TARGET SQL CASE STUDAY

1.Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:
Data type of all columns in the "customers" table.

```
SELECT COUNT(distinct(customer_id)) from TARGET.customers;
```

Total Number of Customers data we have: 99441

```
select count(distinct(customer_unique_id)) from `TARGET.customers` ;
```

```
We have 96096 number of Unique Customers ids.
```

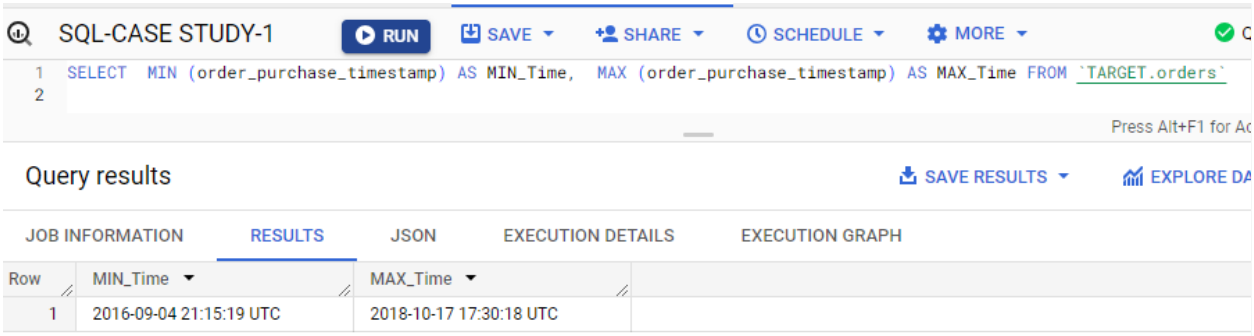
```
select count(distinct(customer_zip_code_prefix)) from `TARGET.customers` ;
```

That means Target has customers from 14994 different locations

```
select count(distinct(customer_city)) from `TARGET.customers` ;
```

Customers are from different 4119 cities and 27 states from Brazil.

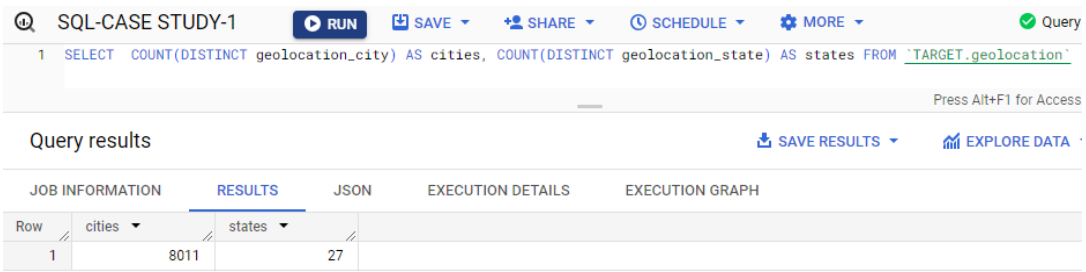
```
1.2
SELECT  MIN (order_purchase_timestamp) AS MIN_Time,  MAX
(order_purchase_timestamp) AS MAX_Time FROM `TARGET.orders`
```




```
The time range between which the orders were placed is from 2016 to 2018.
```

1.3

```
SELECT  COUNT(DISTINCT geolocation_city) AS cities, COUNT(DISTINCT
geolocation_state) AS states FROM `TARGET.geolocation`
```



Customers are from different 8011 cities and 27 states from Brazil

2.1

```
SELECT  EXTRACT(YEAR FROM order_purchase_timestamp) AS year, COUNT(order_id)
AS order_count FROM `TARGET.orders` GROUP BY 1 ORDER BY 1
```

Row	year	order_count
1	2016	329
2	2017	45101
3	2018	54011

compare to 2017 , revenue has increased in 2018 by 21%.

2.2

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month, COUNT(order_id)
AS orders_count FROM `TARGET.orders` GROUP BY 1,2 ORDER BY 1,2
```

Row	order_year	order_month	orders_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

There is a increasing trend in orders , trend sustains during 2018. There a slight fall we can observe during october 2017 following with a great hike in november month and again a fall in end of december 2017 and january 2018.

2.3

```
SELECT order_time, COUNT(order_time) count_order_time FROM (
```

```
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'
WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 19 AND 23 THEN
'Night'
END AS order_time FROM `TARGET.orders` ) a
GROUP BY 1 ORDER BY 2
```

Row	order_time	count_order_time
1	Dawn	5242
2	Morning	27733
3	Night	28331
4	Afternoon	38135

Brazilian customers mostly place their order in afternoon

3.1

```
WITH month_state_data AS (
SELECT a.order_id, EXTRACT(YEAR FROM a.order_purchase_timestamp) AS
Order_year, EXTRACT(MONTH FROM a.order_purchase_timestamp) AS
Order_month,b.customer_state
FROM `TARGET.orders` a  JOIN `TARGET.customers` b ON
a.customer_id=b.customer_id )
SELECT customer_state AS State, Order_year, Order_month, COUNT(order_id) AS
Orders_count FROM month_state_data GROUP BY 1,2,3 ORDER BY 1,2,3
```

Row	State	Order_year	Order_month	Orders_count
1	AC	2017	1	2
2	AC	2017	2	3
3	AC	2017	3	2
4	AC	2017	4	5
5	AC	2017	5	8
6	AC	2017	6	4
7	AC	2017	7	5
8	AC	2017	8	4
9	AC	2017	9	5
10	AC	2017	10	6

3.2

```
SELECT customer_state as state, COUNT(DISTINCT customer_Unique_id) as
c_count, ROUND((COUNT(DISTINCT customer_Unique_id)*100/SUM(COUNT(DISTINCT
customer_Unique_id)) OVER()),3) as c_dist FROM `TARGET.customers` GROUP BY 1
ORDER BY 2 DESC
```

Average number of order are higher during November month , september and october month average orders are comparatively low , in may and july and august have higher average orders compare to other months.

4.1

```
WITH tbl as (
SELECT EXTRACT(YEAR FROM a.order_purchase_timestamp) as year, EXTRACT(MONTH
FROM a.order_purchase_timestamp) AS Month, b.order_value
FROM `TARGET.orders` a JOIN (SELECT order_id, SUM(payment_value) AS
order_value FROM `TARGET.payments` GROUP BY 1) b ON a.order_id = b.order_id
WHERE EXTRACT(YEAR FROM a.order_purchase_timestamp) IN (2017,2018) AND
EXTRACT(MONTH FROM a.order_purchase_timestamp) BETWEEN 1 AND 8 )

SELECT Month, Year_2017,Year_2018,(Year_2018-Year_2017) AS Increase,
ROUND(((Year_2018-Year_2017)/Year_2017)*100,3) AS Increase_in_Percent
FROM (SELECT Month, SUM(CASE WHEN year=2017 THEN year END) AS Year_2017,
SUM(CASE WHEN year=2018 THEN year END) AS Year_2018 FROM tbl GROUP BY 1) c
ORDER BY 1
```

Row	Month	Year_2017	Year_2018	Increase	Increase_in_Percent
1	1	1613600	14668842	13055242	809.075
2	2	3590260	13577104	9986844	278.165
3	3	5409594	14551798	9142204	169.0
4	4	4848868	14002902	9154034	188.787
5	5	7462900	13869714	6406814	85.849
6	6	6545165	12445006	5899841	90.14
7	7	8120442	12697256	4576814	56.362
8	8	8735627	13141216	4405589	50.432

we can observe there's 815% growth increased in terms of orders and 809% growth increment in terms of revenue in January from 2017 to 2018.
growth rate for july and august in 2017 to 2018 is relatively very low!
2017-february, 2017-march,2017-november were the highest growing sale month compare to its previous month.

4.2

```
WITH tab1 AS (
SELECT a.customer_id,SUM(b.order_cost) as cost FROM `TARGET.orders` a JOIN (SELECT
order_id, SUM(payment_value) AS order_cost FROM `TARGET.payments`
GROUP BY 1) b ON a.order_id = b.order_id GROUP BY 1)

SELECT c.customer_state as State, ROUND(SUM(d.cost),3) as sum_OrderCost,
ROUND(AVG(d.cost),3) AS avg_OrderCost FROM `TARGET.customers` c JOIN tab1 d ON
c.customer_id=d.customer_id GROUP BY 1
```

Row	State	sum_OrderCost	avg_OrderCost
1	RN	102718.13	211.79
2	CE	279464.03	209.18
3	RS	890898.54	162.989
4	SC	623086.43	171.319
5	SP	5998226.96	143.687
6	MG	1872257.26	160.916
7	BA	616645.82	182.44
8	RJ	2144379.69	166.852
9	GO	350092.31	173.313
10	MA	152523.02	204.181

The state with the highest total sum of order costs is São Paulo (SP) with a sum of order costs amounting to 5,998,226.96.
The state with the highest average order cost is Paraíba (PB) with an average of order costs equal to 264.078.
On the other hand, the states of Roraima (RR), Acre (AC), and Amapá (AP) have the lowest total sum of order costs, indicating lower overall order activity in these regions.
Additionally, the states of Espírito Santo (ES), Rio Grande do Sul (RS), and Santa Catarina (SC) have the lowest average order costs, suggesting relatively lower individual order values in these states.

4.3

```
WITH tab1 AS (
SELECT a.customer_id,SUM(b.order_freight) as freight FROM `TARGET.orders` a
JOIN (
SELECT order_id, SUM(freight_value) AS order_freight FROM
`TARGET.order_items` GROUP BY 1 ) b
ON a.order_id = b.order_id GROUP BY 1 )

SELECT c.customer_state as Freight, ROUND(SUM(d.freight),3) as sum_freight,
ROUND(AVG(d.freight),3) AS avg_freight FROM `TARGET.customers` c JOIN tab1 d
ON c.customer_id=d.customer_id GROUP BY 1
```

Row	Freight	sum_freight	avg_freight
1	MT	29715.43	32.907
2	MA	31523.77	42.6
3	AL	15914.59	38.722
4	SP	718723.07	17.371
5	MG	270853.46	23.463
6	PE	59449.66	36.074
7	RJ	305589.31	23.945
8	DF	50625.5	23.824
9	RS	135522.74	24.949
10	SE	14111.47	40.903

5.1

```
SELECT order_id, DATE_DIFF(order_delivered_customer_date,
order_purchase_timestamp, DAY) AS delivery_time,
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)
AS diff_estimated_delivery FROM `TARGET.orders` WHERE
order_delivered_customer_date IS NOT NULL AND order_status = 'delivered'
ORDER BY 2 DESC
```

Row	order_id	delivery_time	diff_estimated_delive
1	ca07593549f1816d26a572e06...	209	-181
2	1b3190b2dfa9d789e1f14c05b...	208	-188
3	440d0d17af552815d15a9e41a...	195	-165
4	0f4519c5f1c541ddec9f21b3bd...	194	-161
5	285ab9426d6982034523a855f...	194	-166
6	2fb597c2f772eca01b1f5c561b...	194	-155
7	47b40429ed8cce3aee9199792...	191	-175
8	2fe324febf907e3ea3f2aa9650...	189	-167
9	2d7561026d542c8dbd8f0daea...	188	-159
10	437222e3fd1b07396f1d9ba8c...	187	-144

Average time taken for a carrier to start the delivery is 2 and a half day.
average time taken to complete delivery is 12 days. and median of delivery time is 10 days.
estimated time delivery average is 23 days.
There is a positive correlation between freight value and delivery time

5.2

```
WITH cte AS(
SELECT c.customer_state AS State, AVG(a.freight_value) AS Avg_Freight FROM
`TARGET.order_items` a
JOIN `target.orders` b ON a.order_id = b.order_id JOIN `TARGET.customers` c
ON b.customer_id = c.customer_id GROUP BY 1)
```

```
SELECT f.State as High_avg_Freight, f.Avg_Freight as High_Freight, l.State as
Low_avg_Freight, l.Avg_Freight as Low_Freight FROM (SELECT
State,Avg_Freight,ROW_NUMBER() OVER (ORDER BY Avg_Freight DESC) AS rownum
FROM cte) f JOIN (SELECT State,Avg_Freight,ROW_NUMBER() OVER (ORDER BY
Avg_Freight) AS rownum FROM cte) l
ON f.rownum = l.rownum WHERE f.rownum <= 5 ORDER BY f.rownum
```

Row	High_avg_Freight	High_Freight	Low_avg_Freight	Low_Freight
1	RR	42.98442307692...	SP	15.14727539041...
2	PB	42.72380398671...	PR	20.53165156794...
3	RO	41.06971223021...	MG	20.63016680630...
4	AC	40.07336956521...	RJ	20.96092393168...
5	PI	39.14797047970...	DF	21.04135494596...

5.3

```
WITH cte AS ( SELECT a.customer_state as state,
ROUND(AVG(DATE_DIFF(b.order_delivered_customer_date,
b.order_purchase_timestamp, DAY)),3) AS avg_delivery_time FROM
`TARGET.orders` b JOIN `TARGET.customers` a ON b.customer_id = a.customer_id
WHERE b.order_delivered_customer_date IS NOT NULL GROUP BY 1 )

SELECT h.state as states_high, h.avg_delivery_time as high_avg_delivery_time,
l.state as states_low, l.avg_delivery_time as low_avg_delivey_time
FROM (SELECT *,ROW_NUMBER() OVER(ORDER BY avg_delivery_time DESC) AS rownum
FROM cte) h JOIN (SELECT *,ROW_NUMBER() OVER(ORDER BY avg_delivery_time) AS
rownum FROM cte) l ON h.rownum = l.rownum WHERE h.rownum BETWEEN 1 AND 5
ORDER BY h.rownum
```

Row	states_high	high_avg_delivery_tir	states_low	low_avg_delivey_time
1	RR	28.976	SP	8.298
2	AP	26.731	PR	11.527
3	AM	25.986	MG	11.544
4	AL	24.04	DF	12.509
5	PA	23.316	SC	14.48

5.4

```
SELECT a.customer_state as state,
ROUND(AVG(DATE_DIFF(b.order_estimated_delivery_date,
b.order_delivered_customer_date, DAY)),3) AS avg_delivered_early_in_days FROM
`TARGET.orders` b JOIN `TARGET.customers` a ON b.customer_id = a.customer_id
WHERE b.order_delivered_customer_date IS NOT NULL AND
b.order_delivered_customer_date < b.order_estimated_delivery_date AND
b.order_status = 'delivered' GROUP BY 1 ORDER BY 2 DESC LIMIT 5
```

Row	state	avg_delivered_early_
1	RR	23.75
2	AP	21.875
3	AC	21.26
4	AM	20.281
5	RO	19.864

6.1

```
SELECT a.payment_type, EXTRACT(YEAR FROM b.order_purchase_timestamp) year,
EXTRACT(MONTH FROM b.order_purchase_timestamp) month, COUNT(DISTINCT a.order_id) as
order_count FROM `TARGET.payments` a JOIN `TAGET.orders` b ON a.order_id=b.order_id
GROUP BY 1,2,3 ORDER BY 1,2,3
```

Row	payment_type	year	month	order_count
1	UPI	2016	10	63
2	UPI	2017	1	197
3	UPI	2017	2	398
4	UPI	2017	3	590
5	UPI	2017	4	496
6	UPI	2017	5	772
7	UPI	2017	6	707
8	UPI	2017	7	845
9	UPI	2017	8	938
10	UPI	2017	9	903

The payment type "UPI" consistently has a high number of orders across months and years, with order counts ranging from 63 to 1518. This indicates that UPI is a popular payment method among customers.
"Credit card" is also a widely used payment method, with a significant number of orders in each month and year. The order counts range from 1 to 5867, and the overall order count for credit cards is higher than that of UPI.
In comparison, the payment types "debit card" and "voucher" have relatively lower order counts compared to UPI and credit cards.

6.2

```
SELECT payment_installments,COUNT(*) AS count_of_orders, FROM
`target.payments` GROUP BY 1 HAVING payment_installments >=1
```

Row	payment_installment	count_of_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239
6	6	3920
7	7	1626
8	8	4268
9	9	644
10	10	5328

The majority of payments are made in a single installment, indicating that most orders are paid in full at once.
The number of orders decreases as the number of installments increases, with the lowest number of orders being paid in 22 and 23 installments.
There is a decreasing trend in the number of orders from 2 installments to 7 installments, followed by a slight increase in orders for 8 and 10 installments.
Overall, it can be inferred that the majority of customers prefer to pay for their orders in a single installment, resulting in a decline in order count as the number of installments increases.