Business Case: Target SQL

Scaler DS ML

Shayantan Dey

Sun 21st July

Context:

Target is a globally renowned brand and a prominent retailer in the United States. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver.

This particular business case focuses on the operations of Target in Brazil and provides insightful information about 100,000 orders placed between 2016 and 2018. The dataset offers a comprehensive view of various dimensions including the order status, price, payment and freight performance, customer location, product attributes, and customer reviews.

By analyzing this extensive dataset, it becomes possible to gain valuable insights into Target's operations in Brazil. The information can shed light on various aspects of the business, such as order processing, pricing strategies, payment and shipping efficiency, customer demographics, product characteristics, and customer satisfaction levels.

Dataset:

The data is available in 8 csv files at Google Drive

1. customers.csv
2. sellers.csv
3. order_items.csv
4. geolocation.csv
5. payments.csv
6. reviews.csv
7. orders.csv
8. products.csv

The column description for these csv files is given below.

The **customers.csv** contain following features:

Features	Description
customer_id	ID of the consumer who made the purchase
customer_unique_id	Unique ID of the consumer
customer_zip_code_prefix	Zip Code of consumer's location
customer_city	Name of the City from where order is made
customer_state	State Code from where order is made (Eg. são paulo - SP)

The **sellers.csv** contains following features:

Features	Description
seller_id	Unique ID of the seller registered
seller_zip_code_prefix	Zip Code of the seller's location
seller_city	Name of the City of the seller
seller_state	State Code (Eg. são paulo - SP)

The **order_items.csv** contain following features:

Features	Description
order_id	A Unique ID of order made by the consumers
order_item_id	A Unique ID given to each item ordered in the order
product_id	A Unique ID given to each product available on the site
seller_id	Unique ID of the seller registered in Target
shipping_limit_date	The date before which the ordered product must be shipped
price	Actual price of the products ordered
freight_value	Price rate at which a product is delivered from one point to another

The **geolocations.csv** contain following features:

Features	Description
geolocation_zip_code_prefix	First 5 digits of Zip Code
geolocation_lat	Latitude
geolocation_lng	Longitude
geolocation_city	City
geolocation_state	State

The **payments.csv** contain following features:

Features	Description
order_id	A Unique ID of order made by the consumers
payment_sequential	Sequences of the payments made in case of EMI
payment_type	Mode of payment used (Eg. Credit Card)
payment_installments	Number of installments in case of EMI purchase
payment_value	Total amount paid for the purchase order

The **orders.csv** contain following features:

Features	Description
order_id	A Unique ID of order made by the consumers
customer_id	ID of the consumer who made the purchase
order_status	Status of the order made i.e. delivered, shipped, etc.
order_purchase_timestamp	Timestamp of the purchase
order_delivered_carrier_date	Delivery date at which carrier made the delivery
order_delivered_customer_date	Date at which customer got the product
order_estimated_delivery_date	Estimated delivery date of the products

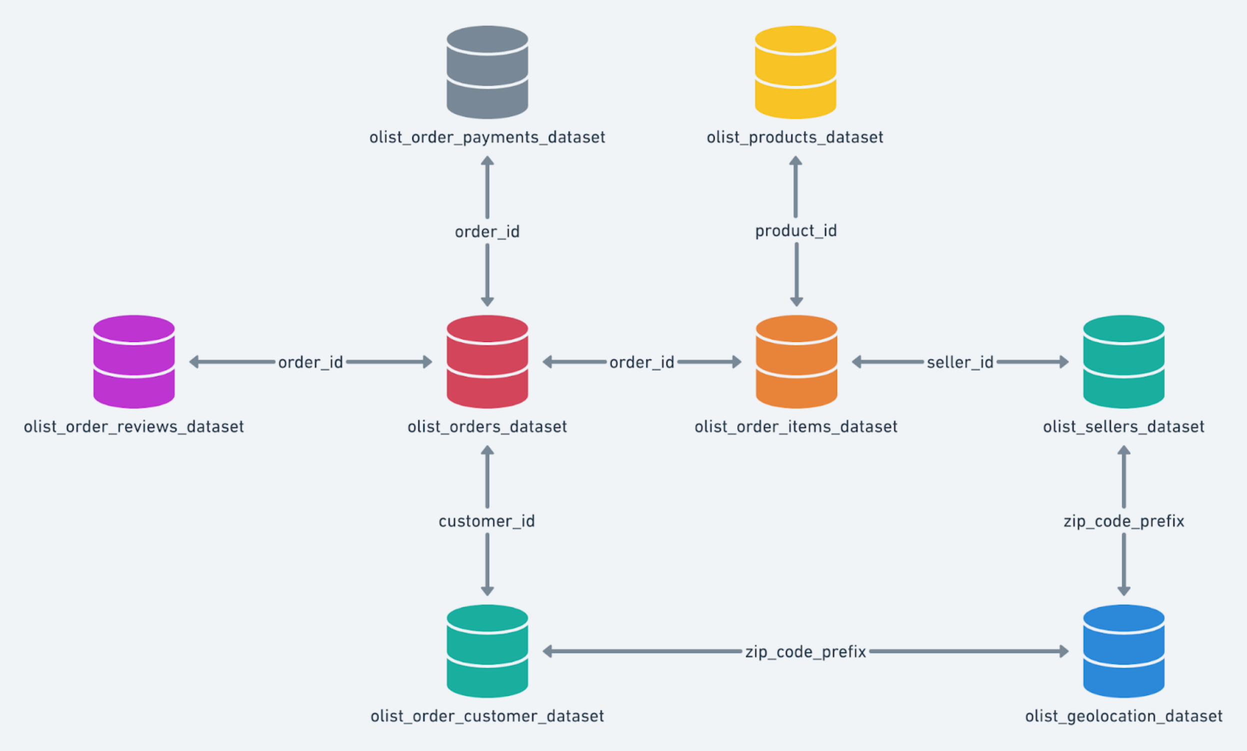
The **reviews.csv** contain following features:

Features	Description
review_id	ID of the review given on the product ordered by the order id
order_id	A Unique ID of order made by the consumers
review_score	Review score given by the customer for each order on a scale of 1-5
review_comment_title	Title of the review
review_comment_message	Review comments posted by the consumer for each order
review_creation_date	Timestamp of the review when it is created
review_answer_timestamp	Timestamp of the review answered

The **products.csv** contain following features:

Features	Description
product_id	A Unique identifier for the proposed project
product_category_name	Name of the product category
product_name_lenght	Length of the string which specifies the name given to the products ordered
product_description_lenght	Length of the description written for each product ordered on the site
product_photos_qty	Number of photos of each product ordered available on the shopping portal
product_weight_g	Weight of the products ordered in grams
product_length_cm	Length of the products ordered in centimeters
product_height_cm	Height of the products ordered in centimeters
product_width_cm	Width of the product ordered in centimeters

Dataset schema:



Observations in the dataset

Two files, order_reviews.csv and geolocation.csv had unclean data.

Issues Identified in the order_reviews.csv file:

Encoding Issue: The file had to be read with ISO-8859-1 encoding instead of UTF-8.

Null Values: The review_comment_title column has many null values.

Date and Time Formatting: The review_creation_date and review_answer_timestamp columns are in string format and not properly parsed as datetime objects.

Steps to Correct Issues:

1. Ensure consistent encoding.
2. Handle null values in review_comment_title.
3. Convert date and time columns to proper datetime format.

Cleaning Data:

1. Strip leading/trailing spaces in text fields.
2. Replace any special characters or non-UTF-8 characters in text fields.
3. Check for null or empty values and handle them appropriately.

4. Convert date and time columns to datetime format.

Issues Identified in the geolocation.csv file:

Encoding Issue: The file had to be read with ISO-8859-1 encoding instead of UTF-8.

Null Values: The review_comment_title column has many null values.

Date and Time Formatting: The review_creation_date and review_answer_timestamp columns are in string format and not properly parsed as datetime objects.

Steps to Correct Issues:

1. Special characters in text fields.
2. Trailing or leading spaces.
3. Null or empty values.
4. Ensure that the file does not have any rows that might cause issues.

Cleaning Data:

1. Strip leading/trailing spaces in text fields.
2. Replace any special characters or non-UTF-8 characters in text fields.
3. Check for null or empty values and handle them appropriately.

Problem Statement:

Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

What does 'good' look like?

1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

1.1. Data type of all columns in the "customers" table.

```
DESCRIBE customers;
```

Table 9: 5 records

Field	Type	Null	Key	Default	Extra
customer_id	text	YES		NA	
customer_unique_id	text	YES		NA	
customer_zip_code_prefix	text	YES		NA	
customer_city	text	YES		NA	
customer_state	text	YES		NA	

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

```sql
SELECT
    MIN(order_purchase_timestamp) AS order_start_date,
    MAX(order_purchase_timestamp) AS order_end_date,
    DATEDIFF(MAX(order_purchase_timestamp), MIN(order_purchase_timestamp))
      AS order_time_range_days
FROM
    orders;
```

Table 10: 1 records

order_start_date	order_end_date	order_time_range_days
2016-09-04 21:15:19	2018-10-17 17:30:18	773

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

```sql
SELECT DISTINCT c.customer_city, c.customer_state, COUNT(*) AS customer_count
FROM orders AS o
JOIN customers AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_city, c.customer_state
ORDER BY customer_count DESC
```

Table 11: Displaying records 1 - 10

customer_city	customer_state	customer_count
sao paulo	SP	15540
rio de janeiro	RJ	6882
belo horizonte	MG	2773
brasilia	DF	2131
curitiba	PR	1521
campinas	SP	1444
porto alegre	RS	1379
salvador	BA	1245
guarulhos	SP	1189
sao bernardo do campo	SP	938

2. In-depth Exploration:

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

The purchases were made in the year 2016, 2017 and 2018.

```sql
SELECT DISTINCT YEAR(order_purchase_timestamp) AS year_of_orders
FROM orders
ORDER BY year_of_orders;
```

Table 12: 3 records

year_of_orders
2016
2017
2018

Trend for 2016 does not show conclusive evidence of a growing trend.

```sql
SELECT DISTINCT MONTHNAME(order_purchase_timestamp) as month,
       MONTH(order_purchase_timestamp) as month_number,
       COUNT(order_id) OVER (PARTITION BY MONTH(order_purchase_timestamp))
        AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2016 AND LOWER(order_status) <> 'canceled'
ORDER BY MONTH(order_purchase_timestamp);
```

Table 13: 3 records

month	month_number	order_count
September	9	2
October	10	300
December	12	1

Trend for 2017 shows growth in month-on-month sale throughout the year.

```sql
SELECT DISTINCT MONTHNAME(order_purchase_timestamp) as month,
       MONTH(order_purchase_timestamp) as month_number,
       COUNT(order_id) OVER (PARTITION BY MONTH(order_purchase_timestamp))
        AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2017 AND LOWER(order_status) <> 'canceled'
ORDER BY MONTH(order_purchase_timestamp);
```

Table 14: Displaying records 1 - 10

month	month_number	order_count
January	1	797
February	2	1763
March	3	2649
April	4	2386
May	5	3671
June	6	3229
July	7	3998
August	8	4304
September	9	4265
October	10	4605

Trend for 2018 shows growth in month-on-month sale throughout the year.

```
SELECT DISTINCT MONTHNAME(order_purchase_timestamp) as month,
       MONTH(order_purchase_timestamp) as month_number,
       COUNT(order_id) OVER (PARTITION BY MONTH(order_purchase_timestamp))
        AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2018 AND LOWER(order_status) <> 'canceled'
ORDER BY MONTH(order_purchase_timestamp);
```

Table 15: 9 records

month	month_number	order_count
January	1	7235
February	2	6655
March	3	7185
April	4	6924
May	5	6849
June	6	6149
July	7	6251
August	8	6428
September	9	1

Finding the sales per year shows a year-on-year growing trend.

```
SELECT DISTINCT YEAR(order_purchase_timestamp) AS year,
       COUNT(order_id) OVER(PARTITION BY YEAR(order_purchase_timestamp))
          AS count_of_orders
FROM orders;
```

Table 16: 3 records

year	count_of_orders
2016	329
2017	45101
2018	54011

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

Highest monthly sales in the given data is as follows, but it fails to show any seasonal trend:

```
SELECT YEAR(order_purchase_timestamp) as year,
       MONTHNAME(order_purchase_timestamp) as month,
       COUNT(*) as order_count
FROM orders
GROUP BY year, month
ORDER BY order_count DESC;
```

Table 17: Displaying records 1 - 10

year	month	order_count
2017	November	7544
2018	January	7269
2018	March	7211
2018	April	6939
2018	May	6873
2018	February	6728
2018	August	6512
2018	July	6292
2018	June	6167
2017	December	5673

While checking the year-wise monthly sales data, we do not see any monthly seasonality:

```sql
SELECT DISTINCT CONCAT(MONTHNAME(order_purchase_timestamp), " ", "2016") as month,
       MONTH(order_purchase_timestamp) as month_number,
       COUNT(order_id) OVER (PARTITION BY MONTH(order_purchase_timestamp))
        AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2016 AND LOWER(order_status) <> 'canceled'
ORDER BY order_count DESC;
```

Table 18: 3 records

month	month_number	order_count
October 2016	10	300
September 2016	9	2
December 2016	12	1

```sql
SELECT DISTINCT CONCAT(MONTHNAME(order_purchase_timestamp), " ", "2017") as month,
       MONTH(order_purchase_timestamp) as month_number,
       COUNT(order_id) OVER (PARTITION BY MONTH(order_purchase_timestamp))
        AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2017 AND LOWER(order_status) <> 'canceled'
ORDER BY order_count DESC;
```

Table 19: Displaying records 1 - 10

month	month_number	order_count
November 2017	11	7507
December 2017	12	5662
October 2017	10	4605
August 2017	8	4304
September 2017	9	4265
July 2017	7	3998
May 2017	5	3671

month	month_number	order_count
June 2017	6	3229
March 2017	3	2649
April 2017	4	2386

```sql
SELECT DISTINCT CONCAT(MONTHNAME(order_purchase_timestamp), " ", "2018") as month,
       MONTH(order_purchase_timestamp) as month_number,
       COUNT(order_id) OVER (PARTITION BY MONTH(order_purchase_timestamp))
        AS order_count
FROM orders
WHERE YEAR(order_purchase_timestamp) = 2018 AND LOWER(order_status) <> 'canceled'
ORDER BY order_count DESC;
```

Table 20: 9 records

month	month_number	order_count
January 2018	1	7235
March 2018	3	7185
April 2018	4	6924
May 2018	5	6849
February 2018	2	6655
August 2018	8	6428
July 2018	7	6251
June 2018	6	6149
September 2018	9	1