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

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

Query:-
```
select column_name, data_type from Target_SQL.INFORMATION_SCHEMA.COLUMNS

where table_name = 'customers';
```

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

Insights :- N/A

Recommendations:- N/A

1. B. Get the time range between which the orders were placed.
2.

Query:-
```
select min(order_purchase_timestamp)as first_order_date_time,

max(order_purchase_timestamp) as last_order_date_time
from Target_SQL.orders;
```

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

3. Insights:- orders were placed between 04th September 2016 to 17th October 2018.
4. Recommendations:- N/A

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

```
select count(distinct customer_city)as city_count,
count(distinct customer_state) as state_count from  Target_SQL.customers;
```

Row	city_count	state_count
1	4119	27

Insights:- Total count of distinct cities from which different customers ordered is 4119 across 27 states.
Recommendation:- N/A

2. **In-depth Exploration:**

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

1.
```
select count(order_id) as count_of_orders,
extract (month from order_purchase_timestamp) as month_wise,
extract (year from order_purchase_timestamp) as year_wise,
from Target_SQL.orders
group by month_wise, year_wise
order by year_wise, month_wise;
```

Row	count_of_orders	month_wise	year_wise
4	800	1	2017
5	1780	2	2017
6	2682	3	2017
7	2404	4	2017
8	3700	5	2017
9	3245	6	2017
10	4026	7	2017
11	4331	8	2017
12	4285	9	2017
13	4631	10	2017

Insights :- We can clearly see that in 2017 number of orders are increasing gradually but for 2018 orders are gradually decreasing also.

Recommendations:- we can give more discount coupons as sales are going down.

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

```
select count(order_id) as count_of_orders,
extract (month from order_purchase_timestamp) as month_wise,
extract (year from order_purchase_timestamp) as year_wise,
from Target_SQL.orders
group by month_wise, year_wise
order by count_of_orders desc;
```

Row	count_of_orders	month_wise	year_wise
1	7544	11	2017
2	7269	1	2018
3	7211	3	2018
4	6939	4	2018
5	6873	5	2018
6	6728	2	2018
7	6512	8	2018
8	6292	7	2018
9	6167	6	2018
10	5673	12	2017

Insights:- Orders were at peak in 11/2017, 01/2018, 03/2018 and so on.
Demand from customers is bit high during last of year and starting of year.

Recommendations:- count of orders goes down gradually as the month progresses.

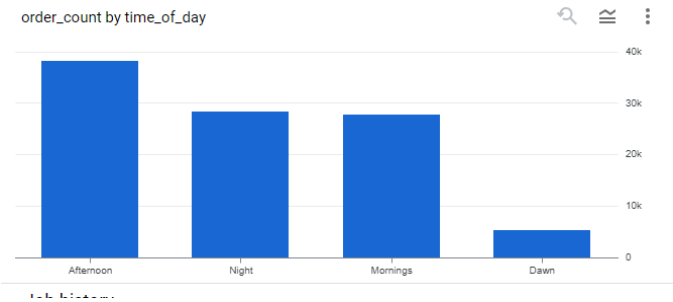
C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings

- 13-18 hrs : Afternoon
- 19-23 hrs : Night

```
select count(order_id) as order_count,
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 'Mornings'
  when extract(hour from order_purchase_timestamp) between 13 and 18 then 'Afternoon'
 else 'Night'
end as time_of_day
from Target_SQL.orders
group by time_of_day
order by order_count desc;
```

Row	order_count	time_of_day
1	38135	Afternoon
2	28331	Night
3	27733	Mornings
4	5242	Dawn



Insights:- Count of orders is highest in afternoon time and lowest in dawn time as it is the sleeping time.

Recommendations:- We can show more offers on the website during the morning, afternoon and night times.

iii. Evolution of E-commerce orders in the Brazil region:

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

```
select count(*) as month_wise_count,c.customer_state, extract(month from o.order_purchase_timestamp) as
month_wise,
extract(year from o.order_purchase_timestamp) as year_wise
from Target_SQL.customers c
left join Target_SQL.orders o
on c.customer_id=o.customer_id
group by month_wise, year_wise, c.customer_state
order by year_wise, month_wise, c.customer_state;
```

Row	month_wise_count	customer_state	month_wise	year_wise
1	1	RR	9	2016
2	1	RS	9	2016
3	2	SP	9	2016
4	2	AL	10	2016
5	4	BA	10	2016
6	8	CE	10	2016
7	6	DF	10	2016
8	4	ES	10	2016
9	9	GO	10	2016
10	4	MA	10	2016

Insights:- month wise count of orders across each state is calculated.
State with state code SP has the highest order count across months in 2018 and 2017 also.

Recommendations:- By looking at the data state wise, more focus on sales should be given on the states where count of orders is less.

B. How are the customers distributed across all the states?

```
select count(distinct customer_id) as unique_cust_count, customer_state
from Target_SQL.customers
group by customer_state
order by unique_cust_count desc;
```

Row	unique_cust_count	customer_state
1	41746	SP
2	12852	RJ
3	11635	MG
4	5466	RS
5	5045	PR
6	3637	SC
7	3380	BA
8	2140	DF
9	2033	ES
10	2020	GO

Insights:- these are the count of number of unique customers present across each state.
Count of unique customers is highest in state SP as month wise order count is also maximum for SP state as seen in previous question.

Recommendations:- We need to focus more on that states where count of customers is less.

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

A. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

```
with cte as (
select extract(month from o.order_purchase_timestamp) as month,
extract(year from o.order_purchase_timestamp) as year,
```

```
round(sum(p.payment_value),2) as order_total_per_month
from Target_SQL.payments p
join Target_SQL.orders o
using (order_id)
WHERE o.order_purchase_timestamp BETWEEN '2017-01-01 00:00:00' AND '2018-08-31 23:59:59'
group by year, month
order by year, month
)
select *,lag(order_total_per_month) over(order by year,month) as previous_month_total,
round(((order_total_per_month-lag(order_total_per_month) over(order by
year,month))/lag(order_total_per_month) over(order by year,month))*100,2) as percentage_increase
 from cte
 order by percentage_increase desc;
```

Row	month	year	order_total_per_mon	previous_month_tota	percentage_increase
1	1	2018	1115004.18	878401.48	705.13
2	2	2018	992463.34	1115004.18	239.99
3	4	2018	1160785.48	1159652.12	177.84
4	3	2018	1159652.12	992463.34	157.78
5	6	2018	1023880.5	1153982.15	100.26
6	5	2018	1153982.15	1160785.48	94.63
7	7	2018	1066540.75	1023880.5	80.04
8	11	2017	1194882.8	779677.88	53.25
9	8	2018	1022425.32	1066540.75	51.61
10	10	2017	779677.88	727762.45	7.13

Insights:- jan 2018 is the month in which percentage increase is highest.
Recommendations:- month on month cost of orders is declining so measures should be taken to improve sales.

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

```
select round(sum(payment_value),2) as total_price, round(avg(payment_value),2) as avg_price,
c.customer_state from Target_SQL.customers c
left join Target_SQL.orders o
using (customer_id)
left join Target_SQL.payments p
using (order_id)
group by c.customer_state
order by total_price desc;
```

Row	total_price	avg_price	customer_state
1	5998226.96	137.5	SP
2	2144379.69	158.53	RJ
3	1872257.26	154.71	MG
4	890898.54	157.18	RS
5	811156.38	154.15	PR
6	623086.43	165.98	SC
7	616645.82	170.82	BA
8	355141.08	161.13	DF
9	350092.31	165.76	GO
10	325967.55	154.71	ES

Insights:- Total price for orders is highest in state SP as we can see from previous questions that count of orders is also maximum for state SP.

Recommendations:- More focus should be put on states where total price for orders is low.

C. Calculate the Total & Average value of order freight for each state.

```
select c.customer_state,round(sum(oi.freight_value),2) as sum_freight, round(avg(oi.freight_value),2) as
avg_freight  from Target_SQL.customers c
left join Target_SQL.orders o
using(customer_id)
left join Target_SQL.order_items oi
using (order_id)
group by c.customer_state
order by sum_freight desc;
```

Row	customer_state	sum_freight	avg_freight
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
10	DF	50625.5	21.04

Insights:- state SP has the the highest sum of freight.

Recommendations:- N/a

V. Analysis based on sales, freight and delivery time.
A. 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.

```
select order_id, (extract(day from order_delivered_customer_date) - extract(day from
order_purchase_timestamp)) as time_to_deliver,
extract(day from order_estimated_delivery_date)- extract (day from order_delivered_customer_date) as
time_difference
from Target_SQL.orders
where order_status ='delivered';
```

Row	order_id	time_to_deliver	time_difference
1	68f47f50f04c4cb6774570cfde...	0	2
2	304e7fc7db4a67a8ab0403ce4...	-28	11
3	c930f0fb9c6fed6ef015de48ea...	-29	12
4	d0462d19e9c58af6416a06e62...	21	-12
5	8d204be4884a2307f1486df72...	-27	13
6	0d8f485ffe96c81fe3e282095e...	-29	12
7	abe6fc40cd1fe4d8d30881130...	23	-17
8	8576190c64f6d9d9ed5055185...	-27	13
9	913e9a5e8da11e9a318ab2d38...	25	-24
10	2ca33108764fd34547c251d1a...	22	-18

Insights:- N/a
Recommendations :- N/A

B. Find out the top 5 states with the highest & lowest average freight value.

```
with cte as
(
  select round(avg(oi.freight_value),2) as avg_freight_value, c.customer_state  from
Target_SQL.order_items oi
join Target_SQL.orders o
using (order_id)
join Target_SQL.customers c
using (customer_id)
group by c.customer_state
order by avg_freight_value
)
(select customer_state, cte.avg_freight_value,
dense_rank() over (order by cte.avg_freight_value asc) as rank
from cte
order by cte.avg_freight_value asc limit 5)
union all
(
  select customer_state, cte.avg_freight_value,
  dense_rank() over (order by cte.avg_freight_value desc) as rank
  from cte
  order by cte.avg_freight_value desc limit 5
);
```

Row	customer_state	avg_freight_value	rank
1	SP	15.15	1
2	PR	20.53	2
3	MG	20.63	3
4	RJ	20.96	4
5	DF	21.04	5
6	RR	42.98	1
7	PB	42.72	2
8	RO	41.07	3
9	AC	40.07	4
10	PI	39.15	5

Insights:- Here are the top 5 and bottom 5 states arranged in increasing order of avg freight value.

Recommendations:- N/A

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

```
with avg_delivery_time as
(
  select c.customer_state, round(avg(extract(day from order_delivered_customer_date) - extract(day from
order_purchase_timestamp)),2) as avg_time_to_deliver
  from Target_SQL.orders o
  join Target_SQL.customers c
  using(customer_id)
  group by c.customer_state
  order by avg_time_to_deliver
)
(select customer_state, avg_time_to_deliver,
dense_rank() over(order by avg_time_to_deliver desc) as rank
from avg_delivery_time
order by avg_time_to_deliver desc limit 5)
union all
(select customer_state, avg_time_to_deliver,
```

```
dense_rank() over(order by avg_time_to_deliver asc) as rank
from avg_delivery_time
order by avg_time_to_deliver asc limit 5);
```

Row	customer_state	avg_time_to_deliver	rank
1	AP	4.31	1
2	RR	1.9	2
3	PE	1.4	3
4	PI	1.31	4
5	CE	1.2	5
6	AM	-1.67	1
7	RO	0.07	2
8	RN	0.11	3
9	MT	0.15	4
10	AL	0.16	5

Insights:- here are the top 5 and last 5 states with avg ordered delivered time
Recommendations:- N/A

D. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
select
c.customer_state,round(avg(timestamp_diff(o.order_estimated_delivery_date,o.order_delivered_customer_date,
day)),2) as avg_delivery_days
 from Target_SQL.orders o
join Target_SQL.customers c
using(customer_id)
where o.order_status='delivered'
group by c.customer_state
order by avg_delivery_days
limit 5;
```

Row	customer_state	avg_delivery_days
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

Insights:- Top 5 states where order delivery is fast.
Recommendations:- Other states where delivery is not fast as compared to other states, should focus on delivering orders faster.

Vi. Analysis based on the payments:
Question 1) Find the month on month no. of orders placed using different payment types.

```
select  extract(year from o.order_purchase_timestamp) as year,
extract (month from o.order_purchase_timestamp) as month,
p.payment_type, count(o.order_id) as count_of_orders
from Target_SQL.orders o
join Target_SQL.payments p
on p.order_id= o.order_id
group by p.payment_type, month, year
order by year, month, payment_type;
```

Row	year	month	payment_type	count_of_orders
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	254
4	2016	10	debit_card	2
5	2016	10	voucher	23
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	583
9	2017	1	debit_card	9
10	2017	1	voucher	61

Insights:- credit card is the mode of payment which is used for purchasing orders across months and then followed by UPI and other modes of payment.
Recommendations:- credit card is popular mode of payment so more exciting offers can be offered to credit card holders.

Question 2) Find the no. of orders placed on the basis of the payment installments that have been paid.

```
select payment_installments, count(distinct order_id) as count_of_orders from
Target_SQL.payments
where payment_installments <>0
group by payment_installments;
```

Row	payment_installment	count_of_orders
1	1	49060
2	2	12389
3	3	10443
4	4	7088
5	5	5234
6	6	3916
7	7	1623
8	8	4253
9	9	644
10	10	5315

Insights:- here is the count of the number of orders placed with minimum payment installment 1 or more than 1.
Recommendations:- N/a