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

1.1 Data type of all columns in the "customers" table.

Ans:

SELECT \*, data\_type
FROM `shubham-scaler.Target.INFORMATION\_SCHEMA.COLUMNS`
WHERE table\_name= 'Customers'

## Output:

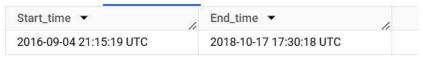


1.2 Get the time range between which the orders were placed.

Ans:

SELECT MIN(order\_purchase\_timestamp) Start\_time, MAX(order\_purchase\_timestamp) End\_time FROM `Target.Orders`

# Output:



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

Ans:

SELECT Count(distinct customer\_city) Cities, Count(distinct customer\_state) States FROM `Target.Customers`C JOIN `Target.Orders` O ON C.customer\_id=O.customer\_id Output:



# 2.In-depth Exploration:

2.1Is there a growing trend in the no. of orders placed over the past years?

## Ans:

WITH Yearly\_orders AS

(SELECT EXTRACT(YEAR from order\_purchase\_timestamp) Order\_year, Count(distinct order\_id)

Total\_orders

FROM `Target.Orders`

Group BY Order\_year)

SELECT Order\_year, Total\_orders, LAG(Total\_orders)OVER(ORDER BY Order\_year)

Previous\_year\_orders,

ROUND((Total\_orders/LAG(Total\_orders)OVER(ORDER BY Order\_year)\*100),2) Yearly\_growth

FROM Yearly\_orders

ORDER BY Order\_year

# Output:

Row //	Order_year ▼	Total_orders ▼	Previous_year_orders	Yearly_growth ▼
1	2016	329	nuli	nuli
2	2017	45101	329	13708.51
3	2018	54011	45101	119.76

2.2Can we see some kind of monthly seasonality in terms of the no. of orders being placed? Ans:

SELECT EXTRACT(MONTH FROM order\_purchase\_timestamp) Monthly\_orders, COUNT(order\_id)

Total\_orders

FROM `Target.Orders`

GROUP BY Monthly\_orders

ORDER BY Monthly\_orders

Row /	Monthly_orders	Total_orders	· //
1	1		8069
2	2	1	8508
3	3	3	9893
4	1	L.	9343
5	5	5	10573
6	$\epsilon$	5	9412
7	7		10318
8	8	3	10843
9	ç	)	4305
10	10	)	4959

2.3During 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

## Ans:

SELECT 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 Time\_of\_order,

COUNT(order\_id) Total\_orders

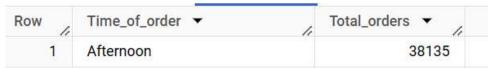
FROM `Target.Orders`

GROUP BY Time\_of\_order

ORDER BY Total\_orders DESC

LIMIT 1

# Output:



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

3.1Get the month on month no. of orders placed in each state.

# Ans:

SELECT C.customer\_state,EXTRACT(Month FROM O.order\_purchase\_timestamp) Month, COUNT(O.order\_id) Total\_orders FROM `Target.Customers` C JOIN `Target.Orders` O ON C.customer\_id=O.customer\_id GROUP BY C.customer\_state, Month ORDER BY Month



## 3.2 How are the customers distributed across all the states?

#### Ans:

SELECT customer\_state, COUNT(customer\_id) No\_of\_customers FROM `Target.Customers` GROUP BY customer\_state
ORDER BY No\_of\_customers DESC

## Output:

	No_of_customers	customer_state ▼	Row /
	41746	SP	1
	12852	RJ	2
	11635	MG	3
	5466	RS	4
	5045	PR	5
	3637	SC	6
	3380	ВА	7
	2140	DF	8
	2033	ES	9
	2020	GO	10

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

4.1 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 Yearly\_order as

(SELECT EXTRACT(Year FROM 0.order\_purchase\_timestamp) Order\_year, SUM(P.payment\_value)

Total\_costs

FROM `Target.Payments` P JOIN `Target.Orders` O ON P.order\_id=0.order\_id

WHERE EXTRACT(Month FROM 0.order\_purchase\_timestamp) BETWEEN 1 AND 8

GROUP BY Order\_year)

SELECT Round(C1.Total\_costs,2) Total\_cost\_2017, Round(C2.Total\_costs,2) Total\_cost\_2018,

ROUND(((C2.Total\_costs-C1.Total\_costs)/C1.Total\_costs)\*100,2) Percentage\_increase

FROM (SELECT Total\_costs FROM Yearly\_order WHERE Order\_year=2017)C1,(SELECT Total\_costs FROM Yearly\_order WHERE Order\_year=2017)C1,(SELECT Total\_costs FROM Yearly\_order WHERE Order\_year=2017)C1,(SELECT Total\_costs FROM Yearly\_order WHERE Order\_year=2018)C2



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

#### Ans:

SELECT C.customer\_state, ROUND(SUM(P.payment\_value),2) Total\_value,
ROUND(AVG(P.payment\_value),2) Average\_value
FROM `Target.Orders` O JOIN `Target.Payments` P ON O.order\_id=P.order\_id JOIN `Target.Customers` C
ON O.customer\_id=C.customer\_id
GROUP BY C.customer\_state
ORDER BY Total\_value

## Output:

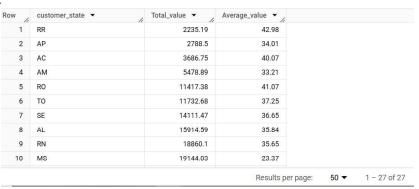
Average_value ▼ //	Total_value ▼	customer_state ▼	Row /
218.8	10064.62	RR	1
232.33	16262.8	AP	2
234.29	19680.62	AC	3
181.6	27966.93	AM	4
233.2	60866.2	RO	5
204.27	61485.33	TO	6
208.44	75246.25	SE	7
227.08	96962.06	AL	8
196.78	102718.13	RN	9
207.11	108523.97	PI	10

# 4.3 Calculate the Total & Average value of order freight for each state

## Ans:

SELECT C.customer\_state, ROUND(SUM(OI.freight\_value),2) Total\_value, ROUND(AVG(OI.freight\_value),2) Average\_value
FROM `Target.Orders` O JOIN `Target.Order\_items` OI ON O.order\_id=OI.order\_id JOIN `Target.Customers`
C ON O.customer\_id=C.customer\_id
GROUP BY C.customer\_state

ORDER BY Total\_value



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

5.1 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.

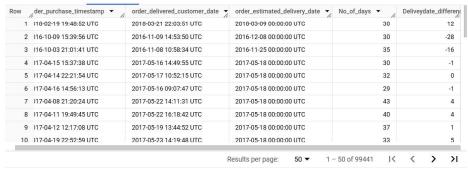
You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- time\_to\_deliver = order\_delivered\_customer\_date order\_purchase\_timestamp
- diff\_estimated\_delivery = order\_delivered\_customer\_date order\_estimated\_delivery\_date

#### Ans:

SELECT order\_purchase\_timestamp, order\_delivered\_customer\_date,order\_estimated\_delivery\_date, DATE\_DIFF(order\_delivered\_customer\_date,order\_purchase\_timestamp,Day) No\_of\_days, DATE\_DIFF(order\_delivered\_customer\_date,order\_estimated\_delivery\_date,Day) Deliveydate\_difference FROM `Target.Orders`

## Output:



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

#### Ans:

WITH Freight AS

(SELECT C.customer\_state, DENSE\_RANK()OVER(ORDER BY AVG(OI.freight\_value)DESC) Desc\_rank, DENSE\_RANK()OVER(ORDER BY AVG(OI.freight\_value)) Asc\_rank
FROM `Target.Customers` C JOIN `Target.Orders` O ON C.customer\_id=O.customer\_id
JOIN `Target.Order\_items` OI ON O.order\_id=OI.order\_id
GROUP BY C.customer\_state)
SELECT FI.customer\_state Highest\_freightvalue, FO.customer\_state Lowest\_freightvalue
FROM Freight FI JOIN Freight FO ON FI.Desc\_rank=FO.Asc\_rank
WHERE FI.Desc\_rank<=5
ORDER BY FI.Desc\_rank

## Output:

Row	Highest_freightvalue ▼	Lowest_freightvalue ▼
1	RR	SP
2	PB	PR
3	RO	MG
4	AC	RJ
5	PI	DF

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

Ans:

WITH DEL\_Time AS

(SELECT

C.customer\_state,AVG(DATE\_DIFF(O.order\_delivered\_customer\_date,O.order\_purchase\_timestamp,Day))

Avg\_Del\_Time

FROM `Target.Customers` C JOIN `Target.Orders` O ON C.customer\_id=O.customer\_id

GROUP BY C.customer\_state),

Ste\_Rank AS

(SELECT customer\_state, DEL\_Time.Avg\_Del\_Time,

DENSE\_RANK()OVER(ORDER BY DEL\_Time.Avg\_Del\_Time Desc) Desc\_rank,

DENSE\_RANK()OVER(ORDER BY DEL\_Time.Avg\_Del\_Time) Asc\_rank

FROM DEL\_Time)

SELECT SR.customer\_state Highest\_delivery\_time, SL.customer\_state Lowest\_delivery\_time

FROM Ste\_Rank SR JOIN Ste\_Rank SL ON SR.Desc\_rank=SL.Asc\_rank

WHERE SR.Desc\_rank<=5

ORDER BY SR.Desc\_rank

Row	Highest_delivery_time ▼	Lowest_delivery_time ▼
1	RR	SP
2	AP	PR
3	AM	MG
4	AL	DF
5	PA	sc

5.4 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 Fast\_Delv AS

(SELECT C.customer\_state,

 $AVG(DATE\_DIFF(O.order\_estimated\_delivery\_date, O.order\_delivered\_customer\_date, Day)) \ Fast\_Delivered\_customer\_date, Day)) \ Fast\_Delivered\_cus$ 

FROM `Target.Customers` C JOIN `Target.Orders` O ON C.customer\_id=O.customer\_id

WHERE O.order\_delivered\_customer\_date IS NOT NULL

GROUP BY 1),

Fast\_Del\_Rank AS

(SELECT customer\_state, DENSE\_RANK()OVER(ORDER BY Fast\_Del Desc) H\_Rank

FROM Fast\_Delv)

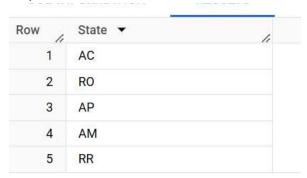
SELECT FDR.customer\_state State

FROM Fast\_Del\_Rank FDR

WHERE H\_Rank<=5

ORDER BY H\_Rank

## Output:



## 6. Analysis based on the payments:

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

#### Ans:

WITH CTE AS

(SELECT DISTINCT O.order\_id, FORMAT\_TIMESTAMP('%Y-%m',order\_purchase\_timestamp) Order\_purchase, P.payment\_type

FROM `Target.Orders` O JOIN `Target.Payments` P ON O.order\_id=P.order\_id

WHERE P.payment\_value>0),

Payment\_type\_count AS

(SELECT Order\_purchase, payment\_type, COUNT(order\_id) No\_of\_orders
FROM CTE
GROUP BY Order\_purchase, payment\_type)
SELECT payment\_type, Order\_purchase, No\_of\_orders,
SUM(No\_of\_orders)OVER(Partition BY payment\_type ORDER BY Order\_purchase) Month\_on\_Month
FROM Payment\_type\_count
ORDER BY 1,2

# Output:

Month_on_Month	No_of_orders ▼	Order_purchase ▼	payment_type ▼	W /
63	63	2016-10	UPI	1
260	197	2017-01	UPI	2
658	398	2017-02	UPI	3
1248	590	2017-03	UPI	4
1744	496	2017-04	UPI	5
2516	772	2017-05	UPI	6
3223	707	2017-06	UPI	7
4068	845	2017-07	UPI	8
5006	938	2017-08	UPI	9
5909	903	2017-09	UPI	10

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

# Ans:

SELECT payment\_installments, COUNT(Distinct order\_id) No\_of\_orders

FROM `Target.Payments`

WHERE payment\_type= 'credit\_card' AND payment\_installments>1 AND payment\_installments>0 GROUP BY 1

HAVING COUNT(order\_id)>=1

Row /	payment_installment	No_of_orders ▼
1	2	12389
2	3	10443
3	4	7088
4	5	5234
5	6	3916
6	7	1623
7	8	4253
8	9	644