Name: Raghu **Date: 17/04/2023**

Business case Study- Target

1. # Time period for which the data is given

select min(order_purchase_timestamp) as first_order_time, max(order_purchase_timestamp) as last_order_time from `Target_sql.orders`

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

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

Cities and States of customers ordered during the given period

select distinct customer_state, customer_city from `Target_sql.customers`

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

Row	customer_state	customer_city
1	RN	acu
2	CE	ico
3	RS	ipe
4	CE	ipu
5	SC	ita
6	SP	itu
7	SP	jau
8	MG	luz
9	SP	poa
10	MG	uba

2. # In-depth Exploration:

Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months

select * from (select extract(month from order_purchase_timestamp) as Month, count(order_id) as Total_orders
from `Target_sql.orders` group by extract(month from order_purchase_timestamp)) TAB1
order by Month

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

Row	Month	Total_orders
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959
11	11	7544
12	12	5674

 # Growing trend: Total number of orders per month in 2018

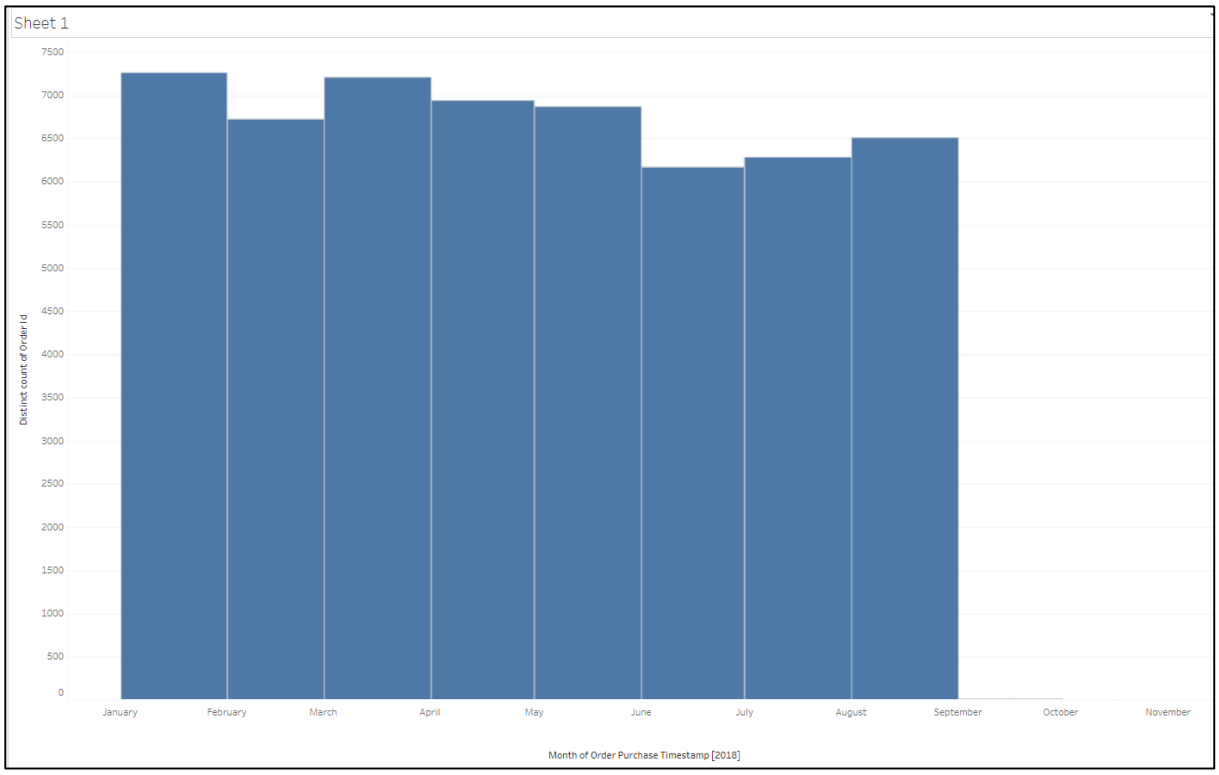
```
select Month, count(order_id) as Total_No_orders from
(select extract(month from order_purchase_timestamp) as Month,extract(year from
order_purchase_timestamp) year,
order_id
from `Target_sql.orders`) tab1
where year = 2018
group by Month
order by Month
```

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

Row	Month	Total_No_orders
1	1	7269
2	2	6728
3	3	7211
4	4	6939
5	5	6873
6	6	6167
7	7	6292
8	8	6512
9	9	16
10	10	4

** From the above two tables, it is clear that, in the month of September (9) and October (10), people tend to buy less.
And people tend to buy more during the period between March and August
So company can increase number of sales by giving discounts and offering large variety of products during the period between March & August. And people tend to buy less in the month ofSeptember & October, company needs to find reason for this lower number of orders. If not

possible to increase the number of orders, company can save storage cost, maintenance cost and shipping cost etc.
Below graph shows Number of orders for each month in 2018



Time do Brazilian customers tend to buy (Checking for 2017, January)

```
select Time_period, count(order_id) as Number_of_orders from
(select extract(hour from order_purchase_timestamp) as Hours,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_period
from `Target_sql.orders`
where extract(month from order_purchase_timestamp)= 1 and extract(year from
order_purchase_timestamp)= 2017)tab1
group by Time_period order by Number_of_orders
```

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

Row	Time_period	Number_of_orde
1	Dawn	45
2	Morning	179
3	Night	231
4	Afternoon	345

** From the above result table, people tend to buy afternoon more and tend to buy less in the morning

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

Month on month orders by states

```
with info_state_wise_orders as (select order_id, extract(month from order_purchase_timestamp) as Month, customer_state from `Target_sql.orders` o join `Target_sql.customers` c on o.customer_id = c.customer_id)

select customer_state, Month, count(order_id) as Total_orders from info_state_wise_orders group by  customer_state, Month order by Total_orders desc
```

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

Row	customer_state	Month	Total_orders
1	SP	8	4982
2	SP	5	4632
3	SP	7	4381
4	SP	6	4104
5	SP	3	4047
6	SP	4	3967
7	SP	2	3357
8	SP	1	3351
9	SP	11	3012
10	SP	12	2357

Month on month orders by states for 2018

```
with info_state_wise_orders_2018 as (select order_id, extract(month from order_purchase_timestamp) as Month, extract(year from order_purchase_timestamp) as Year,customer_state from `Target_sql.orders` o join `Target_sql.customers` c on o.customer_id = c.customer_id)
select customer_state, Month, count(order_id) as Total_orders from info_state_wise_orders_2018 where Year = 2018 group by customer_state, Month order by Total_orders desc
```

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

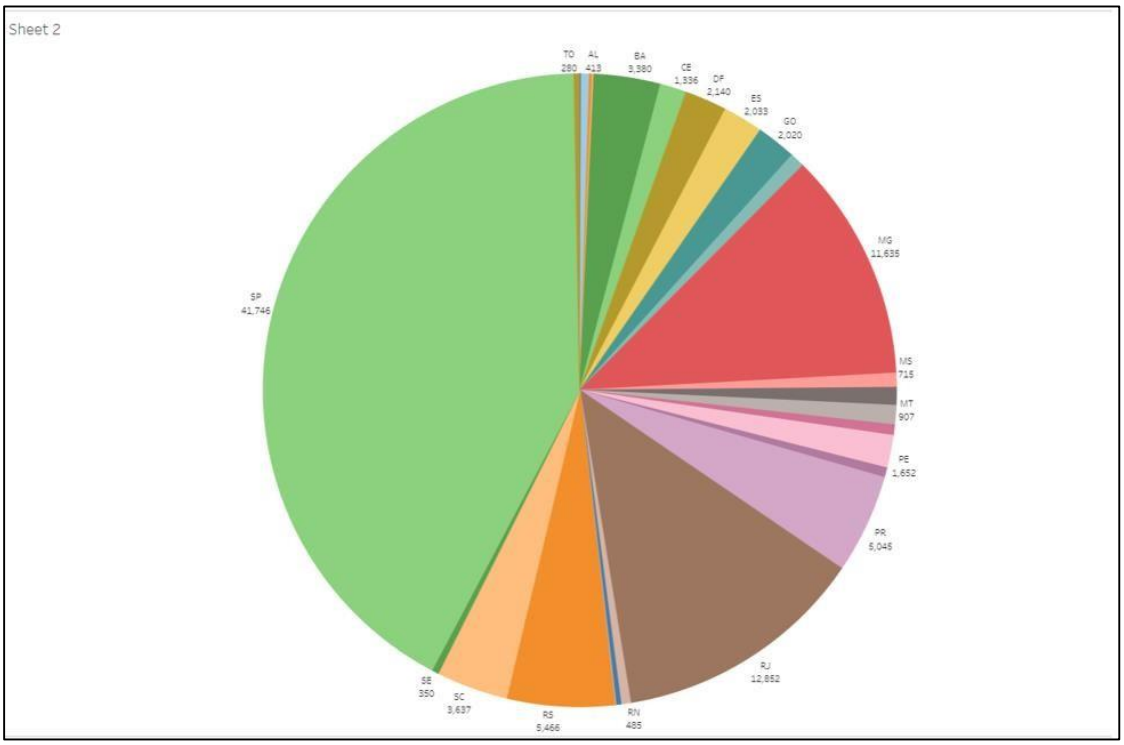
Row	customer_state	Month	Total_orders
1	SP	8	3253
2	SP	5	3207
3	SP	4	3059
4	SP	1	3052
5	SP	3	3037
6	SP	7	2777
7	SP	6	2773
8	SP	2	2703
9	RJ	2	922
10	RJ	3	907

Distribution of customers across the states in Brazil

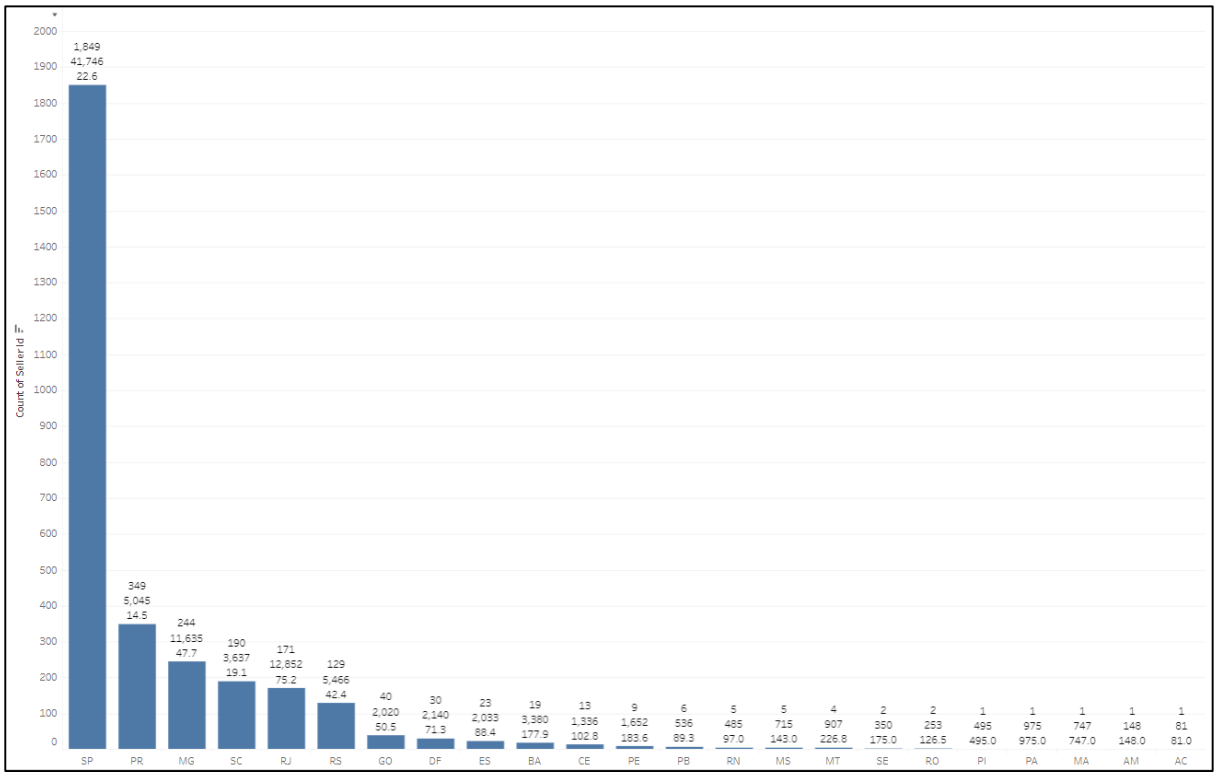
select customer_state, count(customer_id) as Number_of_customers from `Target_sql.customers`
group by customer_state
order by customer_state

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

Row	customer_state	Number_of_cust
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033
9	GO	2020
10	MA	747



** From the above result table & pie chart, state SP has more than 40% customers



** Even though 40% of the customers are from the state SP, (Number of customers)/(Number of seller_ids) is 22 i.e.., customer per seller is 22. But for the state PA, even though % of customers is very less, customer per seller is 975.

State PR has the lowest customer per seller value

So company wants one more seller, it is good to have it in the state which has high customer per seller i.e..,PA. States AP & RR are the two which has lowest number of customers, so company should focus on these cities to increase their market if these two cities have more number of potential customers.

4. Impact of economy

 # % increase in cost of orders from 2017 to 2018 (include months between Jan to Augonly)

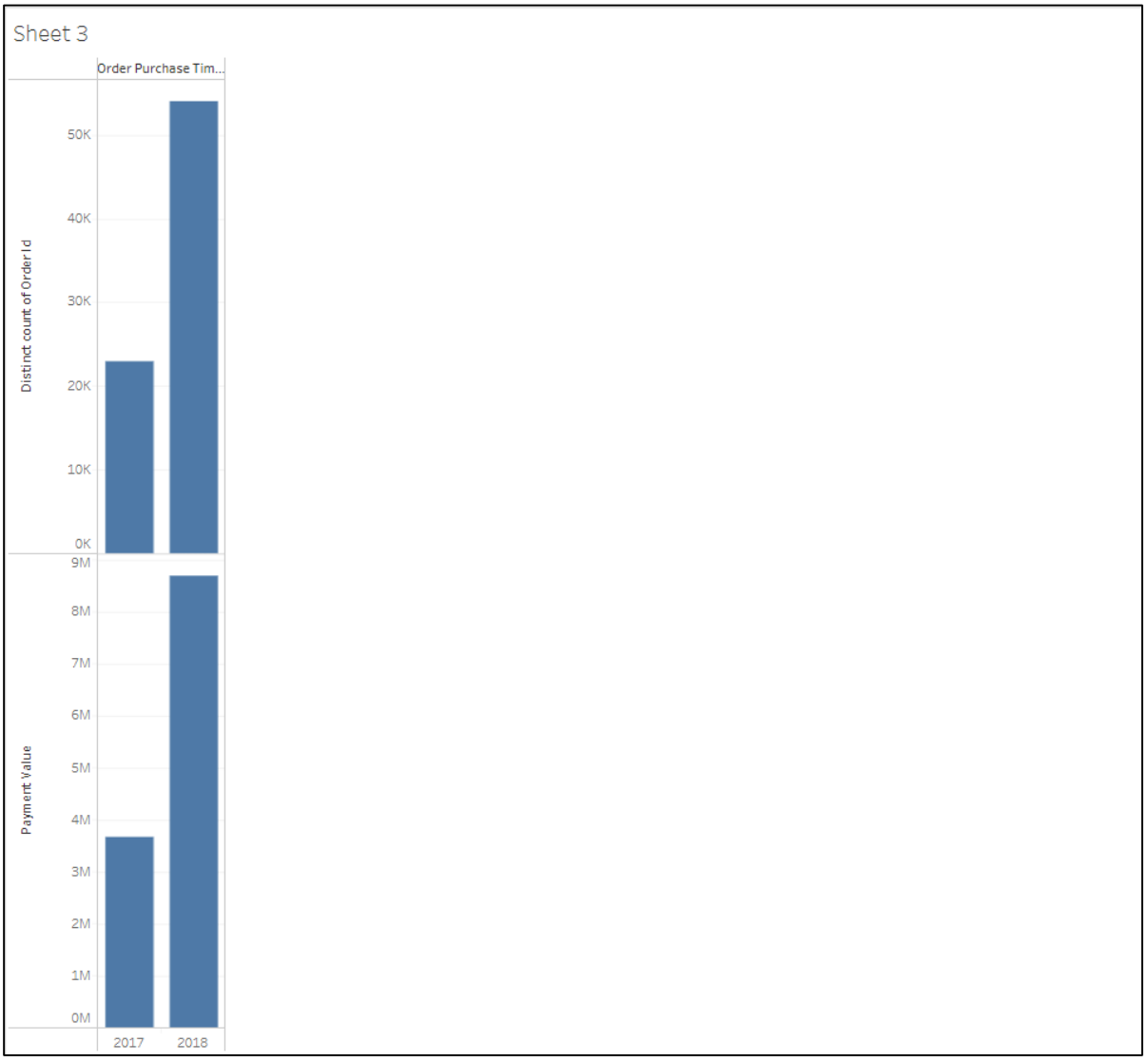
```
create view Target_sql.Order_Payment_info as (select o.order_id, extract(year from
order_purchase_timestamp) year, extract(month from order_purchase_timestamp) as month,
payment_value from `Target_sql.orders` o join `Target_sql.payments` p
on o.order_id = p.order_id);

select round((sum(case when year = 2018 and month between 1 and 8 then payment_value else 0
end) –
sum(case when year = 2017 and month between 1 and 8 then payment_value else 0
end))*100/(sum(case when year = 2018 and month between 1 and 8 then payment_value else 0
end)),2)  percentage_incr_cost
from `Target_sql.Order_Payment_info`
```

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

Row	percentage_incr_
1	57.8

** Around total 58% of total cost of order increased from 2017 to 2018. It might be due to increasein number of orders from 2017 to 2018 or increase in cost of orders, increase in freight value or may be combination of all

** From the below graph, it is clear that % of increase in payment value from 2017 to 2018 (January to August) is same as % of increase in number of orders from 2017 to 2018 (January to August). So due to increase in number of orders from 2017 to 2018, payment value also increasedfrom 2017 to 2018.

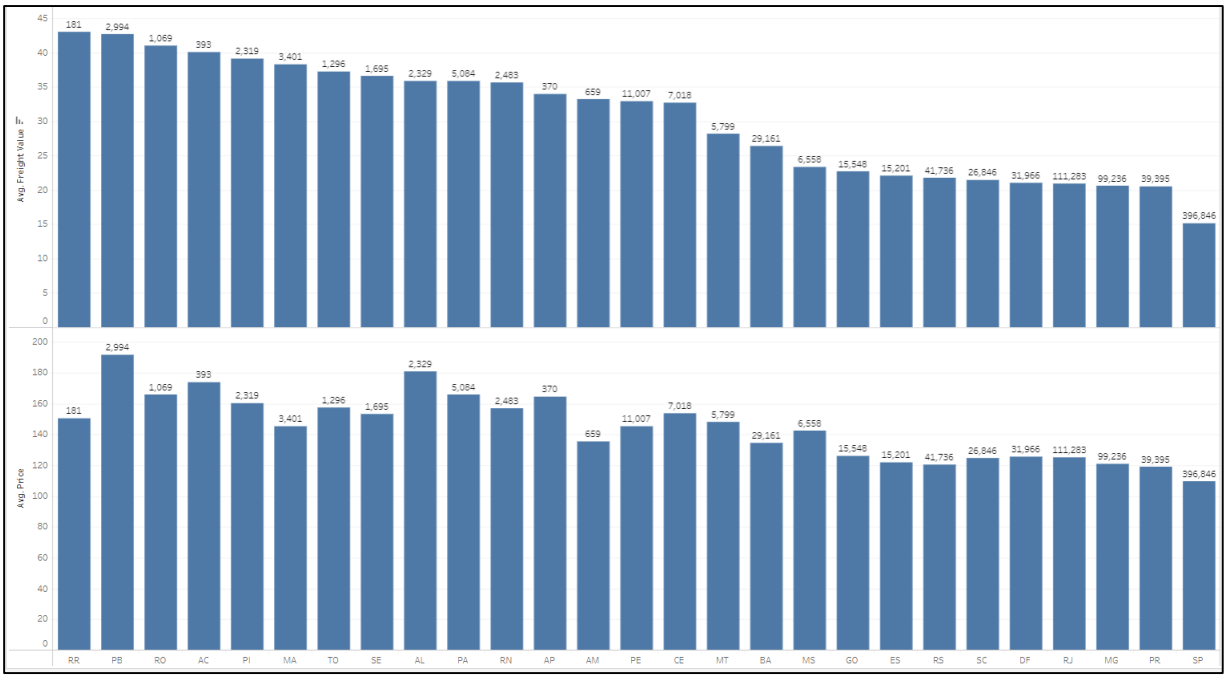


Mean & Sum of price and freight value by customer state

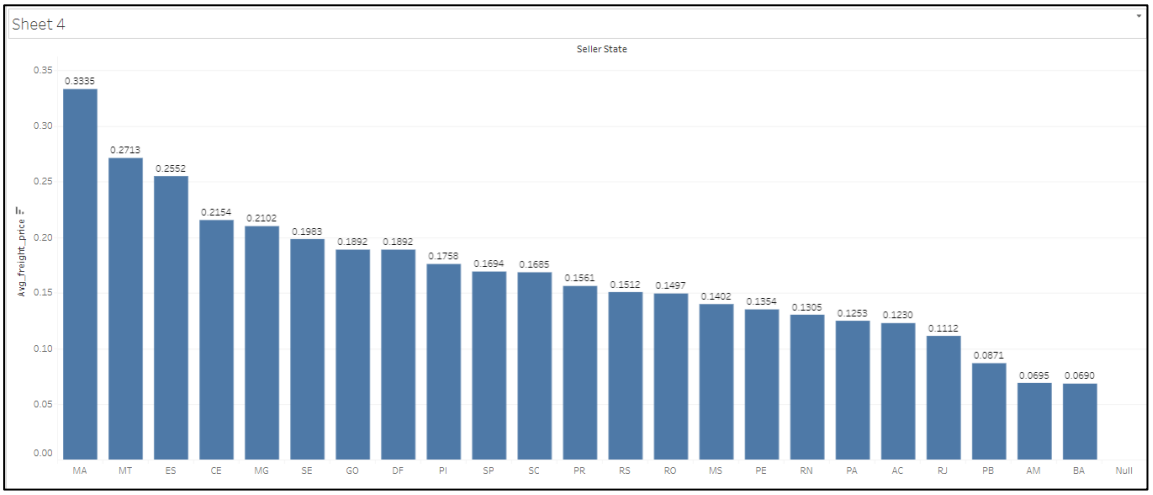
```
with Mean_Total_cost_per_state as (select * from `Target_sql.order_items` oi
join `Target_sql.orders` o on oi.order_id = o.order_id join `Target_sql.customers` c on
o.customer_id = c.customer_id)
select customer_state, avg(price) as Avg_price, sum(price) as Total_price, avg(freight_value) as
Avg_freight_value,
sum(freight_value) as Total_freight_value from Mean_Total_cost_per_state group by
customer_state
```

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

Row	customer_state	Avg_price	Total_price	Avg_freight_valu	Total_freight_val
1	MT	148.297184...	156453.529...	28.1662843...	29715.4300...
2	MA	145.204150...	119648.219...	38.2570024...	31523.7700...
3	AL	180.889211...	80314.81	35.8436711...	15914.5899...
4	SP	109.653629...	5202955.05...	15.1472753...	718723.069...
5	MG	120.748574...	1585308.02...	20.6301668...	270853.460...
6	PE	145.508322...	262788.029...	32.9178626...	59449.6599...
7	RJ	125.117818...	1824092.66...	20.9609239...	305589.310...
8	DF	125.770548...	302603.939...	21.0413549...	50625.4999...
9	RS	120.337453...	750304.020...	21.7358043...	135522.740...
10	SE	153.041168...	58920.8500...	36.6531688...	14111.4699...



** From the above results table & graph, Average freight value for the state RR is high (sum (price)/sum(freight value) is low)



** From the above graph, highest ratio between average freight value and average price is 0.33 for the state MA. i.e.., on an average, around 33 % of actual price is spent for freight.

5. # Analysis on sales, freight and delivery time

days between purchasing, delivering and estimated delivery

```
select date_diff(order_delivered_customer_date, order_purchase_timestamp, day)
days_btw_del_purch,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as
days_btw_est_del from `Target_sql.orders`
```

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

Row	days_btw_del_pu	days_btw_est_de
1	30	-12
2	30	28
3	35	16
4	30	1
5	32	0
6	29	1
7	43	-4
8	40	-4
9	37	-1
10	33	-5

time_to_delivery & diff_estimated_delivery (in hours:minutes:seconds)

```
create view Target_sql.Analysis as (select customer_state as state, price,
freight_value,(order_purchase_timestamp - order_delivered_customer_date) time_to_delivery,
(order_estimated_delivery_date - order_delivered_customer_date) diff_estimated_delivery
from `Target_sql.orders` o join `Target_sql.order_items` oi on o.order_id = oi.order_id  join
`Target_sql.customers` c on c.customer_id = o.customer_id);

select time_to_delivery, diff_estimated_delivery from `Target_sql.Analysis`
where time_to_delivery is not null
```

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

Row	time_to_delivery	diff_estimated_delivery
1	0-0 0 -168:14:41	0-0 0 1088:52:49
2	0-0 0 -722:14:59	0-0 0 -310:3:51
3	0-0 0 -743:13:54	0-0 0 681:6:10
4	0-0 0 -181:40:7	0-0 0 1065:23:1
5	0-0 0 -181:40:7	0-0 0 1065:23:1
6	0-0 0 -262:29:53	0-0 0 989:12:17
7	0-0 0 -853:56:53	0-0 0 397:1:26
8	0-0 0 -565:3:54	0-0 0 228:49:34
9	0-0 0 -311:9:0	0-0 0 -133:12:27
10	0-0 0 -309:37:20	0-0 0 298:32:10

select state, avg(time_to_delivery) as Avg_time_to_delivery,avg(diff_estimated_delivery) as Avg_diff_estimated_delivery, avg(freight_value) as Avg_freight_value from `Target_sql.Analysis` group by state

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

Row	state	Avg_time_to_delivery	Avg_diff_estimated_delivery	Avg_freight_valu
1	MT	0-0 0 -431:4:49.308582449	0-0 0 333:30:17.274831243	28.1662843...
2	MA	0-0 0 -519:34:4.800	0-0 0 221:24:4.645	38.2570024...
3	AL	0-0 0 -587:44:21.852459016	0-0 0 193:22:34.871194379	35.8436711...
4	SP	0-0 0 -209:22:15.899683482	0-0 0 252:19:20.364812781	15.1472753...
5	MG	0-0 0 -287:36:49.457072075	0-0 0 303:20:44.706355965	20.6301668...
6	PE	0-0 0 -438:42:8.667239404	0-0 0 306:20:47.015463917	32.9178626...
7	RJ	0-0 0 -363:33:47.561218719	0-0 0 271:24:6.523540223	20.9609239...
8	DF	0-0 0 -311:0:57.005944798	0-0 0 275:49:59.438641188	21.0413549...
9	RS	0-0 0 -364:31:32.063916517	0-0 0 322:22:7.941790314	21.7358043...
10	SE	0-0 0 -515:12:59.317333333	0-0 0 223:49:3.408	36.6531688...

Top 5 states with highest freight value

select state, avg(freight_value) as Avg_freight_value from `Target_sql.Analysis` group by state order by Avg_freight_value desc limit 5

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

Row	state	Avg_freight_valu
1	RR	42.9844230...
2	PB	42.7238039...
3	RO	41.0697122...
4	AC	40.0733695...
5	PI	39.1479704...

** Above results table shows top 5 states with highest freight value. Freight value depends on mode of transportation, distance, points of pickup and delivery. So company can reduce the freight vale by picking shortest distance if possible, increasing number of orders, changing points ofpickup & delivery, considering cost efficient mode of transportation

Top 5 states with lowest freight value

select state, avg(freight_value) as Avg_freight_value from `Target_sql.Analysis` group by state order by Avg_freight_value asc limit 5

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

Row	state	Avg_freight_valu
1	SP	15.1472753...
2	PR	20.5316515...
3	MG	20.6301668...
4	RJ	20.9609239...
5	DF	21.0413549...

Top 5 states with highest average time to delivery

create view Target_sql.avg_time_to_del as(select state, avg(time_to_delivery) as Avg_time_to_delivery from `Target_sql.Analysis` group by state);

select state from `Target_sql.avg_time_to_del`
order by Avg_time_to_delivery desc limit 5

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

Row	state
1	SP
2	PR
3	MG
4	DF
5	SC

Top 5 lowest average time to delivery

select state from `Target_sql.avg_time_to_del`
order by Avg_time_to_delivery asc limit 5

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

Row	state
1	RR
2	AP
3	AM
4	AL
5	PA

** Above results table shows that states having lowest average time to delivery. Lowest avg time to delivery may be due to delivery agents or due to heavy traffic in that particular state etc.

Company needs to consider this factor in order to make fastest delivery and to give better customer experience by removing agents who are not enough good at their work or by motivating them with incentive

Top 5 states where delivery is really fast

```
create view Target_sql.avg_diff_est_del_time as(select state, avg(diff_estimated_delivery) as
Avg_diff_est_del from `Target_sql.Analysis`
group by state);

select state from `Target_sql.avg_diff_est_del_time`
order by Avg_diff_est_del asc limit 5
```

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

Row	state
1	AL
2	MA
3	SE
4	ES
5	BA

6. Payment type analysis:

Month over Month count of orders for different payment types

```
Create view Target_sql.order_payment_info as (
select o.order_id, extract(month from order_purchase_timestamp) as month,payment_type,
payment_installments from `Target_sql.orders` o join `Target_sql.payments`p on o.order_id =
p.order_id);

select month, payment_type, count(order_id) as Total_orders
from `Target_sql.order_payment_info`
group by month, payment_type
order by month
```

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

Row	month	payment_type	Total_orders
1	1	credit_card	6103
2	1	UPI	1715
3	1	voucher	477
4	1	debit_card	118
5	2	UPI	1723
6	2	credit_card	6609
7	2	voucher	424
8	2	debit_card	82
9	3	credit_card	7707
10	3	UPI	1942

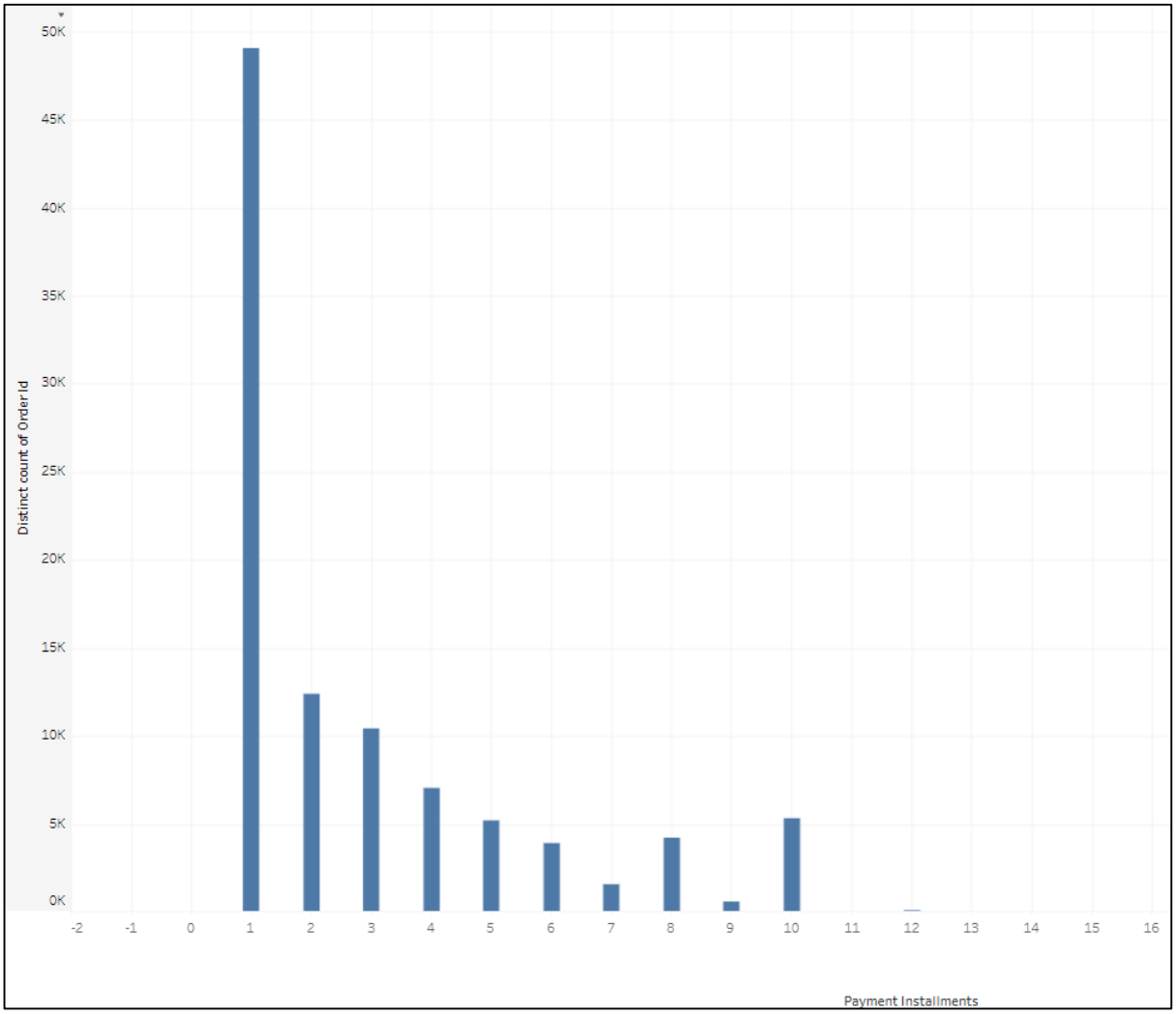
** From the above table results & graph (Graph for only year 2018), payment type for the most of the orders is 'credit_card' followed by 'UPI'

Count of orders based on the no. of payment instalments

```
select payment_installments, count(order_id) as Total_No_orders
from `Target_sql.order_payment_info`
group by payment_installments
order by payment_installments
```

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

Row	payment_installr	Total_No_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644



** From the above results table, customers buy many orders through 1 instalments plan followed by 2 & 3 instalments plan. Customers may not be ready to go for higher instalments plans due to extra charges. Company can check whether the number of orders & customers can be increased by reducing extra charges for high instalments plans.