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.

Answer:

```
SELECT column_name,data_type
FROM target_sql_bc.INFORMATION_SCHEMA.COLUMNS
WHERE table_name = 'customers';
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

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: the customers table has 5 columns, customer_id,customer_unique_id,customer_city,customer_state are of 'STRING' type. Customer_zip_code_prefix is int64 – integer data type.

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

Answer:

```
select min(order_purchase_timestamp) as order_start_date,
max(order_purchase_timestamp) as order_last_date
from target_sql_bc.orders
```

Output Screenshot:

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

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

Insights:

The time range of orders placed by the customers of Target is determined by writing above query for Orders table. The first order is placed on 4th sept 2016 around 9:15 pm and last order placed on 17th oct 2018 4:30 pm.

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

Answer:

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

Output Screenshot:

Query results

JOB INFORMATION | RESULTS | CHART

Row	city_count	state_count
1	4119	27

Insights:
Target customers are from 27 different states and 4119 different cities in Brazil.

2. **In-depth Exploration:**

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

Answer:

```
select extract(year from order_purchase_timestamp) as order_year,
count(order_purchase_timestamp) as no_of_orders_yearly
from target_sql_bc.orders
group by 1
order by 1
```

Output screenshot:

JOB INFORMATION | RESULTS | CHART | JSO

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

Insights:

By seeing the no of orders are increased year on year basis. In the year 2016 orders are 329 and in the year 2017 orders increased to 45101 from 329 in the year 2016. Similarly the orders in 2018 increased to 54011 from 45101 in the year 2017. By seeing the growth in number of orders on yearly basis company is expect to have continuous growth in coming years.

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

Answer:

```
select extract(year from order_purchase_timestamp) as order_year,
extract(month from order_purchase_timestamp) as orders_monthly,
count(order_purchase_timestamp) as no_of_orders_monthly
from target_sql_bc.orders
group by 1,2
order by 1,2
```

Output Screenshot:

Row	order_year	orders_monthly	no_of_orders_monthl
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

Insights:

Insights for the year 2016 reveal a significant decline in orders, particularly noteworthy in September with only 4 orders, followed by a sudden spike to 324 orders in October. Notably, there were no orders recorded for November and December, with only 1 order in each month. This poor performance likely stems from operational disruptions, potentially attributed to issues such as online website server failures or challenges within physical stores.

In contrast, the year 2017 showcased a consistent upward trend in orders, with substantial growth observed from 800 orders in January to 1780 orders in February, nearly doubling compared to the previous month. The peak in orders occurred during the last quarter of 2017, particularly evident in November, indicating a successful sales period.

However, in 2018, order trends exhibited fluctuation throughout the year, culminating in minimal orders during the last quarter. This pattern suggests a potential interruption in operations, underscoring the need for proactive measures to mitigate failures. Strategies such as enhancing online infrastructure resilience or launching targeted campaigns to stimulate sales during the Q4 blackout period are recommended to sustain business momentum.

3. 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

Answer:

```
With final as (
Select case
when order_time between '00:00:00' and '06:59:59' then 'Dawn'
when order_time between '07:00:00' and '12:59:59'then 'Morning'
when order_time between '13:00:00' and '18:59:59'then 'Afternoon'
when order_time between '19:00:00' and '23:59:59'then 'Night'  else 'na' end as
time_of_day ,

From (
select extract(time from order_purchase_timestamp) as order_time from
target_sql_bc.orders )
Order by 1)
select time_of_day,count(time_of_day) as orders_time_day
from final
group by 1
order by 1
```

Output Screenshot:

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

Insights:
The Brazilian customers mostly place their orders during afternoon followed by night and morning with almost same number of orders during morning and night. Whereas the orders placed during dawn are significantly less.
To increase the number of orders in dawn, promotions like offer in sale price,free delivery etc. can be considered

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

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

Answer:

```
SELECT
  FORMAT_date('%Y-%m',o.order_purchase_timestamp ) AS order_month,
  c.customer_state,
  COUNT(o.order_id) AS num_orders
FROM
  target_sql_bc.orders o
JOIN
  target_sql_bc.customers c ON o.customer_id = c.customer_id
GROUP BY
  1,2
ORDER BY
  1,2
```

Output Screenshot:

Row	order_month	customer_state	num_orders
1	2016-09	RR	1
2	2016-09	RS	1
3	2016-09	SP	2
4	2016-10	AL	2
5	2016-10	BA	4
6	2016-10	CE	8
7	2016-10	DF	6
8	2016-10	ES	4
9	2016-10	GO	9
10	2016-10	MA	4

Insights :

The number of orders over different months in different states wise remained almost same except exponential increase during 2017 july to 2018 august.

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

Answer:

```
select customer_state, count(customer_id) as customer_number
from `target_sql_bc.customers`
group by 1
order by 2 desc
```

Output Screenshot:

Row	customer_state	customer_number
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

Insights:

Saupaulo state(SG) have high number of customers and Goiás, estado(GO)state have least number of customers

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).

Answer:

```
with final as
(select FORMAT_date('%Y-%m',o.order_purchase_timestamp ) AS
order_year,ROUND(sum(payment_value),4) as payment_total
from target_sql_bc.orders o
join target_sql_bc.payments p
on o.order_id = p.order_id
```

```
where FORMAT_date('%Y-%m',o.order_purchase_timestamp )  between '2017-01' and
'2017-08'
or
FORMAT_date('%Y-%m',o.order_purchase_timestamp )  between '2018-01' and '2018-08'
group by 1
order by 1,2),
final2 as (
select order_year,payment_total, lag(payment_total) over(order by order_year) as p_total
from final
order by 1 )

select order_year,payment_total,ROUND((((payment_total-p_total)/p_total)*100),2) as
per_increase_of_sales
from final2
order by 1,2
```

Output screenshot:

Row	order_year	payment_total	per_increase_of_sale
1	2017-01	138488.04	*null*
2	2017-02	291908.01	110.78
3	2017-03	449863.6	54.11
4	2017-04	417788.03	-7.13
5	2017-05	592918.82	41.92
6	2017-06	511276.38	-13.77
7	2017-07	592382.92	15.86
8	2017-08	674396.32	13.84
9	2018-01	1115004.18	65.33
10	2018-02	992463.34	-10.99

Insights:
Over the months the percentage of sales decreased when compared to Feb 2019 and in april 2017,june 2017,feb 2018 months have seen drastic decline in sales.

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

Answer:

```
select customer_state, ROUND(sum(price),4) as total_order_price,
ROUND(avg(price),4) as avg_order_price
from target_sql_bc.customers c
```

```
join target_sql_bc.orders o
on  c.customer_id = o.customer_id
join target_sql_bc.order_items oi
on o.order_id = oi.order_id
group by 1
order by 2 desc
```

Output Screenshot:

Row	customer_state	total_order_price	avg_order_price
1	SP	5202955.05	109.6536
2	RJ	1824092.67	125.1178
3	MG	1585308.03	120.7486
4	RS	750304.02	120.3375
5	PR	683083.76	119.0041
6	SC	520553.34	124.6536
7	BA	511349.99	134.6012
8	DF	302603.94	125.7705
9	GO	294591.95	126.2717
10	ES	275037.31	121.9137

Insights:

The above table shows the total order price and average value of order price per state.The SP state has highest order total of 5.2 million with average order price 109.65 and RR state has lowest order total of 7829 with avg. order price of 150.56. the interesting insight is the state AL has the highest average order value of 180.88 and SP state with lowest avg.order price of 109

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

Answer:

```
select customer_state, ROUND(sum(freight_value),4) as
total_freight_value, ROUND(avg(freight_value),4) as
avg_freight_value
from target_sql_bc.customers c
join target_sql_bc.orders o
on  c.customer_id = o.customer_id
join target_sql_bc.order_items oi
on o.order_id = oi.order_id
group by 1
order by 2 desc
```

Output Screenshot:

Row	customer_state	total_freight_value	avg_freight_value
1	SP	718723.07	15.1473
2	RJ	305589.31	20.9609
3	MG	270853.46	20.6302
4	RS	135522.74	21.7358
5	PR	117851.68	20.5317
6	BA	100156.68	26.364
7	SC	89660.26	21.4704
8	PE	59449.66	32.9179
9	GO	53114.98	22.7668
10	DF	50625.5	21.0414

Insights:

The SP state has highest freight value of 0.72 million with average freight value of 15.14 local currency.RR state with lowest value of 2235.19 wit avg,freight value of 42.98 local currency.

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.

   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

   Answer:

```
select customer_id,order_id,
TIMESTAMP_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
delivery_time,
TIMESTAMP_diff(order_delivered_customer_date,order_estimated_delivery_date,day) as
diff_est_delivery
from target_sql_bc.orders
where order_delivered_customer_date is not null
order  by  1,2
```

   Output Screenshot:

Row	customer_id	order_id	delivery_time	diff_est_delivery
1	00012a2ce6f8dcda20d059ce9...	5f79b5b0931d63f1a42989eb6...	13	-5
2	000161a058600d5901f007fab...	a44895d095d7e0702b6a162fa...	9	-9
3	0001fd6190edaaf884bcaf3d49...	316a104623542e4d75189bb3...	5	-15
4	0002414f95344307404f0ace7...	5825ce2e88d5346438686b0b...	28	0
5	000379cdec625522490c315e7...	0ab7fb08086d4af9141453c91...	11	-4
6	0004164d20a9e969af783496f...	cd3558a10d854487b4f907e9b...	8	-13
7	000419c5494106c306a97b56...	07f6c3baf9ac86865b60f640c4...	45	26
8	00046a560d407e99b969756e...	8c3d752c5c02227878fae49ae...	8	-16
9	00050bf6e01e69d5c0fd612f1b...	fa906f338cee30a984d0945b3...	15	-10
10	000598caf2ef4117407665ac3...	9b961b894e797f63622137ff7e...	9	-16

Insights:

NA.

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

Answer:

```
with fvtable as (
select customer_state,  ROUND(avg(freight_value),4) as avg_freight_value,
dense_rank() over(order by ROUND(avg(freight_value),4) desc) as h_rnk,
dense_rank() over(order by ROUND(avg(freight_value),4) asc) as l_rnk,
from target_sql_bc.customers c
join target_sql_bc.orders o
on  c.customer_id = o.customer_id
join target_sql_bc.order_items oi
on o.order_id = oi.order_id
group by 1)

(Select customer_state ,'Top5 high_avg_freight_value' as state_rank
from fvtable
where h_rnk <= 5
order by h_rnk)

union all
(Select customer_state ,'Top5 low_avg_freight_value' as state_rank
from fvtable
where l_rnk <= 5
order by l_rnk)
order by state_rank
order by state_rank
```

Output Screenshot:

Row	customer_state	state_rank
1	PI	Top5 high_avg_freight_value
2	PB	Top5 high_avg_freight_value
3	RR	Top5 high_avg_freight_value
4	AC	Top5 high_avg_freight_value
5	RO	Top5 high_avg_freight_value
6	SP	Top5 low_avg_freight_value
7	PR	Top5 low_avg_freight_value
8	MG	Top5 low_avg_freight_value
9	RJ	Top5 low_avg_freight_value
10	DF	Top5 low_avg_freight_value

Insights:
Above table shows the top5 states with highest and lowest avg, freight values .

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

Answer:

```
with avgdeltm as (
select customer_state,
ROUND(avg(TIMESTAMP_diff(order_delivered_customer_date,order_purchase_timestamp,
day)),4) as avg_delivery_time,
dense_rank () over(order by
ROUND(avg(TIMESTAMP_diff(order_delivered_customer_date,order_purchase_timestamp,
day)),4) desc ) as h_rnk,
dense_rank () over(order by
ROUND(avg(TIMESTAMP_diff(order_delivered_customer_date,order_purchase_timestamp,
day)),4) asc ) as l_rnk
from target_sql_bc.orders o
join target_sql_bc.customers c
on c.customer_id = o.customer_id
where order_delivered_customer_date is not null
group by 1
)

(select customer_state,avg_delivery_time,'top5 high avg_del_time_states' as
state_category
from avgdeltm
where h_rnk <= 5
order by h_rnk)

union all
(select customer_state,avg_delivery_time,'top5 low avg_del_time_states' as state_category
```

```
from avgdeltm
where l_rnk <= 5
order by l_rnk)
order by state_category
```

Output screenshot:

Row	customer_state	avg_delivery_time	state_category
1	RR	28.9756	top5 high avg_del_time_states
2	AP	26.7313	top5 high avg_del_time_states
3	AM	25.9862	top5 high avg_del_time_states
4	AL	24.0403	top5 high avg_del_time_states
5	PA	23.3161	top5 high avg_del_time_states
6	SP	8.2981	top5 low avg_del_time_states
7	DF	12.5091	top5 low avg_del_time_states
8	PR	11.5267	top5 low avg_del_time_states
9	MG	11.5438	top5 low avg_del_time_states
10	SC	14.4796	top5 low avg_del_time_states

Insights:

NA

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

Answer:

```
with abc as (
select customer_state,
ROUND(avg(TIMESTAMP_diff(order_estimated_delivery_date,order_delivered_customer_date,day)),4) as fast_days,
dense_rank() over(order by
ROUND(avg(TIMESTAMP_diff(order_estimated_delivery_date,order_delivered_customer_date,day)),4) desc) as rnk
from target_sql_bc.orders o
join target_sql_bc.customers c
on o.customer_id = c.customer_id
where order_status = 'delivered'
group by 1
order by rnk
)
```

```
select customer_state
from abc
where rnk <= 5
```

Output screenshot:

Row	customer_state
1	AC
2	RO
3	AP
4	AM
5	RR

Insights:

Above table shows top 5 states have fastest delivery rate in brazil.

6. **Analysis based on the payments:**
1. Find the month on month no. of orders placed using different payment types.

Answer:

```
select FORMAT_date('%Y-%m',o.order_purchase_timestamp ) AS yearmonth,
payment_type,
count(o.order_id) as no_of_orders
from target_sql_bc.orders o
join target_sql_bc.payments p
on o.order_id = p.order_id
group by 1,2
order by 1,2
```

Output screenshot:

Row	yearmonth	payment_type	no_of_orders
1	2016-09	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	2017-01	UPI	197
8	2017-01	credit_card	583
9	2017-01	debit_card	9
10	2017-01	voucher	61

Insights:

The first month the orders started with credit card, followed by UPI. People also started to place orders by using store's vouchers, debit card etc. the orders placed using credit card remain high always.
Recommended to increase in n credit card service charges slightly to increase the profit margin.

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

Answer:

```
select payment_installments,count(order_id) as no_of_orders
from  `target_sql_bc.payments`
group by 1
```

Output screenshot:

Row	payment_installment	no_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

Insights: Most of the customers are placed the order by paying 1st instalment. 93 percent of number of orders cover whose at least 8 instalments paid.