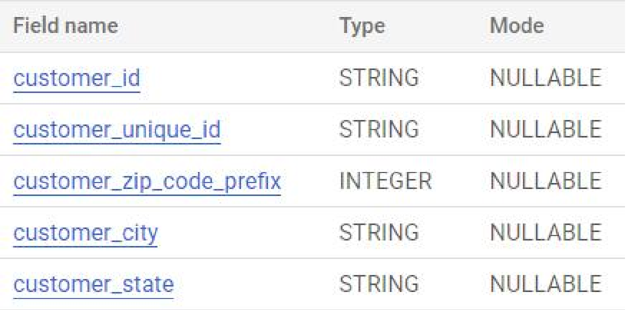
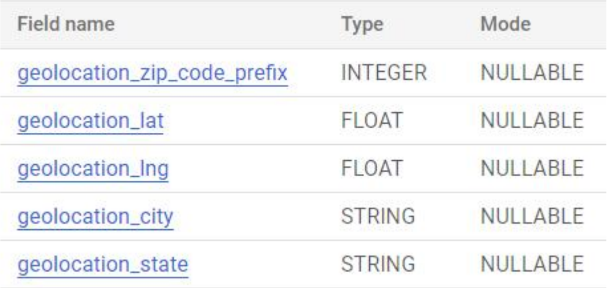
Que1.1 Data type of columns in Table



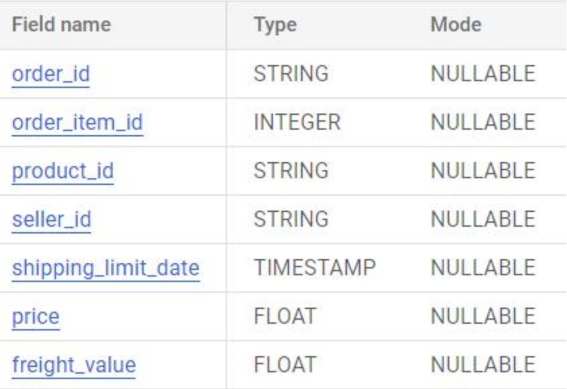
Field name	Type	Mode
customer_id	STRING	NULLABLE
customer_unique_id	STRING	NULLABLE
customer_zip_code_prefix	INTEGER	NULLABLE
customer_city	STRING	NULLABLE
customer_state	STRING	NULLABLE

Customers table



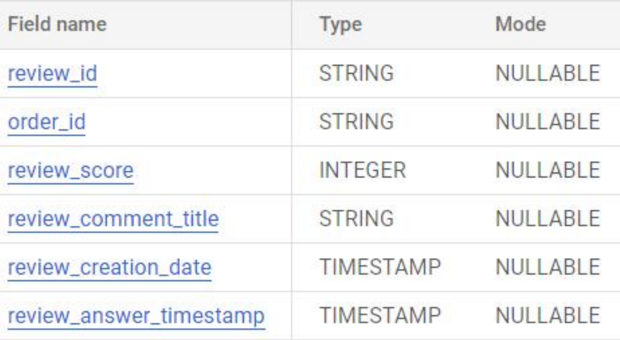
Field name	Type	Mode
geolocation_zip_code_prefix	INTEGER	NULLABLE
geolocation_lat	FLOAT	NULLABLE
geolocation_lng	FLOAT	NULLABLE
geolocation_city	STRING	NULLABLE
geolocation_state	STRING	NULLABLE

Geolocation table



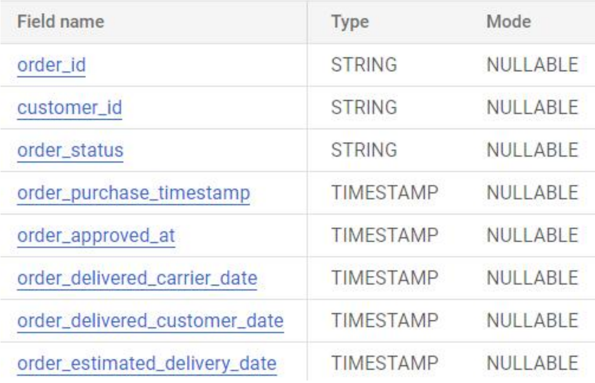
Field name	Type	Mode
order_id	STRING	NULLABLE
order_item_id	INTEGER	NULLABLE
product_id	STRING	NULLABLE
seller_id	STRING	NULLABLE
shipping_limit_date	TIMESTAMP	NULLABLE
price	FLOAT	NULLABLE
freight_value	FLOAT	NULLABLE

Order_items



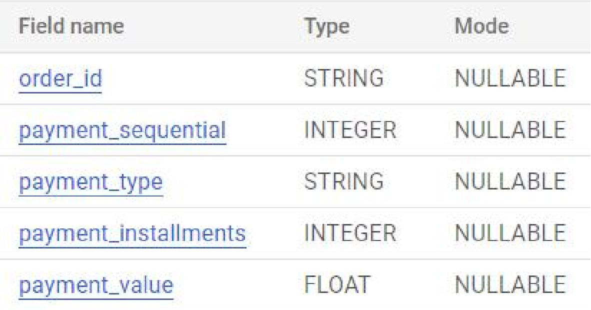
Field name	Type	Mode
review_id	STRING	NULLABLE
order_id	STRING	NULLABLE
review_score	INTEGER	NULLABLE
review_comment_title	STRING	NULLABLE
review_creation_date	TIMESTAMP	NULLABLE
review_answer_timestamp	TIMESTAMP	NULLABLE

Order_reviews



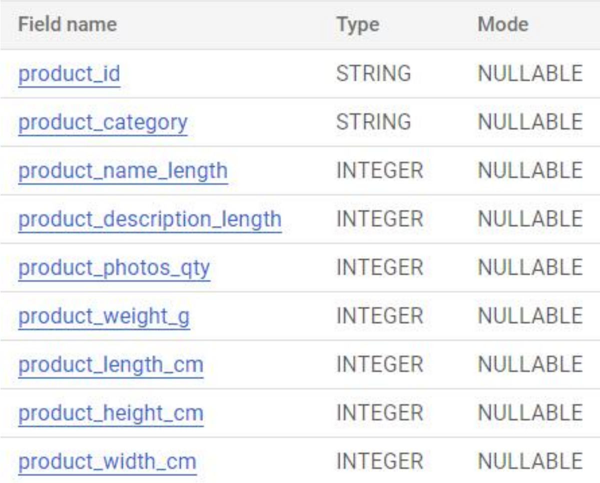
Field name	Type	Mode
order_id	STRING	NULLABLE
customer_id	STRING	NULLABLE
order_status	STRING	NULLABLE
order_purchase_timestamp	TIMESTAMP	NULLABLE
order_approved_at	TIMESTAMP	NULLABLE
order_delivered_carrier_date	TIMESTAMP	NULLABLE
order_delivered_customer_date	TIMESTAMP	NULLABLE
order_estimated_delivery_date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

Orders



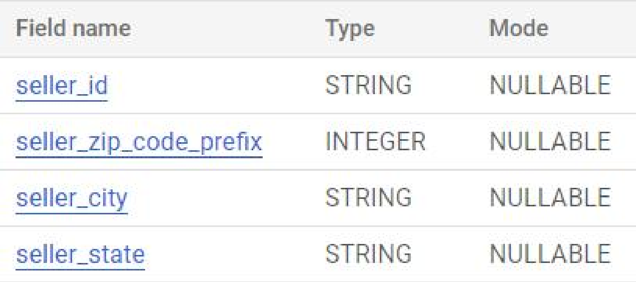
Field name	Type	Mode
order_id	STRING	NULLABLE
payment_sequential	INTEGER	NULLABLE
payment_type	STRING	NULLABLE
payment_installments	INTEGER	NULLABLE
payment_value	FLOAT	NULLABLE

Payments



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

Products



Field name	Type	Mode
seller_id	STRING	NULLABLE
seller_zip_code_prefix	INTEGER	NULLABLE
seller_city	STRING	NULLABLE
seller_state	STRING	NULLABLE

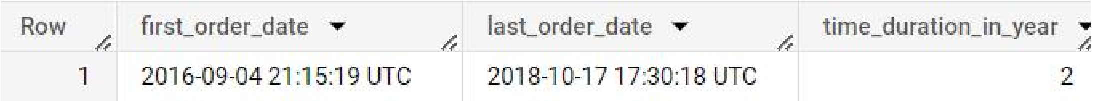
Sellers

Que 1.2 Time period for which the data given

Query:

```
   select min(order_purchase_timestamp) as first_order_date,
      max(order_purchase_timestamp) as last_order_date,
      DATE_DIFF(EXTRACT(DATE FROM (max(order_purchase_timestamp))),
                EXTRACT(DATE FROM(min(order_purchase_timestamp))),
      YEAR) as time_duration_in_year
     from `trgt_dataset234.orders`
```

Output Table:



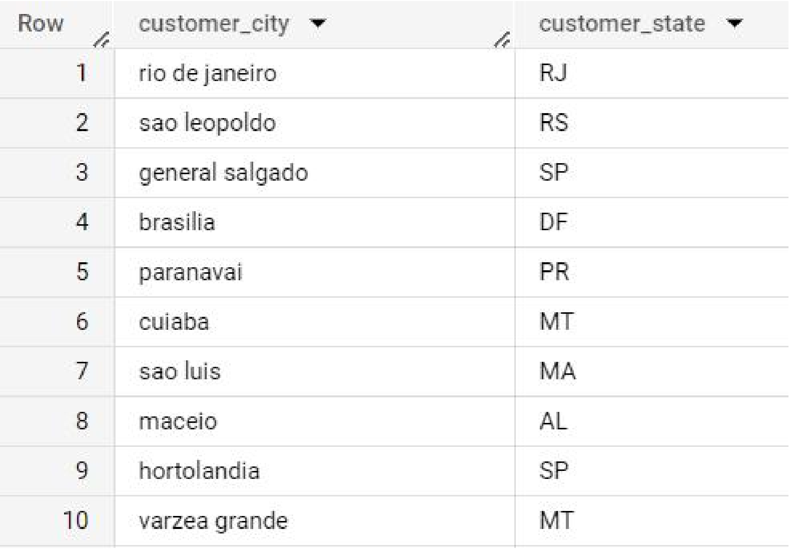
Row	first_order_date	last_order_date	time_duration_in_year
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC	2

Que 1.3 Cities and States of customers ordered during the given period

Query:

```
    SELECT distinct customer_city, customer_state
    from `trgt_dataset234.customers` ct
    join `trgt_dataset234.orders` od
    on ct.customer_id= od.customer_id
    where od.order_purchase_timestamp between "2016-09-04 21:15:19 UTC"
                                        and "2018-10-17 17:30:18 UTC"
```

Output Table:



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

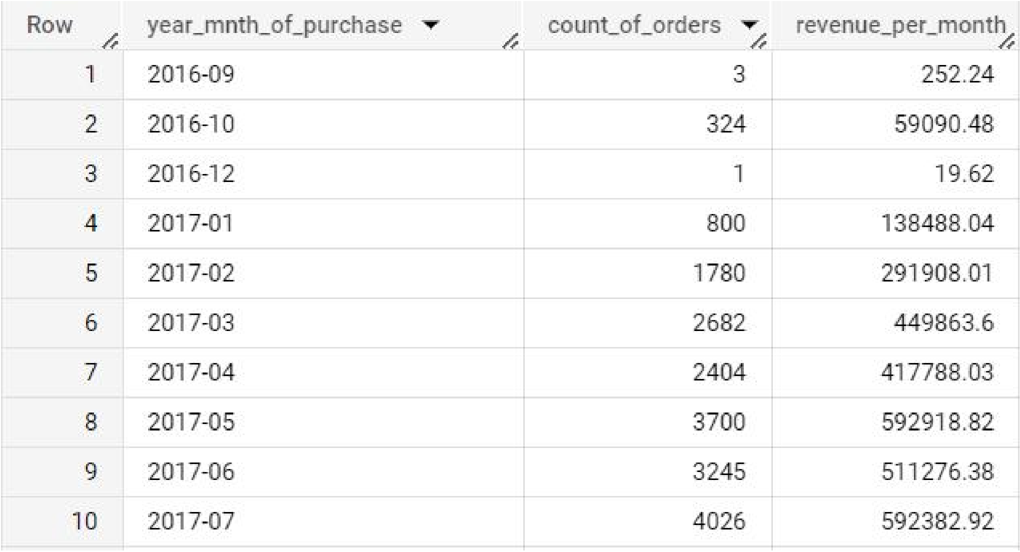
Que 2.1 Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

Query:

```
with purchase_per_month as
       (select order_id,customer_id,
        format_date("%Y-%m", date(order_purchase_timestamp)) as
        year_mnth_of_purchase
        from `trgt_dataset234.orders` ),

   payment_per_order as (SELECT order_id,
                         round(sum(payment_value),2) as total_payment
                         FROM `sclr-sql-project1.trgt_dataset234.payments`
                         group  by order_id)
    SELECT prm.year_mnth_of_purchase,
           count(prm.order_id) as count_of_orders,
           round(sum(po.total_payment),2) as revenue_per_month
   FROM purchase_per_month prm
   join payment_per_order po
   on prm.order_id=po.order_id
   group by prm.year_mnth_of_purchase
   order by prm.year_mnth_of_purchase ,count_of_orders desc,
    revenue_per_month desc
```

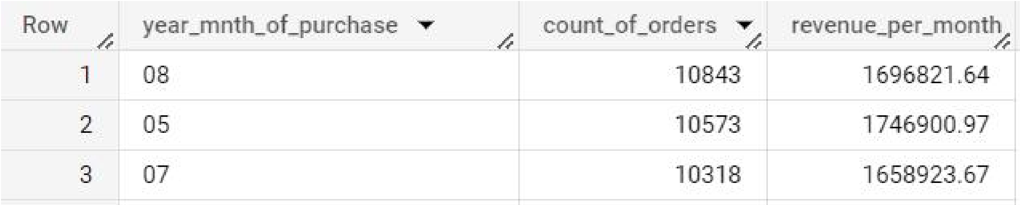
Output Table:



Row	year_mnth_of_purchase	count_of_orders	revenue_per_month
1	2016-09	3	252.24
2	2016-10	324	59090.48
3	2016-12	1	19.62
4	2017-01	800	138488.04
5	2017-02	1780	291908.01
6	2017-03	2682	449863.6
7	2017-04	2404	417788.03
8	2017-05	3700	592918.82
9	2017-06	3245	511276.38
10	2017-07	4026	592382.92

Insight/Recommendation :

- Peak seasonality seen in May,July, and August



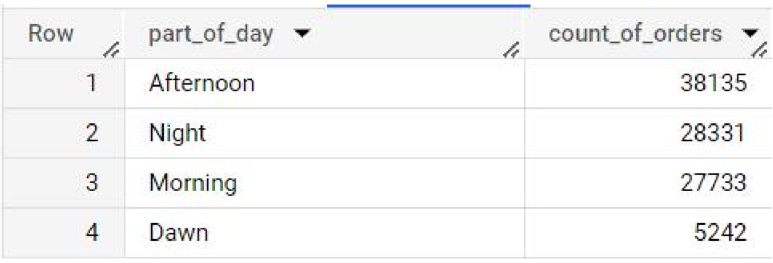
Row	year_mnth_of_purchase	count_of_orders	revenue_per_month
1	08	10843	1696821.64
2	05	10573	1746900.97
3	07	10318	1658923.67

Que 2.2 What time do brazillian customers tends to buy?

Query:

```
    with cte as (select *, extract(hour from order_purchase_timestamp)as hrs
            from `trgt_dataset234.orders`)
     select case
         when hrs between 0 and 6 then "Dawn"
         when hrs between 7 and 12 then "Morning"
         when hrs between 13 and 18 then "Afternoon"
         when hrs between 19 and 23 then "Night"
      end as part_of_day,
      count(order_id) as count_of_orders
    from cte
    group by part_of_day
    order by count_of_orders desc
```

Table Output :



Row	part_of_day	count_of_orders
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

Insight/Recommendation:

1. Mostly Brazillian prefer purchasing in Afternoon so it will be good if manpower in the afternoon slightly more than other parts of day.
2. We can shift some manpower from the dawn to the Afternoon because in the dawn there is less traffic Compare to other parts of day.

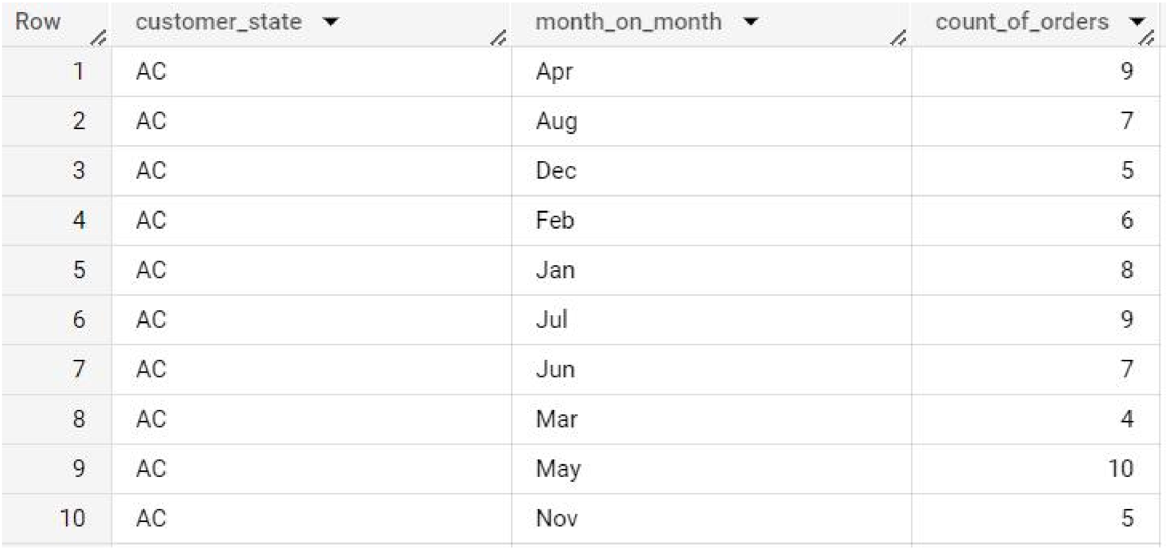
Que 3.1 Get Month on month orders by states

Query:

```
   with cte as (select *,format_date("%b",date(order_purchase_timestamp)) as
                             month_on_month
                from `trgt_dataset234.orders`)

     select customer_state,c.month_on_month,
            count(order_id) count_of_orders
     from cte c
     join `trgt_dataset234.customers` cust
     on c.customer_id= cust.customer_id
     group by customer_state, c.month_on_month
     order by customer_state, c.month_on_month
```

Output Table:



Row	customer_state	month_on_month	count_of_orders
1	AC	Apr	9
2	AC	Aug	7
3	AC	Dec	5
4	AC	Feb	6
5	AC	Jan	8
6	AC	Jul	9
7	AC	Jun	7
8	AC	Mar	4
9	AC	May	10
10	AC	Nov	5

Insight/Recommendation:

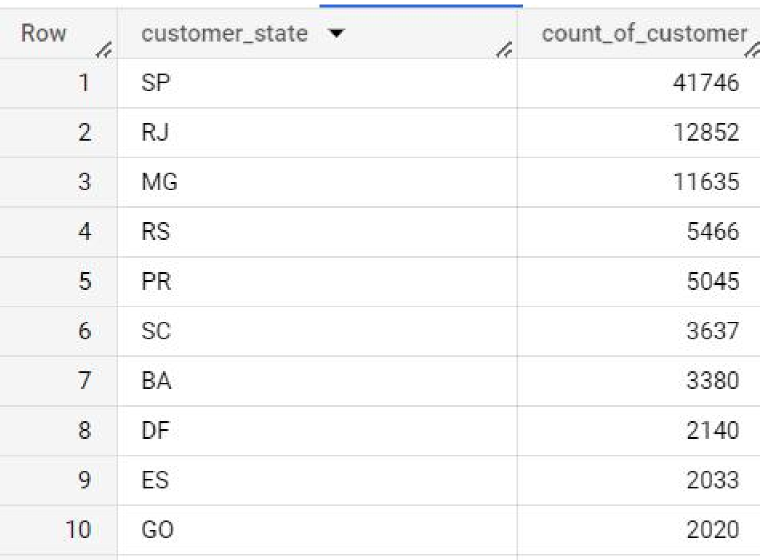
- From 2016 to 2018 data in month of May, July, and August there is growth in orders

Que 3.2 Distribution of customers across the states in Brazil

Query:

```
    select customer_state, count(distinct customer_id) as count_of_customer
      from `trgt_dataset234.customers`
      group by customer_state
      order by count_of_customer desc
```

Output Table:



Row	customer_state	count_of_customer
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

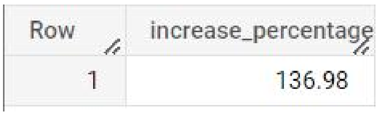
Que 4.1 Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

Query:

```
    with cte1 as (select od.order_id,
                         extract(year from order_purchase_timestamp) as yer,
                         extract(month from order_purchase_timestamp)as mnth,
                         payment_value
                   from `trgt_dataset234.orders` od
                   join `trgt_dataset234.payments` pay
                   on od.order_id=pay.order_id)

      select round(((sum(case when yer=2018
                                   and mnth between 1 and 8 then payment_value end)
         - sum(case when yer=2017 and mnth between 1 and 8 then payment_value end))
          /sum(case when yer=2017 and mnth between 1 and 8 then payment_value
           end))*100 ,2)
          as increase_percentage
      from cte1
```

Output Table:



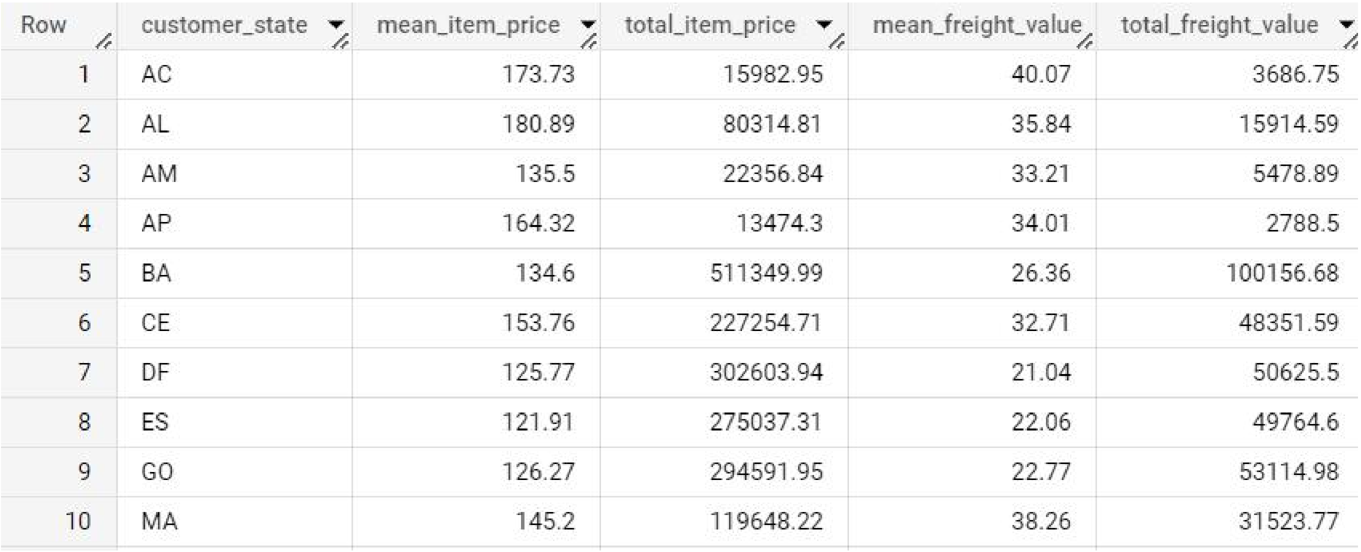
Row	increase_percentage
1	136.98

Que 4.2 Mean & Sum of price and freight value by customer state

Query:

```
with cte as (select cust.customer_id, cust.customer_state,
                      itm.price, itm.freight_value
             from `trgt_dataset234.customers` cust
             join `trgt_dataset234.orders` ord
             on cust.customer_id=ord.customer_id
             join `trgt_dataset234.order_items` itm
             on ord.order_id=itm.order_id)
  select customer_state, round(avg(price),2) as mean_item_price,
         round(sum(price),2) as total_item_price,
         round(avg(freight_value),2) as mean_freight_value,
         round(sum(freight_value),2) as total_freight_value
  from cte
  group by customer_state
  order by customer_state
```

Output Table:



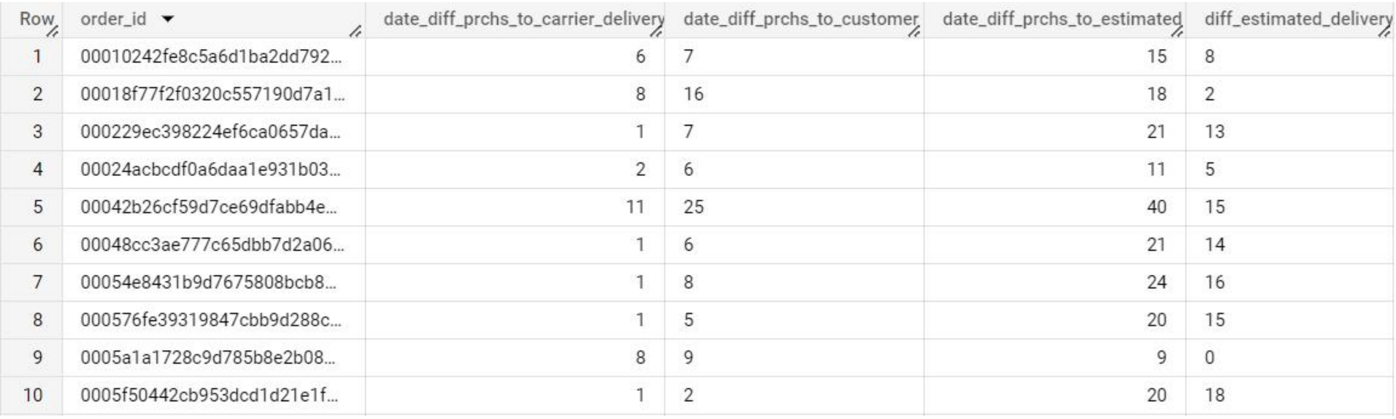
Row	customer_state	mean_item_price	total_item_price	mean_freight_value	total_freight_value
1	AC	173.73	15982.95	40.07	3686.75
2	AL	180.89	80314.81	35.84	15914.59
3	AM	135.5	22356.84	33.21	5478.89
4	AP	164.32	13474.3	34.01	2788.5
5	BA	134.6	511349.99	26.36	100156.68
6	CE	153.76	227254.71	32.71	48351.59
7	DF	125.77	302603.94	21.04	50625.5
8	ES	121.91	275037.31	22.06	49764.6
9	GO	126.27	294591.95	22.77	53114.98
10	MA	145.2	119648.22	38.26	31523.77

Que 5.1 Calculate days between purchasing, delivering and estimated delivery

Query:

```
select order_id,
       date_diff(order_delivered_carrier_date, order_purchase_timestamp, day) as
       date_diff_prchs_to_carrier_delivery,
     if(order_delivered_customer_date is null or order_purchase_timestamp is
      null , "N/A",
     cast(date_diff(order_delivered_customer_date, order_purchase_timestamp,
     day)as string)) as date_diff_prchs_to_customer,
     date_diff(order_estimated_delivery_date, order_purchase_timestamp, day) as
     date_diff_prchs_to_estimated,
         if(order_estimated_delivery_date is null or
          order_delivered_customer_date is null , "N/A",
     cast(date_diff(order_estimated_delivery_date, order_delivered_customer_date,
      day)as string)) as diff_estimated_delivery
from `trgt_dataset234.orders`
order by order_id
```

Output Table:



Row	order_id	date_diff_prchs_to_carrier_delivery	date_diff_prchs_to_customer	date_diff_prchs_to_estimated	diff_estimated_delivery
1	00010242fe8c5a6d1ba2dd792...	6	7	15	8
2	00018f77f2f0320c557190d7a1...	8	16	18	2
3	000229ec398224ef6ca0657da...	1	7	21	13
4	00024acbcdf0a6daa1e931b03...	2	6	11	5
5	00042b26cf59d7ce69dfabb4e...	11	25	40	15
6	00048cc3ae777c65dbb7d2a06...	1	6	21	14
7	00054e8431b9d7675808bcb8...	1	8	24	16
8	000576fe39319847cbb9d288c...	1	5	20	15
9	0005a1a1728c9d785b8e2b08...	8	9	9	0
10	0005f50442cb953dcd1d21e1f...	1	2	20	18

Que 5.2 Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

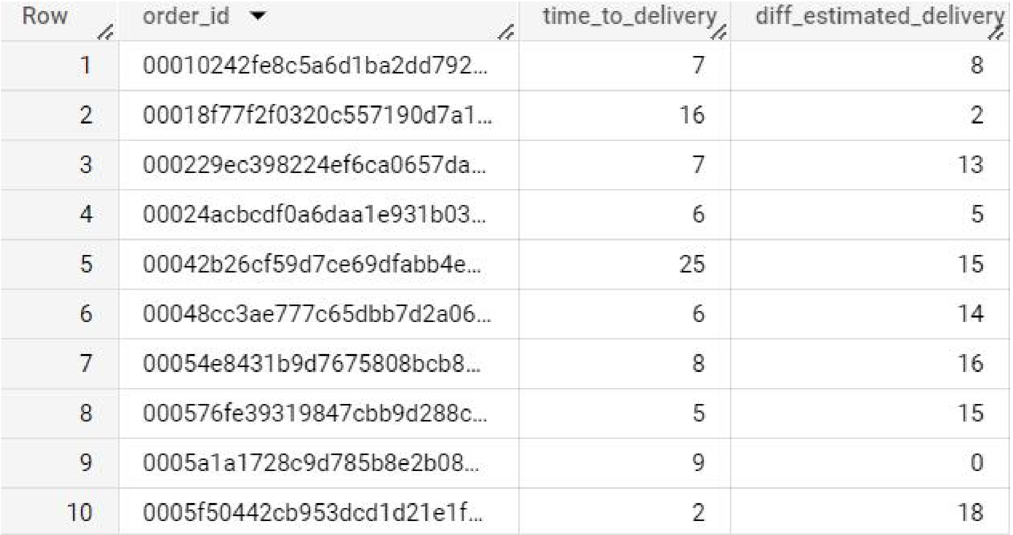
time_to_delivery = order_delivered_customer_date - order_purchase_timestamp

diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date

Query :

```
select order_id,
       date_diff(order_delivered_customer_date,
                 order_purchase_timestamp, day)as time_to_delivery,
       date_diff(order_estimated_delivery_date,
             order_delivered_customer_date, day) as diff_estimated_delivery
 from `trgt_dataset234.orders`
 order by order_id
```

Output Table:



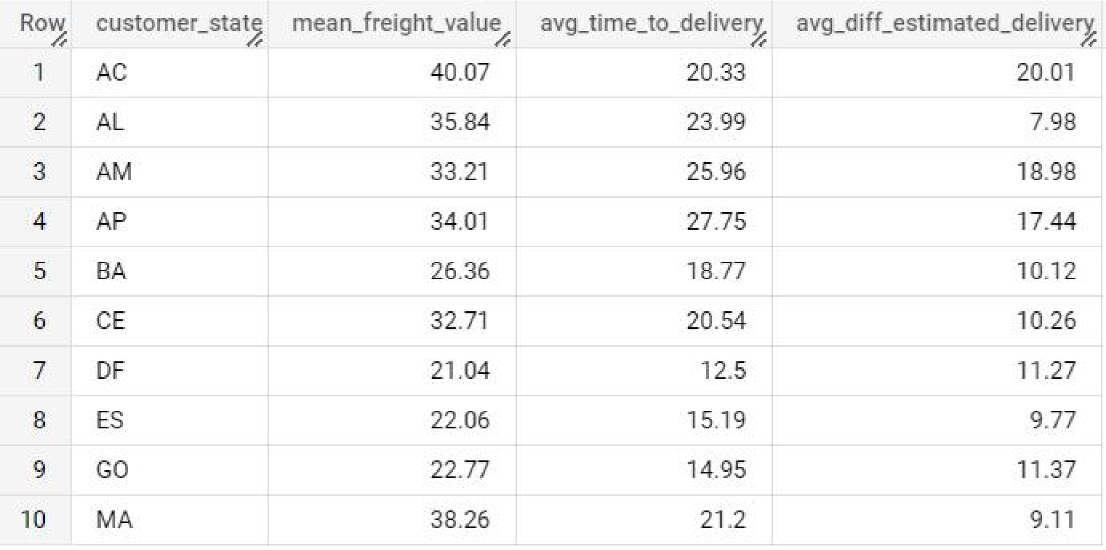
Row	order_id	time_to_delivery	diff_estimated_delivery
1	00010242fe8c5a6d1ba2dd792...	7	8
2	00018f77f2f0320c557190d7a1...	16	2
3	000229ec398224ef6ca0657da...	7	13
4	00024acbcdf0a6daa1e931b03...	6	5
5	00042b26cf59d7ce69dfabb4e...	25	15
6	00048cc3ae777c65dbb7d2a06...	6	14
7	00054e8431b9d7675808bcb8...	8	16
8	000576fe39319847cbb9d288c...	5	15
9	0005a1a1728c9d785b8e2b08...	9	0
10	0005f50442cb953dcd1d21e1f...	2	18

Que 5.3 Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

Query:

```
   with cte as
         (select cust.customer_state,
                 itm.freight_value,
                 date_diff(order_delivered_customer_date,
                 order_purchase_timestamp, day)as time_to_delivery,
                 date_diff(order_estimated_delivery_date,
                 order_delivered_customer_date, day) as diff_estimated_delivery
             from `trgt_dataset234.customers` cust
             join `trgt_dataset234.orders` ord
             on cust.customer_id=ord.customer_id
             join `trgt_dataset234.order_items` itm
             on ord.order_id=itm.order_id)
select cte.customer_state, round(avg(freight_value),2) as mean_freight_value ,
       round(avg(time_to_delivery),2) as avg_time_to_delivery,
       round(avg(diff_estimated_delivery),2) as avg_diff_estimated_delivery
from cte
group by customer_state
order by customer_state
```

Output Table:



Row	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	AC	40.07	20.33	20.01
2	AL	35.84	23.99	7.98
3	AM	33.21	25.96	18.98
4	AP	34.01	27.75	17.44
5	BA	26.36	18.77	10.12
6	CE	32.71	20.54	10.26
7	DF	21.04	12.5	11.27
8	ES	22.06	15.19	9.77
9	GO	22.77	14.95	11.37
10	MA	38.26	21.2	9.11

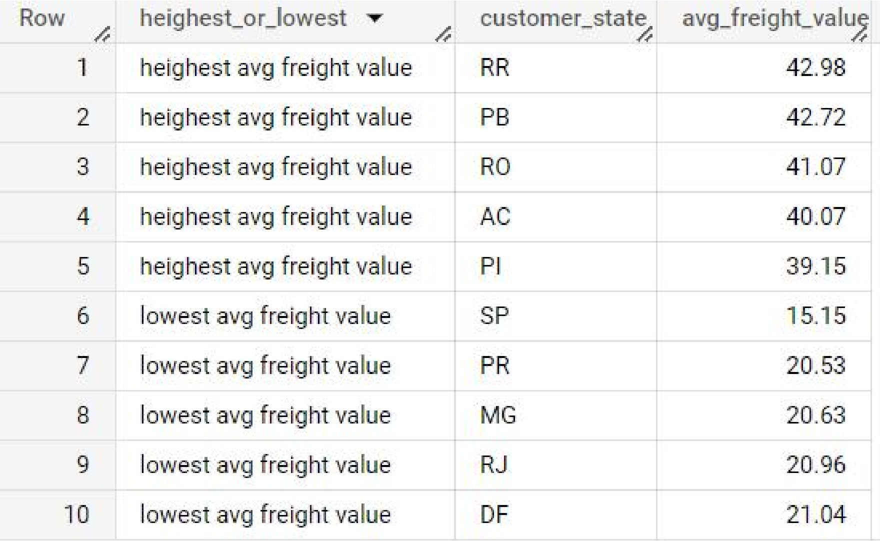
Que 5.4.a Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

Query:

```
    with cte as (select cust.customer_state,
                   itm.freight_value
                 from `trgt_dataset234.customers` cust
                 join `trgt_dataset234.orders` ord
                 on cust.customer_id=ord.customer_id
                 join `trgt_dataset234.order_items` itm
                 on ord.order_id=itm.order_id)

   (select "heighest avg freight value" as heighest_or_lowest,
       customer_state, round(avg(freight_value),2) as avg_freight_value
    from cte
    group by customer_state
    order by avg_freight_value desc
    limit 5)
    union all
   (select "lowest avg freight value" as heighest_or_lowest,
       customer_state, round(avg(freight_value),2) as avg_freight_value
   from cte
   group by customer_state
   order by avg_freight_value asc
   limit 5)
```

Output Table:



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

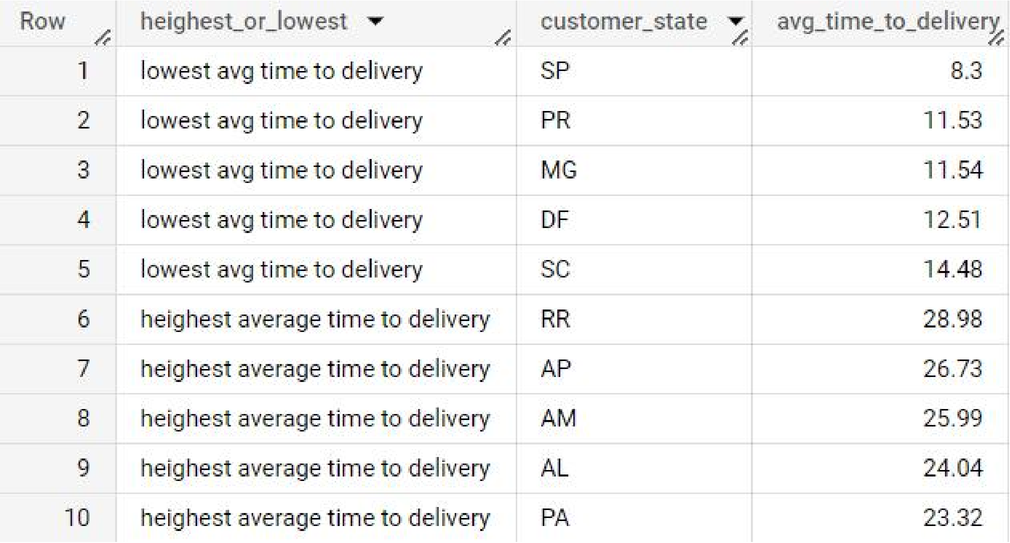
Que 5.4.b Top 5 states with highest/lowest average time to delivery

Query:

```
    with cte1 as
          (select customer_state,
               date_diff(order_delivered_customer_date, order_purchase_timestamp,
               day)as time_to_delivery,
          from `trgt_dataset234.customers` cust
          join `trgt_dataset234.orders` ord
          on cust.customer_id= ord.customer_id)

 (select "lowest avg time to delivery" as heighest_or_lowest,
        customer_state, round(avg(time_to_delivery),2) as avg_time_to_delivery,
  from cte1
  group by customer_state
  order by avg_time_to_delivery asc
  limit 5)
  union all
 (select "heighest average time to delivery" as heighest_or_lowest,
        customer_state, round(avg(time_to_delivery),2) as avg_time_to_delivery,
  from cte1
  group by customer_state
  order by avg_time_to_delivery desc
  limit 5)
```

Output Table:



Row	heighest_or_lowest	customer_state	avg_time_to_delivery
1	lowest avg time to delivery	SP	8.3
2	lowest avg time to delivery	PR	11.53
3	lowest avg time to delivery	MG	11.54
4	lowest avg time to delivery	DF	12.51
5	lowest avg time to delivery	SC	14.48
6	heighest average time to delivery	RR	28.98
7	heighest average time to delivery	AP	26.73
8	heighest average time to delivery	AM	25.99
9	heighest average time to delivery	AL	24.04
10	heighest average time to delivery	PA	23.32

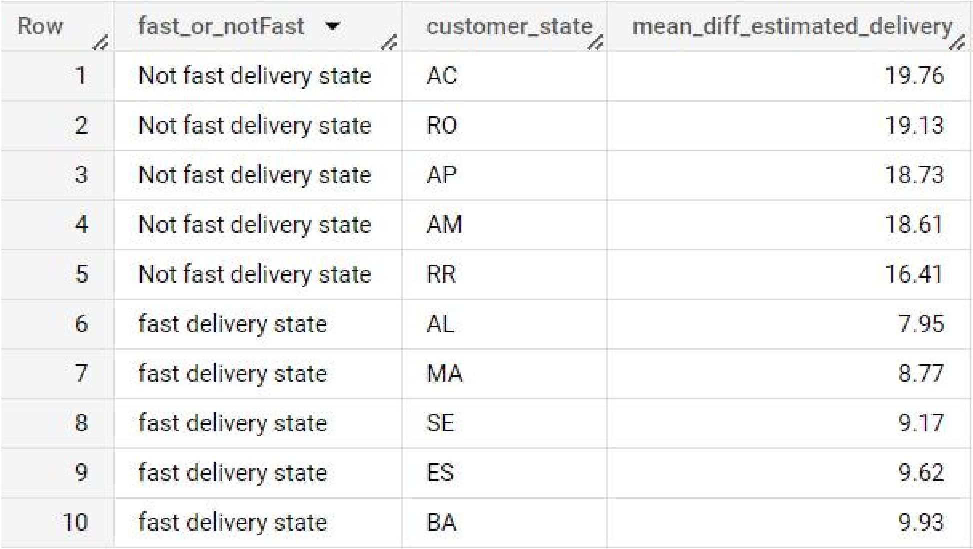
Que 5.4.c Top 5 states where delivery is really fast/ not so fast compared to estimated date

Query:

```
    with cte as
        (select customer_state,
                date_diff(order_estimated_delivery_date,
               order_delivered_customer_date, day) as diff_estimated_delivery,
        from `trgt_dataset234.customers` cust
        join `trgt_dataset234.orders` ord
        on cust.customer_id=ord.customer_id)

   (select "fast delivery state" as fast_or_notFast, customer_state,
        round(avg(diff_estimated_delivery),2) as mean_diff_estimated_delivery
    from cte
    group by customer_state
    order by mean_diff_estimated_delivery asc
    limit 5)
    union all
    (select "Not fast delivery state" as fast_or_notFast,
        customer_state, round(avg(diff_estimated_delivery),2) as
    mean_diff_estimated_delivery
    from cte
    group by customer_state
    order by mean_diff_estimated_delivery desc
    limit 5)
```

Output Table:



Row	fast_or_notFast	customer_state	mean_diff_estimated_delivery
1	Not fast delivery state	AC	19.76
2	Not fast delivery state	RO	19.13
3	Not fast delivery state	AP	18.73
4	Not fast delivery state	AM	18.61
5	Not fast delivery state	RR	16.41
6	fast delivery state	AL	7.95
7	fast delivery state	MA	8.77
8	fast delivery state	SE	9.17
9	fast delivery state	ES	9.62
10	fast delivery state	BA	9.93

Insight/Recommendation :

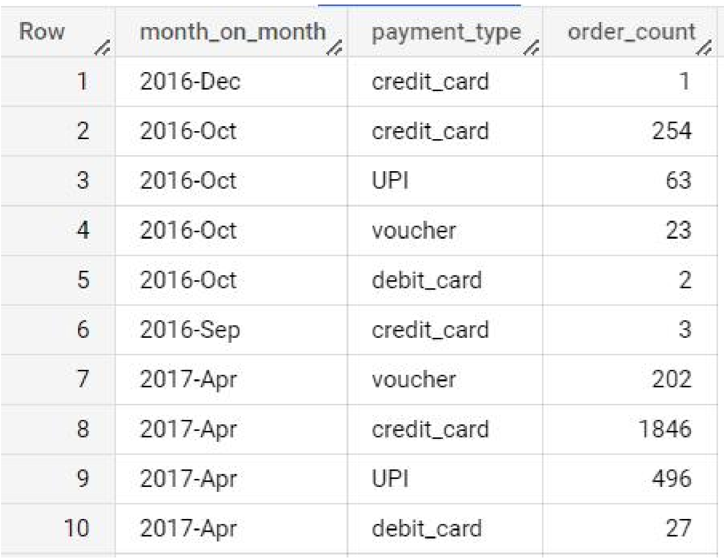
- AC, RO, AP, AM, RR this states are taking more days compare to other states.
- States which are taking more days for delivery this can be reduced by optimizing supply chain.

Que 6.1 Month over Month count of orders for different payment types

Query:

```
with cte as (select od.order_id,
              format_date("%Y-%b",date(order_purchase_timestamp)) as
              month_on_month,
                pay.payment_type
             from `trgt_dataset234.orders` od
             join `trgt_dataset234.payments` pay
         on od.order_id=pay.order_id)
select cte.month_on_month, payment_type ,count(order_id) as order_count
from cte
group by payment_type, cte.month_on_month
order by cte.month_on_month
```

Output Table:



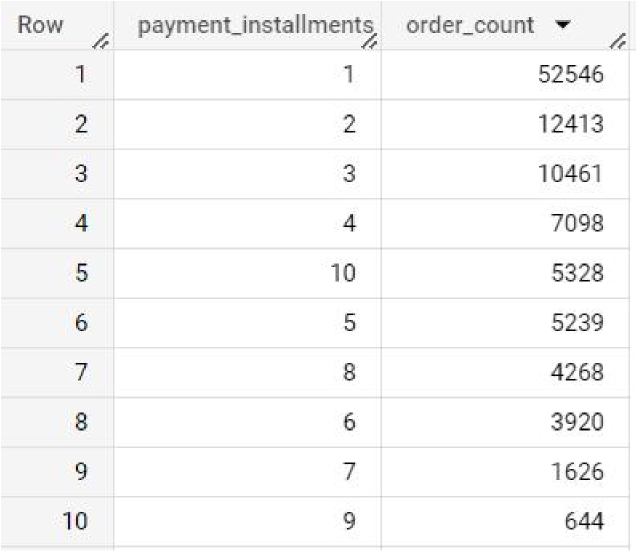
Row	month_on_month	payment_type	order_count
1	2016-Dec	credit_card	1
2	2016-Oct	credit_card	254
3	2016-Oct	UPI	63
4	2016-Oct	voucher	23
5	2016-Oct	debit_card	2
6	2016-Sep	credit_card	3
7	2017-Apr	voucher	202
8	2017-Apr	credit_card	1846
9	2017-Apr	UPI	496
10	2017-Apr	debit_card	27

Que 6.2 Count of orders based on the no. of payment installments

Query:

```
select payment_installments, count(order_id)as order_count
  from `trgt_dataset234.payments`
  group by payment_installments
  order by order_count desc
```

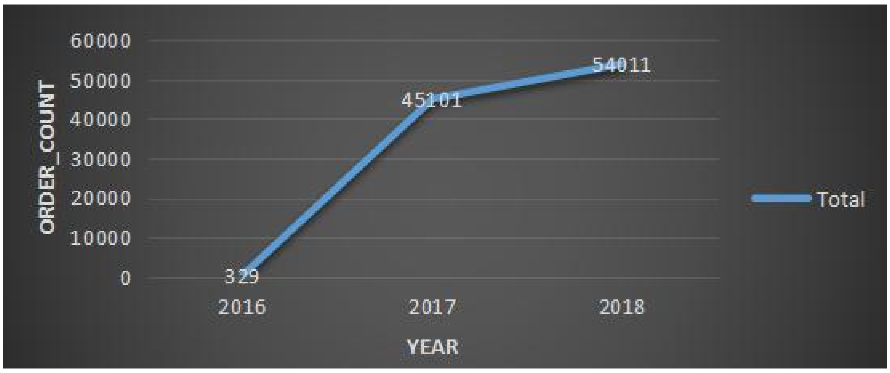
Output Table:



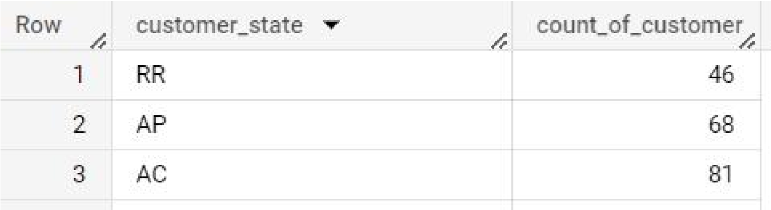
Row	payment_installments	order_count
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920
9	7	1626
10	9	644

Insight/Recommendation :

- There is growing Trend from 2016 to 2018



- To reach to the more customers more advertisement is needed, specially RR, AP, AC this states have less count of customers compare to other states



Row	customer_state	count_of_customer
1	RR	46
2	AP	68
3	AC	81