# **Target SQL Business Case Study:**

# Qno.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?

Ans:

select column\_name,data\_type from target-sql-

12.target\_sql.INFORMATION\_SCHEMA.COLUMNS where table\_name =

'customers';



**Insight**: customers table have 5 Columns with 4 Columns as data\_type "String" and 1 as INT64.

.....

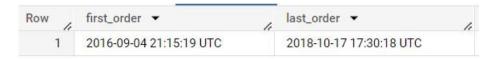
## Qno.1.2 Get the time range between which the orders were placed?

#### Ans:

Select min(order\_purchase\_timestamp) as first\_order,

max(order\_purchase\_timestamp) as last\_order from `target-sql-

12.target sql.orders`;



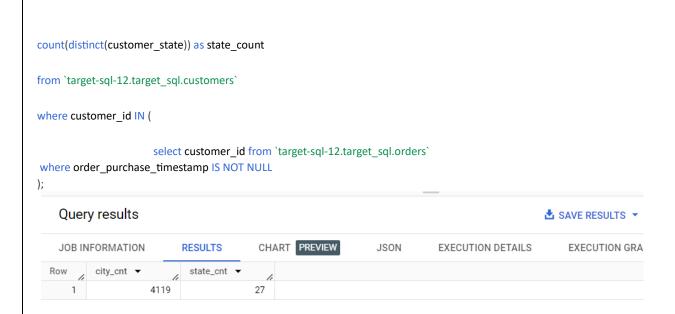
**Insight**: In orders table first orders was created in year 2016, September 4<sup>th</sup> at 21:15 UTC Standard time and latest order was created on 2018, October 17<sup>th</sup> at 17:30 UTC.

.....

## Qno.1.3 Count the Cities & States of customers who ordered during the given period? Ans:

select

count(distinct(customer\_city)) as city\_count,



**Insight**: Using subquery first we need to feed the customer\_id from orders table to customers table as it has all the customer\_id where orders are from 2016-09-04 to 2018-10-17. Then with DISTINCT function we get all city and state from customers table with count of 4119 cities and 27 states.

\_\_\_\_\_\_

# QNo. 2: In-depth Exploration:

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

```
Ans:

SELECT

EXTRACT (YEAR FROM oo.order_purchase_timestamp) AS year,

EXTRACT (MONTH FROM oo.order_purchase_timestamp) AS month,

COUNT (DISTINCT (oo.order_id)) as orders_count,

ROUND (SUM (pp.payment_value), 2) AS revenue

FROM `target-sql-12.target_sql.orders` oo

JOIN `target-sql-12.target_sql.payments` pp

ON oo.order_id = pp.order_id

GROUP BY year, month

ORDER BY year, month;
```

Row /	year ▼	month ▼	orders_count ▼	revenue ▼
1	2016	9	3	252.24
2	2016	10	324	59090.48
3	2016	12	1	19.62
4	2017	1	800	138488.04
5	2017	2	1780	291908.01
6	2017	3	2682	449863.6
7	2017	4	2404	417788.03
8	2017	5	3700	592918.82
9	2017	6	3245	511276.38
10	2017	7	4026	592382.92
11	2017	8	4331	674396.32
12	2017	9	4285	727762.45
13	2017	10	4631	779677.88

**Insight**: Based on the analysis of order count there is definite growing trend observed over the time. Also, we can see steady growth in revenue every month overs the years.

.....

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

## Ans:

```
SELECT
EXTRACT (MONTH from order_purchase_timestamp)
as month,
COUNT (DISTINCT (order_id)) as orders_count

FROM `target-sql-12.target_sql.orders`

GROUP BY month
ORDER BY month;
```

Row /	month ▼	le	orders_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

**Insight**: From above query we get that **May, July and August** has most orders count. But there is no common trend for months appears, orders are gradually increasing from March to August and suddenly getting decreased from September to December.

.....

2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

 $\circ$  0-6 hrs : Dawn  $\circ$  7-12 hrs : Mornings  $\circ$  13-18 hrs : Afternoon

o 19-23 hrs: Night

### Ans:

## 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 "Morning"
WHEN EXTRACT (hour from order\_purchase\_timestamp) BETWEEN 13 AND 18 THEN "Afternoon"

ELSE "Night"

END AS time\_duration,
COUNT(order\_id) as order\_count

FROM `target-sql-12.target\_sql.orders`

GROUP BY time\_duration
ORDER BY order\_count desc;



Insight: From above query we get that Brazilian customers mostly order during afternoon time and least in Dawn.

\_\_\_\_\_\_

# QNo.3: Evolution of E-commerce orders in the Brazil region:

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

Ans:

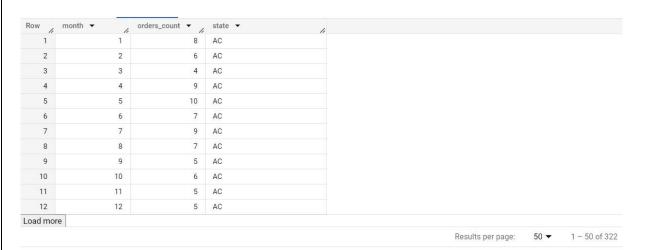
**SELECT** 

EXTRACT (MONTH FROM oo.order\_purchase\_timestamp) AS month, COUNT (DISTINCT (oo.order\_id)) as orders\_count, cc.customer\_state as state

FROM 'target-sql-12.target\_sql.orders' oo

JOIN `target-sql-12.target\_sql.customers` cc
ON oo.customer\_id = cc.customer\_id

GROUP BY state, month ORDER BY state, month;



**Insight**: From above query we got state and month wise orders count resulting 322 rows in total.

## 3.2 How are the customers distributed across all the states?

#### Ans:

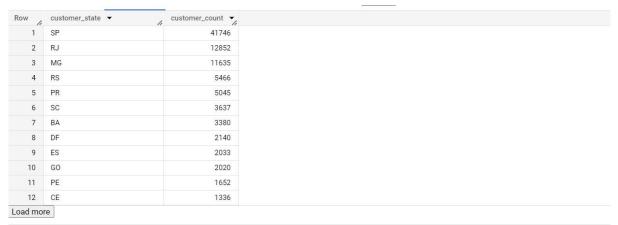
#### **SELECT**

cc.customer state,

COUNT(cc.customer\_id) AS customer\_count

FROM `target-sql-12.target\_sql.customers` cc

GROUP BY cc.customer\_state
ORDER BY customer\_count DESC;



Results per page: 50 ▼ 1 - 27 of 27

**Insight**: From above query we got state wise orders count resulting 27 rows in total. We can infer that state "SP" has most customer with count of 41746 followed by "RJ" with 12852 customers and state "RR" has least number of customers with just 46 counts.

\_\_\_\_\_\_

# Q.no.4: Impact on Economy: Analyze the money movement by ecommerce by looking at order prices, freight and others.

4.1 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.

#### Ans:

## **SELECT**

EXTRACT(MONTH FROM oo.order\_purchase\_timestamp) AS month, (

```
(
SUM(CASE WHEN EXTRACT(YEAR FROM oo.order_purchase_timestamp)=2018 AND
EXTRACT(MONTH FROM oo.order_purchase_timestamp)BETWEEN 1 AND 8 THEN pp.payment_value END)
SUM(CASE WHEN EXTRACT(YEAR FROM oo.order_purchase_timestamp)=2017 AND
EXTRACT(MONTH FROM oo.order_purchase_timestamp)BETWEEN 1 AND 8 THEN pp.payment_value END)
)/

SUM(CASE WHEN EXTRACT(YEAR FROM oo.order_purchase_timestamp)=2017 AND
EXTRACT(MONTH FROM oo.order_purchase_timestamp)BETWEEN 1 AND 8 THEN pp.payment_value END)

)*100 AS percent_increament

FROM 'target-sql-12.target_sql.orders' oo
JOIN 'target-sql-12.target_sql.payments' pp ON oo.order_id = pp.order_id

WHERE
EXTRACT(YEAR FROM oo.order_purchase_timestamp) IN (2017, 2018) AND
EXTRACT(MONTH FROM oo.order_purchase_timestamp) BETWEEN 1 AND 8
GROUP BY month
ORDER BY month;
```

Row / mon	th ▼	percent_increament
1	1	705.1266954171
2	2	239.9918145445
3	3	157.7786066709
4	4	177.8407701149
5	5	94.62734375677
6	6	100.2596912456
7	7	80.04245463390
8	8	51.60600520477



**Insight**: As per result, we can see that average increment for Jan to Aug is ~137% and we see highest increase in Jan with more than 700% increment.

.....

## 4.2 Calculate the Total & Average value of order price for each state.

## Ans:

SELECT

cc.customer\_state,

ROUND(SUM(oi.price),2) AS total\_price,

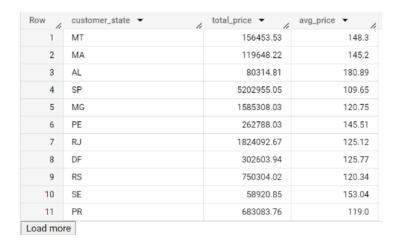
ROUND(AVG(oi.price),2) AS avg\_price

FROM `target-sql-12.target\_sql.orders` oo JOIN

`target-sql-12.target\_sql.order\_items` oi ON oo.order\_id = oi.order\_id

### JOIN

`target-sql-12.target\_sql.customers` cc ON oo.customer\_id = cc.customer\_id GROUP BY cc.customer\_state;



**Insight**: As per result, state SP has highest total price but average is just 109. State RR has least total price of 7829 with average of 150. PB has highest average price.

.....

## 4.3 Calculate the Total & Average value of order freight for each state. Ans:

## SELECT

 $cc.customer\_state,$ 

ROUND(SUM(oi.freight\_value),2) AS total\_freight\_value, ROUND(AVG(oi.freight\_value),2) AS avg\_freight\_value

FROM `target-sql-12.target\_sql.orders` oo JOIN

`target-sql-12.target\_sql.order\_items` oi ON oo.order\_id = oi.order\_id

#### JOIN

`target-sql-12.target\_sql.customers` cc ON oo.customer\_id = cc.customer\_id GROUP BY cc.customer\_state ORDER BY total\_freight\_value DESC;

Row	customer_state ▼	total_freight_value	avg_freight_value
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

**Insight**: As per result, between all 27states, state SP has highest total freight value but average is just least 15. State RR has highest average freight value of 43. PB has highest average price.

\_\_\_\_\_\_

## Q.No 5: Analysis based on sales, freight and delivery time.

5.1 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 estimated delivery date order delivered customer date

Ans: SELECT

order\_id,

DATE\_DIFF(order\_delivered\_customer\_date, order\_purchase\_timestamp, DAY) AS delivery\_time,
DATE\_DIFF(order\_estimated\_delivery\_date, order\_purchase\_timestamp, DAY) AS estimated\_delivery\_time,

DATE\_DIFF(order\_estimated\_delivery\_date, order\_delivered\_customer\_date, DAY) AS estimated\_minus\_actual\_delivery\_time

FROM `target-sql-12.target\_sql.orders`

 $WHERE\ DATE\_DIFF (order\_delivered\_customer\_date,\ order\_purchase\_timestamp,\ DAY)\ IS\ NOT\ NULL$ 

ORDER BY delivery\_time;



**Insight**: As per query result, we see delivery time taken for each order. We have many orders delivering on time but there was an order which got delivered after 200 days of expected delivery time.

.....

# 5.2 Find out the top 5 states with the highest & lowest average freight value. SELECT

cc.customer\_state,

ROUND(AVG(oi.freight\_value),2) AS avg\_freight\_value

FROM `target-sql-12.target\_sql.orders` oo

#### JOIN

`target-sql-12.target\_sql.order\_items` oi ON oo.order\_id = oi.order\_id

## JOIN

`target-sql-12.target\_sql.customers` cc ON oo.customer\_id = cc.customer\_id

GROUP BY cc.customer\_state

ORDER BY avg\_freight\_value ASC LIMIT 5;



**Insight**: As per query result, we see top 5 states with lowest average freight value, SP has lowest value with 15.15 average freight value.

Row	customer_state ▼	avg_freight_value
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

**Insight**: If we sort the query in descending order then, we see top 5 states with highest average freight value, RR has highest value with 42.98 average freight value with PB sharing similar average.

.....

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

## Ans:

#### **SELECT**

cc.customer\_state,

ROUND(AVG(DATE DIFF(order delivered customer date, order purchase timestamp, DAY)),2) AS avg delivery time,

ROUND(AVG(DATE\_DIFF(order\_estimated\_delivery\_date, order\_delivered\_customer\_date, DAY)),2) AS avg\_estimated\_delivery\_time

FROM `target-sql-12.target\_sql.orders` oo

## JOIN

`target-sql-12.target\_sql.customers` cc ON oo.customer\_id = cc.customer\_id

WHERE DATE\_DIFF(order\_purchase\_timestamp, order\_delivered\_customer\_date, DAY) IS NOT NULL AND DATE\_DIFF(order\_estimated\_delivery\_date, order\_delivered\_customer\_date, DAY) IS NOT NULL

**GROUP BY cc.customer\_state** 

ORDER BY avg\_delivery\_time DESC LIMIT 5;

Row /	customer_state ▼	avg_delivery_time ▼	avg_estimated_delivery_time
1	RR	28.98	16.41
2	AP	26.73	18.73
3	AM	25.99	18.61
4	AL	24.04	7.95
5	PA	23.32	13.19

**Insight**: Here are top 5 states with highest Average delivery time. RR has highest with almost 29 days of average delivery time.

ORDER BY avg\_delivery\_time ASC\_LIMIT 5;

Row	customer_state ▼	avg_delivery_time	avg_estimated_delive
1	SP	8.3	10.14
2	PR	11.53	12.36
3	MG	11.54	12.3
4	DF	12.51	11.12
5	SC	14.48	10.61

**Insight**: Here are 5 states with lowest Average delivery time. SP has lowest with around 8 days of average delivery time.

.....

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

#### Ans:

#### **SELECT**

cc.customer\_state,

ROUND(AVG(DATE\_DIFF(oo.order\_delivered\_customer\_date, oo.order\_purchase\_timestamp,DAY)),2) AS avg\_delivery\_time,

ROUND(AVG(DATE\_DIFF(oo.order\_estimated\_delivery\_date, oo.order\_delivered\_customer\_date,DAY)),2) AS difference\_delivery\_time,

FROM `target-sql-12.target\_sql.orders` oo

#### JOIN

`target-sql-12.target\_sql.order\_items` oi ON oo.order\_id = oi.order\_id

#### JOIN

`target-sql-12.target\_sql.customers` cc ON oo.customer\_id = cc.customer\_id

WHERE oo.order\_delivered\_customer\_date IS NOT NULL

GROUP BY cc.customer\_state

ORDER BY avg\_delivery\_time ASC LIMIT 5;

Row /	customer_state ▼	avg_delivery_time > diffe	erence_delivery_t
1	SP	8.26	10.27
2	PR	11.48	12.53
3	MG	11.52	12.4
4	DF	12.5	11.27
5	SC	14.52	10.67

**Insight**: As per query result, here are top 5 state with least average delivery time, State SP has fastest delivery average with just 8 days average delivery period. Whereas state RR has worst average delivery time with 27 days in Brazil.

\_\_\_\_\_\_

# QNo. 6: Analysis based on the payments:

6.1 Find the month-on-month no. of orders placed using different payment types. Ans:

### SELECT

EXTRACT (MONTH FROM oo.order\_purchase\_timestamp) AS month, COUNT (DISTINCT (oo.order\_id)) as orders\_count, pp.payment\_type

FROM `target-sql-12.target\_sql.orders` oo JOIN `target-sql-12.target\_sql.payments` pp ON oo.order\_id = pp.order\_id

GROUP BY month, payment\_type
ORDER BY month, payment\_type;

Row	month ▼	orders_count ▼	payment_type ▼
1	1	1715	UPI
2	1	6093	credit_card
3	1	118	debit_card
4	1	337	voucher
5	2	1723	UPI
6	2	6582	credit_card
7	2	82	debit_card
8	2	288	voucher
9	3	1942	UPI
10	3	7682	credit_card
11	3	109	debit_card
12	3	395	voucher
13	4	1783	UPI

**Insight**: As per analysis there is increased trend from Jan to Aug and then from Sept to Nov. Also, Credit card transactions are most common payment type used by Brazilian customers whereas Debit card transactions are least favoured.

.....

# 6.2 Find the no. of orders placed on the basis of the payment instalments that have been paid. Ans:

**SELECT** 

pp.payment\_installments,

COUNT (DISTINCT(oo.order\_id)) as orders\_count

FROM `target-sql-12.target\_sql.orders` oo JOIN `target-sql-12.target\_sql.payments` pp ON oo.order\_id = pp.order\_id

WHERE pp.payment\_installments >1 AND oo.order\_status != "canceled" GROUP BY payment\_installments

ORDER BY orders\_count DESC;

\_\_\_\_\_\_