# TARGET - BUSINESS CASE STUDY

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

1. **Data type of all columns in the "customers" table.**

   ANS:-
   ```
   SELECT *
   FROM `nov23batchproject-406105.Target.INFORMATION_SCHEMA.COLUMNS`
   WHERE Table_name = 'customers';
   ```

   **OUTPUT:-**

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

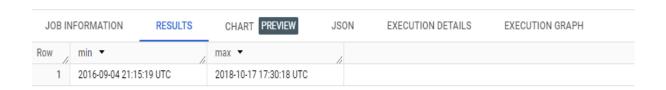| Row | table_name ▼ | column_name ▼ | ordinal_position ▼ | is_nullable ▼ | data_type ▼ |
|---|---|---|---|---|---|
| 1 | customers | customer_id | 1 | YES | STRING |
| 2 | customers | customer_unique_id | 2 | YES | STRING |
| 3 | customers | customer_zip_code_prefix | 3 | YES | INT64 |
| 4 | customers | customer_city | 4 | YES | STRING |
| 5 | customers | customer_state | 5 | YES | STRING |

   **INSIGHTS:-**

   - In this output ,there are five rows in that four rows are "String" Data type except customer_zip_code_prefix which is "INT64" .

   **2.. Get the time range between which the orders were placed.**
   ANS:-
   ```
   SELECT min(order_purchase_timestamp) AS `min`
   ,MAX(order_purchase_timestamp) AS `max`
   FROM `Target.orders`
   ```

   **OUTPUT:-**

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

| Row | min ▼ | max ▼ |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-10-17 17:30:18 UTC |

**3. Count the Cities & States of customers who ordered during the given period.**
ANS:-
```sql
SELECT count(DISTINCT C.customer_city) AS CITY,count(DISTINCT
C.customer_state) AS STATES
FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.customer_id
WHERE  O.order_purchase_timestamp BETWEEN "2016-01-01" AND "2018-12-
31";
```

**OUTPUT:-**

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

| Row | CITY ▾ | STATES ▾ | |
|---|---|---|---|
| 1 | 4119 | 27 | |

**INSIGHTS:-**

- People from Brazil order from 4119 cities and 27 states during 2016-2018.

## 2.In-depth Exploration:-

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

    ANS:-
    ```
    SELECT COUNT(order_id) AS `no_of_orders`,EXTRACT (YEAR FROM
    order_purchase_timestamp) AS `order_year`,
    FROM `Target.orders`
    GROUP BY order_year
    ORDER BY order_year ASC;
    ```

    ## OUTPUT:-

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

    | Row | no_of_orders ▼ | order_year ▼ |
    |---|---|---|
    | 1 | 329 | 2016 |
    | 2 | 45101 | 2017 |
    | 3 | 54011 | 2018 |

    ## INSIGHTS:-

    - There is the gradual increased number of orders from 2016-2018.
    - Comparing the all three years in year 2016 there are only 329 orders which is less than 1000.
    - In 2017, there are 45101 orders which very high compare to the year 2016.
    - In 2018, there are 54011 orders where almost increased near to 10000 orders compare to 2017.

    ## RECOMMENDATIONS:-

    - In 2016 , If Target runs any form of discounts and marketing through advertisements, social media, and digital marketing there may be chance of increasing the number of orders.

    2. **Can we see some kind of monthly seasonality in terms of the no. of orders being placed?**
    ANS:-
    ```
    SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS `month`,
    COUNT(order_id) AS `no_of_orders`
    FROM `Target.orders`
    GROUP BY month
    ORDER BY month;
    ```

## OUTPUT : -

| Row | month | no_of_orders |
|-----|-------|--------------|
| 1 | 1 | 8069 |
| 2 | 2 | 8508 |
| 3 | 3 | 9893 |
| 4 | 4 | 9343 |
| 5 | 5 | 10573 |
| 6 | 6 | 9412 |
| 7 | 7 | 10318 |
| 8 | 8 | 10843 |
| 9 | 9 | 4305 |
| 10 | 10 | 4959 |

Results per page:  50  Activate Windo 1 – 12 of 12
Go to PC settings to

| 11 | 11 | 7544 |
|-----|-----|------|
| 12 | 12 | 5674 |

## INSIGHTS:-

- In the month of May , July and August there are high in sales compared to other months which are 10000 above orders.
- The highest number of orders placed in the month of August i.e. 10843 orders when compared to other months.
- The lowest number of orders placed in the month of September and October i.e. 4305 and 4959 orders which is before the August.
- In the month of January and February orders there are only 439 difference
  January – 8069
  February – 8508
- In the month of March, April and June orders there is a decent amount of orders placed.
  March – 9893
  April – 9343
  June - 9412
- In the month of May , July and August there are high in sales compared to other months which are 10000 above orders.
- There was a medium sales in the month of November and December

## Recommendations:-

- The highest and lowest sales happened in followed by months only on observing that if we run any campaign or discounts at that particular time there might be chance of increasing sales till medium level.
- December is the Christmas Eve month many of them show interest in buying new things like Electronic appliances . Making this an advantage we should give 0 down payment and Emi options and people may buy new electronics .
- And also give the discount for the products which are for long period in the store by giving clearance sale in the month of December.

**3**. **During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)**
- o 0-6 hrs : Dawn
- o 7-12 hrs : Mornings
- o 13-18 hrs : Afternoon
- o 19-23 hrs : Night

ANS:-

```sql
SELECT
    CASE
        WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 0
AND 6 THEN 'Dawn'
        WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 7
AND 12 THEN 'Morning'
        WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 13
AND 18 THEN 'Afternoon'
        WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 19
AND 23 THEN 'Night'
        ELSE 'Unknown'
    END AS time_of_day,
    COUNT(order_id) AS order_count
FROM
    `Target.orders`
GROUP BY
    time_of_day
ORDER BY
    order_count;
```

## OUTPUT : -

| Row | time_of_day ▼ | order_count ▼ |
|-----|---------------|---------------|
| 1 | Dawn | 5242 |
| 2 | Morning | 27733 |
| 3 | Night | 28331 |
| 4 | Afternoon | 38135 |

## INSIGHTS:-

- There is a low order count in the dawn and high order count in Aftenoon.
- There is a Medium order count in Morning and Night when compared to Night.
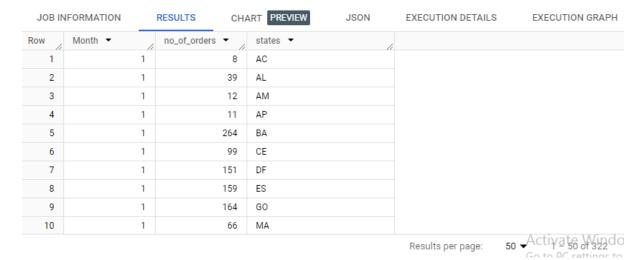
## Recommendations:-

- Put a timer discount for the premium members. So, that they can buy more orders at dawn.

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

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

ANS:-

```sql
SELECT EXTRACT (MONTH FROM order_purchase_timestamp) AS `Month` ,
COUNT(O.order_id) AS `no_of_orders` , C.customer_state AS `states`
FROM `Target.orders` AS O LEFT JOIN `Target.customers` AS C ON
O.customer_id = C.customer_id
GROUP BY Month, states
ORDER BY Month,states ;
```

## OUTPUT:-

| Row | Month ▾ | no_of_orders ▾ | states ▾ |
| --- | --- | --- | --- |
| 1 | 1 | 8 | AC |
| 2 | 1 | 39 | AL |
| 3 | 1 | 12 | AM |
| 4 | 1 | 11 | AP |
| 5 | 1 | 264 | BA |
| 6 | 1 | 99 | CE |
| 7 | 1 | 151 | DF |
| 8 | 1 | 159 | ES |
| 9 | 1 | 164 | GO |
| 10 | 1 | 66 | MA |

Results per page:  50 ▾   1 – 50 of 322

Activate Windo
Go to PC settings to

## INSIGHTS:-

- SP state has high number of orders in all the months.
- RR state has low number of orders in the 1st month.

2. **How are the customers distributed across all the states?**
ANS:-

```
SELECT customer_id , customer_zip_code_prefix, customer_state
FROM `Target.customers`
ORDER by customer_state;
```

## OUTPUT:-

## INSIGHTS:-

- There are 99441 rows and extracted the orders from each zip code from Brazil.

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

1. **Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).**
   **You can use the "payment_value" column in the payments table to get the cost of orders.**
   ANS:-

```
SELECT
    ((SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp ) =
2018 AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND
8 THEN payment_value ELSE 0 END) -
     SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp) =
2017 AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND
8 THEN payment_value ELSE 0 END))
    /
     SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp) =
2017 AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND
8 THEN payment_value ELSE 0 END)
    ) * 100 AS percentage_increase
FROM `Target.payments` AS P  INNER JOIN `Target.orders` AS O ON
P.order_id = O.order_id
WHERE EXTRACT(YEAR FROM order_purchase_timestamp) IN (2017, 2018);
```

## OUTPUT:-

| Row | percentage_increase |
|---|---|
| 1 | 136.9768716466... |

## INSIGHTS:-

- 136.9768716466 percentage increased in the cost of orders from year 2017 to 2018 between January to August only.

## 2.Calculate the Total & Average value of order price for each state.
ANS:-
```
SELECT C.customer_state, ROUND (SUM(OD.price),2) AS `Total_value`,
ROUND (AVG(OD.price),2) AS `Average_value`,
FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.customer_id
INNER JOIN `Target.order_items` AS OD ON O.order_id = OD.order_id
GROUP BY C.customer_state
ORDER BY C.customer_state;
```

## OUTPUT:-

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

| Row | customer_state ▼ | Total_value ▼ | Average_value ▼ |
|---|---|---|---|
| 1 | AC | 15982.95 | 173.73 |
| 2 | AL | 80314.81 | 180.89 |
| 3 | AM | 22356.84 | 135.5 |
| 4 | AP | 13474.3 | 164.32 |
| 5 | BA | 511349.99 | 134.6 |
| 6 | CE | 227254.71 | 153.76 |
| 7 | DF | 302603.94 | 125.77 |
| 8 | ES | 275037.31 | 121.91 |
| 9 | GO | 294591.95 | 126.27 |
| 10 | MA | 119648.22 | 145.2 |

Results per page: 50 ▼ Activate Windows 1-27 of 27 |<

## INSIGHTS:-

- There are total 27 rows present in this output.
- SP state has the highest total value of comparing all other states.
- RR state has the lowest total value of comparing all other states.
- PB state has the highest average value of comparing all other states.
- SP state has the lowest average value of comparing all other states.

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

**ANS:-**

```sql
SELECT C.customer_state, ROUND (SUM(freight_value),2) AS
`Total_Freight_value`, ROUND (AVG(freight_value),2) AS
`Average_Freight_value`,
FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.customer_id
INNER JOIN `Target.order_items` AS OD ON O.order_id = OD.order_id
GROUP BY C.customer_state
ORDER BY C.customer_state;
```

**OUTPUT:-**

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

| Row | customer_state ▼ | Total_Freight_value | Average_Freight_valu |
|---|---|---|---|
| 1 | AC | 3686.75 | 40.07 |
| 2 | AL | 15914.59 | 35.84 |
| 3 | AM | 5478.89 | 33.21 |
| 4 | AP | 2788.5 | 34.01 |
| 5 | BA | 100156.68 | 26.36 |
| 6 | CE | 48351.59 | 32.71 |
| 7 | DF | 50625.5 | 21.04 |
| 8 | ES | 49764.6 | 22.06 |
| 9 | GO | 53114.98 | 22.77 |
| 10 | MA | 31523.77 | 38.26 |

Results per page: 50 ▼   1 – 27 of 27

**INSIGHTS:-**

- There are total 27 rows present in this output.
- RJ  state has the highest  total frieght value of comparing all other states.
- AP state has the lowest  total freight value of comparing all other states.
- RR state has the highest  average freight value of comparing all other states.
- SP  state has the lowest average freight value of comparing all other states.

**5.Analysis based on sales, freight and delivery time.**

**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**

ANS:-
```sql
SELECT 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
`difference_in_estimated_delivery`
FROM `Target.orders`
```

## OUTPUT:-

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

| Row | order_id ▾ | DELIVERY_TIME ▾ | difference_in_estima |
|---|---|---|---|
| 1 | 1950d777989f6a877539f5379... | 30 | 12 |
| 2 | 2c45c33d2f9cb8ff8b1c86cc28... | 30 | -28 |
| 3 | 65d1e226dfaeb8cdc42f66542... | 35 | -16 |
| 4 | 635c894d068ac37e6e03dc54e... | 30 | -1 |
| 5 | 3b97562c3aee8bdedcb5c2e45... | 32 | 0 |
| 6 | 68f47f50f04c4cb6774570cfde... | 29 | -1 |
| 7 | 276e9ec344d3bf029ff83a161c... | 43 | 4 |
| 8 | 54e1a3c2b97fb0809da548a59... | 40 | 4 |
| 9 | fd04fa4105ee8045f6a0139ca5... | 37 | 1 |
| 10 | 302bb8109d097a9fc6e9cefc5... | 33 | 5 |

Results per page: 50 ▾    Activate Windo
1 – 50 of 99441

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

```sql
WITH Maxm AS (
   SELECT  C.customer_state AS `AVG_MAX_STATES`, ROUND
(AVG(D.freight_value)) AS `AVG_MAX_OF_FREIGHT_VALUES`
   FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.Customer_id
   INNER JOIN `Target.order_items` AS D ON O.order_id = D.order_id
   GROUP BY AVG_MAX_STATES
   ORDER  BY AVG_MAX_OF_FREIGHT_VALUES DESC
)

Sa AS (
SELECT  C.customer_state AS `AVG_Min_STATES`, ROUND
(AVG(D.freight_value)) AS `AVG_MIN_OF_FREIGHT_VALUES`
   FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.Customer_id
   INNER JOIN `Target.order_items` AS D ON O.order_id = D.order_id
   GROUP BY AVG_Min_STATES
   ORDER  BY AVG_MIN_OF_FREIGHT_VALUES ASC
)
SELECT AVG_MAX_STATES, AVG_MAX_OF_FREIGHT_VALUES, AVG_Min_STATES,
AVG_MIN_OF_FREIGHT_VALUES
FROM Maxm, Sa
LIMIT 5;
```

**OUTPUT:-**

| Row | AVG_MAX_STATES ▼ | AVG_MAX_OF_FREIG | AVG_Min_STATES ▼ | AVG_MIN_OF_FREIG |
|-----|------------------|------------------|------------------|------------------|
| 1 | AL | 36.0 | MA | 38.0 |
| 2 | AL | 36.0 | MT | 28.0 |
| 3 | AL | 36.0 | MG | 21.0 |
| 4 | AL | 36.0 | AL | 36.0 |
| 5 | AL | 36.0 | SP | 15.0 |

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

```sql
WITH HIGH AS(
```

```sql
SELECT C.customer_state AS `H_STATE`,
ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_tim
estamp,DAY)),2) AS `HIGHEST_AVERAGE_DELIVERY`
FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.customer_id
GROUP BY  C.customer_state
ORDER BY C.customer_state DESC
),
LOW AS(
  SELECT C.customer_state AS
`L_STATE`,ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_pu
rchase_timestamp,DAY)),2) AS `LOWEST_AVERAGE_DELIVERY`
FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.customer_id
GROUP BY  C.customer_state
ORDER BY C.customer_state ASC
)
SELECT H_STATE, HIGHEST_AVERAGE_DELIVERY, L_STATE,
LOWEST_AVERAGE_DELIVERY
FROM HIGH, LOW
LIMIT 5;
```

**OUTPUT:-**

| Row | H_STATE ▼ | HIGHEST_AVERAGE | L_STATE ▼ | LOWEST_AVERAGE_I |
|-----|-----------|-----------------|-----------|------------------|
| 1 | DF | 12.51 | RS | 14.82 |
| 2 | DF | 12.51 | SP | 8.3 |
| 3 | DF | 12.51 | RJ | 14.85 |
| 4 | DF | 12.51 | DF | 12.51 |
| 5 | DF | 12.51 | PR | 11.53 |

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

**You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.**

```sql
SELECT
C.customer_state,AVG(DATE_DIFF(order_delivered_customer_date,order_e
stimated_delivery_date,DAY)) AS `delivery_state_fast`
FROM `Target.customers` AS C INNER JOIN `Target.orders` AS O ON
C.customer_id = O.customer_id
GROUP BY  C.customer_state
ORDER BY C.customer_state DESC
LIMIT 5;
```

| Row | customer_state ▼ | delivery_state_fast |
|---|---|---|
| 1 | TO | -11.2591240875... |
| 2 | SP | -10.1353253488... |
| 3 | SE | -9.17313432835... |
| 4 | SC | -10.6058641105... |
| 5 | RS | -12.9818488023... |

## 6.Analysis based on the payments:

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

```sql
SELECT EXTRACT (MONTH FROM O.order_purchase_timestamp) AS `Month` ,
COUNT(O.order_id) AS `no_of_orders`,P.payment_type
FROM `Target.orders` AS O INNER JOIN `Target.payments` AS P On
O.order_id= P.order_id
GROUP BY Month, P.payment_type
Order BY Month;
```

**OUTPUT:-**

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

| Row | Month ▼ | no_of_orders ▼ | payment_type ▼ |
|---|---|---|---|
| 1 | 1 | 6103 | credit_card |
| 2 | 1 | 1715 | UPI |
| 3 | 1 | 477 | voucher |
| 4 | 1 | 118 | debit_card |
| 5 | 2 | 1723 | UPI |
| 6 | 2 | 6609 | credit_card |
| 7 | 2 | 424 | voucher |
| 8 | 2 | 82 | debit_card |
| 9 | 3 | 7707 | credit_card |
| 10 | 3 | 1942 | UPI |

Results per page: 50 ▼  Activate Windo
1 – 50 of 50

## INSIGHTS:-

- In every month credit card payments orders are high when compared other modes of payments.
- In every month debit card payments orders are low when compared other modes of payments.

## RECOMMENDATIONS:-

- If we give credit card offers for some particular amount then there is a chance of getting more orders.
- And for debit cards if we give some cash back while ordering then there is a chance of getting more orders.

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

```sql
SELECT COUNT(O.order_id) AS `no_of_orders`,P.payment_sequential
FROM `Target.orders` AS O RIGHT JOIN `Target.payments` AS P On
O.order_id= P.order_id
GROUP BY P.payment_sequential;
```

## OUTPUT:-

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

| Row | no_of_orders ▾ | payment_sequential |
|---|---|---|
| 1 | 3039 | 2 |
| 2 | 278 | 4 |
| 3 | 10 | 14 |
| 4 | 99360 | 1 |
| 5 | 581 | 3 |
| 6 | 13 | 13 |
| 7 | 34 | 10 |
| 8 | 170 | 5 |
| 9 | 118 | 6 |
| 10 | 29 | 11 |

Results per page: 50 ▾   1 – 29 of 29

Activate Windows