Business Case: Target SQL

Mindset:

1. Evaluation will be kept lenient, so make sure you attempt this case study.
2. It is understandable that you might struggle with getting started on this. Just brainstorm, discuss with peers, or get help from TAs.
3. Try to attempt this before it is discussed in the Live Case Discussion with the Instructor.
4. There is no right or wrong answer. We have to become comfortable dealing with uncertainty in business. This is exactly the skill we want to develop.

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.

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.

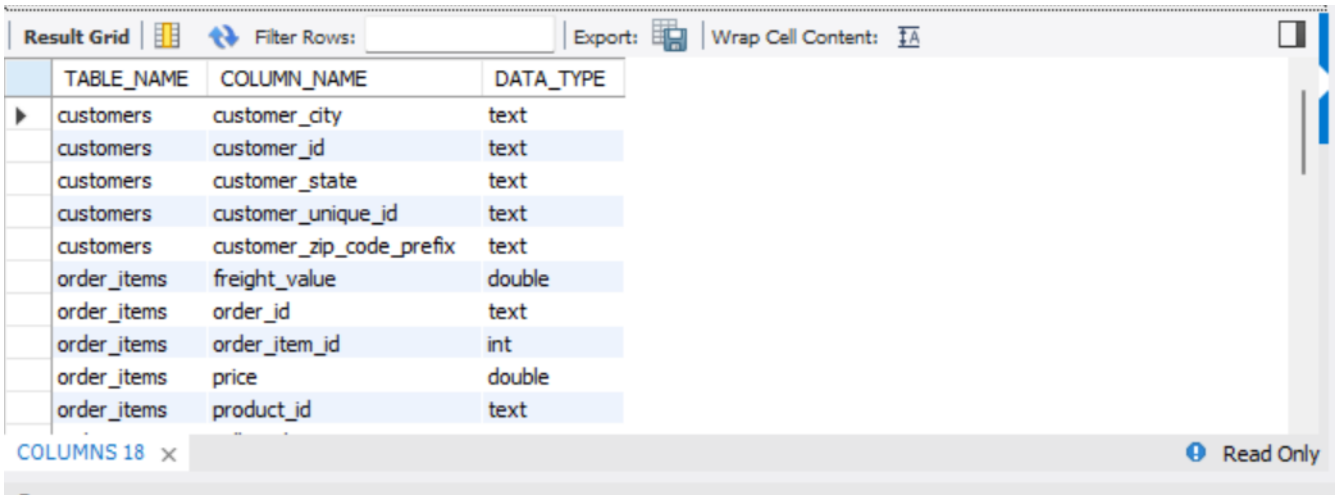
Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

1.1 Data Types of Columns of Tables

Query:-

```
SELECT TABLE_NAME,

COLUMN_NAME,

DATA_TYPE

FROM INFORMATION_SCHEMA.COLUMNS

WHERE TABLE_SCHEMA = "target-case1"

GROUP BY TABLE_NAME,  COLUMN_NAME

ORDER BY TABLE_NAME,  COLUMN_NAME;
```

Result:-



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TABLE_NAME	COLUMN_NAME	DATA_TYPE
customers	customer_city	text
customers	customer_id	text
customers	customer_state	text
customers	customer_unique_id	text
customers	customer_zip_code_prefix	text
order_items	freight_value	double
order_items	order_id	text
order_items	order_item_id	int
order_items	price	double
order_items	product_id	text

COLUMNS 18 | Read Only

- Data Types of all the tables given in dataset

1.2 Time period for which the data is given

Query:-

SELECT

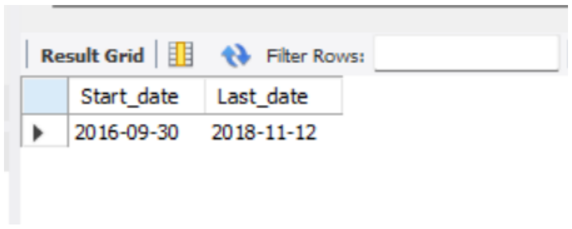
MIN(DATE(order_estimated_delivery_date)) AS Start_date,

MAX(DATE(order_estimated_delivery_date)) AS Last_date

FROM

Orders

Result:-



Result Grid | Filter Rows:

Start_date	Last_date
2016-09-30	2018-11-12

- Database starts entry from 2016-09-30 and last entry on 2018-11-12

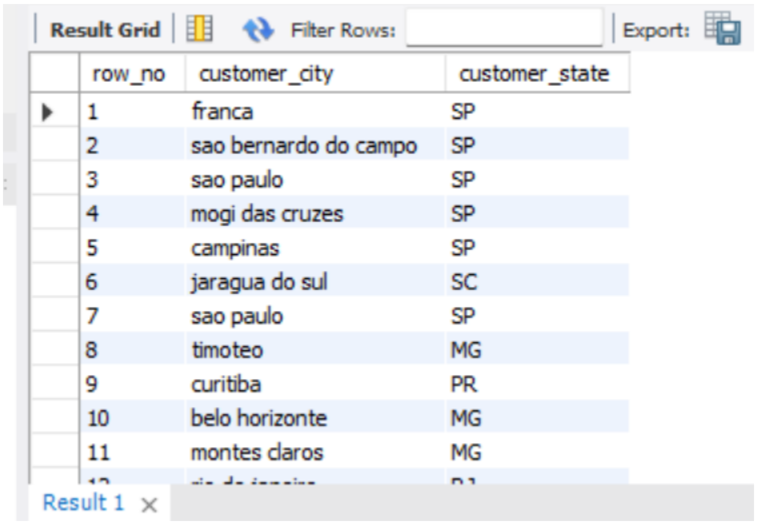
1.3 Cities and States covered in the dataset

Query:-

select row_number() over() as row_no, customer_city, customer_state

from customers

Result:-



row_no	customer_city	customer_state
1	franca	SP
2	sao bernardo do campo	SP
3	sao paulo	SP
4	mogi das cruzes	SP
5	campinas	SP
6	jaragua do sul	SC
7	sao paulo	SP
8	timoteo	MG
9	curitiba	PR
10	belo horizonte	MG
11	montes claros	MG

- 10 records of Cities and States from customers table

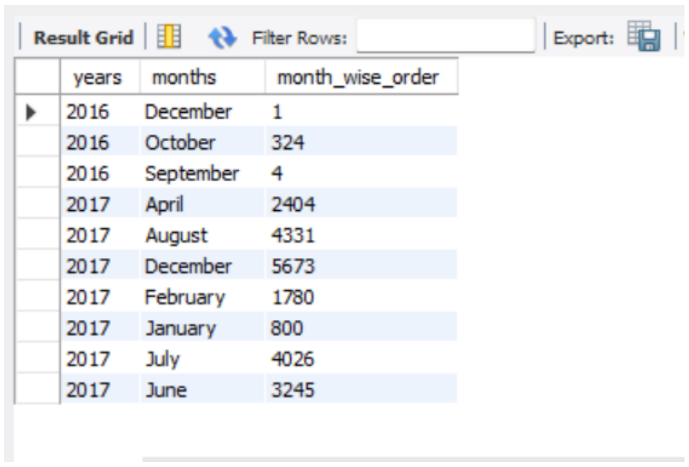
In-depth Exploration:

2.1 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?

Query:-

```
SELECT
   YEAR(order_purchase_timestamp) AS years,
   MONTHNAME(order_purchase_timestamp) AS months,
   COUNT(order_id) AS month_wise_order
FROM
   orders
GROUP BY years , months
ORDER BY years , months;
```

Result:-



years	months	month_wise_order
2016	December	1
2016	October	324
2016	September	4
2017	April	2404
2017	August	4331
2017	December	5673
2017	February	1780
2017	January	800
2017	July	4026
2017	June	3245

- Total no of order given in every month.

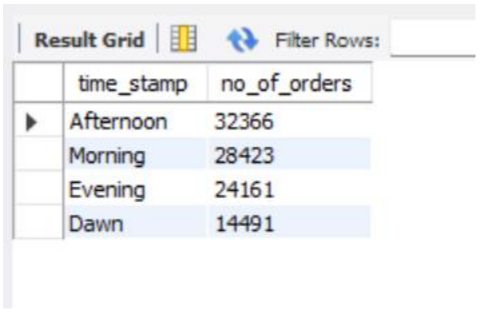
2.2 What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Query:-

```
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 5 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

Result:-



Result Grid | Filter Rows:

time_stamp	no_of_orders
Afternoon	32366
Morning	28423
Evening	24161
Dawn	14491

- Breakdown for the 4 main periods of calendar day (afternoon, morning, evening and dawn) with the order given in that period.

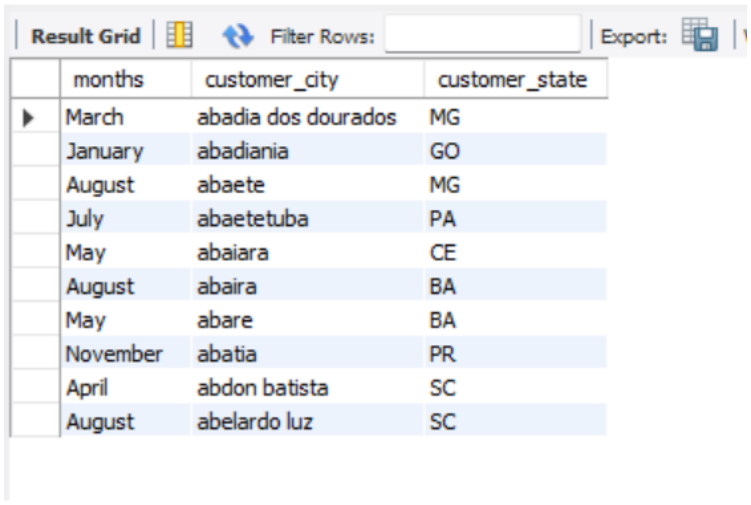
Evolution of E-commerce orders in the Brazil region:

3.1 Get month on month orders by states

Query:-

```
SELECT
   MONTHNAME(order_purchase_timestamp) AS months,
   customer_city,
   customer_state
FROM
   orders o
      JOIN
   customers c ON o.customer_id = c.customer_id
GROUP BY customer_city
ORDER BY customer_city , customer_state
```

Result:-



months	customer_city	customer_state
March	abadia dos dourados	MG
January	abadiania	GO
August	abaete	MG
July	abaetetuba	PA
May	abaiara	CE
August	abaira	BA
May	abare	BA
November	abatia	PR
April	abdon batista	SC
August	abelardo luz	SC

- Orders given in month within city and state.

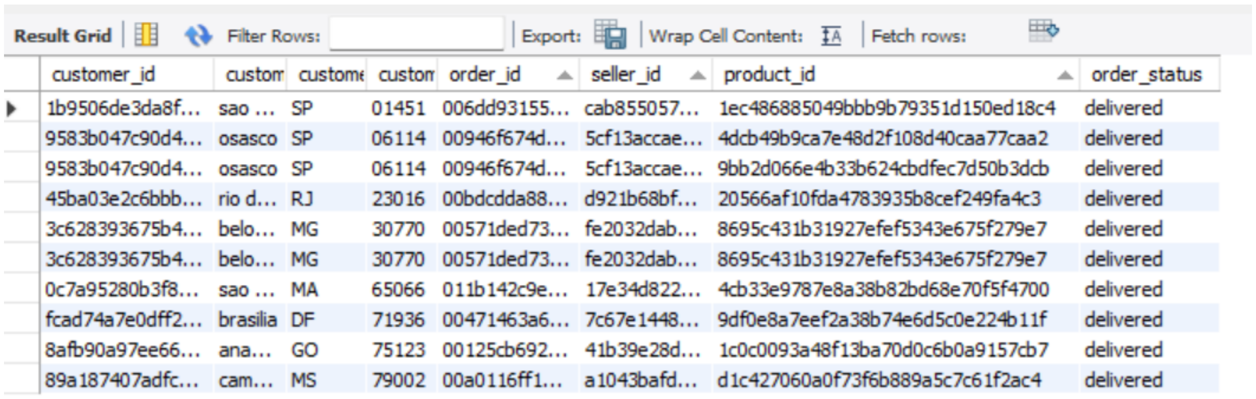
3.2 .How are customers distributed in Brazil

Query:-

```
SELECT
  c.customer_id,
  c.customer_city,
  c.customer_state,
  c.customer_zip_code_prefix,
  o.order_id,
  oi.seller_id,
  oi.product_id,
  o.order_status
FROM
  customers c
    JOIN
  orders o ON c.customer_id = o.customer_id
    JOIN
  order_items oi ON o.order_id = oi.order_id
WHERE
```

```
    order_status = 'delivered'
```

Result:-



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

customer_id	custom	custome	custom	order_id	seller_id	product_id	order_status
1b9506de3da8f...	sao ...	SP	01451	006dd93155...	cab855057...	1ec486885049bbb9b79351d150ed18c4	delivered
9583b047c90d4...	osasco	SP	06114	00946f674d...	5cf13accae...	4dcb49b9ca7e48d2f108d40caa77caa2	delivered
9583b047c90d4...	osasco	SP	06114	00946f674d...	5cf13accae...	9bb2d066e4b33b624cbdfec7d50b3dcb	delivered
45ba03e2c6bbb...	rio d...	RJ	23016	00bdcdda88...	d921b68bf...	20566af10fda4783935b8cef249fa4c3	delivered
3c628393675b4...	belo...	MG	30770	00571ded73...	fe2032dab...	8695c431b31927efef5343e675f279e7	delivered
3c628393675b4...	belo...	MG	30770	00571ded73...	fe2032dab...	8695c431b31927efef5343e675f279e7	delivered
0c7a95280b3f8...	sao ...	MA	65066	011b142c9e...	17e34d822...	4cb33e9787e8a38b82bd68e70f5f4700	delivered
fcad74a7e0dff2...	brasilia	DF	71936	00471463a6...	7c67e1448...	9df0e8a7eef2a38b74e6d5c0e224b11f	delivered
8afb90a97ee66...	ana...	GO	75123	00125cb692...	41b39e28d...	1c0c0093a48f13ba70d0c6b0a9157cb7	delivered
89a187407adfc...	cam...	MS	79002	00a0116ff1...	a1043bafd...	d1c427060a0f73f6b889a5c7c61f2ac4	delivered

- Database given orders delivered to customers

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

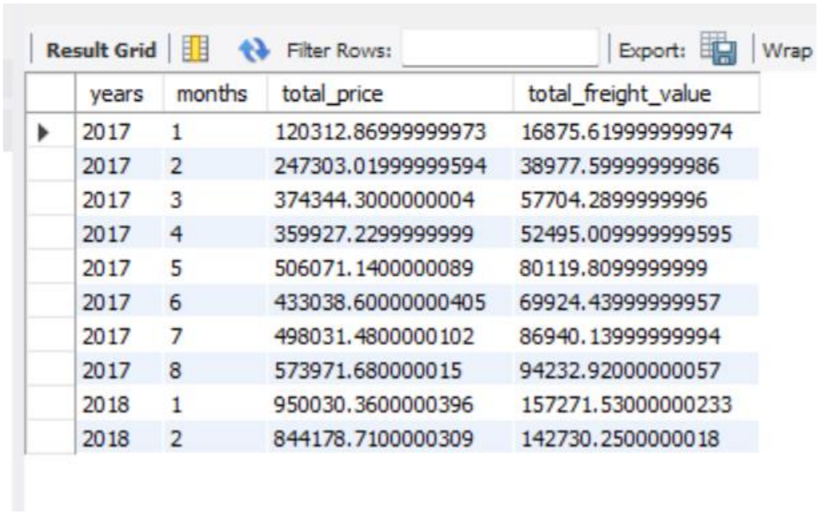
4.1 Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

Query:-

```
SELECT
    YEAR(o.order_purchase_timestamp) AS years,
    MONTH(o.order_purchase_timestamp) AS months,
    SUM(oi.price) AS total_price,
    SUM(oi.freight_value) AS total_freight_value
FROM
    orders o
        JOIN
    order_items oi ON o.order_id = oi.order_id
```

```
WHERE
    MONTH(order_purchase_timestamp) BETWEEN 1 AND 8
GROUP BY years , months
ORDER BY years , months;
```

Result:-



years	months	total_price	total_freight_value
2017	1	120312.86999999973	16875.619999999974
2017	2	247303.01999999594	38977.59999999986
2017	3	374344.3000000004	57704.2899999996
2017	4	359927.2299999999	52495.009999999595
2017	5	506071.1400000089	80119.8099999999
2017	6	433038.60000000405	69924.43999999957
2017	7	498031.4800000102	86940.13999999994
2017	8	573971.680000015	94232.92000000057
2018	1	950030.3600000396	157271.5300000233
2018	2	844178.7100000309	142730.250000018

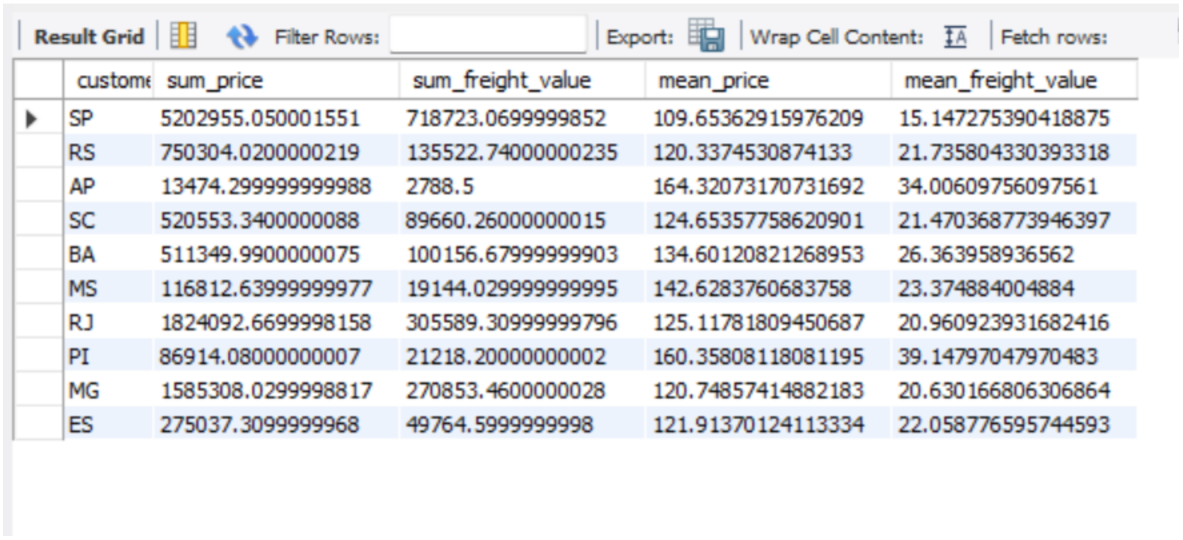
- Increasing cost of order from 2017 to 2018 (Jan to Aug only).

4.2 Mean & Sum of price and freight value by customer state

Query:-

```
SELECT
    customer_state,
    SUM(oi.price) AS sum_price,
    SUM(oi.freight_value) AS sum_freight_value,
    AVG(oi.price) AS mean_price,
    AVG(oi.freight_value) AS mean_freight_value
FROM
    orders o
    JOIN
  order_items oi ON o.order_id = oi.order_id
    JOIN
  customers c ON o.customer_id = c.customer_id
GROUP BY customer_state
```

Result:-



custome	sum_price	sum_freight_value	mean_price	mean_freight_value
SP	5202955.050001551	718723.0699999852	109.65362915976209	15.147275390418875
RS	750304.0200000219	135522.74000000235	120.3374530874133	21.735804330393318
AP	13474.299999999988	2788.5	164.32073170731692	34.00609756097561
SC	520553.3400000088	89660.26000000015	124.65357758620901	21.470368773946397
BA	511349.9900000075	100156.67999999903	134.60120821268953	26.363958936562
MS	116812.63999999977	19144.029999999995	142.6283760683758	23.374884004884
RJ	1824092.6699998158	305589.30999999796	125.11781809450687	20.960923931682416
PI	86914.08000000007	21218.20000000002	160.35808118081195	39.14797047970483
MG	1585308.0299998817	270853.4600000028	120.74857414882183	20.630166806306864
ES	275037.3099999968	49764.5999999998	121.91370124113334	22.058776595744593

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

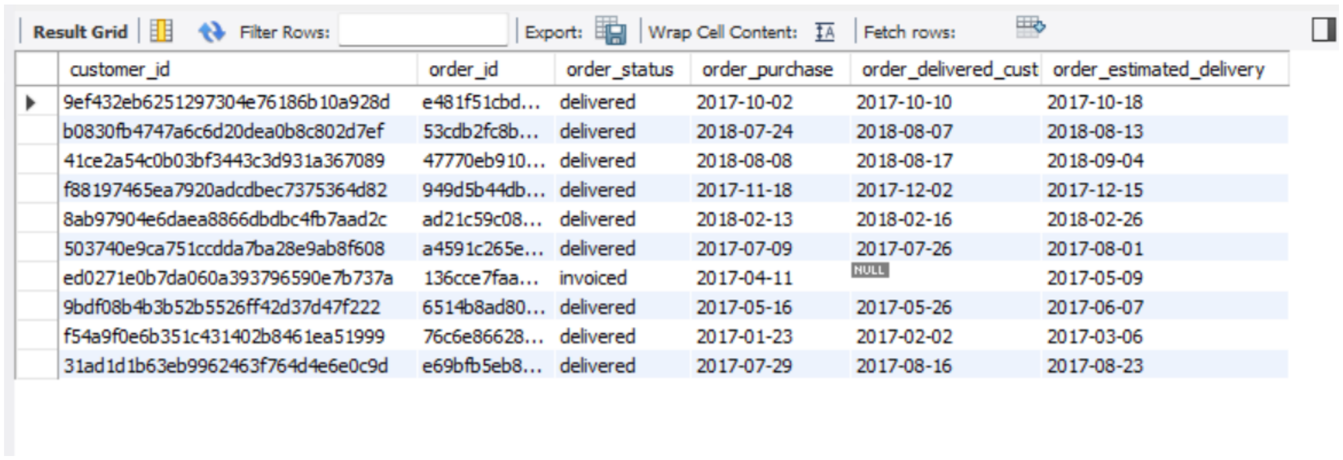
Analysis on sales, freight and delivery time

5.1 Calculate days between purchasing, delivering and estimated delivery

Query:-

```
SELECT
  customer_id,
  order_id,
  order_status,
  DATE(order_purchase_timestamp) AS order_purchase,
  DATE(order_delivered_customer_date) AS order_delivered_customer,
  DATE(order_estimated_delivery_date) AS order_estimated_delivery
FROM
  orders;
```

Result:-



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

customer_id	order_id	order_status	order_purchase	order_delivered_cust	order_estimated_delivery
9ef432eb6251297304e76186b10a928d	e481f51cbd...	delivered	2017-10-02	2017-10-10	2017-10-18
b0830fb4747a6c6d20dea0b8c802d7ef	53cdb2fc8b...	delivered	2018-07-24	2018-08-07	2018-08-13
41ce2a54c0b03bf3443c3d931a367089	47770eb910...	delivered	2018-08-08	2018-08-17	2018-09-04
f88197465ea7920adcdbec7375364d82	949d5b44db...	delivered	2017-11-18	2017-12-02	2017-12-15
8ab97904e6daea8866dbdbc4fb7aad2c	ad21c59c08...	delivered	2018-02-13	2018-02-16	2018-02-26
503740e9ca751ccdda7ba28e9ab8f608	a4591c265e...	delivered	2017-07-09	2017-07-26	2017-08-01
ed0271e0b7da060a393796590e7b737a	136cce7faa...	invoiced	2017-04-11	NULL	2017-05-09
9bdf08b4b3b52b5526ff42d37d47f222	6514b8ad80...	delivered	2017-05-16	2017-05-26	2017-06-07
f54a9f0e6b351c431402b8461ea51999	76c6e86628...	delivered	2017-01-23	2017-02-02	2017-03-06
31ad1d1b63eb9962463f764d4e6e0c9d	e69bfb5eb8...	delivered	2017-07-29	2017-08-16	2017-08-23

- Calculate days between purchasing, delivering and estimated delivery

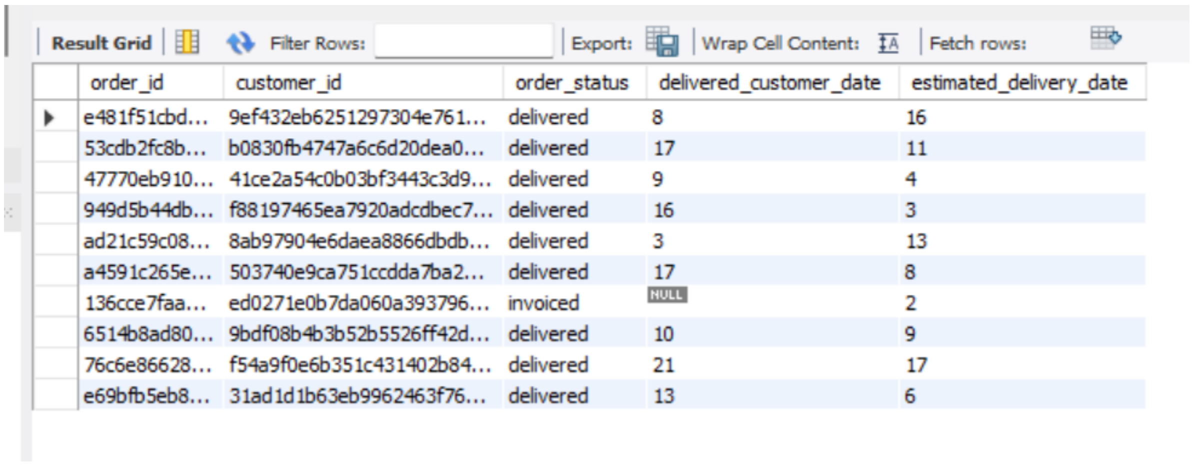
5.2 Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:-

- time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
- diff_estimated_delivery = order_estimated_delivery_date order_delivered_customer_date

Query:-

```
order_delivered_customer_date --
select customer_id,order_id,order_status,date( order_purchase_timestamp) as order_purchase
,
 date (order_delivered_customer_date) as order_delivered_customer,
 date (order_estimated_delivery_date) as order_estimated_delivery
from orders;
```

Result:-


Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

order_id	customer_id	order_status	delivered_customer_date	estimated_delivery_date
e481f51cbd...	9ef432eb6251297304e761...	delivered	8	16
53cdb2fc8b...	b0830fb4747a6c6d20dea0...	delivered	17	11
47770eb910...	41ce2a54c0b03bf3443c3d9...	delivered	9	4
949d5b44db...	f88197465ea7920adcdbec7...	delivered	16	3
ad21c59c08...	8ab97904e6daea8866dbdb...	delivered	3	13
a4591c265e...	503740e9ca751ccdda7ba2...	delivered	17	8
136cce7faa...	ed0271e0b7da060a393796...	invoiced	NULL	2
6514b8ad80...	9bdf08b4b3b52b5526ff42d...	delivered	10	9
76c6e86628...	f54a9f0e6b351c431402b84...	delivered	21	17
e69bfb5eb8...	31ad1d1b63eb9962463f76...	delivered	13	6

- Created new columns for time_of_time and diff_estimated_ as delivered_customer_date and estimated_delivery_date

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

Query:-

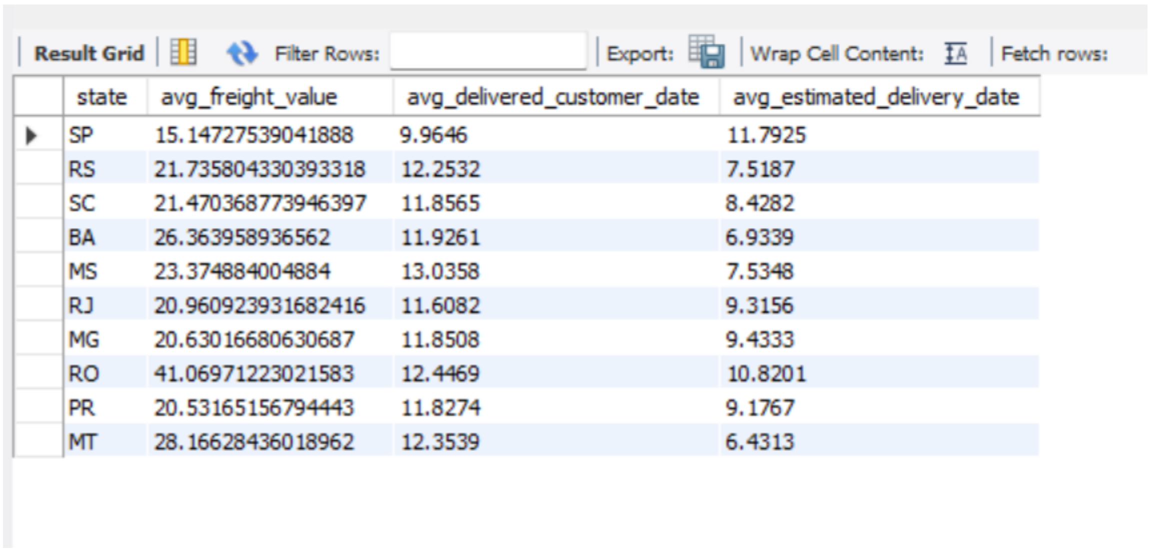
```
SELECT
  state,
  AVG(freight_value) AS avg_freight_value,
  AVG(delivered_customer_date) AS avg_delivered_customer_date,
  AVG(estimated_delivery_date) AS avg_estimated_delivery_date
FROM
  (SELECT
    c.customer_state AS state,
      freight_value,
      ABS((DAY(o.order_delivered_customer_date) - DAY(o.order_purchase_timestamp))) AS
delivered_customer_date,
      ABS((DAY(o.order_estimated_delivery_date) - DAY(o.order_purchase_timestamp))) AS
estimated_delivery_date
  FROM
    orders o
```

JOIN customers c ON o.customer_id = c.customer_id

JOIN order_items oi ON o.order_id = oi.order_id) x

GROUP BY state

Result:-



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

state	avg_freight_value	avg_delivered_customer_date	avg_estimated_delivery_date
SP	15.14727539041888	9.9646	11.7925
RS	21.735804330393318	12.2532	7.5187
SC	21.470368773946397	11.8565	8.4282
BA	26.363958936562	11.9261	6.9339
MS	23.374884004884	13.0358	7.5348
RJ	20.960923931682416	11.6082	9.3156
MG	20.63016680630687	11.8508	9.4333
RO	41.06971223021583	12.4469	10.8201
PR	20.53165156794443	11.8274	9.1767
MT	28.16628436018962	12.3539	6.4313

- Freight_value, time_to_delivery, diff_estimated_delivery according to order within state

5.4 Sort the data to get the following:-

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

-- desc--

Query:-

select customer_state,avg(oi.freight_value) as avg_freight_value

from orders o

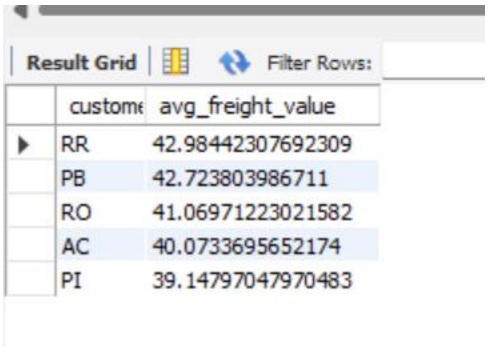
join order_items oi on o.order_id =oi.order_id

join customers c on o.customer_id=c.customer_id

group by customer_state

order by avg_freight_value desc limit 5

Result:-



Result Grid | Filter Rows:

customer	avg_freight_value
RR	42.98442307692309
PB	42.723803986711
RO	41.06971223021582
AC	40.0733695652174
PI	39.14797047970483

-- asc--

Query:-

select customer_state,avg(oi.freight_value) as avg_freight_value

from orders o

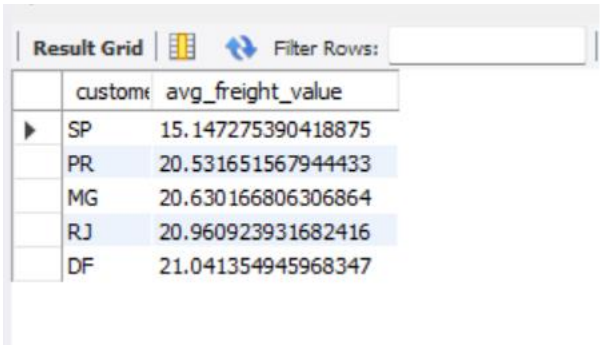
join order_items oi on o.order_id =oi.order_id

join customers c on o.customer_id=c.customer_id

group by customer_state

order by avg_freight_value limit 5

Result:-



Result Grid | Filter Rows:

customer	avg_freight_value
SP	15.147275390418875
PR	20.531651567944433
MG	20.630166806306864
RJ	20.960923931682416
DF	21.041354945968347

- Highest and lowest average freight value in state

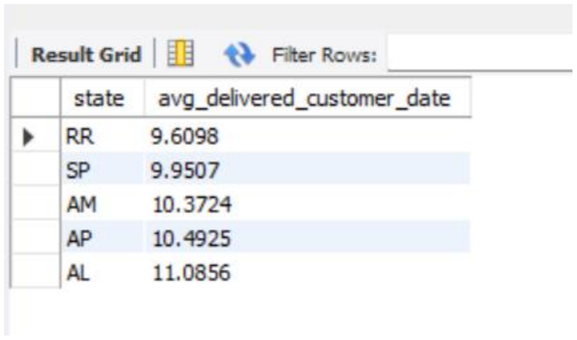
5.6 Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

-- Lowest--

Query:-

```
select state, avg(delivered_customer_date) as avg_delivered_customer_date
from
(select c.customer_state as state,
 abs((day(o.order_delivered_customer_date)-day(o.order_purchase_timestamp))) as
delivered_customer_date
from orders o
join customers c on o.customer_id=c.customer_id)x
group by state
order by avg_delivered_customer_date limit 5
```

Result:-



```
state  avg_delivered_customer_date
RR     9.6098
SP     9.9507
AM     10.3724
AP     10.4925
AL     11.0856
```

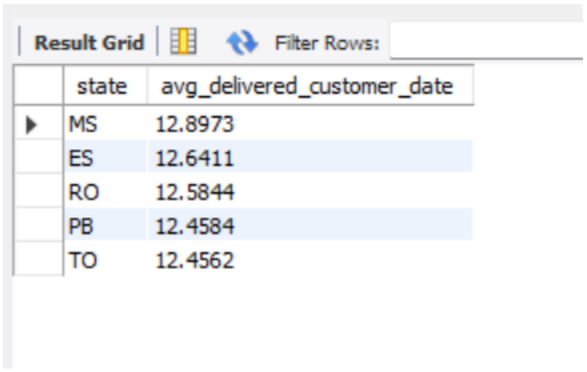
-- Highest--

Query:-

```
select state, avg(delivered_customer_date) as avg_delivered_customer_date
from
(select c.customer_state as state,
 abs((day(o.order_delivered_customer_date)-day(o.order_purchase_timestamp))) as
delivered_customer_date
from orders o
join customers c on o.customer_id=c.customer_id)x
group by state
```

order by avg_delivered_customer_date desc limit 5

Result:-



state	avg_delivered_customer_date
MS	12.8973
ES	12.6411
RO	12.5844
PB	12.4584
TO	12.4562

- Highest and lowest average time to delivery with state

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

-- Fast--

Query:-

select state, avg(estimated_delivery_date) as avg_estimated_delivery_date

from

(select c.customer_state as state,

abs((day(o.order_estimated_delivery_date)-day(o.order_purchase_timestamp))) as

estimated_delivery_date

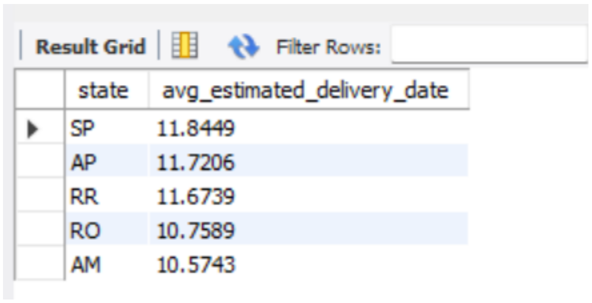
from orders o

join customers c on o.customer_id=c.customer_id)x

group by state

order by avg_estimated_delivery_date desc limit 5

Result:-


Result Grid | Filter Rows:

state	avg_estimated_delivery_date
SP	11.8449
AP	11.7206
RR	11.6739
RO	10.7589
AM	10.5743

-- Not fast--

Query:-

select state, avg(estimated_delivery_date) as avg_estimated_delivery_date

from

(select c.customer_state as state,

abs((day(o.order_estimated_delivery_date)-day(o.order_purchase_timestamp))) as

estimated_delivery_date

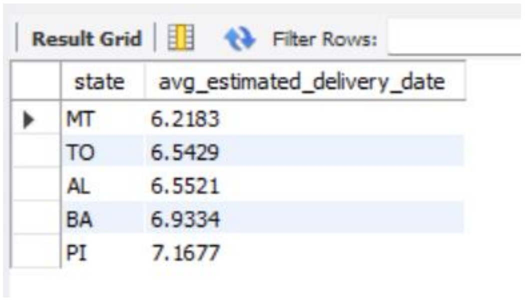
from orders o

join customers c on o.customer_id=c.customer_id)x

group by state

order by avg_estimated_delivery_date limit 5

Result:-


Result Grid | Filter Rows:

state	avg_estimated_delivery_date
MT	6.2183
TO	6.5429
AL	6.5521
BA	6.9334
PI	7.1677

- Delivery is really fast/ not so fast compared to estimated date within state

6.1 Payment type analysis:

-- i) Month over Month count of orders for different payment types—

Query:-

select payment_type, count(months) as months

from

(select monthname(o.order_approved_at) as months,payment_type

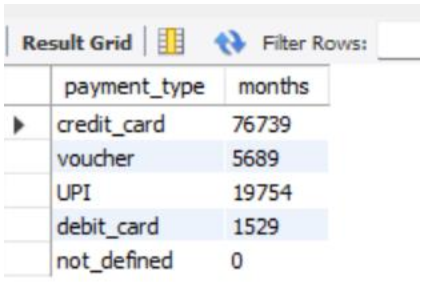
from payments p

join orders o on p.order_id=o.order_id

) x

group by payment_type

Result:-



payment_type	months
credit_card	76739
voucher	5689
UPI	19754
debit_card	1529
not_defined	0

- Database from 2016-09-30 to 2018-11-12 , type of payment done and how many time it has been done during in database

-- ii) .Distribution of payment installments and count of orders--

Query:-

select p.payment_installments ,count(o.order_id) as count_of_orders

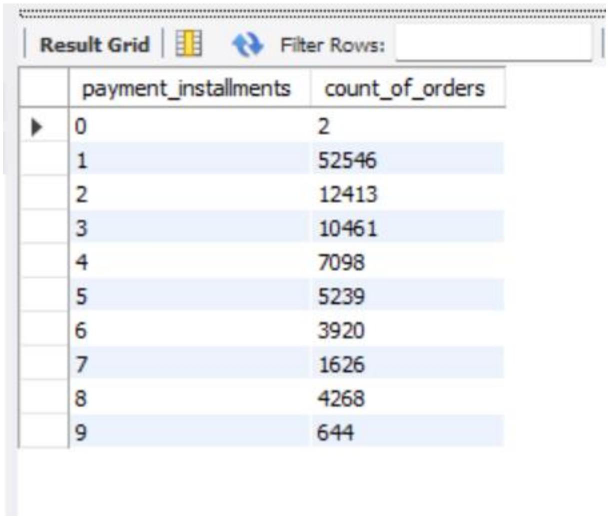
from payments p

join orders o on p.order_id=o.order_id

group by payment_installments

order by payment_installments

Result:-


Result Grid
Filter Rows:

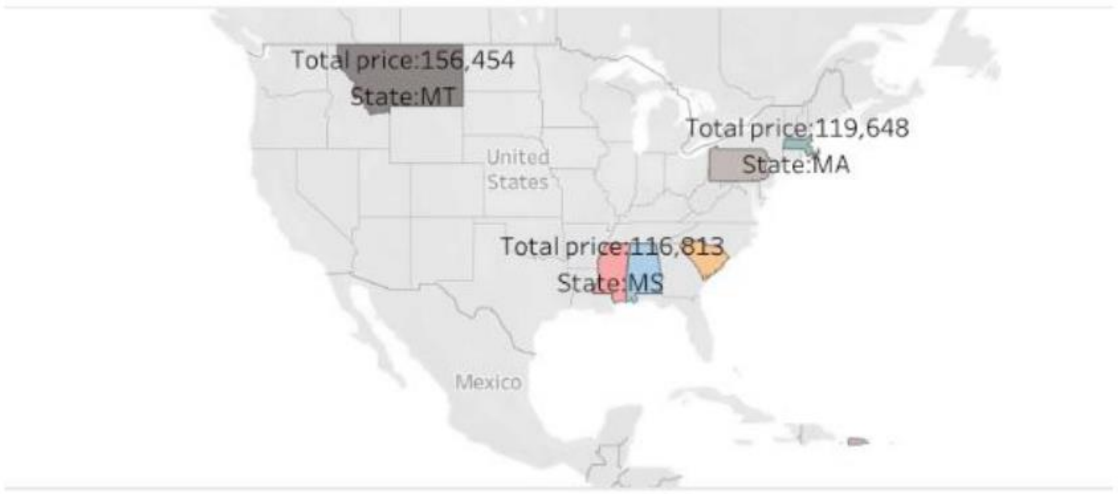
payment_installments	count_of_orders
0	2
1	52546
2	12413
3	10461
4	7098
5	5239
6	3920
7	1626
8	4268
9	644

- Total of order in number of installments in case of EMI purchases:

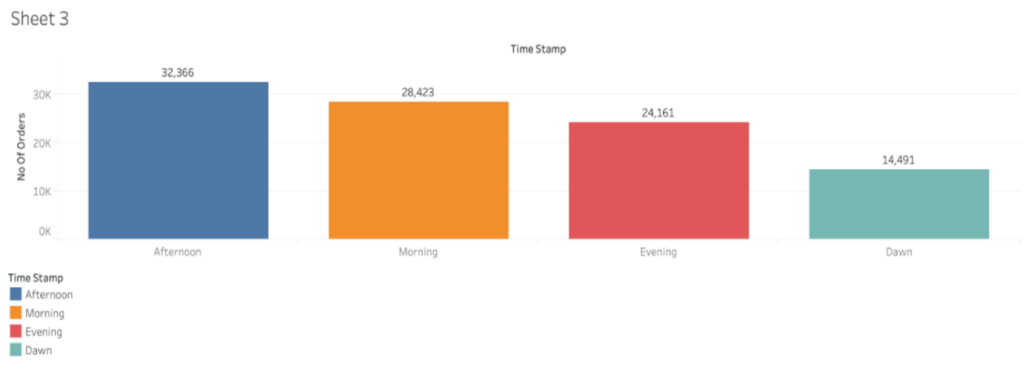
Analysis:-

In Database starting entry from 2016-09-30 and last entry on 2018-11-12.

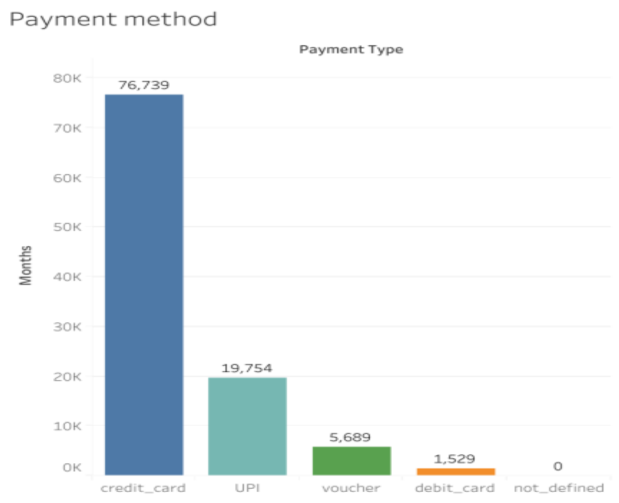
1. Highest sell in state.



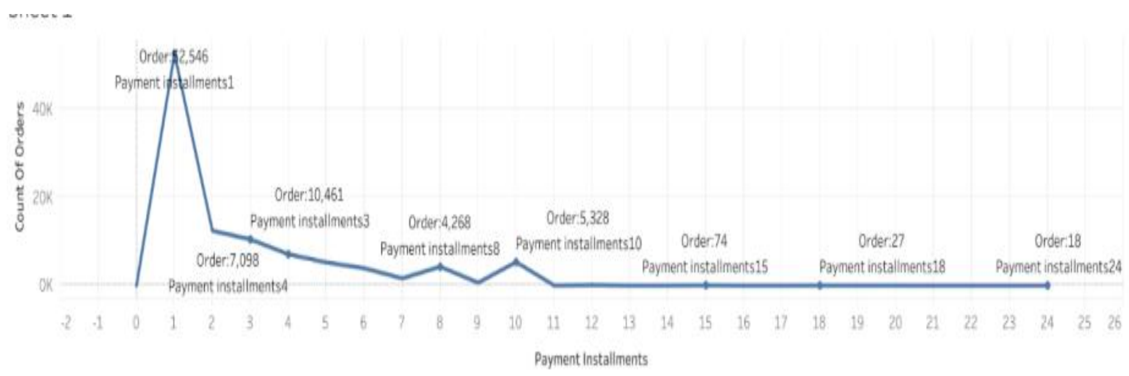
2. Highest no. of customers order in the afternoon.



3. Mostly customers order by using credit card then other payment method.



4. Customers payment installments (EMI) use to order


Order:52,546
Payment installments1
Order:10,461
Payment installments3
Order:7,098
Payment installments4
Order:4,268
Payment installments8
Order:5,328
Payment installments10
Order:74
Payment installments15
Order:27
Payment installments18
Order:18
Payment installments24
Count Of Orders
Payment Installments