Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/

Target SQL

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

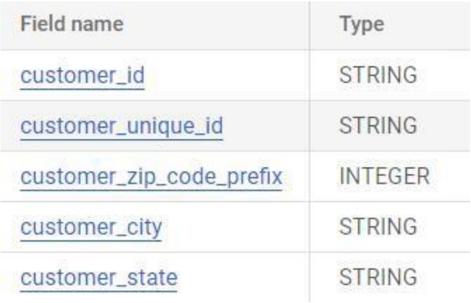
1. **Data types of column in a table :**

There are totally 8 tables available in the given context and each table is related in the following way

Customers and orders are related through customer id
Orders and payments are related through order id
Orders and ordered items are related through order id
Orders and reviewrs are realted to through order id
Orders and products are related through order item table through product id
Order items and sellers are realted through seller id
Customers and sellers are related to geolocation through zipcode prefix

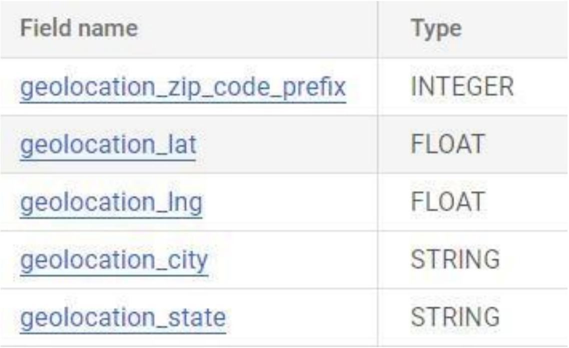
Below mentioned are the data types of the each table

Customers.csv



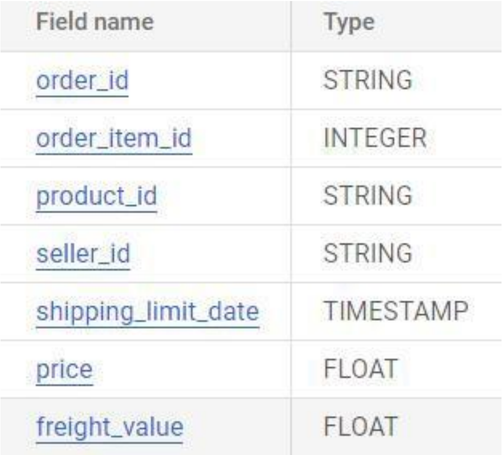
Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

Geolocation.csv



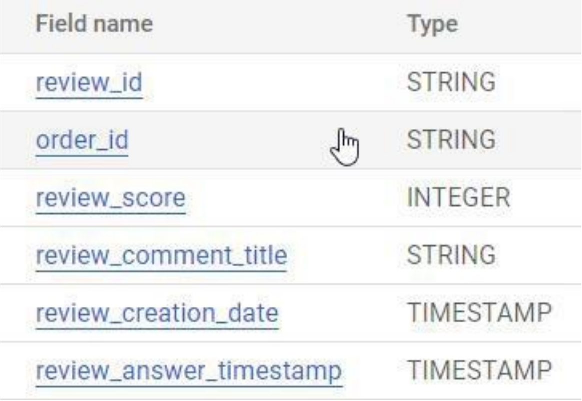
Field name	Type
geolocation_zip_code_prefix	INTEGER
geolocation_lat	FLOAT
geolocation_lng	FLOAT
geolocation_city	STRING
geolocation_state	STRING

Order_items.csv



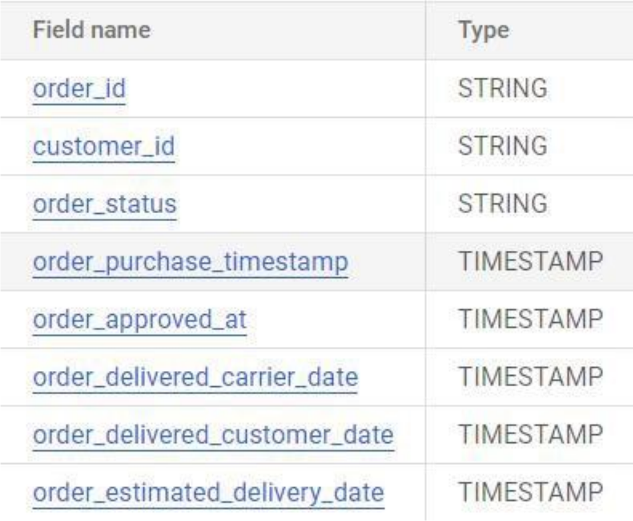
Field name	Type
order_id	STRING
order_item_id	INTEGER
product_id	STRING
seller_id	STRING
shipping_limit_date	TIMESTAMP
price	FLOAT
freight_value	FLOAT

Order_reviews.csv



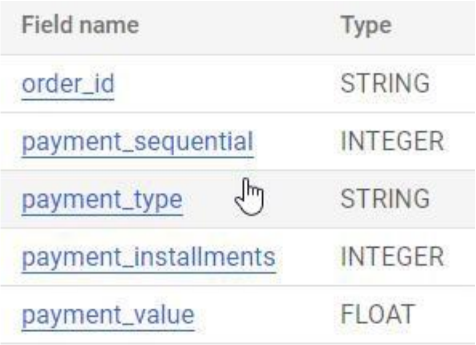
Field name	Type
review_id	STRING
order_id	STRING
review_score	INTEGER
review_comment_title	STRING
review_creation_date	TIMESTAMP
review_answer_timestamp	TIMESTAMP

Orders.csv



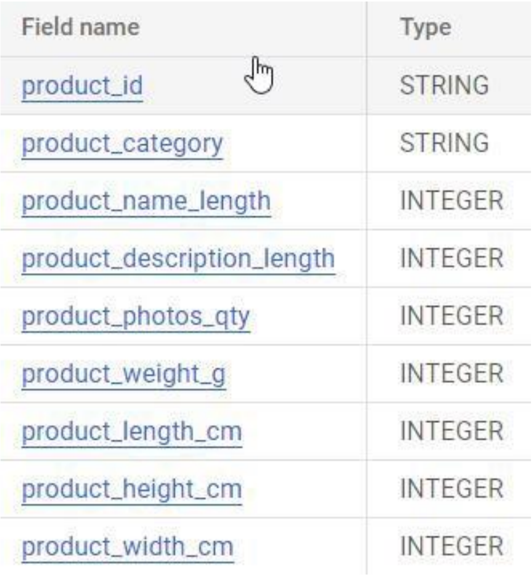
Field name	Type
order_id	STRING
customer_id	STRING
order_status	STRING
order_purchase_timestamp	TIMESTAMP
order_approved_at	TIMESTAMP
order_delivered_carrier_date	TIMESTAMP
order_delivered_customer_date	TIMESTAMP
order_estimated_delivery_date	TIMESTAMP

Payments.csv



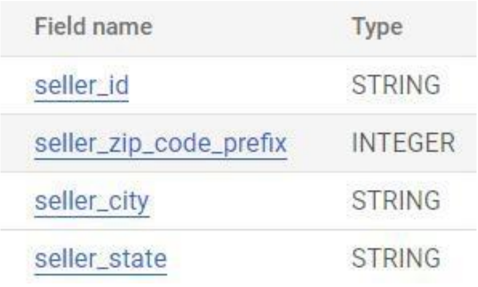
Field name	Type
order_id	STRING
payment_sequential	INTEGER
payment_type	STRING
payment_installments	INTEGER
payment_value	FLOAT

Products.csv



Field name	Type
product_id	STRING
product_category	STRING
product_name_length	INTEGER
product_description_length	INTEGER
product_photos_qty	INTEGER
product_weight_g	INTEGER
product_length_cm	INTEGER
product_height_cm	INTEGER
product_width_cm	INTEGER

Sellers.csv



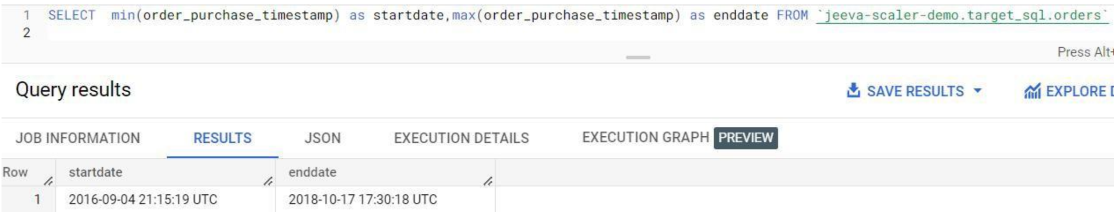
Field name	Type
seller_id	STRING
seller_zip_code_prefix	INTEGER
seller_city	STRING
seller_state	STRING

2. Time period for which data is given

Time period of data is from 04-Sep-2016 to 17-Oct-2018

Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/

Select min(order_purchase_timestamp) as startdate,max(order_purchase_timestamp) as enddate from ‘jeeva-scaler-demo.target_sql.orders’



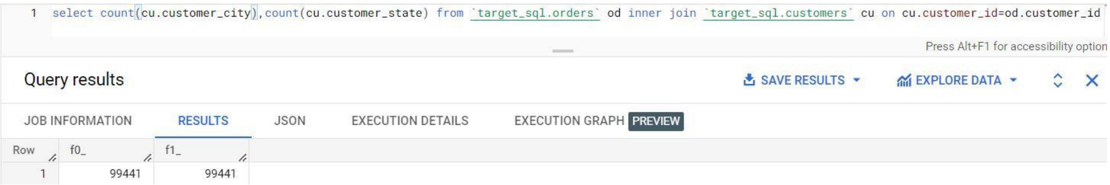
```
SELECT  min(order_purchase_timestamp) as startdate,max(order_purchase_timestamp) as enddate FROM `jeeva-scaler-demo.target_sql.orders`
```

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

3. Cities and States of customers ordered during the given period

In Total there are 99441 cities and states record corresponding to customers who ordered during the time period.There are totally 4119 unique cities across 27 unique states.

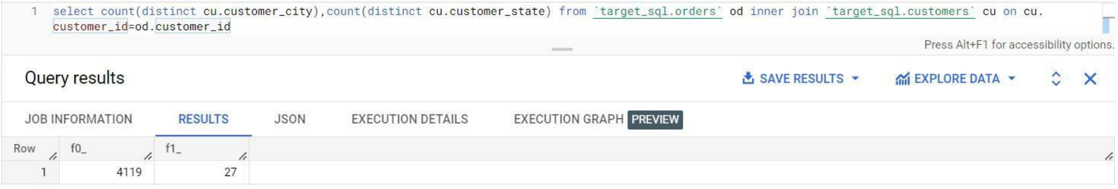
total count of cities and states



```
select count(cu.customer_city),count(cu.customer_state) from `target_sql.orders` od inner join `target_sql.customers` cu on cu.customer_id=od.customer_id
```

Row	f0_	f1_
1	99441	99441

Cleaned up data / refined data of cities and states

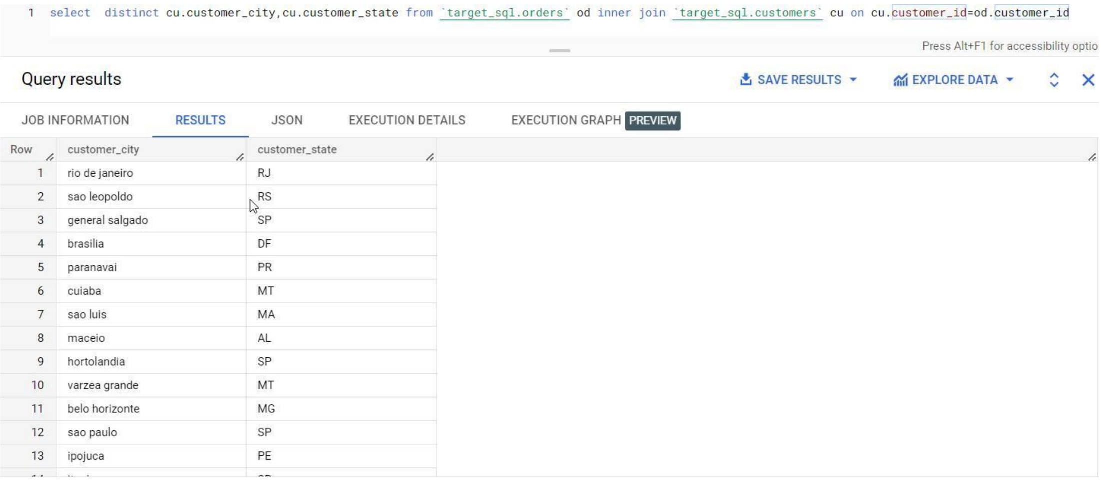


```
select count(distinct cu.customer_city),count(distinct cu.customer_state) from `target_sql.orders` od inner join `target_sql.customers` cu on cu.
customer_id=od.customer_id
```

Row	f0_	f1_
1	4119	27

Overall list of cities and states for the orders during the given time period

Select distinct cu.customer_city,cu.customer_state from ‘target.sql_orders’ od inner join target_sql.customers cu on cu.customer_id=od.customer_id



```
select  distinct cu.customer_city,cu.customer_state from `target_sql.orders` od inner join `target_sql.customers` cu on cu.customer_id=od.customer_id
```

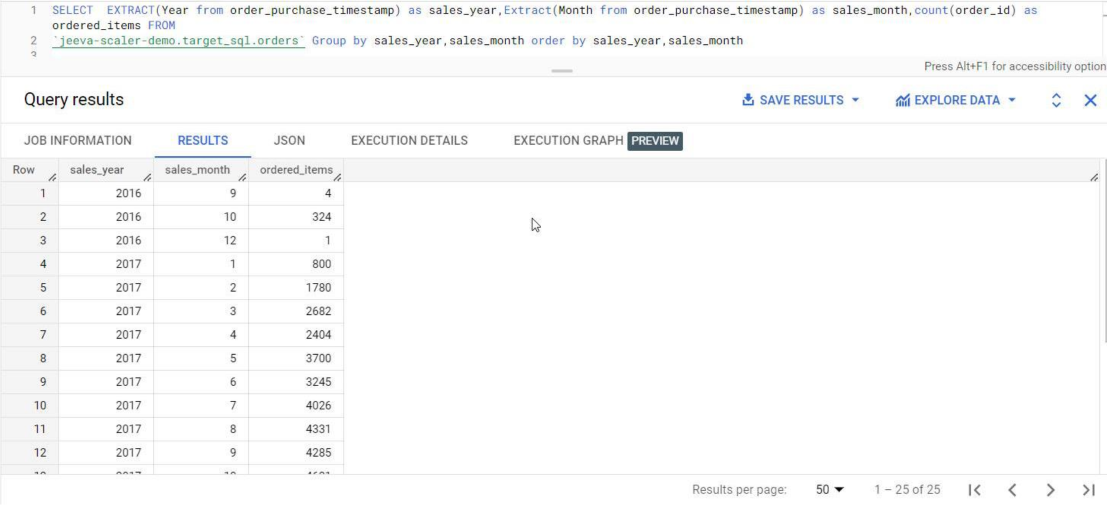
Row	customer_city	customer_state
1	rio de janeiro	RJ
2	sao leopoldo	RS
3	general salgado	SP
4	brasilia	DF
5	paranavai	PR
6	cuiaba	MT
7	sao luis	MA
8	maceio	AL
9	hortolandia	SP
10	varzea grande	MT
11	belo horizonte	MG
12	sao paulo	SP
13	ipojuca	PE

Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/

2. In-depth Exploration:

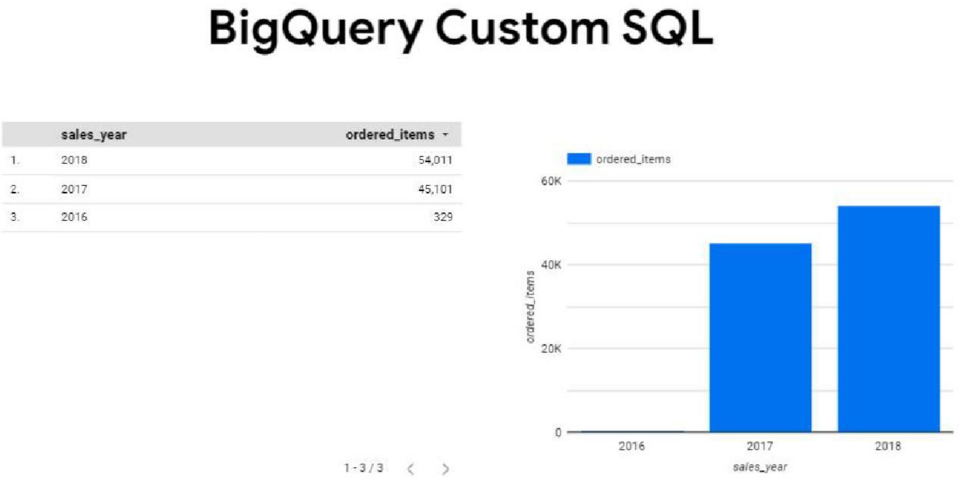
1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

```
SELECT  EXTRACT(Year from order_purchase_timestamp) as sales_year,Extract(Month from order_purchase_timestamp) as sales_month,count(order_id) as ordered_items FROM
`jeeva-scaler-demo.target_sql.orders` Group by sales_year,sales_month order by sales_year,sales_month
```



```
SELECT  EXTRACT(Year from order_purchase_timestamp) as sales_year,Extract(Month from order_purchase_timestamp) as sales_month,count(order_id) as
ordered_items FROM
`jeeva-scaler-demo.target_sql.orders` Group by sales_year,sales_month order by sales_year,sales_month
```

Row	sales_year	sales_month	ordered_items
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026
11	2017	8	4331
12	2017	9	4285

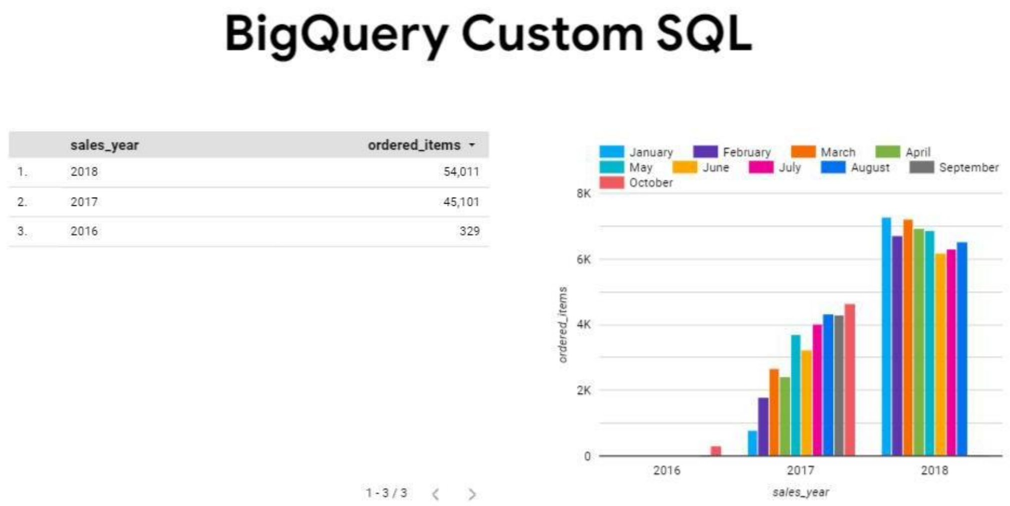


BigQuery Custom SQL

	sales_year	ordered_items
1.	2018	54,011
2.	2017	45,101
3.	2016	329

There is a steady increase in sales on year on year basis , starting from 329 items in 2016 , followed by 45101 items in 2017 and then 54011 items in 2018.

Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/



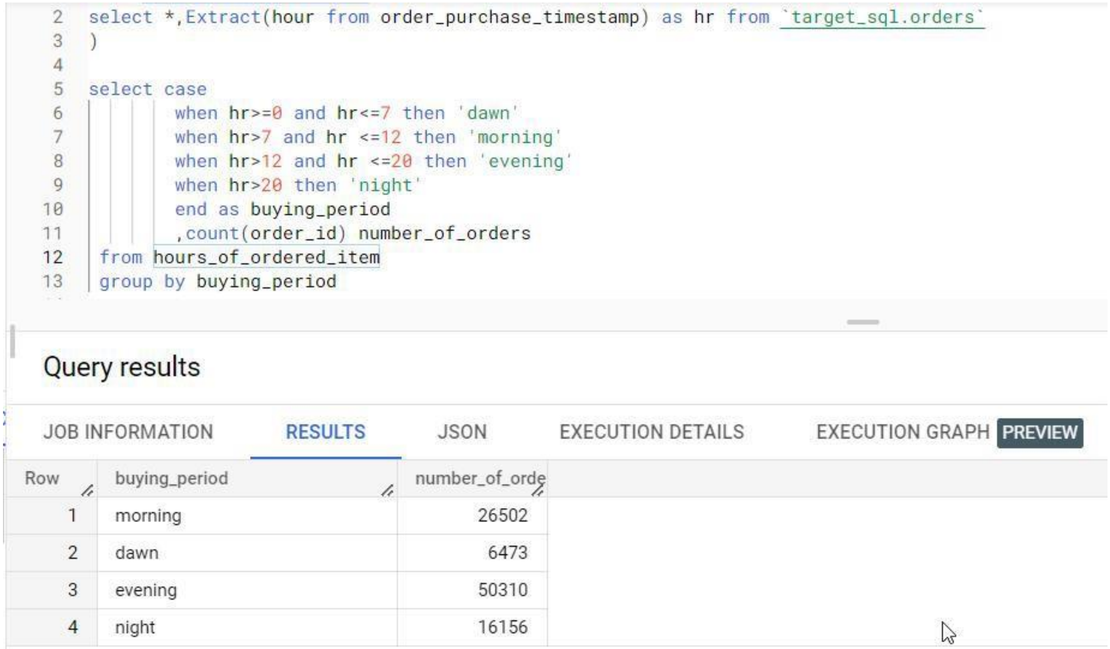
BigQuery Custom SQL

	sales_year	ordered_items ▾
1.	2018	54,011
2.	2017	45,101
3.	2016	329

Similarly there is peak in sales in october month for the year 2016 , october month in 2017 , January month in 2018.

2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

```
with hours_of_ordered_item as (
select *,Extract(hour from order_purchase_timestamp) as hr from `target_sql.orders`
)

select case
        when hr>=0 and hr<=7 then 'dawn'
        when hr>7 and hr <=12 then 'morning'
        when hr>12 and hr <=20 then 'evening'
        when hr>20 then 'night'
        end as buying_period
        ,count(order_id) number_of_orders
 from hours_of_ordered_item
 group by buying_period
```



```
select *,Extract(hour from order_purchase_timestamp) as hr from `target_sql.orders`
)

select case
        when hr>=0 and hr<=7 then 'dawn'
        when hr>7 and hr <=12 then 'morning'
        when hr>12 and hr <=20 then 'evening'
        when hr>20 then 'night'
        end as buying_period
        ,count(order_id) number_of_orders
 from hours_of_ordered_item
 group by buying_period
```

Query results

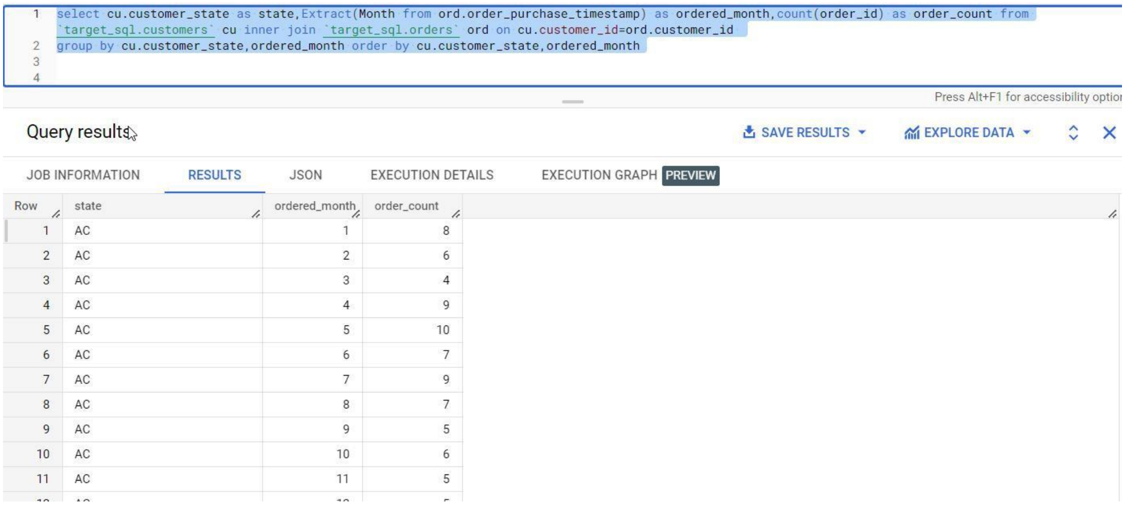
Row	buying_period	number_of_orde
1	morning	26502
2	dawn	6473
3	evening	50310
4	night	16156

We observe that brazilians are most active buyers during evening time and are very little active during the dawn.

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

1. Get month on month orders by states

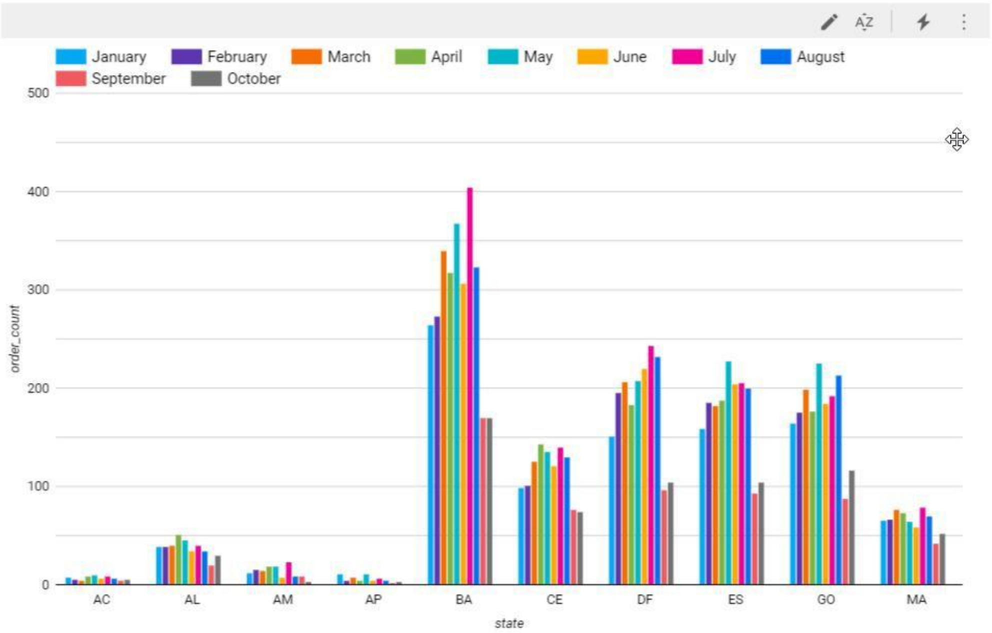
```
select cu.customer_state as state,Extract(Month from ord.order_purchase_timestamp) as ordered_month,count(order_id) as order_count from `target_sql.customers` cu inner join `target_sql.orders` ord on cu.customer_id=ord.customer_id
group by cu.customer_state,ordered_month order by cu.customer_state,ordered_month
```



```
select cu.customer_state as state,Extract(Month from ord.order_purchase_timestamp) as ordered_month,count(order_id) as order_count from
`target_sql.customers` cu inner join `target_sql.orders` ord on cu.customer_id=ord.customer_id
group by cu.customer_state,ordered_month order by cu.customer_state,ordered_month
```

Row	state	ordered_month	order_count
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
11	AC	11	5

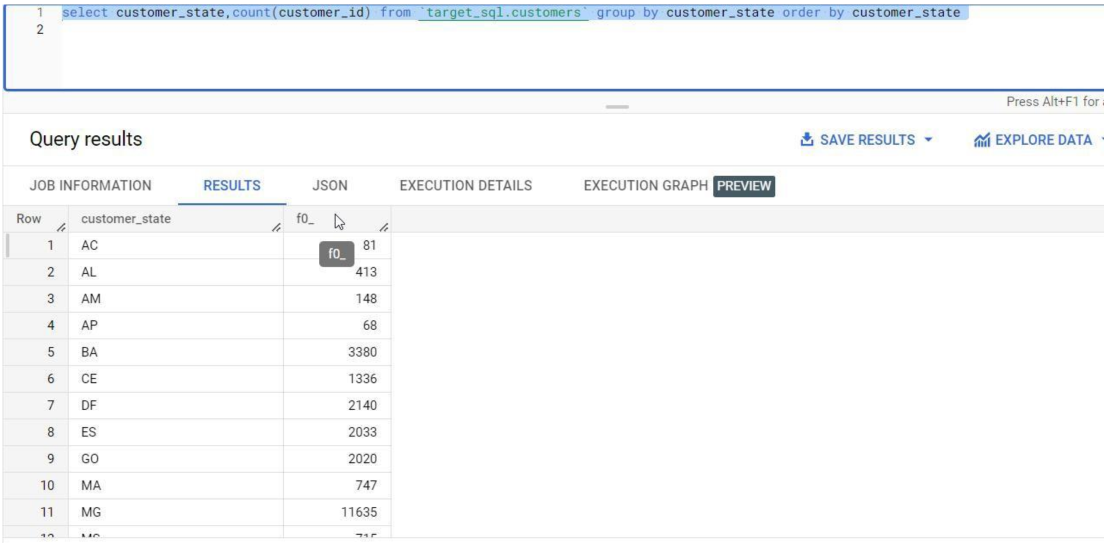
BigQuery Custom SQL



2. Distribution of customers across the states in Brazil

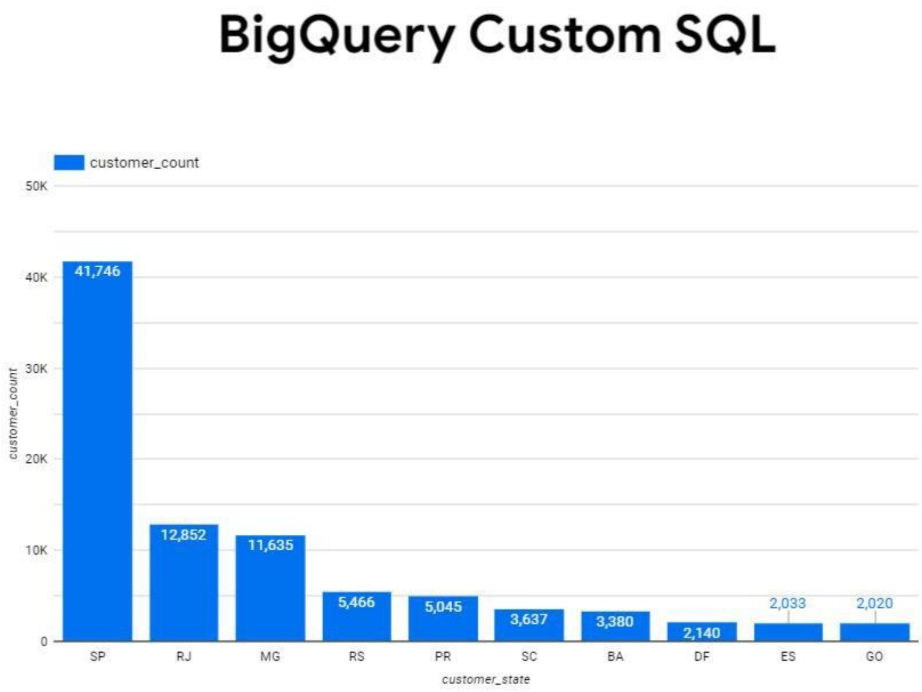
```
select customer_state,count(customer_id) from `target_sql.customers` group by customer_state order by customer_state
```

Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/



```
select customer_state,count(customer_id) from `target_sql.customers` group by customer_state order by customer_state
```

Row	customer_state	f0_
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
11	MG	11635

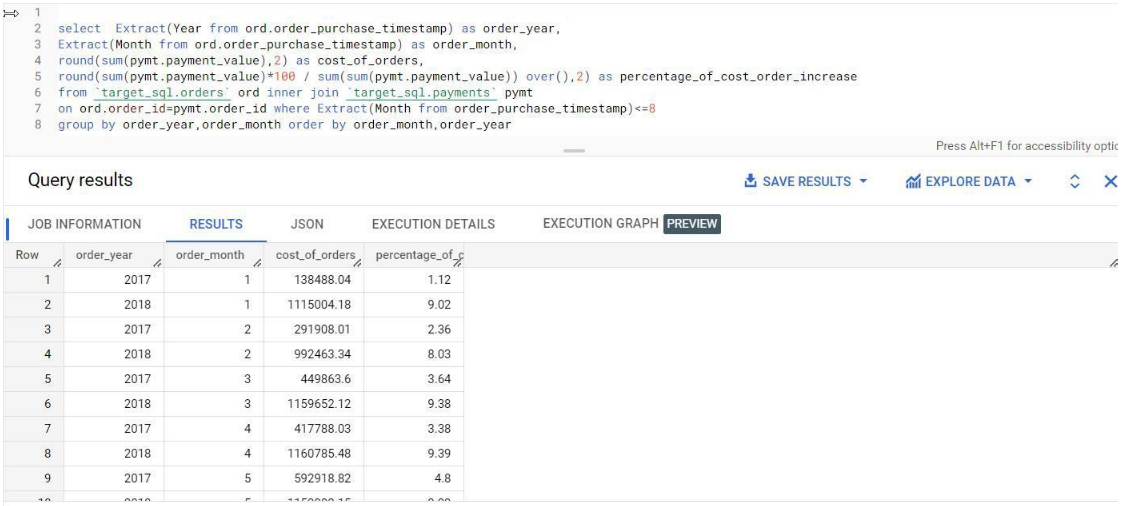


As we can see more customers are distributed in SP followed by the list of different states in brazil

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

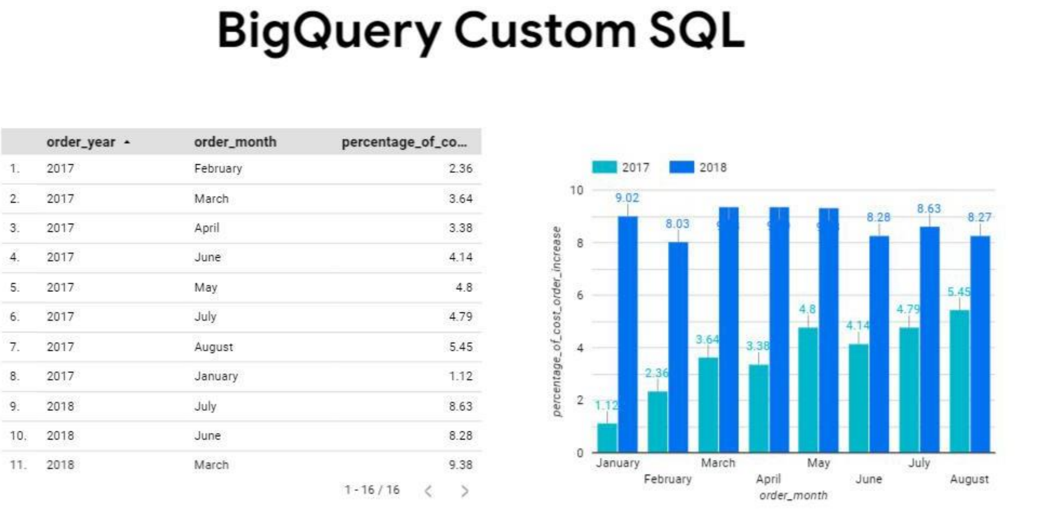
1. **Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table**

```
select  Extract(Year from ord.order_purchase_timestamp) as order_year,
Extract(Month from ord.order_purchase_timestamp) as order_month,
round(sum(pymt.payment_value),2) as cost_of_orders,
round(sum(pymt.payment_value)*100 / sum(sum(pymt.payment_value)) over(),2) as percentage_of_cost_order_incr
ease
from `target_sql.orders` ord inner join `target_sql.payments` pymt
on ord.order_id=pymt.order_id where Extract(Month from order_purchase_timestamp)<=8
group by order_year,order_month order by order_month,order_year
```



```
select  Extract(Year from ord.order_purchase_timestamp) as order_year,
Extract(Month from ord.order_purchase_timestamp) as order_month,
round(sum(pymt.payment_value),2) as cost_of_orders,
round(sum(pymt.payment_value)*100 / sum(sum(pymt.payment_value)) over(),2) as percentage_of_cost_order_increase
from `target_sql.orders` ord inner join `target_sql.payments` pymt
on ord.order_id=pymt.order_id where Extract(Month from order_purchase_timestamp)<=8
group by order_year,order_month order by order_month,order_year
```

Row	order_year	order_month	cost_of_orders	percentage_of_c
1	2017	1	138488.04	1.12
2	2018	1	1115004.18	9.02
3	2017	2	291908.01	2.36
4	2018	2	992463.34	8.03
5	2017	3	449863.6	3.64
6	2018	3	1159652.12	9.38
7	2017	4	417788.03	3.38
8	2018	4	1160785.48	9.39
9	2017	5	592918.82	4.8



BigQuery Custom SQL

	order_year ▴	order_month	percentage_of_co...
1.	2017	February	2.36
2.	2017	March	3.64
3.	2017	April	3.38
4.	2017	June	4.14
5.	2017	May	4.8
6.	2017	July	4.79
7.	2017	August	5.45
8.	2017	January	1.12
9.	2018	July	8.63
10.	2018	June	8.28
11.	2018	March	9.38

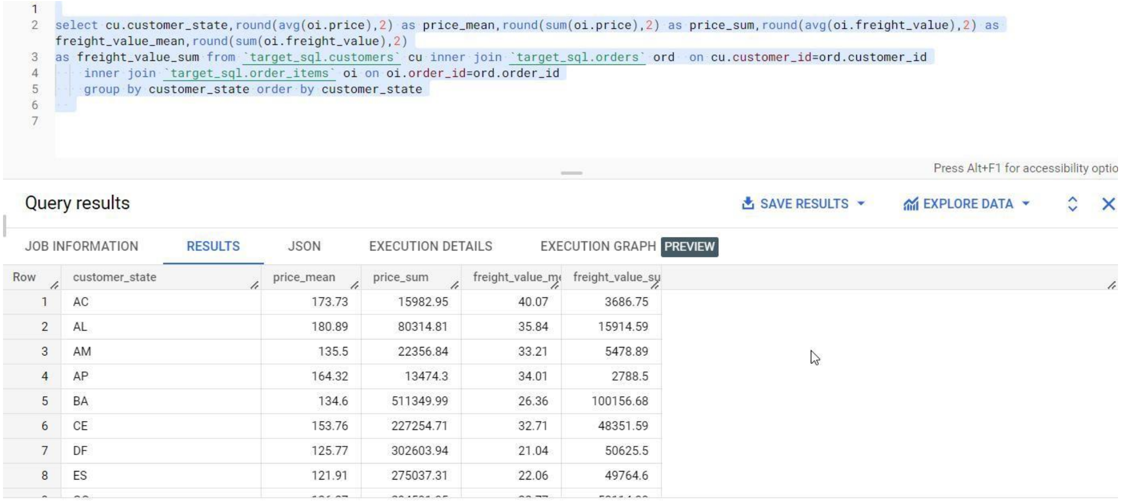
1 - 16 / 16

We observe that the there is good percentage of increase between 2017 and 2018 for the same set of months as shown in the bar chart.

2. Mean & Sum of price and freight value by customer state

```
select cu.customer_state,round(avg(oi.price),2) as price_mean,round(sum(oi.price),2) as price_sum,round(avg
(oi.freight_value),2) as freight_value_mean,round(sum(oi.freight_value),2)
as freight_value_sum from `target_sql.customers` cu inner join `target_sql.orders` ord  on cu.customer_id=o
rd.customer_id
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    group by customer_state order by customer_state
```

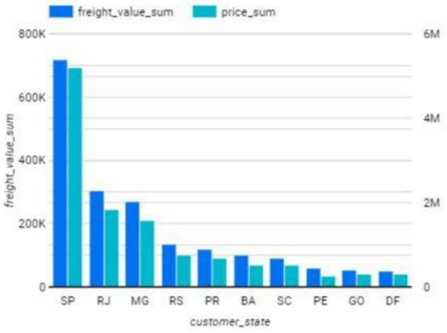
Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/

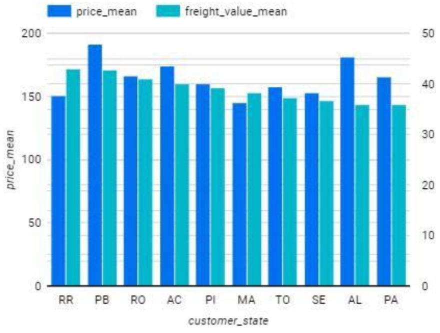


```sql

select cu.customer_state,round(avg(oi.price),2) as price_mean,round(sum(oi.price),2) as price_sum,round(avg(oi.freight_value),2) as
freight_value_mean,round(sum(oi.freight_value),2)
as freight_value_sum from `target_sql.customers` cu inner join `target_sql.orders` ord  on cu.customer_id=ord.customer_id
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    group by customer_state order by customer_state

```

Row	customer_state	price_mean	price_sum	freight_value_me	freight_value_su
1	AC	173.73	15982.95	40.07	3686.75
2	AL	180.89	80314.81	35.84	15914.59
3	AM	135.5	22356.84	33.21	5478.89
4	AP	164.32	13474.3	34.01	2788.5
5	BA	134.6	511349.99	26.36	100156.68
6	CE	153.76	227254.71	32.71	48351.59
7	DF	125.77	302603.94	21.04	50625.5
8	ES	121.91	275037.31	22.06	49764.6

BigQuery Custom SQL





5. Analysis on sales, freight and delivery time

- **Calculate days between purchasing, delivering and estimated delivery**
- **Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:**
- **time_to_delivery = order_purchase_timestamp-order_delivered_customer_date**
- **diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date**
- **Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery**
- **Sort the data to get the following:**

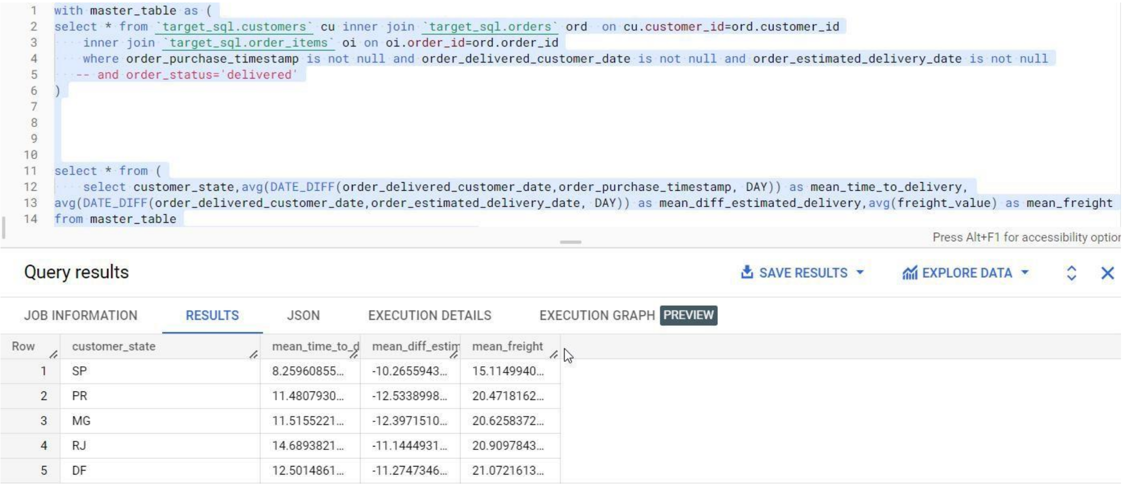
```sql
with master_table as (
select * from `target_sql.customers` cu inner join `target_sql.orders` ord  on cu.customer_id=ord.customer_
id
```

```
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
   -- and order_status='delivered'
)
```

- **Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5**

Ascending :

```
select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_freight asc limit 5
```

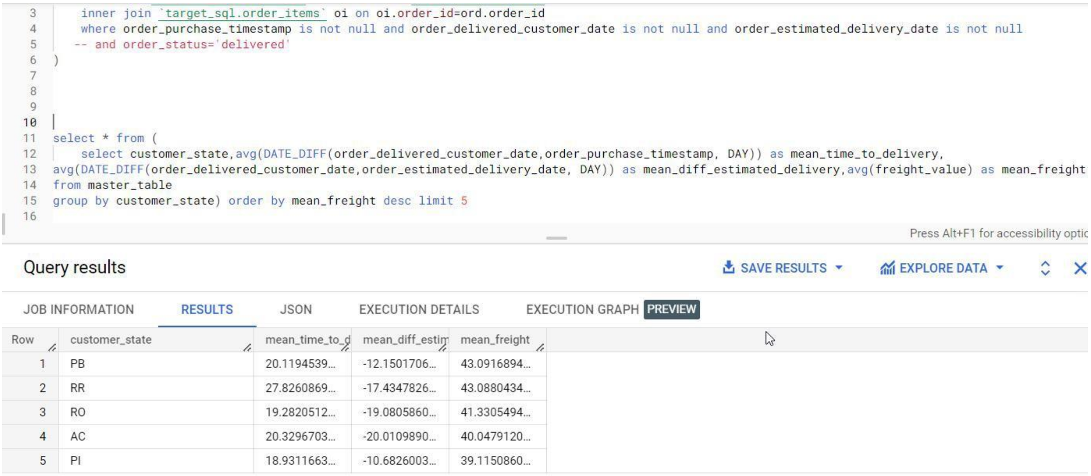


```
with master_table as (
select * from `target_sql.customers` cu inner join `target_sql.orders` ord  on cu.customer_id=ord.customer_id
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
   -- and order_status='delivered'
)




select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
```

Row	customer_state	mean_time_to_d	mean_diff_estir	mean_freight
1	SP	8.25960855...	-10.2655943...	15.1149940...
2	PR	11.4807930...	-12.5338998...	20.4718162...
3	MG	11.5155221...	-12.3971510...	20.6258372...
4	RJ	14.6893821...	-11.1444931...	20.9097843...
5	DF	12.5014861...	-11.2747346...	21.0721613...

Descending :

```
Select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_freight desc limit 5
```



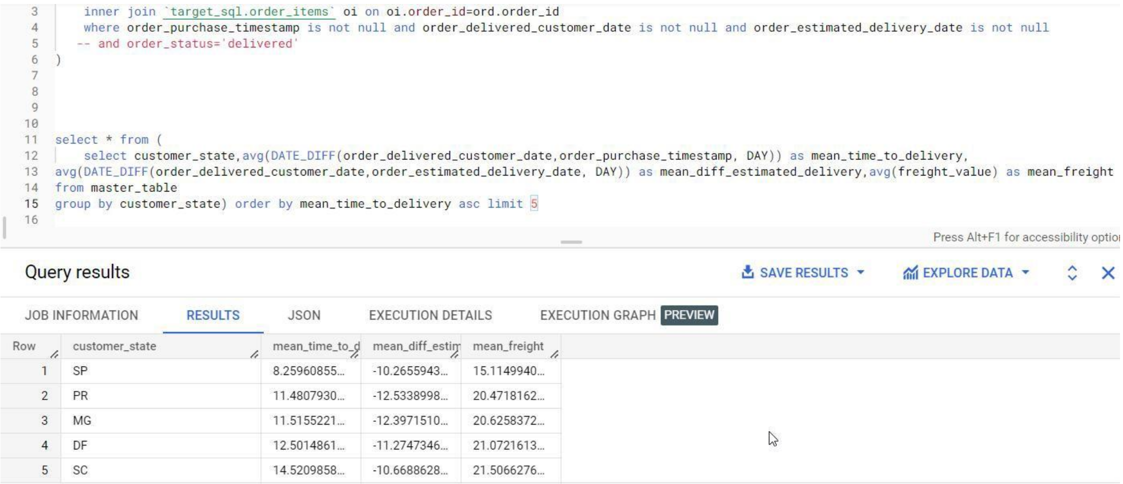
```
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
   -- and order_status='delivered'
)




select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_freight desc limit 5

```

Row	customer_state	mean_time_to_d	mean_diff_estir	mean_freight
1	PB	20.1194539...	-12.1501706...	43.0916894...
2	RR	27.8260869...	-17.4347826...	43.0880434...
3	RO	19.2820512...	-19.0805860...	41.3305494...
4	AC	20.3296703...	-20.0109890...	40.0479120...
5	PI	18.9311663...	-10.6826003...	39.1150860...

Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/

- **Top 5 states with highest/lowest average time to delivery**

Ascending :

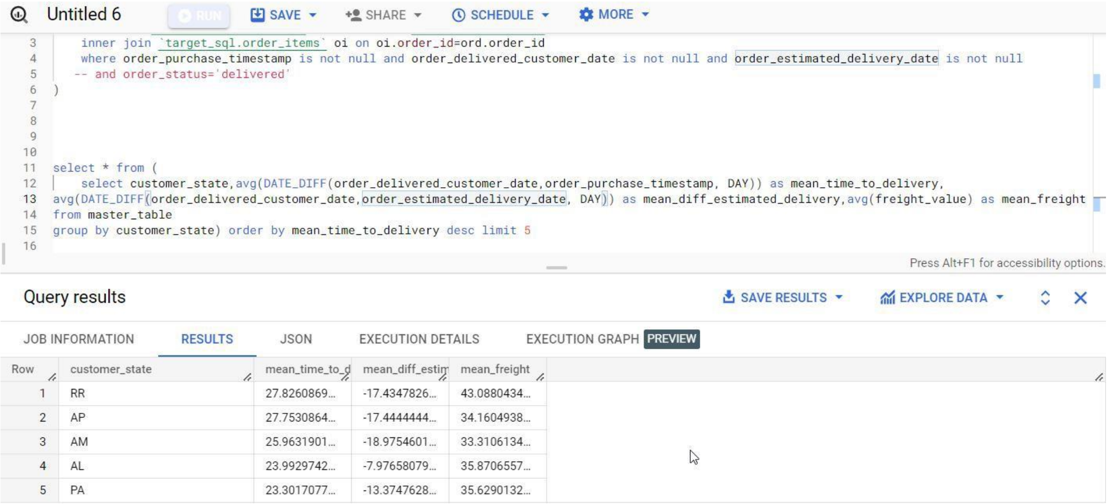


```
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
  -- and order_status='delivered'
)




select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_time_to_delivery asc limit 5
```

Row	customer_state	mean_time_to_d	mean_diff_estim	mean_freight
1	SP	8.25960855...	-10.2655943...	15.1149940...
2	PR	11.4807930...	-12.5338998...	20.4718162...
3	MG	11.5155221...	-12.3971510...	20.6258372...
4	DF	12.5014861...	-11.2747346...	21.0721613...
5	SC	14.5209858...	-10.6688628...	21.5066276...

Descending :

```
select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_time_to_delivery desc limit 5
```



```
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
  -- and order_status='delivered'
)




select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_time_to_delivery desc limit 5
```

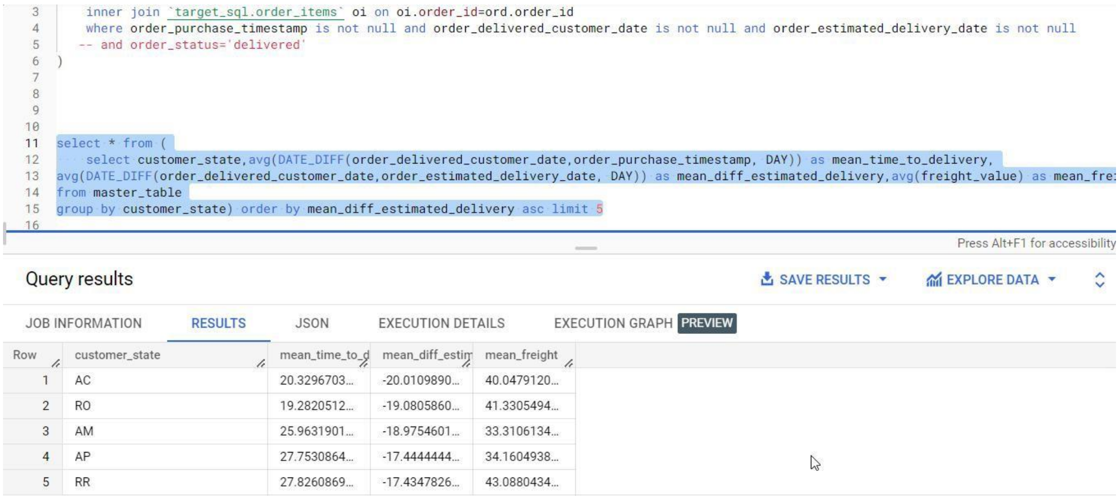
Row	customer_state	mean_time_to_d	mean_diff_estim	mean_freight
1	RR	27.8260869...	-17.4347826...	43.0880434...
2	AP	27.7530864...	-17.4444444...	34.1604938...
3	AM	25.9631901...	-18.9754601...	33.3106134...
4	AL	23.9929742...	-7.97658079...	35.8706557...
5	PA	23.3017077...	-13.3747628...	35.6290132...

- **Top 5 states where delivery is really fast/ not so fast compared to estimated date**

Fastest Delivery

Ascending:

```
select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_diff_estimated_delivery asc limit 5
```

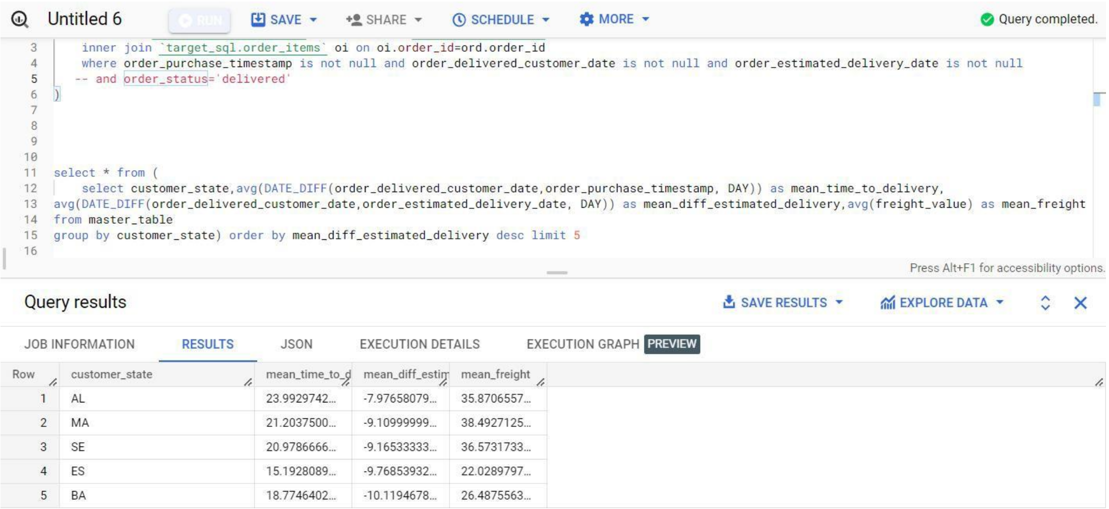


```
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
  -- and order_status='delivered'
)




select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_fre
from master_table
group by customer_state) order by mean_diff_estimated_delivery asc limit 5
```

Row	customer_state	mean_time_to_d	mean_diff_estim	mean_freight
1	AC	20.3296703...	-20.0109890...	40.0479120...
2	RO	19.2820512...	-19.0805860...	41.3305494...
3	AM	25.9631901...	-18.9754601...	33.3106134...
4	AP	27.7530864...	-17.4444444...	34.1604938...
5	RR	27.8260869...	-17.4347826...	43.0880434...

Descedning :

```
select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_diff_estimated_delivery desc limit 5
```



```
    inner join `target_sql.order_items` oi on oi.order_id=ord.order_id
    where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
  -- and order_status='delivered'
)




select * from (
    select customer_state,avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) as mean_time_to_delivery,
avg(DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date, DAY)) as mean_diff_estimated_delivery,avg(freight_value) as mean_freight
from master_table
group by customer_state) order by mean_diff_estimated_delivery desc limit 5
```

Row	customer_state	mean_time_to_d	mean_diff_estim	mean_freight
1	AL	23.9929742...	-7.97658079...	35.8706557...
2	MA	21.2037500...	-9.10999999...	38.4927125...
3	SE	20.9786666...	-9.16533333...	36.5731733...
4	ES	15.1928089...	-9.76853932...	22.0289797...
5	BA	18.7746402...	-10.1194678...	26.4875563...

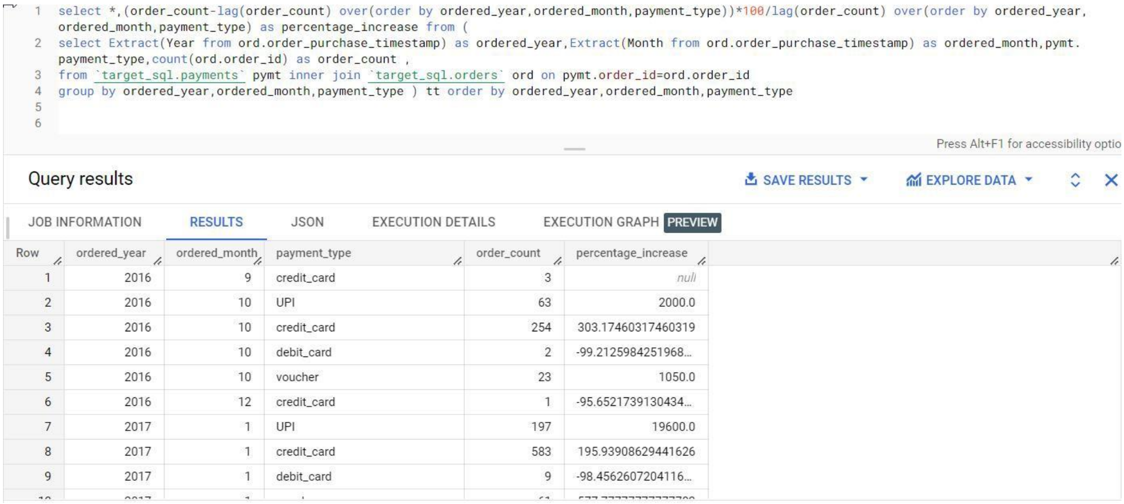
6. Payment type analysis:

Month over Month count of orders for different payment types

```
select *,(order_count-lag(order_count) over(order by ordered_year,ordered_month,payment_type))*100/lag(order_count) over(order by
 ordered_year,ordered_month,payment_type) as percentage_increase from (
```

Ankit
Github Profile: https://github.com/mrankit560
Linkedin profile : https://www.linkedin.com/in/theankitpaul/

```
select Extract(Year from ord.order_purchase_timestamp) as ordered_year,Extract(Month from ord.order_purchas
e_timestamp) as ordered_month,pymt.payment_type,count(ord.order_id) as order_count ,
from `target_sql.payments` pymt inner join `target_sql.orders` ord on pymt.order_id=ord.order_id
group by ordered_year,ordered_month,payment_type ) tt order by ordered_year,ordered_month,payment_type
```



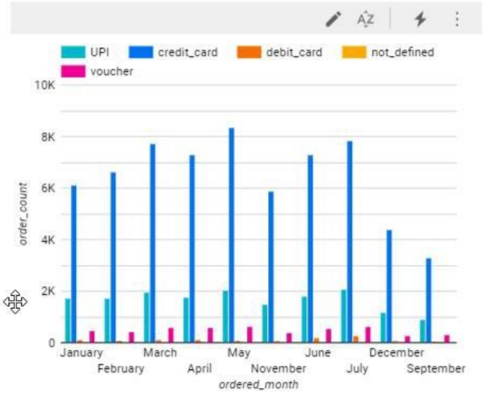
```
select *,(order_count-lag(order_count) over(order by ordered_year,ordered_month,payment_type))*100/lag(order_count) over(order by ordered_year,
ordered_month,payment_type) as percentage_increase from (
select Extract(Year from ord.order_purchase_timestamp) as ordered_year,Extract(Month from ord.order_purchase_timestamp) as ordered_month,pymt.
payment_type,count(ord.order_id) as order_count ,
from `target_sql.payments` pymt inner join `target_sql.orders` ord on pymt.order_id=ord.order_id
group by ordered_year,ordered_month,payment_type ) tt order by ordered_year,ordered_month,payment_type
```

Row	ordered_year	ordered_month	payment_type	order_count	percentage_increase
1	2016	9	credit_card	3	null
2	2016	10	UPI	63	2000.0
3	2016	10	credit_card	254	303.17460317460319
4	2016	10	debit_card	2	-99.2125984251968...
5	2016	10	voucher	23	1050.0
6	2016	12	credit_card	1	-95.6521739130434...
7	2017	1	UPI	197	19600.0
8	2017	1	credit_card	583	195.93908629441626
9	2017	1	debit_card	9	-98.4562607204116...

BigQuery Custom SQL

	payment_type	order_count
1.	credit_card	76,795
2.	UPI	19,784
3.	voucher	5,775
4.	debit_card	1,529
5.	not_defined	3

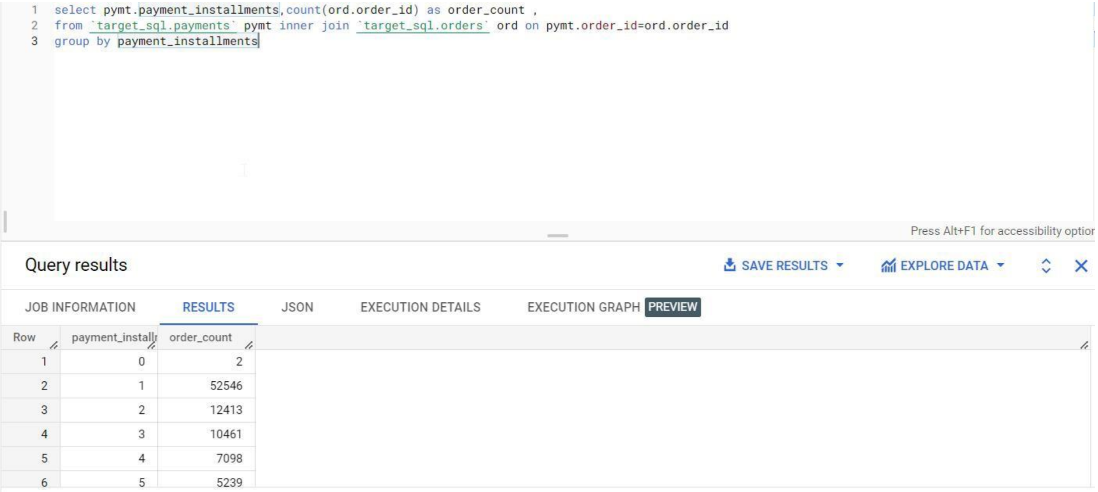
1 - 5 / 5



We could clearly see that credit card orders are rapidly increasing on month over month basis followed by UPI payments.

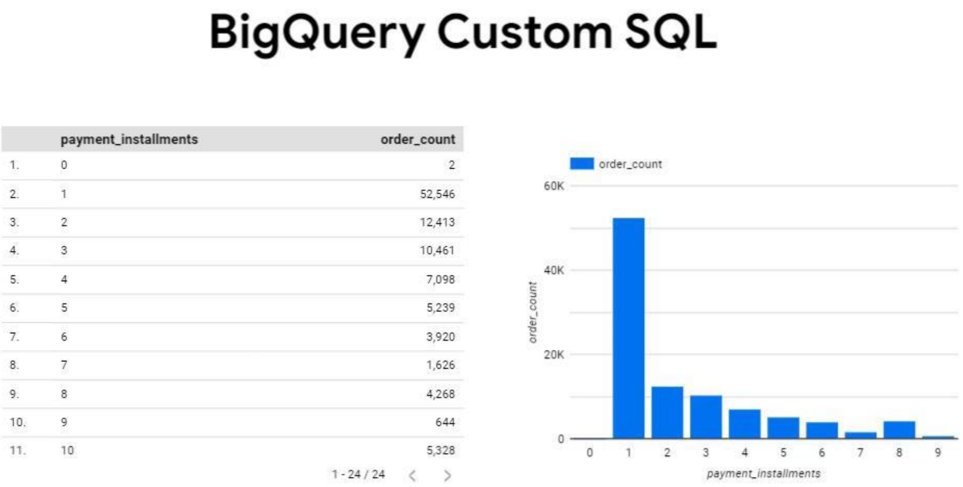
Count of orders based on the no. of payment installments

```
select pymt.payment_installments,count(ord.order_id) as order_count ,
from `target_sql.payments` pymt inner join `target_sql.orders` ord on pymt.order_id=ord.order_id
group by payment_installments
```



```
select pymt.payment_installments,count(ord.order_id) as order_count ,
from `target_sql.payments` pymt inner join `target_sql.orders` ord on pymt.order_id=ord.order_id
group by payment_installments
```

Row	payment_install	order_count
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239



BigQuery Custom SQL

	payment_installments	order_count
1.	0	2
2.	1	52,546
3.	2	12,413
4.	3	10,461
5.	4	7,098
6.	5	5,239
7.	6	3,920
8.	7	1,626
9.	8	4,268
10.	9	644
11.	10	5,328

1 - 24 / 24

More orders prefer single installment followed by 2 , 3

7. Actionable insights

Following are the insights from the EDA

- There is a steady increase in sales on year on year basis , starting from 329 items in 2016 , followed by 45101 items in 2017 and then 54011 items in 2018.
- Similarly there is peak in sales in october month for the year 2016 , october month in 2017 , January month in 2018.
- More customer base in SP state and less customer base in RR state
- PB has the highest mean frieght value and SP scores the least
- SP tops with lowest delivery time whereas RR requires more time to deliver
- AC tops with quick delivery than committed / estimated time whereas AL lags little bit compartively.
- Debit card orders are very less whereas credit card and UPI are the most preferred payment types
- Most people prefer single payment installment and very few prefer 9 month installment rate

8. Recommendation

Following are the recommendations from the EDA
1. RR customer base is really low as well as RR delivery time is also comparitvely low , so it if delivery time is improved we might see good increase in RR customer base.
2. As credit card payments are more preferred , credit related discounts could be encouraged more by providing offers with new payment installement schemes