Retailer Data Analysis using BigQuery by Diptyajit Das

Problem Statement:

An in-depth analysis of the dataset featuring 100,000 orders placed in Brazil between 2016 and 2018 provides an opportunity to understand the operational dynamics of a prominent retailer. The dataset covers dimensions like order status, pricing, payment and freight performance, customer locations, product attributes, and reviews. By delving into this comprehensive dataset, we aim to uncover valuable insights into the retailer's operations in Brazil.

The investigation will focus on key areas, including order processing efficiency, pricing strategies, effectiveness in payment and shipping processes, customer demographics, product characteristics, and overall customer satisfaction. The goal is to identify operational strengths, areas for improvement, and strategic opportunities that can contribute to enhancing the retailer's performance in the Brazilian market.

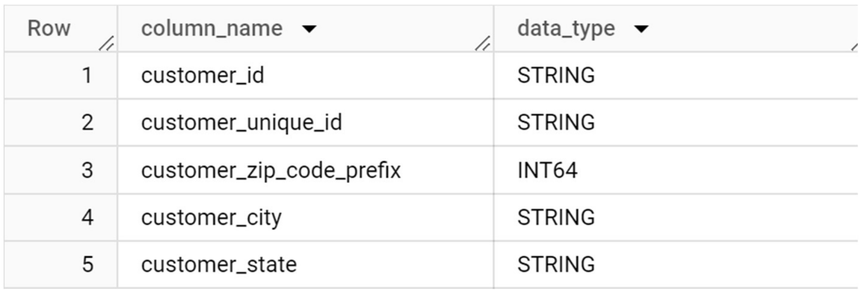
---

1. Initial exploration of dataset like checking the characteristics of data

1.A. Data type of all columns in the “customers” table.

```
#Query:
select column_name,data_type
from target.INFORMATION_SCHEMA.COLUMNS
where table_name='customers'
```

```
#Result:
```



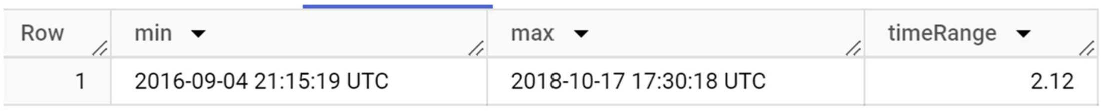
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

#Insight: Four String datatype columns are present and one integer datatype column is present. So we are storing string type heavy data.

1.B. Time range between which the orders were placed.

```
#Query:
select min(order_purchase_timestamp) as min,
max(order_purchase_timestamp) as max,
round(timestamp_diff(max(order_purchase_timestamp),min(order_purchase_timestamp),da
y)/365,2) as timeRange
from target.orders;
```

#Result:



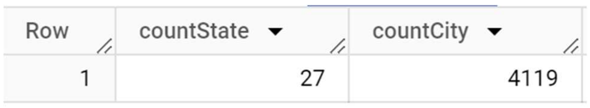
Row	min	max	timeRange
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC	2.12

#Insight: Orders were made over a span of 2.12 years starting from September 2016 to October 2018.

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

```
#Query:
select count(distinct c.customer_state) as countState,
count(distinct c.customer_city)as countCity
from target.customers c join target.orders o
using(customer_id);
```

#Result:



Row	countState	countCity
1	27	4119

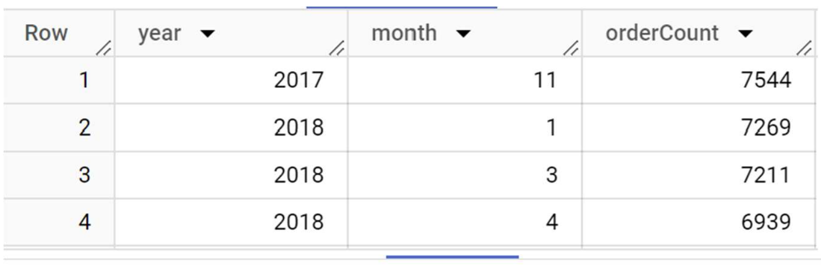
#Insight: Sales have coverage of 4119 cities over 27 states.

2. In-depth Exploration

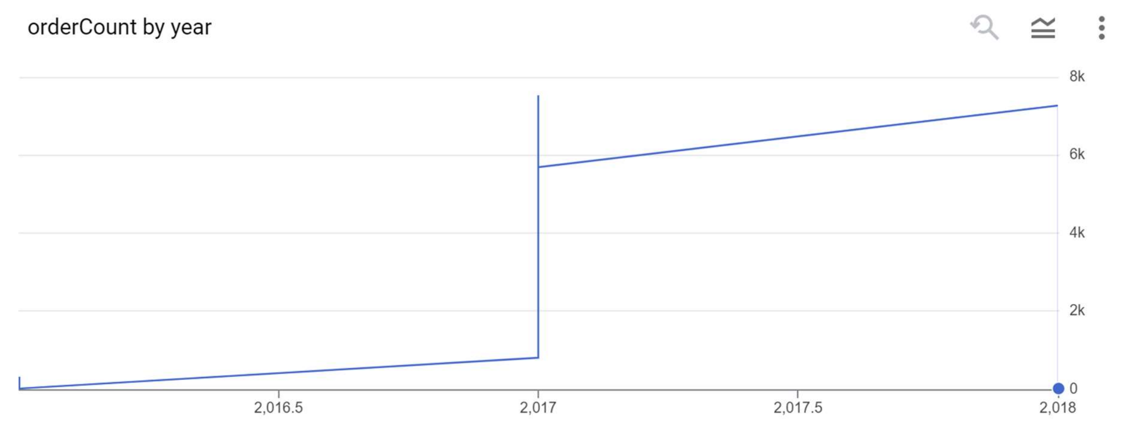
2.A. Is there a growing trend in the no. of orders placed over the past years?

```
#Query:
select extract(year from order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month,
count(distinct order_id) as orderCount
from target.orders
group by 1,2
order by 3 desc;
```

```
#Result:
```



Row	year	month	orderCount
1	2017	11	7544
2	2018	1	7269
3	2018	3	7211
4	2018	4	6939

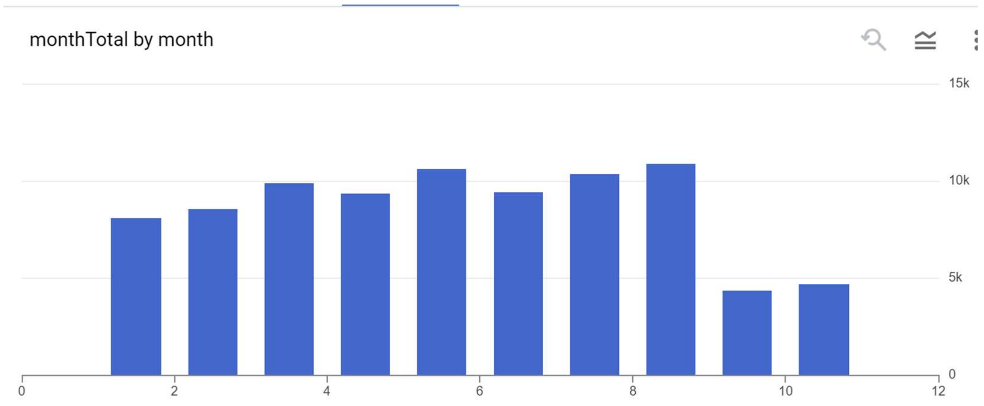


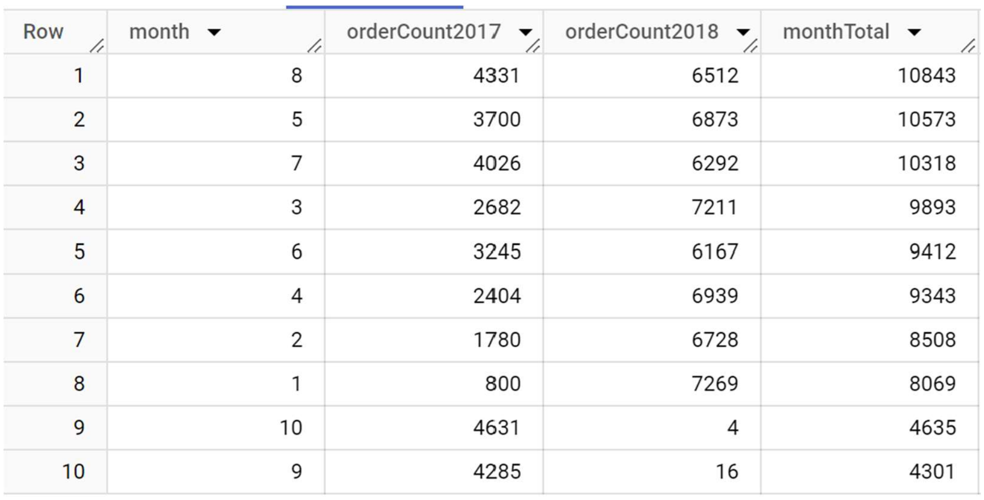
#Insight: OrderCount has always increased.In 2017 start we see a huge jump and it has even increased further after that.

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

```
#Query:
with T as(select extract(year from order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month,
count(distinct order_id) as orderCount
from target.orders
group by 1,2),
T1 as(select month as month2017,orderCount as orderCount2017 from T where
year=2017),
T2 as(select month as month2018,orderCount as orderCount2018 from T where
year=2018)
select month2017 as
month,orderCount2017,orderCount2018,orderCount2017+orderCount2018 as monthTotal
from T1 a join T2 b on month2017=month2018
order by monthTotal desc;
```

#Result:





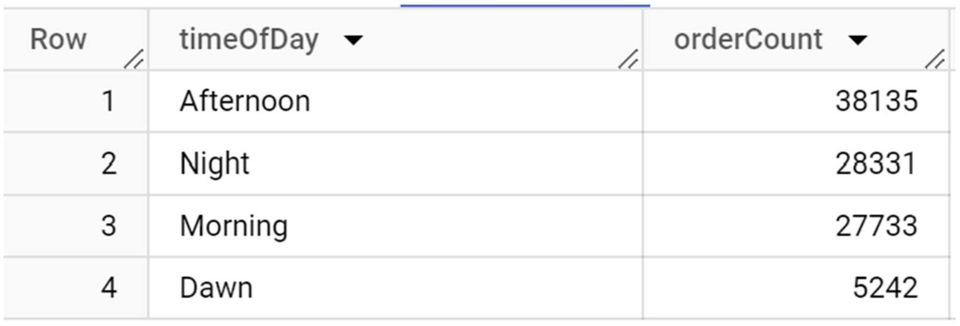
Row	month	orderCount2017	orderCount2018	monthTotal
1	8	4331	6512	10843
2	5	3700	6873	10573
3	7	4026	6292	10318
4	3	2682	7211	9893
5	6	3245	6167	9412
6	4	2404	6939	9343
7	2	1780	6728	8508
8	1	800	7269	8069
9	10	4631	4	4635
10	9	4285	16	4301

#Insight: Peak of orderCount is in August with maximum aggregated orderCount over 2017 and 2018.

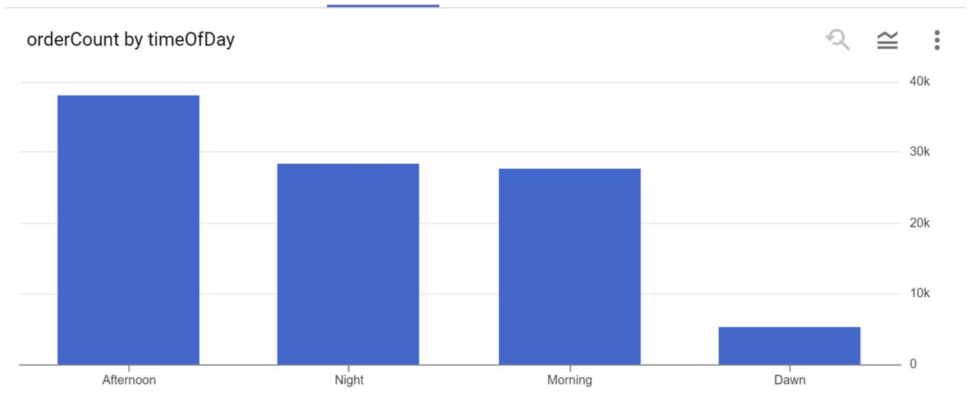
2.C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

```
#Query:
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 timeOfDay,count(order_id) as orderCount
from target.orders
group by 1
order by 2 desc;
```

Result:



Row	timeOfDay	orderCount
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242



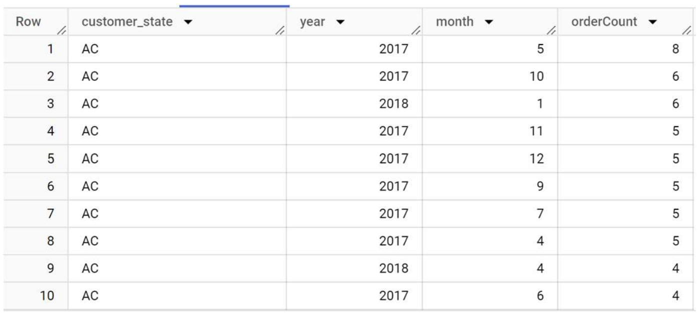
#Insight: Highest orders are made during afternoon followed by night and morning.
Recommendation: Can arrange for afternoon limited time sale.

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

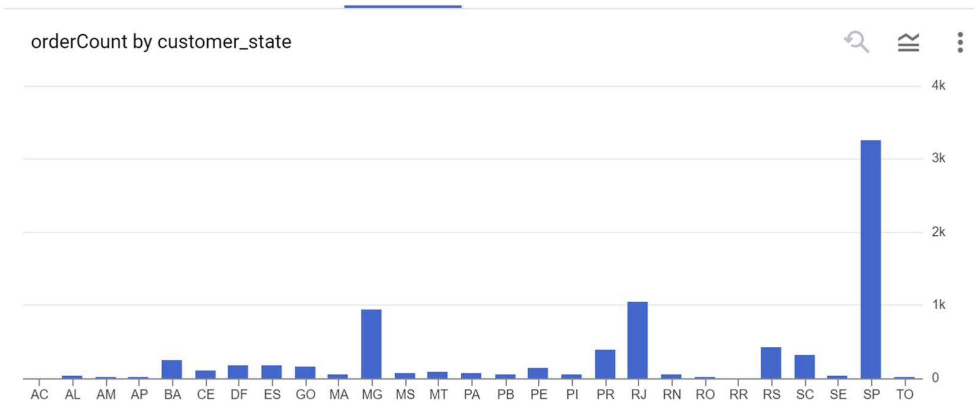
3.A. The month on month no. of orders placed in each state.

```
#Query:
select customer_state,
extract(year from order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month,
count(order_id) as orderCount
from target.orders join target.customers using(customer_id)
group by 1,2,3
order by 1,4 desc;
```

#Result:



Row	customer_state	year	month	orderCount
1	AC	2017	5	8
2	AC	2017	10	6
3	AC	2018	1	6
4	AC	2017	11	5
5	AC	2017	12	5
6	AC	2017	9	5
7	AC	2017	7	5
8	AC	2017	4	5
9	AC	2018	4	4
10	AC	2017	6	4

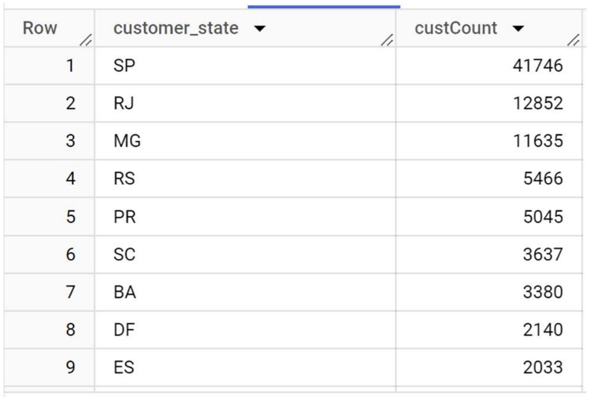


#Insight: OrderCount is highest in SP,RJ and MG states.

3.B. How are the customers distributed across all the states?

```
#Query:
select customer_state,count(distinct customer_id) as custCount
from target.customers
group by 1
order by 2 desc;
```

Result:



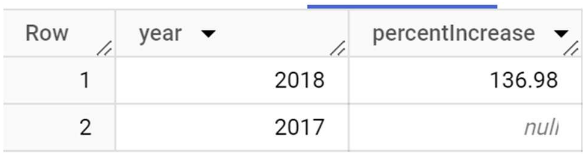
Row	customer_state	custCount
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033

#Insight: Customer count is maximum in SP,RJ,MG states which also supports the previous orderCount data.

4. Impact on Economy

4.A. The % increase in the cost of orders from year 2017 to 2018 (including months between Jan to Aug only).

```
#Query:
with T as
(
  select extract(year from o.order_purchase_timestamp) as year,
  sum(p.payment_value) as total from target.orders o join target.payments p
using(order_id)
  where extract(year from o.order_purchase_timestamp) in(2017,2018)
  and extract(month from o.order_purchase_timestamp) between 1 and 8
  group by 1
),
T1 as (select *,lag(total) over(order by year) lagTotal from T)
select year,round((total-lagTotal)*100/lagTotal,2) as percentIncrease from T1;

#Result:
```

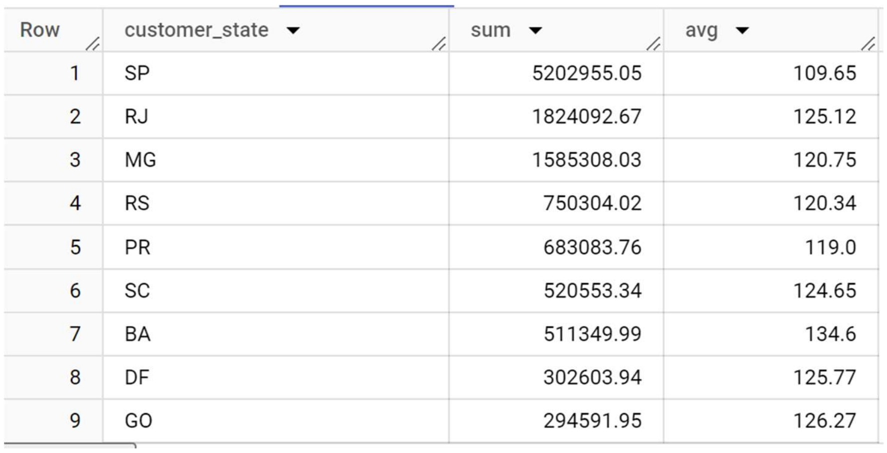


Row	year	percentIncrease
1	2018	136.98
2	2017	null

#Insight: 137% increase in costOfOrder is observed which is a good sign.

4.B. The Total & Average value of order price for each state.

```
#Query:
select c.customer_state,round(sum(oi.price),2) as sum,round(avg(oi.price),2) as avg
from target.customers c join target.orders using(customer_id)
join target.order_items oi using(order_id)
group by 1
order by 2 desc, 3 desc;

#Result:
```



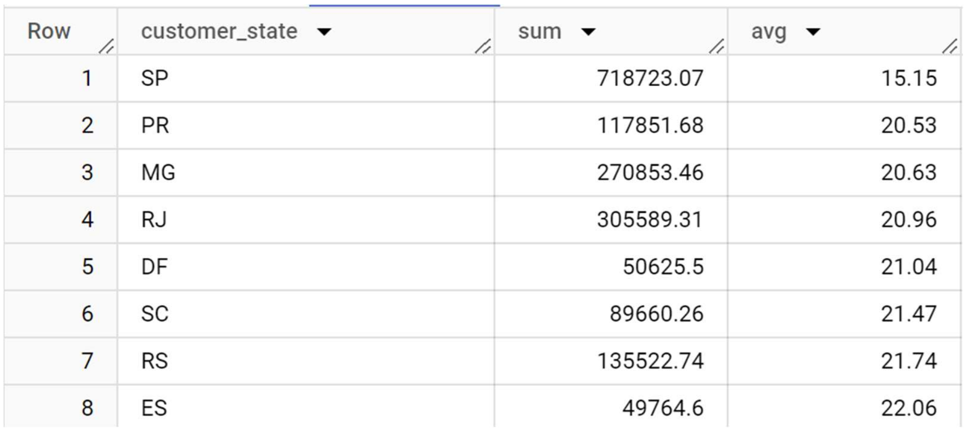
Row	customer_state	sum	avg
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65
7	BA	511349.99	134.6
8	DF	302603.94	125.77
9	GO	294591.95	126.27

#Insight: Again our previous result is matched with states SP RJ MG having highest sum of transactions but other states have higher average.

4.C. The Total & Average value of order freight for each state.

```
#Query:
select c.customer_state,round(sum(oi.freight_value),2) as
sum,round(avg(oi.freight_value),2) as avg
from target.customers c join target.orders using(customer_id)
join target.order_items oi using(order_id)
group by 1
order by 3;
```

Result:



Row	customer_state	sum	avg
1	SP	718723.07	15.15
2	PR	117851.68	20.53
3	MG	270853.46	20.63
4	RJ	305589.31	20.96
5	DF	50625.5	21.04
6	SC	89660.26	21.47
7	RS	135522.74	21.74
8	ES	49764.6	22.06

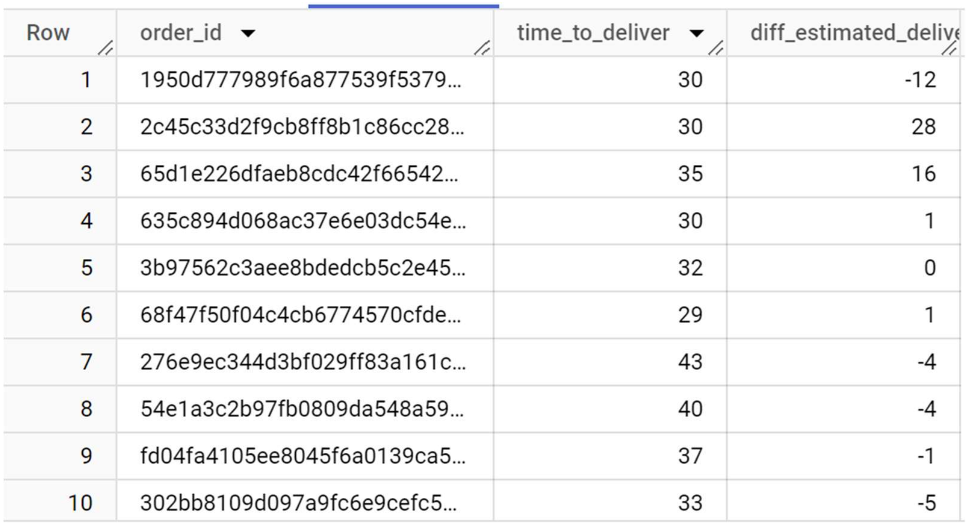
#Insight: SP with lowest average freight rate can have better sales.

5. Analysis on sales, freight and delivery time

5.A The no. of days taken to deliver each order from the order's purchase date as delivery time. And the difference (in days) between the estimated & actual delivery date of an order.

```
#Query:
select order_id,
timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
time_to_deliver,
timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as
diff_estimated_delivery
from target.orders;
```

```
#Result:
```

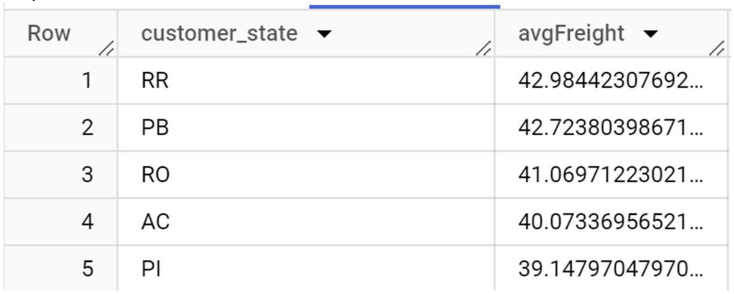


Row	order_id	time_to_deliver	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	33	-5

#Insight: Negative values mean greater delivery time than estimated.
#Recommendation: Actual delivery time and estimated delivery time should be closer.

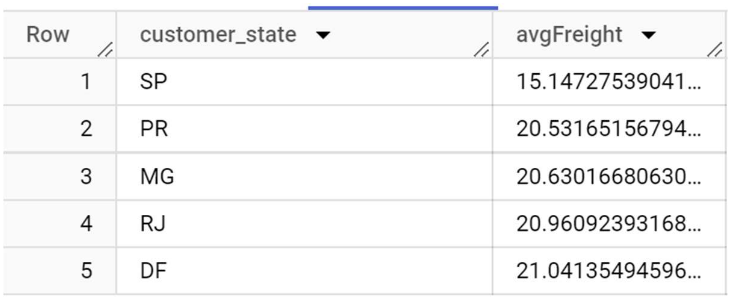
5.B. The top 5 states with the highest & lowest average freight value.

```
#Query:
select customer_state,avg(freight_value) as avgFreight
from target.orders o join target.customers c using(customer_id)
join target.order_items od using(order_id)
group by 1
order by 2 desc
limit 5;

select customer_state,avg(freight_value) as avgFreight
from target.orders o join target.customers c using(customer_id)
join target.order_items od using(order_id)
group by 1
order by 2
limit 5;

#Result:

Top 5:
```



Row	customer_state	avgFreight
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...

```
Bottom 5:
```

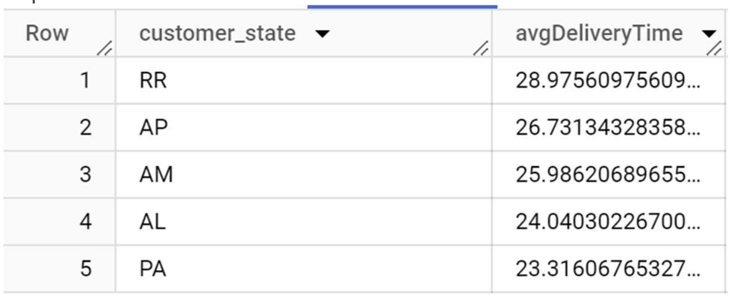


Row	customer_state	avgFreight
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...

`#Insight: States with lower avgFreight rates are getting more sales done easily.`

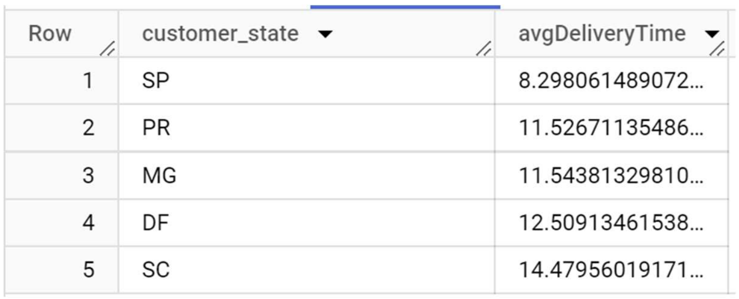
5.C. The top 5 states with the highest & lowest average delivery time.

```
#Query:
select customer_state,
avg(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))as
avgDeliveryTime
from target.orders o join target.customers c using(customer_id)
group by 1
order by 2 desc
limit 5;

select customer_state,
avg(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))as
avgDeliveryTime
from target.orders o join target.customers c using(customer_id)
group by 1
order by 2
limit 5;

#Result:

Top 5:
```



Row	customer_state	avgDeliveryTime
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...
4	AL	24.04030226700...
5	PA	23.31606765327...

```
Bottom 5:
```



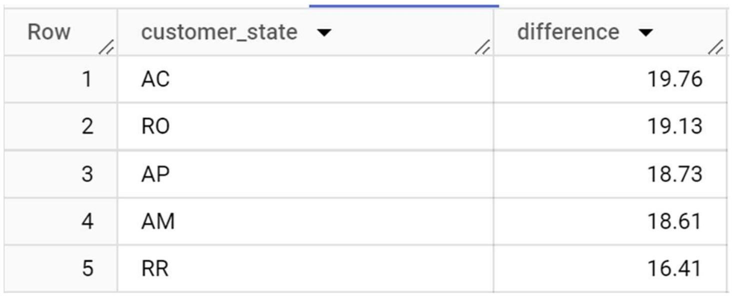
Row	customer_state	avgDeliveryTime
1	SP	8.298061489072...
2	PR	11.52671135486...
3	MG	11.54381329810...
4	DF	12.50913461538...
5	SC	14.47956019171...

#Insight: SP has lowest avgDeliveryTime and also has the best sales.
#Recommendation: Delivery time is directly related to customer experience so it should be reduced as much as possible.

5.D. The top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
#Query:
select customer_state,
round(avg(timestamp_diff(order_estimated_delivery_date,order_delivered_customer_dat
e,day)),2)as difference
from target.orders o join target.customers c using(customer_id)
where order_status='delivered'
group by 1
order by difference desc
limit 5;
```

#Result:



Row	customer_state	difference
1	AC	19.76
2	RO	19.13
3	AP	18.73
4	AM	18.61
5	RR	16.41

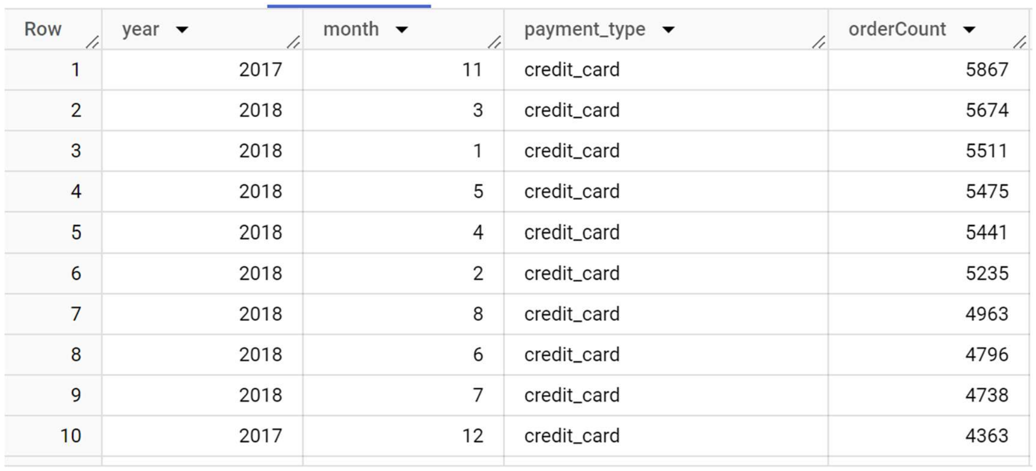
#Insight: Average delivery time is mostly lower than average estimated delivery time which is good.
#Recommendation: 20 days difference is also not acceptable should provide correct estimates.There should be a range like 0-10 days buffer for providing estimates.

6. Payment type analysis

6.A. The month on month no. of orders placed using different payment types.

```
#Query:
select extract(year from order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month,
payment_type,
count(distinct order_id) as orderCount
from target.orders o join target.payments p using(order_id)
group by 1,2,3
order by 4 desc;
```

```
#Result:
```



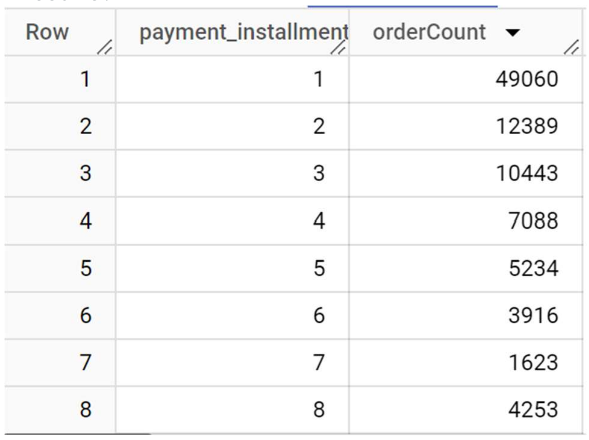
Row	year	month	payment_type	orderCount
1	2017	11	credit_card	5867
2	2018	3	credit_card	5674
3	2018	1	credit_card	5511
4	2018	5	credit_card	5475
5	2018	4	credit_card	5441
6	2018	2	credit_card	5235
7	2018	8	credit_card	4963
8	2018	6	credit_card	4796
9	2018	7	credit_card	4738
10	2017	12	credit_card	4363

#Insight: Maximum orders are placed using credit cards on November 2017.
Recommendation: Can give special discounts on credit card purchases.

6.B. The no. of orders placed on the basis of the payment installments that have been paid.

```
#Query:
select payment_installments,count(distinct order_id) as orderCount
from target.payments
where payment_installments>0
group by 1
order by 1;
```

```
#Result:
```



Row	payment_installment	orderCount
1	1	49060
2	2	12389
3	3	10443
4	4	7088
5	5	5234
6	6	3916
7	7	1623
8	8	4253

#Insight: Most people are purchasing the products in one go.
#Recommendation: Can offer bigger discounts on larger intsallment plans to increase overall sales profit.