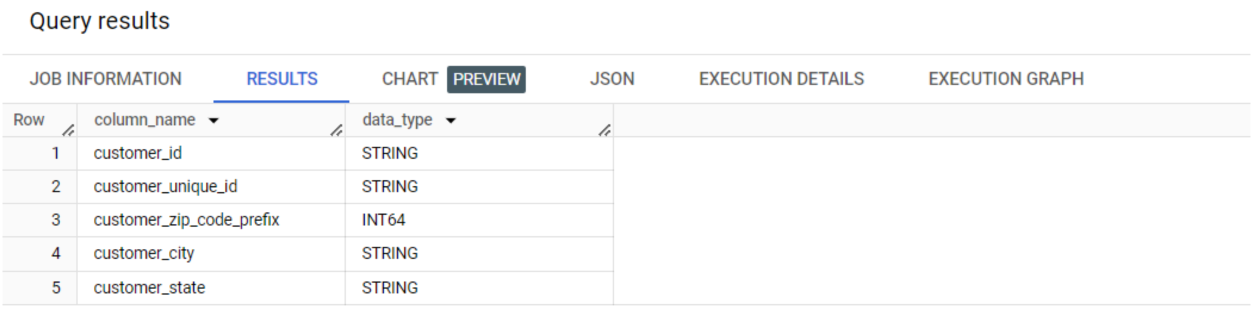
1.1 Data type of all columns in the "customers" table.

```
select column_name,data_type
from `nifty-cabinet-402403.Target.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'customers';
```

Result output:-



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

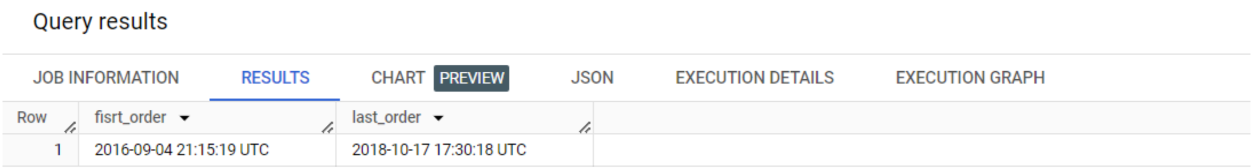
Insights :- There are 4 columns with data types as string and only 1 column with data type as INT64 that is integer.

Recommendations :- N/A

1.2 Get the time range between which the orders were placed.

```
select
  min(order_purchase_timestamp) as `fisrt_order`,
  max(order_purchase_timestamp) as `last_order`
from `Target.orders`;
```

Result Output :-



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

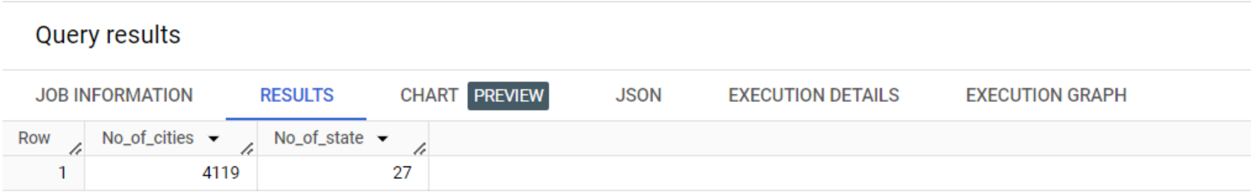
Insights:- The First order was placed on 4th Sept 2016 at 09:15 PM UTC and the last order was placed on 17th Oct 2018 at 5:30 UTC.

Recommendations :- N/A

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

```
select
  count(distinct c.customer_city) as `No_of_cities`,
  count( distinct c.customer_state) as `No_of_state`
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id;
```

Result Output :-



Row	No_of_cities	No_of_state
1	4119	27

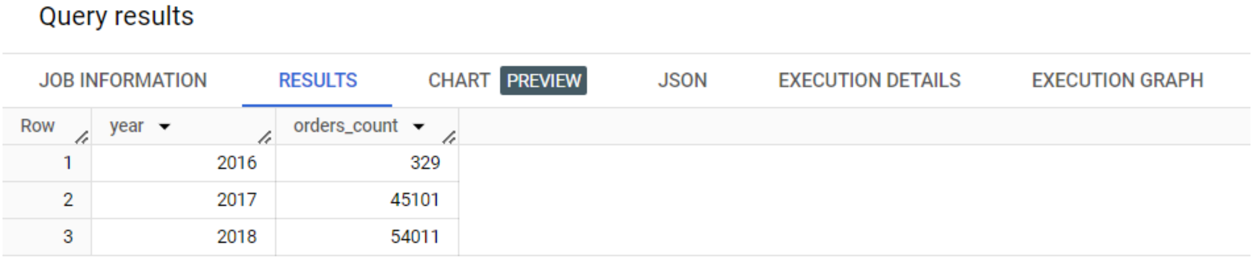
Insights :- The count of the cities of the customers is 4119 and the count of the states of the customers is 27 who ordered during the given time period.

Recommendation :- N/A

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

```
select
  extract(year from order_purchase_timestamp) as `year`,
  count(order_id) as orders_count
from `Target.orders`
group by 1
order by 1;
```

Result output :-



Row	year	orders_count
1	2016	329
2	2017	45101
3	2018	54011

Insights :- The number of orders increased significantly from 2016 to 2017, with a growth rate of 13608.51%. The growth rate slowed down in 2018, but it was still positive at 19.76%. This suggests that the business is experiencing strong growth.

Recommendation :-

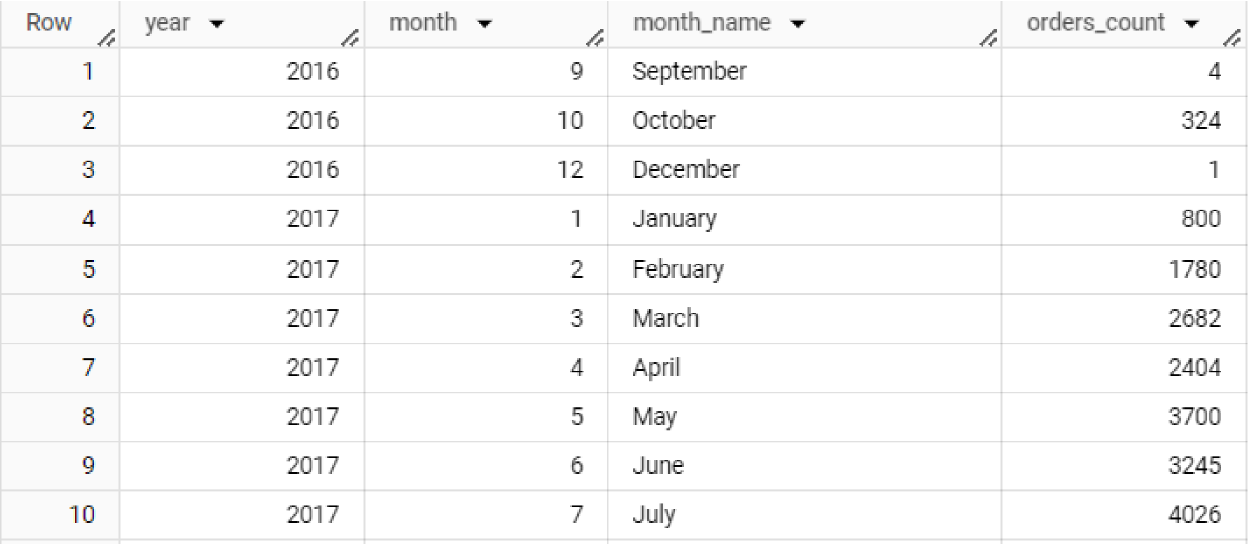
1.Analyze customer data and identify segments with high potential for growth.

2.Analyze the order fulfillment process and identify areas for improvement.

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

```
select
  extract(year from order_purchase_timestamp) as `year`,
  extract(month from order_purchase_timestamp) as `month`,
  format_date('%B',order_purchase_timestamp) as `month_name`,
  count(order_id) as `orders_count`
from `Target.orders`
group by 1,2,3
order by 1,2;
```

Result Output:-



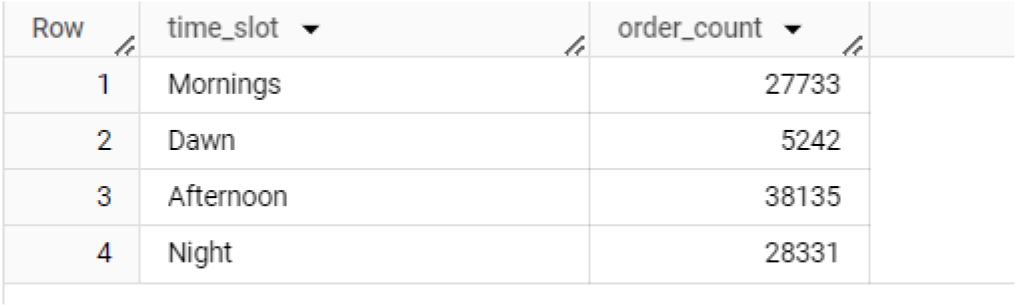
Row	year	month	month_name	orders_count
1	2016	9	September	4
2	2016	10	October	324
3	2016	12	December	1
4	2017	1	January	800
5	2017	2	February	1780
6	2017	3	March	2682
7	2017	4	April	2404
8	2017	5	May	3700
9	2017	6	June	3245
10	2017	7	July	4026

Insights :-
The percentage change from the previous month shows significant fluctuations. There seems to be a seasonal pattern, with orders increasing in the spring and summer months and decreasing in the fall and winter months.
Recommendations :-
1.Develop marketing and sales strategies that target peak and off-peak seasons. For example, you could offer discounts or promotions during the off-peak seasons to encourage customers to buy.
2.Monitor inventory levels to ensure sufficient stock during peak seasons. This will help you to avoid stockouts and lost sales.

2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
0-6 hrs : Dawn
7-12 hrs : Mornings
13-18 hrs : Afternoon
19-23 hrs : Night

```
select
  case
    when extract(hour from order_purchase_timestamp) >=0 and extract(hour from
order_purchase_timestamp) <=6 then 'Dawn'
    when extract(hour from order_purchase_timestamp) >=7 and extract(hour from
order_purchase_timestamp) <= 12 then 'Mornings'
    when extract(hour from order_purchase_timestamp) >=13 and extract(hour from
order_purchase_timestamp) <= 18 then 'Afternoon'
    else 'Night'
  end as `time_slot`,
  count(order_id) as `order_count`
from `Target.orders`
group by `time_slot`;
```

Result Output:-



Row	time_slot	order_count
1	Mornings	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

Insights :- Most amount of orders are placed during the Afternoon time and least amount of orders are placed during the Dawn

Recommendations :-

1.Focus marketing and promotional efforts on increasing orders during time slots with lower order counts, such as Mornings and Night.

2.Consider offering special discounts or promotions during these time slots to attract more customers.

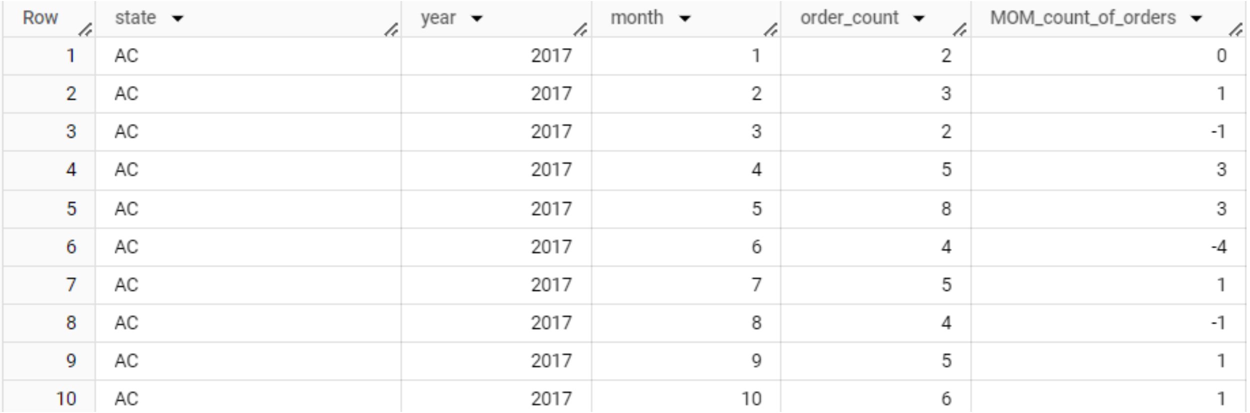
3.Analyze the reasons behind the differences in order counts across different time slots. This information can be used to develop targeted marketing campaigns and improve overall order volume.

3.1 Get the month on month no. of orders placed in each state.

```
With A as
(
select
  c.customer_state as `state`,
  extract(year from o.order_purchase_timestamp) as `year`,
  extract(month from o.order_purchase_timestamp) as `month`,
  count(o.order_id) as `order_count`
from `Target.orders` o
inner join `Target.customers` c
on o.customer_id = c.customer_id
group by 1,2,3
order by 1,2,3
)

select
  A.state,
  A.year,
  A.month,
  A.order_count,
  ifnull(A.order_count - lag(order_count)over(partition by A.state order by
A.year,A.month),0) as `MOM_count_of_orders`
from A
order by 1,2,3;
```

Result Output :-



Row	state	year	month	order_count	MOM_count_of_orders
1	AC	2017	1	2	0
2	AC	2017	2	3	1
3	AC	2017	3	2	-1
4	AC	2017	4	5	3
5	AC	2017	5	8	3
6	AC	2017	6	4	-4
7	AC	2017	7	5	1
8	AC	2017	8	4	-1
9	AC	2017	9	5	1
10	AC	2017	10	6	1

Insights :-1. Positive values indicate an increase in orders, while negative values indicate a decrease.

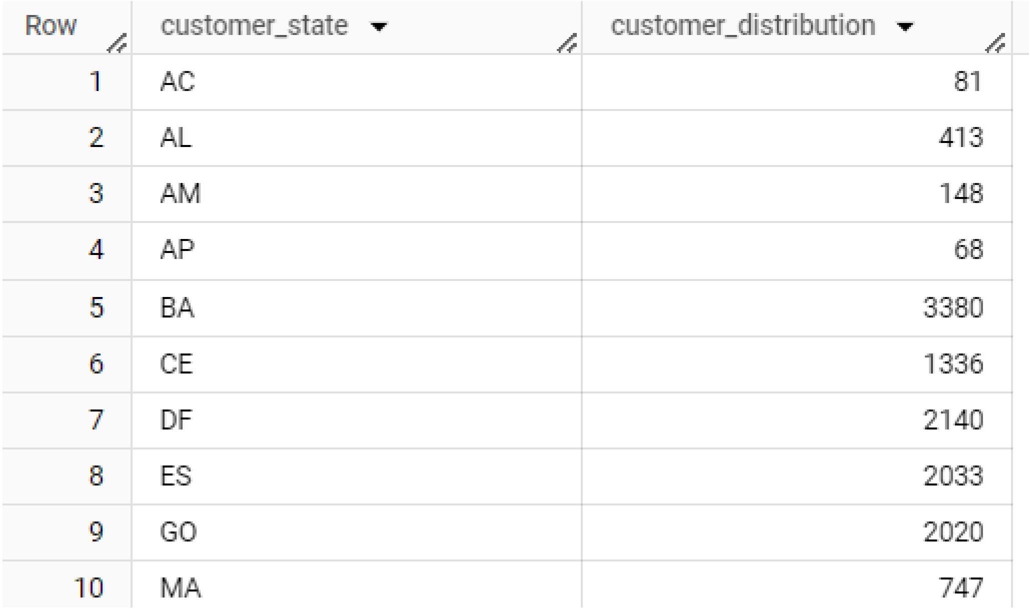
2.States with consistently high or low month-to-month variability can be flagged for further investigation.

Recommendations :- **For states with positive MOM changes, consider exploring the factors contributing to the increase in orders. This could include marketing campaigns, seasonal trends, or other external factors.**

3.2 How are the customers distributed across all the states?

```
select
  customer_state,
  count(distinct customer_id) as `customer_distribution`
from `Target.customers`
group by 1
order by 1;
```

Result Output :-



Row	customer_state	customer_distribution
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

Insights :- 1. Most customers are present in SP state count = 41746 and least number of customers are present in RR count = 46.

2.The top 5 states account for a significant portion (approximately 52%) of the total customer distribution.

3.There are several states with very low customer distribution (below 100).

Recommendation :-

1.Consider targeted campaigns or incentives to increase customer distribution in states with low numbers.

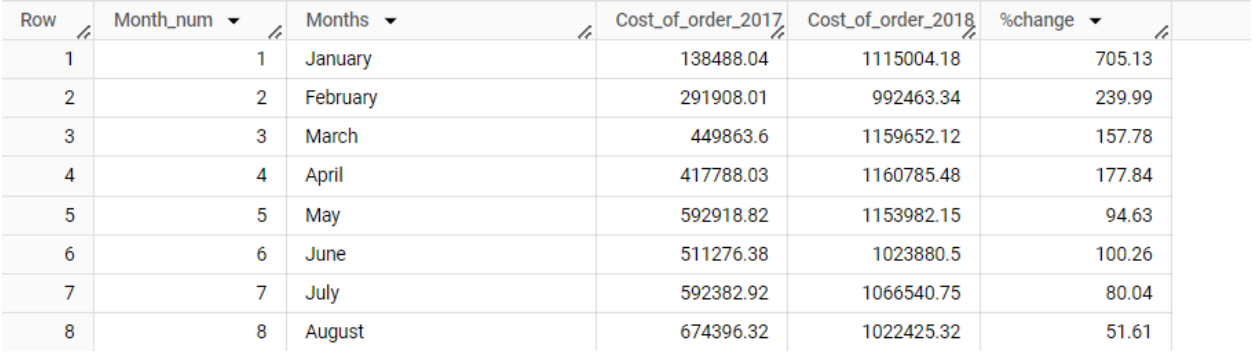
2.Regularly monitor customer distribution trends to identify areas of growth and decline.

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

```
with
  Cost_details as
  (
  select
  extract(month from o.order_purchase_timestamp) as `Month_num`,
  format_timestamp('%B',o.order_purchase_timestamp) as `Months`,
  round(sum(case when extract(year from o.order_purchase_timestamp) = 2017 then
p.total_value_of_order else 0 end),2) as `Cost_of_order_2017`,
  round(sum(case when extract(year from o.order_purchase_timestamp) = 2018 then
p.total_value_of_order else 0 end),2) as `Cost_of_order_2018`
  from `Target.orders` o
  left join
  (
    select order_id,sum(payment_value) as `total_value_of_order`
    from `Target.payments`
    group by order_id
  ) p
  on o.order_id = p.order_id
  where extract(month from o.order_purchase_timestamp) between 1 and 8
  group by 1,2
  order by 1 asc
  )

select
  *,
  round(((Cost_details.Cost_of_order_2018
-Cost_details.Cost_of_order_2017)/Cost_details.Cost_of_order_2017) * 100,2) as
`%change`
from Cost_details;
```

Result Output:-



Row	Month_num	Months	Cost_of_order_2017	Cost_of_order_2018	%change
1	1	January	138488.04	1115004.18	705.13
2	2	February	291908.01	992463.34	239.99
3	3	March	449863.6	1159652.12	157.78
4	4	April	417788.03	1160785.48	177.84
5	5	May	592918.82	1153982.15	94.63
6	6	June	511276.38	1023880.5	100.26
7	7	July	592382.92	1066540.75	80.04
8	8	August	674396.32	1022425.32	51.61

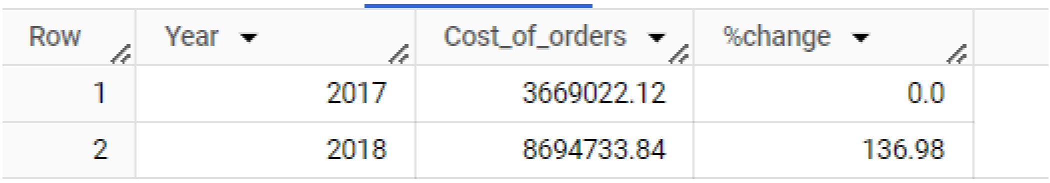
Insights :-

1.above query gives the %change of cost of orders from Jan to Aug for 2017 and 2018

```
with
  Cost_details as
  (
  select
  extract(year from o.order_purchase_timestamp) as `Year`,
  round(sum(case when extract(year from o.order_purchase_timestamp) = 2017 then
p.total_value_of_order else p.total_value_of_order end),2) as
  `Cost_of_orders`
  from `Target.orders` o
  left join
  (
    select order_id,sum(payment_value) as `total_value_of_order`
    from `Target.payments`
    group by order_id
  ) p
  on o.order_id = p.order_id
  where o.order_purchase_timestamp between '2017-01-01' and '2018-12-31' and
extract(month from o.order_purchase_timestamp) between 1 and 8
  group by 1
  order by 1 asc
  )

select
  *,
  round(ifnull(((Cost_of_orders - lag(Cost_of_orders) over(order by Year
asc))/lag(Cost_of_orders) over(order by Year asc)) *100,0) ,2) as `%change`
from Cost_details
order by Year asc;
```

Result Output:-



Row	Year	Cost_of_orders	%change
1	2017	3669022.12	0.0
2	2018	8694733.84	136.98

Insights :-

1.The total cost of orders increased significantly from 2017 to 2018, with a percentage change of 136.98%.

2.The average cost of orders was also higher in 2018 compared to 2017.

3.2018 had the highest cost of orders, while 2017 had the lowest.

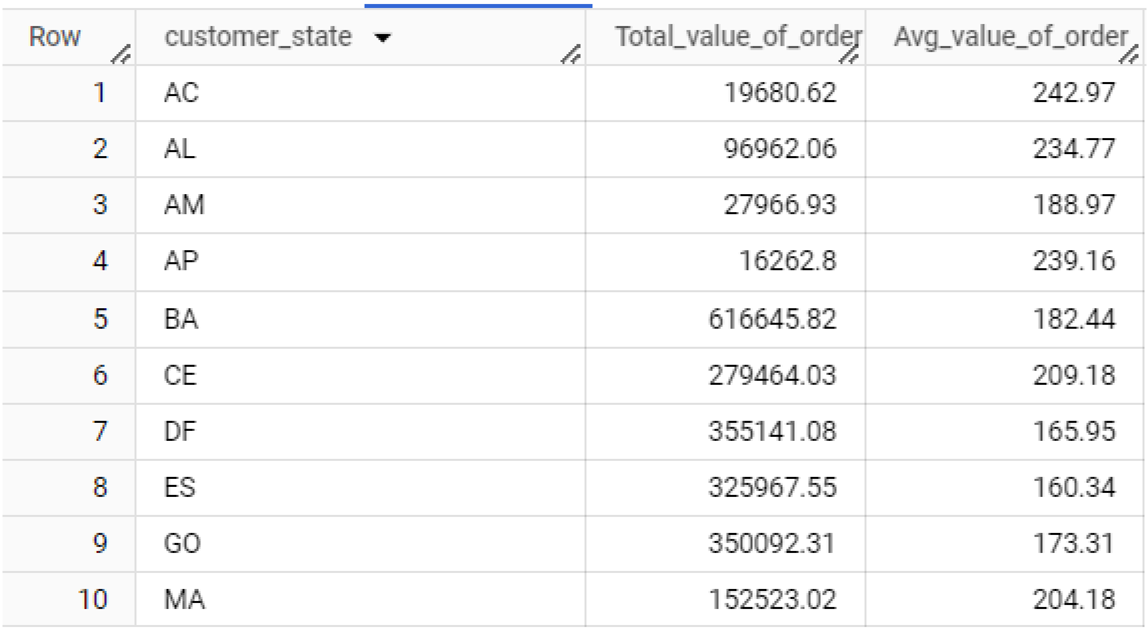
Recommendations :-

1.Implement strategies to control or reduce the cost of orders, such as negotiating with suppliers for better prices, optimizing ordering processes, or finding alternative suppliers.

4.2 Calculate the Total & Average value of order price for each state.

```
select
  distinct c.customer_state,
  round(sum(p.value_of_order) over(partition by c.customer_state),2) as
`Total_value_of_order`,
  round(avg(p.value_of_order) over(partition by c.customer_state),2) as
`Avg_value_of_order`
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
inner join
  (
    select order_id,sum(payment_value) as `value_of_order`
    from `Target.payments`
    group by order_id
  ) p
on o.order_id = p.order_id
order by 1;
```

Result Output :-



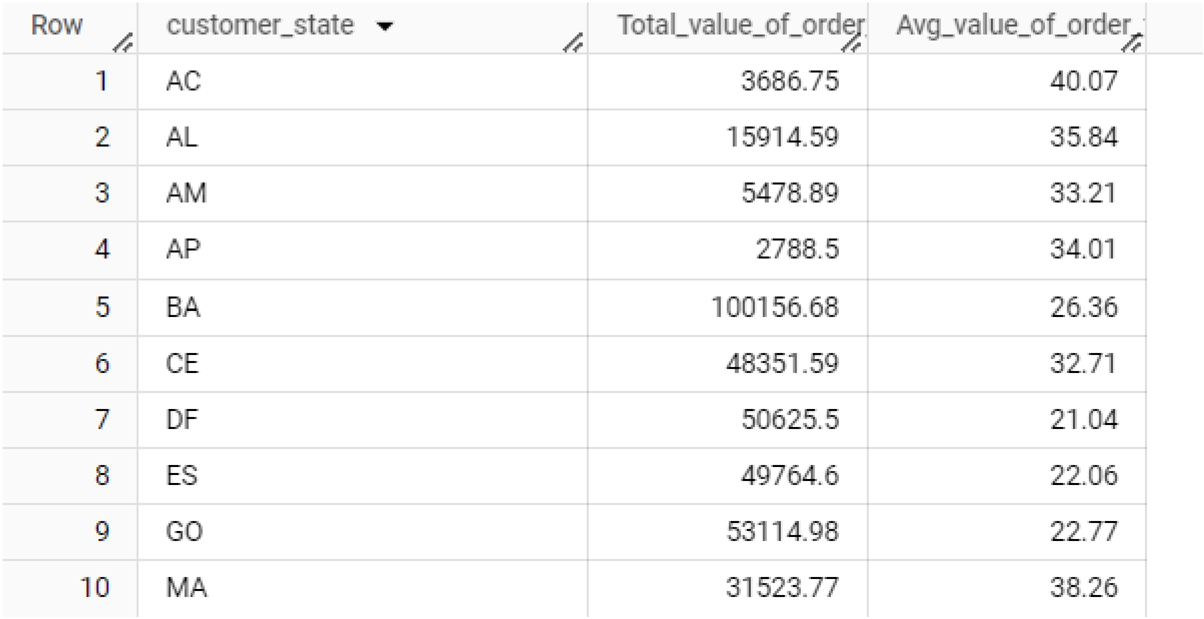
Row	customer_state	Total_value_of_order	Avg_value_of_order
1	AC	19680.62	242.97
2	AL	96962.06	234.77
3	AM	27966.93	188.97
4	AP	16262.8	239.16
5	BA	616645.82	182.44
6	CE	279464.03	209.18
7	DF	355141.08	165.95
8	ES	325967.55	160.34
9	GO	350092.31	173.31
10	MA	152523.02	204.18

Insights :-
1.The state with max Total value of order and the state with least avg is state SP, the min Total value of order is for state RR and the state with max avg is PB
Recommendations :-
1.We need to improve in the state with the less avg value of order,it will directly impact the overall performance.

4.3 Calculate the Total & Average value of order freight for each state.

```
select
  distinct c.customer_state,
  round(sum(oi.freight_value) over(partition by c.customer_state),2) as
`Total_value_of_order_freight`,
  round(avg(oi.freight_value) over(partition by c.customer_state),2) as
`Avg_value_of_order_freight`
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
inner join `Target.order_items` oi
on o.order_id = oi.order_id
order by 1;
```

Result Output:-



Row	customer_state	Total_value_of_order	Avg_value_of_order_
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26

Insights :-

1.The state with the max total value of order freight and the state with least avg of order freight is state SP, the min Total value of order freight is for state RR and the state with max avg value of freight is RR

Recommendations:-

1.We need to reduce the freight charges in the state with the more avg value of order,it will directly impact the overall performance.We can achieve this by ship on off-peak days, and be known for loading quickly.

5.1 Find the no. of days taken to deliver each order from the order's purchase date as delivery time.

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

Do this in a single query.

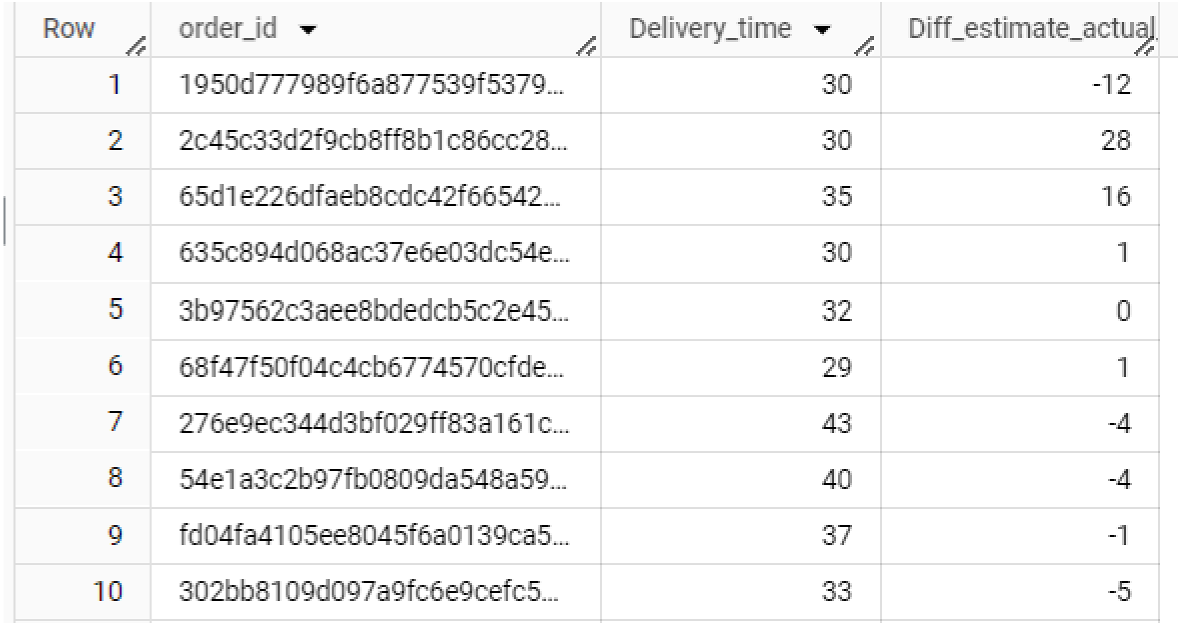
You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

time_to_deliver = order_delivered_customer_date - order_purchase_timestamp

diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date

```
select
  order_id,
  if(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) is
null,0,timestamp_diff(order_delivered_customer_date,
   order_purchase_timestamp,day)) as `Delivery_time`,
  if(timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date,day)
is null,0,timestamp_diff(order_estimated_delivery_date,
   order_delivered_customer_date,day)) as `Diff_estimate_actual_delivery`
from `Target.orders`;
```

Result Output :-



Row	order_id	Delivery_time	Diff_estimate_actual
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

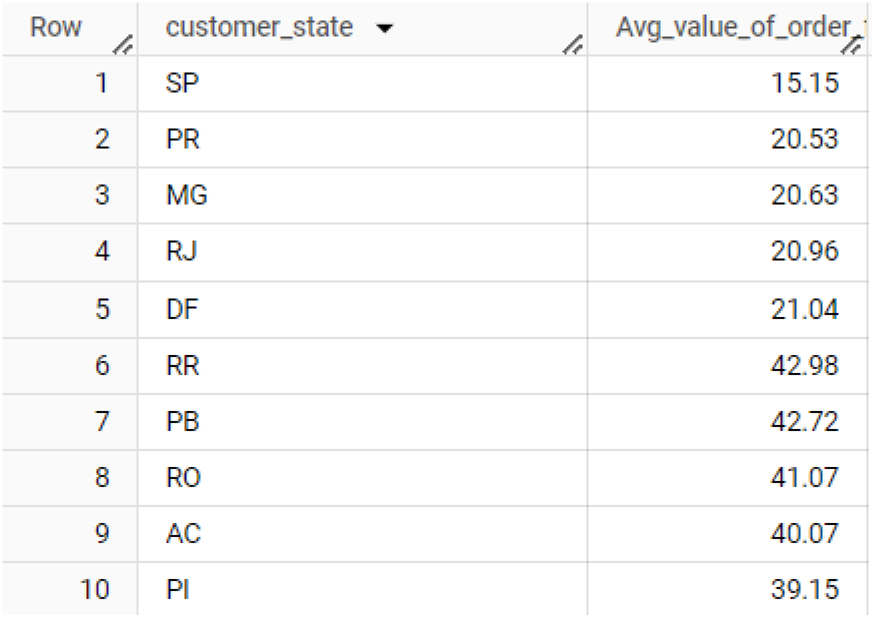
Insights :-
The max and min delivery time is as follows: 209,0. The max diff of estimate and actual delivery date is 146 and min diff is -188 days that would be counted as delivery before the estimate date.
Recommendations:-
we would need to work on the making the increasing the employees so that the Delivery time for each order should be min close to 0 and diff between estimate and actual should be ideally least and we are seeing this in some cases that actual delivery is less than estimate we should promote this pattern for growing the customer engagement for further new orders.

5.2 Find out the top 5 states with the highest & lowest average freight value.

This below query gives the top 5 results with the states with the lowest average value In low to high fashion and next 5 results with the states with the highest average value in high to low fashion

```
(select
  distinct c.customer_state,
  round(avg(oi.freight_value) over(partition by c.customer_state),2) as `Avg_value_of_order_freight`
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
inner join `Target.order_items` oi
on o.order_id = oi.order_id
order by 2 desc
limit 5)
union all
(select
  distinct c.customer_state,
  round(avg(oi.freight_value) over(partition by c.customer_state),2) as `Avg_value_of_order_freight`
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
inner join `Target.order_items` oi
on o.order_id = oi.order_id
order by 2 asc
limit 5);
```

Result Output:-



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

Insights:-
1.The states with the lowest average value are SP,PR,MG,RJ,DF in low to high order with SP being the lowest and DF being the highest (least 5 states.)
2.The states with the highest average value are RR,PB,RO,AC,PI in high to low order with RR being the highest and PI being the lowest(top 5 states).
Recommendations :-
1.Leverage volume discounts and negotiate with different carriers to secure competitive rates.
2.We need to analyze delivery routes and explore alternative shipping methods, and implement fuel-efficient practices to reduce costs.
3.Implement bulk ordering, consolidate shipments, and utilize multi-modal transportation for cost optimization.

5.3 Find out the top 5 states with the highest & lowest average delivery time.

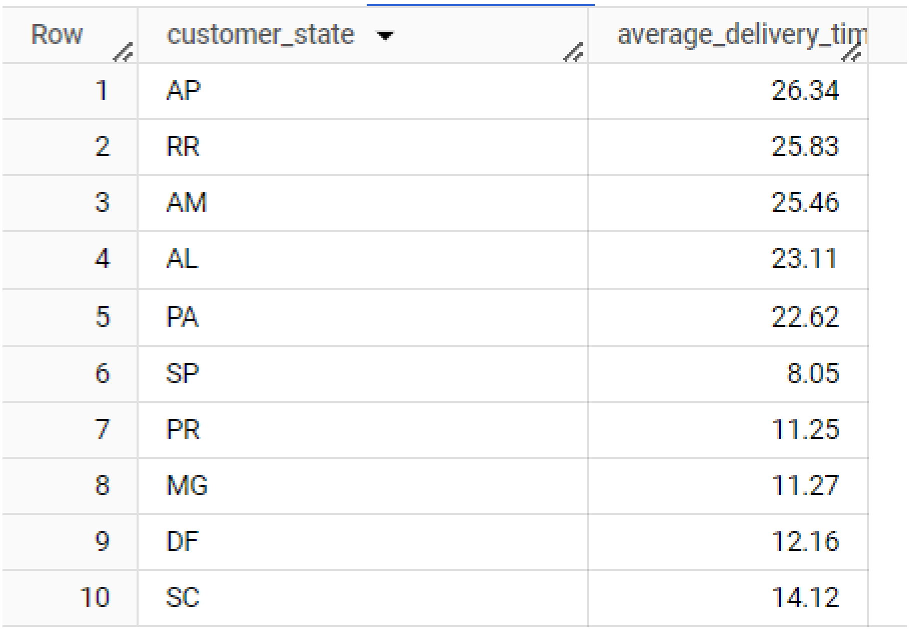
This below query gives the top 5 results with the states with the highest average value In high to low fashion and next 5 results with the states with the lowest average value in low to high fashion

```
(with delivery_time_details as
(
  select
    o.order_id,
    c.customer_state,
    if(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) is
null,0,timestamp_diff(order_delivered_customer_date,
    order_purchase_timestamp,day)) as `Delivery_time`
  from `Target.customers` c
  inner join `Target.orders` o
  on c.customer_id = o.customer_id
)

select
  customer_state,
  round(avg(Delivery_time),2) as `average_delivery_time`
from delivery_time_details
group by 1
order by 2 desc
limit 5)
union all
(with delivery_time_details as
(
  select
    o.order_id,
    c.customer_state,
    if(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) is
null,0,timestamp_diff(order_delivered_customer_date,
    order_purchase_timestamp,day)) as `Delivery_time`
  from `Target.customers` c
  inner join `Target.orders` o
  on c.customer_id = o.customer_id
)

select
  customer_state,
  round(avg(Delivery_time),2) as `average_delivery_time`
from delivery_time_details
group by 1
order by 2 asc
limit 5
);
```

```
Result Output :-
```



Row	customer_state	average_delivery_tim
1	AP	26.34
2	RR	25.83
3	AM	25.46
4	AL	23.11
5	PA	22.62
6	SP	8.05
7	PR	11.25
8	MG	11.27
9	DF	12.16
10	SC	14.12

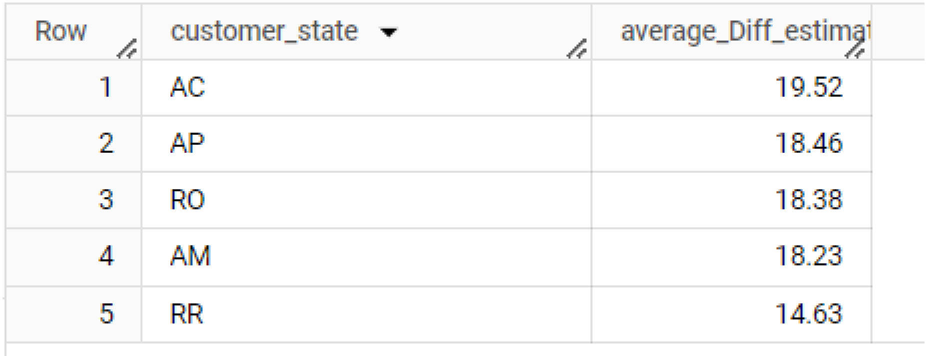
Insights :-
1.The top 5 states with the most avg delivery time are AP,RR,AM,AL,PA in High to Low format.
2.The least 5 states with the least avg delivery time are SP,PR,MG,DF,SC in low to high format.
Recommendations :-
1.Implement a culture of continuous improvement to identify and implement new strategies for optimizing delivery time.
2.Train delivery personnel on efficient delivery practices and customer service skills.
3.Implement technologies like GPS tracking, route optimization software, and customer communication platforms to improve delivery time.

5.4 Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
With delivery_time_details as
  (
    select
      o.order_id,
      c.customer_state,
if(timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date,day) is
null,0,timestamp_diff(order_estimated_delivery_date,
        order_delivered_customer_date,day)) as `Diff_estimate_actual_delivery`
    from `Target.customers` c
    inner join `Target.orders` o
    on c.customer_id = o.customer_id
  )

select
  customer_state,
  round(avg(Diff_estimate_actual_delivery),2) as
`average_Diff_estimate_actual_delivery`
from delivery_time_details
group by 1
order by 2 desc
limit 5;
```

Result Output:-



Row	customer_state	average_Diff_estimat
1	AC	19.52
2	AP	18.46
3	RO	18.38
4	AM	18.23
5	RR	14.63

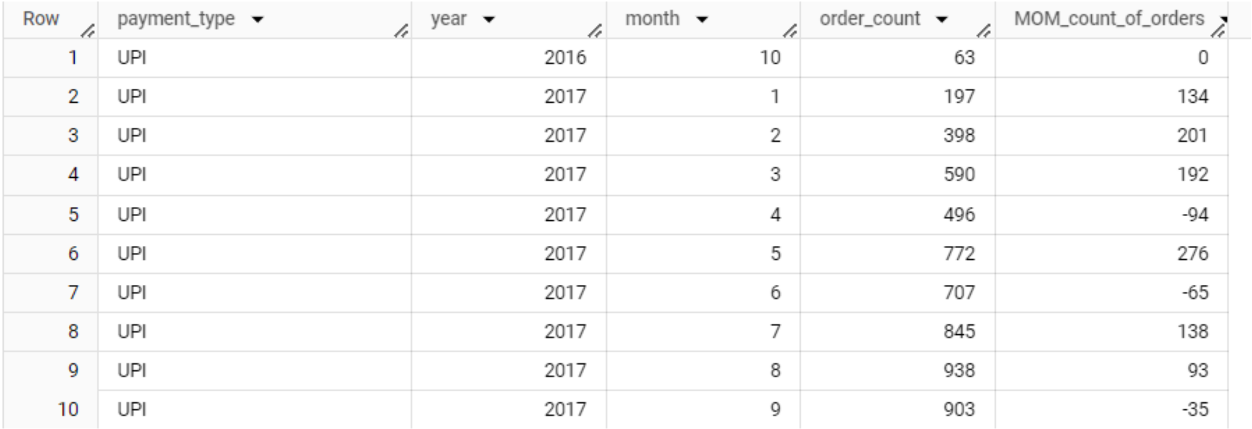
Insights :-
1.The top 5 states with the order delivery is fast compared to estimated delivery are AC,AP,RO,AM,RR in slow to fast order.
Recommendations:-
1.We should try to keep this factor more so we can increase the staff for logistical support.

6.1 Find the month on month no. of orders placed using different payment types.

```
With A as
(
select
  p.payment_type,
  extract(year from o.order_purchase_timestamp) as `year`,
  extract(month from o.order_purchase_timestamp) as `month`,
  count(o.order_id) as `order_count`
from `Target.orders` o
inner join
  (
    select order_id,payment_type
    from `Target.payments`
  ) p
on o.order_id = p.order_id
group by 1,2,3
order by 1,2,3
)

select
  distinct
  A.payment_type,
  A.year,
  A.month,
  A.order_count,
  ifnull(A.order_count - lag(order_count)over(partition by A.payment_type order by
A.year,A.month),0) as `MOM_count_of_orders`
from A
order by 1,2,3;
```

Result Output:-



Row	payment_type	year	month	order_count	MOM_count_of_orders
1	UPI	2016	10	63	0
2	UPI	2017	1	197	134
3	UPI	2017	2	398	201
4	UPI	2017	3	590	192
5	UPI	2017	4	496	-94
6	UPI	2017	5	772	276
7	UPI	2017	6	707	-65
8	UPI	2017	7	845	138
9	UPI	2017	8	938	93
10	UPI	2017	9	903	-35

Insights :-

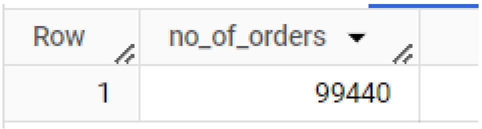
1.UPI is the most popular payment method, accounting for the majority of orders.UPI orders experienced substantial growth throughout 2017 and 2018, suggesting increasing adoption.Order count shows a seasonal pattern with peaks in November and December.

2.Credit card is the second most used payment method, with a significant increase in orders throughout 2017 and 2018.Similar to UPI, credit card orders show a seasonal peak in November and December.
3.Debit card is a less preferred payment method with a much smaller order count compared to UPI and credit card.Debit card usage fluctuates significantly throughout the period, making it difficult to identify consistent trends.
4.Voucher usage seems moderate with consistent order counts throughout the period.Like other payment methods, voucher orders peak in November and December.There is a significant drop in voucher orders in September 2018, followed by a further decline in October 2018.
Recommendations :-
1.Given their dominance and growth, prioritize marketing and promotional efforts towards UPI and credit card users.
2.We need to adjust different marketing strategies for seasonal peaks.

6.2 Find the no. of orders placed on the basis of the payment installments that have been paid.

```
with A as
  (
    select order_id,count(payment_installments) as `no_of_payments_for_each_order`
    from `Target.payments`
    group by 1
    having count(payment_installments) > 0
  )
select count(order_id) as `no_of_orders`
from A;
```

Result output :-



Row	no_of_orders
1	99440

Insights :-
The total number of orders placed with at least one payment is 99440.
Recommendations :- N/A