1.
Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset.
A)Data type of all columns in the “customers” table.
QUERY:-

```
SELECT

COLUMN_NAME, DATA_TYPE

FROM

`targetsql-403414.TargetSQL.INFORMATION_SCHEMA.COLUMNS`

WHERE

TABLE_NAME = 'customers';
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION | RESULTS | CHART PREVIEW | 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

PERSONAL HISTORY PROJECT HISTORY REFRESH

INSIGHTS:-
Most of the columns of CUSTOMERS table are of string type.

B)Get the time range between which the orders were placed.
QUERY:-

```
SELECT MIN(order_purchase_timestamp) AS `FROM`,MAX(order_purchase_timestamp) AS `TO` FROM (
  SELECT * FROM `targetsql-403414.TargetSQL.orders`
  WHERE order_status = 'created'
  ORDER BY order_purchase_timestamp
);
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

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

Row	FROM	TO
1	2017-11-06 13:12:34 UTC	2018-02-09 17:21:04 UTC

INSIGHT:-
Orders started from 06th Nov 2017 till 9th feb 2018. That is winter period.
C) Count the Cities & States of customers who ordered during the given period.
QUERY:-

```sql
SELECT
  COUNT(DISTINCT `customer_city`) AS City_count,
  COUNT(DISTINCT `customer_state`) AS State_count
FROM
`TargetSQL.customers` C
INNER JOIN
`TargetSQL.orders` O
USING(customer_id);
```

RESULT:-

Query results

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

Row	City_count	State_count
1	4119	27

INSIGHT:-
There are total 4119 cities and 27 states where orders were placed.

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

```sql
SELECT Year,Count(Year) As `Count` FROM
(SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS Year
FROM `TargetSQL.orders`)
GROUP BY Year
ORDER BY 1;
```

Result:-

Query results

SAVE RESULTS EXPLORE DATA

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

Row	Year	Count
1	2016	329
2	2017	45101
3	2018	54011

INSIGHT:-
Yes we can clearly see growth around 20% in the year 2017 and 2018 orders.
B)Can we see some kind of monthly seasonality in terms of the no. of orders being placed?
Query:-

```sql
SELECT month, COUNT(*) AS `count` FROM (
  SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS month
  FROM `TargetSQL.orders`
)
GROUP BY month
ORDER BY 2 DESC;
```

RESULT:-

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON

Row	month	count
1	8	10843
2	5	10573
3	7	10318
4	3	9893

INSIGHT:-
In The month August Sales was at its PEAK whereas sales in the months May and July was also High.

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

QUERY:-

```sql
SELECT
CASE
  WHEN Hour >= 0 AND Hour <= 6 THEN 'Dawn'
  WHEN Hour >= 7 AND Hour <= 12 THEN 'Mornings'
  WHEN Hour >= 13 AND Hour <= 18 THEN 'Afternoon'
  WHEN Hour >= 19 AND Hour <= 23 THEN 'Night'
  END
AS Daytime,
Count(*) AS `number_of_orders`
FROM (SELECT EXTRACT(HOUR FROM order_purchase_timestamp) AS Hour
FROM `TargetSQL.orders`)
GROUP BY 1
ORDER BY 2 DESC
;
```

RESULT:-

Query results

SAVE RESU

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

Row	Daytime	number_of_orders
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

INSIGHT:-

Most of the customers in brazil place their order during Afternoon.Whereas very few people place orders during Dawn time.

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

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

QUERY:-

```sql
SELECT
  C.customer_state,
  EXTRACT(MONTH FROM O.order_purchase_timestamp) AS month,
  COUNT(C.customer_state) AS `Number Of Orders`
FROM `TargetSQL.orders` O
INNER JOIN `TargetSQL.customers` C
USING(`customer_id`)
GROUP BY 1,2
ORDER BY 1,2;
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

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

Row	customer_state	month	Number Of Orders
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

INSIGHT:-sales is Highest SP,RJ and MG region.

B)How are the customers distributed across all the states?

QUERY:-

```sql
SELECT
  `customer_state`,
  Count(DISTINCT customer_id) AS count FROM `TargetSQL.customers`
GROUP BY 1
ORDER BY 2
DESC;
```

RESULT:-

Query results

SAVE RESULTS EXI

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION G

Row	customer_state	count
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

INSIGHT:-
No. of customers in SP state is 3 times higher than the other state which is ranked at 2nd position which is also the highest in all states.

4.Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.
A)Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
QUERY:-

```
SELECT YEAR, ROUND(SUM(payment_value),2) AS `COST` FROM (
  SELECT
    EXTRACT(YEAR FROM O.order_purchase_timestamp) AS YEAR,
    EXTRACT(MONTH FROM O.order_purchase_timestamp) AS MONTH,
    P.payment_value
  FROM `TargetSQL.orders` O INNER JOIN `TargetSQL.payments` P
  USING(order_id)
)
WHERE YEAR IN(2017,2018) AND MONTH >=8
GROUP BY YEAR;
```

RESULT:-

Query results

SAVE RESULTS EXF

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION G

Row	YEAR	COST
1	2017	4255120.93
2	2018	1027454.53

INSIGHTS:-
Order cost in year 2018 is almost 4 times lower than 2017.

B)Calculate the Total & Average value of order price for each state.
QUERY:-

4. B RUN SAVE QUERY SHARE SCHEDULE MORE Query completed.

```google_sql_query
SELECT
 DISTINCT C.customer_state,
 ROUND(SUM(P.payment_value) OVER(partition by C.customer_state),2) AS Total_Price,
 ROUND(SUM(P.payment_value) OVER (PARTITION BY C.customer_state) / COUNT(*) OVER (PARTITION BY C.customer_state),2) AS Average_Price
FROM `TargetSQL.customers` C
LEFT JOIN `TargetSQL.orders` O
USING(customer_id)
LEFT JOIN `TargetSQL.payments` P
USING(order_id);
```

RESULT:-

Query results SAVE RESULTS EXPLORE DATA

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

Row	customer_state	Total_Price	Average_Price
1	MA	152523.02	198.86
2	RO	60866.2	233.2
3	AP	16262.8	232.33
4	AC	19680.62	234.29
5	SE	75246.25	208.44
6	TO	61485.33	204.27
7	PI	108523.97	207.11
8	AM	27966.93	181.6
9	DF	355141.08	161.13
10	MG	1872257.26	154.71

INSIGHT:- SP region has the highest Total_price to average price ratio.

C)Calculate the Total & Average value of order freight for each state.
QUERY:-

```google_sql_query
SELECT
 DISTINCT C.customer_state,
 ROUND(SUM(OI.freight_value) OVER(partition by C.customer_state),2) AS total_freight_value,
 ROUND(SUM(OI.freight_value) OVER (PARTITION BY C.customer_state) / COUNT(*) OVER (PARTITION BY C.customer_state),2) AS average_freight_value
FROM `TargetSQL.order_items` OI
INNER JOIN `TargetSQL.orders` O
USING(order_id)
INNER JOIN `TargetSQL.customers` C
USING(customer_id);
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

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

Row	customer_state	total_freight_value	average_freight_valu
1	BA	100156.68	26.36
2	SC	89660.26	21.47
3	RS	135522.74	21.74
4	PI	21218.2	39.15
5	ES	49764.6	22.06
6	RR	2235.19	42.98
7	SP	718723.07	15.15
8	CE	48351.59	32.71
9	RO	11417.38	41.07
10	RJ	305589.31	20.96

INSIGHTS:-

5.Analysis based on sales, freight and delivery time.
A)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.
QUERY:-

```
SELECT
  order_id,
  TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_deliever,
  TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date, Day) AS diff_estimated_delievery
FROM `TargetSQL.orders`
WHERE order_status = 'delivered';
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

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

Row	order_id	time_to_deliever	diff_estimated_delie
1	635c894d068ac37e6e03dc54e...	30	1
2	3b97562c3aee8bdedcb5c2e45...	32	0
3	68f47f50f04c4cb6774570cfde...	29	1
4	276e9ec344d3bf029ff83a161c...	43	-4
5	54e1a3c2b97fb0809da548a59...	40	-4
6	fd04fa4105ee8045f6a0139ca5...	37	-1
7	302bb8109d097a9fc6e9cefc5...	33	-5
8	66057d37308e787052a32828...	38	-6
9	19135c945c554eebfd7576c73...	36	-2
10	4493e45e7ca1084efcd38ddeb...	34	0

INSIGHTS:-
Most of the order take around 32 days to deliver and are usually late.

B)Find out the top 5 states with the highest & lowest average freight value.
QUERY:-

```
SELECT * FROM (
  SELECT
  DISTINCT customer_state,
  ROUND(SUM(OI.freight_value) OVER (PARTITION BY C.customer_state) / COUNT(*) OVER (PARTITION BY C.customer_state),2)
AS average_freight_value
  FROM `TargetSQL.order_items` OI
  INNER JOIN `TargetSQL.orders` O
  USING(order_id)
  INNER JOIN `TargetSQL.customers` C
  USING(customer_id)
  ORDER BY 2
  LIMIT 5
) A
UNION ALL
SELECT * FROM (
  SELECT
  DISTINCT customer_state,
  ROUND(SUM(OI.freight_value) OVER (PARTITION BY C.customer_state) / COUNT(*) OVER (PARTITION BY C.customer_state),2)
AS average_freight_value
  FROM `TargetSQL.order_items` OI
  INNER JOIN `TargetSQL.orders` O
  USING(order_id)
  INNER JOIN `TargetSQL.customers` C
  USING(customer_id)
  ORDER BY 2 DESC
  LIMIT 5
)
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

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

Row	customer_state	average_freight_valu
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04
6	PI	39.15
7	AC	40.07
8	RO	41.07
9	PB	42.72
10	RR	42.98

INSIGHT:-The lowest average freight value is around 15 for SP whereas highest is 42.98 for RR.
C)Find out the top 5 states with the highest & lowest average delivery time.
QUERY:-

```sql
SELECT * FROM (SELECT
DISTINCT customer_state,
ROUND(
  SUM(
    TIMESTAMP_DIFF(
      order_delivered_customer_date,
      order_purchase_timestamp,
      DAY)
    )
    OVER (PARTITION BY C.customer_state) / COUNT(*) OVER (PARTITION BY C.customer_state),2) AS average_delievery_time
FROM `TargetSQL.orders` O
INNER JOIN `TargetSQL.customers` C
USING(customer_id)
ORDER BY 2
LIMIT 5
) A
UNION ALL
SELECT * FROM (
  SELECT
DISTINCT customer_state,
ROUND(
  SUM(
    TIMESTAMP_DIFF(
      order_delivered_customer_date,
      order_purchase_timestamp,
      DAY)
    )
    OVER (PARTITION BY C.customer_state) / COUNT(*) OVER (PARTITION BY C.customer_state),2) AS average_delievery_time
FROM `TargetSQL.orders` O
INNER JOIN `TargetSQL.customers` C
USING(customer_id)
ORDER BY 2 DESC
LIMIT 5
) B
ORDER BY average_delievery_time;
```

RESULT:-

Query results

SAVE RESULTS EXPLORE DATA

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

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

INSIGHT:-

The average delivery time is lowest in 8.05 in SP whereas It is 26.34 in AP.

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

QUERY:-

```sql
SELECT customer_state FROM (
  SELECT
DISTINCT customer_state,
FLOOR(
  SUM(
  TIMESTAMP_DIFF(
    order_delivered_customer_date, order_purchase_timestamp, DAY
  )
) OVER(PARTITION BY customer_state)/COUNT(*) OVER(PARTITION BY customer_state)
)
AS actual_delievery_days,
FLOOR(
  SUM(
  TIMESTAMP_DIFF(
    order_estimated_delivery_date, order_purchase_timestamp, DAY
  )
) OVER(PARTITION BY customer_state)/COUNT(*) OVER(PARTITION BY customer_state)
)
AS estimated_delievery_days
FROM `TargetSQL.orders`
INNER JOIN `TargetSQL.customers`
USING(customer_id)
WHERE 1 < 2 AND order_status = 'delivered'
ORDER BY 1
)
LIMIT 5;
```

RESULT:-

Query results

SAVE RESULTS

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

Row	customer_state
1	AC
2	AL
3	AM
4	AP
5	BA

INSIGHT:-
These are top five states where delivery is really fast even faster than estimated delivery date.

6.Analysis based on the payments:
A)Find the month on month no. of orders placed using different payment types.
QUERY:-

```sql
SELECT
  EXTRACT(MONTH FROM O.order_purchase_timestamp) AS Month,payment_type, COUNT(*) AS Number_of_orders
FROM `TargetSQL.payments` P LEFT JOIN
`TargetSQL.orders` O
USING(order_id)
GROUP BY EXTRACT(MONTH FROM order_purchase_timestamp),payment_type
```

RESULT:-

Query results

SAVE RESULTS

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

Row	Month	payment_type	Number_of_orders
1	5	credit_card	8350
2	4	credit_card	7301
3	1	voucher	477
4	4	voucher	572
5	10	voucher	318
6	9	not_defined	1
7	8	not_defined	2
8	6	voucher	563
9	5	voucher	613
10	3	voucher	591

INISGHT:- Most of the orders are coming from credit card.months do not seem to have any effect on no. of orders with respect to payment types.

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

QUERY:-

```
SELECT payment_installments,COUNT(*) AS No_of_orders FROM `TargetSQL.payments`
WHERE payment_value > 0
GROUP BY payment_installments;
```

RESULT:-

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

Row	payment_installment	No_of_orders
1	0	2
2	1	52537
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:-

Very few people place orders on 0 EMIs and option for emi greater than 10 months.