TARGET- CASE STUDY

Insights & Recommendations are highlighted in yellow

1) Checking the structure & characteristics of the dataset:

1.
```
select column_name,data_type from `target-scaler-427818.Target_study.INFORMATION_SCHEMA.COLUMNS` where table_name='customers';
```

JOB INFORMATION | RESULTS | CHART | JSON

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

2.
```
select min(order_purchase_timestamp) as start_date,max(order_purchase_timestamp) as end_date from `Target_study.orders`;
```

Row	start_date	end_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

3.
```
select count(distinct c.customer_city) as city_count,count(distinct c.customer_state) as state_count from `Target_study.orders` o inner join `Target_study.customers`c on o.customer_id=c.customer_id;
```

Row	city_count	state_count
1	4119	27

2) In-depth Exploration

1)
```
with cte as (select order_id,extract(year from order_purchase_timestamp) as order_year from `Target_study.orders`)

select order_year,count(order_id) as total_orders from cte group by order_year order by order_year;
```

JOB INFORMATION | RESULTS | CHAI

Row	order_year	total_orders
1	2016	329
2	2017	45101
3	2018	54011

The number of orders placed has been increasing over the past few years. Despite having data for only the last three months of 2016 and the first ten months of 2018, the trend between 2017 and 2018 clearly indicates a growing number of orders.

2)
```
with cte as (select order_id,extract(month from order_purchase_timestamp) as
order_month from `Target_study.orders`),

cte_1 as

(select order_month,count(order_id) as total_orders from cte group by order_month
order by order_month),
cte_2 as
(select *, case
when order_month in(12,1,2) then 'summer'
when order_month in(3,4,5) then 'autumn'
when order_month in(6,7,8) then 'winter'
else 'spring' end as brazil_monthly_season from cte_1)
select brazil_monthly_season, sum(cte_2.total_orders) as season_wise_orders from
cte_2 group by brazil_monthly_season order by season_wise_orders;
```

brazil_monthly_season	season_wise_orders
spring	16808
summer	22251
autumn	29809
winter	30573

As far as the seasonal trend in Brazil is concerned, the number of orders placed during spring is the lowest, while the highest number of orders is placed during winter.

3)
```
with cte as (select order_id, extract(hour from order_purchase_timestamp) as
hours from `Target_study.orders`),

cte_1 as
(select *,case
when hours between 0 and 6 then 'dawn'
when hours between 7 and 12 then 'mornings'
when hours between 13 and 18 then 'afternoon'
when hours between 19 and 23 then 'night' end as time_of_day from cte)
select time_of_day,count(order_id) total_orders from cte_1 group by time_of_day
order by total_orders;
```

time_of_day	total_orders
dawn	5242
mornings	27733
night	28331
afternoon	38135

From the above output, it is evident that the maximum number of orders were placed in the afternoon.

3) Evolution of E-commerce orders in the Brazil region:

1)
```
with cte as (select o.order_id,format_timestamp('%Y-%m',order_purchase_timestamp) as month_wise,c.customer_state from `Target_study.orders`o inner join `Target_study.customers`c on o.customer_id=c.customer_id)
select customer_state,month_wise,count(order_id) as total_orders from cte group by customer_state,month_wise order by customer_state,month_wise;
```

customer_state	month_wise	total_orders
AC	2017-01	2
AC	2017-02	3
AC	2017-03	2
AC	2017-04	5
AC	2017-05	8
AC	2017-06	4
AC	2017-07	5
AC	2017-08	4
AC	2017-09	5
AC	2017-10	6

2)
```
with cte as (select customer_state,count(distinct customer_unique_id) as total_unique_customers from `Target_study.customers` group by customer_state order by total_unique_customers desc)

select
*,round((cte.total_unique_customers/sum(cte.total_unique_customers)over())*100,2)
as percentage_distribution from cte order by cte.total_unique_customers desc;
```

customer_state	total_unique_customers	percentage_distribution
SP	40302	41.92
RJ	12384	12.88
MG	11259	11.71
RS	5277	5.49
PR	4882	5.08
SC	3534	3.68
BA	3277	3.41
DF	2075	2.16
ES	1964	2.04
GO	1952	2.03

From the above table, it is evident that most of the customers who placed orders are from the state named 'SP'.

4) Impact on Economy:

1)

```
with cte as
(select o.order_id,format_timestamp('%Y-%m',o.order_purchase_timestamp) as
Year_month_wise,extract(month from o.order_purchase_timestamp) as
month_wise,p.payment_value from `Target_study.orders`o inner join
Target_study.payments p on o.order_id=p.order_id where format_timestamp('%Y-
%m',o.order_purchase_timestamp) between '2017-01' and '2017-08'),
cte_2 as
(select cte.Year_month_wise,sum(payment_value) as total_value from cte group by
cte.Year_month_wise),
cte_3 as
(select o.order_id,format_timestamp('%Y-%m',o.order_purchase_timestamp) as
Year2_month2_wise,extract(month from o.order_purchase_timestamp) as
month2_wise,p.payment_value from `Target_study.orders`o inner join
Target_study.payments p on o.order_id=p.order_id where format_timestamp('%Y-
%m',o.order_purchase_timestamp) between '2018-01' and '2018-08'),
cte_4 as
(select cte_3.Year2_month2_wise,sum(payment_value) as total_value_2018 from cte_3
group by cte_3.Year2_month2_wise),
cte_5 as
(select * , row_number() over(order by cte_2.Year_month_wise) as refer_1 from
cte_2),
cte_6 as
(select * , row_number() over(order by cte_4.Year2_month2_wise) as refer_2 from
cte_4)
select
c5.Year_month_wise,c5.total_value,c6.Year2_month2_wise,c6.total_value_2018,round(((
c6.total_value_2018/c5.total_value)-1)*100,2) as
percentage_increase from cte_5 c5 inner join cte_6 c6 on c5.refer_1=c6.refer_2;
```

Year_month_wise	total_value	Year2_month2_wise	total_value_2018	percentage_increase
2017-01	138488.0399999...	2018-01	1115004.180000...	705.13
2017-02	291908.0099999...	2018-02	992463.3400000...	239.99
2017-03	449863.6000000...	2018-03	1159652.119999...	157.78
2017-04	417788.0300000...	2018-04	1160785.479999...	177.84
2017-05	592918.8200000...	2018-05	1153982.149999...	94.63
2017-06	511276.3800000...	2018-06	1023880.499999...	100.26
2017-07	592382.9200000...	2018-07	1066540.750000...	80.04
2017-08	674396.3200000...	2018-08	1022425.320000...	51.61

The table above shows the month-wise comparison between January to August of 2017 and 2018. For example, the value for January 2018 is 705% higher than the value for January 2017, and the value for August 2018 is 51.6% higher than the value for August 2017.

2)
```
with cte as (select
oi.order_id,oi.price,oi.freight_value,o.customer_id,c.customer_state  from
`Target_study.order_items`oi inner join `Target_study.orders` o on
oi.order_id=o.order_id inner join `Target_study.customers`c on
o.customer_id=c.customer_id)

select customer_state,sum(price) as total_sum,avg(price) overall_average from cte
group by customer_state;
```

customer_state	total_sum	overall_average
SP	5202955.050002...	109.6536291597...
RJ	1824092.669999...	125.1178180945...
PR	683083.7600000...	119.0041393728...
SC	520553.3400000...	124.6535775862...
DF	302603.9399999...	125.7705486284...
MG	1585308.029999...	120.7485741488...
PA	178947.8099999...	165.6924166666...
BA	511349.9900000...	134.6012082126...
GO	294591.9499999...	126.2717316759...
RS	750304.0200000...	120.3374530874...

3)
```
with cte as (select
oi.order_id,oi.price,oi.freight_value,o.customer_id,c.customer_state  from
`Target_study.order_items`oi inner join `Target_study.orders` o on
oi.order_id=o.order_id inner join `Target_study.customers`c on
o.customer_id=c.customer_id)

select customer_state,sum(freight_value) as total_freight,avg(freight_value)
average_freight from cte group by customer_state;
```

customer_state	total_freight	average_freight
SP	718723.0699999...	15.14727539041...
RJ	305589.3100000...	20.96092393168...
PR	117851.6800000...	20.53165156794...
SC	89660.26000000...	21.47036877394...
DF	50625.49999999...	21.04135494596...
MG	270853.4600000...	20.63016680630...
PA	38699.30000000...	35.83268518518...
BA	100156.6799999...	26.36395893656...
GO	53114.97999999...	22.76681525932...
RS	135522.7400000...	21.73580433039...

5) Analysis based on sales, freight and delivery time.

1)
```
select order_id,order_status,order_purchase_timestamp,order_delivered_customer_date,order_estimated_delivery_date, date_diff(extract(date from order_delivered_customer_date),extract(date from order_purchase_timestamp), day) as time_to_deliver,date_diff(extract(date from order_delivered_customer_date),extract(date from order_estimated_delivery_date),day) as diff_estimated_delivery from `Target_study.orders` where order_status= 'delivered' and order_delivered_customer_date is not null ;
```

order_id	order_status	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliver	diff_estimated_delivery
635c894d068ac37e6e03dc54e...	delivered	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	2017-05-18 00:00:00 UTC	31	-2
3b97562c3aee8bdedcb5c2e45...	delivered	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	2017-05-18 00:00:00 UTC	33	-1
68f47f50f04c4cb6774570cfde...	delivered	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	2017-05-18 00:00:00 UTC	30	-2
276e9ec344d3bf029ff83a161c...	delivered	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	2017-05-18 00:00:00 UTC	44	4
54e1a3c2b97fb0809da548a59...	delivered	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	2017-05-18 00:00:00 UTC	41	4
fd04fa4105ee8045f6a0139ca5...	delivered	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	2017-05-18 00:00:00 UTC	37	1
302bb8109d097a9fc6e9cefc5...	delivered	2017-04-19 22:52:59 UTC	2017-05-23 14:19:48 UTC	2017-05-18 00:00:00 UTC	34	5
66057d37308e787052a32828...	delivered	2017-04-15 19:22:06 UTC	2017-05-24 08:11:57 UTC	2017-05-18 00:00:00 UTC	39	6
19135c945c554eebfd7576c73...	delivered	2017-07-11 14:09:37 UTC	2017-08-16 20:19:32 UTC	2017-08-14 00:00:00 UTC	36	2
4493e45e7ca1084efcd38ddeb...	delivered	2017-07-11 20:56:34 UTC	2017-08-14 21:37:08 UTC	2017-08-14 00:00:00 UTC	34	0

2)
```
with cte as (select oi.order_id,oi.price,oi.freight_value,o.customer_id,c.customer_state  from `Target_study.order_items`oi inner join `Target_study.orders` o on
```

```
oi.order_id=o.order_id inner join `Target_study.customers`c on
o.customer_id=c.customer_id),cte_1 as

(select customer_state,avg(freight_value) average_freight from cte group by
customer_state),
cte_2 as
(select *,dense_rank() over(order by cte_1.average_freight desc) as highest from
cte_1 order by cte_1.average_freight desc limit 5),
cte_3 as
(select *,dense_rank() over(order by cte_1.average_freight asc) as lowest from
cte_1 order by cte_1.average_freight asc limit 5)
select c2.customer_state as higest_5_states,c2.average_freight,c3.customer_state as
lowest_5_states,c3.average_freight from cte_2 c2 inner join cte_3 c3 on
c2.highest=c3.lowest order by c3.lowest;
```

higest_5_states	average_freight	lowest_5_states	average_freight_1
RR	42.98442307692...	SP	15.14727539041...
PB	42.72380398671...	PR	20.53165156794...
RO	41.06971223021...	MG	20.63016680630...
AC	40.07336956521...	RJ	20.96092393168...
PI	39.14797047970...	DF	21.04135494596...

In the above table, 'highest' is misspelled as 'higest'

3)
```
with cte as (select order_id,customer_id,date_diff(extract(date from
order_delivered_customer_date),extract(date from order_purchase_timestamp), day) as
time_to_deliver from `Target_study.orders` where order_status= 'delivered' and
order_delivered_customer_date is not null),cte_1 as

(select c2.customer_state,avg(c1.time_to_deliver) as average_delivery_time
from cte c1 inner join `Target_study.customers` c2 on c1.customer_id=c2.customer_id
group by c2.customer_state),
cte_2 as
(select customer_state,cte_1.average_delivery_time,dense_rank() over(order by
cte_1.average_delivery_time desc) as top_5
from cte_1 order by cte_1.average_delivery_time desc limit 5),
cte_3 as
(select customer_state,cte_1.average_delivery_time,dense_rank() over(order by
cte_1.average_delivery_time asc) as bottom_5
from cte_1 order by cte_1.average_delivery_time asc limit 5)
select c2.customer_state as states_with_highest_average_delivery_time
,c2.average_delivery_time,c3.customer_state as
states_with_lowest_average_delivery_time,c3.average_delivery_time from cte_2 c2
inner join cte_3 c3 on c2.top_5=c3.bottom_5 order by c3.bottom_5;
```

states_with_highest_average_delivery_time	average_delivery_time	states_with_lowest_average_delivery_time	average_delivery_time_1
RR	29.341463414634148	SP	8.7005729243838132
AP	27.179104477611947	PR	11.938045906967316
AM	26.358620689655169	MG	11.944953320415706
AL	24.501259445843843	DF	12.899038461538467
PA	23.725158562367902	SC	14.902989283699952

From 5.2 and 5.3, we can understand that higher freight values correspond to longer delivery times, and vice versa, particularly in the states of RR, SP, PR, MG, and DF.

```
4) with cte as (select
order_id,customer_id,order_purchase_timestamp,order_delivered_customer_date,order_e
stimated_delivery_date, date_diff(extract(date from
order_delivered_customer_date),extract(date from order_purchase_timestamp), day) as
time_to_deliver,date_diff(extract(date from
order_delivered_customer_date),extract(date from
order_estimated_delivery_date),day) as
diff_estimated_delivery,date_diff(extract(date from
order_estimated_delivery_date),extract(date from order_purchase_timestamp), day) as
est_to_deliver from `Target_study.orders` where order_status= 'delivered' and
order_delivered_customer_date is not null),

cte_1 as
(select
c1.customer_state,c.time_to_deliver,c.est_to_deliver,c.diff_estimated_delivery from
cte c inner join `Target_study.customers`c1 on c.customer_id=c1.customer_id),
cte_2 as
(select customer_state, avg(time_to_deliver) as
avg_time_to_deliver,avg(est_to_deliver) as
avg_est_time,avg(diff_estimated_delivery) as sample_try
from cte_1 group by customer_state),
cte_3 as
(select customer_state,(avg_est_time-avg_time_to_deliver) as
earliest,cte_2.sample_try from cte_2 order by earliest desc)
select *,dense_rank() over(order by earliest desc) as ranking from cte_3 order by
earliest desc limit 5;
```

customer_state	earliest	sample_try	ranking
AC	20.72499999999...	-20.7249999999...	1
RO	20.10288065843...	-20.1028806584...	2
AP	19.68656716417...	-19.6865671641...	3
AM	19.56551724137...	-19.5655172413...	4
RR	17.29268292682...	-17.2926829268...	5

In the above table, sample_try refers to a different approach that yields a similar result, but with a negative value.

Recommendations:

1) ***States such as RR, RO, and AC, which are among the top 5 for the fastest order deliveries compared to the estimated delivery date, also rank among the top 5 for the highest average freight value (refer to 5.2). To address this, we could consider reducing the freight value by opting for a delivery partner that charges less, rather than choosing a high-speed delivery partner with higher charges.***

2) ***Additionally, states including AP and AM, which are among the top 5 states where orders are delivered significantly faster than the estimated delivery date, are also among the top 5 states with the highest average delivery time. We can address this discrepancy in two ways:***

***i) Reduce the Estimated Delivery Time:** This would prevent customers from being deterred by the potential delay and improve their likelihood of placing an order.*

***ii) Reduce the Freight Value:** This could help increase company profit, as outlined in the action plan mentioned in the first recommendation.*

6) Analysis based on the payments:

```
1) with cte as (select o.order_id,format_timestamp('%Y-
%m',o.order_purchase_timestamp) as month_wise,p.payment_type from
`Target_study.orders`o inner join `Target_study.payments` p on
o.order_id=p.order_id)

select payment_type,cte.month_wise, count(distinct order_id) as no_of_orders from
cte group by payment_type,cte.month_wise order by cte.month_wise,payment_type;
```

Row	payment_type	month_wise	no_of_orders
1	credit_card	2016-09	3
2	UPI	2016-10	63
3	credit_card	2016-10	253
4	debit_card	2016-10	2
5	voucher	2016-10	11
6	credit_card	2016-12	1
7	UPI	2017-01	197
8	credit_card	2017-01	582
9	debit_card	2017-01	9
10	voucher	2017-01	33

Recommendation:
From the table above, we can see that the number of credit card and UPI payments is comparatively higher. Therefore, we could partner with banks and UPI providers so that when customers earn points or rewards through the use of credit cards or UPI, they can redeem them for a 'TARGET' voucher worth XYZ amount. This strategy could help increase our visibility and potential conversion rate.

```
2) with cte as (select order_id,count(distinct payment_sequential) as
number_of_installments from `Target_study.payments` group by order_id order by
number_of_installments)

select cte.number_of_installments as installments,count(distinct order_id) as
number_of_orders from cte where cte.number_of_installments>=1 group by
cte.number_of_installments order by cte.number_of_installments;
```

Row	installments	number_of_orders
1	1	96479
2	2	2382
3	3	301
4	4	108
5	5	52
6	6	36
7	7	28
8	8	11
9	9	9
10	10	5

Greater the number of installments, lesser the number of orders.(mostly)

(Or)

```
2) select payment_installments,count(distinct order_id) as number_of_orders from
`Target_study.payments` group by payment_installments order by
payment_installments;
```

Row	payment_installment	number_of_orders
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644

I have provided 2 different solutions as you can see above