TARGET BUSINESS CASE

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

1.1Data type of all columns in the "customers" table.

```
2
3  SELECT COLUMN_NAME,DATA_TYPE
4  FROM `Target_Dataset.INFORMATION_SCHEMA.COLUMNS`
5  WHERE TABLE_NAME = 'customers'
```

Query results — SAVE RESULTS

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	COLUMN_NAME	DATA_TYPE
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Query:
```
SELECT COLUMN_NAME,DATA_TYPE
FROM `Target_Dataset.INFORMATION_SCHEMA.COLUMNS`
WHERE TABLE_NAME = 'customers'
```

INSIGHTS: **We can see in above Output Customer_id, customer_unique_id ,customer_city, customer_state is in String :**

Means we can conclude that these columns data are comprising of numbers & Text so that's why we are refering it as an String

customer_zip_code_prefix is in Integer form: By this we can conclude that customer adresses's zip code would be an integer e.g. 45666

Recommendation: **Since question is asking for Data type therefore, no recommendation here;**

1.2 Get the time range between which the orders were placed.

```
SELECT min(order_purchase_timestamp) as Begin_Time,
       max(order_purchase_timestamp) as End_Time
FROM `Target_Dataset.orders`;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON

Row	Begin_Time	End_Time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Query :
```
SELECT min(order_purchase_timestamp) as Begin_Time,
       max(order_purchase_timestamp) as End_Time
FROM `Target_Dataset.orders`;
```

Insights : By this output we can see that Order has been placed between above mentioned range

Recommendation: Target can incur more expenses on marketing, so that people get aware about these products & purchase more

By doing this Number of Orders & Time range of purchasing products both can be increase.

1.3 Count the Cities & States of customers who ordered during the given period.

```
SELECT
    COUNT(DISTINCT c.customer_city) AS unique_cities,
    COUNT(DISTINCT c.customer_state) AS unique_states
FROM `Target_Dataset.customers`c
JOIN `Target_Dataset.orders` o
ON c.customer_id = o.customer_id
WHERE EXTRACT(YEAR FROM TIMESTAMP(o.order_purchase_timestamp)) BETWEEN 2016 AND 2018;
```

Query results SAVE RESULTS

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	unique_cities	unique_states
1	4119	27

Query:
```
SELECT
    COUNT(DISTINCT c.customer_city) AS unique_cities,
    COUNT(DISTINCT c.customer_state) AS unique_states
FROM `Target_Dataset.customers` c
JOIN `Target_Dataset.orders` o ON c.customer_id = o.customer_id
```

```
WHERE EXTRACT(YEAR FROM TIMESTAMP(o.order_purchase_timestamp)) BETWEEN 2016 AND 2018;
```

Insights: By the above output we can conclude;
27 States where products have been ordered by customers in which 4,119 No of cities

Recommendation: These No of orders can be increased if Target launch their posters like an advertisement at Bus stand, Metro Station, Movies halls etc where number of people gathered more.

2.In-depth Exploration:

2.1 Is there a growing trend in the no. of orders placed over the past years?

```
SELECT COUNT(DISTINCT(order_id))as No_of_order,
       EXTRACT(year from order_purchase_timestamp) as year
FROM`Target_Dataset.orders`
GROUP BY 2
ORDER BY 2 asc;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

Row	No_of_order	year
1	329	2016
2	45101	2017
3	54011	2018

Query:
```
SELECT COUNT(DISTINCT(order_id))as No_of_order,
       EXTRACT(year from order_purchase_timestamp) as year
FROM`Target_Dataset.orders`
GROUP BY 2
ORDER BY 2 asc;
```

Insights : By the above Output we can conclude that:

In Year 2017 : 44772 No of order has increased as compared to 2016 (growth of 13,609%)

In Year 2018 : 8910 No of order has increased from year 2017 (growth of 16% only) & 53682 from year 2016 (growth of 16317%)

Recommendations: **Since % of order purchase ratio has been decreased in Year 2018 ,we would suggest that try to maintain the customers trust by improving your products quality , keep checking customer reviews on each products.**

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

```
SELECT  COUNT(DISTINCT(order_id))as No_of_order,
        EXTRACT(month from order_purchase_timestamp) as Month,
        EXTRACT(year from order_purchase_timestamp) as  Year
FROM`Target_Dataset.orders`
GROUP BY 2,3
ORDER BY 3,2,1 asc;
```

JOB INFORMATION | RESULTS | CHART | JSON

Row	No_of_order	Month	Year
1	4	9	2016
2	324	10	2016
3	1	12	2016
4	800	1	2017
5	1780	2	2017
6	2682	3	2017
7	2404	4	2017
8	3700	5	2017
9	3245	6	2017

Row	No_of_order	Month	Year
10	4026	7	2017
11	4331	8	2017
12	4285	9	2017
13	4631	10	2017
14	7544	11	2017
15	5673	12	2017
16	7269	1	2018
17	6728	2	2018

Query :

```
SELECT COUNT(DISTINCT(order_id))as No_of_order,
       EXTRACT(month from order_purchase_timestamp) as Month,
       EXTRACT(year from order_purchase_timestamp) as year
FROM`Target_Dataset.orders`
GROUP BY 2,3
ORDER BY 3,2,1 asc;
```

Insights : From above output we can see that :

Monthly No of orders has been fluctuating & It doesn't a create a good impression of Target among customers .

Recommendation: Launch quality products as per season according, so that people can use them according to month season.

2. 3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

```
SELECT COUNT(DISTINCT(order_id)) as No_of_order,
CASE
    WHEN Hour BETWEEN 0 AND 6 THEN 'Dawn'
    WHEN Hour BETWEEN 7 AND 12 THEN 'Mornings'
    WHEN Hour BETWEEN 13 AND 18 THEN 'Afternoon'
    WHEN Hour BETWEEN 19 AND 23 THEN  'Night'
END AS Order_Time
FROM
(SELECT DISTINCT(order_id),
        EXTRACT(hour from order_purchase_timestamp) as Hour
From `Target_Dataset.orders`) a
GROUP BY Order_Time
```

Row	No_of_order	Order_Time
1	27733	Mornings
2	5242	Dawn
3	38135	Afternoon
4	28331	Night

Query:

```
SELECT COUNT(DISTINCT(order_id)) as No_of_order,
CASE
   WHEN Hour BETWEEN 0 AND 6 THEN 'Dawn'
   WHEN Hour BETWEEN 7 AND 12 THEN 'Mornings'
   WHEN Hour BETWEEN 13 AND 18 THEN 'Afternoon'
   WHEN Hour BETWEEN 19 AND 23 THEN  'Night'
END AS Order_Time
FROM
(SELECT DISTINCT(order_id),
        EXTRACT(hour from order_purchase_timestamp) as Hour
From `Target_Dataset.orders`) a
GROUP BY Order_Time
```

Insights: By the above output we can say that:

In Dawn time No of orders are less as compared to others time due to various factors such as customers wake up late & No of orders **38135** reflects customers usually prefer afternoon time for buying the product

Recommendations: Target can launch any impressive offers for Dwan time range, so that people can get attracted & purchase more

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

3. 1 Get the month on month no. of orders placed in each state.

```
SELECT c.customer_state,
       COUNT(DISTINCT(o.order_id))as No_of_order,
       EXTRACT(month from o.order_delivered_customer_date) as Month,
       EXTRACT(year from o.order_delivered_customer_date) as year
FROM `Target_Dataset.orders` o
INNER JOIN `Target_Dataset.customers` c
ON o.customer_id = c.customer_id
WHERE o.order_status = 'delivered'AND o.order_delivered_customer_date IS NOT NULL
GROUP BY 1,3,4
```

Row	customer_state	No_of_order	Month	year
1	GO	71	5	2017
2	SP	1454	5	2017
3	RS	201	5	2017
4	BA	129	5	2017
5	MG	414	5	2017
6	MT	41	5	2017
7	RJ	520	5	2017
8	SC	155	5	2017
9	SE	16	5	2017
10	PE	23	4	2017

Query:

```
SELECT c.customer_state,
       COUNT(DISTINCT(o.order_id))as No_of_order,
       EXTRACT(month from o.order_delivered_customer_date) as Month,
       EXTRACT(year from o.order_delivered_customer_date) as year
FROM `Target_Dataset.orders` o
INNER JOIN `Target_Dataset.customers` c
ON o.customer_id = c.customer_id
WHERE o.order_status = 'delivered'AND o.order_delivered_customer_date IS NOT NULL
GROUP BY 1,3,4
```

Insights: As we can see from above Output, No. of orders values are different in each state as per Year 2017.It reflects customers demand of products as per categorised states.

Recommendations: For higher sales, Target can aim for those states where No of orders are less

3. 2 How are the customers distributed across all the states?

```
SELECT COUNT(DISTINCT(customer_id)) as No_of_customer,
       customer_state
FROM `Target_Dataset.customers`
GROUP BY 2
ORDER BY 1 asc;
```

Row	No_of_customer	customer_state
1	46	RR
2	68	AP
3	81	AC
4	148	AM
5	253	RO
6	280	TO
7	350	SE
8	413	AL
9	485	RN
10	495	PI

Query:
```
SELECT COUNT(DISTINCT(customer_id)) as No_of_customer,
       customer_state
FROM `Target_Dataset.customers`
GROUP BY 2
ORDER BY 1 asc;
```

Insights: We can see from above output No of customers are very less in RR state as compared to other states.

Recommendations : Target can do survey in each State for knowing the customers demand & By doing this they can launch their products as per customer demand in a state where Numbers of customers are less.

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

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

```
1
2  WITH CTE AS (
3    SELECT
4      ROUND(SUM(p.payment_value),2) as Cost_of_orders,
5      EXTRACT (year from o.order_purchase_timestamp) as Year
6    FROM `Target_Dataset.orders` o
7    INNER JOIN `Target_Dataset.payments` p
8    ON o.order_id = p.order_id
9    WHERE EXTRACT (year from o.order_purchase_timestamp) IN (2017,2018)
10   AND EXTRACT (month from o.order_purchase_timestamp) BETWEEN 1 AND 8
11   GROUP BY 2)
12 SELECT
13   ((SUM(CASE
14     WHEN CTE.year = 2018 THEN CTE.Cost_of_orders ELSE 0
15   END) -
16   SUM(CASE
17     WHEN CTE.year = 2017 THEN CTE.Cost_of_orders ELSE 0
18   END)) / SUM(CASE
19     WHEN CTE.year = 2017 THEN CTE.Cost_of_orders ELSE 0
20   END)) * 100 AS year_on_year_increase_Percent
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

Row	year_on_year_increas
1	136.9768716466...

Query:

```
WITH CTE AS (
    SELECT
    ROUND(SUM(p.payment_value),2) as Cost_of_orders,
    EXTRACT (year from o.order_purchase_timestamp) as Year
  FROM `Target_Dataset.orders` o
  INNER JOIN `Target_Dataset.payments` p
  ON o.order_id = p.order_id
  WHERE EXTRACT (year from o.order_purchase_timestamp) IN (2017,2018)
  AND EXTRACT (month from o.order_purchase_timestamp) BETWEEN 1 AND 8
  GROUP BY 2)
SELECT
  ((SUM(CASE
    WHEN CTE.year = 2018 THEN CTE.Cost_of_orders ELSE 0
  END) -
  SUM(CASE
    WHEN CTE.year = 2017 THEN CTE.Cost_of_orders ELSE 0
  END)) / SUM(CASE
    WHEN CTE.year = 2017 THEN CTE.Cost_of_orders ELSE 0
```

```
  END)) * 100 AS year_on_year_increase
FROM CTE;
```

Insights: **From above output we can see % No of order increament from year 2017 to 2018 is 136. 97, No of order increament could be happen for many reasons like enhance the quality of product, incurred expenses on branding.**

Recommendation: **This % can increased in upcoming years if Target focused on customer demand only that what actually customer desired in product**

4.2 Calculate the Total & Average value of order price for each state.

```
SELECT c.customer_state,
       ROUND(SUM(oi.price),2) AS Total_value,
       ROUND(AVG(oi.price),2) AS Average_value
FROM`Target_Dataset.orders`o
JOIN `Target_Dataset.order_items` oi
ON o.order_id = oi.order_id
JOIN`Target_Dataset.customers`c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY Total_value desc,
         Average_value desc;
```

Row	customer_state	Total_value	Average_value
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65
7	BA	511349.99	134.6
8	DF	302603.94	125.77
9	GO	294591.95	126.27
10	ES	275037.31	121.91

Query :

```
SELECT c.customer_state,
       ROUND(SUM(oi.price),2) AS Total_value,
       ROUND(AVG(oi.price),2) AS Average_value
FROM`Target_Dataset.orders`o
JOIN `Target_Dataset.order_items` oi
```

```
ON o.order_id = oi.order_id
JOIN`Target_Dataset.customers`c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY Total_value desc,
         Average_value desc;
```

Insights: From the above output we can see Product values keep varying as per States, This could be happen because of different categorization of products

Recommendation : Target can focus more on quality of products and can try to focus on minimize the cost of product, By this no of customers can increase & sales turnover would get high

4. 3 Calculate the Total & Average value of order freight for each state.

```
SELECT c.customer_state,
       ROUND(SUM(freight_value)) AS Total_freight,
       ROUND(AVG(freight_value)) AS Average_freight
FROM`Target_Dataset.orders`o
JOIN `Target_Dataset.order_items` oi
ON o.order_id = oi.order_id
JOIN`Target_Dataset.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY Total_freight asc,
         Average_freight asc;
```

Row	customer_state	Total_freight	Average_freight
1	RR	2235.0	43.0
2	AP	2789.0	34.0
3	AC	3687.0	40.0
4	AM	5479.0	33.0
5	RO	11417.0	41.0
6	TO	11733.0	37.0
7	SE	14111.0	37.0
8	AL	15915.0	36.0
9	RN	18860.0	36.0
10	MS	19144.0	23.0

Query :

```
SELECT c.customer_state,
       ROUND(SUM(freight_value)) AS Total_freight,
       ROUND(AVG(freight_value)) AS Average_freight
FROM`Target_Dataset.orders`o
JOIN `Target_Dataset.order_items` oi
ON o.order_id = oi.order_id
JOIN`Target_Dataset.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY Total_freight asc,
         Average_freight asc;
```

Insights : From the above Output we can say that Freight value has been changing as per States,

It could happened because of longest or shortest distance from the warehouse location.

Recommendation: Target can invest to buy different state locations warehouses so that freight value can decreased

5. Analysis based on sales, freight and delivery time

5. 1 Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
Do this in a single query.

You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- time_to_deliver = order_delivered_customer_date - order_purchase_timestamp
- diff_estimated_delivery = order_delivered_customer_date - order_estimated_delivery_date

```
SELECT DISTINCT(order_id),
       date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as Delivery_time,
       date_diff(order_delivered_customer_date,order_estimated_delivery_date,day) as Diff_estimated_delivery
FROM `Target_Dataset.orders`
WHERE order_status = 'delivered'
AND order_delivered_customer_date IS NOT NULL
ORDER BY 3,2 desc ;
```

Row	order_id	Delivery_time	Diff_estimated_deliv
1	0607f0efea4b566f1eb8f7d3c2...	3	-146
2	c72727d29cde4cf870d569bf6...	6	-139
3	eec7f369423b033e549c02f3c...	20	-134
4	c2bb89b5c1dd978d507284be...	16	-123
5	40dc2ba6f322a17626aac6244...	7	-108
6	1a695d543b7302aa9446c8d5f...	12	-83
7	39e0115911bf404857e14baa7...	11	-82
8	559eea5a72341a4c82dbce988...	13	-77
9	c5132855100a12d63ed4e8ae0...	12	-77
10	38930f76efb00b138f4d632e4d...	11	-77

Query :

```
SELECT DISTINCT(order_id),
       date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as Delivery_time,
      date_diff(order_delivered_customer_date,order_estimated_delivery_date,day) as Diff_estimated_delivery
FROM `Target_Dataset.orders`
WHERE order_status = 'delivered'
AND order_delivered_customer_date IS NOT NULL
ORDER BY 3,2 desc ;
```

Insights: From above Output we can say that date of delivery of products are not estimated in a proper way.

Recommendations: Target can do focus on estimation date of delivery of products, It would impact directly to customers as if they will received the products exactly on same date as mentioned & By doing this trust can be achieved from customers.

5.2 Find out the top 5 states with the highest & lowest average freight value.

```sql
With Table_1 as
  (SELECT c.customer_state,
        ROUND(AVG(oi.freight_value),2) as Avg_Freight_Value,
        dense_rank() over(order by ROUND(AVG(oi.freight_value),2) desc) as h_rnk,
        dense_rank() over (order by  ROUND(AVG(oi.freight_value),2) asc)  as l_rnk
FROM `Target_Dataset.orders` o

INNER JOIN `Target_Dataset.customers`c
ON o.customer_id = c.customer_id

INNER JOIN `Target_Dataset.order_items`oi
ON o.order_id =  oi.order_id

GROUP BY 1)

(SELECT customer_state,Avg_Freight_Value,
'Highest_Top_5_State' as Type
FROM Table_1
WHERE h_rnk <=5
ORDER BY h_rnk)

UNION ALL
```

```sql
(SELECT customer_state,Avg_Freight_Value,
'Lowest_Low_5_State' as Type
FROM Table_1
WHERE l_rnk <=5
ORDER BY l_rnk)

ORDER BY Avg_Freight_Value desc ;
```

Row	customer_state	Avg_Freight_Value	Type
1	RR	42.98	Highest_Top_5_State
2	PB	42.72	Highest_Top_5_State
3	RO	41.07	Highest_Top_5_State
4	AC	40.07	Highest_Top_5_State
5	PI	39.15	Highest_Top_5_State
6	DF	21.04	Lowest_Low_5_State
7	RJ	20.96	Lowest_Low_5_State
8	MG	20.63	Lowest_Low_5_State
9	PR	20.53	Lowest_Low_5_State
10	SP	15.15	Lowest_Low_5_State

Query: `With Table_1 as`

```sql
(SELECT c.customer_state,
      ROUND(AVG(oi.freight_value),2) as Avg_Freight_Value,
      dense_rank() over(order by ROUND(AVG(oi.freight_value),2) desc) as h_rnk,
```

```
        dense_rank() over (order by  ROUND(AVG(oi.freight_value),2) asc)  as l_rnk
FROM `Target_Dataset.orders` o

INNER JOIN `Target_Dataset.customers`c
ON o.customer_id = c.customer_id

INNER JOIN `Target_Dataset.order_items`oi
ON o.order_id =  oi.order_id

GROUP BY 1)

(SELECT customer_state,Avg_Freight_Value,
'Highest_Top_5_State' as Type
FROM Table_1
WHERE h_rnk <=5
ORDER BY h_rnk)

UNION ALL

(SELECT customer_state,Avg_Freight_Value,
'Lowest_Low_5_State' as Type
FROM Table_1
WHERE l_rnk <=5
ORDER BY l_rnk)

ORDER BY Avg_Freight_Value desc;
```

Insights: By the above output we can see the difference in which state products value are high and where ‘s low

Recommendation : Target can identify the customer demand and charged the values accordingly

5. 3 Find out the top 5 states with the highest & lowest average delivery time.

```
With Table_1 as
  (SELECT c.customer_state,
        ROUND(AVG(Timestamp_diff(o.order_delivered_customer_date, o.order_purchase_timestamp,day)),2) as AVG_Delivery_Time,
        dense_rank() over(order by AVG(Timestamp_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)) desc) as h_rnk,
        dense_rank() over(order by AVG(Timestamp_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)) asc) as l_rnk

FROM `Target_Dataset.orders` o

INNER JOIN `Target_Dataset.customers`c
ON o.customer_id = c.customer_id

INNER JOIN `Target_Dataset.order_items`oi
ON o.order_id =  oi.order_id

GROUP BY 1)

(SELECT customer_state,AVG_Delivery_Time,
'Highest_Top_5_State' as Type
FROM Table_1
WHERE h_rnk <=5
ORDER BY h_rnk)
```

```
UNION ALL

(SELECT customer_state,AVG_Delivery_Time,
'Lowest_Low_5_State' as Type
FROM Table_1
WHERE l_rnk <=5
ORDER BY l_rnk)

ORDER BY AVG_Delivery_Time desc ;
```

Row	customer_state	AVG_Delivery_Time	Type
1	RR	27.83	Highest_Top_5_State
2	AP	27.75	Highest_Top_5_State
3	AM	25.96	Highest_Top_5_State
4	AL	23.99	Highest_Top_5_State
5	PA	23.3	Highest_Top_5_State
6	SC	14.52	Lowest_Low_5_State
7	DF	12.5	Lowest_Low_5_State
8	MG	11.52	Lowest_Low_5_State
9	PR	11.48	Lowest_Low_5_State
10	SP	8.26	Lowest_Low_5_State

Query:

```
With Table_1 as

  (SELECT c.customer_state,
         ROUND(AVG(Timestamp_diff(o.order_delivered_customer_date, o.order_purchase_timestamp,day)),2) as
AVG_Delivery_Time,
          dense_rank() over(order by
AVG(Timestamp_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)) desc) as h_rnk,
          dense_rank() over(order by
AVG(Timestamp_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)) asc) as l_rnk

FROM `Target_Dataset.orders` o

INNER JOIN `Target_Dataset.customers`c
ON o.customer_id = c.customer_id

INNER JOIN `Target_Dataset.order_items`oi
ON o.order_id =  oi.order_id

GROUP BY 1)

(SELECT customer_state,AVG_Delivery_Time,
'Highest_Top_5_State' as Type
FROM Table_1
WHERE h_rnk <=5
ORDER BY h_rnk)

UNION ALL

(SELECT customer_state,AVG_Delivery_Time,
'Lowest_Low_5_State' as Type
```

```
FROM Table_1
WHERE l_rnk <=5
ORDER BY l_rnk)

ORDER BY AVG_Delivery_Time desc ;
```

Insights: By above Output we can see the Top 5 high states where average delivery time are more & 5 Low states where average delivery time are less, This could happened because customers location distance might be too far or short from Target warehouses location.

Recommendations: Target can open warehouses in a state where delivery time are much high ,so that time can be saved or more delivery can be done on any other states

5.4 Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
With Table_1 as
  (SELECT c.customer_state,
          ROUND(AVG(Timestamp_diff(o.order_delivered_customer_date,order_estimated_delivery_date,day )),2) as Diff_Estimated_Delivery_date,
          dense_rank() over(order by ROUND(AVG(Timestamp_diff(o.order_delivered_customer_date,order_estimated_delivery_date,day)),2) ) as
h_rnk
FROM `Target_Dataset.orders`o

INNER JOIN `Target_Dataset.customers`c
ON o.customer_id = c.customer_id

INNER JOIN `Target_Dataset.order_items`oi
ON o.order_id =  oi.order_id

WHERE o.order_delivered_customer_date IS NOT NULL
AND o.order_status = 'delivered'
GROUP BY 1)

SELECT customer_state,
       Diff_Estimated_Delivery_date,
FROM Table_1
WHERE h_rnk <= 5
ORDER BY h_rnk
```

Row	customer_state	Diff_Estimated_Deliv
1	AC	-20.01
2	RO	-19.08
3	AM	-18.98
4	AP	-17.44
5	RR	-17.43

Query :

```
With Table_1 as
  (SELECT c.customer_state,
          ROUND(AVG(Timestamp_diff(o.order_delivered_customer_date,order_estimated_delivery_date,day
)),2) as Diff_Estimated_Delivery_date,
          dense_rank() over(order by
ROUND(AVG(Timestamp_diff(o.order_delivered_customer_date,order_estimated_delivery_date,day)),2) ) as
h_rnk
FROM `Target_Dataset.orders`o

INNER JOIN `Target_Dataset.customers`c
ON o.customer_id = c.customer_id

INNER JOIN `Target_Dataset.order_items`oi
ON o.order_id =  oi.order_id
```

```
WHERE o.order_delivered_customer_date IS NOT NULL
AND o.order_status = 'delivered'
GROUP BY 1)

SELECT customer_state,
       Diff_Estimated_Delivery_date,
FROM Table_1
WHERE h_rnk <= 5
ORDER BY h_rnk
```

Insights : From the above Output we can say that estimation of actual date of delivery is not calculated in a proper way or product's delivery might be early of any other reasons

Recommendations : Target can calculate the days of distance of delivery in a proper way, So that customers trust can be achieved.

6. Analysis based on the payments:

6. 1 Find the month on month no. of orders placed using different payment types.

```
SELECT
       EXTRACT(Year from o.order_purchase_timestamp) as Year,
       EXTRACT(month from o.order_purchase_timestamp) as Month,
       p.payment_type,
       COUNT(DISTINCT(o.order_id)) as No_of_order
FROM `Target_Dataset.orders`o
JOIN  `Target_Dataset.payments`p
ON o.order_id = p.order_id
GROUP BY 1,2,3
ORDER BY 1 asc,
         2 asc,
         3 asc;
```

Row	Year	Month	payment_type	No_of_order
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	253
4	2016	10	debit_card	2
5	2016	10	voucher	11
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	582
9	2017	1	debit_card	9
10	2017	1	voucher	33

Query : SELECT

```
        EXTRACT(Year from o.order_purchase_timestamp) as Year,
        EXTRACT(month from o.order_purchase_timestamp) as Month,
        p.payment_type,
        COUNT(DISTINCT(o.order_id)) as No_of_order
FROM `Target_Dataset.orders`o
JOIN  `Target_Dataset.payments`p
ON o.order_id = p.order_id
GROUP BY 1,2,3
ORDER BY 1 asc,
         2 asc,
         3 asc;
```

Insights: From above Output we can see different type of payment methods customers are using for doing the payment of product.

Recommendation: Target can enable others payments option like NEFT, UPI, QR scanner etc for making payment transactions easy to Customers .

6.2 Find the no. of orders placed on the basis of the payment installments that have been paid.

```
SELECT  p.payment_installments,
        COUNT(DISTINCT(o.order_id)) as No_of_order
FROM `Target_Dataset.orders`o
JOIN  `Target_Dataset.payments`p
ON o.order_id = p.order_id
WHERE o.order_status!= 'canceled'
GROUP BY 1
ORDER BY 1 asc,
         2 asc;
```

Row	payment_installment	No_of_order
1	0	2
2	1	48732
3	2	12329
4	3	10374
5	4	7046
6	5	5204
7	6	3894
8	7	1617
9	8	4224
10	9	638
11	10	5279

Query :

```
SELECT  p.payment_installments,
        COUNT(DISTINCT(o.order_id)) as No_of_order
FROM `Target_Dataset.orders`o
```

```
JOIN  `Target_Dataset.payments`p
ON o.order_id = p.order_id
WHERE o.order_status!= 'canceled'
GROUP BY 1
ORDER BY 1 asc,
         2 asc;
```

Insights: **From the above Output we can see that customers have choosed different Installments options like yearly, monthly, quarterly & doing their payments accordingly.**

Recommendations : **Target can ace those customers who choose the options on yearly basis & according launch their schemes , So that customers would attract and purchased more product.**