TARGET SQL – 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: **Query:**

    Select column_name, data_type

    FROM `scaler-dsml-sql-444412.Target.INFORMATION_SCHEMA.COLUMNS`

    where table_name = 'customers';

    Output:

column_name	data_type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INT64
customer_city	STRING
customer_state	STRING

    Insights:

    This query provides list of all columns in customers table along with their data types.

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

    Ans: **Query:**

    Select MIN(order_purchase_timestamp) as first_order,

    MAX(order_purchase_timestamp) as last_order

    from `Target.orders`;

    Output:

first_order	last_order
2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

    Insights:

    By getting minimum and maximum order date, we can get overall time range for orders been placed.

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

    Ans: **Query:**

    Select COUNT(DISTINCT c.customer_city) AS count_of_cities,

COUNT(DISTINCT c.customer_state) AS count_of_states

from `Target.customers` c

JOIN `Target.orders` o ON c.customer_id = o.customer_id

where EXTRACT(YEAR from o.order_purchase_timestamp) BETWEEN 2016 AND 2018;

Output:

count_of_cities	count_of_states
4119	27

Insights:

This query is useful for geographic distribution and we can know how wide the orders are spread.

2. **In-depth Exploration:**

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

Ans: **Query:**

Select YEAR, no_of_orders,

LAG(no_of_orders,1) OVER(ORDER BY YEAR) as lag_year,

ROUND((no_of_orders/(LAG(no_of_orders,1) OVER(ORDER BY YEAR)))*100,2) as growing_trend_percentage

from

(SELECT EXTRACT(YEAR FROM order_purchase_timestamp) as Year, COUNT(order_id) as no_of_orders

from `Target.orders`

GROUP BY 1

ORDER BY 1) tbl

ORDER BY 1,3;

Output:

YEAR	no_of_orders	lag_year	growing_trend_percentage
2016	329	*null*	*null*
2017	45101	329	13708.51
2018	54011	45101	119.76

Insights:

There is growing trend from year after year as we can see positive values in growing_trend_percentage column. It also means company is implementing effective strategies for placing orders and reviewing at every interval.

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

Ans: **Query:**

```
Select EXTRACT(YEAR FROM order_purchase_timestamp) AS year,

EXTRACT(MONTH FROM order_purchase_timestamp) AS month,

COUNT(*) AS order_count

from `Target.orders`

GROUP BY 1,2

ORDER BY 1,2;
```

Output:

year	month	order_count
2016	9	4
2016	10	324
2016	12	1
2017	1	800
2017	2	1780
2017	3	2682
2017	4	2404
2017	5	3700
2017	6	3245
2017	7	4026

Note: There are total 25 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

Insights:

We can see output where there is an increasing in monthly seasonality wherein activities might be life festivals, campaigns, holidays or events.

And wherein we can also see downwards trend on some monthly seasons, so overcome this we might implement some strategies such as campaigns, events or promotional activities.

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

Ans: **Query:**

```
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
'Mornings'

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'

END AS DAY,

COUNT(order_id) as no_of_orders

FROM `Target.orders`

GROUP BY 1

ORDER BY 2 DESC

LIMIT 1;
```

Output:

DAY	no_of_orders
Afternoon	38135

Insights:

We can see most orders are placed at Afternoon, might be customers are getting proper customer **support and delivery response in afternoon.**

Recommendations:

We can increase staffing for proper prompt customer support when activity is high.

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

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

    Ans: **Query:**

```
Select c.customer_state,
EXTRACT(month from o.order_purchase_timestamp) as month,
COUNT(o.order_id) as cnt_of_orders
from `Target.orders` o
INNER JOIN `Target.customers` c
USING (customer_id)
GROUP BY 1, 2
ORDER BY 1, 2;
```

Output:

customer_state	month	cnt_of_orders
AC	1	8
AC	2	6
AC	3	4
AC	4	9
AC	5	10
AC	6	7
AC	7	9
AC	8	7
AC	9	5
AC	10	6

Note: There are total 322 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

Insights:

This shows us that by state wise how much is the sales volume and seasonality sales by month on month which help us whether to increase inventory or promote maketing.

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

Ans: **Query:**

```
Select customer_state, count(customer_id) as no_of_customers

from `Target.customers`

group by 1

order by 2 desc;
```

Output:

customer_state	no_of_customers
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033
GO	2020

Note: There are total 27 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

Insights:

States with highest customer shows the customer engagement and states with lowest customer would require customer acquisition and promote marketing.

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: **Query:**

```
Select Year, total_payment,

LAG(total_payment) OVER (ORDER BY Year) as previous_year_payment,

ROUND((((total_payment) - (LAG(total_payment) OVER (ORDER BY Year))) /
(LAG(total_payment) OVER (ORDER BY Year))) * 100,2) as
percentage_increase_in_cost_of_orders

from

(Select EXTRACT(YEAR from o.order_purchase_timestamp) as Year,

ROUND(SUM(p.payment_value),2) as total_payment

from `Target.payments` p

INNER JOIN `Target.orders` o

USING (order_id)

where (EXTRACT(YEAR from order_purchase_timestamp) BETWEEN 2017 AND 2018) AND

(EXTRACT(MONTH from order_purchase_timestamp) BETWEEN 1 AND 8)

group by Year) tbl

ORDER BY 1;
```

Output:

Year	total_payment	previous_year_payment	percentage_increase_in_cost_of_orders
2017	3669022.12	*null*	*null*
2018	8694733.84	3669022.12	136.98

Insights:

This output shows that there is an increase in Year 2018 compared to Year 2017 from Jan to Aug months which indicate to strategize for inventory, staffing and customer support.

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

Ans: **Query:**

```
select c.customer_state,
ROUND(SUM(oi.price),2) as total_order,
ROUND(AVG(oi.price),2) as average_order
from `Target.order_items` oi
JOIN `Target.orders` o ON oi.order_id = o.order_id
JOIN `Target.customers` c ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY 1;
```

Output:

customer_state	total_order	average_order
AC	15982.95	173.73
AL	80314.81	180.89
AM	22356.84	135.5
AP	13474.3	164.32
BA	511349.99	134.6
CE	227254.71	153.76
DF	302603.94	125.77
ES	275037.31	121.91
GO	294591.95	126.27
MA	119648.22	145.2

Note: There are total 27 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

Insights:

We can see from the output that the data shows state wise total orders and average orders wherein we can see which states has higher orders to prioritize the Target Business segment and inventory management for these states.

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

Ans: **Query:**

```
select c.customer_state,
ROUND(SUM(oi.freight_value),2) as total_freight,
ROUND(AVG(oi.freight_value),2) as average_freight
FROM `Target.order_items` oi
JOIN `Target.orders` o ON oi.order_id = o.order_id
JOIN `Target.customers` c ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY 1;
```

Output:

customer_state	total_freight	average_freight
AC	3686.75	40.07
AL	15914.59	35.84
AM	5478.89	33.21
AP	2788.5	34.01
BA	100156.68	26.36
CE	48351.59	32.71
DF	50625.5	21.04
ES	49764.6	22.06
GO	53114.98	22.77
MA	31523.77	38.26

Note: There are total 27 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

Insights:

We can see from the output that the data shows state wise total freight and average freight wherein we can see which states has higher freight to optimize delivery routes and negotiate better rates with vendors.

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: **Query:**

```
select order_id,

DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_deliver,

DATE_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY) AS diff_estimated_delivery

from `Target.orders`

where order_delivered_customer_date IS NOT NULL;
```

Output:

order_id	time_to_deliver	diff_estimated_delivery
1950d777989f6a877539f53795b4c3c3	30	12
2c45c33d2f9cb8ff8b1c86cc28c11c30	30	-28
65d1e226dfaeb8cdc42f665422522d14	35	-16
635c894d068ac37e6e03dc54eccb6189	30	-1
3b97562c3aee8bdedcb5c2e45a50d5e1	32	0
68f47f50f04c4cb6774570cfde3a9aa7	29	-1
276e9ec344d3bf029ff83a161c6b3ce9	43	4
54e1a3c2b97fb0809da548a59f64c813	40	4
fd04fa4105ee8045f6a0139ca5b49f27	37	1
302bb8109d097a9fc6e9cefc5917d1f3	33	5

Note: There are total 96476 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here

Insights:

This output shows of time_to_deliver wherein indicates efficient logistics for shorter delivery time and improvement in longer delivery time and dif_estimated_delivery indicates positive values for late delivery, negative values for early delivery and zero for on time delivery.

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

Ans: **Query:**

- Top 5 states with highest average freight value.

```
Select DISTINCT customer_state, highest_avg_freight_value

from

(Select c.customer_state,

ROUND(SUM(oi.freight_value) OVER (PARTITION BY c.customer_state)/COUNT(DISTINCT oi.freight_value) OVER(PARTITION BY c.customer_state),2) AS highest_avg_freight_value

FROM `Target.order_items` oi

JOIN `Target.orders` o ON oi.order_id = o.order_id

JOIN `Target.customers` c ON o.customer_id = c.customer_id

) tbl

ORDER BY highest_avg_freight_value DESC

LIMIT 5;
```

- Top 5 states with lowest average freight value.

```
Select DISTINCT customer_state, lowest_avg_freight_value

from

(Select c.customer_state,

ROUND(SUM(oi.freight_value) OVER (PARTITION BY c.customer_state)/COUNT(DISTINCT oi.freight_value) OVER(PARTITION BY c.customer_state),2) AS lowest_avg_freight_value

FROM `Target.order_items` oi

JOIN `Target.orders` o ON oi.order_id = o.order_id

JOIN `Target.customers` c ON o.customer_id = c.customer_id

) tbl

ORDER BY lowest_avg_freight_value ASC

LIMIT 5;
```

Output:

- Top 5 states with highest average freight value.

customer_state	highest_avg_freight_value
SP	186.39
RJ	112.23
MG	107.87
RS	78.34
PR	71.77

- Top 5 states with lowest average freight value.

customer_state	lowest_avg_freight_value
MS	43.51
AP	44.26
AM	47.64
MT	50.19
AL	51.17

Insights:

Through this output we can make assumption that highest average freight values might be due to remote areas and lowest average freight values might be closer to distribution centres or storage warehouses.

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

Ans: **Query:**

- Top 5 states with highest average delivery time.

```
Select DISTINCT customer_state, highest_avg_delivery_time

FROM

(SELECT c.customer_state,

ROUND((SUM(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp,
DAY)) OVER (PARTITION BY c.customer_state) / COUNT(DISTINCT o.order_id) OVER
(PARTITION BY c.customer_state)),2) as highest_avg_delivery_time

FROM `Target.orders` o

JOIN `Target.customers` c ON o.customer_id = c.customer_id

WHERE order_status = 'delivered'

) tbl

ORDER BY highest_avg_delivery_time DESC

LIMIT 5;
```

- Top 5 states with lowest average delivery time.

```
Select DISTINCT customer_state, lowest_avg_delivery_time

FROM

(SELECT c.customer_state,

ROUND((SUM(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp,
DAY)) OVER (PARTITION BY c.customer_state) / COUNT(DISTINCT o.order_id) OVER
(PARTITION BY c.customer_state)),2) as lowest_avg_delivery_time

FROM `Target.orders` o

JOIN `Target.customers` c ON o.customer_id = c.customer_id

WHERE order_status = 'delivered'

) tbl

ORDER BY lowest_avg_delivery_time ASC

LIMIT 5;
```

Output:

- Top 5 states with highest average delivery time.

customer_state	highest_avg_delivery_time
RR	28.98
AP	26.73
AM	25.99
AL	24.04
PA	23.32

- Top 5 states with lowest average delivery time.

customer_state	lowest_avg_delivery_time
SP	8.3
PR	11.53
MG	11.54
DF	12.51
SC	14.48

Insights:

Average delivery time higher might be geographical challenges like delivery transport issues or no distribution centres or warehouses.

Average delivery time lower might be closer to delivery centres or warehouses or shorter distance.

Recommendations:

To evaluate and optimise delivery routes, establish warehouses or delivery centres closer distances or partner with logistics for timely delivery.

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.

Ans: **Query:**

```
Select DISTINCT customer_state, avg_actual_delivery_days, avg_estimated_delivery_days,

ROUND((avg_estimated_delivery_days - avg_actual_delivery_days),2) AS
delivery_time_difference

from

(select c.customer_state,

ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp,
DAY)) OVER (PARTITION BY c.customer_state),2) AS avg_actual_delivery_days,

ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date, o.order_purchase_timestamp,
DAY)) OVER (PARTITION BY c.customer_state),2) AS avg_estimated_delivery_days

from `Target.orders` o

JOIN `Target.customers` c ON o.customer_id = c.customer_id

where o.order_status = 'delivered'

) tbl

ORDER BY 4 DESC

LIMIT 5;
```

Output:

customer_state	avg_actual_delivery_days	avg_estimated_delivery_days	delivery_time_difference
AC	20.64	40.73	20.09
RO	18.91	38.39	19.48
AP	26.73	45.87	19.14
AM	25.99	44.92	18.93
RR	28.98	45.63	16.65

Insights:

This output shows that states show efficiency in delivering orders faster than estimated time. Faster delivery lead to higher customer satisfaction and loyalty.

6. **Analysis based on the payments:**

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

    Ans: **Query:**

```
Select EXTRACT(MONTH FROM o.order_purchase_timestamp) as month,
p.payment_type,
COUNT(o.order_id) as no_of_orders
from `Target.orders` o
JOIN `Target.payments` p ON o.order_id = p.order_id
GROUP BY 1,2
ORDER BY 1,2;
```

    Output:

month	payment_type	no_of_orders
1	UPI	1715
1	credit_card	6103
1	debit_card	118
1	voucher	477
2	UPI	1723
2	credit_card	6609
2	debit_card	82
2	voucher	424
3	UPI	1942
3	credit_card	7707

    Note: There are total 50 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

    Insights:

    This output shows us that the month on month payment method and no. of orders. For example, credit card has a no. of orders high then customers prefer credit card as their payment method mode.

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

Ans: **Query:**

```
Select payment_installments, COUNT(order_id) AS order_count
from `Target.payments`
GROUP BY 1
ORDER BY 1;
```

Output:

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

Note: There are total 24 rows we got as an output but as per question screenshot of the first 10 rows from the output should be displayed here.

Insights:

This output shows that no. of orders with payment installment 1 that customer prefer immediate payment and understanding customer payment preference can help tailor marketing strategies and payment options to better suit their needs. Also, analysis of payment installments can provide financial planning and cash flow.

Recommendations:

We can introduce flexible payment plans as per customers demand of payment installment for high demand orders.