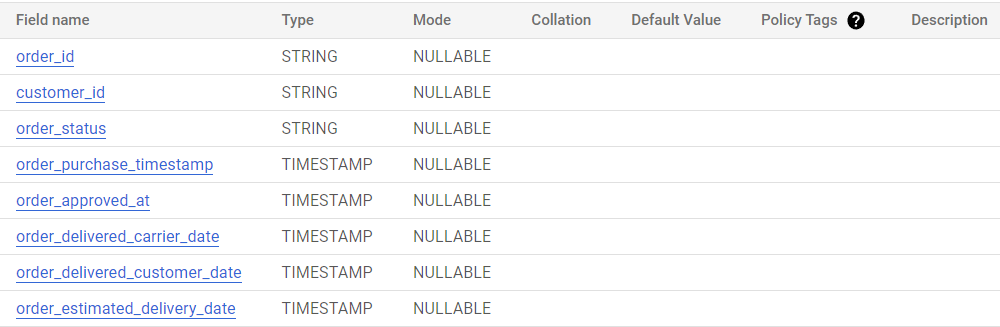
**Q1** **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**

**Part (1):Data type of columns in a table:**

**Ans:** For example, data type from orders table:



**Part (2):Time period for which the data is given:**

**Query Code:**

select

min(order\_purchase\_timestamp) as min\_time\_stamp,

max(order\_purchase\_timestamp) as max\_time\_stamp

from `scaler-dsml-sql-373605.target\_sql.orders`

**Query Result:**



**Part (3): Cities and States of customers ordered during the given period:**

**Query Code:**

select

distinct

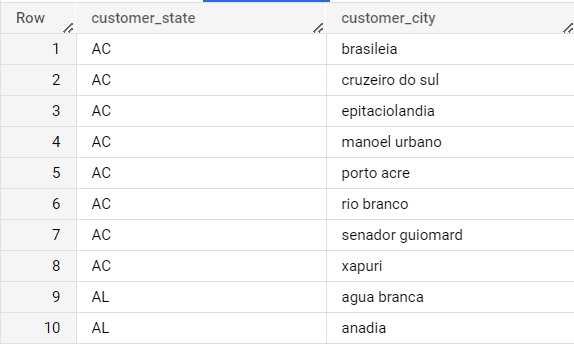
customer\_state,

customer\_city

from `scaler-dsml-sql-373605.target\_sql.customers`

order by customer\_state,customer\_city

**Query Result:**

****

**Q:2** **In-depth Exploration:**

**Part (1) (i) : Is there a growing trend on e-commerce in Brazil?**

**Query Code:**

select

extract (year from timestamp (o.order\_purchase\_timestamp )) as year,

extract (month from timestamp (o.order\_purchase\_timestamp )) as month,

count(o.order\_id) as cnt\_ord

from `scaler-dsml-sql-373605.target\_sql.orders` as o

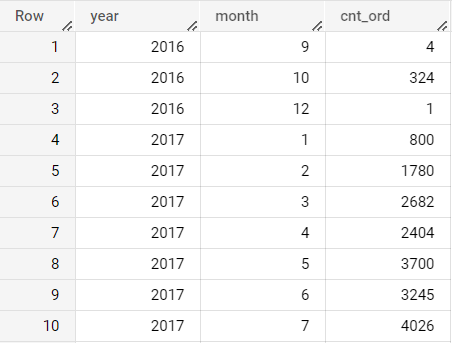
inner join `scaler-dsml-sql-373605.target\_sql.customers`as c

on o.customer\_id=c.customer\_id

group by year,month

order by year,month

**Query Result:**

****

**Part(1) (ii) How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?**

**Query Code:**

with cte\_1 as(

select

extract (year from timestamp (o.order\_purchase\_timestamp )) as year,

extract (month from timestamp (o.order\_purchase\_timestamp )) as month,

count(o.order\_id) as cnt\_ord

from `scaler-dsml-sql-373605.target\_sql.orders` as o

inner join `scaler-dsml-sql-373605.target\_sql.customers`as c

on o.customer\_id=c.customer\_id

group by year,month

order by year,month

),

cte\_2 as (

select

year,

month,

cnt\_ord

from cte\_1

where year=2016

),

cte\_3 as(

select

year,

month,

cnt\_ord

from cte\_1

where year=2017

),

cte\_4 as (

select

year,

month,

cnt\_ord

from cte\_1

where year=2018

)

select

ct3.month,

ifnull(ct2.cnt\_ord,0) as cnt\_ord\_2016,

ifnull(ct3.cnt\_ord,0) as cnt\_ord\_2017,

ifnull(ct4.cnt\_ord,0) as cnt\_ord\_2018,

(ifnull(ct2.cnt\_ord,0)+ifnull(ct3.cnt\_ord,0)+ifnull(ct4.cnt\_ord,0)) as total\_cnt\_ord

from cte\_3 as ct3

left join cte\_2 as ct2

on ct3.month=ct2.month

left join cte\_4 as ct4

on ct3.month=ct4.month

order by ct3.month

**Query Result:**

**Table

Description automatically generated**

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

Here, I took assumptions like

* Dawn period is 00:00:00 to 05:59:59
* Morning period is 06:00:00 to 11:59:59
* Afternoon period is 12:00:00 to 17:59:59
* Night period is 18:00:00 to 23:59:59

**Query Code:**

select distinct

Hour\_stamp,

count(order\_id) over(partition by hour\_stamp) as cnt\_ord

from (select

customer\_id,

order\_id,

case

when extract (hour from timestamp (order\_purchase\_timestamp) )>=1 and extract (hour from timestamp (order\_purchase\_timestamp) )<6 then "dawn"

when extract (hour from timestamp (order\_purchase\_timestamp) )>=6 and extract (hour from timestamp (order\_purchase\_timestamp) )<12 then "Morning"

when extract (hour from timestamp (order\_purchase\_timestamp) )>=12 and extract (hour from timestamp (order\_purchase\_timestamp) )<18 then "Afternoon"

else "Night"

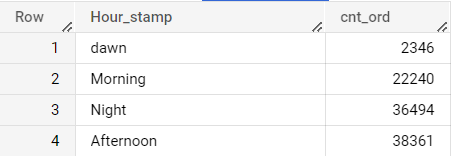
End as Hour\_stamp from `scaler-dsml-sql-373605.target\_sql.orders`) t

inner join `scaler-dsml-sql-373605.target\_sql.customers` c

on t.customer\_id=c.customer\_id

order by cnt\_ord

**Query Result:**



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

**Part (1): Get month on month orders by states**

**Query Code:**

select

c.customer\_state,

extract (year from timestamp (o.order\_purchase\_timestamp)) as year,

extract (month from timestamp (o.order\_purchase\_timestamp)) as month,

count(o.order\_id) as ord\_cnt,

from `scaler-dsml-sql-373605.target\_sql.orders` o

inner join `scaler-dsml-sql-373605.target\_sql.customers` c

on o.customer\_id=c.customer\_id

group by c.customer\_state,year,month

order by year,month,ord\_cnt desc

**Query Result:**

**Table

Description automatically generated**

**Part (2): Distribution of customers across the states in Brazil**

**Query Code:**

select

customer\_state,

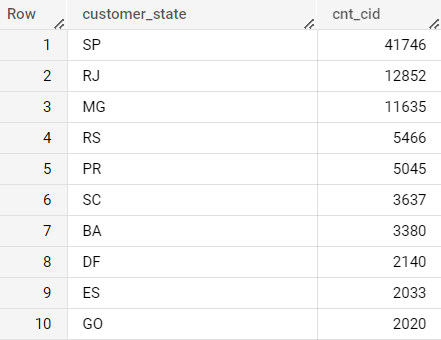
count(customer\_id) as cnt\_cid

from `scaler-dsml-sql-373605.target\_sql.customers`

group by customer\_state

order by cnt\_cid desc

**Query Result:**

****

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

**Part (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 Code:**

with cte\_1 as (

select

extract (year from timestamp(order\_purchase\_timestamp)) as year,

order\_id,

order\_purchase\_timestamp

from `scaler-dsml-sql-373605.target\_sql.orders`

),

cte\_2 as (

select

order\_id,

order\_purchase\_timestamp,

extract (month from timestamp(order\_purchase\_timestamp)) as month

from cte\_1

where extract (month from timestamp(order\_purchase\_timestamp)) between 1 and 8

),

cte\_3 as (

select distinct

ct1.year,

ct2.month,

count(distinct py.order\_id) over(partition by ct2.month) as cnt\_ord,

round(sum(py.payment\_value) over(partition by ct2.month),2) as sum\_month\_payment\_value,

round(avg(py.payment\_value) over(partition by ct2.month),2) as avg\_month\_payment\_value

from cte\_2 as ct2

inner join `scaler-dsml-sql-373605.target\_sql.payments` as py

on ct2.order\_id=py.order\_id

inner join cte\_1 as ct1

on ct1.order\_id=py.order\_id

where ct1.year=2017

order by ct2.month

),

cte\_4 as (

select distinct

ct1.year,

ct2.month,

count(distinct py.order\_id) over(partition by ct2.month) as cnt\_ord,

round(sum(py.payment\_value) over(partition by ct2.month),2) as sum\_month\_payment\_value,

round(avg(py.payment\_value) over(partition by ct2.month),2) as avg\_month\_payment\_value

from cte\_2 as ct2

inner join `scaler-dsml-sql-373605.target\_sql.payments` as py

on ct2.order\_id=py.order\_id

inner join cte\_1 as ct1

on ct1.order\_id=py.order\_id

where ct1.year=2018

order by ct2.month

)

select

concat(ct3.year,"-",ct4.year) as years,

ct3.month,

ct4.cnt\_ord as cnt\_ord\_2018,

ct3.cnt\_ord as cnt\_ord\_2017,

round(((ct4.cnt\_ord-ct3.cnt\_ord)/ct3.cnt\_ord)\*100,2) as percent\_ord\_change,

ct4.sum\_month\_payment\_value as sum\_month\_payment\_value\_2018,

ct3.sum\_month\_payment\_value as sum\_month\_payment\_value\_2017,

round(((ct4.sum\_month\_payment\_value-ct3.sum\_month\_payment\_value)/ct3.sum\_month\_payment\_value)\*100,2) as percent\_sum\_PV\_change,

ct4.avg\_month\_payment\_value as avg\_month\_payment\_value\_2018,

ct3.avg\_month\_payment\_value as avg\_month\_payment\_value\_2017,

round(((ct4.avg\_month\_payment\_value-ct3.avg\_month\_payment\_value)/ct3.avg\_month\_payment\_value)\*100,2) as percent\_avg\_PV\_change

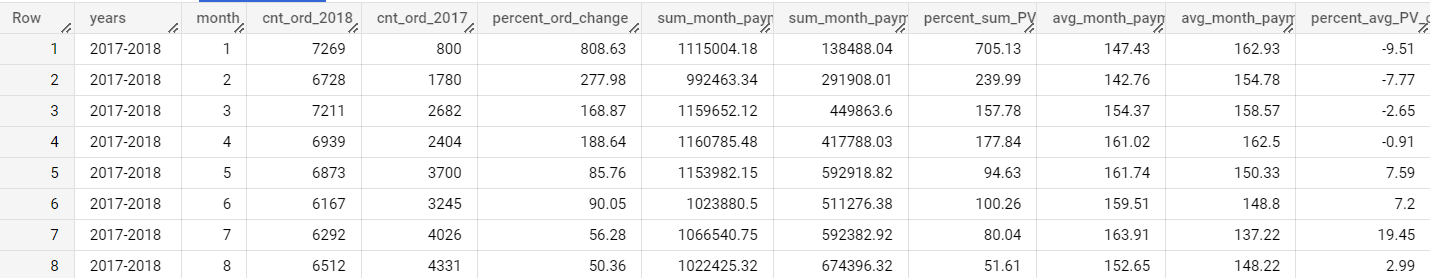
from cte\_3 as ct3

inner join cte\_4 as ct4

on ct3.month=ct4.month

order by ct3.month

**Query Result:**

****

**Part (2): Mean & Sum of price and freight value by customer state**

**Query Code:**

with cte\_1 as (

select

order\_id,

customer\_id

from `scaler-dsml-sql-373605.target\_sql.orders`

),

cte\_2 as (

select

c.customer\_state,

c.customer\_id

from `scaler-dsml-sql-373605.target\_sql.customers` c

inner join `scaler-dsml-sql-373605.target\_sql.orders` o

on c.customer\_id= o.customer\_id

),

cte\_3 as (

select

o.order\_id,

o.customer\_id,

oe.price,

oe.freight\_value,

from `scaler-dsml-sql-373605.target\_sql.orders`o

inner join `scaler-dsml-sql-373605.target\_sql.order\_items` oe

on oe.order\_id=o.order\_id

)

select

distinct

ct2.customer\_state,

round(avg(ct3.price) over(partition by ct2.customer\_state ),2) as avg\_price,

round(sum(ct3.price) over(partition by ct2.customer\_state ),2) as sum\_price,

round(avg(ct3.freight\_value) over(partition by ct2.customer\_state ),2) as avg\_freight,

round(sum(ct3.freight\_value) over(partition by ct2.customer\_state ),2) as sum\_freight,

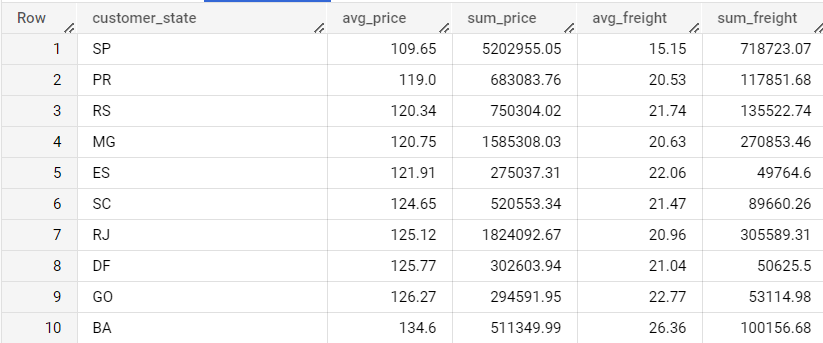
from cte\_2 as ct2

left join cte\_3 as ct3

on ct2.customer\_id=ct3.customer\_id

order by avg\_price,avg\_freight,sum\_price,sum\_freight

**Query Result:**

****

**Q:5 Analysis on sales, freight, and delivery time**

**Part (1): Calculate days between purchasing, delivering and estimated delivery**

**Query Code:**

select

order\_id,

DATE\_DIFF(DATE\_4,DATE\_1,day) as expected\_days,

DATE\_DIFF(DATE\_3,DATE\_1,day) as actual\_days,

DATE\_DIFF(DATE\_2,DATE\_1,day) as dispatch\_days

from (select

order\_id,

extract(date from timestamp(order\_purchase\_timestamp)) as DATE\_1,

extract(date from timestamp(order\_delivered\_carrier\_date)) as DATE\_2,

extract(date from timestamp(order\_delivered\_customer\_date)) as DATE\_3,

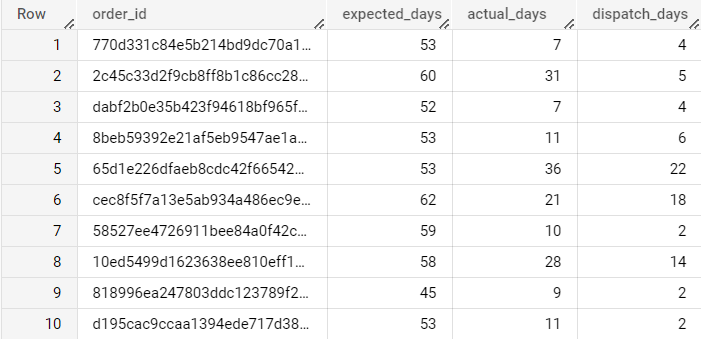
extract(date from timestamp(order\_estimated\_delivery\_date)) as DATE\_4,

from `scaler-dsml-sql-373605.target\_sql.orders`

) as t

where DATE\_1 is not null and DATE\_2 is not null and DATE\_3 is not null and DATE\_4 is not null

**Query Result:**

****

**Part (2): 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**

**Query code:**

select

order\_id,

DATETIME\_DIFF(DATETIME\_2,DATETIME\_1,day) as time\_to\_delivery,

DATETIME\_DIFF(DATETIME\_3,DATETIME\_2,day) as diff\_estimated\_delivery

from

(select

order\_id,

extract(DATETIME  from timestamp(order\_purchase\_timestamp)) as DATETIME\_1,

extract(DATETIME  from timestamp(order\_delivered\_customer\_date)) as DATETIME\_2,

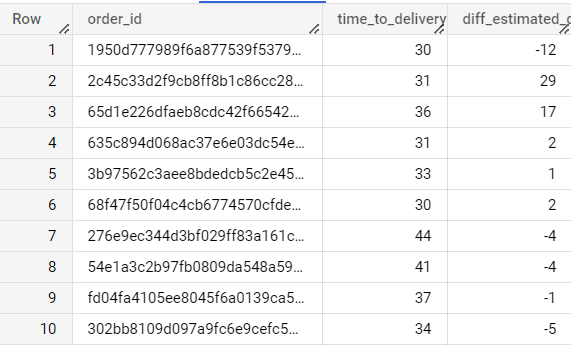
extract(DATETIME  from timestamp(order\_estimated\_delivery\_date)) as DATETIME\_3,

from `scaler-dsml-sql-373605.target\_sql.orders`

) as t

where DATETIME\_1 is not null and DATETIME\_2 is not null and DATETIME\_3 is not null

**Query Result:**

****

**Part (3) Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery**

**Query code:**

with cte\_1 as (

select

order\_id,

customer\_id,

DATETIME\_DIFF(DATETIME\_2,DATETIME\_1,day) as time\_to\_delivery,

DATETIME\_DIFF(DATETIME\_3,DATETIME\_2,day) as diff\_estimated\_delivery

from (select

order\_id,

customer\_id,

extract(DATETIME  from timestamp(order\_purchase\_timestamp)) as DATETIME\_1,

extract(DATETIME  from timestamp(order\_delivered\_customer\_date)) as DATETIME\_2,

extract(DATETIME  from timestamp(order\_estimated\_delivery\_date)) as DATETIME\_3,

from `scaler-dsml-sql-373605.target\_sql.orders`

) as t

where DATETIME\_1 is not null and DATETIME\_2 is not null and DATETIME\_3 is not null

)

select

distinct

c.customer\_state,

round(avg(oe.freight\_value) over(partition by c.customer\_state),2) as avg\_freight,

round(avg(ct1.time\_to\_delivery) over(partition by c.customer\_state),2) as avg\_time\_to\_delivery,

round(avg(ct1.diff\_estimated\_delivery) over(partition by c.customer\_state),2) as avg\_diff\_estimated\_delivery

from `scaler-dsml-sql-373605.target\_sql.customers` as c

inner join cte\_1 as ct1

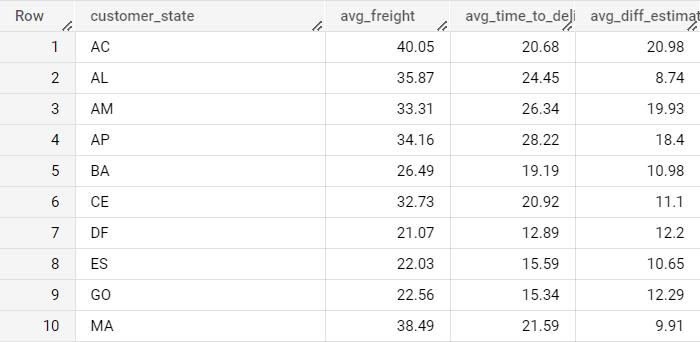
on ct1.customer\_id=c.customer\_id

inner join `scaler-dsml-sql-373605.target\_sql.order\_items` as oe

on oe.order\_id=ct1.order\_id

order by c.customer\_state,avg\_freight,avg\_time\_to\_delivery,avg\_diff\_estimated\_delivery desc

**Query Result:**

****

**Part(4) Sort the data to get the following:**

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

**Query Code:**

select

distinct

c.customer\_state,

round(avg(oe.freight\_value) over(partition by c.customer\_state),2) as avg\_freight,

from `scaler-dsml-sql-373605.target\_sql.orders` as o

inner join `scaler-dsml-sql-373605.target\_sql.customers` as c

on c.customer\_id=o.customer\_id

inner join `scaler-dsml-sql-373605.target\_sql.order\_items` as oe

on oe.order\_id=o.order\_id

order by avg\_freight desc

limit 5

**Query Result:**

**Table

Description automatically generated**

**Part (6) Top 5 states with highest/lowest average time to delivery**

**Query Code:**

with cte\_1 as (

select

order\_id,

customer\_id,

DATETIME\_DIFF(DATETIME\_2,DATETIME\_1,day) as time\_to\_delivery,

from (select

order\_id,

customer\_id,

extract(DATETIME  from timestamp(order\_purchase\_timestamp)) as DATETIME\_1,

extract(DATETIME  from timestamp(order\_delivered\_customer\_date)) as DATETIME\_2

from `scaler-dsml-sql-373605.target\_sql.orders`

) as t

where DATETIME\_1 is not null and DATETIME\_2 is not null

)

select

distinct

c.customer\_state,

round(avg(ct1.time\_to\_delivery) over(partition by c.customer\_state),2) as avg\_time\_to\_delivery

from `scaler-dsml-sql-373605.target\_sql.customers` as c

inner join cte\_1 as ct1

on ct1.customer\_id=c.customer\_id

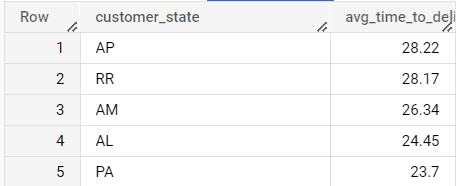
inner join `scaler-dsml-sql-373605.target\_sql.order\_items` as oe

on oe.order\_id=ct1.order\_id

order by avg\_time\_to\_delivery desc

limit 5

**Query Result:**

****

**Part (7) Top 5 states where delivery is really fast/ not so fast compared to estimated date**

**Query Code:**

with cte\_1 as (

select

order\_id,

customer\_id,

DATETIME\_DIFF(DATETIME\_2,DATETIME\_1,day) as time\_to\_delivery,

DATETIME\_DIFF(DATETIME\_3,DATETIME\_2,day) as diff\_estimated\_delivery

from (select

order\_id,

customer\_id,

extract(DATETIME  from timestamp(order\_purchase\_timestamp)) as DATETIME\_1,

extract(DATETIME  from timestamp(order\_delivered\_customer\_date)) as DATETIME\_2,

extract(DATETIME  from timestamp(order\_estimated\_delivery\_date)) as DATETIME\_3,

from `scaler-dsml-sql-373605.target\_sql.orders`

) as t

where DATETIME\_1 is not null and DATETIME\_2 is not null and DATETIME\_3 is not null

)

select\*,

from (

select

distinct

c.customer\_state,

round(avg(ct1.time\_to\_delivery) over(partition by c.customer\_state),2) as avg\_time\_to\_delivery,

round(avg(ct1.diff\_estimated\_delivery) over(partition by c.customer\_state),2) as avg\_diff\_estimated\_delivery

from `scaler-dsml-sql-373605.target\_sql.customers` as c

inner join cte\_1 as ct1

on ct1.customer\_id=c.customer\_id

inner join `scaler-dsml-sql-373605.target\_sql.order\_items` as oe

on oe.order\_id=ct1.order\_id ) as t

order by avg\_diff\_estimated\_delivery desc

limit 5

**Query Result:**

**Table

Description automatically generated**

**Q:6 Payment type analysis:**

**Part (1) Month over Month count of orders for different payment types**

**Query Code:**

select

extract (year from timestamp (o.order\_purchase\_timestamp )) as year,

extract (month from timestamp (o.order\_purchase\_timestamp )) as month,

p.payment\_type,

count(distinct o.order\_id) as cnt\_ord

from `scaler-dsml-sql-373605.target\_sql.orders` as o

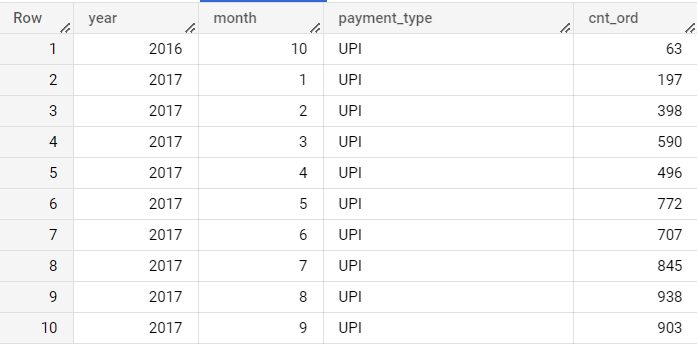
inner join `scaler-dsml-sql-373605.target\_sql.payments` p

on o.order\_id=p.order\_id

group by year,month,p.payment\_type

order by p.payment\_type,year,month

**Query Result:**

****

**Part (2) Count of orders based on the no. of payment installments**

**Query Code:**

select

distinct

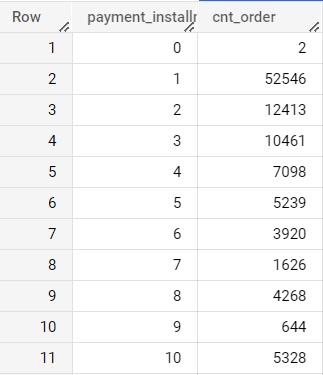
payment\_installments,

count(order\_id) over(partition by payment\_installments) as cnt\_order

from `scaler-dsml-sql-373605.target\_sql.payments`

order by payment\_installments,cnt\_order desc

**Query Result**:



**Q7) Actionable Insights:**

We have been asked to perform data analytics for the given data set, which has information of 100k orders from 2016 to 2018 made at Target in Brazil.

1. In our analysis, we have come across total unique customer States i.e., 27 and 4119 customer cities, The State MG having highest number of cities i.e., 745 and State RR having lowest number of cities i.e., 2.
2. E-commerce growing trend in the number of orders placed by customers has been observed in the first eight months of the year 2017 and 2018 and generally orders peak in August month.
3. More than 75% of the total orders preferred by customers in afternoon to night timestamp zone.
4. It has been observed that the approximate 42% of total orders placed by customers who belong to State SP and State RR having lowest order count i.e.,0.07% of total orders.
5. Top 5 States of Brazil, which have the highest customer potential i.e., SP,RJ,MG,RS and PR which represents 77% of total customers.
6. Highest percentage increase in average cost per order (payment value)  from 2017 to 2018 has been observed in month of July i.e., 19.45% and Highest percentage increase in number of orders from 2017 to 2018 is observed in the March month i.e., 808.63%
7. The State SP having lowest mean freight charges value of 15.15 while State RR has the highest mean freight value of 42.98.
8. The minimum average time of delivery of product is approximately 9 days for states like SP and maximum average time of delivery of product is approximately 29 days for states like RR and AP.
9. It has been observed that month over month and year over year customers' most preferred first payment type is Credit Card and second most preferred payment type is UPI.
10. Based on analysis, more than 50% of customers preferred payment done through one instalment and less than 10% of customers wanted to go with payment having more than 6 instalments.

**Q8) Recommendations:**

As per the findings, the organisation has a major business presence in 55% of total cities in Brazil and even though State MG has the highest number of cities but in terms of customer count it is at third position. So, to gain more customers from other States we can adopt following recommendations:

* As the standard deviation of actual time of delivery of product in Brazil is approximately 10% which is very high and it is directly related to the freight cost and overall cost of product it can easily improve through by adopting proper Network planning like optimal number of locations, size of warehouses/distribution centres, sourcing strategies and best distribution channels.
* Build strong relationship between Value Proposition and operation Strategy like; Everyday low pricing vs Cost efficiency, Customer Experience vs Responsiveness through configure to order, Product innovation vs Efficiency through outsourcing manufacturing & logistics for retainability of customer and expansion of presence in new areas.
* Just to create a unified view of demand, organisations may adopt following steps like; Generate forecast of Consumer demand; Forecast the retailer order; optimise to generate supply plan; aggregate week supply plan to generate financial plan; compare financial plan and adjust trade plan.