	849	
	SP	santo andre	796	
	SP	osasco	746	
	SP	santos	713	
	GO	goiania	692	
	SP	sao joao do campo	661	

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

We noticed that the payment value is the total value of order. If an order is bought on instalment payment value shows full amount instead of partial payments.

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

with temp as (

SELECT

distinct(orders.order\_id),

orders.order\_purchase\_timestamp,

order\_items.price,

order\_items.freight\_value,

```

round(order_items.price + order_items.freight_value,2) as net_price

FROM

orders

INNER JOIN

order_items ON orders.order_id = order_items.order_id

)

SELECT

ROUND((MAX(a.total_price) - MIN(a.total_price)) / MIN(a.total_price) * 100,

2) AS percentage_increase_2017_2018

FROM

(SELECT

SUM(temp.net_price) AS total_price,

YEAR(temp.order_purchase_timestamp) AS year

FROM

temp

WHERE

YEAR(temp.order_purchase_timestamp) = 2017

AND MONTH(temp.order_purchase_timestamp) BETWEEN 1 AND 8 UNION SELECT

SUM(temp.net_price) AS total_price,

YEAR(temp.order_purchase_timestamp) AS year

FROM

temp

WHERE

```

YEAR(temp.order\_purchase\_timestamp) = 2018

AND MONTH(temp.order\_purchase\_timestamp) BETWEEN 1 AND 8) a;

percentage_increase_2017_2018
138.16

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

WITH temp AS(

SELECT

customer\_state,

order\_items.price + order\_items.freight\_value AS net\_price

FROM

order\_items

LEFT JOIN

orders ON order\_items.order\_id = orders.order\_id

LEFT JOIN

customers ON orders.customer\_id = customers.customer\_id)

SELECT

temp.customer\_state,

ROUND(AVG(temp.net\_price), 2) AS Mean\_state,

ROUND(SUM(temp.net\_price), 2) AS sum\_statewise



FROM

temp

GROUP BY temp.customer\_state;

	customer_sta...	Mean_state	sum_statewise
►	SP	124.8	5921678.12
	SC	146.12	610213.6
	RJ	146.08	2129681.98
	MG	141.38	1856161.49
	MT	176.46	186168.96
	GO	149.04	347706.93
	PR	139.54	800935.44
	AL	216.73	96229.4
	RS	142.07	885826.76
	DF	146.81	353229.44
	BA	160.97	611506.67
	CE	186.47	275606.3

## 5. Analysis on sales, freight and delivery time

### 1. Calculate days between purchasing, delivering and estimated delivery

SELECT

order\_id,

order\_purchase\_timestamp,

order\_delivered\_customer\_date,

order\_estimated\_delivery\_date,

DATEDIFF(order\_delivered\_customer\_date,

order\_purchase\_timestamp) AS purchase\_vs\_deliver,

DATEDIFF(order\_estimated\_delivery\_date,

```

        order_purchase_timestamp) AS purch_vs_estimated,

DATEDIFF(order_estimated_delivery_date,

        order_delivered_customer_date) AS estimated_vs_actual_delivery

FROM

orders;

```

order_id	order_purchase_timestamp	order_delivered_custo...	order_estimated_del...	purchase_vs_deli...	purch_vs_estimated	estimated_vs_actual_deliv...
e481f51cbdc54678b7cc491...	2017-10-02 10:56:33	2017-10-10 21:25:13	2017-10-18 00:00:00	8	16	8
53cdb2fc8bc7dce0b6741e2...	2018-07-24 20:41:37	2018-08-07 15:27:45	2018-08-13 00:00:00	14	20	6
47770eb9100c2d0c44946d9...	2018-08-08 08:38:49	2018-08-17 18:06:29	2018-09-04 00:00:00	9	27	18
949d5b44dbf5de918fe9c16f...	2017-11-18 19:28:06	2017-12-02 00:28:42	2017-12-15 00:00:00	14	27	13
ad21c59c0840e6cb83a9ceb...	2018-02-13 21:18:39	2018-02-16 18:17:02	2018-02-26 00:00:00	3	13	10
a4591c265e18cb1dcee5288...	2017-07-09 21:57:05	2017-07-26 10:57:55	2017-08-01 00:00:00	17	23	6
136cce7faa42fdb2cefd53fdc...	2017-04-11 12:22:08		2017-05-09 00:00:00	NULL	28	NULL
6514b8ad8028c9f2cc2374d...	2017-05-16 13:10:30	2017-05-26 12:55:51	2017-06-07 00:00:00	10	22	12
76c6e866289321a7c93b82b...	2017-01-23 18:29:09	2017-02-02 14:08:10	2017-03-06 00:00:00	10	42	32
e69bfb5eb8e0ed6a785585...	2017-07-29 11:55:02	2017-08-16 17:14:30	2017-08-23 00:00:00	18	25	7
e6ce16cb79ec1d90b1da908...	2017-05-16 19:41:10	2017-05-29 11:18:31	2017-06-07 00:00:00	13	22	9
34513ce0c4fab462a55830c...	2017-07-13 19:58:11	2017-07-19 14:04:48	2017-08-08 00:00:00	6	26	20
82566a660a982b15fb86c90...	2018-06-07 10:06:19	2018-06-19 12:05:52	2018-07-18 00:00:00	12	41	29

## 2. Create columns:

1.  $\text{time\_to\_delivery} = \text{order\_purchase\_timestamp} - \text{order\_delivered\_customer\_date}$
2.  $\text{diff\_estimated\_delivery} =$   
 $\text{order\_estimated\_delivery\_date} - \text{order\_delivered\_customer\_date}$

```

SELECT

        order_purchase_timestamp,

        order_delivered_customer_date,

DATEDIFF(order_delivered_customer_date,

        order_purchase_timestamp) AS time_to_delivery,

DATEDIFF(order_delivered_customer_date,

        order_estimated_delivery_date) AS diff_estimated_delivery

FROM

```

orders;

	order_purchase_timestamp	order_delivered_custo...	time_to_delivery	diff_estimated_delivery
▶	2017-10-02 10:56:33	2017-10-10 21:25:13	8	-8
	2018-07-24 20:41:37	2018-08-07 15:27:45	14	-6
	2018-08-08 08:38:49	2018-08-17 18:06:29	9	-18
	2017-11-18 19:28:06	2017-12-02 00:28:42	14	-13
	2018-02-13 21:18:39	2018-02-16 18:17:02	3	-10
	2017-07-09 21:57:05	2017-07-26 10:57:55	17	-6
	2017-04-11 12:22:08		NULL	NULL
	2017-05-16 13:10:30	2017-05-26 12:55:51	10	-12
	2017-01-23 18:29:09	2017-02-02 14:08:10	10	-32
	2017-07-29 11:55:02	2017-08-16 17:14:30	18	-7
	2017-05-16 19:41:10	2017-05-29 11:18:31	13	-9
	2017-07-13 19:58:11	2017-07-19 14:04:48	6	-20
	2018-06-07 10:06:10	2018-06-10 12:05:52	12	-20

3. Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery

WITH temp AS(

SELECT

customer\_state, order\_items.freight\_value,

DATEDIFF(order\_delivered\_customer\_date,

order\_purchase\_timestamp) AS time\_to\_delivery,

DATEDIFF(order\_estimated\_delivery\_date,

order\_delivered\_customer\_date) AS diff\_estimated\_delivery

FROM

order\_items

LEFT JOIN

orders ON order\_items.order\_id = orders.order\_id

LEFT JOIN

customers ON orders.customer\_id = customers.customer\_id)

SELECT

```

temp.customer_state, avg(temp.freight_value) as avg_frieght,
avg(temp.time_to_delivery), avg(temp.diff_estimated_delivery)
FROM
temp
group by temp.customer_state;

```

	customer_sta...	avg_frieght	avg(temp.time_to_delive...	avg(temp.diff_estimated_deliv...
►	SP	15.147275390418828	8.6623	11.2079
	RJ	20.96092393168237	15.0748	12.0148
	MT	28.16628436018961	17.9074	14.5718
	MG	20.630166806307006	11.9207	13.3426
	GO	22.766815259322748	15.3355	12.2942
	PR	20.531651567944436	11.8931	13.4861
	AL	35.843671171171145	24.4473	8.7354
	RS	21.735804330393236	15.1345	14.1342
	BA	26.36395893656196	19.1925	10.9826
	CE	32.71420162381595	20.9215	11.1038
	PE	32.91786267995568	18.2245	13.4502
	PI	39.14797047970481	19.3174	11.5277
	MS	23.374884004884013	15.4599	11.2293
	ES	22.058776595744607	15.5874	10.6463
	SC	21.470368773946475	14.9502	11.5727

#### 4. Sort the data to get the following:

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

```
WITH temp AS(
```

```
SELECT
```

```
customer_state, order_items.freight_value,
```

```
DATEDIFF(order_delivered_customer_date,
```

```
order_purchase_timestamp) AS time_to_delivery,
```

```
DATEDIFF(order_estimated_delivery_date,
```

```
order_delivered_customer_date) AS diff_estimated_delivery
```

```
FROM
```

```
order_items
```

LEFT JOIN

orders ON order\_items.order\_id = orders.order\_id

LEFT JOIN

customers ON orders.customer\_id = customers.customer\_id)

SELECT

temp.customer\_state,

AVG(temp.freight\_value),

AVG(temp.time\_to\_delivery),

AVG(temp.diff\_estimated\_delivery)

FROM

temp

GROUP BY temp.customer\_state

ORDER BY AVG(temp.freight\_value)

LIMIT 5;

Result Grid		Filter Rows:	Search	Export:
	customer_sta...	AVG(temp.freight_val...	AVG(temp.time_to_delive...	AVG(temp.diff_estimated_deliv...
▶	SP	15.147275390418828	8.6623	11.2079
▶	PR	20.531651567944436	11.8931	13.4861
▶	MG	20.630166806307006	11.9207	13.3426
▶	RJ	20.96092393168237	15.0748	12.0148
▶	DF	21.04135494596833	12.8938	12.2004

WITH temp AS(

SELECT

customer\_state, order\_items.freight\_value,

```
DATEDIFF(order_delivered_customer_date,
         order_purchase_timestamp) AS time_to_delivery,
DATEDIFF(order_estimated_delivery_date,
         order_delivered_customer_date) AS diff_estimated_delivery

FROM

order_items

LEFT JOIN
orders ON order_items.order_id = orders.order_id

LEFT JOIN
customers ON orders.customer_id = customers.customer_id)

SELECT

temp.customer_state,
AVG(temp.freight_value),
AVG(temp.time_to_delivery),
AVG(temp.diff_estimated_delivery)

FROM

temp

GROUP BY temp.customer_state

ORDER BY AVG(temp.freight_value) DESC

LIMIT 5;
```

	customer_sta...	AVG(temp.freight_val...	AVG(temp.time_to_delive...	AVG(temp.diff_estimated_deliv...	
►	RR	42.98442307692309	28.1739	18.3261	
	PB	42.72380398671099	20.5461	13.0375	
	RO	41.06971223021583	19.6557	20.0403	
	AC	40.073369565217384	20.6813	20.9780	
	PI	39.14797047970481	19.3174	11.5277	

## 2. Top 5 states with highest/lowest average time to delivery

WITH temp AS(

SELECT

customer\_state, order\_items.freight\_value,

DATEDIFF(order\_delivered\_customer\_date,

order\_purchase\_timestamp) AS time\_to\_delivery,

DATEDIFF(order\_estimated\_delivery\_date,

order\_delivered\_customer\_date) AS diff\_estimated\_delivery

FROM

order\_items

LEFT JOIN

orders ON order\_items.order\_id = orders.order\_id

LEFT JOIN

customers ON orders.customer\_id = customers.customer\_id)

SELECT

temp.customer\_state,

AVG(temp.freight\_value),

AVG(temp.time\_to\_delivery),

```

    AVG(temp.diff_estimated_delivery)

FROM

    temp

GROUP BY temp.customer_state

ORDER BY AVG(temp.time_to_delivery)

LIMIT 5;

```

	customer_sta...	AVG(temp.freight_val...	AVG(temp.time_to_delive...	AVG(temp.diff_estimated_deliv...
▶	SP	15.147275390418828	8.6623	11.2079
	PR	20.531651567944436	11.8931	13.4861
	MG	20.630166806307006	11.9207	13.3426
	DF	21.04135494596833	12.8938	12.2004
	SC	21.470368773946475	14.9502	11.5727

\*\*\*\*\*

```

WITH temp AS(

SELECT

    customer_state, order_items.freight_value,

    DATEDIFF(order_delivered_customer_date,

        order_purchase_timestamp) AS time_to_delivery,

    DATEDIFF(order_estimated_delivery_date,

        order_delivered_customer_date) AS diff_estimated_delivery

FROM

    order_items

    LEFT JOIN

    orders ON order_items.order_id = orders.order_id

    LEFT JOIN

```



customers ON orders.customer\_id = customers.customer\_id)

SELECT

temp.customer\_state,

AVG(temp.freight\_value),

AVG(temp.time\_to\_delivery),

AVG(temp.diff\_estimated\_delivery)

FROM

temp

GROUP BY temp.customer\_state

ORDER BY AVG(temp.time\_to\_delivery) DESC

LIMIT 5;

	customer_sta...	AVG(temp.freight_val...	AVG(temp.time_to_delive...	AVG(temp.diff_estimated_deliv...
►	AP	34.00609756097561	28.2222	18.3951
	RR	42.98442307692309	28.1739	18.3261
	AM	33.205393939393936	26.3374	19.9325
	AL	35.843671171171145	24.4473	8.7354
	PA	35.83268518518525	23.7021	14.2505

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

TOP 5 really fast:

WITH temp AS(

SELECT

customer\_state, order\_items.freight\_value,

DATEDIFF(order\_delivered\_customer\_date,

order\_purchase\_timestamp) AS time\_to\_delivery,

```

DATEDIFF(order_estimated_delivery_date,
          order_delivered_customer_date) AS diff_estimated_delivery
FROM
  order_items
  LEFT JOIN
    orders ON order_items.order_id = orders.order_id
  LEFT JOIN
    customers ON orders.customer_id = customers.customer_id)

SELECT
  temp.customer_state,
  AVG(temp.freight_value),
  AVG(temp.time_to_delivery),
  AVG(temp.diff_estimated_delivery)
FROM
  temp
GROUP BY temp.customer_state
ORDER BY AVG(temp.diff_estimated_delivery) DESC
LIMIT 5;

```

	customer_sta...	AVG(temp.freight_val...	AVG(temp.time_to_delive...	AVG(temp.diff_estimated_delivery)
▶	AC	40.073369565217384	20.6813	20.9780
	RO	41.06971223021583	19.6557	20.0403
	AM	33.205393939393936	26.3374	19.9325
	AP	34.00609756097561	28.2222	18.3951
	RR	42.98442307692309	28.1739	18.3261

=====

```
WITH temp AS(
SELECT
    customer_state, order_items.freight_value,
    DATEDIFF(order_delivered_customer_date,
        order_purchase_timestamp) AS time_to_delivery,
    DATEDIFF(order_estimated_delivery_date,
        order_delivered_customer_date) AS diff_estimated_delivery
FROM
    order_items
    LEFT JOIN
    orders ON order_items.order_id = orders.order_id
    LEFT JOIN
    customers ON orders.customer_id = customers.customer_id)

SELECT
    temp.customer_state,
    AVG(temp.freight_value),
    AVG(temp.time_to_delivery),
    AVG(temp.diff_estimated_delivery)
FROM
    temp
GROUP BY temp.customer_state
ORDER BY AVG(temp.diff_estimated_delivery)
LIMIT 5;
```

	customer_sta...	AVG(temp.freight_val...	AVG(temp.time_to_delive...	AVG(temp.diff_estimated_delivery)
▶	AL	35.843671171171145	24.4473	8.7354
	MA	38.25700242718453	21.5900	9.9063
	SE	36.653168831168806	21.4187	10.0027
	ES	22.058776595744607	15.5874	10.6463
	BA	26.36395893656196	19.1925	10.9826

## 6. Payment type analysis:

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

```

SELECT

    YEAR(o.order_purchase_timestamp) AS year,

    MONTH(o.order_purchase_timestamp) AS month,

    p.payment_type,

    COUNT(o.order_id) AS count_of_orders

FROM

    orders o

    INNER JOIN

        payments p ON o.order_id = p.order_id

GROUP BY year , month , p.payment_type

ORDER BY year , month;
```

	year	month	payment_type	count_of_orders
▶	2016	9	credit_card	3
	2016	10	credit_card	254
	2016	10	debit_card	2
	2016	10	UPI	63
	2016	10	voucher	23
	2016	12	credit_card	1
	2017	1	credit_card	583
	2017	1	debit_card	9
	2017	1	UPI	197
	2017	1	voucher	61
	2017	2	credit_card	1356
	2017	2	debit_card	13
	2017	2	UPI	398
	2017	2	voucher	119
	2017	3	credit_card	2016

## 2. Distribution of payment instalments and count of orders

SELECT

payment\_installments, COUNT(order\_id) AS count\_of\_orders

FROM

payments

GROUP BY payment\_installments

ORDER BY count\_of\_orders DESC;

	payment_installments	count_of_orders
▶	1	52546
	2	12413
	3	10461
	4	7098
	10	5328
	5	5239
	8	4268
	6	3920
	7	1626
	9	644
	12	133
	15	74
	18	27
	11	23
	24	18

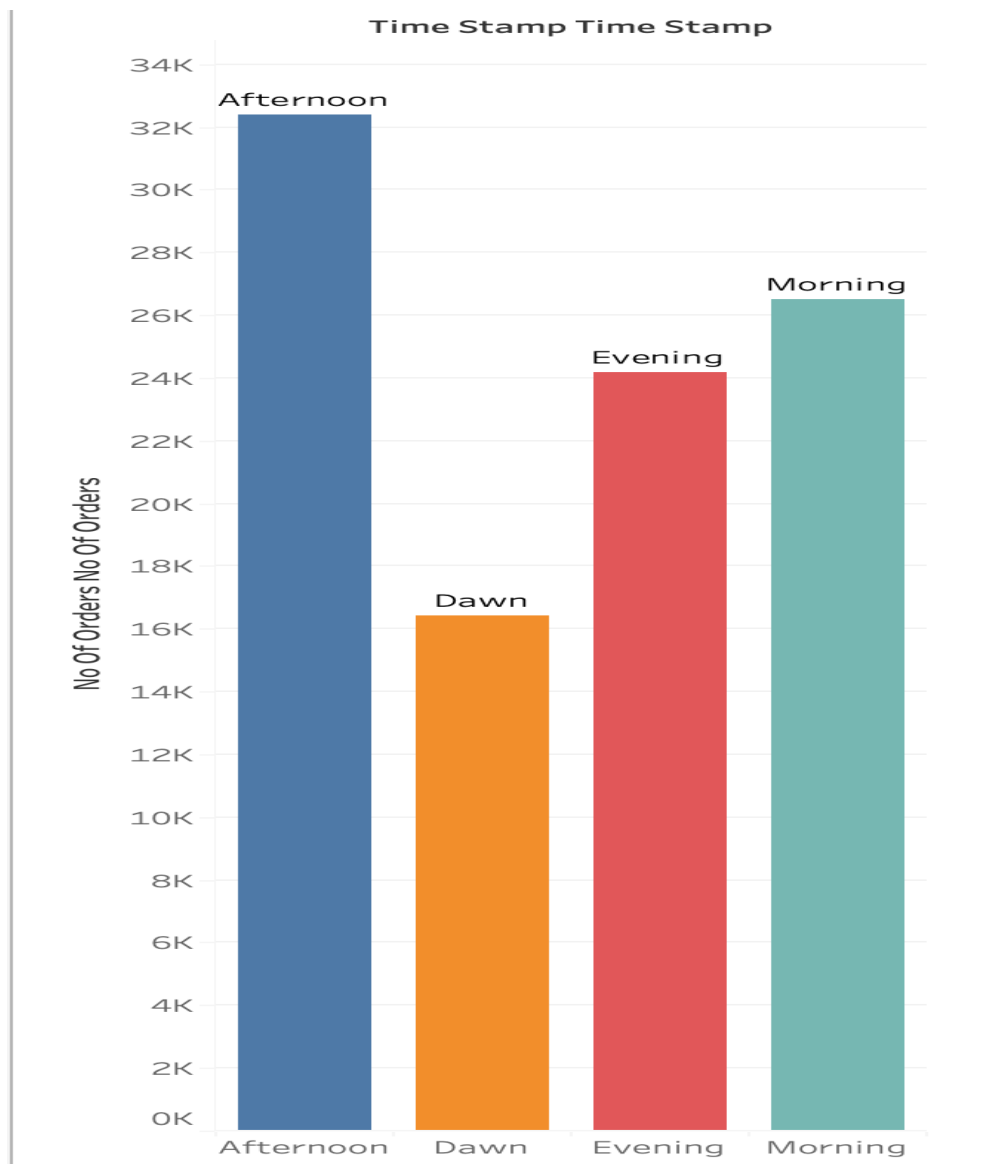
---

## 7. Analysis:

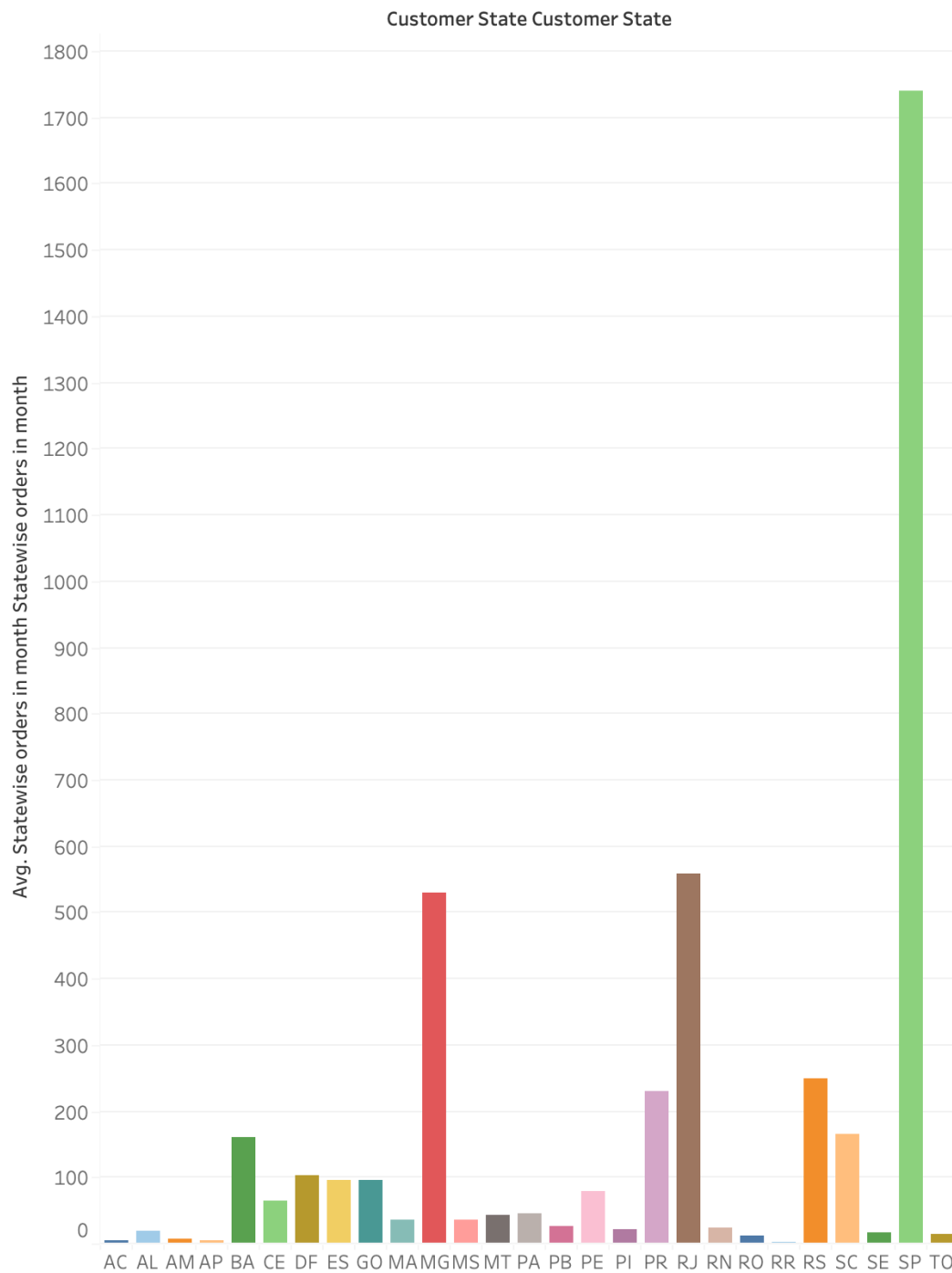
- We can see that the time period of the data is from 9th Sept 2016 to 17th October 2018.
- We have noticed that most of the orders are placed in the afternoon.
- We observed the data we can conclude that there is a growing trend in orders. Initially the trend was very low then the number of orders increased and fell drastically.

We can notice that data is **Left skewed**. Which means it has negative skewness. We can observe the orders were high in the month of Oct, Nov, Dec, Jan feb and march. Which seems to be the Winter season. So We can see that people made purchases because of the Christmas season and chose to get delivery at home during winters.

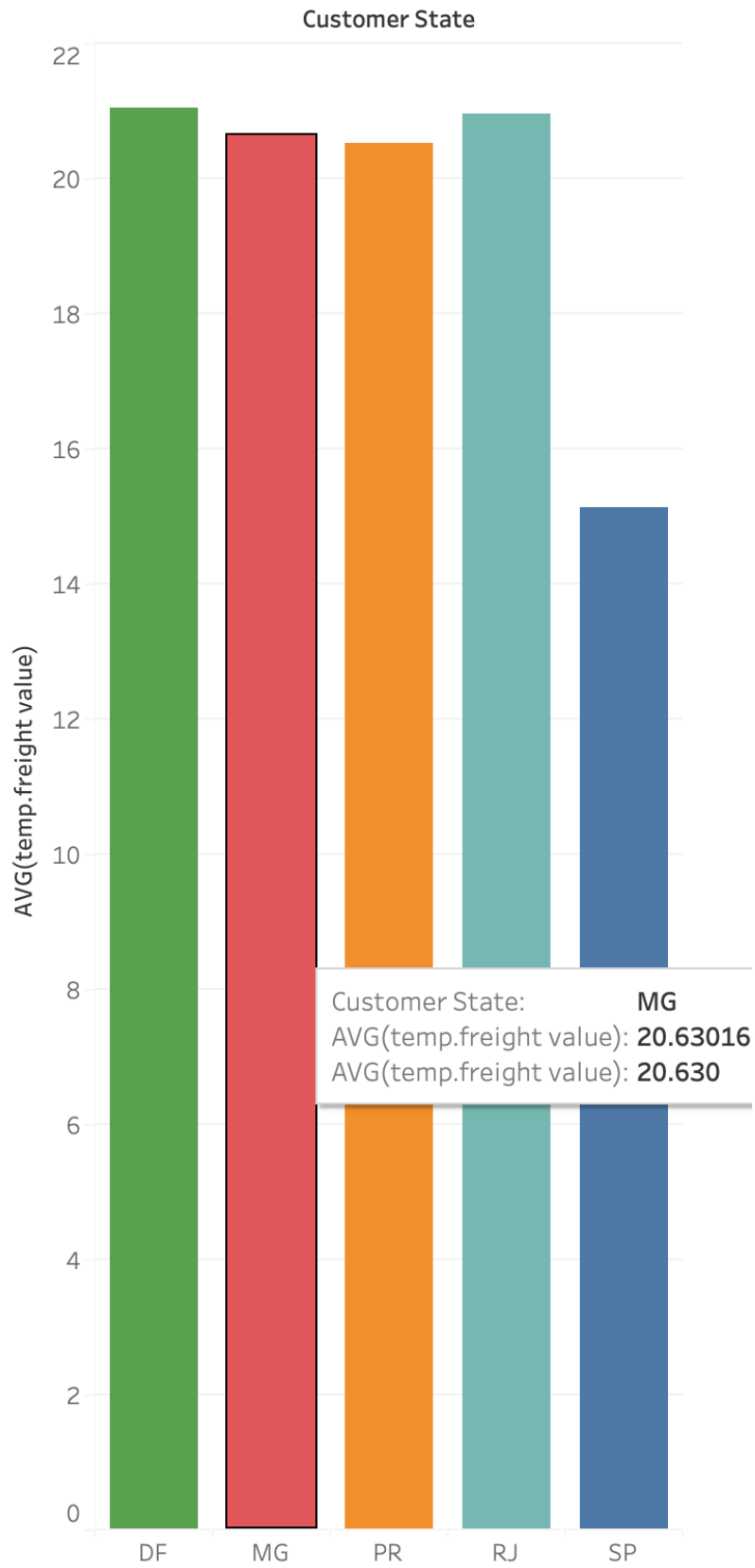
- We can see that maximum number of order are made in Afternoon.



- Maximum orders are placed in SP (Sou Paulo)

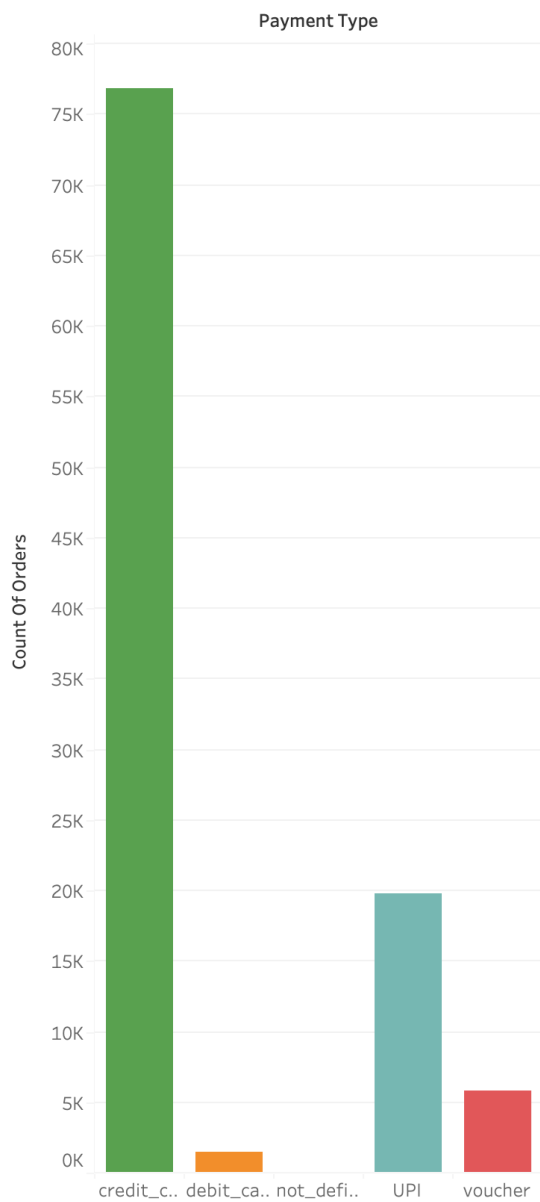


- We noticed that the payment value is the total value of order. If an order is bought on instalment payment value shows full amount instead of partial payments.
- 
-

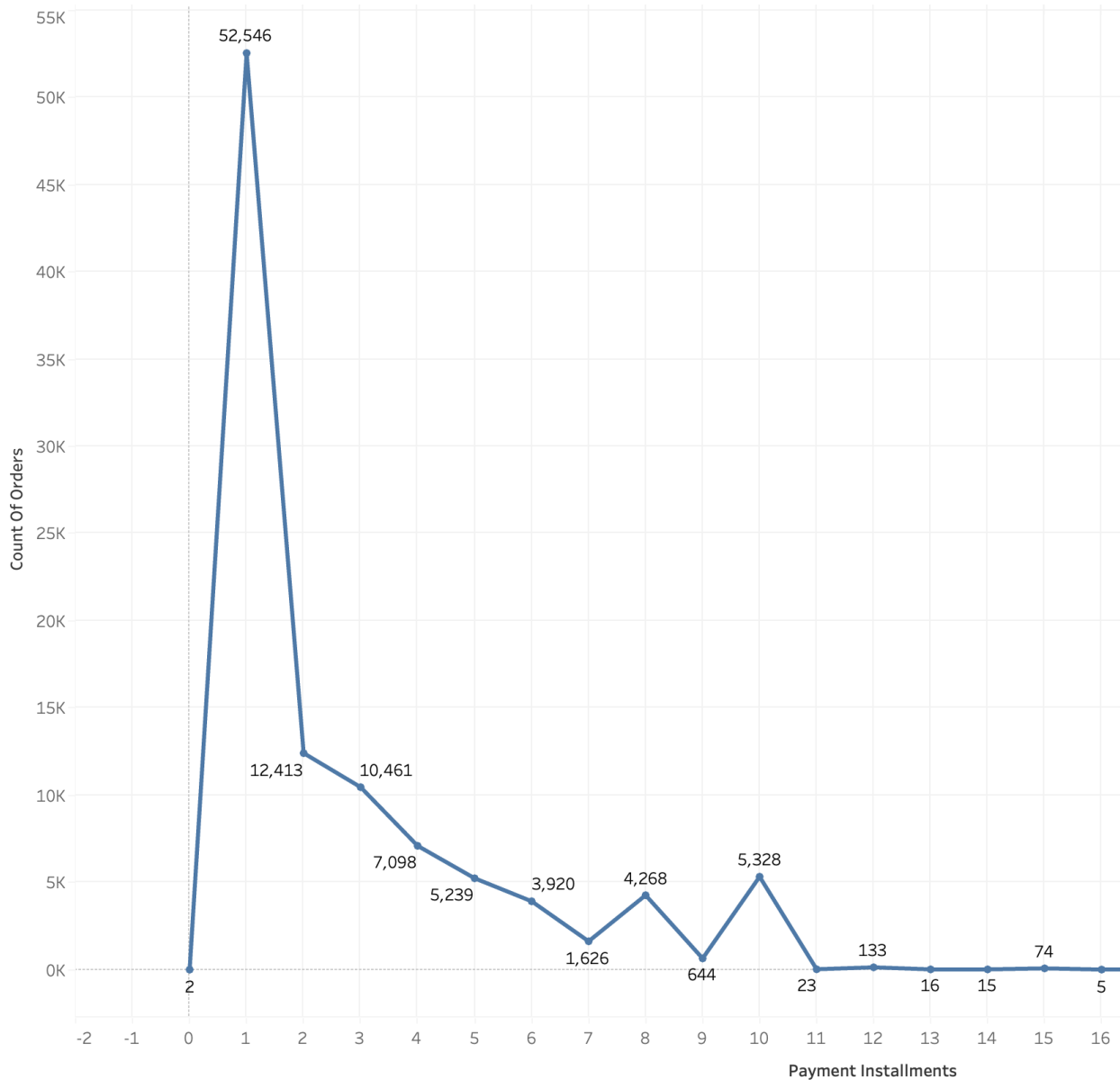


- Top 5 avg freight Value are in states DF, MG, PR, RJ and SP.





We can see that most of the payments are made using Credit Cards and least of the order are paid using Debit Card.



In the above graph we can see Most of the people wants prefers to have low number of Installments.

#### 8: Recommendation:

- In the above observation we noticed most of the people order the foods during Afternoon. We can safely reduce the discount if any is given in those hours. In order to prompt the operation.
- On the same premises most of the staff needs to be aligned during these hours.

- Most of the orders came from sao paulo. If the company can try to expand they can easily open new stores in the outskirts of the sou paulo.
- Company needs to further analyse to see why there is so much gap in sales in other states vs sao paulo.
- Looking at the pattern of the instalments. We saw people prefer to opt for a low number of instalments. So, the rate of interest on EMI can be increased because it is not a driving factor.
- Almost all of the people spend on credit cards. People spend on the money they loan from the bank instead of their own.