

Q1. Business Case: Target SQL:

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.

```
--Data type of all columns in the "customers" table.  
SELECT  
    column_name,  
    data_type  
FROM  
    Target.INFORMATION_SCHEMA.COLUMNS  
WHERE  
    table_name = 'customers';
```

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

Observation/Insights:

- customer_id, customer_unique_id, customer_city and customer_state is of string data type.
- Customer_zip_code_prefix data type is Integer.

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

```
--Get the time range between which the orders were placed.
```

```
select MIN(extract(DATE from order_purchase_timestamp)) FIRST_ORDER_DATE,  
MAX(extract(DATE from order_purchase_timestamp)) LAST_ORDER_DATE  
FROM `Target.orders`;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON
Row	FIRST_ORDER_DATE	LAST_ORDER_DATE		
1	2016-09-04	2018-10-17		

- All orders placed in between 2016-09-04 and 2018-10-17

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

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

```
SELECT count(distinct customer_city) count_city, count(distinct customer_state)
count_state
FROM
`Target.customers` c
join
`Target.orders` o
on c.customer_id=o.customer_id
where o.order_purchase_timestamp between '2016-09-04' and '2018-10-17';
```

Query results

JOB INFORMATION		RESULTS	CHART
Row	count_city	count_state	
1	4119	27	

- All orders have been created from 4119 cities and 27 states.

In-depth Exploration:

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

```
--Is there a growing trend in the no. of orders placed over the past years?
select extract(year from order_purchase_timestamp) as Year, count(order_id)
as orders_placed
from
`Target.orders`
group by extract(year from order_purchase_timestamp)
order by Year;
```

Query results

JOB INFORMATION		RESULTS	CHART
Row	Year	orders_placed	
1	2016	329	
2	2017	45101	
3	2018	54011	

Observations/insights:

- Yes, there is a growing trend in the number of orders placed over past years as per output. Orders placed has been increasing from 2016 to 2018.

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

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

```
select extract(Year from order_purchase_timestamp) Year, Extract(month from
order_purchase_timestamp) Month,
format_datetime('%B', datetime(order_purchase_timestamp)) as
Month_Name, count(order_id) as No_order_placed
from
`Target.orders`
group by extract(Year from order_purchase_timestamp), Extract(month from
order_purchase_timestamp), format_datetime('%B', datetime(order_purchase_times
tamp))
order by Year, Month, Month_Name;
```

Query results



JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION DETAILS	EX
Row	Year	Month	Month_Name	No_order_placed		
1	2016	9	September	4		
2	2016	10	October	324		
3	2016	12	December	1		
4	2017	1	January	800		
5	2017	2	February	1780		
6	2017	3	March	2682		
7	2017	4	April	2404		
8	2017	5	May	3700		
9	2017	6	June	3245		
10	2017	7	July	4026		
11	2017	8	August	4331		

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

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

```
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 as order_time, count(order_id) order_count
from
`Target.orders` o
join
`Target.customers` c
on o.customer_id=c.customer_id
where c.customer_state='DF'
group by
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
order by order_count desc;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON
Row	order_time ▼	order_count ▼		
1	Afternoon	820		
2	Night	614		
3	Morning	608		
4	Dawn	98		

OR

```
select order_time, count(order_id) order_count
from
(select
case
when Extract(hour from o.order_purchase_timestamp) between 0 and 6 then 'Dawn'
when Extract(hour from o.order_purchase_timestamp) between 7 and 12 then 'Morning'
when Extract(hour from o.order_purchase_timestamp) between 13 and 18 then
'Afternoon'
else 'Night'
end as order_time, order_id
from
`Target.orders` o
join
`Target.customers` c
on o.customer_id=c.customer_id
where c.customer_state='DF') table1
group by order_time
order by order_count desc;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON
Row	order_time ▼	order_count ▼		
1	Afternoon	820		
2	Night	614		
3	Morning	608		
4	Dawn	98		

- during Afternoon time Brazilian customers have ordered max orders of 820.

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

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

```
--Get the month on month no. of orders placed in each state.
select
extract(month from o.order_purchase_timestamp) as
month_of_order,c.customer_state,count(o.order_id) order_count
from
`Target.orders` o
join
`Target.customers` c
on o.customer_id=c.customer_id
group by extract(month from o.order_purchase_timestamp),c.customer_state
order by month_of_order,c.customer_state ;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION DET
Row	month_of_order	customer_state		order_count	
1	1	AC		8	
2	1	AL		39	
3	1	AM		12	
4	1	AP		11	
5	1	BA		264	
6	1	CE		99	
7	1	DF		151	
8	1	ES		159	
9	1	GO		164	
10	1	MA		66	
11	1	MG		971	

3.2 How are the customers distributed across all the states?

--How are the customers distributed across all the states?

```
select customer_state,count(customer_id) count_of_customer
from `Target.customers`
group by customer_state
order by count_of_customer desc;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON
Row	customer_state	count_of_customer		
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		
11	PE	1652		

- State SP has highest customers and RJ and MJ has 2nd highest and 3 highest customers.
- 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.

```

1. --1. Get the % increase in the cost of orders from year 2017 to 2018
   (include months between Jan to Aug only).
2. --You can use the "payment_value" column in the payments table to get the
   cost of orders.
3. with cte1 as(
4. select sum(payment_value) cost_of_order_2017
5. from
6. `Target.payments` p
7. join
8. `Target.orders` o
9. on p.order_id=o.order_id
10. where order_purchase_timestamp between '2017-01-01' and '2017-08-31'),
11.
12. cte2 as(
13. select sum(payment_value) cost_of_order_2018
14. from
15. `Target.payments` p
16. join
17. `Target.orders` o
18. on p.order_id=o.order_id
19. where order_purchase_timestamp between '2018-01-01' and '2018-08-31')
20. select ((cte2.cost_of_order_2018-
   cte1.cost_of_order_2017)/cte1.cost_of_order_2017)*100 as
   percentage_increase_in_cost_of_orders
21. from cte1,cte2;

```

Query results		
JOB INFORMATION		RESULTS
Row	percentage_increase	
1	138.5298787105...	

- There is 138.52% increase in cost of orders value in 2017 to 2018

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

```

--Calculate the Total & Average value of order price for each state.
select c.customer_state,sum(i.price) as total_order_price,Avg(i.price) avg_price
from

```



```

`Target.customers` c
join
`Target.orders` o
on c.customer_id=o.customer_id
join
`Target.order_items` i
on o.order_id=i.order_id
group by c.customer_state;

```

Query results

JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION
Row	customer_state	total_order_price	avg_price		
1	RN	83034.97999999...	156.9659357277...		
2	CE	227254.70999999...	153.7582611637...		
3	RS	750304.02000000...	120.3374530874...		
4	SC	520553.34000000...	124.6535775862...		
5	SP	5202955.050001...	109.6536291597...		
6	MG	1585308.029999...	120.7485741488...		
7	BA	511349.99000000...	134.6012082126...		
8	RJ	1824092.669999...	125.1178180945...		

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

```

--Calculate the Total & Average value of order freight for each state.
select c.customer_state,sum(i.freight_value) as
total_freight_price,Avg(i.freight_value) avg_freight_price
from
`Target.customers` c
join
`Target.orders` o
on c.customer_id=o.customer_id
join
`Target.order_items` i
on o.order_id=i.order_id
group by c.customer_state
order by total_freight_price desc;

```

Query results

JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION
Row	customer_state	total_freight_price	avg_freight_price		
1	SP	718723.0699999...	15.14727539041...		
2	RJ	305589.3100000...	20.96092393168...		
3	MG	270853.4600000...	20.63016680630...		
4	RS	135522.7400000...	21.73580433039...		
5	PR	117851.6800000...	20.53165156794...		
6	BA	100156.6799999...	26.36395893656...		
7	SC	89660.26000000...	21.47036877394...		
8	PE	59449.65999999...	32.91786267995...		
9	GO	53114.97999999...	22.76681525932...		
10	DF	50625.49999999...	21.04135494596...		

Insights:

- Customer State SP has highest total freight 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

--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.

```
select
order_id,timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,Day)
time_to_deliver,timestamp_diff(order_delivered_customer_date,order_estimated_delive
ry_date,Day) diff_estimated_delivery_date
from
`Target.orders`
```

```
where order_status='delivered'
order by time_to_deliver desc;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION DETA
Row	order_id	time_to_deliver	diff_estimated_delive		
1	ca07593549f1816d26a572e06...	209	181		
2	1b3190b2dfa9d789e1f14c05b...	208	188		
3	440d0d17af552815d15a9e41a...	195	165		
4	0f4519c5f1c541ddec9f21b3bd...	194	161		
5	285ab9426d6982034523a855f...	194	166		
6	2fb597c2f772eca01b1f5c561b...	194	155		
7	47b40429ed8cce3aee9199792...	191	175		
8	2fe324febf907e3ea3f2aa9650...	189	167		
9	2d7561026d542c8dbd8f0daea...	188	159		
10	437222e3fd1b07396f1d9ba8c...	187	144		
11	c27815f7e3dd0b926b5855262...	187	162		

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

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

```
with cte1 as
(select c.customer_state,avg(i.freight_value) avg_freight_value
from
`Target.customers` c
join
`Target.orders` o
on c.customer_id=o.customer_id
join
`Target.order_items` i
on o.order_id=i.order_id
group by c.customer_state
order by avg_freight_value desc
limit 5),
```

```
cte2 as
(select c.customer_state,avg(i.freight_value) avg_freight_value
from
`Target.customers` c
join
`Target.orders` o
on c.customer_id=o.customer_id
join
`Target.order_items` i
on o.order_id=i.order_id
```

```

group by c.customer_state
order by avg_freight_value asc
limit 5)
select * from cte1
union all
select * from cte2
order by avg_freight_value desc;

```

JOB INFORMATION		RESULTS	CHART	JSON	EX
Row	customer_state	avg_freight_value			
1	RR	42.98442307692...			
2	PB	42.72380398671...			
3	RO	41.06971223021...			
4	AC	40.07336956521...			
5	PI	39.14797047970...			
6	DF	21.04135494596...			
7	RJ	20.96092393168...			
8	MG	20.63016680630...			
9	PR	20.53165156794...			
10	SP	15.14727539041...			

Insights:

- States RR, PB, RO, AC, PI has top 5 highest freight value. Which means it has longer shipped distance.
- States DF, RJ, MG, PR, SP has 5 lowest freight value. Which means it has shorter shipping distance and better logistical networks. We can negotiate with providers for high freight value areas to reduce cost.

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

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

```

with cte1 as(
select
customer_state,avg(timestamp_diff(order_delivered_customer_date,order_purchase_time
stamp,Day)) avg_delivery_time
from
`Target.orders` o
join
`Target.customers` c
on o.customer_id=c.customer_id
where order_status='delivered'
group by customer_state

```

```

order by avg_delivery_time desc
limit 5),
cte2 as(
  select
customer_state,avg(timestamp_diff(order_delivered_customer_date,order_purchase_time
stamp,Day)) avg_delivery_time
from
`Target.orders` o
join
`Target.customers` c
on o.customer_id=c.customer_id
where order_status='delivered'
group by customer_state
order by avg_delivery_time
limit 5
)
select * from cte1
union all
select * from cte2
order by avg_delivery_time desc ;

```

Query results

JOB INFORMATION		RESULTS	CHART	JSON	E
Row	customer_state	avg_delivery_time			
1	RR	28.97560975609...			
2	AP	26.73134328358...			
3	AM	25.98620689655...			
4	AL	24.04030226700...			
5	PA	23.31606765327...			
6	SC	14.47956019171...			
7	DF	12.50913461538...			
8	MG	11.54381329810...			
9	PR	11.52671135486...			
10	SP	8.298061489072...			

Insights:

- RR,AP,AM,AL,PA has worst performing as compared to orders
- SP,PR,MG,DF,SC has best performance in terms of delivery time.

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.

--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.

```
select
customer_state,avg(timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date,day)) as how_fast_delivery
from
`Target.orders` o
join
`Target.customers` c
on o.customer_id=c.customer_id
where order_status='delivered'
group by customer_state
order by how_fast_delivery desc
limit 5;
```

Query results

JOB INFORMATION		RESULTS	CHART	JSON
Row	customer_state	how_fast_delivery		
1	AC	19.762500000000...		
2	RO	19.13168724279...		
3	AP	18.73134328358...		
4	AM	18.60689655172...		
5	RR	16.41463414634...		

Insights:

- Customer state AC, RO, AP, AM, RR has fast order delivery as compared to estimated date of delivery.
- This says this state has better logistic infrastructure, warehouse management, faster processing time.
- Implement this strategy in slow delivery states.

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

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

```
select extract(Year from order_purchase_timestamp) Year,extract(month from
order_purchase_timestamp)
month,format_datetime('%B',datetime(order_purchase_timestamp)) Month_name
,payment_type,count(distinct o.order_id) order_count
from
`Target.orders` o
join
`Target.payments` p
on o.order_id=p.order_id
```

```
group by extract(Year from order_purchase_timestamp),extract(month from
order_purchase_timestamp),format_datetime('%B',datetime(order_purchase_timestamp)),
payment_type
order by Year,month,order_count desc;
```

Query results

[SAVE RESULTS](#)

[EXPLORE I](#)

JOB INFORMATION		RESULTS	CHART	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	Year	month	Month_name	payment_type	order_count	
1	2016	9	September	credit_card	3	
2	2016	10	October	credit_card	253	
3	2016	10	October	UPI	63	
4	2016	10	October	voucher	11	
5	2016	10	October	debit_card	2	
6	2016	12	December	credit_card	1	
7	2017	1	January	credit_card	582	
8	2017	1	January	UPI	197	
9	2017	1	January	voucher	33	
10	2017	1	January	debit_card	9	
11	2017	2	February	credit_card	1347	

Insight:

- Credit Card is most commonly used payment types
- UPI is 2nd mostly used payment types .showing as it's use is growing fastly.

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

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

```
select payment_installments,count(distinct o.order_id) orders_placed
from
`Target.orders` o
join
`Target.payments` p
on o.order_id=p.order_id
group by payment_installments
order by payment_installments;
```

Query results

JOB INFORMATION		RESULTS	CHART
Row	payment_installment	orders_placed	
1	0	2	
2	1	49060	
3	2	12389	
4	3	10443	
5	4	7088	
6	5	5234	
7	6	3916	
8	7	1623	

Load more

INSIGHTS:

- Installment 1 plan is more popular among the customers.
- Majority of orders placed with a single payment installment. It means customer don't prefer to spread their payment into multiple payments.

Recommendation:

- Offer discount to single payment customers to encourage immediate revenue collection