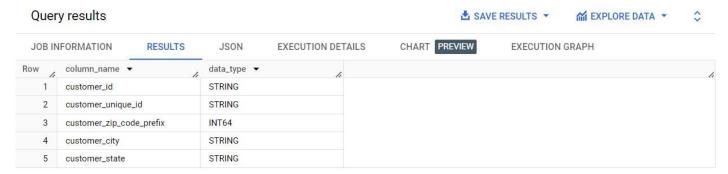
# **BUSINESS CASE ON TARGET**

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.

### CODE:





Insights: we have data types like strings and integers in the "customers table".

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

#### CODE:

select
min(order\_purchase\_timestamp) as first\_order\_placed,
max(order\_purchase\_timestamp) as last\_order\_placed from
`Target\_sql.orders`



Insights: The market has runned or been open for 2 years from the date:2016-09-04 to 2018-10-17 this has been known from the "orders table".

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

#### CODE:

select

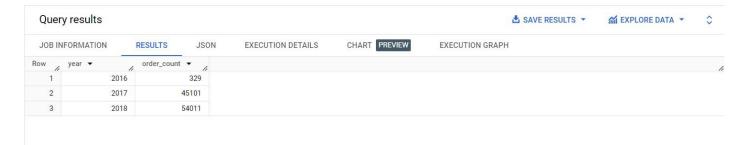
count(DISTINCT(c.customer\_city)) as city, count(DISTINCT(c.customer\_state)) as state from `Target\_sql.customers` c join `Target\_sql.orders` o on c.customer\_id = o.customer\_id where o.order\_purchase\_timestamp between '2016-09-04' and '2018-10-17'



Insights: these are the city and state of the customers

### 2. In-depth Exploration:

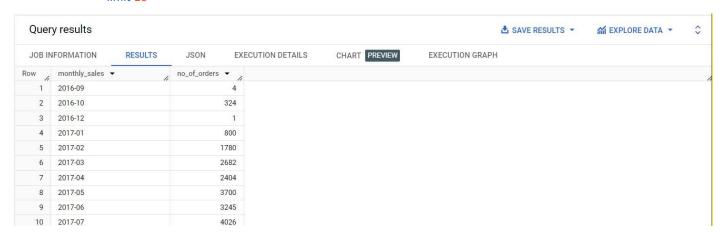
a.ls there a growing trend in the no. of orders placed over the past years? CODE:
select distinct(extract(Year from order\_purchase\_timestamp)) as year, (count (extract(Year
from order\_purchase\_timestamp)))as order\_count from `Target\_sql.orders`
where extract(Year from order\_purchase\_timestamp)
between 2016 and 2018
group by 1 order by 1



Insights:yes , there is a growing trend in the orders over the years

CODE:

```
select format_date('%Y-%m',order_purchase_timestamp)as monthly_sales , count(order_id)
as no_of_orders from `Target_sql.orders`
group by 1
order by 1
limit 10
```



Insights: yes we can find the monthly seasonality in the sales

**ORDER BY** 

order\_count DESC;

c. During 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 CODE: **SELECT CASE** WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) BETWEEN O 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' WHEN EXTRACT(HOUR FROM order\_purchase\_timestamp) BETWEEN 19 AND 23 THEN 'Night' END AS time\_of\_day, COUNT(\*) AS order\_count FROM `Target\_sql.orders` **GROUP BY time of day** 

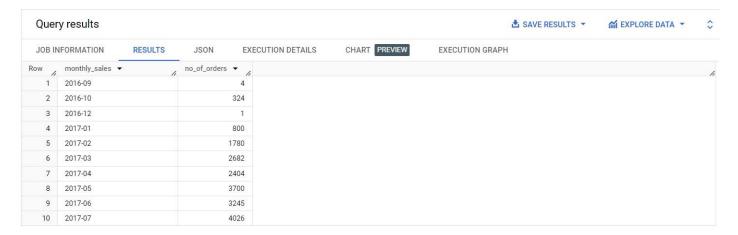


Insights: afternoon orders are top compared to any other and least is dawn orders

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

### CODE:

```
select format_date('%Y-%m',order_purchase_timestamp)as monthly_sales , count(order_id) as no_of_orders from `Target_sql.orders` group by 1 order by 1 limit 10
```

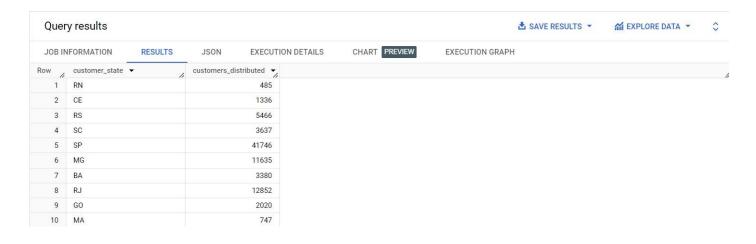


#### Insights:

b. How are the customers distributed across all the states?

CODE:

```
select customer_state,
count(customer_id) as customers_distributed from
`Target_sql.customers` group by customer_state limit 10
```



Insights: customers distribution across the states are found and displayed

### 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).

```
You can use the "payment_value" column in the payments table to get the cost of orders.
with cte_17 as(
select round(sum(p.payment_value),2)as cost_of_orders_17
from (select * from `Target.orders`
where order purchase timestamp between '2017-01-01 00:00:00' and '2017-08-31 23:59:59'
)filtered orders
left join 'Target.payments' as p
on filtered_orders.order_id=p.order_id),
cte_18 as(
select round(sum(p.payment_value),2)as cost_of_orders_18
from (select * from `Target.orders`
where order_purchase_timestamp between '2018-01-01 00:00:00' and '2018-08-31 23:59:59'
)filtered_orders
left join 'Target.payments' as p
on filtered_orders.order_id=p.order_id)
select *,round((((cte_18.cost_of_orders_18-cte_17.cost_of_orders_17)/cte_17.cost_of_orders_17)*100),2) as
cost_incr_percent
from cte_17,cte_18
```

JOB IN	IFORMATION	RESULTS	JSON	EXECUTION [
Row	cost_of_orders_17	cost_of_orde	ers_18	cost_incr_percent
1	3669022.12	86947	33.84	136.98

**Insight:** The cost of orders in 2017 from Jan-Aug is 3669022.12 The cost of orders in 2018 from Jan-Aug is 8694733.84

The % increment in cost from 2017 to 2018 sales is 136.98

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

Ans:

```
select c.customer_state,
round(sum(oi.price),2) as Total_order_price,
round(avg(oi.price),2) as Avg_order_price
from `Target.customers` c
left 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 1
order by 1
```

JOB IN	IFORMATION	RESULTS	JSON EXI	ECUTION DETAILS
Row	customer_state `	•	Total_order_price 🍷	Avg_order_price 🔻
1	AC		15982.95	173.73
2	AL		80314.81	180.89
3	AM		22356.84	135.5
4	AP		13474.3	164.32
5	BA		511349.99	134.6
6	CE		227254.71	153.76
7	DF		302603.94	125.77
8	ES		275037.31	121.91
9	GO		294591.95	126.27
10	MA		119648.22	145.2

Insight: The above results shows the total order price, average order price for each state.

**4.c.** Calculate the Total & Average value of order freight for each state.

```
select c.customer_state,

round(sum(oi.freight_value),2) as Total_order_freight,
round(avg(oi.freight_value),2) as Avg_order_freight
from `Target.customers` c
left 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 1
order by 3 desc,1
```

JOB IN	IFORMATION	RESULTS	JSON EX	ECUTION DETAILS
Row	customer_state   The state   T	,	Total_order_freight	Avg_order_freight >
1	RR	//	2235.19	42.98
2	PB		25719.73	42.72
3	RO		11417.38	41.07
4	AC		3686.75	40.07
5	PI		21218.2	39.15
6	MA		31523.77	38.26
7	то		11732.68	37.25
8	SE		14111.47	36.65
9	AL		15914.59	35.84
10	PA		38699.3	35.83

**Insight: The** results are ordered according the Avg.order. Freight in descending order and it is more with the value 42.98 for the state 'RR'

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

5.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. 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\_estimated\_delivery\_date order\_delivered\_customer\_date

```
select order_id,
order_purchase_timestamp as purchase_date,
order_estimated_delivery_date as estimated_date,
order_delivered_customer_date as delivered_date,
date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_deliver_days,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day)as diff_estimated_delivery
from `Target.orders`
where order_delivered_customer_date is not NULL
order by 2
limit 50
```

JUI	3 INFORMATION RESULTS	JSON EXECU	JTION DETAILS CHA	RT PREVIEW EXECU	TION GRAPH	
Row	order_id ▼	purchase_date ▼	estimated_date ▼	delivered_date ▼	time_to_deliver_days	diff_estimated_deli
1	bfbd0f9bdef84302105ad712db	2016-09-15 12:16:38 UTC	2016-10-04 00:00:00 UTC	2016-11-09 07:47:38 UTC	54	-36
2	3b697a20d9e427646d925679	2016-10-03 09:44:50 UTC	2016-10-27 00:00:00 UTC	2016-10-26 14:02:13 UTC	23	0
3	be5bc2f0da14d8071e2d45451	2016-10-03 16:56:50 UTC	2016-11-07 00:00:00 UTC	2016-10-27 18:19:38 UTC	24	10
4	65d1e226dfaeb8cdc42f66542	2016-10-03 21:01:41 UTC	2016-11-25 00:00:00 UTC	2016-11-08 10:58:34 UTC	35	16
5	a41c8759fbe7aab36ea07e038	2016-10-03 21:13:36 UTC	2016-11-29 00:00:00 UTC	2016-11-03 10:58:07 UTC	30	25
6	d207cc272675637bfed0062ed	2016-10-03 22:06:03 UTC	2016-11-23 00:00:00 UTC	2016-10-31 11:07:42 UTC	27	22
7	cd3b8574c82b42fc8129f6d50	2016-10-03 22:31:31 UTC	2016-11-23 00:00:00 UTC	2016-10-14 16:08:00 UTC	10	39
8	ae8a60e4b03c5a4ba9ca0672c	2016-10-03 22:44:10 UTC	2016-12-01 00:00:00 UTC	2016-11-03 14:04:50 UTC	30	27
9	ef1b29b591d31d57c0d733746	2016-10-03 22:51:30 UTC	2016-11-25 00:00:00 UTC	2016-11-01 15:14:45 UTC	28	23
10	0a0837a5eee9e7a9ce2b1fa83	2016-10-04 09:06:10 UTC	2016-11-24 00:00:00 UTC	2016-10-22 14:51:18 UTC	18	32

**Insight**: The difference between "purchase\_date" and "time\_to\_deliver" is huge and also much difference is exist in between "delivered\_date" and "estimated date".

In the above table for the 1<sup>st</sup> row "diff\_estimated\_delivery" is shown as -36 indicates an order is delivered after 36 days of "estimated\_date"

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

```
with high_avg as(
select c.customer state,
avg(oi.freight_value) as Avg_order_freight,
row_number()over(order by avg(oi.freight_value) desc)as rownum
from `Target.customers` c
left 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 1),
low avg as(
 select c.customer_state,
avg(oi.freight_value) as Avg_order_freight,
row number()over(order by avg(oi.freight value))as rownum
from `Target.customers` c
left 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 1)
select h.customer state, h.Avg order freight as High avg order freight,
     l.customer_state,l.Avg_order_freight as Low_avg_order_freight
from high_avg h
join low avg I
on h.rownum=l.rownum
limit 5
```

### Query results

JOB IN	JOB INFORMATION		ESULTS	JSON	EXECUTION DETAILS	CHART PREVIEW
Row	customer_state	<b>▼</b> //	High_avg_c	order_freight 🤟	customer_state_1 ▼	Low_avg_order_freight
1	RR		42.9844	23076923093	SP	15.147275390419248
2	PB		42.7238	03986710941	PR	20.531651567944248
3	RO		41.0697	12230215842	MG	20.630166806306541
4	AC		40.0733	69565217405	RJ	20.96092393168248
5	PI		39.1479	70479704767	DF	21.041354945968383

**Insight:** The above results indicates the top 5 states with highest avg.order.freight('RR,PB,RO,AC,PI) and other 5 states with lowest avg.order.freight('SP,PR,MG,RJ,DF)

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

```
WITH High_delivery as(
select c.customer_state,
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)) as avg_delivery_time,
row_number()over(order by avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day))desc)as rownum
from `Target.customers` c
join `Target.orders` o
on c.customer_id=o.customer_id
group by 1),
```

```
low_delivery as(
    select c.customer_state,
    avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)) as avg_delivery_time,
    row_number()over(order by avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)))as rownum
    from `Target.customers` c
    join `Target.orders` o
    on c.customer_id=o.customer_id
    group by 1)

select h.customer_state,h.avg_delivery_time as High_avg_delvry_time,
    l.customer_state,l.avg_delivery_time as Low_avg_delvry_time

from High_delivery h
    join low_delivery I
    on h.rownum=l.rownum

limit 5
```

JOB IN	NFORMATION	RESULTS	JSON	EXECUTION DETAILS	CHART PREVIEW
Row	customer_state ▼	High_av	g_delvry_time 🤟	customer_state_1 ▼	Low_avg_delvry_time
1	RR	28.975	609756097562	SP	8.2980614890725874
2	AP	26.731	343283582085	PR	11.526711354864908
3	AM	25.986	206896551728	MG	11.543813298106569
4	AL	24.040	302267002513	DF	12.509134615384616
5	PA	23.316	067653276981	SC	14.479560191711331

**Insight:** The above results indicates the top 5 states with highest avg.delivery. time('RR,AP,AM,AL,PA) and other 5 states with lowest avg. delivery. time ('SP,PR,MG,DF,SC)

5.D 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.

```
with cte as(
select c.customer_state,
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)) as avg_actual_delivery_days,
avg(date_diff(order_estimated_delivery_date,order_purchase_timestamp,day))as avg_estimated_delivery_days
from `Target.customers` c
left join `Target.orders` o
on c.customer_id=o.customer_id
where order_delivered_customer_date is not NULL
group by 1)

select c.customer_state,(c.avg_estimated_delivery_days-c.avg_actual_delivery_days)as fast_delivery
from cte c
order by 2
limit 5
```

JOB IN	IFORMATION	RESULTS	JSON EX
Row	customer_state	<b>~</b>	fast_delivery ▼
1	AL		8.168765743073
2	MA		8.966527196652
3	SE		9.453731343283
4	ES		9.888220551378
5	CE		10.18686473807

**Insight:** The above table shows the states with the fastest delivery are AL,MA,SE,ES,CE.

### 6. Analysis based on the payments:

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

```
select
format_date('%Y-%m',order_purchase_timestamp) as Monthly_sales,
p.payment_type,
count(p.order_id) as order_count
from `Target.orders` o
join `Target.payments` p
on o.order_id=p.order_id
group by 1,2
order by 1,2
```

## Query results

JOB IN	IFORMATION	RESULTS	JSON	EXECUTION DET	TAILS	CHART
Row	Monthly_sales ▼	h	payment_type	• //	order_count	<b>-</b>
1	2016-09		credit_card			3
2	2016-10		UPI			63
3	2016-10		credit_card			254
4	2016-10		debit_card			2
5	2016-10		voucher			23
6	2016-12		credit_card			1
7	2017-01		UPI			197
8	2017-01		credit_card			583
9	2017-01		debit_card			9
10	2017-01		voucher			61

**Insight:** By observing the above results the Brazilian people uses "credit card payment" more.

**Recommendation:** Suggest that providing of credit cards to customers may increase in sales.

6.b.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 order\_count
from `Target.payments`
where payment\_installments>=1 and payment\_sequential>=1
group by 1
order by 1

JOB IN	NFORMATION	RESUL	JSON
Row	payment_installment	s 🕶	order_count ▼
1		1	52546
2		2	12413
3		3	10461
4		4	7098
5		5	5239
6		6	3920
7		7	1626
8		8	4268
9		9	644
10		10	5328

**Insight:** We can see many customers are choosing EMI options.

**Recommendation:** Providing interest free or low interest on EMI may increase sales.