Business Case - Target SQL

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.
    SQL Query:
    SELECT COLUMN_NAME, DATA_TYPE
    FROM TargetSQL.INFORMATION_SCHEMA.COLUMNS
    WHERE TABLE_NAME = 'customers'

    Query results

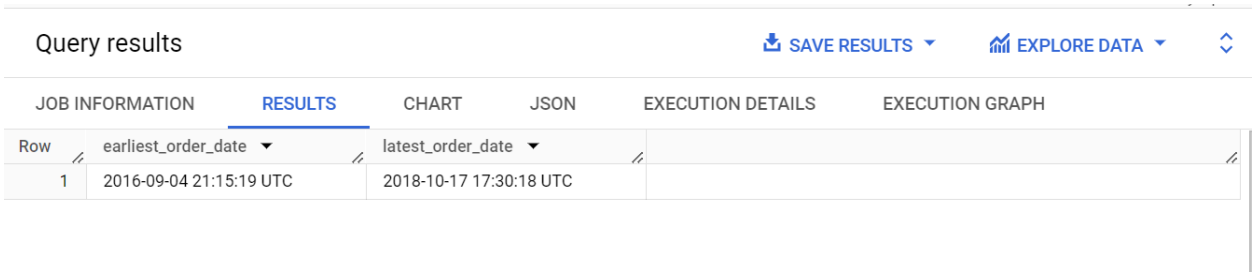
    JOB INFORMATION | RESULTS | CHART | JSON | E

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

    Inference:
    - **With this, we gain a foundational understanding of the customer table data structure and data types.**
    - **This insight is required for further Query writing as part of our analysis.**

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

    SQL Query:
    SELECT MIN(order_purchase_timestamp) AS earliest_order_date,
    MAX(order_purchase_timestamp) AS latest_order_date
    FROM TargetSQL.orders;



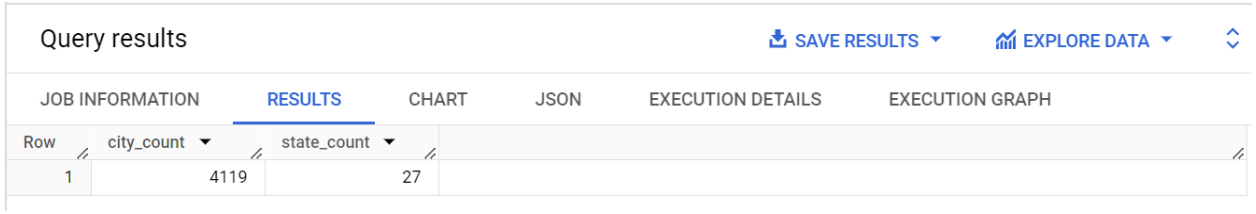
Inference:

- **By taking the order_purchase_timestamp as the indication for the order purchased date, the first order was placed on "2016-09-04 21:15:19 UTC" and the latest order was placed on "2018-10-17 17:30:18 UTC".**
- **This gives us an overall idea of the dataset for the date range.**

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

SQL Query:

```
SELECT
 COUNT(DISTINCT c.customer_CITY) as city_count,
 COUNT(DISTINCT c.customer_state) as state_count
FROM
 TargetSQL.customers c
JOIN
 TargetSQL.orders o
ON
 c.customer_id=o.customer_id
WHERE
 order_purchase_timestamp BETWEEN '2016-09-04 21:15:19 UTC'
 AND '2018-10-17 17:30:18 UTC';
```



Inference:

- **The count of cities and states from which the customers have ordered between the given date (earliest and the latest order date) is extracted by joining the customers and the orders table.**
- **Based on this the count of cities is 4119 and states is 27.**
- **This data gives us an idea on the business landscape in Brazil.**

2. **In-depth Exploration:**

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

SQL Query:

```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS Year,
  COUNT(order_id) AS order_count
FROM
  TargetSQL.orders
GROUP BY
  Year
ORDER BY
  Year;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

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

Inference:

- **By extracting the year from the order_purchase_timestamp column and taking the count of orders per year by grouping the order by year we can see the Year-on-year growth of orders placed.**
- **This data will give us the growth of orders in terms of number in each year.**
- **This data can further be used to identify the growth factors and insightful decisions can be taken for market expansion.**

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

SQL Query:

```
SELECT
  EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH,
  COUNT(order_id) AS order_count
FROM
  TargetSQL.orders
GROUP BY
  MONTH
ORDER BY
  MONTH;
```

Query results

JOB INFORMATION | RESULTS | CHA

Row	MONTH	order_count
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
11	11	7544
12	12	5674

Inference:

- **Based on the above data, the number of orders fluctuates throughout the year with certain peaks and lows.**
- **The high-performing month is August with 10843 orders followed by May and July.**
- **The low-performing month is September with 4305 orders followed by October and December.**
- **The fluctuations in the order resemble seasonal changes in the orders throughout the year.**
- **Based on this data, decisions can be made for resource allocation, inventory management and warehouse management.**

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

SQL Query:

```
SELECT
 CASE
  WHEN TIME BETWEEN 0 AND 6 THEN 'Dawn'
  WHEN TIME BETWEEN 7 AND 12 THEN 'Morning'
  WHEN TIME BETWEEN 13 AND 18 THEN 'Afternoon'
  ELSE 'Night'
 END as Time_of_the_day,
 SUM(order_count) AS order_count
```

```
FROM (
 SELECT
  EXTRACT(HOUR FROM order_purchase_timestamp) AS TIME,
  COUNT(order_id) AS order_count
 FROM
  TargetSQL.orders
 GROUP BY
  TIME
)
GROUP BY
 Time_of_the_day
ORDER BY
 order_count DESC;
```

Query results

JOB INFORMATION | RESULTS | CHART

Row	Time_of_the_day	order_count
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

Inference:

- **This analysis will provide us the insights into how the purchases happen concerning the time of the day.**
- **Based on the above output data Brazilians order mostly during the Afternoon and order less at Dawn while the periods Night and Morning also see a significant order activity.**
- **With these insights, important decisions can be taken on resource shift allocations, logistics and inventory**

3. **Evolution of E-commerce orders in the Brazil region:**
    1. **Get the month-on-month no. of orders placed in each state.**

SQL Query:

```
SELECT
 c.customer_state AS state,
 EXTRACT(MONTH FROM o.order_purchase_timestamp) AS order_month,
 COUNT(o.order_id) AS order_count
FROM
 TargetSQL.orders o
```

```
 JOIN TargetSQL.customers c ON o.customer_id = c.customer_id
GROUP BY
 c.customer_state, order_month
ORDER BY
 c.customer_state, order_month;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTI

Row	state	order_month	order_count
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

Inference:

- **By grouping the counted orders month on month for every state we will give a detailed breakdown of orders placed each month, grouped by state.**
- **These insights can be used to analyze the shopping patterns of the customers based on the month and marketing efforts to increase sales.**

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

SQL Query:

```
SELECT
 customer_state AS state,
 COUNT(customer_id) AS customer_count
FROM
 TargetSQL.customers
GROUP BY
 customer_state
```

```
ORDER BY
 customer_count DESC;
```

Query results

JOB INFORMATION | RESULTS | CHART | J

Row	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

Inference:

- **Based on the above data we can understand the customer distribution across all the states of Brazil.**
- **State SP has the highest number of customers where as state RR have the lowest number of customers.**
- **The state with higher customers represents the best market area and will help us focus our marketing efforts, logistics and resource allocations where needed most.**

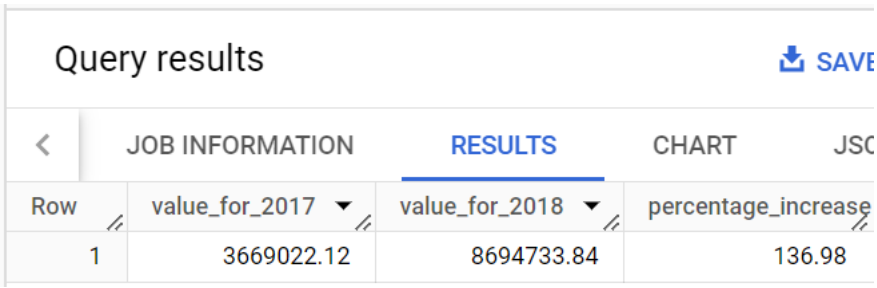
4. **Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.**

   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.**

SQL Query:

```
WITH
 CTE_1 AS (
 SELECT
  ROUND(SUM(p.payment_value),2) AS value_for_2017
```

```
 FROM
  TargetSQL.payments p
 JOIN
  TargetSQL.orders o
 ON
  p.order_id=o.order_id
 WHERE
  EXTRACT(YEAR
  FROM
   order_purchase_timestamp)=2017
  AND (EXTRACT(MONTH
    FROM
     order_purchase_timestamp) BETWEEN 1
    AND 8) ),
 CTE_2 AS (
 SELECT
  ROUND(SUM(p.payment_value),2) AS value_for_2018
 FROM
  TargetSQL.payments p
 JOIN
  TargetSQL.orders o
 ON
  p.order_id=o.order_id
 WHERE
  EXTRACT(YEAR
  FROM
   order_purchase_timestamp)=2018
  AND (EXTRACT(MONTH
    FROM
     order_purchase_timestamp) BETWEEN 1
    AND 8) )
SELECT
 CTE_1.value_for_2017,
 CTE_2.value_for_2018,
 ROUND((CTE_2.value_for_2018-CTE_1.value_for_2017)/CTE_1.value_for_2017*100,2)
AS percentage_increase
FROM
 CTE_1,
 CTE_2;
```



Row	value_for_2017	value_for_2018	percentage_increase
1	3669022.12	8694733.84	136.98

Inference:

- **There is an increase in the percentage of total payment value by 137 % for the year 2018 compared to the year 2017.**
- **The growth in the total payment value signifies the growth of the e-commerce business in Brazil as this could mean there is an increase in customer base, high order rate and increase in average order value.**
- **The average delivery time should be reduced by implementing analytic solutions to retain customers.**
- **To Attract traditional shoppers, the user experience must be increased.**
- **Marketing efforts should be increased by focusing on the brands.**

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

SQL Query:

```
SELECT
  c.customer_state AS State,
  ROUND(SUM(oi.price),2) AS Total_value,
  ROUND(AVG(oi.price),2) AS Average_value
FROM
  TargetSQL.customers c
JOIN
  TargetSQL.orders o
ON
  c.customer_id=o.customer_id
JOIN
  TargetSQL.order_items oi
ON
  o.order_id=oi.order_id
GROUP BY
  c.customer_state
ORDER BY
  c.customer_state;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTI

Row	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

Inference:

- **This Analysis will give us insights into the state with the highest total order value, average order value.**
- **These insights are needed for making important business decisions about logistics, Warehouse and inventory management.**

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

SQL Query:

```
SELECT
  c.customer_state AS State,
  ROUND(SUM(oi.freight_value),2) AS Total_freight_value,
  ROUND(AVG(oi.freight_value),2) AS Average_freight_value
FROM
  TargetSQL.customers c
JOIN
  TargetSQL.orders o
ON
  c.customer_id=o.customer_id
JOIN
  TargetSQL.order_items oi
ON
  o.order_id=oi.order_id
```

```
GROUP BY
 c.customer_state
ORDER BY
 c.customer_state;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTI

Row	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

Inference:

- **This Analysis will give us insights into the state with the highest total freight value, and average freight value.**
- **These insights can be used to make informed decisions involving optimizing logistics and distribution strategies.**
- **The company can prioritize resources, improve delivery efficiency, and negotiate better rates or partnerships with carriers in those regions.**

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

SQL 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 TargetSQL.orders
  WHERE order_delivered_customer_date IS NOT NULL
     AND order_purchase_timestamp IS NOT NULL;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTI

Row	order_id	time_to_deliver	diff_estimated_delive
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

Inference:

- **This data gives us insights into the delivery time taken for an order and the difference between the estimated time of delivery and the actual delivery date.**
- **These insights can be further used in decision-making for logistics to improve the delivery time.**
- **The reduced delivery time will help the business grow by retaining the customers and giving the edge to the competitors.**

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

Top 5 states with the highest freight value:

SQL Query:

```
SELECT
 c.customer_state AS State,
 ROUND(AVG(oi.freight_value),2) AS Average_freight_value
FROM
 TargetSQL.customers c
JOIN
 TargetSQL.orders o
ON
 c.customer_id=o.customer_id
JOIN
 TargetSQL.order_items oi
ON
 o.order_id=oi.order_id
WHERE
 o.order_delivered_customer_date IS NOT NULL
GROUP BY
 c.customer_state
ORDER BY
 Average_freight_value DESC
LIMIT
 5;
```

Query results

JOB INFORMATION | RESULTS | CHART | JS

Row	State	Average_freight_valu
1	PB	43.09
2	RR	43.09
3	RO	41.33
4	AC	40.05
5	PI	39.12

Top 5 states with the lowest freight value:

SQL Query:

```
SELECT
 c.customer_state AS State,
 ROUND(AVG(oi.freight_value),2) AS Average_freight_value
FROM
 TargetSQL.customers c
```

```
JOIN
 TargetSQL.orders o
ON
 c.customer_id=o.customer_id
JOIN
 TargetSQL.order_items oi
ON
 o.order_id=oi.order_id
WHERE
 o.order_delivered_customer_date IS NOT NULL
GROUP BY
 c.customer_state
ORDER BY
 Average_freight_value ASC
LIMIT
 5;
```

Query results

JOB INFORMATION | RESULTS | CHART | J

Row	State	Average_freight_valu
1	SP	15.11
2	PR	20.47
3	MG	20.63
4	RJ	20.91
5	DF	21.07

Inference:

- **This data gives us insights into the top 5 states with the highest and lowest freight value.**
- **These insights can be further used in decision-making to allocate resources effectively, optimize supply chain operations, and identify opportunities for cost savings or market expansion in underperforming regions.**

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

Top 5 states with lowest average delivery time:
SQL Query:

```
SELECT
 c.customer_state,
```

```
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_time
stamp, DAY)),2) AS avg_time_to_deliver
FROM
 TargetSQL.orders o
JOIN
 TargetSQL.customers c
ON
 o.customer_id=c.customer_id
WHERE
 o.order_delivered_customer_date IS NOT NULL
 AND o.order_purchase_timestamp IS NOT NULL
GROUP BY
 c.customer_state
ORDER BY
 avg_time_to_deliver DESC
LIMIT
 5;
```

Query results

JOB INFORMATION | RESULTS | CHART | JS

Row	customer_state	avg_time_to_deliver
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

Top 5 states with lowest average delivery time:
SQL Query:

```
SELECT
 c.customer_state,

ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_time
stamp, DAY)),2) AS avg_time_to_deliver
FROM
 TargetSQL.orders o
JOIN
 TargetSQL.customers c
ON
```

```
 o.customer_id=c.customer_id
WHERE
 o.order_delivered_customer_date IS NOT NULL
 AND o.order_purchase_timestamp IS NOT NULL
GROUP BY
 c.customer_state
ORDER BY
 avg_time_to_deliver
LIMIT
 5;
```

Query results

JOB INFORMATION | RESULTS | CHART | J:

Row	customer_state	avg_time_to_deliver
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

Inference:

- **This Analysis gives us information about the states with the highest and lowest average delivery times.**
- **These insights can be used to focus on underperforming regions by finding out the root cause and implementing changes in logistics and resource allocation.**

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 dates to figure out how fast the delivery was for each state.

   States with fast delivery:
   SQL Query:

```
SELECT
 c.customer_state,

ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_purchase_times
tamp, DAY)),2) AS avg_time_to_deliver
```

```
FROM
 TargetSQL.orders o
JOIN
 TargetSQL.customers c
ON
 o.customer_id=c.customer_id
WHERE
 o.order_estimated_delivery_date IS NOT NULL
 AND o.order_purchase_timestamp IS NOT NULL
GROUP BY
 c.customer_state
ORDER BY
 avg_time_to_deliver
LIMIT
 5;
```

Query results

JOB INFORMATION | RESULTS | CHART | JS

Row	customer_state	avg_time_to_deliver
1	SP	18.81
2	DF	24.06
3	MG	24.22
4	PR	24.25
5	ES	25.27

States with slow delivery:
SQL Query:

```
SELECT
 c.customer_state,

ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_purchase_times
tamp, DAY)),2) AS avg_time_to_deliver
FROM
 TargetSQL.orders o
JOIN
 TargetSQL.customers c
ON
 o.customer_id=c.customer_id
WHERE
 o.order_estimated_delivery_date IS NOT NULL
```

```
 AND o.order_purchase_timestamp IS NOT NULL
GROUP BY
 c.customer_state
ORDER BY
 avg_time_to_deliver DESC
LIMIT
 5;
```

Query results

JOB INFORMATION | RESULTS | CHART | JS

Row	customer_state	avg_time_to_deliver
1	RR	46.17
2	AP	45.71
3	AM	44.76
4	AC	40.77
5	RO	38.41

Inference:

- **This Analysis gives us information about the states with the fastest and slowest average delivery.**
- **These insights can be used to find out the best practices from the states with faster deliveries and implement them in regions with slower deliveries, thereby improving overall delivery efficiency and customer satisfaction.**

6. **Analysis based on the payments**
   1. **Find the month on month no. of orders placed using different payment types.**

SQL Query:

```
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_od_orders
FROM
 TargetSQL.orders o
JOIN
```

```
 TargetSQL.payments p
ON
 o.order_id=p.order_id
GROUP BY
 Year,
 Month,
 payment_type
ORDER BY
 Year,
 Month;
```

Query results

SAVE RESUL

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | E

Row	Year	Month	payment_type	No_od_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

Inference:

- **This Analysis gives us information on payment patterns each month based on payment type.**
- **These insights can be used to optimize payment processing strategies, tailor marketing efforts to preferred payment methods, and forecast the financials by understanding customer payment behaviours throughout the year.**

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

SQL Query:

```
SELECT
 payment_installments,
 COUNT(order_id) AS No_of_orders
FROM
```

```
 TargetSQL.payments
GROUP BY
 payment_installments
ORDER BY
 payment_installments;
```

Query results

JOB INFORMATION | RESULTS | CHA

Row	payment_installment	No_of_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644

Inference:

- **This Analysis gives us information on the number of orders based on the payment instalments.**
- **These insights can be used for marketing, by targeting promotions and offers to customers who prefer instalment payments, enhancing customer retention, and tailoring financing options to boost sales in segments that favour instalment plans.**