Sankaran Meenakshi Sundaram

Target SQL:

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

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

Code:

```
SELECT column_name, data_type
FROM `businesscase-target-sql.Ecommerce.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers'
```

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

OBSERVATION:

The customer table contain customer_id, customer_unique_id, customer_zip_code_prefix, customer_city and customer_state as columns and STRING, STRING, INTEGER, STRING and STRING as datetype respectively.

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

Code:

```
SELECT MIN(order_purchase_timestamp) AS first_date,
       MAX(order_purchase_timestamp) AS final_date
FROM `Ecommerce.orders`
```

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

OBSERVATION:

During the Operation of Target in Brazil , the orders were placed between 2016 September 04, 21:15 hrs to 2018 October 17,17:30 hrs

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

Code:

```
SELECT COUNT(DISTINCT C.customer_city) AS No_of_City,
       COUNT(DISTINCT C.customer_state) AS No_of_State
FROM `Ecommerce.customers` C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
```

Row	No_of_City	No_of_State
1	4119	27

OBSERVATION:

The customers for Target in Brazil are from 4119 Cities and 27 states during the given period.

2. In-depth Exploration:

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

Code:

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS Year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
COUNT(order_id) AS orders_per_month
FROM `Ecommerce.orders`
GROUP BY 1,2
ORDER BY 1,2
```

Row	Year	Month	orders_per_month
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026

OBSERVATION:

The Operation of Target in Brazil is between 2016 September and 2018 October. During the first three months the orders are not in trend and then it started in an increasing trend till 2017 November and then it was hovering around there till 2018 August and it drastically reduces at the end of operation.

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

Code:

```
SELECT *, DENSE_RANK() OVER(ORDER BY orders_per_month DESC) AS rank
FROM
(SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
        COUNT(DISTINCT order_id) AS orders_per_month
FROM `Ecommerce.orders`
GROUP BY 1) T
ORDER BY rank
```

Row	Month	orders_per_month	rank
1	8	10843	1
2	5	10573	2
3	7	10318	3
4	3	9893	4
5	6	9412	5
6	4	9343	6
7	2	8508	7
8	1	8069	8
9	11	7544	9
10	12	5674	10
11	10	4959	11

OBSERVATION:

The top three positions based on the number of orders per month are during Auguust, May and July may be because of festive seasons and summer.

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

Code:

```
WITH CTE AS
(SELECT Hours, CASE
WHEN Hours BETWEEN 0 AND 6 THEN 'Dawn: 0-6 hrs'
WHEN Hours BETWEEN 7 AND 12 THEN 'Mornings: 7-12 hrs'
WHEN Hours BETWEEN 13 AND 18 THEN 'Afternoon: 13-18 hrs'
```

```
WHEN Hours BETWEEN 19 AND 23 THEN 'Night: 19-23 hrs' END AS Hours_Range,
No_of_orders
FROM
(SELECT EXTRACT(HOUR FROM order_purchase_timestamp) AS Hours,
        COUNT(order_id) AS No_of_orders
FROM `Ecommerce.orders`
GROUP BY 1
ORDER BY 1) T)
SELECT Hours_Range,SUM(No_of_orders) AS Total_Orders
FROM CTE
GROUP BY Hours_Range
ORDER BY 2 DESC
```

Row	Hours_Range	Total_Orders
1	Afternoon: 13-18 hrs	38135
2	Night: 19-23 hrs	28331
3	Mornings: 7-12 hrs	27733
4	Dawn: 0-6 hrs	5242

OBSERVATION:

Based on the four types of hours range the Brazilian customers placed most of the orders during Afternoon: 13-18 hrs. This may be because most of the customer are available for shopping at Afternoon. This would be the best time to introduce some new product and find its movement and collect more reviews.

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

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

Code:

```
SELECT EXTRACT(YEAR FROM O.order_purchase_timestamp) AS Year,
EXTRACT(MONTH FROM O.order_purchase_timestamp) AS Month,
C.customer_state,
COUNT(O.order_id) AS Total_Orders
FROM `Ecommerce.customers` C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
GROUP BY 1,2,3
ORDER BY 1,2
```

Row	Year	Month	customer_state	Total_Orders
1	2016	9	RR	1
2	2016	9	RS	1
3	2016	9	SP	2
4	2016	10	SP	113
5	2016	10	RS	24
6	2016	10	RJ	56
7	2016	10	MT	3
8	2016	10	GO	9
9	2016	10	MG	40
10	2016	10	CE	8

OBSERVATION:

The Total order column gives the month on month, number of orders placed on each state.

3.2 How are the customers distributed across all the states?

Code:

```
SELECT customer_state,
COUNT(customer_unique_id) AS customer_count
FROM `Ecommerce.customers`
GROUP BY 1
ORDER BY 2 DESC
```

Row	customer_state	customer_count
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

OBSERVATION:

The number of customers from highest to lowest on each state are shown above. We can see the State SP has the highest customers and the state RR has the lowest customer.

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). You can use the "payment_value" column in the payments table to get the cost of orders.

Code:

```
SELECT *,
ROUND(((T1.Payment_value_2018-
T1.Payment_value_2017)/T1.Payment_value_2018)*100,2) AS Percentage_increase
FROM
(WITH CTE AS
(SELECT Year, Month, payment_value_2017
FROM
(SELECT EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month,
ROUND(SUM(p.payment_value),2) AS Payment_value_2017
FROM `Ecommerce.orders` o
INNER JOIN `Ecommerce.payments` p
ON o.order_id = p.order_id
GROUP BY 1,2
ORDER BY 1,2) T
WHERE (Year = 2017 AND Month BETWEEN 1 AND 8) OR (Year = 2018 AND Month BETWEEN
1 AND 8)
ORDER BY Year, Month)
SELECT
Month, Payment_value_2017,
LEAD(Payment_value_2017,8) OVER(ORDER BY Year, Month) AS Payment_value_2018,
FROM CTE
ORDER BY Year,Month
LIMIT 8) T1
```

Row	Month	Payment_value_2017	Payment_value_2018	Percentage_increase
1	1	138488.04	1115004.18	87.58
2	2	291908.01	992463.34	70.59
3	3	449863.6	1159652.12	61.21
4	4	417788.03	1160785.48	64.01
5	5	592918.82	1153982.15	48.62
6	6	511276.38	1023880.5	50.06
7	7	592382.92	1066540.75	44.46
8	8	674396.32	1022425.32	34.04

OBSERVATION:

The Percentage increase in the cost of order from the year 2017 to 2018 for the months January to August are shown above. The highest percentage increase is in the month of January and the lowest percentage is in the month of August.

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

Code:

```
SELECT C.customer_state,
       ROUND(SUM(OI.price),2) AS Total_Order_Value,
       ROUND(AVG(OI.price),2) AS Avg_Order_Value
FROM `Ecommerce.customers` C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
INNER JOIN `Ecommerce.order_items` OI
ON O.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY Total_Order_Value DESC
```

Row	customer_state	Total_Order_Value	Avg_Order_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

OBSERVATION:

The Total and Average order value of order price on each state are shown above, we can see that the State SP has highest total order price followed by RJ, MG and so on and the state RR has the lowest order price.

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

Code:

```
SELECT C.customer_state,
ROUND(SUM(OI.freight_value),2) AS Total_freight,
ROUND(AVG(OI.freight_value),2) AS Avg_freight
```

```
FROM `Ecommerce.customers` C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
INNER JOIN `Ecommerce.order_items` OI
ON O.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY Total_freight DESC
```

Row	customer_state	Total_freight	Avg_freight
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04

OBSERVATION:

The Total and Average value of order freight on each state are shown above, we can see that the State SP has highest total freight value followed by RJ, MG and so on and the state RR has the lowest total freight valse.

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_estimated_delivery_date - order_delivered_customer_date**

Code:

```
SELECT order_id,
DATETIME_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY) AS
time_to_deliver,
DATETIME_DIFF(order_estimated_delivery_date,  order_delivered_customer_date,
DAY) AS diff_estimated_delivery
FROM `Ecommerce.orders`
WHERE order_delivered_customer_date IS NOT NULL
```

Row	order_id	time_to_deliver	diff_estimated_delivery
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

OBSERVATION:

The above table shows the time to deliver each order in days and difference between estimated delivery and order purchase date in days, The negative values in diff_estimated_delivery column indicates the delivery date is higher than the estimate date that is late delivery.

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

Top 5 states with highest average freight value.

Code:

```
SELECT C.customer_state,
ROUND(AVG(OI.freight_value),2) AS Avg_freight,
FROM `Ecommerce.customers` C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
INNER JOIN `Ecommerce.order_items` OI
ON O.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY Avg_freight DESC
LIMIT 5
```

Row	customer_state	Avg_freight
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

OBSERVATION:

Top 5 states with highest average freight value are shown above. We see that the state RR has the highest average freight value and followed by PB, RO, AC and PI

Top 5 states with lowest average freight value.

Code:

```
SELECT C.customer_state,
ROUND(AVG(OI.freight_value),2) AS Avg_freight,
FROM `Ecommerce.customers` C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
INNER JOIN `Ecommerce.order_items` OI
ON O.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY Avg_freight
LIMIT 5
```

Row	customer_state	Avg_freight
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

OBSERVATION:

Top 5 states with lowest average freight value are shown above. We see that the state SP has the lowest average freight value and followed by PR, MG, RJ and DF.

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

Top 5 states with highest average delivery time.

Code:

```
SELECT C.customer_state,
ROUND(AVG(DATETIME_DIFF(O.order_delivered_customer_date,O.order_purchase_tim
estamp, DAY)),0) AS avg_delivery_day,
FROM `Ecommerce.customers`C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
WHERE O.order_delivered_customer_date IS NOT NULL
GROUP BY C.customer_state
ORDER BY avg_delivery_day DESC
LIMIT 5
```

Row	customer_state	avg_delivery_day
1	RR	29.0
2	AP	27.0
3	AM	26.0
4	AL	24.0
5	PA	23.0

OBSERVATION:

Top 5 states with highest average delivery time in days are shown above. The state RR has highest average delivery time followed by AP, AM, AL and PA.

Top 5 states with lowest average delivery time.

Code:

```
SELECT C.customer_state,
ROUND(AVG(DATETIME_DIFF(O.order_delivered_customer_date,O.order_purchase_tim
estamp, DAY)),0) AS avg_delivery_day,
FROM `Ecommerce.customers`C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
WHERE O.order_delivered_customer_date IS NOT NULL
GROUP BY C.customer_state
ORDER BY avg_delivery_day, customer_state
LIMIT 5
```

Row	customer_state	avg_delivery_day
1	SP	8.0
2	MG	12.0
3	PR	12.0
4	DF	13.0
5	SC	14.0

OBSERVATION:

Top 5 states with lowest average delivery time in days are shown above. The state SP has lowest average delivery time followed by MG, PR, DF and SC.

5.4 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.

Code:

```
SELECT C.customer_state,
ROUND(AVG(DATETIME_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY)),0) AS Avg_fast_delivery
FROM `Ecommerce.customers`C
INNER JOIN `Ecommerce.orders`O
ON C.customer_id = O.customer_id
WHERE O.order_delivered_customer_date IS NOT NULL
GROUP BY C.customer_state
ORDER BY Avg_fast_delivery DESC
LIMIT 5
```

Row	customer_state	Avg_fast_delivery
1	AC	20.0
2	AP	19.0
3	AM	19.0
4	RO	19.0
5	RR	16.0

OBSERVATION:

Top 5 states with fastest average delivery time in days are shown above. The state AC has fastest delivery of orders followed by AP, AM, RO and RR.

6. Analysis based on the payments:

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

Code:

```
SELECT EXTRACT(YEAR FROM O.order_purchase_timestamp) AS Year,
EXTRACT(MONTH FROM O.order_purchase_timestamp) AS Month,
P.payment_type,
COUNT(O.order_id) AS No_of_orders
FROM `Ecommerce.orders` O
INNER JOIN `Ecommerce.payments` P
ON O.order_id = P.order_id
GROUP BY 1,2,3
ORDER BY 1,2,4 DESC
```

Row	Year	Month	payment_type	No_of_orders
1	2016	9	credit_card	3
2	2016	10	credit_card	254
3	2016	10	UPI	63
4	2016	10	voucher	23
5	2016	10	debit_card	2
6	2016	12	credit_card	1
7	2017	1	credit_card	583
8	2017	1	UPI	197
9	2017	1	voucher	61
10	2017	1	debit_card	9

OBSERVATION:

The number of orders placed month on month using Credit card, UPI, Voucher and Debit card are shown above. In 2016 September and 2017 January only credit card is used for payments and in all the other months Credit card, UPI, Voucher and Debit card are used. The maximum payments are done using credit card.

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

Code:

```
SELECT payment_installments, COUNT(order_id) No_of_orders,
ROUND(SUM(payment_value),2) AS Total_paymants
FROM `Ecommerce.payments`
GROUP BY payment_installments
ORDER BY No_of_orders DESC
```

Row	payment_installments	No_of_orders	Total_paymants
1	1	52546	5907233.36
2	2	12413	1579283.03
3	3	10461	1491103.8
4	4	7098	1163907.61
5	10	5328	2211577.34
6	5	5239	961174.3
7	8	4268	1313423.34
8	6	3920	822611.81
9	7	1626	305157.39
10	9	644	131015.92

OBSERVATION:

From the above output table, we can see that there are 24 unique instalments starting from 0 to 24. The maximum orders are placed based on 1 instalment. There is only one order which is placed on 22 and 23 instalments. There are 2 orders placed on 0 instalments. The maximum number of customers prefer 1 instalment.

OVERALL INSIGHTS:

- ✓ The Operation of Target in Brazil was for 25 months. During the first 3 months the orders are not in trend, for the next 11 months the order flow was trending, for the next 9 months the orders was around same level and for the last 2 months the order completely low.
- ✓ The top three performing months based on the orders are Auguust, May and July.
- ✓ The Brazilian customers placed most of the orders during Afternoon: 13-18 hrs.
- ✓ The number of customers is high in the state SP and low in the state RR.
- ✓ The highest percentage increase in the cost of orders is in the month of January and the lowest percentage is in the month of August.
- ✓ The total order price is high in the state SP and low in the state RR.
- ✓ The total value of order freight is high in the state SP and low in the state RR.
- ✓ The highest average freight value is in the state RR.

- ✓ The lowest average freight value is in the state SP.
- ✓ The average delivery time is high for the state RR.
- ✓ The average delivery time is low for the state SP.
- ✓ The state AC has the fast delivery when compared to the estimated delivery.
- ✓ The maximum payments are done using credit card.
- ✓ The maximum number of customers prefer 1 instalment.

RECOMMENDATIONS:

- ✓ The state SP is very important for the Target since the business is good so the Target could perform trial and error operations if it wants to introduce new product and collect reviews from large customer base especially during the afternoon 13-18 hrs.
- ✓ The state RR has not performed well regarding business. Since the customer count, order flows are low and the average freight cost, average delivery time are high, inventory management and logistic operations should be improved. It may attract more customers and improve the business.
- ✓ Attracting offers on credit card might increase the business during non-trending months.