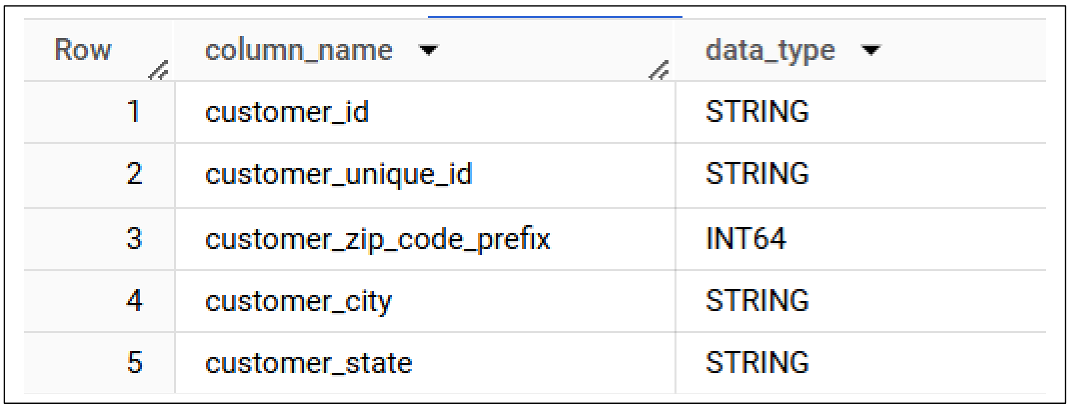
Target-Business_Case:-

1) Exploratory Analysis: -

a) Data type of all columns in the customer table:

Query:

```
SELECT column_name, data_type
FROM `target.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers';
```



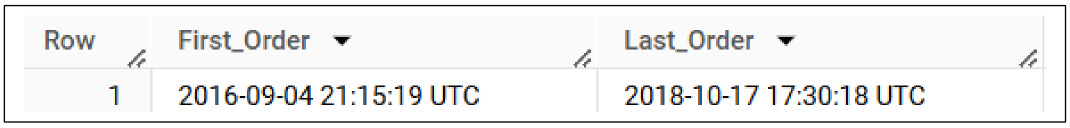
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Observation: There are 5 columns in the customers table and only **customer_zip_code_prefix** is of **INT** data type and rest all are of **STRING** data type**.**

b) Time range between which the orders were placed:

Query:

```
SELECT min(order_purchase_timestamp) as First_Order,
max(order_purchase_timestamp) as Last_Order
FROM target.orders;
```



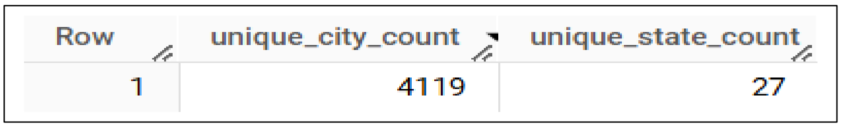
Row	First_Order	Last_Order
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Observation: As per the given data-set, **First Order** was placed on **4th Apr, 2016 around 9:15 PM** and **Last Order** was placed on **17th Oct, 2018 around 5:30PM**.

c) Count the Cities & States of customers who ordered during the given period:

Query:

```
SELECT count (distinct lower(trim(customer_city))) as
unique_city_count,
count (distinct lower(trim(customer_state))) as unique_state_count
FROM target.customers;
```



Row	unique_city_count	unique_state_count
1	4119	27

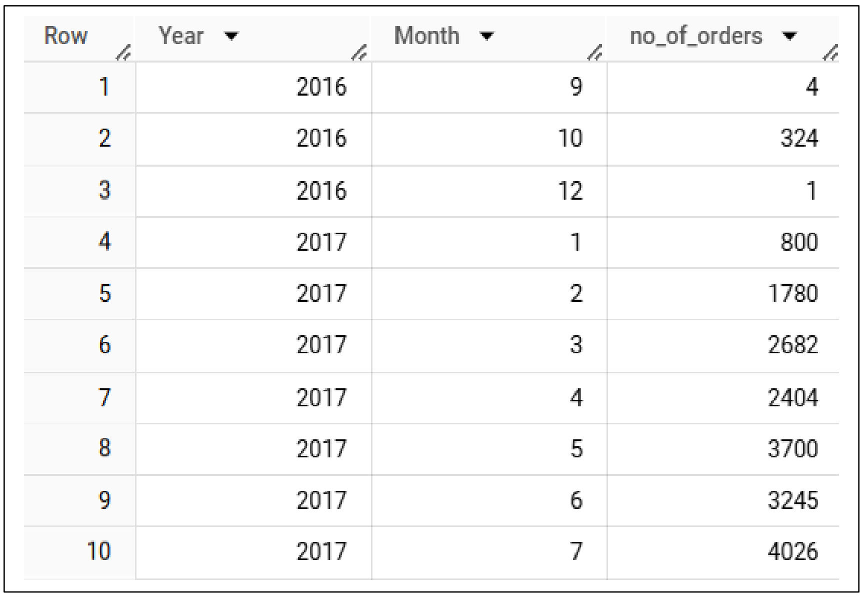
Observation: There are total **4,119 unique cities** and **27 different states** from which various customers have placed orders.

2) In depth exploration:

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(*) as
no_of_orders
FROM target.orders
GROUP BY Year, Month
ORDER BY Year, Month
```



Row	Year	Month	no_of_orders
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

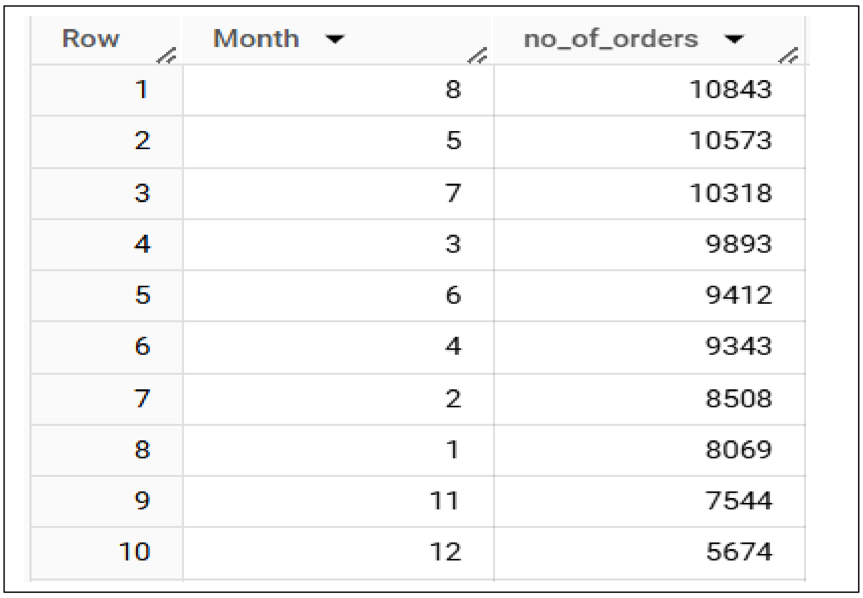
Observation:

- There is sudden increase in count of no of orders from **2016** to **2017**.
- In **2017**, there is gradual increase in the counts till **Oct-17** and in **Nov-17** there is sudden serge in counts and in **Dec-17** the counts decrease.
- In **2018**, the counts decrease gradually from **Jan-18** till **Aug-18** and after that the counts plummet so low that the counts are barely **16** in **Sept-18** and **4** in **Oct-18**.

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

Query:

```
SELECT extract (month from order_purchase_timestamp) as Month,
count (*) as no_of_orders
FROM target.orders
GROUP BY  Month
ORDER BY  no_of_orders desc
```

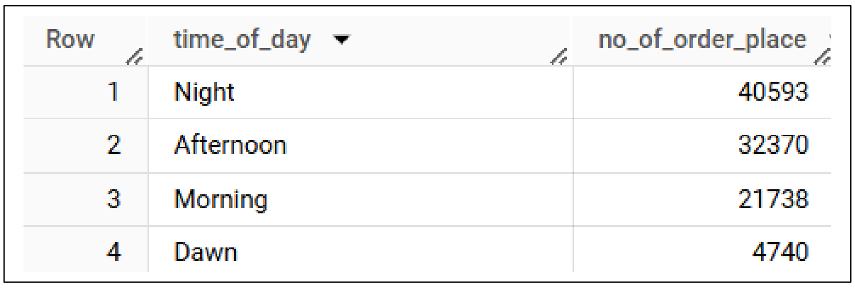


Row	Month	no_of_orders
1	8	10843
2	5	10573
3	7	10318
4	3	9893
5	6	9412
6	4	9343
7	2	8508
8	1	8069
9	11	7544
10	12	5674

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

Query:

```
SELECT time_of_day, count(*) as no_of_order_place
FROM
(
SELECT order_time, case when order_time between '00:00:00' and
'06:00:00' then 'Dawn'
when order_time between '07:00:00' and '12:00:00' then 'Morning'
when order_time between '13:00:00' and '18:00:00' then
'Afternoon'
else 'Night' end as time_of_day
FROM
(
SELECT extract (time from order_purchase_timestamp) as order_time
FROM `target.orders`
)
)
GROUP BY time_of_day
ORDER BY no_of_order_place desc
```



Row	time_of_day	no_of_order_place
1	Night	40593
2	Afternoon	32370
3	Morning	21738
4	Dawn	4740

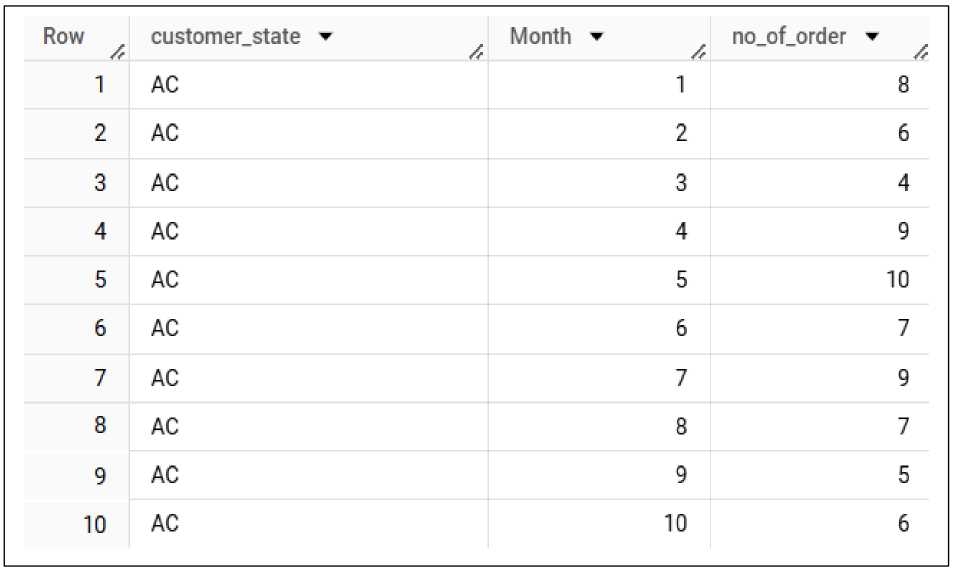
Observation: Significant portion of the customers place order during **'Night'** and **'Afternoon'** time which suggest majority portion of the customers are working professionals who ordered after working hours after 6:00PM or during lunch break in the **'Afternoon'**.

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

a) Month on Month no of orders placed in each state:

Query:

```
SELECT c.customer_state,
extract (month from o.order_purchase_timestamp) as Month,
count (*) as no_of_order
FROM target.customers as c join target.orders as o
ON c.customer_id=o.customer_id
GROUP BY c.customer_state, Month
ORDER BY c.customer_state, Month
```

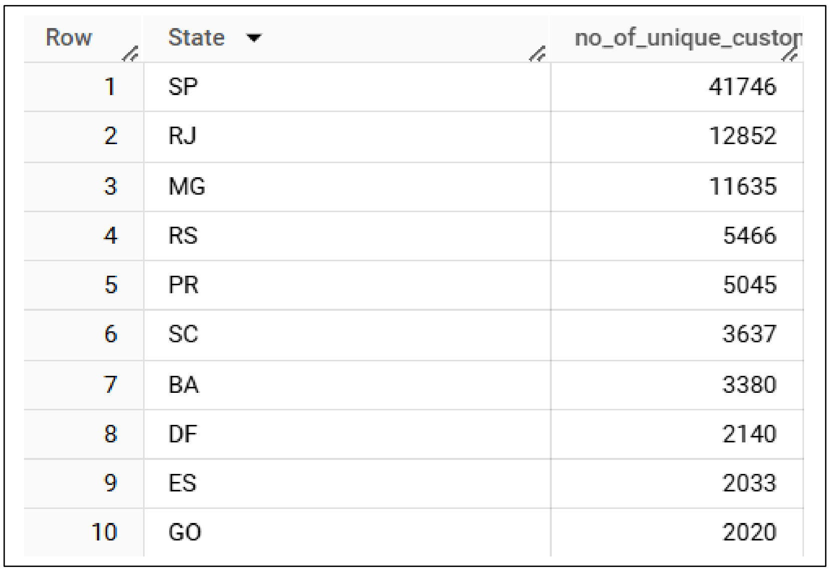


Row	customer_state	Month	no_of_order
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

b) No of Unique customers distributed across all the states:

Query:

```
SELECT customer_state as State, count (distinct customer_id) as
no_of_unique_customers
FROM `target.customers`
GROUP BY State
ORDER BY no_of_unique_customers desc
```



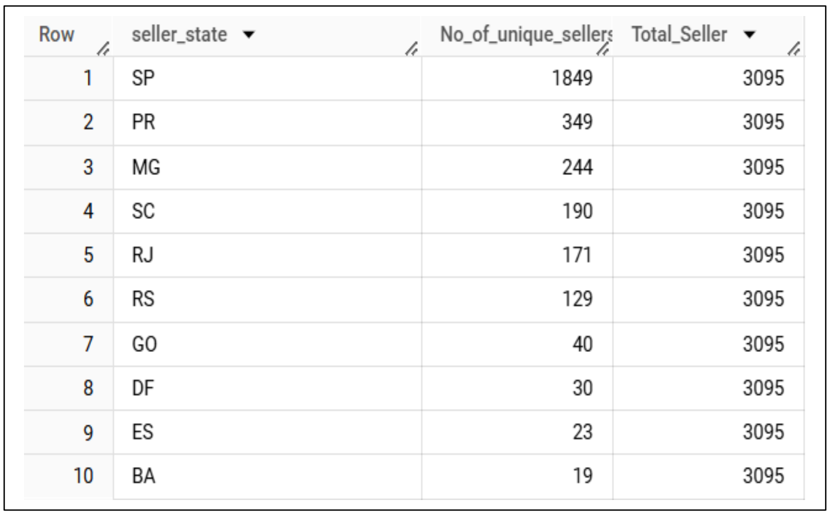
Row	State	no_of_unique_custon
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Observation: About **42%** unique customers are from the state **'SP'** followed by **24%** unique customers which are from the states **'RJ'** followed by **'MG'**.

c) No of Unique Seller distributed across all the states:

Query:

```
WITH t as
(
SELECT seller_state, count(distinct seller_id) as No_of_sellers
FROM target.sellers
GROUP BY seller_state
ORDER BY NO_of_sellers desc
)
SELECT *, SUM(No_of_sellers) OVER () AS Total_Seller
FROM t
ORDER BY t.No_of_sellers desc
```



Row	seller_state	No_of_unique_sellers	Total_Seller
1	SP	1849	3095
2	PR	349	3095
3	MG	244	3095
4	SC	190	3095
5	RJ	171	3095
6	RS	129	3095
7	GO	40	3095
8	DF	30	3095
9	ES	23	3095
10	BA	19	3095

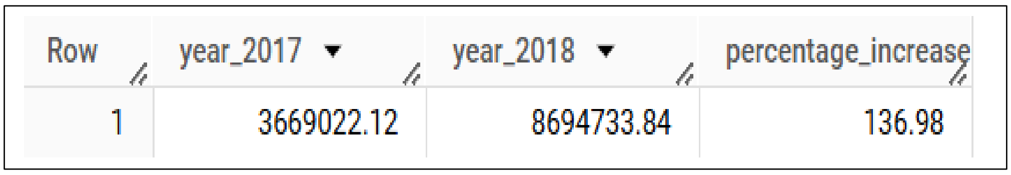
Observation: About **60%** unique sellers are in the state **'SP'** and **19%** sellers belong to the state **'PR'** and **'MG'**.

4) Analysing the money movement by e-commerce:

a) % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

Query:

```
WITH MonthlyTotals AS (
  SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month,
    SUM(p.payment_value) AS total_payment_value
  FROM
    target.payments AS p
  JOIN
    target.orders AS o
  ON
    p.order_id = o.order_id
  WHERE
    EXTRACT(YEAR FROM o.order_purchase_timestamp) IN (2017, 2018)
    AND EXTRACT(MONTH FROM o.order_purchase_timestamp) IN (1, 2, 3,
4, 5, 6, 7, 8)
  GROUP BY
    Year, Month
),
YearlyTotals AS (
  SELECT
    Year,
    SUM(total_payment_value) AS total_yearly
  FROM
    MonthlyTotals
  GROUP BY
    Year
)
SELECT
  y2017.total_yearly AS year_2017,
  y2018.total_yearly AS year_2018,
  ((y2018.total_yearly - y2017.total_yearly) * 100 /
y2017.total_yearly) AS percentage_increase
FROM
  (SELECT total_yearly FROM YearlyTotals WHERE Year = 2017) AS
y2017,
  (SELECT total_yearly FROM YearlyTotals WHERE Year = 2018) AS
y2018;
```



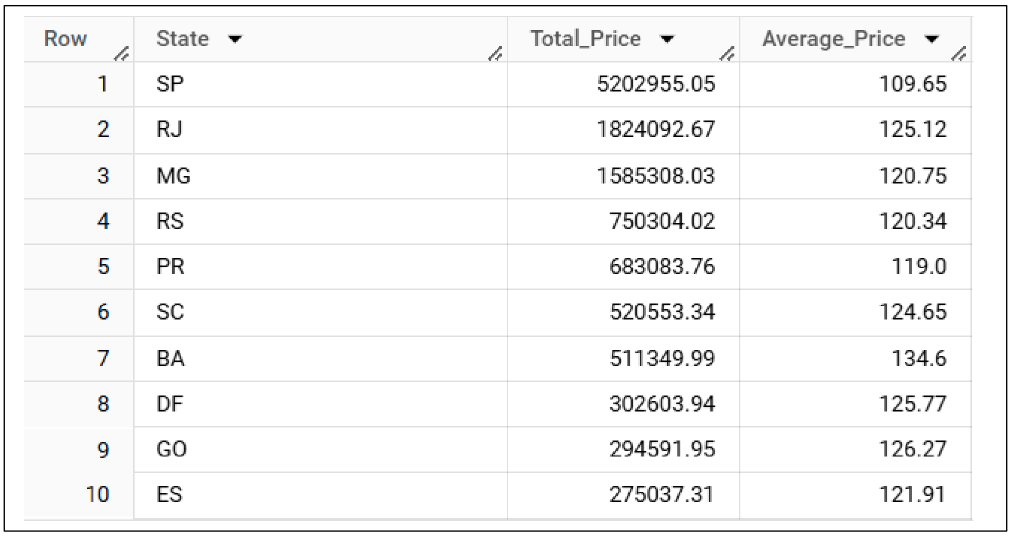
Row	year_2017	year_2018	percentage_increase
1	3669022.12	8694733.84	136.98

Observation: There is **136.98%** increase in cost of order from the year **2017 to 2018**.

b) Total & Average value of order price for each state:

Query:

```
SELECT customer_state as State, round(sum(price),2) as Total_Price,
round(AVG(price),2) as Average_Price
FROM
(
SELECT ot.order_id, ot.order_item_id, ot.price, c.customer_state
FROM target.order_items as ot join target.orders as o ON
ot.order_id=o.order_id
join target.customers as c on o.customer_id=c.customer_id
)
GROUP BY State
ORDER BY Total_Price desc
```

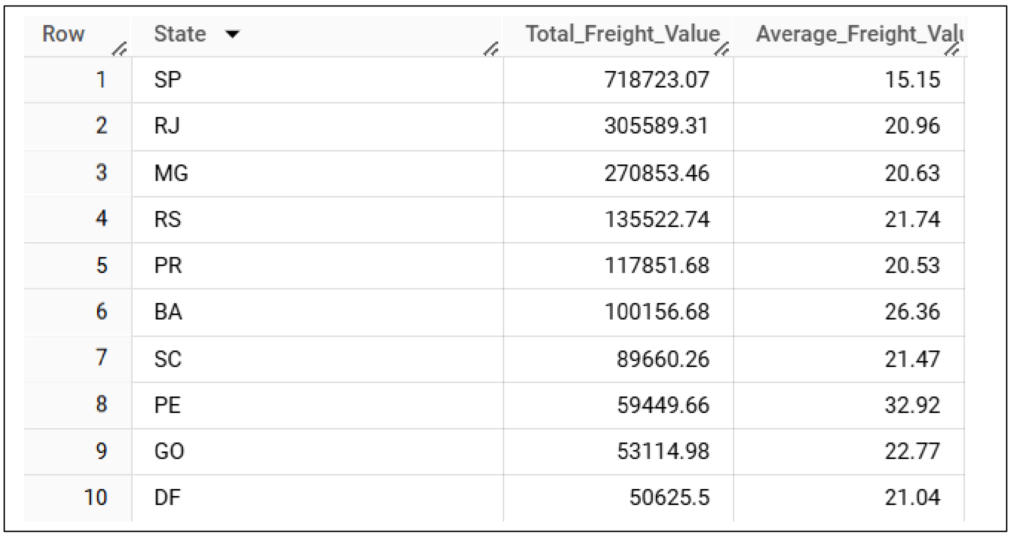


Row	State	Total_Price	Average_Price
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65
7	BA	511349.99	134.6
8	DF	302603.94	125.77
9	GO	294591.95	126.27
10	ES	275037.31	121.91

c) Total & Average value of order freight for each state:

Query:

```
SELECT customer_state as State, round(sum(freight_value),2) as
Total_Freight_Value, round(AVG(freight_value),2) as
Average_Freight_Value
FROM
(
SELECT ot.order_id, ot.order_item_id, ot.freight_value,
c.customer_state
FROM target.order_items as ot join target.orders as o ON
ot.order_id=o.order_id
join target.customers as c on o.customer_id=c.customer_id
)
GROUP BY State
ORDER BY Total_Total_Freight_Value desc
```



Row	State	Total_Freight_Value	Average_Freight_Valu
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04

5) Analysis based on sales, freight and delivery time:

a) No of days taken to deliver each order from the order's purchase date to actual delivery time.

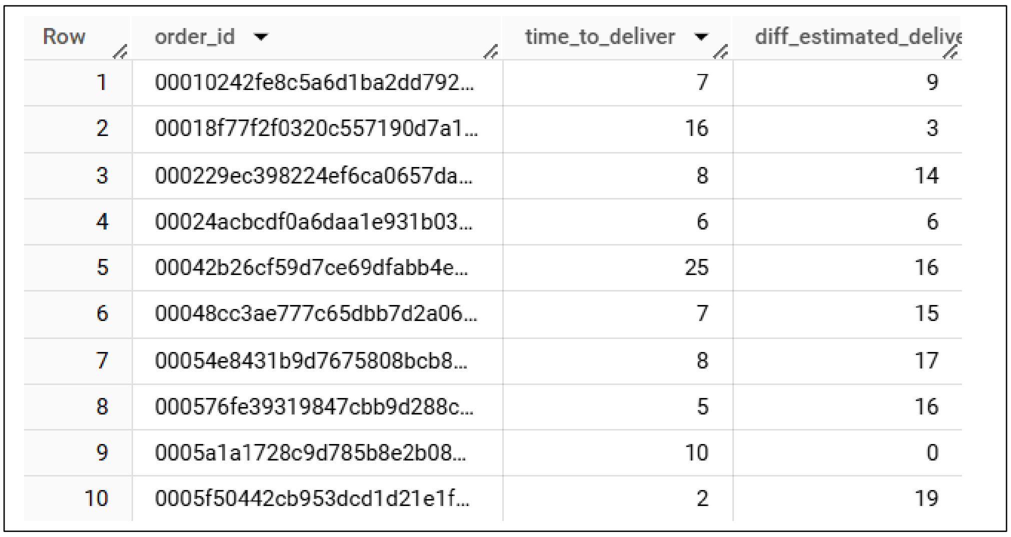
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

Query:

```
SELECT distinct order_id,

DATE_DIFF(date(order_delivered_customer_date),
date(order_purchase_timestamp), DAY) as
time_to_deliver,DATE_DIFF(date(order_estimated_delivery_date),
date(order_delivered_customer_date), DAY) as diff_estimated_delivery

FROM target.orders
WHERE order_status='delivered'
ORDER BY order_id;
```

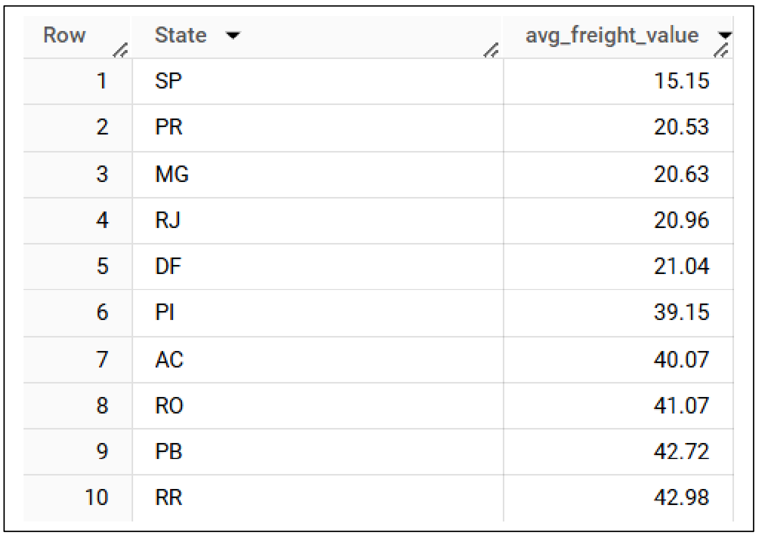


Row	order_id	time_to_deliver	diff_estimated_delive
1	00010242fe8c5a6d1ba2dd792...	7	9
2	00018f77f2f0320c557190d7a1...	16	3
3	000229ec398224ef6ca0657da...	8	14
4	00024acbcdf0a6daa1e931b03...	6	6
5	00042b26cf59d7ce69dfabb4e...	25	16
6	00048cc3ae777c65dbb7d2a06...	7	15
7	00054e8431b9d7675808bcb8...	8	17
8	000576fe39319847cbb9d288c...	5	16
9	0005a1a1728c9d785b8e2b08...	10	0
10	0005f50442cb953dcd1d21e1f...	2	19

b) The top 5 states with the highest & lowest average freight value:

Query:

```
with ranked_state as (
SELECT distinct c.customer_state as State,
round(avg(ot.freight_value),2) as avg_freight_value,
ROW_NUMBER() OVER (ORDER BY ROUND(AVG(ot.freight_value), 2) DESC) AS
rank_desc,
ROW_NUMBER() OVER (ORDER BY ROUND(AVG(ot.freight_value), 2) ASC) AS
rank_asc
FROM target.order_items as ot join target.orders as o ON
ot.order_id=o.order_id
join target.customers as c on o.customer_id=c.customer_id
GROUP BY State
)
SELECT State, avg_freight_value
FROM ranked_state
WHERE rank_asc<=5 or rank_desc<=5
ORDER BY avg_freight_value
```

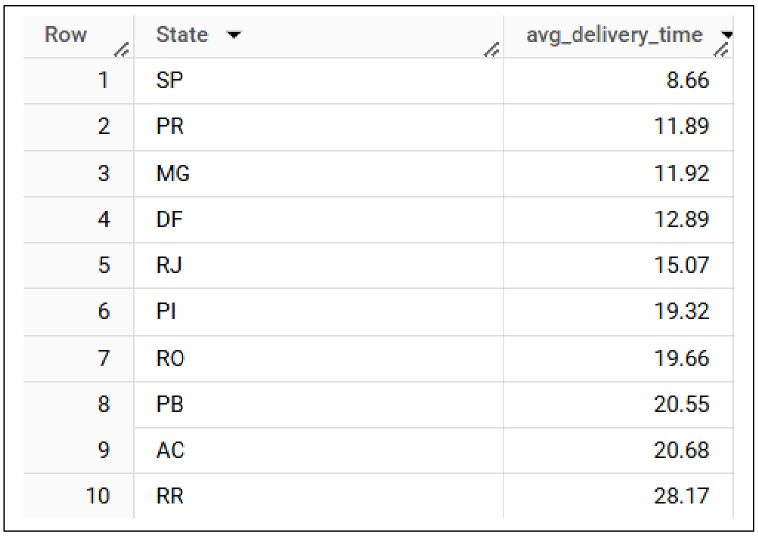


Row	State	avg_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04
6	PI	39.15
7	AC	40.07
8	RO	41.07
9	PB	42.72
10	RR	42.98

c) The top 5 states with the highest & lowest average delivery time:

Query:

```
with ranked_state as (
SELECT distinct c.customer_state as
State,round(avg(DATE_DIFF(date(o.order_delivered_customer_date),
date(o.order_purchase_timestamp), DAY)),2) as avg_delivery_time,
ROW_NUMBER() OVER (ORDER BY ROUND(AVG(ot.freight_value), 2) DESC) AS
rank_desc,
ROW_NUMBER() OVER (ORDER BY ROUND(AVG(ot.freight_value), 2) ASC) AS
rank_asc
FROM target.order_items as ot join target.orders as o ON
ot.order_id=o.order_id
join target.customers as c on o.customer_id=c.customer_id
GROUP BY State
)
SELECT State, avg_delivery_time
FROM ranked_state
WHERE rank_asc<=5 or rank_desc<=5
ORDER BY avg_delivery_time
```

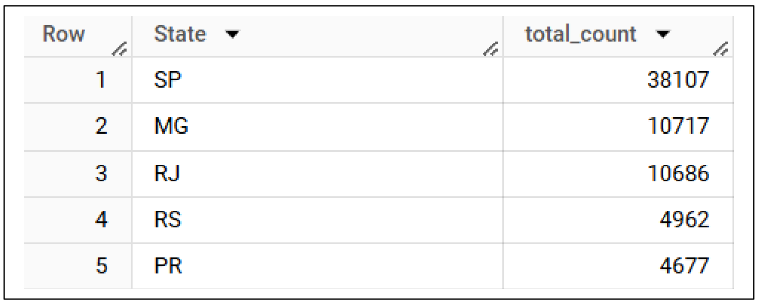


Row	State	avg_delivery_time
1	SP	8.66
2	PR	11.89
3	MG	11.92
4	DF	12.89
5	RJ	15.07
6	PI	19.32
7	RO	19.66
8	PB	20.55
9	AC	20.68
10	RR	28.17

d) Top 5 states where the order delivery is really fast as compared to the estimated date of delivery:

Query:

```
SELECT State, count(*) as total_count
FROM
(
SELECT c.customer_state as State,
date(o.order_delivered_customer_date) as
Delivery_date,date(o.order_estimated_delivery_date) as
Estimated_delivery_date
FROM target.customers as c join target.orders as o
ON c.customer_id=o.customer_id
WHERE o.order_status='delivered' and
date(o.order_delivered_customer_date)<date(o.order_estimated_deliver
y_date)
)
GROUP BY State
ORDER BY total_count desc
LIMIT 5
```



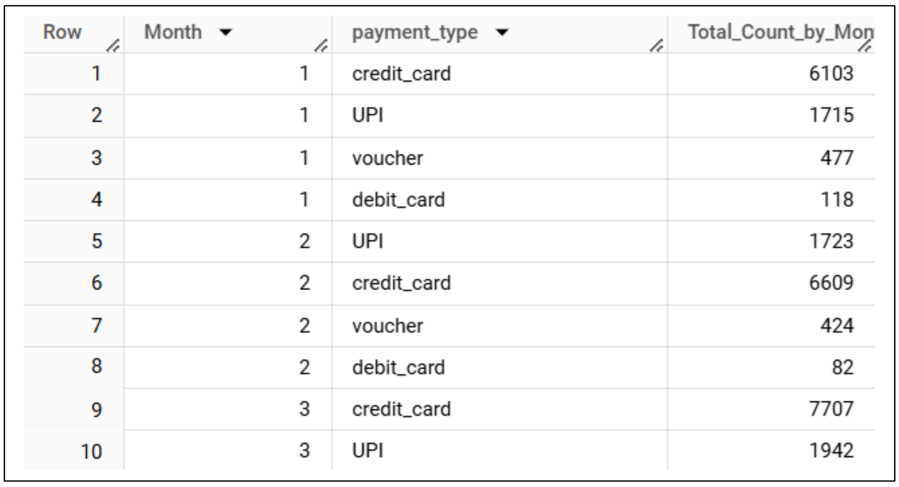
Row	State	total_count
1	SP	38107
2	MG	10717
3	RJ	10686
4	RS	4962
5	PR	4677

6) Analysis based on the payments:

a) The month on month no. of orders placed using different payment types:

Query:

```
SELECT  extract(month from date(o.order_purchase_timestamp)) as
Month, p.payment_type, count(*) as Total_Count_by_Month
FROM target.orders as o join target.payments as p
ON o.order_id=p.order_id
GROUP BY Month, payment_type
ORDER BY Month
```

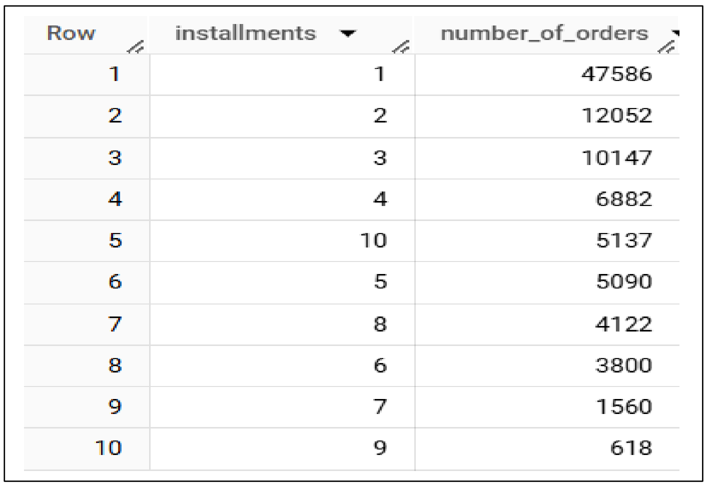


Row	Month	payment_type	Total_Count_by_Mon
1	1	credit_card	6103
2	1	UPI	1715
3	1	voucher	477
4	1	debit_card	118
5	2	UPI	1723
6	2	credit_card	6609
7	2	voucher	424
8	2	debit_card	82
9	3	credit_card	7707
10	3	UPI	1942

b) The no of orders placed on the basis of the payment instalments that have been paid:

Query:

```
SELECT p.payment_installments AS installments, COUNT(DISTINCT
p.order_id) AS number_of_orders
FROM target.payments AS p JOIN target.orders AS o
ON p.order_id = o.order_id
WHERE o.order_status = 'delivered' AND p.payment_installments!=0
GROUP BY p.payment_installments
ORDER BY number_of_orders desc
```



Row	installments	number_of_orders
1	1	47586
2	2	12052
3	3	10147
4	4	6882
5	10	5137
6	5	5090
7	8	4122
8	6	3800
9	7	1560
10	9	618