BUSINESS CASE

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

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

```
Select column_name, data_type
From `target.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

INSIGHT :

Customer table have different type of data_types in the table with string and integer with a primary key.

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

```
Select distinct min(order_purchase_timestamp) over() as `min`,
max(order_purchase_timestamp) over() as `max`

From `target.orders`
```

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

INSIGHT :

Using orders table extract the minimum value from order_purchase_timestamp' column for Earliest order time and the maximum value from 'order_purchase_timestamp' column for Latest order time.

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

```
Select
  count(distinct c.customer_city) as `city`, count(distinct
c.customer_state)as `state`
From
  Target.customers as c join target.orders as o using(customer_id)
Where
   o.order_purchase_timestamp between '2016-09-04 21:15:19' and '2018-10-17
17:30:18'
```

w	city	state
1	4119	27

INSIGHT :

we got Earliest order time and Latest order time from the previous table, by using that we found Count of Cities & States of customers

2. In-depth Exploration:

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

```
with `yearly_orders` as
  (
    select extract(year from order_purchase_timestamp) as `year`,
count(order_purchase_timestamp) as `no_of_orders`
    from target.orders
    group by year
    order by year asc
  )

select
  Year, no_of_orders, lag(no_of_orders) over(order by year) as
`previous_year`, round((no_of_orders - LAG(no_of_orders) OVER (ORDER BY
year)) / NULLIF(LAG(no_of_orders) OVER (ORDER BY year), 0),2) * 100 AS
`growth_rate`
from
  yearly_orders
order by
  year
```

Row	year	no_of_orders	previous_year	growth_rate
1	2016	329	*null*	*null*
2	2017	45101	329	13609.0
3	2018	54011	45101	20.0

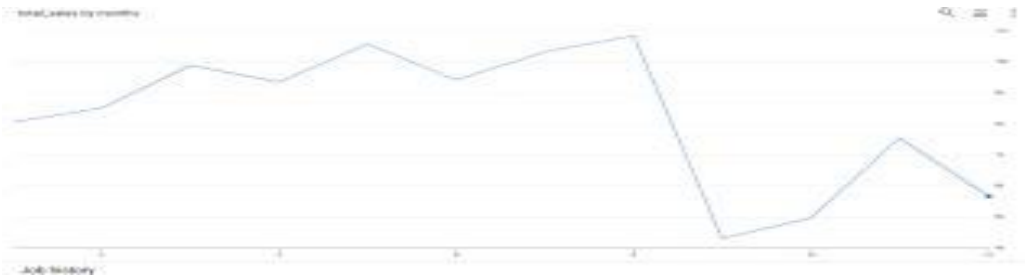
INSIGHT :

I have observed a positive growth trending, comparing with the previous year orders.

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

```
select
  extract(month from order_purchase_timestamp) as `months`, count(order_id)
as `total_sales`
from
  target.orders
group by
  extract(month from order_purchase_timestamp)
order by
  months asc
```

Row	months	total_sales
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 :

Aggregate the number of orders by month to identify seasonal trends. In this table we found a peak sales on third quarter July and august.

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

```
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 time_of_day,
    COUNT(*) AS num_orders
FROM target.orders
Group by time_of_day
Order by num_orders desc
```

Row	time_of_day	num_orders
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

INSIGHT :

The Brazilian customers mostly place their orders in afternoon time of the day and at second Brazilian mostly place order at night and very low number of order placed in dawn time

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

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

```
select
  c.customer_state,extract(month from
o.order_purchase_timestamp) as `months`,
count(o.order_purchase_timestamp) as `no_of_orders`
from
  target.customers as c join target.orders as o
using(customer_id)
group by
  months, c.customer_state
```

Row	customer_state	months	no_of_orders
1	RN	1	51
2	RN	12	30
3	RN	5	39
4	CE	2	101
5	CE	3	126
6	CE	5	136
7	CE	4	143
8	RS	3	569
9	RS	6	526
10	SC	8	365
11	SC	12	193

INSIGHT :

Extract month from “order_pruchase_timestamp” and use aggregate function count to count the orders to find the Month on month orders placed in each state.

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

```
select
  customer_state, count(*) as `no_of_customers`
from
  target.customers
group by
  customer_state
order by
  no_of_customers desc
```

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

INSIGHT :

Using aggregate function count to count the customers for each state in customer table helps to see the customers distributed across all the states

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

1. 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(year from order_purchase_timestamp) as year,
    round(sum(payment_value)) as cost_of_orders
    from `target.orders` as o
    join `target.payments` as p
    on o.order_id = p.order_id
    where extract(year from order_purchase_timestamp) = 2017 and
    extract(month from order_purchase_timestamp) between 1 and 8 or
    extract(year from order_purchase_timestamp) = 2018 and extract(month
from order_purchase_timestamp) between 1 and 8
    group by year)
select
round((sum(case when year = 2018 then cost_of_orders else 0 end) -sum(case
when year = 2017 then cost_of_orders else 0 end)) * 100.00 /
sum(case when year = 2017 then cost_of_orders else 0 end), 2) as
percent_increase
from cte
```

Row	percent_increase
1	136.98

INSIGHT :

Using cte method extract year and sum of payment value by joining the table orders and payment and filter months from 1 to 8 for both 2017 and 2018 year and using cte table using this formula percentage Increase=cost_2017cost_2018−cost_2017×100 we get the percentage_increase value.

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

```
select
  c.customer_state, round(sum(p.payment_value), 2)as `total_order_price`,
round(avg(p.payment_value), 2) as `average_order_price`
from
  target.customers as c join target.orders as o using(customer_id)join
target.payments as p using(order_id)
group by
  c.customer_state
```

Row	customer_state	total_order_price	average_order_price
1	RN	102718.13	196.78
2	CE	279464.03	199.9
3	RS	890898.54	157.18
4	SC	623086.43	165.98
5	SP	5998226.96	137.5
6	MG	1872257.26	154.71
7	BA	616645.82	170.82
8	RJ	2144379.69	158.53
9	GO	350092.31	165.76
10	MA	152523.02	198.86

INSIGHT :

Sum function used to find the total order price and avg function used to find the average of the order price by joining customers and orders and payments table and group the customer state to find the total and average of orders price for each state.

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

```
select
  c.customer_state, round(sum(i.freight_value),2) as
`total_freight`, round(avg(i.freight_value),2) as
`avg_freight`
from
  target.customers as c join target.orders as o
using(customer_id)join target.order_items as i
using(order_id)
group by
      1
```

Row	customer_state	total_freight	avg_freight
1	RN	18860.1	35.65
2	CE	48351.59	32.71
3	RS	135522.74	21.74
4	SC	89660.26	21.47
5	SP	718723.07	15.15
6	MG	270853.46	20.63
7	BA	100156.68	26.36
8	RJ	305589.31	20.96
9	GO	53114.98	22.77
10	MA	31523.77	38.26

INSIGHT :

Sum function used to find the total freight and avg function used to find the average of freight by joining customers and orders and payments table and group the customer state to find the total and average value of order freight for each state.

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

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.

```
select order_id, date_diff(order_delivered_customer_date,
order_purchase_timestamp, day)as `time_to_deliver`,
date_diff(order_delivered_customer_date,
order_estimated_delivery_date, day)as
`diff_estimated_delivery`
from `target.orders`
order by time_to_deliver desc
```

Row	order_id	time_to_deliver	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

INSIGHT :

order_delivered_customer_date - order_purchase_timestamp to find the time to deliver and order_delivered_customer_date - order_estimated_delivery_date to find difference estimated delivery by using orders table to find the days taken to deliver each order from the order's purchase date as delivery time.

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

Top 5 states Higher average freight values:

```
select c.customer_state, round(avg(i.freight_value)) as
`avg_value` from target.customers as c join target.orders as
o using(customer_id)join target.order_items as i
using(order_id) group by 1 order by avg_value desc limit 5
```

Row	customer_state	avg_value
1	PB	43.0
2	RR	43.0
3	RO	41.0
4	AC	40.0
5	PI	39.0

Top 5 states lower average freight values:

```
select c.customer_state, round(avg(i.freight_value)) as `avg_value` from
target.customers as c join target.orders as o using(customer_id)join
target.order_items as i using(order_id) group by 1 order by avg_value asc limit
5
```

Row	customer_state	avg_value
1	SP	15.0
2	PR	21.0
3	SC	21.0
4	RJ	21.0
5	MG	21.0

INSIGHT :

Average function used to find the average value by joining customer and orders and order_items table and group the customer state column and order by avg_value in descending order or ascending order to get highest and lowest average freight value and limit by 5

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

Top 5 states Higher average delivery time:

```
select c.customer_state, round(avg(DATE_DIFF(o.order_delivered_customer_date,
o.order_purchase_timestamp, DAY))) as`delivery_time` from target.customers as c
join target.orders as o using(customer_id) group by 1 order by delivery_time
desc limit 5
```

Row	customer_state	delivery_time
1	RR	29.0
2	AP	27.0
3	AM	26.0
4	AL	24.0
5	PA	23.0

Top 5 states lowest average delivery time:

```
select c.customer_state, round(avg(DATE_DIFF(o.order_delivered_customer_date,
o.order_purchase_timestamp, DAY))) as`delivery_time` from target.customers as c
join target.orders as o using(customer_id) group by 1 order by delivery_time asc
limit 5
```

Row	customer_state	delivery_time
1	SP	8.0
2	PR	12.0
3	MG	12.0
4	DF	13.0
5	SC	14.0

INSIGHT :

Average function and Date_diff used to find the average value by joining customer and orders and order_items table and group the customer state column and order by avg_value in descending order or ascending order to get highest and lowest average delivery time and limit by 5

5. 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(date_diff(o.order_estimated_delivery_date,o.order_delivered_custom
er_date, DAY))) as `fast_delivery`
from
  target.customers as c join target.orders as o using(customer_id)
where
  o.order_estimated_delivery_date is not null and
o.order_delivered_customer_date is not null
group by
  1
order by
  fast_delivery
limit
  5
```

Row	customer_state	fast_delivery
1	AL	8.0
2	MA	9.0
3	SE	9.0
4	ES	10.0
5	SP	10.0

INSIGHT :

Average function and Date_diff used to find the average value by joining customer and orders and filter to remove null values and group the customer_state to find the top 5 states where the order delivery is really fast as compared to the estimated date of delivery

6. Analysis based on the payments:

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(*) as `total
orders`
from
 target.orders as o join target.payments as p using(order_id)
group by
 year,month, p.payment_type
order by
 year,month, p.payment_type asc
```

Row	year	month	payment_type	total 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

INSIGHT :

Extract year and month and count the orders by joining orders and payments and group year, month and payment type to get the month on month no. of orders placed using different payment types

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

```
select payment_installments,count(*) as num_orders
from `target.payments`
group by payment_installments;
```

Row	payment_installment	num_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644

INSIGHT :

Count function used to count the orders from payment table and group by payment instalment to get the orders placed on the basis of the payment installments that have been paid