Target SQL - Business CaseStudy.

1) To gain insights into its characteristics, I began by examining the basic properties of the dataset. I checked the number of rows and columns to get an idea of its size. This would help me understand the scope and potential complexities of the data.

i) Data type of all columns in the "customers" table.

Query -

```
SELECT
 column_name,
 data_type
FROM
 `targetsql-390511.Target.INFORMATION_SCHEMA.COLUMNS`
WHERE
 table_name = 'customers'
```

Outcome -

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

Note - As mentioned in the session, attaching the schema for just one table to basically give an idea that how to approach this particular instance of querying the schema.

---

ii) The time range between which the orders were placed -

Query -

```
SELECT
 MIN(order_purchase_timestamp) AS min_purchase_timestamp,
 MAX(order_purchase_timestamp) AS max_purchase_timestamp
FROM
`Target.orders`
```

Outcome -

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

Insight - This query helped us to identify the Min & Max date from the orders table.

---

ii) Count the Cities & States of customers who ordered during the given period -

Query -

```
SELECT
COUNT(DISTINCT customer_city) AS city_count,
COUNT(DISTINCT customer_state) AS state_count
FROM `Target.customers`
```

Outcome -

Row	city_count	state_count
1	4119	27

Insight - This query helped us to identify the total Distinct count of Cities & States from the Customers table.

----------X----------X----------

2) In-depth Exploration -

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

Query -

```
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS purchase_year,
COUNT(*) AS order_count
FROM
`Target.orders`
GROUP BY
purchase_year
ORDER BY
purchase_year
```

Outcome -

Row	purchase_year	order_count
1	2016	329
2	2017	45101
3	2018	54011

Insight - This query helped us to identify the drastic increasing count of orders over the years 2016, 2017, 2018, where 2016 being the least of all and 2018 being the Highest count by 164 times.

---

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

Query -

```
SELECT
```

```
EXTRACT(YEAR FROM order_purchase_timestamp) AS purchase_year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS purchase_month,
COUNT(*) AS order_count
FROM
`Target.orders`
GROUP BY
purchase_year, purchase_month
ORDER BY
purchase_year, purchase_month
```

Outcome -

Row	purchase_year	purchase_month	order_count
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

Insights - This query helped us to identify the drastic increasing count of orders over the years as well as month on month basis, where 2016 being the least of all. Looking at the increasing ordercount, it seems to be that Oct month of 2016 had the highest flow of orders, where in 2017 had a very gradual increase of orders throught the year and finally 2018 has a very consistent and somewhat equal distribution in terms of the flow of orders.

---

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

Query -

```
SELECT
CASE
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >=0 AND EXTRACT(HOUR FROM order_purchase_timestamp) <=6 THEN 'Dawn'
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >=7 AND EXTRACT(HOUR FROM order_purchase_timestamp) <=12 THEN 'Morning'
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >=13 AND EXTRACT(HOUR FROM order_purchase_timestamp) <=18 THEN 'Afternoon'
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >=19 AND EXTRACT(HOUR FROM order_purchase_timestamp) <=23 THEN 'Night'
END AS purchase_timings,
COUNT(*) AS Order_count
FROM `Target.orders`
GROUP BY purchase_timings
ORDER BY Order_count DESC
```

Outcome -

Row	purchase_timings	Order_count
1	Afternoon	38135

2	Night	28331
3	Morning	27733
4	Dawn	5242

Insights - This query helped us to identify the timing windows of the orders that are placed by the customers. Segregated as 4 different windows namely Dawn, Morning, Afternoon and Night.

----------X----------X----------

3) Evolution of E-commerce orders in the Brazil region -

i) Month on month no. of orders placed in each state.

Query -

```
SELECT
 EXTRACT(YEAR FROM o.order_purchase_timestamp) AS purchase_year,
 EXTRACT(MONTH FROM o.order_purchase_timestamp) AS purchase_month,
 c.customer_state,
 COUNT(*) AS order_count
FROM
 `Target.orders`o
JOIN `Target.customers`c  ON o.customer_id = c.customer_id
GROUP BY
 purchase_year, purchase_month, c.customer_state
ORDER BY
 purchase_year, purchase_month, c.customer_state
```

Outcome -

Row	purchase_year	purchase_month	customer_state	order_count
1	2016	9	RR	1
2	2016	9	RS	1
3	2016	9	SP	2
4	2016	10	AL	2
5	2016	10	BA	4
6	2016	10	CE	8
7	2016	10	DF	6
8	2016	10	ES	4
9	2016	10	GO	9
10	2016	10	MA	4

Insights - This query helped us to identify that the highest order count over the years 2017 & 2018 is SP (Sao Paulo)

Note - it is not added in the above screenshot as it only consists the first 10 rows according to the instruction. As the first 10 row is a result of ordering for neat presentation.

----

ii) How are the customers distributed across all the states?

Query -

```
SELECT
 customer_state,
 COUNT(*) AS customer_count
FROM
 `Target.customers`
GROUP BY
 customer_state
ORDER BY
 customer_count DESC
```

Outcome -

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

Insights - This query helped us to identify that the Top 3 order contributing states are SP (Sao Paulo), RJ (Rio de Janeiro) & MG (Minas Gerais)

----------X----------X----------

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

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

Query -

```
SELECT
 (cost_2018 - cost_2017) / (cost_2017) * 100 AS cost_increase_percentage
FROM (
 SELECT
  SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017 THEN p.payment_value ELSE 0 END) AS cost_2017,
  SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2018 THEN p.payment_value ELSE 0 END) AS cost_2018
 FROM
  `Target.orders` o
 JOIN
  `Target.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
)
```

Outcome -

Row	cost_increase_percentage
1	136.97687164666226

Insights - This query helped us to identify that the cost has been increased by 136.9% through 2017 till 2018.

---

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

Query -

```
SELECT
 c.customer_state,
 ROUND(SUM(price),2) AS total_order_price,
 ROUND(AVG(price),2) AS average_order_price
FROM
 `Target.orders` o
JOIN
  `Target.customers`c
ON
 c.customer_id = o.customer_id
JOIN
 `Target.order_items` oi
ON
 o.order_id = oi.order_id
GROUP BY
 customer_state
```

Outcome -

Row	customer_state	total_order_price	average_order_price
1	MT	156453.53	148.3
2	MA	119648.22	145.2
3	AL	80314.81	180.89
4	SP	5202955.05	109.65
5	MG	1585308.03	120.75
6	PE	262788.03	145.51
7	RJ	1824092.67	125.12
8	DF	302603.94	125.77
9	RS	750304.02	120.34
10	SE	58920.85	153.04

Insights - This query helped us to identify that the Total value & Average value of orders across the states.

---

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

Query -

```
SELECT
 c.customer_state,
 ROUND(SUM(oi.freight_value),2) AS total_freight_value,
 ROUND(AVG(oi.freight_value),2) AS average_freight_value
FROM
 `Target.orders` o
JOIN
  `Target.customers`c
ON
 c.customer_id = o.customer_id
JOIN
 `Target.order_items` oi
ON
 o.order_id = oi.order_id
GROUP BY
 customer_state
```

Outcome -

Row	customer_state	total_freight_value	average_freight_value
1	MT	29715.43	28.17
2	MA	31523.77	38.26
3	AL	15914.59	35.84
4	SP	718723.07	15.15
5	MG	270853.46	20.63
6	PE	59449.66	32.92
7	RJ	305589.31	20.96
8	DF	50625.5	21.04
9	RS	135522.74	21.74
10	SE	14111.47	36.65

Insights - This query helped us to identify that the Total & Average of freight value across the states.

----------X----------X----------

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

i) 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.

Query -

```
SELECT
order_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_deliver,
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS diff_estimated_delivery
FROM `Target.orders`
```

Outcome -

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

Insights - This query helped us to identify the Time required for each delivery from the date of purchase & the Time difference between estimate and actual delivery times.

---

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

Query -

```
WITH top_states AS (
  SELECT
    customer_state,
    AVG(freight_value) AS average_freight_value,
    ROW_NUMBER() OVER (ORDER BY AVG(freight_value) DESC) AS rank_highest,
    ROW_NUMBER() OVER (ORDER BY AVG(freight_value) ASC) AS rank_lowest
  FROM
    `Target.order_items` AS oi
  JOIN
    `Target.orders` AS o
  ON
    oi.order_id = o.order_id
  JOIN
    `Target.customers` AS c
  ON
    o.customer_id = c.customer_id
  GROUP BY
    customer_state
)
SELECT
  customer_state,
  average_freight_value
FROM
  top_states
WHERE
  rank_highest <= 5
  OR rank_lowest <= 5
ORDER BY
  average_freight_value DESC
```

Outcome -

Row	customer_state	average_freight_valu
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...
6	DF	21.04135494596...
7	RJ	20.96092393168...
8	MG	20.63016680630...
9	PR	20.53165156794...
10	SP	15.14727539041...

Insights - This query helped us to identify theTop and Bottom 5 values with respect to the freight values

---

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

Query -

```
WITH delivery_times AS (
  SELECT
    c.customer_state,
    AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)) AS average_delivery_time
  FROM
    `Target.orders` AS o
  JOIN
    `Target.customers` AS c
  ON
    o.customer_id = c.customer_id
  WHERE
    o.order_status = 'delivered'
  GROUP BY
    c.customer_state
)
SELECT
  customer_state,
  average_delivery_time
FROM (
  SELECT
    customer_state,
    average_delivery_time,
    ROW_NUMBER() OVER (ORDER BY average_delivery_time DESC) AS rank_highest,
    ROW_NUMBER() OVER (ORDER BY average_delivery_time ASC) AS rank_lowest
  FROM
    delivery_times
)
WHERE
  rank_highest <= 5 OR rank_lowest <= 5
ORDER BY
  average_delivery_time DESC
```

Outcome -

Row	customer_state	average_delivery_time
1	RR	28.975609756097562
2	AP	26.731343283582085
3	AM	25.986206896551728
4	AL	24.040302267002513
5	PA	23.316067653276981
6	SC	14.475183305132528
7	DF	12.509134615384616
8	MG	11.54218777523343
9	PR	11.526711354864908
10	SP	8.2980935447227022

Insights - This query helped us to identify the Top and Bottom 5 values with respect to the delivery time required.

---

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

Query -

```
WITH delivery_time_diff AS (
 SELECT
  c.customer_state,
  AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_estimated_delivery_date, DAY)) AS delivery_time_difference
 FROM
  `Target.orders` AS o
 JOIN
  `Target.customers` AS c
 ON
  o.customer_id = c.customer_id
 WHERE
  o.order_status = 'delivered'
 GROUP BY
  c.customer_state
)


SELECT
 customer_state,
 delivery_time_difference
FROM (
 SELECT
  customer_state,
  delivery_time_difference,
  ROW_NUMBER() OVER (ORDER BY delivery_time_difference ASC) AS rank_fastest
 FROM
  delivery_time_diff
)
WHERE
 rank_fastest <= 5
ORDER BY
```

```
delivery_time_difference ASC
```

Outcome -

Row	customer_state	delivery_time_difference
1	AC	-19.762500000000006
2	RO	-19.13168724279836
3	AP	-18.731343283582088
4	AM	-18.60689655172413
5	RR	-16.414634146341463

Insights - This query helped us to identify the Top 5 deliveries where the the delivery was done much earlier than its estimated delivery date.

----------X----------X----------

6) Analysis based on the payments.

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

Query -

```
WITH monthly_orders AS (
  SELECT
    FORMAT_TIMESTAMP('%Y-%m', o.order_purchase_timestamp) AS purchase_month,
    p.payment_type,
    COUNT(DISTINCT o.order_id) AS order_count
  FROM
    `Target.orders` AS o
  JOIN
    `Target.payments` AS p
  ON
    o.order_id = p.order_id
  GROUP BY
    purchase_month,
    p.payment_type
)
SELECT
  purchase_month,
  payment_type,
  order_count
FROM
  monthly_orders
ORDER BY
  purchase_month,
  payment_type
```

Outcome -

Row	purchase_month	payment_type	order_count
1	2016-09	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	2017-01	UPI	197
8	2017-01	credit_card	582
9	2017-01	debit_card	9
10	2017-01	voucher	33

---

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

Query -

```
SELECT
 payment_installments,
 COUNT(DISTINCT order_id) AS order_count
FROM
 `Target.payments`
GROUP BY
 payment_installments
```

Outcome -

Row	payment_installment	order_count
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644