# Target-SQL Business Case

## Dataset exploratory Analysis

- Exploring the Data set, all the tables and its columns.
  This query will provide a list of columns for each specified table, including the column name, data type, and whether the column allows NULL values.

```
SELECT
  table_name,
  column_name,
  data_type,
  ordinal_position
FROM
`bigquery-scaler-tutorial.Target_SQL_business_case_2025.INFORMATION_SCHEMA.COLUMNS`
ORDER BY
  table_name,
  ordinal_position;
```

| Row | table_name | column_name | data_type | ordinal_position |
|---|---|---|---|---|
| 1 | customers | customer_id | STRING | 1 |
| 2 | customers | customer_unique_id | STRING | 2 |
| 3 | customers | customer_zip_code_prefix | INT64 | 3 |
| 4 | customers | customer_city | STRING | 4 |
| 5 | customers | customer_state | STRING | 5 |
| 6 | geolocation | geolocation_zip_code_prefix | INT64 | 1 |
| 7 | geolocation | geolocation_lat | FLOAT64 | 2 |
| 8 | geolocation | geolocation_lng | FLOAT64 | 3 |
| 9 | geolocation | geolocation_city | STRING | 4 |
| 10 | geolocation | geolocation_state | STRING | 5 |

```
SELECT table_name, COUNT(DISTINCT column_name) AS column_count FROM (SELECT
  table_name,
  column_name,
  data_type,
  ordinal_position
FROM
`bigquery-scaler-tutorial.Target_SQL_business_case_2025.INFORMATION_SCHEMA.COLUMNS`
ORDER BY
  table_name,
  ordinal_position) GROUP BY table_name
```

| Row | table_name | column_count |
|---|---|---|
| 1 | order_items | 7 |
| 2 | sellers | 4 |
| 3 | geolocation | 5 |
| 4 | products | 9 |
| 5 | orders | 8 |
| 6 | payments | 5 |
| 7 | customers | 5 |
| 8 | order_reviews | 6 |

**There are a total 8 columns in the data set.**

- Exploring the time range within which orders has been placed.

```
SELECT  MIN(order_purchase_timestamp) AS first_order_placed_date,
MAX(order_purchase_timestamp) AS last_order_placed_date
FROM `Target_SQL_business_case_2025.orders`
```

| Row | first_order_placed_date | last_order_placed_date |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-10-17 17:30:18 UTC |

**So First order was placed on 4th september 2016 at 15:19 UTC and Last order was Placed on 17th october 2018 17:30 UTC**

- Locations(cities and states) customers ordered during given time period

```
SELECT COUNT(DISTINCT customer_state) AS states, COUNT(DISTINCT customer_city) AS
city
FROM `Target_SQL_business_case_2025.customers`
WHERE customer_id IN (SELECT customer_id FROM
`Target_SQL_business_case_2025.orders`);
```

| Row | states | city |
|---|---|---|
| 1 | 27 | 4119 |

**So from 27 different states and 4,119 cities people has placed order Between  4th september 2016 to 17th october 2018 with Target.**

## In Depth Exploration

- Understanding the trend for orders placed for past years.

```
SELECT EXTRACT(year FROM order_purchase_timestamp) AS year_,
EXTRACT(month FROM order_purchase_timestamp) AS month_,
COUNT(DISTINCT order_id) AS orders
FROM `Target_SQL_business_case_2025.orders`
GROUP BY 1,2
ORDER BY 1,2;
```

| Row | year_ | month_ | 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 |

**From the data we can see the following**

- **Nov 2016 data is missing**
- **We can see an increase in no. of orders placed from jan 2017 which shows dip in november 2017.**
- **Again we can see an increase in no. of orders placed from jan 2018 which shows a steep fall after august 2018.**

**So we can see a monthly seasonality that there is an increase in no. of orders placed from jan to november then there is a drop in sales after november.**

- **Trying to find the time of the day when maximum orders gets placed**

```
SELECT
SUM(CASE WHEN HOUR BETWEEN 0 AND 6 THEN orders END ) AS dawn,
SUM(CASE WHEN HOUR BETWEEN 7 AND 12 THEN orders END ) AS Morning,
SUM(CASE WHEN HOUR BETWEEN 13 AND 18 THEN orders END ) AS afternoon ,
```

```
SUM(CASE WHEN HOUR BETWEEN 19 AND 24 THEN orders END ) AS night
FROM
(SELECT EXTRACT(HOUR FROM order_purchase_timestamp) AS HOUR,
COUNT(DISTINCT order_id) AS orders
FROM `Target_SQL_business_case_2025.orders`
GROUP BY 1)
```

| Row | dawn | Morning | afternoon | night |
|---|---|---|---|---|
| 1 | 5242 | 27733 | 38135 | 28331 |

**Since the bulk of orders are concentrated in the afternoon, we could offer attractive discounts or special promotions during off-peak hours to help balance the order volume throughout the day.**

## Evolution of E-commerce orders in the Brazil region:

Analyzing the month on month no. of orders placed in each state of brazil.

```
SELECT c.customer_state, EXTRACT(month FROM o.order_purchase_timestamp) AS
order_month, COUNT(DISTINCT o.order_id) AS orders
FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.customers`c
ON o.customer_id = c.customer_id
GROUP BY 1,2
ORDER BY 1,2;
```

| Row | customer_state | order_month | orders |
|---|---|---|---|
| 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 |

```sql
/*Analysing the distribution of customers across state*/
SELECT customer_state, COUNT(DISTINCT customer_id) AS customer_count,
FROM `Target_SQL_business_case_2025.customers`
GROUP BY customer_state
ORDER BY customer_count 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 |

**So SP has the maximum number of customers followed by RJ and MG in 2nd and 3rd place.**
**RR has the least no. of customers.**

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

- Analyzing the cost of order from 2017 to 2018 and only from jan to aug because as per previous analysis this is the peak time

```sql
WITH base AS(
  SELECT EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year_,
SUM(p.payment_value) AS cost
  FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.payments` p ON o.order_id = p.order_id
  WHERE (EXTRACT(YEAR FROM o.order_purchase_timestamp) BETWEEN 2017 AND 2018) AND
(EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8)
  GROUP BY 1
),
```

```
base1 AS(
  SELECT *, LEAD(cost, 1) OVER(ORDER BY year_) AS next_year_cost
  FROM base
)
SELECT *, ROUND((next_year_cost-cost)/cost*100, 2) AS percentage_increase FROM
base1;
```

| Row | year_ | cost | next_year_cost | percentage_increase |
|---|---|---|---|---|
| 1 | 2017 | 3669022.119999… | 8694733.840000… | 136.98 |
| 2 | 2018 | 8694733.840000… | null | null |

**Hence, there is a 137% increase in the cost of orders placed**

- Checking Total & Average value of order price for each states of brazil.

```
SELECT c.customer_state,ROUND(SUM(p.payment_value),2) AS total_cost,
ROUND(AVG(p.payment_value), 2) AS avg_cost
  FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.payments` p ON o.order_id = p.order_id
  JOIN `Target_SQL_business_case_2025.customers` c ON c.customer_id = o.customer_id
  GROUP BY 1
  ORDER BY 2 DESC,3 DESC;
```

| Row | customer_state | total_cost | avg_cost |
|---|---|---|---|
| 1 | SP | 5998226.96 | 137.5 |
| 2 | RJ | 2144379.69 | 158.53 |
| 3 | MG | 1872257.26 | 154.71 |
| 4 | RS | 890898.54 | 157.18 |
| 5 | PR | 811156.38 | 154.15 |
| 6 | SC | 623086.43 | 165.98 |
| 7 | BA | 616645.82 | 170.82 |
| 8 | DF | 355141.08 | 161.13 |
| 9 | GO | 350092.31 | 165.76 |
| 10 | ES | 325967.55 | 154.71 |

**So the total cost of orders is highest in the SP state of brazil.**

- Calculating the Total & Average value of order freight for each state.

```
  SELECT c.customer_state,ROUND(SUM(oi.freight_value),2) AS total_freight_cost,
ROUND(AVG(oi.freight_value), 2) AS avg_freight_cost
```

```
  FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.order_items` oi ON o.order_id = oi.order_id
  JOIN `Target_SQL_business_case_2025.customers` c ON c.customer_id = o.customer_id
  GROUP BY 1
  ORDER BY 2 DESC,3 DESC;
```

| Row | customer_state | total_freight_cost | avg_freight_cost |
|---|---|---|---|
| 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 |

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

- Finding no. of days taken to deliver each order from the order's purchase date as delivery time.
- Also, calculating the difference (in days) between the estimated & actual delivery date of an order.

```
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_SQL_business_case_2025.orders`
```

| Row | order_id | time_to_deliver | diff_estimated_deliv |
|---|---|---|---|
| 1 | 65d1e226dfaeb8cdc42f66542... | 35 | -16 |
| 2 | 2c45c33d2f9cb8ff8b1c86cc28... | 30 | -28 |
| 3 | 1950d777989f6a877539f5379... | 30 | 12 |
| 4 | bfbd0f9bdef84302105ad712db... | 54 | 36 |
| 5 | 98974b076b01553d49ee6467... | 43 | -6 |
| 6 | c4b41c36dd589e901f6879f25... | 36 | -14 |
| 7 | d2292ff2201e74c5db154d1b7... | 29 | -20 |
| 8 | 95e01270fcbae986342340010... | 30 | -19 |
| 9 | ed8c7b1b3eb256c70ce0c7423... | 44 | -5 |
| 10 | 5cc475c7c03290048eb2e742c... | 68 | 18 |

```
let's find out the top 5 states with the highest & lowest average freight value.
(SELECT c.customer_state, ROUND(AVG(oi.freight_value), 2) AS avg_freight_val
FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.order_items` oi ON o.order_id = oi.order_id
JOIN `Target_SQL_business_case_2025.customers` c ON c.customer_id = o.customer_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5)
UNION ALL
(SELECT c.customer_state, ROUND(AVG(oi.freight_value), 2) AS avg_freight_val
FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.order_items` oi ON o.order_id = oi.order_id
JOIN `Target_SQL_business_case_2025.customers` c ON c.customer_id = o.customer_id
GROUP BY 1
ORDER BY 2
LIMIT 5)
```

| Row | customer_state | avg_freight_val |
|---|---|---|
| 1 | RR | 42.98 |
| 2 | PB | 42.72 |
| 3 | RO | 41.07 |
| 4 | AC | 40.07 |
| 5 | PI | 39.15 |
| 6 | SP | 15.15 |
| 7 | PR | 20.53 |
| 8 | MG | 20.63 |
| 9 | RJ | 20.96 |
| 10 | DF | 21.04 |

**So, highest avg freight val is for state RR and lowest is for SP**

- Let's find the top 5 states with the highest & lowest average delivery time.

```
(SELECT c.customer_state, ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_purchase_timestamp, DAY)), 2) AS avg_time_to_deliver
FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5)
UNION ALL
(SELECT c.customer_state, ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_purchase_timestamp, DAY)), 2) AS avg_time_to_deliver
FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY 2
LIMIT 5);
```

| 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 |
| 6 | SP | 8.3 |
| 7 | PR | 11.53 |
| 8 | MG | 11.54 |
| 9 | DF | 12.51 |
| 10 | SC | 14.48 |

- Let's find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,
o.order_delivered_customer_date, DAY)),2) AS avg_diff_estimated_delivery
FROM `Target_SQL_business_case_2025.orders` o JOIN
`Target_SQL_business_case_2025.customers`c
ON o.customer_id = c.customer_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5;
```

| Row | customer_state | avg_diff_estimated_c |
|---|---|---|
| 1 | AC | 19.76 |
| 2 | RO | 19.13 |
| 3 | AP | 18.73 |
| 4 | AM | 18.61 |
| 5 | RR | 16.41 |

**These are the top 5 states where the order reached the customer before the estimated delivery time.**

## Analysis based on the payments:

- Let's find the month on month no. of orders placed using different payment types.

```
SELECT COUNT(DISTINCT o.order_id), EXTRACT(MONTH FROM  o.order_purchase_timestamp)
AS order_month, p.payment_type
FROM `Target_SQL_business_case_2025.payments` p JOIN
`Target_SQL_business_case_2025.orders` o
ON p.order_id = o.order_id
GROUP BY 2,3
ORDER BY 1 DESC, 2;
```

| Row | f0_ | order_month | payment_type |
|---|---|---|---|
| 1 | 8308 | 5 | credit_card |
| 2 | 8235 | 8 | credit_card |
| 3 | 7810 | 7 | credit_card |
| 4 | 7682 | 3 | credit_card |
| 5 | 7276 | 4 | credit_card |
| 6 | 7248 | 6 | credit_card |
| 7 | 6582 | 2 | credit_card |
| 8 | 6093 | 1 | credit_card |
| 9 | 5867 | 11 | credit_card |
| 10 | 4364 | 12 | credit_card |

**So the highest orders were placed through credit cards.**

- let's find the no. of orders placed on the basis of the payment installments that have been paid.

```
SELECT COUNT(DISTINCT o.order_id), p.payment_installments
FROM `Target_SQL_business_case_2025.payments` p JOIN
`Target_SQL_business_case_2025.orders` o
ON p.order_id = o.order_id
GROUP BY 2
ORDER BY 1 DESC;
```

| Row | f0_ | payment_installment |
|---|---|---|
| 1 | 49060 | 1 |
| 2 | 12389 | 2 |
| 3 | 10443 | 3 |
| 4 | 7088 | 4 |
| 5 | 5315 | 10 |
| 6 | 5234 | 5 |
| 7 | 4253 | 8 |
| 8 | 3916 | 6 |
| 9 | 1623 | 7 |
| 10 | 644 | 9 |

**So, max orders were placed on 1st payments installment.**

## Summary

Here is the complete summary of insights from the data and SQL queries used to find them.

1. **Dataset Exploratory Analysis**
    - **Insight:** The dataset includes multiple tables, each containing various columns with specific data types and constraints.
    - **SQL Use:** Queried `INFORMATION_SCHEMA.COLUMNS` to retrieve metadata about tables, including column names, data types, and NULL constraints.
2. **In-Depth Exploration**
    - **Insight:** Orders exhibit a seasonal trend, increasing from January to November, then declining after.
    - **SQL Use:** Used `EXTRACT(YEAR, MONTH)` and `COUNT(DISTINCT order_id)` to analyze order volume trends over time.
3. **Evolution of E-commerce Orders in Brazil**
    - **Insight:** Orders were placed from 27 states and over 4,000 cities, with SP having the most customers.

- **SQL Use:** Used `COUNT(DISTINCT customer_city, customer_state)` to assess geographical distribution.

4. **Impact on Economy**
    - **Insight:** Order costs increased by 137% between 2017 and 2018, with SP having the highest total order cost.
    - **SQL Use:** Used `SUM(payment_value)`, `AVG(payment_value)`, and `LEAD()` window function to analyze cost trends.
5. **Analysis Based on Sales, Freight, and Delivery Time**
    - **Insight:** Delivery times varied significantly, with RR having the highest average freight cost and SP the lowest.
    - **SQL Use:** Used `DATE_DIFF()` to calculate actual vs. estimated delivery times and `AVG(freight_value)` to compare freight costs.
6. **Analysis Based on Payments**
    - **Insight:** Most payments were made via credit cards, and the majority of orders were paid in a single installment.
    - **SQL Use:** Used `COUNT(DISTINCT order_id)`, `GROUP BY payment_type, payment_installments` to analyze payment preferences.