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

(i) Data type of all columns in the "customers" table.

Ans.
```
select column_name, data_type
FROM target_project.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: In the above extracted 'customers' table, String data type columns are more than INT datatype. (4 strings columns, only one INT data type column)

(ii) Get the time range between which the orders were placed.

Ans.
```
select
min(order_purchase_timestamp) as initial_order,
max(order_purchase_timestamp) as last_order
from `target_project.orders`;
```

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

Insight: The time range between initial order and last order is almost two years one month.

(iii) Count the Cities & States of customers who ordered during the given period.

Ans.
```
select count(distinct customer_city) as no_of_cities,
count(distinct customer_state) as no_of_states
from `target_project.customers` c
inner join `target_project.orders` o
on c.customer_id = o.customer_id;
```

Row	no_of_cities	no_of_states
1	4119	27

Insight: The company has received customer orders from 4119 distinct cities, which are spread out among 27 states.

2. **In-depth Exploration:**

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

Ans.
```
with count_of_orders as
(select extract(year from order_purchase_timestamp) as years,
count(*) as no_of_orders
from `target_project.orders`
group by years)

select years, no_of_orders,
case when no_of_orders > lag(no_of_orders) over(order by years) then
'growing'
     when no_of_orders < lag(no_of_orders) over(order by years) then
'declining'
     else 'stagnant' end as trend
from count_of_orders
order by years;
```

Row	years	no_of_orders	trend
1	2016	329	stagnant
2	2017	45101	growing
3	2018	54011	growing

Insight: Overall the no of orders has an increasing trend when compared to its each previous year. The no of orders for 2017 was remarkably high(13,608% of 2016 orders) and 2018 orders were also increased with respect to 2017 but in a declining stage.

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

Ans.

```
with monthly_seasonality AS

(select extract(year from order_purchase_timestamp) AS years,
        extract(month from order_purchase_timestamp) AS month,
        count(*) AS no_of_orders
  from`target_project.orders`
  group by month, years)

select years, month, no_of_orders,
sum(no_of_orders) over(partition by month ORDER BY month range between
unbounded preceding and current row) AS total_monthly_orders
from monthly_seasonality
order by month, years;
```

Row	years	month	no_of_orders	total_monthly_orders
1	2017	1	800	8069
2	2018	1	7269	8069
3	2017	2	1780	8508
4	2018	2	6728	8508
5	2017	3	2682	9893
6	2018	3	7211	9893
7	2017	4	2404	9343
8	2018	4	6939	9343
9	2017	5	3700	10573
10	2018	5	6873	10573

Insight: As the data is grouped with respect to both months and years, I can observe that no of orders column(which is with respect to month of each individual year) is almost increasing in the succeeding year when compared to the preceding year except for the months of 9 and 10, and coming to total_monthly_columns, 8th month orders(10843) were highest among all of the twelve months from all the years within the time range.

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

Ans.

```
with customers_orders as

(select case when extract(hour from o.order_purchase_timestamp) >= 0 and
extract(hour from o.order_purchase_timestamp) <= 6 then 'Dawn_orders'

             when extract(hour from o.order_purchase_timestamp) >= 7 and
extract(hour from o.order_purchase_timestamp) <= 12 then 'Morning_orders'
             when extract(hour from o.order_purchase_timestamp) >= 13 and
extract(hour from o.order_purchase_timestamp) <= 18 then 'afternoon_orders'
             else 'night_orders' end as brazilian_customers_orders,
             count(*) as no_of_orders
from `target_project.orders` o
group by brazilian_customers_orders
order by no_of_orders desc)

select brazilian_customers_orders, no_of_orders,
round((no_of_orders/sum(no_of_orders) over() * 100)) as percentage_of_total_orders
from customers_orders
order by no_of_orders desc
```

Row	brazilian_customers_orders	no_of_orders	percentage_of_total
1	afternoon_orders	38135	38.0
2	night_orders	28331	28.0
3	Morning_orders	27733	28.0
4	Dawn_orders	5242	5.0

Insight: As per the result I can observe that during the whole two years period, Brazilian customers mostly placed their orders during afternoon hours, which accounted to approx. 38% of the total orders, and then stood night orders and Morning orders individually each at (28%) and with 5%, Dawn orders took the last position.

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

(i)Get the month on month no. of orders placed in each state.

Ans.
```
select c.customer_state, extract(month from
 o.order_purchase_timestamp) as months, count(*) as no_of_orders
from `target_project.customers` c
inner join `target_project.orders` o
on c.customer_id = o.customer_id
group by c.customer_state, months
order by months, customer_state;
```

Row	customer_state	months	no_of_orders
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

Insight: As I can observe from the data (Total no of orders) has been divided for each month with respect to each 27 states among the years within the time range. Almost SP state stands top at every month in the count of no of orders.

(ii) How are the customers distributed across all the states?

Ans.
```
select c.customer_state, count(distinct c.customer_id) as
no_of_customers
from `target_project.customers` c
inner join target_project.geolocation g
on c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
group by customer_state;
```

Row	customer_state	no_of_customers
1	RN	483
2	CE	1332
3	RS	5462
4	SC	3637
5	SP	41731
6	MG	11624
7	BA	3371
8	RJ	12839
9	GO	2011
10	MA	743

Insight: As I can observe from the data that the target company have its largest base of customers (41731) from the state of SP among the years 2016, 17, 18 cumulatively. And lowest base is from the state of RR with just 46 customers.

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

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

Ans.

```
with Total_cost_percentage_change as

(select extract(year from o.order_purchase_timestamp) as years,
sum(p.payment_value) as total_cost_of_orders
from `target_project.orders` o
inner join target_project.payments p
on o.order_id = p.order_id
where extract(year from o.order_purchase_timestamp) in (2017, 2018) and
extract(month from o.order_purchase_timestamp) between 1 and 8
group by years)

select years, round(total_cost_of_orders) as total_cost,
round((total_cost_of_orders- lag(total_cost_of_orders) over(order by
years))/ lag(total_cost_of_orders) over(order by years) *100) as
percentage_change_in_total_cost
from Total_cost_percentage_change
group by years, total_cost_of_orders
order by years;
```

Row	years	total_cost	percentage_change_i
1	2017	3669022.0	*null*
2	2018	8694734.0	137.0

Insight: As I can observe based on the cost analysis for the years (2017,2018) between Jan to august months, the cost in 2018 increased approx. 137% of 2017. Which is incredibly high and it also denotes that no of orders and scale of operations of the business were significantly increased.

(ii) Calculate the Total & Average value of order price for each state

Ans.

```
with Total_and_Average_value as

(select c.customer_state, sum(ot.price) as Total_order_price,
avg(ot.price) as avg_order_price
from `target_project.customers` c
inner join target_project.orders o
on c.customer_id = o.customer_id
inner join `target_project.order_items` ot
on o.order_id = ot.order_id
group by c.customer_state)

select customer_state, round(Total_order_price) as total_order_price ,
round(avg_order_price) as avg_order_price
from Total_and_Average_value
order by customer_state;
```

Row	customer_state	total_order_price	avg_order_price
1	AC	15983.0	174.0
2	AL	80315.0	181.0
3	AM	22357.0	135.0
4	AP	13474.0	164.0
5	BA	511350.0	135.0
6	CE	227255.0	154.0
7	DF	302604.0	126.0
8	ES	275037.0	122.0
9	GO	294592.0	126.0
10	MA	119648.0	145.0

Insight: Based on the results, I can inference that total order price was highest for SP state (5202955.0) because it has greater no of orders comparative to other states. In contrast, PB state (191) was highest in terms of average order price.

(iii)Calculate the Total & Average value of order freight for each state

Ans.

```
with Total_and_Average_value as

(select c.customer_state, sum(ot.freight_value) as
Total_freight_price,
avg(ot.freight_value) as avg_freight_price
from `target_project.customers` c
inner join target_project.orders o
on c.customer_id = o.customer_id
inner join `target_project.order_items` ot
on o.order_id = ot.order_id
group by c.customer_state)

select customer_state, round(Total_freight_price) as
total_freight_price ,
round(avg_freight_price) as avg_freight_price
from Total_and_Average_value
order by customer_state;
```

Row	customer_state	total_freight_price	avg_freight_price
1	AC	3687.0	40.0
2	AL	15915.0	36.0
3	AM	5479.0	33.0
4	AP	2789.0	34.0
5	BA	100157.0	26.0
6	CE	48352.0	33.0
7	DF	50625.0	21.0
8	ES	49765.0	22.0
9	GO	53115.0	23.0
10	MA	31524.0	38.0

Insight: Based on the result, I can inference that the total freight price was highest for SP state (718723.0.), as it has more no of orders, it will have more charges cost. In contrast, PB and RR states (43) were highest in terms of average freight price, because they have less number of orders.

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

(i) 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.

Ans.
```
select order_id,
 max(date_diff(order_delivered_customer_date,
order_purchase_timestamp, DAY)) as time_to_deliver,
 max(date_diff(order_estimated_delivery_date,
order_delivered_customer_date, DAY)) as diff_estimated_diff
 from `target_project.orders`
 group by order_id
 order by order_id
```

Row	order_id	time_to_deliver	diff_estimated_diff
1	00010242fe8c5a6d1ba2dd792...	7	8
2	00018f77f2f0320c557190d7a1...	16	2
3	000229ec398224ef6ca0657da...	7	13
4	00024acbcdf0a6daa1e931b03...	6	5
5	00042b26cf59d7ce69dfabb4e...	25	15
6	00048cc3ae777c65dbb7d2a06...	6	14
7	00054e8431b9d7675808bcb8...	8	16
8	000576fe39319847cbb9d288c...	5	15
9	0005a1a1728c9d785b8e2b08b...	9	0
10	0005f50442cb953dcd1d21e1f...	2	18

Insight: As I can observe that the output is so huge, it had resulted into 99441 rows. The given output was this huge because we tried to find time took for delivery and difference of delivery from estimated time to actual date of delivery based on each order-id, (we grouped it under individual order_id) and hence the result is huge.

(ii)Find out the top 5 states with the highest & lowest average freight value.

Ans.

```
with Average_value as

(select c.customer_state, avg(ot.freight_value) as avg_freight_value,
dense_rank() over (order by avg(ot.freight_value)desc) as
rank_of_top_freight_states,
dense_rank() over (order by avg(ot.freight_value)) as
rank_of_bottom_freight_states,
case when dense_rank() over(order by avg(ot.freight_value)desc) <= 5 then
'top_class' else 'bottom_class' end as category
from `target_project.customers` c
inner join target_project.orders o
on c.customer_id = o.customer_id
inner join `target_project.order_items` ot
on o.order_id = ot.order_id
group by c.customer_state)

(select customer_state, avg_freight_value, rank_of_top_freight_states as
rank, category
from Average_value
order by  avg_freight_value desc
limit 5)
union all
(select customer_state, avg_freight_value, rank_of_bottom_freight_states as
rank, category
from Average_value
order by avg_freight_value asc
limit 5)
```

Row	customer_state	avg_freight_value	rank	category
1	RR	42.98442307692...	1	top_class
2	PB	42.72380398671...	2	top_class
3	RO	41.06971223021...	3	top_class
4	AC	40.07336956521...	4	top_class
5	PI	39.14797047970...	5	top_class
6	SP	15.14727539041...	1	bottom_class
7	PR	20.53165156794...	2	bottom_class
8	MG	20.63016680630...	3	bottom_class
9	RJ	20.96092393168...	4	bottom_class
10	DF	21.04135494596...	5	bottom_class

Insight: As I can observe based on the analysis, states of (RR, PB, RO, AC, PI) stood as top 5 states with highest avg_freight_values, where as in contrast states of (SP, PR, MG, RJ, DF) stood as bottom states with lowest avg_freight_values. Here the reason of why SP state has low average because of its huge orders and due to the greater number of orders the average is going to be affected significantly. In fact, states with more no of orders, the average of freight cost will be comparatively lower than other states whose orders are very less.

(iii)Find out the top 5 states with the highest & lowest average delivery time.

Ans.

```
with Average_value as
(select c.customer_state,
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY)) as
Avg_time_to_deliver,
dense_rank() over(order by
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY))desc) as
rank_of_top_states,
dense_rank() over(order by
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY))) as
rank_of_bottom_states,
case when dense_rank() over(order by
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY))desc)
<=5 then 'top_class' else 'bottom_class' end as category
from `target_project.customers` c
inner join target_project.orders o
on c.customer_id = o.customer_id
inner join `target_project.order_items` ot
on o.order_id = ot.order_id
group by c.customer_state)

(select customer_state, Avg_time_to_deliver, rank_of_top_states as rank, category
from Average_value
order by Avg_time_to_deliver desc
limit 5)
union all
(select customer_state, Avg_time_to_deliver, rank_of_bottom_states as rank,
category
from Average_value
order by Avg_time_to_deliver asc
limit 5)
```

Row	customer_state	Avg_time_to_deliver	rank	category
1	RR	27.826086956521738	1	top_class
2	AP	27.753086419753075	2	top_class
3	AM	25.963190184049076	3	top_class
4	AL	23.992974238875881	4	top_class
5	PA	23.301707779886126	5	top_class
6	SP	8.25960855241909	1	bottom_class
7	PR	11.480793060718735	2	bottom_class
8	MG	11.515522180072811	3	bottom_class
9	DF	12.501486199575384	4	bottom_class
10	SC	14.520985846754517	5	bottom_class

`Insight:` As I can observe based on the analysis, states of (RR, PB, RO, AC, PI) stood as top 5 states with highest average time to deliver, where as in contrast states of (SP, PR, MG, RJ, DF) stood as bottom states with lowest average. Now, what's the reason for the low average of top states with most number of orders, well the reason will be the same as in the above question I stated, because of more number of orders, the average will be significantly affected.

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

Ans.

```
with Average_value as
(select c.customer_state,
avg(date_diff(order_estimated_delivery_date,order_delivered_customer_date, DAY)) as
actual_Avg_estimated_diff,
dense_rank() over(order by
avg(date_diff(order_estimated_delivery_date,order_delivered_customer_date,
DAY))desc) as rank_of_top_states
from `target_project.customers` c
inner join target_project.orders o
on c.customer_id = o.customer_id
inner join `target_project.order_items` ot
on o.order_id = ot.order_id
group by c.customer_state)

select customer_state, actual_Avg_estimated_diff, rank_of_top_states
from Average_value
order by actual_Avg_estimated_diff desc
limit 5
```

Row	customer_state	actual_Avg_estimated_diff	rank_of_top_states
1	AC	20.010989010989018	1
2	RO	19.080586080586084	2
3	AM	18.975460122699381	3
4	AP	17.444444444444443	4
5	RR	17.434782608695652	5

Insight: As I can observe based on the analysis, states of (AC, RO, AM, AP, RR) stood highest In terms of Avg of difference between estimated delivery date and actual delivery date. The highest average difference is approx. 20 days for the state AC. If we observe deeply all these above states have less no of orders and hence their average will be significantly higher than the other states.

6. **Analysis based on the payments:**

(i)Find the month on month no. of orders placed using different payment types.
Ans.

```
select p.payment_type, extract(month from o.order_purchase_timestamp) as months,
count(*) as no_of_orders
from `target_project.orders` o
inner join target_project.payments p
on o.order_id = p.order_id
group by p.payment_type, months
order by months, payment_type;
```

Row	payment_type	months	no_of_orders
1	UPI	1	1715
2	credit_card	1	6103
3	debit_card	1	118
4	voucher	1	477
5	UPI	2	1723
6	credit_card	2	6609
7	debit_card	2	82
8	voucher	2	424
9	UPI	3	1942
10	credit_card	3	7707
11	debit_card	3	109
12	voucher	3	591

Insight: From the analysis, I can observe that the number of orders data is divided under four payment types, further grouped with respect to each month corresponding from the time range. Also based on the above data I could observe that, greater number of orders were from credit card type from almost in every month. And low number of orders are from debit card type.

(ii)Find the no. of orders placed on the basis of the payment instalments that have been paid.

Ans.

```
select p.payment_installments, count(*) as no_of_orders
from `target_project.orders` o
inner join target_project.payments p
on o.order_id = p.order_id
group by  p.payment_installments
order by  p.payment_installments;
```

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

Insight: Here, the number of orders were grouped under payment instalments and among the 24 types of instalments options, highest orders were from 1 type (52546 orders) and the lowest orders were from 22 and 23 types with only 1 order each. It means most of the customers are just paying in one instalment for the orders booked.