TARGET-SQL-Business-Case

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

A. Data type of all columns in the “customers” table.

Ans:
```
SELECT column_name,data_type
    FROM bussiness-study.Target_SQL.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

Insight:- Customer basic details

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

Ans:
```
SELECT  MIN(order_purchase_timestamp) Min, MAX(order_purchase_timestamp) Max
    FROM Target_SQL.orders;
```

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

Insight: - It’s showing the difference of an order placing keep increasing from 2016 to 2018

Recommendation:- Using visual representation we can see min and max orders placed in a month

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

Ans:
```
SELECT COUNT (distinct customer_city) No_of_city, COUNT (distinct customer_state) No_of_states
    FROM`Target_SQL.customers` c
    INNER JOIN `Target_SQL.orders` o
    ON c.customer_id = o.customer_id;
```

Row	No_of_city	No_of_states
1	4119	27

Insight:- there is 4119 city who placed order from 27 states

Recommendation:- Find out the most customer ordered from which city on daily basis to give offer to them

II. In-depth Exploration:

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

Ans:

```
SELECT
    EXTRACT (YEAR From order_purchase_timestamp) as Year,
    EXTRACT (MONTH From order_purchase_timestamp) as Month,
    Count (order_id) as num_of_orders
    FROM `Target_SQL.orders`
    GROUP BY Year, Month
    ORDER BY Year, Month LIMIT 10;
```

Row	Year	Month	num_of_orders
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

Insight:- As we can see the number of order is increasing year by year

Recommendation:- Increase the server count to handle the traffic

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

Ans:

```
SELECT
    COUNT (order_id) as no_of_orders,
    EXTRACT (MONTH FROM order_purchase_timestamp) as Monthly
    FROM `Target_SQL.orders`
    GROUP BY Monthly
    ORDER BY no_of_orders desc, Monthly;
```

Row	no_of_orders	Monthly
1	10843	8
2	10573	5
3	10318	7
4	9893	3
5	9412	6
6	9343	4
7	8508	2
8	8069	1
9	7544	11
10	5674	12
11	4959	10
12	4305	9

Insight:- No. of orders keep increasing between the month of Aug and April

Recommendation:- Focus on the product which is getting more popularity

C. 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

Ans:
```
SELECT COUNT (o.order_id) as order_count,
    CASE
      WHEN EXTRACT (HOUR FROM o.order_purchase_timestamp) <= 6 THEN 'Dawn'
      WHEN EXTRACT (HOUR FROM o.order_purchase_timestamp) BETWEEN 7 and 12 THEN 'Mornings'
      WHEN EXTRACT (HOUR FROM o.order_purchase_timestamp) BETWEEN 13 and 18 THEN 'Afternoon'
      WHEN EXTRACT (HOUR FROM o.order_purchase_timestamp) BETWEEN 19 and 23 THEN 'Night'
    END as Hour
      FROM `Target_SQL.orders` o
      JOIN `Target_SQL.customers` c
      ON o.customer_id = c.customer_id
      GROUP BY Hour
      ORDER BY order_count desc;
```

Row	order_count	Hour
1	38135	Afternoon
2	28331	Night
3	27733	Mornings
4	5242	Dawn

Insight: - Most of the orders placed in Afternoon

Recommendation: - Increase the product stock in afternoon for availability

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

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

Ans:
```
SELECT c.customer_state,

EXTRACT (MONTH FROM o.order_purchase_timestamp) as Month,
COUNT (o.order_id) as order_count
FROM `Target_SQL.orders`o
INNER JOIN `Target_SQL.customers`c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state, Month
ORDER BY order_count desc, Month
LIMIT 10;
```

Row	customer_state	Month	order_count
1	SP	8	4982
2	SP	5	4632
3	SP	7	4381
4	SP	6	4104
5	SP	3	4047
6	SP	4	3967
7	SP	2	3357
8	SP	1	3351
9	SP	11	3012
10	SP	12	2357

Insight:- Most of the order from SP state

Recommendation:- Give them special discount if the customer order is greater than 2

B. How are the customers distributed across all the states?

Ans:
```
SELECT customer_state,

COUNT (customer_unique_id) as No_of_customers
FROM `Target_SQL.customers`
 GROUP BY customer_state
ORDER BY No_of_customers desc
LIMIT 10;
```

Row	customer_state	No_of_customers
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

Insight:- Above 10 state has most orders
Recommendation:- Give special discount to top three state

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

A. 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:

```
WITH Year_2017 as
(
    SELECT
    EXTRACT (MONTH from o.order_purchase_timestamp) AS Month, ROUND (SUM (p.payment_value)) as cost_of_2017
    FROM `Target_SQL.orders` o JOIN `Target_SQL.payments` p
    USING (order_id)
    WHERE (EXTRACT (YEAR from o.order_purchase_timestamp) = 2017) and (EXTRACT (MONTH from o.order_purchase_timestamp) between 1 and 8)
    GROUP BY Month
    ORDER BY Month),

    Year_2018 as
    (
    SELECT
    EXTRACT (MONTH from o.order_purchase_timestamp) AS Month, ROUND (SUM (p.payment_value)) as cost_of_2018
    FROM `Target_SQL.orders` o JOIN `Target_SQL.payments` p
    USING (order_id)
    WHERE (EXTRACT (YEAR from o.order_purchase_timestamp) = 2018) and (EXTRACT (MONTH from o.order_purchase_timestamp) between 1 and 8)
    GROUP BY Month
    ORDER BY Month
    )
    Select *, round (((cost_of_2018 - cost_of_2017)/cost_of_2017)*100, 2) as Percentage
    FROM Year_2017 join Year_2018
    USING (Month)
    ORDER BY Month;
```

Row	Month	cost_of_2017	cost_of_2018	Percentage
1	1	138488.0	1115004.0	705.13
2	2	291908.0	992463.0	239.99
3	3	449864.0	1159652.0	157.78
4	4	417788.0	1160785.0	177.84
5	5	592919.0	1153982.0	94.63
6	6	511276.0	1023880.0	100.26
7	7	592383.0	1066541.0	80.04
8	8	674396.0	1022425.0	51.61

Insight:- There is slightest difference in the cost of 2017 and 2018 and percentage increase between the month of Jan and April

Recommendation:- Increase the service by location which is increasing in demand

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

Ans:
```
SELECT
        s.seller_state,
        ROUND (SUM (ot.price)) as Total_price,
        ROUND (AVG (ot.price)) as Avg_price
FROM `Target_SQL.order_items` ot
LEFT JOIN `Target_SQL.sellers` s
ON ot.seller_id = s.seller_id
GROUP BY s.seller_state
ORDER BY s.seller_state
LIMIT 10;
```

Row	seller_state	Total_price	Avg_price
1	AC	267.0	267.0
2	AM	1177.0	392.0
3	BA	285562.0	444.0
4	CE	20241.0	215.0
5	DF	97749.0	109.0
6	ES	47690.0	128.0
7	GO	66399.0	128.0
8	MA	36409.0	90.0
9	MG	1011565.0	115.0
10	MS	8552.0	171.0

Insight:- From state BA and MG has been placed most order

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

Ans:

```
SELECT
        s.seller_state,
        ROUND (SUM (ot.freight_value)) AS Total_freight_value,
        ROUND (AVG (ot.freight_value)) as Avg_freight_value
FROM `Target_SQL.order_items` ot
LEFT JOIN `Target_SQL.sellers` s
ON ot.seller_id = s.seller_id
GROUP BY s.seller_state
ORDER BY s.seller_state
LIMIT 10;
```

Row	seller_state	Total_freight_value	Avg_freight_value
1	AC	33.0	33.0
2	AM	82.0	27.0
3	BA	19701.0	31.0
4	CE	4360.0	46.0
5	DF	18494.0	21.0
6	ES	12171.0	33.0
7	GO	12565.0	24.0
8	MA	12141.0	30.0
9	MG	212595.0	24.0
10	MS	1199.0	24.0

Insight:- **From state BA and MG has been placed most order**

v. Analysis based on sales, freight and delivery time.

A. 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

Ans:

```
SELECT
```

```
        order_id,
        CASE
          WHEN DATE_DIFF (order_delivered_customer_date, order_purchase_timestamp, DAY) >=0
          THEN DATE_DIFF (order_delivered_customer_date, order_purchase_timestamp, DAY)
          ELSE NULL
        END AS time_to_deliver,
        CASE
          WHEN DATE_DIFF (order_estimated_delivery_date, order_delivered_customer_date, DAY) >= 0
          THEN DATE_DIFF (order_estimated_delivery_date, order_delivered_customer_date, DAY)
          ELSE NULL
        END AS diff_estimated_delivery
FROM `Target_SQL.orders`
```

Row	order_id	time_to_deliver	diff_estimated_delive
1	1950d777989f6a877539f5379…	30	*null*
2	2c45c33d2f9cb8ff8b1c86cc28…	30	28
3	65d1e226dfaeb8cdc42f66542…	35	16
4	635c894d068ac37e6e03dc54e…	30	1
5	3b97562c3aee8bdedcb5c2e45…	32	0
6	68f47f50f04c4cb6774570cfde…	29	1
7	276e9ec344d3bf029ff83a161c…	43	*null*
8	54e1a3c2b97fb0809da548a59…	40	*null*
9	fd04fa4105ee8045f6a0139ca5…	37	*null*
10	302bb8109d097a9fc6e9cefc5…	33	*null*

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

Ans:
```
WITH Statefreight AS (

  SELECT
        s.seller_state,
        ROUND (AVG (ot.freight_value)) AS avg_freight
  FROM `Target_SQL.order_items` ot
  JOIN `Target_SQL.sellers` s
  ON ot.seller_id = s.seller_id
  GROUP BY s.seller_state
  )

  SELECT
        seller_state,
        avg_freight
  FROM (
        SELECT
          seller_state,
          avg_freight,
          ROW_NUMBER () OVER (ORDER BY avg_freight DESC) AS high_rank,
          ROW_NUMBER () OVER (ORDER BY avg_freight ASC) AS low_rank
        FROM State freight
        )
  WHERE high_rank <= 5 OR low_rank <= 5
  ORDER BY avg_freight DESC;
```

Row	seller_state	avg_freight
1	RO	51.0
2	CE	46.0
3	PB	39.0
4	PI	37.0
5	ES	33.0
6	PR	23.0
7	DF	21.0
8	RJ	19.0
9	PA	19.0
10	SP	18.0

Insight:- As seen most of the orders from state RO
Recommendation:- Increase transportation in the state of RO to provide service effectively

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

Ans:

```
WITH DeliveryTime AS (
  SELECT
        c.customer_state,
        ROUND (AVG (DATE_DIFF (o.order_delivered_customer_date, o.order_purchase_timestamp, DAY))) AS avg_delivery_time

  FROM `Target_SQL.customers` c
  INNER JOIN `Target_SQL.orders` o
  ON c.customer_id = o.customer_id
  GROUP BY c.customer_state
)

  SELECT
        customer_state,
        avg_delivery_time
  FROM (
        SELECT
          customer_state,
          avg_delivery_time,
          ROW_NUMBER () OVER (ORDER BY avg_delivery_time ASC) AS low_rank,
          ROW_NUMBER () OVER (ORDER BY avg_delivery_time DESC) AS high_rank
  FROM DeliveryTime
)
WHERE low_rank <= 5 OR high_rank <= 5
ORDER BY avg_delivery_time;
```

Row	customer_state	avg_delivery_time
1	SP	8.0
2	MG	12.0
3	PR	12.0
4	DF	13.0
5	SC	14.0
6	PA	23.0
7	AL	24.0
8	AM	26.0
9	AP	27.0
10	RR	29.0

D. 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.

Ans:

```
SELECT
        c.customer_state,
        ROUND (AVG (DATE_DIFF (o.order_delivered_customer_date, o.order_purchase_timestamp, DAY))) AS avg_delivery_time,
        ROUND (AVG (DATE_DIFF (o.order_estimated_delivery_date, o.order_delivered_customer_date, DAY))) AS avg_diff_estimated_delivery
FROM `Target_SQL.customers` c
INNER JOIN `Target_SQL.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_diff_estimated_delivery ASC
LIMIT 5;
```

Row	customer_state	avg_delivery_time	avg_diff_estimated_c
1	AL	24.0	8.0
2	MA	21.0	9.0
3	SE	21.0	9.0
4	ES	15.0	10.0
5	SP	8.0	10.0

VI. Analysis based on the payments:

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

Ans:

```
SELECT
        EXTRACT (MONTH FROM o.order_purchase_timestamp) AS Month,
        EXTRACT (YEAR FROM o.order_purchase_timestamp) AS Year,
        p.payment_type,
        COUNT (o.order_id) AS order_count
FROM `Target_SQL.orders` o
INNER JOIN `Target_SQL.payments` p
ON o.order_id = p.order_id
```

```
GROUP BY Month, Year, p.payment_type
ORDER BY Year, Month;
```

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

Insight:- Most of the order are from payment type credit card

Recommendation:- Give 50% discount to the customer using credit card

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

Ans:
```
SELECT
      COUNT (DISTINCT o.order_id) AS order_count,
      p.payment_installments
FROM `Target_SQL.orders` o
JOIN  `Target_SQL.payments` p
ON o.order_id = p.order_id
WHERE p.payment_installments = 0
GROUP BY o.order_id, p.payment_installments
ORDER BY p.payment_installments;
```

Row	order_count	payment_installment
1	1	0
2	1	0

Insight:- Min order payment installment is done

Recommendation:- show them more recommendation to buy new product