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.**

Ans:

```
SELECT column_name, data_type
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers'
```

Screenshot/output:

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

Or

	Field name	Type	Mode
☐	customer_id	STRING	NULLABLE
☐	customer_unique_id	STRING	NULLABLE
☐	customer_zip_code_prefix	INTEGER	NULLABLE
☐	customer_city	STRING	NULLABLE
☐	customer_state	STRING	NULLABLE

Inference/Insights:
We can get this information directly due to BigQuery's UI which is shown in 2nd Screenshot. Or we can use the query written above for the same output which is shown in screenshot 1.

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

Ans:

```
SELECT MAX(order_purchase_timestamp) AS last_order_timestamp,
MIN(order_purchase_timestamp) AS first_order_timestamp
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`
```

Screenshot/output:

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

Inference/Insights:
According to the data set, the initial order is placed on September 4, 2016 at 15:19:00 UTC, and the last order is placed on October 17, 2018 at 17:30:18 UTC.

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

Ans:

```
SELECT COUNT(DISTINCT(geolocation_city)) AS total_city,
COUNT(DISTINCT(geolocation_state)) AS total_state
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders` o JOIN
`focused-beacon-393103.Business_Case_Study_Target_SQL.customers` c ON
o.customer_id=c.customer_id
JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.geolocation` g ON
c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
```

Screenshot/output:

Row	total_city	total_state
1	5812	27

Inference/Insights:
Because there are no null values or missing data when linking tables.
Customers from 5812 cities and 27 states placed orders that are non-repeated between 2016-09-04 21:15:19 UTC and 2018-10-17 17:30:18 UTC.

2. **In-depth Exploration:**
   1. **Is there a growing trend in the no. of orders placed over the past years?**

Ans:

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year, COUNT(order_id) AS
number_of_orders
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`
GROUP BY 1
ORDER BY 1
```

Screenshot/output:

year	number_of_orders
2016	329
2017	45101
2018	54011

Inference/Insights:
We can see how many orders have been placed per year from the snapshot, but we cannot compute the percentage rise or compare it year to year because we do not have complete data on orders for the entire years of 2016 and 2018, as some months are missing.

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

Ans:

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year, EXTRACT(MONTH
FROM order_purchase_timestamp) AS month, COUNT(order_id) AS number_of_orders
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`
GROUP BY 1,2
ORDER BY 1,2
```

Screenshot/output:

year	month	number_of_orders
2016	9	4
2016	10	324
2016	12	1
2017	1	800
2017	2	1780
2017	3	2682
2017	4	2404
2017	5	3700
2017	6	3245
2017	7	4026
2017	8	4331
2017	9	4285
2017	10	4631
2017	11	7544
2017	12	5673
2018	1	7269
2018	2	6728
2018	3	7211
2018	4	6939
2018	5	6873
2018	6	6167
2018	7	6292
2018	8	6512
2018	9	16
2018	10	4

Inference/Insights:
December 2016 is missing from the data collection. We can observe that the number of orders increased significantly in October 2016 compared to the previous month, but decreased significantly in December of the same year.
Because of the forthcoming holiday season in the country, the biggest number of orders in 2017 were in November, and the lowest number of orders were in January. In addition, with the exception of the first quarter of 2017, the second month always has the highest number of orders in that quarter, followed by the third month.
2018 began with the highest number of orders in that year, and the number of orders remained very high throughout the year when compared to the previous year's months. The number of orders was extremely low in September and October of 2018.

3. **During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)**
    1. **0-6 hrs : Dawn**
    2. **7-12 hrs : Mornings**
    3. **13-18 hrs : Afternoon**

4. **19-23 hrs : Night**

Ans:

```
WITH base AS (
SELECT order_id,
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'
ELSE 'night'
END AS time_of_day
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`)

SELECT time_of_day, COUNT(order_id) as number_of_orders
FROM base
GROUP BY 1
ORDER BY 2
```

Screenshot/output:

time_of_day	number_of_orders
afternoon	38135
night	28331
morning	27733
dawn	5242

Inference/Insights:

Customers typically place their orders in the afternoon, followed by the night, morning, and dawn.

3. **Evolution of E-commerce orders in the Brazil region:**
    1. **Get the month on month no. of orders placed in each state.**

ANS:

```
WITH monthly_order AS (
SELECT customer_id, EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`)

SELECT customer_state, year, month, COUNT(m.customer_id) AS number_of_orders
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.customers` c JOIN
monthly_order m ON c.customer_id = m.customer_id
GROUP BY 1,2,3
ORDER BY 4 DESC
```

Screenshot:

customer_state	year	month	number_of_orders
SP	2018	8	3253
SP	2018	5	3207
SP	2018	4	3059
SP	2018	1	3052
SP	2018	3	3037
SP	2017	11	3012
SP	2018	7	2777
SP	2018	6	2773
SP	2018	2	2703
SP	2017	12	2357

Inference/Insights:
The screenshot shows the number of orders placed in each state in each month. Also, when compared to the other states, Sao Paulo has the highest number of orders throughout the year, with August 2018 being the peak.

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

```
SELECT customer_state, COUNT(customer_id) AS number_of_customers
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.customers`
GROUP BY 1
ORDER BY 2 DESC
```

Screenshot:

customer_state	number_of_customers
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033
GO	2020
PE	1652
CE	1336
PA	975
MT	907
MA	747

Inference/Insights:

Most of the customers are from Sao Paulo and lowest number of customers are from Roraima.
Note: COUNT(customer_id), COUNT(customer_unique_id), COUNT(DISTINCT(customer_id)) all of them are showing the same results.

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).**
       You can use the "payment_value" column in the payments table to get the cost of orders.

ANS:

```
WITH base AS
(SELECT order_id, order_purchase_timestamp
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`
WHERE (EXTRACT(YEAR FROM order_purchase_timestamp) BETWEEN 2017 AND 2018) AND
(EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8)),

base1 AS (
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
ROUND(SUM(payment_value),2) AS total_cost
FROM base b JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.payments`
p ON b.order_id=p.order_id
GROUP BY 1),

base2 AS (
SELECT *, LAG(total_cost) OVER (ORDER BY total_cost) AS previous_year_cost
FROM base1)

SELECT year, total_cost, ROUND(IFNULL(((total_cost-
previous_year_cost)/previous_year_cost)*100,0),0) AS percentage_increase
FROM base2
ORDER BY 1
```

Screenshot:

year	total_cost	percentage_increase
2017	3669022.12	0.0
2018	8694733.84	137.0

Inference/Insights:
In 2018, there is an increase of 137% in the cost of orders. So, it could mean several meanings like an indicate that the prices of the products being ordered have increased, or that customers are ordering more expensive items. It could also suggest that there is an increase in demand for the products being offered, which is driving up the prices.

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

ANS:

```
SELECT customer_state, ROUND(SUM(price),2) AS total_value, ROUND(AVG(price),2) AS
average_value
```

```
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.order_items` p JOIN
`focused-beacon-393103.Business_Case_Study_Target_SQL.orders` o ON
p.order_id=o.order_id
JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.customers` c ON
o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 1
```

Screenshot:

customer_state	total_value	average_value
AC	15982.95	173.73
AL	80314.81	180.89
AM	22356.84	135.5
AP	13474.3	164.32
BA	511349.99	134.6
CE	227254.71	153.76
DF	302603.94	125.77
ES	275037.31	121.91
GO	294591.95	126.27
MA	119648.22	145.2

Inference/Insights:
As per the result, Sao Paulo has the highest Total value of order price which is 5202955.05 and Paraiba has the highest average value of order price which is 191.48. This suggests that while Sao Paulo may have a larger volume of orders, the average value of each order in Paraiba is higher.

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

```
SELECT customer_state, ROUND(SUM(freight_value),2) AS total_freight_value,
ROUND(AVG(freight_value),2) AS average_freight_value
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.order_items` oi JOIN
`focused-beacon-393103.Business_Case_Study_Target_SQL.orders` o ON
oi.order_id=o.order_id
JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.customers` c ON
o.customer_id=c.customer_id
GROUP BY 1
ORDER BY 1
```

Screenshot:

customer_state	total_freight_value	average_freight_valu
AC	3686.75	40.07
AL	15914.59	35.84
AM	5478.89	33.21
AP	2788.5	34.01
BA	100156.68	26.36
CE	48351.59	32.71
DF	50625.5	21.04
ES	49764.6	22.06
GO	53114.98	22.77
MA	31523.77	38.26

Inference/Insights:
As per the result, the total cost of shipping or transporting goods for all orders in São Paulo is 718723.07, which is the highest, while the average cost of shipping or transporting goods for each order in Roraima is 42.98, which is the highest.

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:
        1. **time_to_deliver = order_delivered_customer_date - order_purchase_timestamp**
        2. **diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date**

    ANS:
    ```
    SELECT order_id, DATE_DIFF(order_delivered_customer_date,
    order_purchase_timestamp, DAY) AS time_to_deliver,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS
    diff_estimated_delivery
    FROM (SELECT order_id, order_delivered_customer_date, order_purchase_timestamp,
    order_estimated_delivery_date FROM `focused-beacon-
    393103.Business_Case_Study_Target_SQL.orders` WHERE
    order_delivered_customer_date IS NOT NULL) a
    ```

    Screenshot:

order_id	time_to_deliver	diff_estimated_delive
1950d777989f6a877539f53795b4c3c3	30	-12
2c45c33d2f9cb8ff8b1c86cc28c11c30	30	28
65d1e226dfaeb8cdc42f665422522d14	35	16
635c894d068ac37e6e03dc54eccb6189	30	1
3b97562c3aee8bdedcb5c2e45a50d5e1	32	0
68f47f50f04c4cb6774570cfde3a9aa7	29	1
276e9ec344d3bf029ff83a161c6b3ce9	43	-4
54e1a3c2b97fb0809da548a59f64c813	40	-4
fd04fa4105ee8045f6a0139ca5b49f27	37	-1
302bb8109d097a9fc6e9cefc5917d1f3	33	-5

Inference/Insights:
In the 'diff_estimated_delivery' column, a negative value indicates how many days it took to deliver the order beyond the estimated delivery date, while a positive value means how early the order has been delivered. Time is measured in days.

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

```
WITH base AS (
SELECT customer_state, ROUND(AVG(freight_value),2) AS average_freight_value
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.order_items` oi JOIN
`focused-beacon-393103.Business_Case_Study_Target_SQL.orders` o ON
oi.order_id=o.order_id
JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.customers` c ON
o.customer_id=c.customer_id
GROUP BY 1),

base1 AS (
(SELECT *
FROM base
ORDER by average_freight_value
LIMIT 5)
UNION ALL
(SELECT *
FROM base
ORDER by average_freight_value DESC
LIMIT 5))

SELECT * FROM base1 ORDER BY 2
```

Screenshot:

customer_state	average_freight_valu
SP	15.15
PR	20.53
MG	20.63
RJ	20.96
DF	21.04
PI	39.15
AC	40.07
RO	41.07
PB	42.72
RR	42.98

customer_state

Inference/Insights:
As per the screenshot, the first 5 rows show the top 5 states with the lowest average freight value arranged in ascending order, and the last 5 rows show the top 5 states with the highest average freight value arranged in ascending order.

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

```
WITH base AS (
SELECT order_id, customer_id, DATE_DIFF(order_delivered_customer_date,
order_purchase_timestamp, DAY) AS time_to_deliver
FROM (SELECT order_id, customer_id, order_delivered_customer_date,
order_purchase_timestamp FROM `focused-beacon-
393103.Business_Case_Study_Target_SQL.orders` WHERE order_delivered_customer_date
IS NOT NULL) a),

base1 AS (
SELECT customer_state, ROUND(AVG(time_to_deliver),0) AS average_time_to_deliver
FROM base b JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.customers`
c ON b.customer_id=c.customer_id
GROUP BY 1),

base2 AS (
(SELECT *
FROM base1
ORDER by average_time_to_deliver
LIMIT 5)
UNION ALL
(SELECT *
FROM base1
ORDER by average_time_to_deliver DESC
LIMIT 5))

SELECT * FROM base2 ORDER BY 2
```

Screenshot:

customer_state	average_time_to_deliver
SP	8.0
MG	12.0
PR	12.0
DF	13.0
SC	14.0
PA	23.0
AL	24.0
AM	26.0
AP	27.0
RR	29.0

Inference/Insights:
As per the screenshot, the first 5 rows show the top 5 states with the lowest average delivery time arranged in ascending order, and the last 5 rows show the top 5 states with the highest average delivery time arranged in ascending order. Time is measured in days.

4. **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.

ANS:

```
WITH base AS (
SELECT order_id, customer_id, DATE_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY) AS diff_estimated_delivery
FROM (SELECT order_id, customer_id, order_delivered_customer_date,
order_estimated_delivery_date FROM `focused-beacon-
393103.Business_Case_Study_Target_SQL.orders` WHERE order_delivered_customer_date
IS NOT NULL) a)

SELECT customer_state, AVG(diff_estimated_delivery) AS average_diff_estimated_delivery
FROM base b JOIN `focused-beacon-393103.Business_Case_Study_Target_SQL.customers`
c ON b.customer_id=c.customer_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5
```

Screenshot:

customer_state	average_diff_estimat
AC	19.76250000000...
RO	19.13168724279...
AP	18.73134328358...
AM	18.60689655172...
RR	16.41463414634...

Inference/Insights:

The values in the 'average_diff_estimated_delivery' column is in days and arranged in descending order. Therefore, in the state of Acre, orders are delivered 19 days earlier than the estimated delivery date and the whole table shows the top 5 state where the order delivered is really fast

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

ANS:

```
WITH monthly_order AS (
SELECT order_id, EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.orders`)

SELECT year, month, payment_type, COUNT(o.order_id) AS number_of_orders
FROM monthly_order o JOIN `focused-beacon-
393103.Business_Case_Study_Target_SQL.payments` p ON o.order_id=p.order_id
GROUP BY 1,2,3
ORDER BY 1,2,3
```

Screenshot:

year	month	payment_type	number_of_orders
2016	9	credit_card	3
2016	10	UPI	63
2016	10	credit_card	254
2016	10	debit_card	2
2016	10	voucher	23
2016	12	credit_card	1
2017	1	UPI	197
2017	1	credit_card	583
2017	1	debit_card	9
2017	1	voucher	61
2017	2	UPI	398
2017	2	credit_card	1356
2017	2	debit_card	13

Inference/Insights:

This table shows the monthly and yearly numbers of orders made using different payment types. From the table, we can see that most orders are made via credit card, followed by UPI.

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

ANS:

```
SELECT payment_installments, COUNT(order_id) AS number_of_orders
FROM `focused-beacon-393103.Business_Case_Study_Target_SQL.payments`
WHERE payment_installments>0 AND payment_value>0
GROUP BY 1
```

Screenshot:

payment_installment	number_of_orders
1	52537
2	12413
3	10461
4	7098
5	5239
6	3920
7	1626
8	4268
9	644
10	5328

Inference/Insights:
From the table we can conclude that most of the orders have been made in a single instalment.