TARGET SQL

1. **In the database given, we have 6 tables. Please find exploratory analysis steps like checking the structure & characteristics of the dataset for the tables given below:**

- Customers table: There are a total of 99441 rows. There were no missing values found in any of the features.

Column	Data type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

There are 4119 distinct cities and 27 distinct states in our dataset.

- Geolocation table: There are a total of 1,000,163 rows. There were no missing values found in any of the features.

Column	Data type
geolocation_zip_code_prefix	INTEGER
geolocation_lat	FLOAT
geolocation_lng	FLOAT
geolocation_city	STRING
geolocation_state	STRING

- order_items table: There are a total of 112,650 rows. There were no missing values found in any of the features.

Column	Data type
order_id	STRING
order_item_id	INTEGER
product_id	STRING
seller_id	STRING
shipping_limit_date	TIMESTAMP
price	FLOAT
freight_value	FLOAT

- order_reviews table: There are a total of 99224 rows. I found 87675 Null values in review_comment_title feature. Reviews were created and answered from 31-12-2017 to 01-01-2018.

Column	Data type
review_id	STRING
order_id	STRING
review_score	INTEGER
review_comment_title	STRING
review_creation_date	TIMESTAMP
review_answer_timestamp	TIMESTAMP

- orders table: There are a total of 99441 rows. I found 160 Null values in order_approved_at**,** 1783 Null values in order_delivered_carrier_date and 2965 NULL values in order_delivered_customer_date columns. We have orders purchased from 4th Sep 2016 to 17th Oct 2018.

Column	Data type
order_id	STRING
customer_id	STRING
order_status	STRING
order_purchase_timestamp	TIMESTAMP
order_approved_at	TIMESTAMP
order_delivered_carrier_date	TIMESTAMP
order_delivered_customer_date	TIMESTAMP
order_estimated_delivery_date	TIMESTAMP

- Payments table: There are a total of 103886 rows. There were no missing values found in any of the features.

Column	Data type
order_id	STRING
payment_sequential	INTEGER
payment_type	STRING
payment_installments	INTEGER
payment_value	FLOAT

- products table: There are a total of 32951 rows. I found 610 Null values in product_category**,** product_name_length, product_description_length and product_photos_qty columns. There were 2 null values in the product weight and dimension columns.

Column	Data type
product_id	STRING
product_category	STRING
product_name_length	INTEGER
product_description_length	INTEGER
product_photos_qty	INTEGER
product_weight_g	INTEGER
product_length_cm	INTEGER
product_height_cm	INTEGER
product_width_cm	INTEGER

- sellers table: There are a total of 3095 rows. There were no missing values found in any of the features.

Column	Data type
seller_id	STRING
seller_zip_code_prefix	INTEGER
seller_city	STRING
seller_state	STRING

2. How many orders do we have for each order status?

Query:

```
SELECT order_status, COUNT(*) as total_count

FROM `scalerproject4.targetsql.orders`
GROUP BY order_status
```

Answer:

order_status	total_count
created	5
shipped	1107
approved	2
canceled	625
invoiced	314
delivered	96478
processing	301
unavailable	609

3. **Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?**

Query:

- ```
  SELECT years, months, COUNT(*) as month_count
  FROM
  (
  SELECT EXTRACT(MONTH FROM order_purchase_timestamp) as months, EXTRACT(YEAR FROM order_purchase_timestamp
  ) as years
  FROM `scalerproject4.targetsql.orders`
  )
  GROUP BY years, months
  ORDER BY years, months
  ```

- ```
  SELECT years, COUNT(*) as year_count
  FROM
  (
  SELECT EXTRACT(YEAR FROM order_purchase_timestamp) as years
  FROM `scalerproject4.targetsql.orders`
  )
  GROUP BY years
  ORDER BY years
  ```

Answer:

We can see that the no of orders at Target are increasing every year and every month. This clearly shows the growth in ecommerce.

years	months	month_count
2016	9	4
2016	10	324
2016	12	1
2017	1	800
2017	2	1780
2017	3	2682
2017	4	2404
2017	5	3700
2017	6	3245
2017	7	4026
2017	8	4331
2017	9	4285
2017	10	4631
2017	11	7544
2017	12	5673
2018	1	7269
2018	2	6728
2018	3	7211
2018	4	6939
2018	5	6873
2018	6	6167
2018	7	6292
2018	8	6512
2018	9	16
2018	10	4

years	year_count
2016	329
2017	45101
2018	54011

4. On what day of week brazilians customers tend to do online purchasing?

Query:

```
SELECT *,
 CASE
 WHEN day_of_week = 1
 THEN 'SUNDAY'
 WHEN day_of_week = 2
 THEN 'MONDAY'
 WHEN day_of_week = 3
 THEN 'TUESDAY'
 WHEN day_of_week = 4
 THEN 'WEDNESDAY'
 WHEN day_of_week = 5
 THEN 'THURSDAY'
 WHEN day_of_week = 6
 THEN 'FRIDAY'
 WHEN day_of_week = 7
 THEN 'SATURDAY'
 END AS day_of_week_names
FROM
(
SELECT EXTRACT(DAYOFWEEK FROM order_purchase_timestamp) as day_of_week, count(*) AS week_day_count
FROM `scalerproject4.targetsql.orders`
GROUP BY day_of_week
ORDER BY week_day_count DESC
)
```

day_of_week	week_day_count	day_of_week_names
2	16196	Monday
3	15963	Tuesday
4	15552	Wednesday
5	14761	Thursday
6	14122	Friday
1	11960	Sunday
7	10887	Saturday

Answer: We can see that most number of purchases are made on weekdays as compared to weekends with Monday topping the table.

5. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Query:

```
SELECT *, count(*) AS tot_order_count
FROM
(
SELECT
 Case
 WHEN EXTRACT(HOUR FROM order_purchase_timestamp) IN (23, 0, 1,2,3,4)
 THEN 'NIGHT : 23:00 to 04:59'
 WHEN EXTRACT(HOUR FROM order_purchase_timestamp) IN (5,6,7,8,9,10,11)
 THEN 'MORNING : 05:00 to 11:59'
 WHEN EXTRACT(HOUR FROM order_purchase_timestamp) IN (12,13,14,15,16)
 THEN 'AFTERNOON : 12:00 to 16:59'
 WHEN EXTRACT(HOUR FROM order_purchase_timestamp) IN (17,18,19,20,21,22)
 THEN 'EVENING : 17:00 to 22:59'
 END AS times_of_day
FROM `scalerproject4.targetsql.orders`
)
GROUP BY times_of_day
ORDER BY tot_order_count DESC
```

times_of_day	tot_order_count
EVENING : 17:00 to 22:59	36127
AFTERNOON : 12:00 to 16:59	32211
MORNING : 02:00 to 11:59	22428
NIGHT : 23:00 to 01:59	8675

Answer: We can see that most number of purchases are made during the evenings which is from 17:00 to 22:59.

6. Feature Extraction:

- Order_purchase_year

```
SELECT *, EXTRACT(YEAR FROM order_purchase_timestamp) as year
FROM `scalerproject4.targetsql.orders`
```

- Order_purchase_month

```
SELECT *, EXTRACT(MONTH FROM order_purchase_timestamp) as month
FROM `scalerproject4.targetsql.orders`
```

- Order_purchase_date

```
SELECT *, EXTRACT(DATE FROM order_purchase_timestamp) as date
FROM `scalerproject4.targetsql.orders`
```

- Order_purchase_day

```
SELECT *, EXTRACT(DAY FROM order_purchase_timestamp) as day
FROM `scalerproject4.targetsql.orders`
```

- Order_purchase_dayofweek

```
SELECT *, EXTRACT(DAYOFWEEK FROM order_purchase_timestamp) as day_of_week
FROM `scalerproject4.targetsql.orders`
```

- Order_purchase_dayofweek_name

```
SELECT *,
 CASE
 WHEN day_of_week = 1
 THEN 'SUNDAY'
 WHEN day_of_week = 2
 THEN 'MONDAY'
 WHEN day_of_week = 3
 THEN 'TUESDAY'
 WHEN day_of_week = 4
 THEN 'WEDNESDAY'
 WHEN day_of_week = 5
 THEN 'THURSDAY'
 WHEN day_of_week = 6
 THEN 'FRIDAY'
 WHEN day_of_week = 7
```

```
 THEN 'SATURDAY'
 END AS day_of_week_names
FROM
(
SELECT *, EXTRACT(DAYOFWEEK FROM order_purchase_timestamp) as day_of_week
FROM `scalerproject4.targetsql.orders`
)
```

- Order_purchase_hour

```
SELECT *, EXTRACT(HOUR FROM order_purchase_timestamp) as hour
FROM `scalerproject4.targetsql.orders`
```

- Order_purchase_time_day

```
SELECT *, EXTRACT(TIME FROM order_purchase_timestamp) as time
FROM `scalerproject4.targetsql.orders`
```

7. **Month on month orders by region**

Query:

```
SELECT cu.customer_state, EXTRACT(YEAR FROM order_purchase_timestamp) as year, EXTRACT(MONTH FROM order_purchase_timestamp) as month, count(*) as state_count
FROM `scalerproject4.targetsql.orders` AS ord
LEFT JOIN `scalerproject4.targetsql.Customers` as cu
 ON cu.customer_id = ord.customer_id
GROUP BY cu.customer_state, year, month
ORDER BY cu.customer_state, year, month
LIMIT 10
```

Answer:

customer_state	year	month	state_count
AC	2017	1	2
AC	2017	2	3
AC	2017	3	2
AC	2017	4	5
AC	2017	5	8
AC	2017	6	4
AC	2017	7	5
AC	2017	8	4
AC	2017	9	5
AC	2017	10	6

8. **Total of customer orders by state**

Query:

```
SELECT cu.customer_state, count(*) as state_count

FROM `scalerproject4.targetsql.orders` AS ord

LEFT JOIN `scalerproject4.targetsql.Customers` as cu

 ON cu.customer_id = ord.customer_id

GROUP BY cu.customer_state

ORDER BY state_count DESC
```

Answer:

customer_state	state_count
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033
GO	2020
PE	1652
CE	1336
PA	975
MT	907
MA	747
MS	715
PB	536
PI	495
RN	485
AL	413
SE	350
TO	280
RO	253
AM	148
AC	81
AP	68
RR	46

9. Top 10 Brazilian cities most no. of orders

Query:

```
SELECT cu.customer_city, count(*) as city_count

FROM `scalerproject4.targetsql.orders` AS ord

LEFT JOIN `scalerproject4.targetsql.Customers` as cu

 ON cu.customer_id = ord.customer_id

GROUP BY cu.customer_city

ORDER BY city_count DESC

LIMIT 10
```

Answer:

customer_city	city_count
sao paulo	15540
rio de janeiro	6882
belo horizonte	2773
brasilia	2131
curitiba	1521
campinas	1444
porto alegre	1379
salvador	1245
guarulhos	1189
sao bernardo do campo	938

10. How are customers distributed in Brazil

Query:

```
SELECT customer_state, count(*) as state_customer_count

FROM `scalerproject4.targetsql.Customers`

GROUP BY customer_state

ORDER BY state_customer_count DESC
```

Answer: Highest customer count in SP state.

customer_state	state_customer_count
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033
GO	2020
PE	1652
CE	1336
PA	975
MT	907
MA	747
MS	715
PB	536
PI	495
RN	485
AL	413
SE	350
TO	280
RO	253
AM	148
AC	81
AP	68
RR	46

11. City wise number of unique customers

Query:

```
SELECT customer_city, count(DISTINCT customer_id) as city_customer_count

FROM `scalerproject4.targetsql.Customers`

GROUP BY customer_city

ORDER BY city_customer_count DESC

LIMIT 10
```

Answer: Top count in Sao Paulo city.

customer_city	city_customer_count
sao paulo	15540
rio de janeiro	6882
belo horizonte	2773
brasilia	2131
curitiba	1521
campinas	1444
porto alegre	1379
salvador	1245
guarulhos	1189
sao bernardo do campo	938

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

Queries:

- CTE:

```
WITH orders_cte AS
(
 SELECT EXTRACT(MONTH FROM od.order_purchase_timestamp) as purchase_month,
  EXTRACT(YEAR FROM od.order_purchase_timestamp) as purchase_year,
  count(od.order_id) as order_count, sum(oi.price) as sum_price, sum(oi.freight_value) as sum_freight,
  sum(oi.price) / count(od.order_id) AS price_per_order,
  sum(oi.freight_value) / count(od.order_id) AS freight_per_order
 FROM `scalerproject4.targetsql.orders` AS od
 LEFT JOIN `scalerproject4.targetsql.order_items` as oi
  ON od.order_id = oi.order_id
 GROUP BY purchase_month, purchase_year
)
```

- Total amount sold in 2017 between Jan to August:

```
SELECT SUM(sum_price)
FROM orders_cte
WHERE purchase_year = 2017 AND purchase_month BETWEEN 1 AND 8
```

ANS: 3113000.319999788

- Total amount sold in 2018 between Jan to august:

```
SELECT SUM(sum_price)
FROM orders_cte
WHERE purchase_year = 2018 AND purchase_month BETWEEN 1 AND 8
```

ANS: 7385905.8000043072

- % increase from 2017 to 2018:

(7385905.8000043072 - 3113000.319999788)/ 3113000.319999788 * 100

ANS : 137.26 %

- NEW_CTE:

```
WITH orderswithcustomers_cte AS
(
 SELECT cu.customer_state, oi.price, oi.freight_value
 FROM `scalerproject4.targetsql.orders` AS od
 LEFT JOIN `scalerproject4.targetsql.order_items` as oi
  ON od.order_id = oi.order_id
 LEFT JOIN `scalerproject4.targetsql.Customers` as cu
  ON cu.customer_id = od.customer_id
 )
```

- Mean & Sum of price by customer state:

```
 SELECT customer_state, SUM(price) as price_sum, AVG(price) as price_avg
 FROM orderswithcustomers_cte
 GROUP BY customer_state
 ORDER BY customer_state
```

customer_state	price_sum	price_avg
AC	15982.95	173.7277174
AL	80314.81	180.8892117
AM	22356.84	135.496
AP	13474.3	164.3207317
BA	511349.99	134.6012082
CE	227254.71	153.7582612
DF	302603.94	125.7705486
ES	275037.31	121.9137012
GO	294591.95	126.2717317
MA	119648.22	145.2041505
MG	1585308.03	120.7485741
MS	116812.64	142.6283761
MT	156453.53	148.2971848
PA	178947.81	165.6924167
PB	115268.08	191.4752159
PE	262788.03	145.5083223
PI	86914.08	160.3580812
PR	683083.76	119.0041394
RJ	1824092.67	125.1178181
RN	83034.98	156.9659357
RO	46140.64	165.9735252
RR	7829.43	150.5659615
RS	750304.02	120.3374531
SC	520553.34	124.6535776
SE	58920.85	153.0411688
SP	5202955.05	109.6536292
TO	49621.74	157.5293333

- Mean & Sum of freight value by customer state:

```
SELECT customer_state, SUM(freight_value) as freight_value_sum, AVG(freight_value) as freight_value_avg

FROM orderswithcustomers_cte

GROUP BY customer_state

ORDER BY customer_state
```

customer_state	freight_value_sum	freight_value_avg
AC	3686.75	40.07336957
AL	15914.59	35.84367117
AM	5478.89	33.20539394
AP	2788.5	34.00609756
BA	100156.68	26.36395894
CE	48351.59	32.71420162
DF	50625.5	21.04135495
ES	49764.6	22.0587766
GO	53114.98	22.76681526
MA	31523.77	38.25700243
MG	270853.46	20.63016681
MS	19144.03	23.374884
MT	29715.43	28.16628436
PA	38699.3	35.83268519
PB	25719.73	42.72380399
PE	59449.66	32.91786268
PI	21218.2	39.14797048
PR	117851.68	20.53165157
RJ	305589.31	20.96092393
RN	18860.1	35.65236295
RO	11417.38	41.06971223
RR	2235.19	42.98442308
RS	135522.74	21.73580433
SC	89660.26	21.47036877
SE	14111.47	36.65316883
SP	718723.07	15.14727539
TO	11732.68	37.24660317

13. Analysis on sales, freight and delivery time

Queries:

CTE:

```
WITH CTE AS
(
SELECT customer_state, AVG(freight_value) AS avg_freight_value, AVG(time_to_delivery) AS avg_time_to_delivery, AVG(di
ff_estimated_delivery) AS avg_diff_estimated_delivery
FROM
(
SELECT cu.customer_state, oi.freight_value, DATE_DIFF(EXTRACT (DATE FROM od.order_purchase_timestamp),EXTRACT(D
ATE FROM od.order_delivered_customer_date), DAY) AS time_to_delivery,
  DATE_DIFF(EXTRACT (DATE FROM od.order_estimated_delivery_date),EXTRACT(DATE FROM od.order_delivered_custom
er_date), DAY) AS diff_estimated_delivery
FROM `scalerproject4.targetsql.orders` as od
LEFT JOIN `scalerproject4.targetsql.Customers` AS cu
  ON cu.customer_id = od.customer_id
LEFT JOIN `scalerproject4.targetsql.order_items` AS oi
  ON oi.order_id = od.order_id
)
GROUP BY customer_state
)
```

- Top 5 states with highest/lowest average freight value

  Highest:

  ```
  SELECT *
  FROM  CTE
  ORDER BY avg_freight_value DESC
  LIMIT 5
  ```

customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
RR	42.98442308	-28.17391304	18.32608696
PB	42.72380399	-20.54607509	13.03754266
RO	41.06971223	-19.65567766	20.04029304
AC	40.07336957	-20.68131868	20.97802198
PI	39.14797048	-19.31739962	11.52772467

  Lowest:

  ```
  SELECT *
  FROM CTE
  ORDER BY avg_freight_value ASC
  LIMIT 5
  ```

customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
SP	15.14727539	-8.662252654	11.20791077
PR	20.53165157	-11.89307842	13.48610374
MG	20.63016681	-11.92072463	13.34264922
RJ	20.96092393	-15.07479146	12.01477449
DF	21.04135495	-12.89384289	12.20042463

- Top 5 states with highest/lowest average time to delivery

  Highest:

```
SELECT *
FROM CTE
ORDER BY avg_time_to_delivery ASC
LIMIT 5
```

customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
AP	34.00609756	-28.22222222	18.39506173
RR	42.98442308	-28.17391304	18.32608696
AM	33.20539394	-26.33742331	19.93251534
AL	35.84367117	-24.44730679	8.735362998
PA	35.83268519	-23.70208729	14.25047438

  Lowest:

```
SELECT *
FROM CTE
ORDER BY avg_time_to_delivery DESC
LIMIT 5
```

customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
SP	15.14727539	-8.662252654	11.20791077
PR	20.53165157	-11.89307842	13.48610374
MG	20.63016681	-11.92072463	13.34264922
DF	21.04135495	-12.89384289	12.20042463
SC	21.47036877	-14.95021962	11.5727184

- Top 5 states where delivery is really fast/ not so fast compared to estimated date

  Fast_delivery:

```
SELECT *
FROM CTE
ORDER BY avg_diff_estimated_delivery ASC
LIMIT 5
```

customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
AC	40.07336957	-20.68131868	20.97802198
RO	41.06971223	-19.65567766	20.04029304
AM	33.20539394	-26.33742331	19.93251534
AP	34.00609756	-28.22222222	18.39506173
RR	42.98442308	-28.17391304	18.32608696

Not_so_fast_delivery:

```
SELECT *
FROM CTE
ORDER BY avg_diff_estimated_delivery DESC
LIMIT 5
```

customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
AL	35.84367117	-24.44730679	8.735362998
MA	38.25700243	-21.59	9.90625
SE	36.65316883	-21.41866667	10.00266667
ES	22.0587766	-15.58741573	10.64629213
BA	26.36395894	-19.19250611	10.98262286

14. Payment type analysis

- Count of orders for different payment types:

  Query:

```
SELECT pa.payment_type, COUNT(*) as order_count
FROM `scalerproject4.targetsql.payments` as pa
LEFT JOIN `scalerproject4.targetsql.orders` as od
 ON od.order_id = pa.order_id
GROUP BY pa.payment_type
```

  Answer:

payment_type	order_count
credit_card	76795
voucher	5775
not_defined	3
debit_card	1529
UPI	19784

- Distribution of payment installments and count of orders:

  Query:

```
SELECT pa.payment_installments, COUNT(*) as order_count
FROM `scalerproject4.targetsql.payments` as pa
LEFT JOIN `scalerproject4.targetsql.orders` as od
 ON od.order_id = pa.order_id
GROUP BY pa.payment_installments
```

Answer:

payment_installments	order_count
0	2
1	52546
2	12413
3	10461
4	7098
5	5239
6	3920
7	1626
8	4268
9	644
10	5328
11	23
12	133
13	16
14	15
15	74
16	5
17	8
18	27
20	17
21	3
22	1
23	1
24	18

- Count of orders for different payment types Month over Month:

  Query:

```
SELECT EXTRACT(YEAR FROM od.order_purchase_timestamp) as years, EXTRACT(MONTH FROM od.order_purchase_
timestamp) as months, pa.payment_installments, COUNT(*) as order_count

FROM `scalerproject4.targetsql.payments` as pa

LEFT JOIN `scalerproject4.targetsql.orders` as od

 ON od.order_id = pa.order_id

GROUP BY years, months, pa.payment_installments

ORDER BY years, months

LIMIT 10
```

  Answer:

years	months	payment_installments	order_count
2016	9	2	1
2016	9	1	1
2016	9	3	1
2016	10	1	144
2016	10	7	13
2016	10	2	30
2016	10	3	43
2016	10	6	18
2016	10	5	20
2016	10	4	26