

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

Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

Insights: As seen from the above data types, the customer ID is a unique identifier assigned to each customer. Both customer_city and customer_state are attributes that provide information about the location of the customer. Customer_zip_code indicates the postal code associated with the customer's address.

Recommendations: In cases where multiple systems are involved, the customer ID will be helpful to use as a common identifier to integrate and link customer data across different systems. Also if you want to get the customer order information then the customer city, customer state and customer zip code will help you to get that details.

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

```
SELECT  
Min(order_purchase_timestamp) as MinTime, Max(order_purchase_timestamp) as MaxTime  
from `Target.orders`
```

MinTime ▼	MaxTime ▼
2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights: We will get a number of customers from the state and city who placed 100,000 orders between 2016 to 2018.

Recommendation: You will see a peak in this time so be ready for your inventory. As it has been seen from the data peak in this period, we recommend to the business team enhance in business inventory. Also if you are planning to sell, it's better to keep this during this period between the festive seasons is very near.

3. Count the number of Cities and States in our dataset.

```
SELECT Distinct c.customer_state,c.customer_city
from `Target.orders` as o
join `Target.customers` as c
on o.customer_id=c.customer_id
order by 1,2
```

Row	customer_state	customer_city
2	AC	cruzeiro do sul
3	AC	epitaciolandia
4	AC	manoel urbano
5	AC	porto acre
6	AC	rio branco
7	AC	senador guiomard
8	AC	xapuri
9	AL	agua branca
10	AL	anadia
11	AL	arapiraca
12	AL	atalaia
13	AL	barra de santo antonio

Insights: We are getting cities and states of customers who ordered during this period.

Recommendation: As per the data, you are recommended to utilize the information about the cities and states of customers to create **targeted marketing** campaigns. By understanding the geographical distribution of customers, we can tailor marketing messages and promotions to specific regions. This can help the organization increase customer engagement and drive sales.

Also, identify states with a high concentration of customers and consider opening new stores in those areas. By catering to the needs and preferences of customers in these regions, we can increase market share and revenue.

2. In-depth Exploration:

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

```

SELECT *
from(
SELECT
Extract(Year from o.order_purchase_timestamp) as Year,
Extract(Month from o.order_purchase_timestamp) as Month,
count(*) as total_sales
from `Target.orders` as o
join `Target.customers` as c
on o.customer_id=c.customer_id
group by Extract(Month from o.order_purchase_timestamp),Extract(Year from o.
order_purchase_timestamp)) as X
order by X.Year,X.Month

```

Row	Year	Month	total_sales
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
11	2017	8	4331
12	2017	9	4285

Insights: If we want any specific month we can use where extract month, we can sort it by total_sales and get month and year where sales were at peak and low. There is upward and downward movement in the number of orders over the year.

Recommendation: As per the data, we can make recommendations for business decisions such as there is a growing trend in the number of orders from the month of July to Sept. It Indicates an increasing customer base or demand for products/services. In such a case, you could recommend strategies to further capitalize on the trend, such as expanding operations, optimizing inventory management, or implementing marketing initiatives to sustain and enhance customer engagement.

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

```
SELECT X.Times_of_Day,count(*) as total_sales
from
(
  SELECT case when Extract(Hour from o.order_purchase_timestamp) in (6,7,8,9,10,11) then "Morning"
  when Extract(Hour from o.order_purchase_timestamp) in (12,13,14,15,16,17) then "Afternoon"
  when Extract(Hour from o.order_purchase_timestamp) in (18,19,20,21,22,23) then "Night" else "Dawn" end as Times_of_Day
  from Target.orders as o
  join Target.customers as c
  on o.customer_id=c.customer_id
) as X
group by X.Times_of_day
order by 2 desc
```

Row	Times_of_Day	total_sales
1	Afternoon	38361
2	Night	34100
3	Morning	22240
4	Dawn	4740

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

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

```
Select * from
(
  SELECT c.customer_state,
  Extract(Month from o.order_purchase_timestamp) as Month,count(*) as total_sales
  from Target.orders as o
  join Target.customers as c
  on o.customer_id=c.customer_id
  group by
  Extract(Month from o.order_purchase_timestamp), c.customer_state
) as X
order by 2,1
```

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	customer_state ▼	Month ▼	total_sales ▼	
1	AC	1	8	
2	AL	1	39	
3	AM	1	12	
4	AP	1	11	
5	BA	1	264	
6	CE	1	99	
7	DF	1	151	
8	ES	1	159	
9	GO	1	164	
10	MA	1	66	
11	MG	1	971	

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

```
SELECT customer_state , count(*) as Total_customers_in_the_state
from `Target.customers`
group by customer_state
order by 1
```

Row	customer_state ▼	Total_customers_in_t
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033
9	GO	2020
10	MA	747
11	MG	11635
12	MS	715
13	MT	807

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 A as
(
  select EXTRACT(Month from o.order_purchase_timestamp) as Month,
  sum(p.payment_value) as sum_payment_value
  from `Target.orders` as o
  join `Target.payments` as p
  on o.order_id=p.order_id
  where EXTRACT(Month from o.order_purchase_timestamp) in (1,2,3,4,5,6,7,8) and
  EXTRACT(Year from o.order_purchase_timestamp)= 2018
  group by EXTRACT(Month from o.order_purchase_timestamp)
),B as
(
  select EXTRACT(Month from o.order_purchase_timestamp) as Month,
  sum(p.payment_value) as Sum_payment_value
  from `Target.orders` as o
  join `Target.payments` as p
  on o.order_id=p.order_id
  where EXTRACT(Month from o.order_purchase_timestamp) in (1,2,3,4,5,6,7,8) and
  EXTRACT(Year from o.order_purchase_timestamp)= 2017
  group by EXTRACT(Month from o.order_purchase_timestamp)
)
SELECT A.month,
(A.sum_payment_value-B.sum_payment_value)/B.sum_payment_value as percentage_increase
from A join B
on A.Month=B.Month
order by 1
```

Row	month	percentage_increase
1	1	7.051266954171...
2	2	2.3999181454459
3	3	1.577786066709...
4	4	1.778407701149...
5	5	0.946273437567...
6	6	1.002596912456...
7	7	0.800424546339...
8	8	0.516060052047...

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

```

with order_details as
(
    select
        c.customer_state as state,
        round(sum(i.price),2) as total_amount,
        count(distinct o.order_id) as total_orders
    from `Target.orders` as o
    inner join `Target.order_items` as i
    on o.order_id = i.order_id
    inner join `Target.customers` as c
    on o.customer_id = c.customer_id
    group by state
)

select
    state,total_amount, total_orders,
    total_amount/total_orders as avg_rise
from order_details

```

Row	state	total_amount	total_orders	avg_rise
1	MT	156453.53	903	173.2597231450...
2	MA	119648.22	740	161.6867837837...
3	AL	80314.81	411	195.4131630170...
4	SP	5202955.05	41375	125.7511794561...
5	MG	1585308.03	11544	137.3274454261...
6	PE	262788.03	1648	159.4587560679...
7	RJ	1824092.67	12762	142.9315679360...
8	DF	302603.94	2125	142.4018541176...
9	RS	750304.02	5432	138.1266605301...
10	SE	58920.85	345	170.7850724637...
11	PR	683083.76	4998	136.6714205682...
12	PA	178947.81	970	184.4822783505...

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

```

SELECT c.customer_state,
SUM(o.freight_value) as Sum_of_freight, SUM(o.price)/count(o.price) as Mean_of_Price,
from `Target.order_items` as o
join `Target.orders` as od
on o.order_id=od.order_id
join `Target.customers` as c
on od.customer_id= c.customer_id
group by c.customer_state
order by 1

```

Row	customer_state	Sum_of_freight	Mean_of_Price
1	AC	3686.750000000...	173.7277173913...
2	AL	15914.589999999...	180.8892117117...
3	AM	5478.890000000...	135.4960000000...
4	AP	2788.500000000...	164.3207317073...
5	BA	100156.6799999...	134.6012082126...
6	CE	48351.589999999...	153.7582611637...
7	DF	50625.499999999...	125.7705486284...
8	ES	49764.599999999...	121.9137012411...
9	GO	53114.979999999...	126.2717316759...
10	MA	31523.770000000...	145.2041504854...
11	MG	270853.4600000...	120.7485741488...
12	MS	19144.030000000...	142.6283760683...

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.

```

SELECT o.order_id,
DATE_DIFF(DATE(o.order_delivered_customer_date),
DATE(o.order_purchase_timestamp),DAY) as diff_purchase_delivery,
DATE_DIFF(DATE(o.order_estimated_delivery_date),
DATE(o.order_delivered_customer_date),DAY) as diff_estimated_delivery,
from `Target.orders` as o
join `Target.customers` as c
on c.customer_id=o.customer_id

```


JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION
Row	order_id		diff_purchase_delive	diff_estimated_delive	
3	65d1e226dfaeb8cdc42f66542...		36	17	
4	635c894d068ac37e6e03dc54e...		31	2	
5	3b97562c3aee8bdedcb5c2e45...		33	1	
6	68f47f50f04c4cb6774570cfde...		30	2	
7	276e9ec344d3bf029ff83a161c...		44	-4	
8	54e1a3c2b97fb0809da548a59...		41	-4	
9	fd04fa4105ee8045f6a0139ca5...		37	-1	
10	302bb8109d097a9fc6e9cefc5...		34	-5	
11	66057d37308e787052a32828...		39	-6	
12	19135c945c554eebfd7576c73...		36	-2	
13	4493e45e7ca1084efcd38ddeb...		34	0	
14	70c77e51e0f179d75a64a6141...		43	-11	

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

➔ Top 5 highest freight value:

```
SELECT c.customer_state,
AVG(oi.freight_value) as avg_of_freight
from `Target.order_items` as oi
join `Target.orders` as o
on oi.order_id=o.order_id
join `Target.customers` as c
on c.customer_id=o.customer_id
group by c.customer_state
order by 2 desc
limit 5
```

Row	customer_state	avg_of_freight
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...

➔ Top 5 lowest value :

```
SELECT c.customer_state,
AVG(oi.freight_value) as avg_of_freight
from `Target.order_items` as oi
join `Target.orders` as o
on oi.order_id=o.order_id
join `Target.customers` as c
on c.customer_id=o.customer_id
group by c.customer_state
order by 2 asc
limit 5
```

Row	customer_state	avg_of_freight
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...

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

➔ Lowest 5 :

```
SELECT c.customer_state,
AVG(DATE_DIFF(DATE(o.order_delivered_customer_date),
DATE(o.order_purchase_timestamp),DAY)) as time_of_delivery
from `Target.orders` as o
join `Target.customers` as c
on c.customer_id=o.customer_id
group by c.customer_state
order by 2
limit 5
```

Row	customer_state	time_of_delivery
1	SP	8.700530929744...
2	PR	11.93804590696...
3	MG	11.94654337296...
4	DF	12.89903846153...
5	SC	14.90752748801...

➔ Highest 5 :

```
SELECT c.customer_state,
AVG(DATE_DIFF(DATE(o.order_delivered_customer_date),
DATE(o.order_purchase_timestamp),DAY)) as time_of_delivery
from `Target.orders` as o
join `Target.customers` as c
on c.customer_id=o.customer_id
group by c.customer_state
order by 2 desc
limit 5
```

Row	customer_state	time_of_delivery
1	RR	29.34146341463...
2	AP	27.17910447761...
3	AM	26.35862068965...
4	AL	24.50125944584...
5	PA	23.72515856236...

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

➔ Slow top 5 delivery :

```
SELECT c.customer_state,
AVG(DATE_DIFF(DATE(o.order_estimated_delivery_date),
DATE(o.order_delivered_customer_date),DAY)) as diff_estimated_delivery
from `Target.orders` as o
join `Target.customers` as c
on c.customer_id=o.customer_id
group by c.customer_state
order by 2 desc
limit 5
```

Row	customer_state	diff_estimated_delivery
1	AC	20.72499999999...
2	RO	20.10288065843...
3	AP	19.68656716417...
4	AM	19.56551724137...
5	RR	17.29268292682...

➔ Fast top 5 delivery :

```
SELECT c.customer_state,  
AVG(DATE_DIFF(DATE(o.order_estimated_delivery_date),  
DATE(o.order_delivered_customer_date),DAY)) as diff_estimated_delivery  
from `Target.orders` as o  
join `Target.customers` as c  
on c.customer_id=o.customer_id  
group by c.customer_state  
order by 2  
limit 5
```

Row	customer_state	diff_estimated_delivery
1	AL	8.707808564231...
2	MA	9.571827057182...
3	SE	10.02089552238...
4	ES	10.49624060150...
5	BA	10.79453316953...

6. Analysis based on the payments:

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

```
Select EXTRACT(MONTH from o.order_purchase_timestamp), p.payment_type,  
count(o.order_id) as Count_of_orders  
from `Target.orders` as o  
join `Target.payments` as p  
on o.order_id=p.order_id  
group by Extract(Month from o.order_purchase_timestamp), p.payment_type  
order by 1
```

Row	f0_	payment_type	Count_of_orders
2	1	credit_card	6103
3	1	debit_card	118
4	1	UPI	1715
5	2	credit_card	6609
6	2	voucher	424
7	2	UPI	1723
8	2	debit_card	82
9	3	voucher	591
10	3	credit_card	7707
11	3	UPI	1942
12	3	debit_card	109
13	4	credit_card	7301

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

```

Select p.payment_installments,
count(o.order_id) as Count_of_orders
from `Target.orders` as o
join `Target.payments` as p
on o.order_id=p.order_id
group by p.payment_installments
order by 1

```

Row	payment_installment	Count_of_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
11	10	5328