Target E-commerce Data Analysis Case Study

Author-

Rohan Nayak

Aspiring Data Analyst | SQL

Tools Used

- **SQL (BigQuery)**
- **MS Word (documentation)**

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.

QUERY FOR THIS

```
SELECT

column_name,

 data_type

FROM target.INFORMATION_SCHEMA.COLUMNS

WHERE  table_name = 'customers'
```

OUTPUT

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 – N/A

RECCOMENDATION – N/A

ASSUMPTION- N/A

1B. Get the time range between which the orders were placed.

QUERY FOR THIS

```
SELECT
  MIN(EXTRACT(year
    FROM order_purchase_timestamp)) AS min_year,
  MAX(EXTRACT(year
    FROM order_purchase_timestamp)) AS max_year
FROM target.orders
```

OUTPUT

Row	min_year	max_year
1	2016	2018

INSIGHT - Identifies the business activity period

RECCOMENDATION - Analyze yearly/monthly order growth

ASSUMPTION - N/A

1C. Count the Cities & States of customers who ordered during the given period.

QUERY FOR THIS

```
SELECT
  COUNT(DISTINCT geolocation_city) AS city,
  COUNT(DISTINCT geolocation_state) AS state
FROM target.geolocation
```

OUTPUT

Row	city	state
1	8011	27

INSIGHT

- Customers who placed orders come from 8,011 distinct cities across 27 states.
- This shows very wide geographic reach and strong national penetration of the platform.

RECCOMENDATION

- Target High-Potential Locations

ASSUMPTION - N/A

II. in-depth Exploration:

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

Query for this

```
WITH
 cte AS(
 SELECT
  order_id,
  EXTRACT(year
  FROM order_purchase_timestamp) AS year,
  EXTRACT(month
  FROM order_purchase_timestamp) AS month
 FROM target.orders )
SELECT
 year,
 month,
 COUNT(order_id) AS order_count
FROM
 cte
GROUP BY
 Year , month
ORDER BY
 Year ,  month
```

output

Row	year	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

INSIGHT

- year over year it shows the volume of the trend If it's fluctuate, is its declines etc.

RECCOMENDATION

– if orders are increasing Improve inventory, boost high performing month/ year , invest in marketing.

ASSUMPTION

- Timestamp Column is Clean and Valid , No missing, null, or corrupted timestamps exist , we assume timestamps are stored in a uniform timezone (usually UTC). No timezone conversion is needed before extracting YEAR/MONTH.

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

Query for this

```
WITH
  cte AS(
  SELECT
    EXTRACT(month
    FROM order_purchase_timestamp) AS month,
    EXTRACT(year
    FROM order_purchase_timestamp) AS year,
    *
  FROM  target.orders)
SELECT
  year,
  month,
  COUNT (order_id) AS total_ord
FROM
  cte
GROUP BY
  year,
  month
ORDER BY year,
  month DESC;
```

output

Row	year	month	total_ord
1	2016	12	1
2	2016	10	324
3	2016	9	4
4	2017	12	5673
5	2017	11	7544
6	2017	10	4631
7	2017	9	4285
8	2017	8	4331
9	2017	7	4026
10	2017	6	3245

INSIGHTS

- we get a clear view of how order count changes month-to-month across different years.
- Also , sales dips in certain months
- And yearly growth patterns

Assumptions

- We assumes no NULL or corrupted timestamp values.
  If timestamps are missing, those orders won't be counted. Assumes timestamps do not require conversion (e.g., from UTC to local time).

Recommendation

- If many timestamps are NULL, monthly counts will be incorrect.

---

2 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**

Query for this

```
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 time_hour,

  COUNT(*) AS total_orders

FROM target.orders

GROUP BY time_hour

ORDER BY total_orders DESC;
```

output

Row	time_hour	total_orders
1	afternoon	38135
2	night	28331
3	mornings	27733
4	dawn	5242

INSIGHT

– This grouping clearly shows when users are **most active** on the platform.
The highest value in total_orders gives the peak window.

Night peaks may suggest:

- Younger users
- Relaxed browsing time
- Offers/flash sales in late evening

Morning peaks may indicate:

- Routine ordering (e.g., groceries)

RECCOMENDATION

– If most orders occur at a particular time (e.g., night), we have to optimise our business accordingly

- Customer support staffing
- Warehouse load handling
- Server performance during peak traffic

ASSUMPTION

- order_purchase_timestamp reflects the accurate time the customer placed the order . Assumes that this timestamp captures the **actual purchase moment**, not processing, approval, or delivery time.

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

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

Query for this

```
WITH cte AS (
   SELECT c.customer_state,
      EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
      EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year
   FROM target.orders o
   JOIN target.customers c
      ON c.customer_id = o.customer_id )
SELECT year , month,
   COUNT(customer_state) AS total_orders
FROM cte
GROUP BY year, month
ORDER BY year, month DESC;
```

Output

Row	year	month	total_orders
1	2016	12	1
2	2016	10	324
3	2016	9	4
4	2017	12	5673
5	2017	11	7544
6	2017	10	4631
7	2017	9	4285
8	2017	8	4331
9	2017	7	4026

INSIGHT

- Identifying peak months of order activity.

November–January shows high orders → Year-end shopping season trend.

March 2017 orders vs March 2018 orders → check growth/decline.

RECCOMENDATION

- Increase marketing budgets during those high-performing months.

Launch seasonal promotions (festivals, payday sales, holiday sales).

Increase inventory before the months that show consistently high orders.

ASSUMPTIONS

- Orders increase during holidays, festivals (Diwali, Christmas, New Year).

Orders decrease during non-festive months or off-season periods.

Some states may have stronger demand because of:

- Higher population density
- Better delivery infrastructure

---

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

Query for this

```
select customer_state , count(distinct(customer_id)) as total_cus
from target.customers
group by customer_state
order by total_cus desc;
```

output

Row	customer_state	total_cus
1	SP	Open sort menu
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033

INSIGHT

- states with the Highest Customer Base Are Your Key Markets.
- Some states may have significantly fewer customers. Indicates unequal market penetration—your business is stronger in certain regions.

RECCOMENDATION

– Focus More Marketing on Top-Performing States

- Run targeted digital ads.
- Offer loyalty programs for customers in those states.

Improve Logistics & Delivery in Low-Performing States

If customer count is low:

- Improve delivery speed.
- Reduce shipping costs.
- Partner with local delivery agencies.

ASSUMPTION- N/A

---

4. 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).

Query for this

```
WITH cte AS(
  SELECT EXTRACT ( YEAR
    FROM o.order_purchase_timestamp) AS year_order_purchase,
```

```
  ROUND (SUM(p.payment_value),2) AS total_payments
 FROM target.order_info o
 JOIN target.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 EXTRACT(YEAR
  FROM o.order_purchase_timestamp))
SELECT ((
  SELECT total_payments
  FROM cte
  WHERE year_order_purchase = 2018) - (
  SELECT total_payments
  FROM cte
  WHERE year_order_purchase = 2017)) / (
 SELECT total_payments
 FROM cte
 WHERE year_order_purchase = 2017) * 100 AS new_perctg;
```

output

Row	new_perctg
1	136.9768716466...

INSIGHT

The total cost of orders increased by ~137% from 2017 to 2018 (Jan–Aug).

This indicates very strong growth in e-commerce adoption and customer spending within just one year.

RECOMMENDATION

Invest more in high-performing states and categories driving this growth.

ASSUMPTION

All completed and valid payment records are included; cancelled or missing payments are excluded.

4B. Calculate the Total & Average value of order price for each state

QUERY FOR THIS

```
SELECT
  c.customer_state,
  SUM(oi.price) AS total_order_price,
  AVG(oi.price) AS avg_order_price
FROM `target.order_item` 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 customer_state
ORDER BY total_order_price DESC;
```

Output

Row	customer_state	total_order_price	avg_order_price
1	SP	5202955.049999...	109.6536291597...
2	RJ	1824092.669999...	125.1178180945...
3	MG	1585308.030000...	120.7485741488...
4	RS	750304.0200000...	120.3374530874...
5	PR	683083.760000001	119.0041393728...
6	SC	520553.3399999...	124.6535775862...
7	BA	511349.9899999...	134.6012082126...

INSIGHTS

- Sao Paulo (SP) contributes the highest total order value, making it the largest revenue-generating state.
- States like BA and RJ show higher average order value, indicating customers place higher-ticket purchases despite lower total volume.

RECOMMENDATION

- Focus marketing and seller expansion in SP to further scale revenue.
- Promote premium products and EMI offers in states with higher average order value (BA, RJ).

ASSUMPTION

- price represents the actual product price excluding freight.
- Aggregation is done at state level using customer location.

4C. Calculate the Total & Average value of order freight for each state.

QUERY FOR THIS

```
SELECT c.customer_state,
  SUM (oi.freight_value) AS total_freight,
  AVG (oi.freight_value) AS avg_freight
FROM `target.order_item` 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 customer_state
ORDER BY avg_freight DESC;
```

Output

Row	customer_state	total_freight	avg_freight
1	RR	2235.189999999...	42.98442307692...
2	PB	25719.73	42.72380398671...
3	RO	11417.38000000...	41.06971223021...
4	AC	3686.749999999...	40.07336956521...
5	PI	21218.20000000...	39.14797047970...
6	MA	31523.76999999...	38.25700242718...
7	TO	11732.68	37.24660317460...

INSIGHTS

- States like RR, PB, and RO have the highest average freight cost, indicating higher logistics expenses.
- These states likely represent geographically remote or low-density regions, increasing delivery complexity.

RECCOMENDATION

- Optimize logistics routes or partner with local delivery providers in high-freight states.
- Introduce minimum order value thresholds for free or discounted shipping.

ASSUMPTION

- All delivered orders are included without excluding any product category.
- freight_value represents the delivery cost per item.

---

5. 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 orders.

QUERY FOR THIS

```
SELECT o.order_id, c.customer_state,

DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY)
AS time_to_deliver,

DATE_DIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date,
DAY) AS diff_estimated_delivery

FROM `target.orders` o

JOIN `target.customers` c

  ON o.customer_id = c.customer_id

WHERE o.order_delivered_customer_date IS NOT NULL;
```

Output

Row	order_id	customer_state	time_to_deliver	diff_estimated_d...
1	65d1e226dfaeb8cdc42f665422...	RJ	35	16
2	2c45c33d2f9cb8ff8b1c86cc28...	SC	30	28
3	1950d777989f6a877539f53795...	MG	30	-12
4	bfbd0f9bdef84302105ad712db...	SP	54	-36
5	98974b076b01553d49ee64679...	SP	43	6
6	c4b41c36dd589e901f6879f25a...	SP	36	14
7	d2292ff2201e74c5db154d1b7a...	SP	29	20

INSIGHTS

- Delivery time varies significantly across states, ranging from ~29 to 54 days.
- Some orders were delivered earlier than estimated (positive difference), while others were delayed (negative difference).

RECOMMENDATION

- Improve delivery time consistency by analyzing delayed orders and identifying bottlenecks.
- Adjust estimated delivery dates for regions with frequent delays to improve customer trust.

ASSUMPTION

- time_to_deliver is calculated as order_delivered_customer_date – order_purchase_timestamp.
- diff_estimated_delivery represents the difference between estimated and actual delivery date.

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

QUERY FOR

----5 highest states----

```
SELECT c.customer_state,
  AVG(oi.freight_value) AS avg_freight
FROM `target.order_item` 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 customer_state
ORDER BY avg_freight DESC
LIMIT 5;
```

----5 lowest sate----

```
ORDER BY avg_freight ASC
LIMIT 5;
```

Output 5 highest sates

Row	customer_state	avg_freight
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...

Output 5 lowest sates

Row	customer_state	avg_freight
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...

INSIGHTS 5 LOWEST AVERAGE

- Sao Paulo (SP) has the lowest average freight cost, benefiting from strong logistics infrastructure.
- States like PR, MG, RJ, and DF also show relatively low freight charges, indicating better seller proximity and connectivity.

INSIGHTS 5 HIGHEST AVERAGE

- RR, PB, RO, AC, and PI have the highest average freight costs among all states.
- These states are generally geographically remote or less densely connected, leading to higher transportation expenses.

RECOMMENDATION FOR BOTH

- Leverage these states as logistics hubs or fulfillment centers.(LOWEST)
- Promote free or discounted shipping offers in low-freight regions to drive more volume.(LOWEST)
- Explore regional fulfillment centers or last-mile delivery partners in high-freight states.(HIGHEST)

ASSUMPTION FOR BOTH

- Freight cost is analyzed based on the customer's state.
- avg_freight is calculated using the average freight value per order item.

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

QUERY FOR

----5 highest sates----

```
SELECT c.customer_state,
  AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY))
    AS avg_delivery_days
FROM `target.orders` o
JOIN `target.customers` c
  ON o.customer_id = c.customer_id
WHERE o.order_delivered_customer_date IS NOT NULL
GROUP BY customer_state
ORDER BY avg_delivery_days DESC
LIMIT 5;
```

----5 lowest sates----

```
ORDER BY avg_delivery_days ASC
LIMIT 5;
```

Output 5 highest sate

Row	customer_state	avg_delivery_days
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...
4	AL	24.04030226700...
5	PA	23.31606765327...

Output 5 lowest sales

Row	customer_state	avg_delivery_days
1	SP	8.298061489072...
2	PR	11.52671135486...
3	MG	11.54381329810...
4	DF	12.50913461538...
5	SC	14.47956019171...

INSIGHTS FOR 5 LOWEST SALES

- Sao Paulo (SP) has the fastest average delivery time, reflecting excellent logistics efficiency.
- States like PR, MG, DF, and SC also demonstrate quick and reliable deliveries.

INSIGHTS FOR 5 HIGHEST SALES

- RR, AP, AM, AL, and PA experience the longest average delivery times, with deliveries taking over 23–29 days.

- These states are largely remote or geographically challenging, leading to longer transit durations.

RECOMMENDATION FOR BOTH

- Establish regional warehouses or micro-fulfillment centers in or near these states. (HIGHEST)
- Partner with local logistics providers to improve last-mile delivery. (HIGHEST)
- Use these states as benchmarks for delivery performance.(LOWEST)
- Expand same-day or next-day delivery pilots in fast-performing regions. (LOWEST)

ASSUMPTION

- avg_delivery_days is calculated using order_delivered_customer_date – order_purchase_timestamp.

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

QUERY FOR THIS

```
SELECT c.customer_state,

  AVG(DATE_DIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date, DAY)) AS avg_days_early

FROM `target.orders` o

JOIN `target.customers` c

  ON o.customer_id = c.customer_id

WHERE o.order_delivered_customer_date IS NOT NULL

GROUP BY customer_state

ORDER BY avg_days_early DESC

LIMIT 5;
```

Output

Row	customer_state	avg_days_early
1	AC	19.76249999999...
2	RO	19.13168724279...
3	AP	18.73134328358...
4	AM	18.60689655172...
5	RR	16.41463414634...

INSIGHTS

- AC, RO, AP, AM, and RR consistently deliver orders 15–20 days earlier than the estimated delivery date.
- This suggests that delivery estimates in these states are highly conservative.

RECOMMENDATION

- Recalibrate estimated delivery dates to be more realistic in these states.
- Promote faster-than-promised delivery as a competitive advantage.

ASSUMPTION

- Only delivered orders with valid estimated dates are included.
- Positive values indicate early delivery.

---

6. Analysis based on the payments:

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

QUERY FOR THIS

```
SELECT

  EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,

  EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,p.payment_type,

  COUNT(DISTINCT o.order_id) AS total_orders

FROM `target.orders` o

JOIN `target.payments` p

  ON o.order_id = p.order_id

GROUP BY year, month, payment_type

ORDER BY year, month, total_orders DESC;
```

Output

Row	year	month	payment_type	total_orders
1	2016	9	credit_card	3
2	2016	10	credit_card	253
3	2016	10	UPI	63
4	2016	10	voucher	11
5	2016	10	debit_card	2
6	2016	12	credit_card	1
7	2017	1	credit_card	582

INSIGHTS

- Credit cards dominate across all months, making them the most preferred payment method.
- Alternative payment methods like UPI, vouchers, and debit cards contribute a smaller but growing share.

RECOMMENDATIONS

- Strengthen credit card partnerships and promotional offers.
- Encourage adoption of UPI and digital wallets through discounts or cashback.
- Optimize checkout flow to highlight popular payment options.

ASSUMPTIONS

- Each order is counted once per payment type using distinct order_id.

6B. Find the no. of orders placed on the basis of the payment installments that have been paid.

QUERY FOR THIS

```
SELECT payment_installments,
COUNT(DISTINCT order_id) AS total_orders
FROM `target.payments`
GROUP BY payment_installments
ORDER BY payment_installments;
```

Output

Row	payment_installm...	total_orders
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916

INSIGHTS

- Single-payment orders (1 installment) dominate, accounting for the largest share of transactions.
- As the number of installments increases, the number of orders steadily declines.
- Very few orders are placed with 0 installments, indicating limited edge cases or data anomalies.

RECOMMENDATION

- Promote single-payment incentives (cashback, discounts) to increase faster revenue realization.
- Offer targeted EMI schemes (2–4 installments) for higher-value products.

ASSUMPTION

- payment_installments indicates the number of installments chosen per order.
- Each order is counted once, regardless of payment sequence.