Target E-com

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.**
  - **Get the time range between which the orders were placed.**
  - **Count the Cities & States of customers who ordered during the given period.**

1.1

- select column_name, data_type

from `target-449309.26112611.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

1.2

- select min(order_purchase_timestamp) as starting_of_range ,

max(order_purchase_timestamp) as ending_of_range

from `target-449309.26112611.orders`

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

conclusion-Time range starts in the year 2016 and ends in 2018.

```
1.3
 select count(distinct customer_city) as no_of_cities, customer_state as no_of_states from
`target-449309.261126.customers`
group by 2
order by 1
desc
```

Row	no_of_cities	no_of_states
1	745	MG
2	629	SP
3	379	RS
4	364	PR
5	353	BA
6	240	SC
7	178	GO
8	161	CE
9	152	PE
10	149	RJ

Conclusion-The following data shows the number of cities(in each state) in descending order.

- **In-depth Exploration:**
  - **Is there a growing trend in the no. of orders placed over the past years?**
  - **Can we see some kind of monthly seasonality in terms of the no. of orders being placed?**

- **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**

```
2.1)
with yearly_order as(
   SELECT count(order_id) as number_of_orders,
extract(year from order_purchase_timestamp) as years,

FROM `target-449309.26112611.orders`
group by 2
order by 2)

select years,number_of_orders,
 lag(number_of_orders,1)over(order by years) as previous_year_orders,
 round(100*(number_of_orders - ( lag(number_of_orders,1)over(order by years) ))/(
lag(number_of_orders,1)over(order by years) ),2)        as percentage_increase

from
yearly_order
order by years
```

Row	years	number_of_orders	previous_year_orders	percentage_increase
1	2016	329	*null*	*null*
2	2017	45101	329	13608.51
3	2018	54011	45101	19.76

conclusion-"Yes, there is a clear upward trend. Notably, the growth from 2016 to 2017 was significantly higher than the growth observed from 2017 to 2018."

2.2

```
with monthly_order as(
  SELECT count(order_id) as number_of_orders,
extract(month from order_purchase_timestamp) as
months,

FROM `target-449309.26112611.orders`
group by 2
order by 2 )

select months,number_of_orders,
 lag(number_of_orders,1)over(order by months) as previous_months_orders,
 round(100*(number_of_orders - ( lag(number_of_orders,1)over(order by months) ))/(
lag(number_of_orders,1)over(order by months) ),2)       as percentage_increase

from
monthly_order
order by 1
```

Row	months	number_of_orders	previous_months_orc	percentage_increase
1	1	8069	*null*	*null*
2	2	8508	8069	5.44
3	3	9893	8508	16.28
4	4	9343	9893	-5.56
5	5	10573	9343	13.16
6	6	9412	10573	-10.98
7	7	10318	9412	9.63
8	8	10843	10318	5.09
9	9	4305	10843	-60.3
10	10	4959	4305	15.19

Conclusion-There is a mixed result when number of orders are compared with their previous orders.
Months 1-5: Steady growth, peaking at month 5.
Month 6: Sudden drop(-10.98%).
Month 7-8: Growth resumes.
Month 9: Major fall(-60.3%).
Month 10: Partial recovery(+15.19%).

-
```
with asf as(

select  extract(hour from order_purchase_timestamp) as
hours from `target-449309.26112611.orders`)

select time_of_the_day,
count(time_of_the_day) as hourss
 from(
select case when hours between 0 and 6 then 'dawn'
 when hours between 7 and 12 then 'morning'
 when  hours between 13 and 18 then 'afternoon'
when hours between 19 and 23 then 'night' end  as
time_of_the_day from asf)
group by time_of_the_day
order by 2 desc
limit 1
```

Row	time_of_the_day	hourss
1	afternoon	38135

In the afternoon there are maximum number of orders being placed.

3-

Evolution of E-commerce orders in the Brazil region:

- **Get the month on month no. of orders placed in each state.**
- **How are the customers distributed across all the states?**

```
3).1)
 select count(o.order_id) as number_of_orders, extract(month from
o.order_purchase_timestamp ) as month,
c.customer_state as state
FROM `target-449309.26112611.orders` o
join `target-449309.26112611.customers`
c on o.customer_id = c.customer_id
group by 2,3
order by 2,3
```

Row	number_of_orders	month	state
1	8	1	AC
2	39	1	AL
3	12	1	AM
4	11	1	AP
5	264	1	BA
6	99	1	CE
7	151	1	DF
8	159	1	ES
9	164	1	GO
10	66	1	MA

3.2)

```
select count(distinct customer_id) as number_of_customers, customer_state as state from
`target-449309.26112611.customers`
group by 2
order by 1
desc
```

Row	number_of_custome	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

Following data shows the number of customers in each state with highest to lowest.

4

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

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

- **Calculate the Total & Average value of order freight for each state**

4.1

```
with cte as(
  select *
  from `target-
  449309.26112611.payments` p join
  `target-449309.26112611.orders` o on
  p.order_id = o.order_id),
  seventeen as(
    select sum(case when date(order_purchase_timestamp) between '2017-01-01'
and '2017-08-31' then payment_value else null end) as sum1,


    sum(case when date(order_purchase_timestamp) between '2018-01-01' and '2018-08-
31' then
    payment_value else null end) as
    sum2 from cte
  )
  select round(100*(sum2-sum1)/sum1,2) as
  precentage_inc from seventeen
```

Row	precentage_inc
1	136.98

conclusion-"The cost of orders has more than doubled, showing an increase of approximately 137%."

4.2

- with final as

```
(select p.price, c.customer_state,o.order_id
from `target-449309.26112611.order_items`
p join `target-449309.26112611.orders` o
on p.order_id = o.order_id
join `target-449309.26112611.customers`
c on o.customer_id = c.customer_id
)

select customer_state, round(sum( price)/ count(distinct order_id),3) as average_price,
round(sum(price),3) as total_price
from final
group by 1
order by 2
desc
```

Row	customer_state	average_price	total_price
1	PB	216.669	115268.08
2	AP	198.151	13474.3
3	AC	197.32	15982.95
4	AL	195.413	80314.81
5	RO	186.804	46140.64
6	PA	184.482	178947.81
7	TO	177.856	49621.74
8	PI	176.296	86914.08
9	MT	173.26	156453.53
10	RN	172.272	83034.98

Conclusion-"The data indicates that states such as (PB, (AP), and (AC) exhibit a higher purchasing capacity for more expensive articles. This trend is also reflected in the overall increase in total spending from these states.

4.3

```
select round(sum(i.freight_value),2) total_freight, round(sum(i.freight_value)/count(distinct
o.order_id),2) as avg_freight,
 c.customer_state
from `target-449309.26112611.orders` o
join `target-449309.26112611.customers`
c on o.customer_id = c.customer_id
join `target-449309.26112611.order_items` i
on o.order_id = i.order_id
group by 3
order by 2
desc
```

Row	total_freight	avg_freight	customer_state
1	2235.19	48.59	RR
2	25719.73	48.35	PB
3	11417.38	46.22	RO
4	3686.75	45.52	AC
5	21218.2	43.04	PI
6	31523.77	42.6	MA
7	11732.68	42.05	TO
8	2788.5	41.01	AP
9	14111.47	40.9	SE
10	38699.3	39.9	PA

Conclusion-The above data gives us the information about the states who have more average freight charges.

5-

Analysis based on sales, freight and delivery time-

- **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_delivered_customer_date - order_estimated_delivery_date**
- **Find out the top 5 states with the highest & lowest average freight value.**
- **Find out the top 5 states with the highest & lowest average delivery time.**
- **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.

```
5.1)
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-449309.26112611.orders`

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_between_estimated_and_actual_delivery
from `target-449309.26112611.orders`
```

Row	order_id	time_to_deliver	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	-28
3	65d1e226dfaeb8cdc42f66542...	35	-16
4	635c894d068ac37e6e03dc54e...	30	-1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	-1
7	276e9ec344d3bf029ff83a161c...	43	4
8	54e1a3c2b97fb0809da548a59...	40	4
9	fd04fa4105ee8045f6a0139ca5...	37	1
10	302bb8109d097a9fc6e9cefc5...	33	5

conclusion-"The data helps identify which order items take longer than expected for delivery and which ones are delivered in less time."

```
5.2)
with cte as(select c.customer_state, round(sum(i.freight_value)/count(distinct o.order_id),2) as
avg_freight_cost
```

```
from `target-449309.26112611.customers`
c join `target-449309.26112611.orders` o
on c.customer_id = o.customer_id

join `target-449309.26112611.order_items` i
on o.order_id = i.order_id
group by 1
order by 2
desc
),
high as(
select customer_state,avg_freight_cost
from cte
order by avg_freight_cost desc
limit 5),

low as(
   select customer_state, avg_freight_cost
   from cte
 order by 2
   limit 5
)
select customer_state,avg_freight_cost,
'high_value' as value_type,
from high

union all

select customer_state,avg_freight_cost, 'low_value' as value_type, from
low
```

Row	customer_state	avg_freight_cost	value_type
1	RR	48.59	high_value
2	PB	48.35	high_value
3	RO	46.22	high_value
4	AC	45.52	high_value
5	PI	43.04	high_value
6	SP	17.37	low_value
7	MG	23.46	low_value
8	PR	23.58	low_value
9	DF	23.82	low_value
10	RJ	23.95	low_value

conclusion-The data provides insights into which states manage their freight charges efficiently and which do not."

accoridng to the above table, RR has highest average freight cost.

5.3) with cte as(select c.customer_state,round(sum(

date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day))/count(o.ord

er_id),2) as avg_number_of_days

from `target-449309.26112611.customers`

c join `target-449309.26112611.orders` o

on c.customer_id = o.customer_id

group by 1

),

high as(

select

customer_state,avg_number_of_days

from cte

order by 2 desc

limit 5),

```
low as(
   select customer_state,
   avg_number_of_days from cte
 order by 2
   limit 5
)
select
customer_state,avg_number_of_days,
'high_value' as value_type,
from high

union all

select customer_state,avg_number_of_days, 'low_value' as value_type,
from low
```

Row	customer_state	avg_number_of_days	value_type
1	AP	26.34	high_value
2	RR	25.83	high_value
3	AM	25.46	high_value
4	AL	23.11	high_value
5	PA	22.62	high_value
6	SP	8.05	low_value
7	PR	11.25	low_value
8	MG	11.27	low_value
9	DF	12.16	low_value
10	SC	14.12	low_value

Conclusion-The data helps us identify which states are efficient in their delivery times and which are not, enabling us to explore ways to improve logistics in underperforming regions. Notably, the state of RR records the highest average delivery time.

```
5.4) select c.customer_state,
round(sum(date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,
day))/count(distinct o.order_id),2) as avg_delivery_time_days,

round(sum(date_diff(o.order_estimated_delivery_date,o.order_purchase_timestamp,day))/c
ount(distinct o.order_id),2) as avg_estimated_time,

round( round(sum(date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,
day))/count(distinct o.order_id),2) -
round(sum(date_diff(o.order_estimated_delivery_date,o.order_purchase_timestamp,day))/c
ount(o.order_id),2),2) as diff_estimated_and_delivery

from `target-449309.26112611.customers`
c join `target-449309.26112611.orders` o

on c.customer_id = o.customer_id
group by 1
order by 4
limit 5
```

Row	customer_state	avg_delivery_time_da	avg_estimated_time	diff_estimated_and_d
1	AC	20.38	40.77	-20.39
2	RR	25.83	46.17	-20.34
3	RO	18.17	38.41	-20.24
4	AP	26.34	45.71	-19.37
5	AM	25.46	44.76	-19.3

Conclusion-States such as AC,RR,RO,AP,AM are the most efficient states with respect to delivery status.

6-
Analysis based on the payments:

- **Find the month on month no. of orders placed using different payment types.**

6-

```
select payment_type,
extract(year from order_purchase_timestamp) as yr,
extract(month from order_purchase_timestamp) as
month, count(*) as order_total
from `target-
449309.26112611.payments` p join
`target-449309.26112611.orders` o on
p.order_id = o.order_id
group by 1,2,3
order by 2,3
```

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

Conclusion- Credit Card is the most frequently used payment method with high order totals (e.g.,254 and 583).
-UPI usage is rising in 2017 compared to 2016,indicating growing customer trust and adoption.
-Voucher and Debit Card orders exist but show lower order totals and frequency.

7-

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

7.

```
select payment_installments, count(distinct o.order_id) as num_of_orders from
`target-449309.26112611.payments` p
join `target-449309.26112611.orders` o
on p.order_id = o.order_id
where payment_installments >=1 and payment_value > 0
group by 1
order by 1
```

Row	payment_installment	num_of_orders
1	1	49057
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

Overall Suggestions for Target:

- Optimize Delivery in Slow Regions: Focus on improving logistics in states like RR, where delivery times are significantly higher.
- Analyze Freight Inefficiencies: Review freight management in less cost-efficient states to reduce unnecessary shipping expenses.
- Leverage High-Spending States: States like PB, AP, and AC show strong purchasing power—consider launching premium product lines or targeted promotions there.
- Monitor High-Delay Items: Identify and address SKUs with consistently delayed delivery to enhance customer satisfaction.
- Balance Inventory Accordingly: Use insights on order value and delivery efficiency to align inventory placement and reduce transit time.
- Promote UPI and credit card paments through cashback or loyalty programs to boost revenue.

- Review voucher strategy-possibly underused or offering low value;consider bundling with offers.