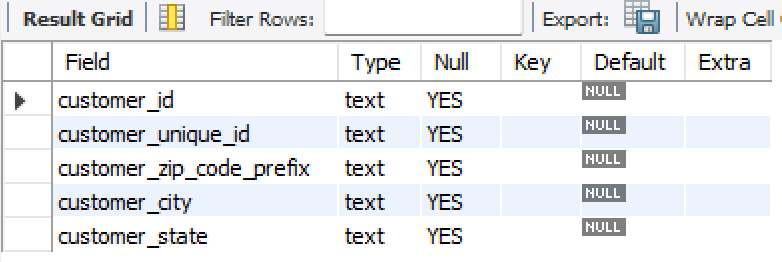
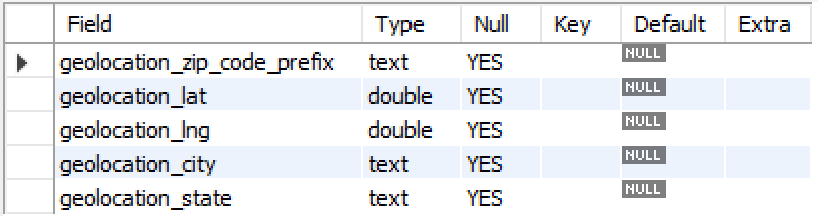
1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset**
   1. Data type of columns in a table

SQL: describe customers;

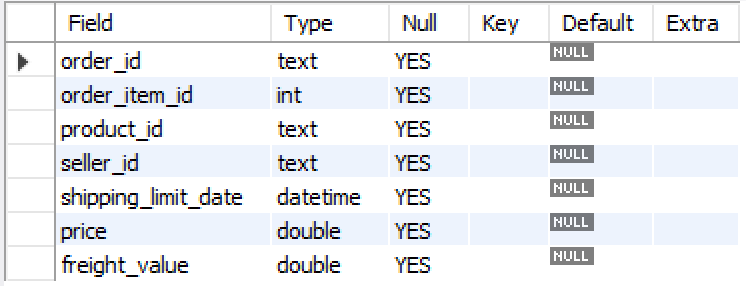
Customers:



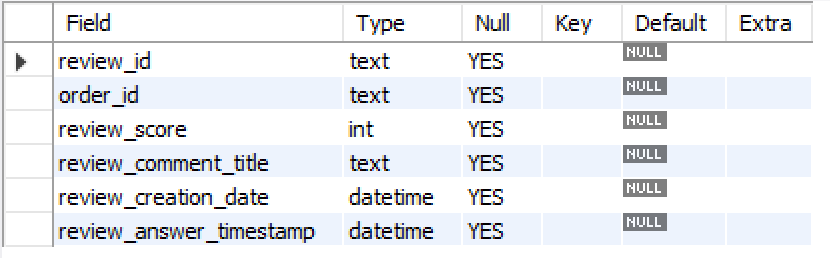
SQL: describe geolocation;



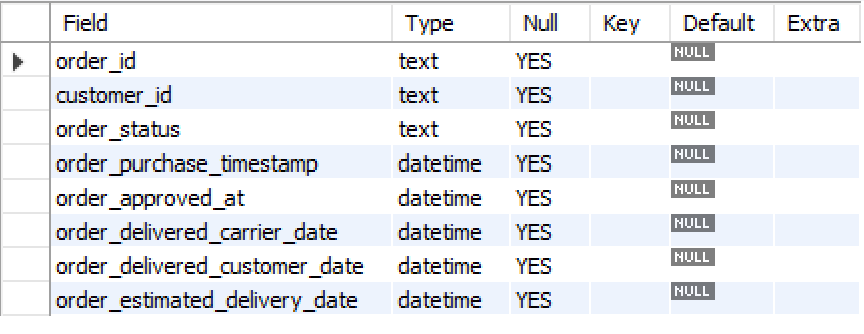
SQL: describe order\_items;



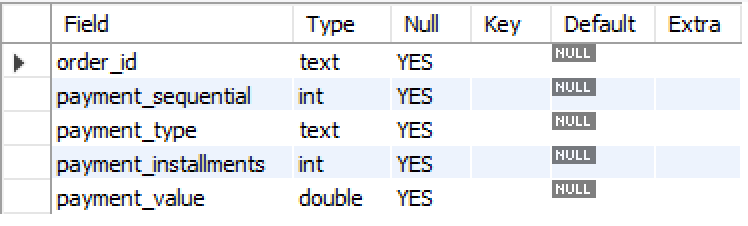
SQL: describe order\_reviews;



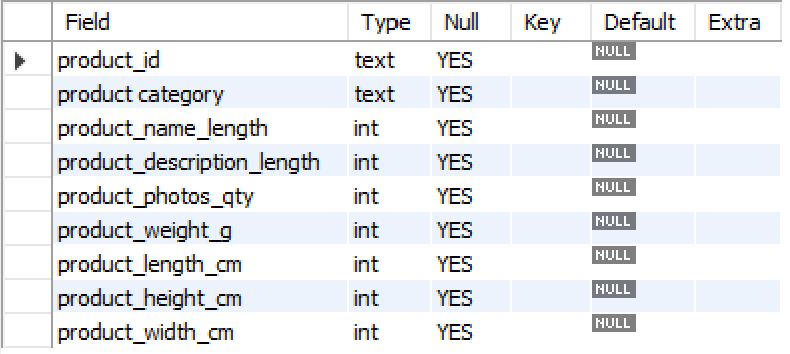
SQL: describe orders;



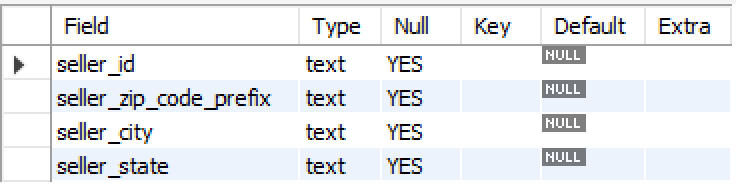
SQL: describe payments;



SQL: describe products;



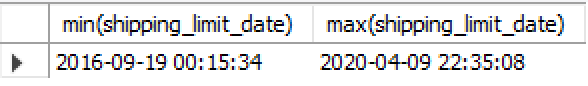
SQL: describe sellers;



* 1. Time period for which the data is given

SQL: select min(shipping\_limit\_date),max(shipping\_limit\_date)

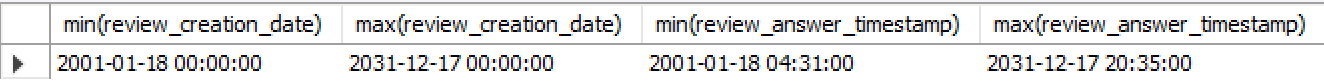
from order\_items;



Insight: seems like in order\_items table the data is available for the time frame 19-09-2016 to 09-04-2020

SQL: select min(review\_creation\_date),max(review\_creation\_date), min(review\_answer\_timestamp), max(review\_answer\_timestamp)

from order\_reviews;



Insight: seems like the order\_reviews table has data for the time frame 18-01-2001 and 17-12-2031

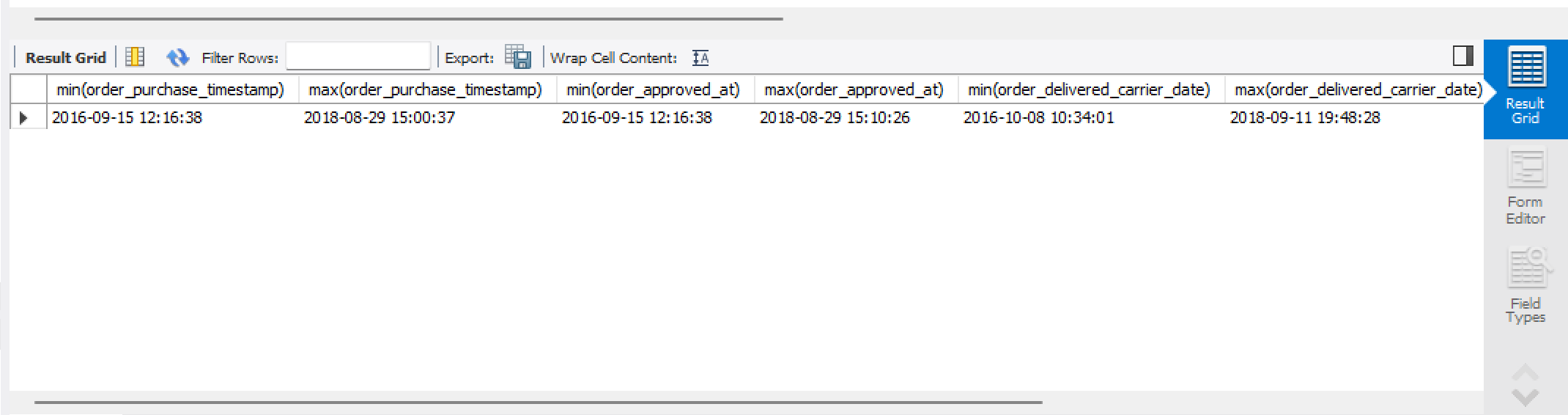
SQL: select min(order\_purchase\_timestamp),max(order\_purchase\_timestamp),

min(order\_approved\_at), max(order\_approved\_at),

min(order\_delivered\_carrier\_date),max(order\_delivered\_carrier\_date),

min(order\_delivered\_customer\_date),max(order\_delivered\_customer\_date)

from orders;



Insight: seems like in the orders table there is data for the data frame 15-09-2016 and 17-10-2018

* 1. Cities and States of customers ordered during the given period

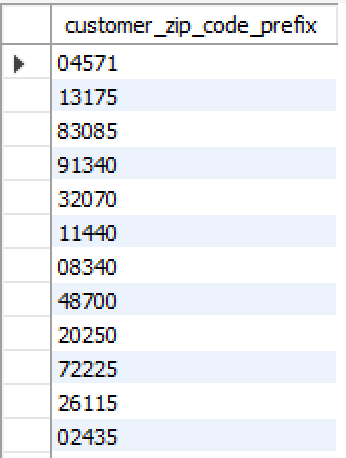
Answer:

1. Query to find out the zip code prefix for customers who placed orders:

select customer\_zip\_code\_prefix from customers

where customer\_id in

(select customer\_id from orders);



1. Query to find the city and states above customers placed each orders from:

select zip.customer\_id, geo.geolocation\_city, geo.geolocation\_state

from

(select customer\_zip\_code\_prefix, customer\_id from customers

where customer\_id in

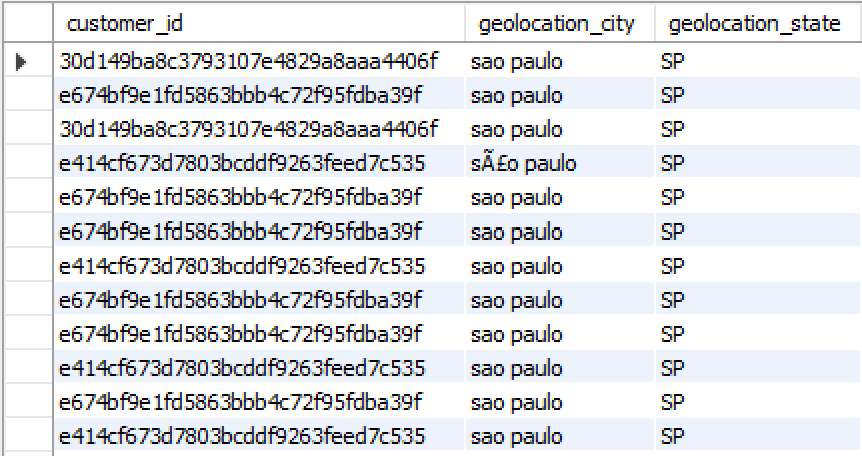
(select customer\_id from orders)

) as zip

join

geolocation as geo

on zip.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix



1. Query to find out distinct cities from where the orders were made:

select distinct(geo.geolocation\_city)

from

(select customer\_zip\_code\_prefix, customer\_id from customers

where customer\_id in

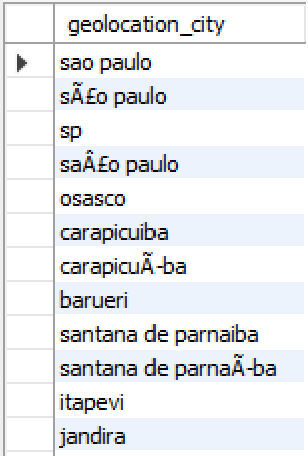
(select customer\_id from orders)

) as zip

join

geolocation as geo

on zip.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix



**Insight**: As can be seen above the same city names are entered using different character sets and as a result the data consistency / cleanliness is impacted.

**Recommendation**: standardize the way through which the city name can be entered. This could be done by either auto-suggestion, using a single character set or maybe tokenize the location if possible.

1. Query to find out the distinct states from where the orders were made:

select distinct( geo.geolocation\_state)

from

(select customer\_zip\_code\_prefix, customer\_id from customers

where customer\_id in

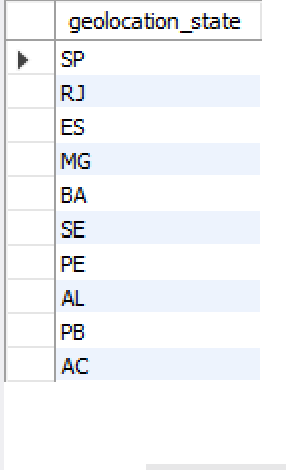
(select customer\_id from orders)

) as zip

join

geolocation as geo

on zip.customer\_zip\_code\_prefix = geo.geolocation\_zip\_code\_prefix



**Insight**: As can be seen all the orders came from these 10 states.

**Recommendation**: Need to focus on acquiring customers outside these 10 states as well to expand the business

1. **In-depth Exploration**:
   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?

Answer:

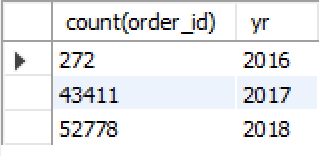
1. Query to find the total number of orders placed each year in brazil:

select count(order\_id) , year(order\_purchase\_timestamp) as yr

from orders

group by yr

order by yr



**Insight**: as can be seen from the above result between 2016 to 2018 there is a clear growing trend on e-commerce in Brazil.

1. Query to find out number of product categories sold per year:

select year(od.order\_purchase\_timestamp) as yr, count("pd.product category") as categories\_sold

from

orders as od

join

order\_items as oi

on od.order\_id = oi.order\_id

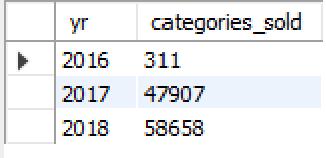
join

products as pd

on oi.product\_id = pd.product\_id

group by yr

order by yr



**Insight**: As can be seen from above output between 2016 and 2018 there are more and more product categories sold each year which reaffirms out hypothesis that there is a growing trend on e-commerce in Brazil.

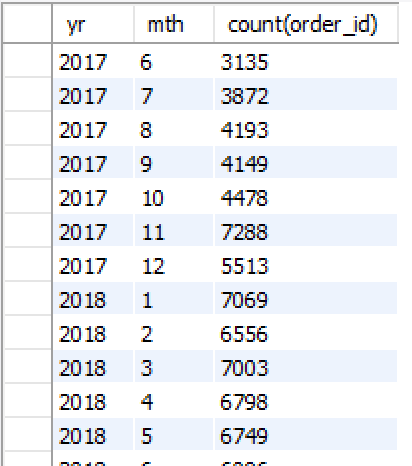
1. Query to find the total number of orders placed each month each year:

select year(order\_purchase\_timestamp) as yr ,month(order\_purchase\_timestamp) as mth, count(order\_id)

from orders

group by yr,mth

order by yr,mth



**Insight**: From the above output it seems like the last quarter (October, November and December) of the year (especially the month of November) each records the highest number of monthly sales each year.

* 1. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Answer:

Assumption: using a 24hr time format

Dawn is between 2 :00 to 5:59 hrs

Morning is between 6:00 to 11:59 hrs

Afternoon is between 12:00 to 17:59 hrs

Night is between 18:00 to 01:59 hrs

Query to find the total sales done during each quarter of the day:

select case

when hr between 2 and 6 then "Dawn"

when hr between 7 and 11 then "Morning"

when hr between 12 and 18 then "Afternoon"

else "Night"

end as dayq,

sum(sub.ct) as order\_count

from

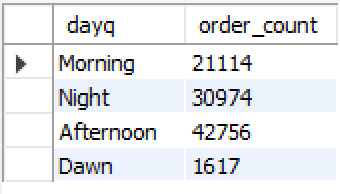
(select hour(order\_purchase\_timestamp) as hr, count(order\_id) as ct

from orders

group by hr

)as sub

group by dayq;



**Insight**: From the above query output we can conclude that Brazilian customers tend to make maximum purchases during afternoon.

1. Evolution of E-commerce orders in the Brazil region:
   1. Get month on month orders by states

select

year(od.order\_purchase\_timestamp) as yr ,

month(od.order\_purchase\_timestamp) as mth,

cst.customer\_state as state,

count(order\_id) as ct

from

orders as od

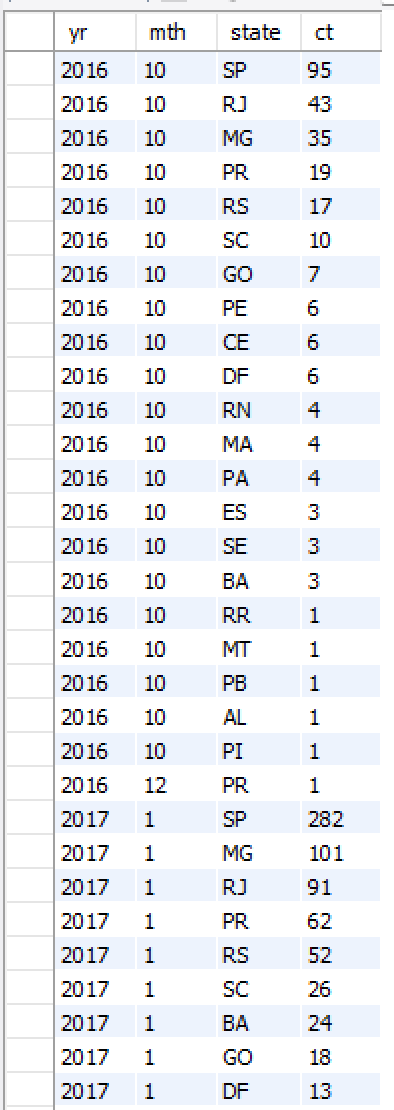
join

customers as cst

on od.customer\_id = cst.customer\_id

group by yr, mth, state

order by yr, mth, ct desc



**Insight**: as seen in the result MG, PR, RJ, RS , SC and SP almost every month drive the highest sales being in the top 6 of highest number of orders each month.

**Recommendation**: RR, AC, AP and RO are mostly present in the bottom five each month on the number of orders per month. SO more focus needs to be made to be made on these states.

* 1. Distribution of customers across the states in Brazil

select

customer\_state,

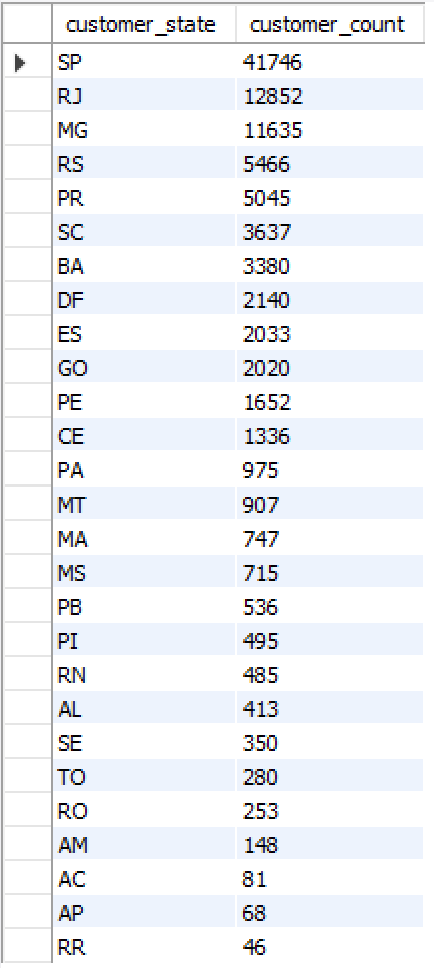
count(customer\_id) as customer\_count

from

customers

group by customer\_state

order by customer\_count desc



For a deeper analysis on the distribution of customer on each city in each state:

select

customer\_state,

customer\_city,

count(customer\_id) as customer\_count

from

customers

group by customer\_state , customer\_city

order by customer\_state , customer\_city, customer\_count desc



**Insight**: The bottom 10 states have combined less than one the count of customer in one state SP alone , Which means that the concentration on customers are lopsided in favour of a few states.

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

select

sum(pt.payment\_value)

from

orders as od

join

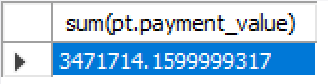
payments as pt

on od.order\_id = pt.order\_id

where year(od.order\_purchase\_timestamp) = 2017 and

month(od.order\_purchase\_timestamp) between 1 and 8

2017 total order value between Jan and Aug = 3471714.1599999317



select

sum(pt.payment\_value)

from

orders as od

join

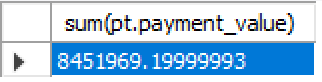
payments as pt

on od.order\_id = pt.order\_id

where year(od.order\_purchase\_timestamp) = 2018 and

month(od.order\_purchase\_timestamp) between 1 and 8

2017 total order value between Jan and Aug = 8451969.19999993



**Insight** : % increase in cost of orders from 2017 to 2018 = (8451969.199 – 3471714.16)/( 3471714.16) \* 100 = 143.452 %

* 1. Mean & Sum of price and freight value by customer state

select

cst.customer\_state as state,

avg(oi.price) as avg\_price,

sum(oi.price) as total\_price,

avg(oi.freight\_value) as avg\_freight,

sum(oi.freight\_value) as total\_freight

from

orders as od

join

order\_items as oi

on od.order\_id = oi.order\_id

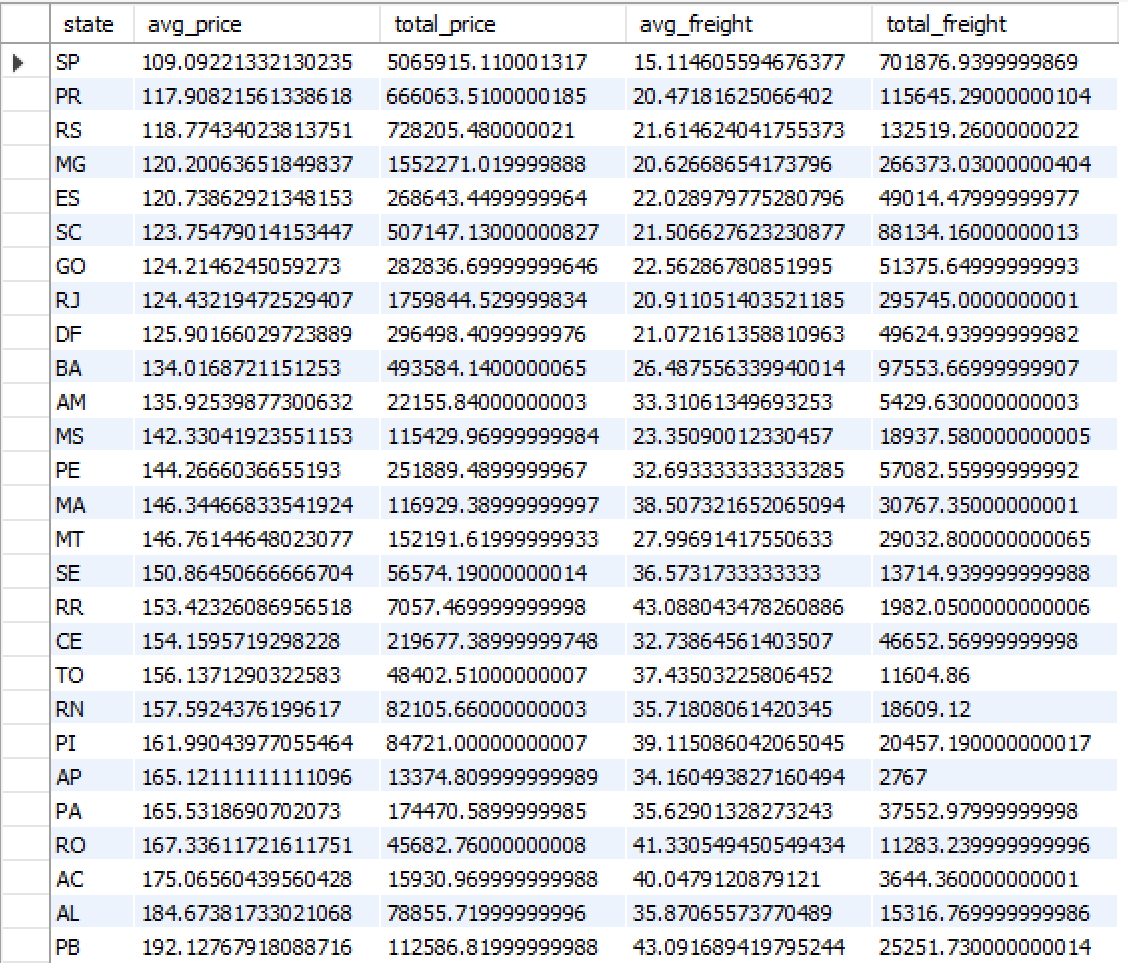
join

customers as cst

on od.customer\_id = cst.customer\_id

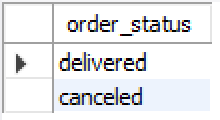
group by state

order by avg\_price, total\_price, avg\_freight, total\_freight



1. Analysis on sales, freight and delivery time
2. Calculate days between purchasing, delivering and estimated delivery

select distinct order\_status from orders



Insight: there are 2 types of order status ‘delivered’ and ‘canceled’. So we have to do the analysis only on the delivered orders

select

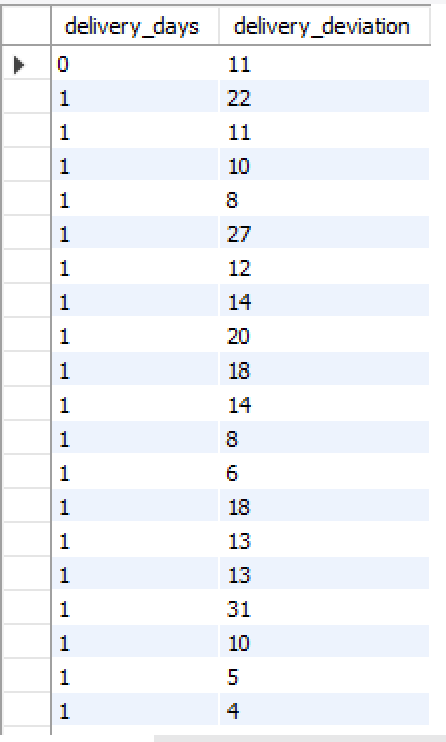
datediff(order\_delivered\_customer\_date, order\_purchase\_timestamp) as delivery\_days,

datediff(order\_estimated\_delivery\_date, order\_delivered\_carrier\_date) as delivery\_deviation

from orders

where order\_status = "delivered"

order by delivery\_days



select

max(datediff(order\_delivered\_customer\_date, order\_purchase\_timestamp)) as max\_delivery\_days,

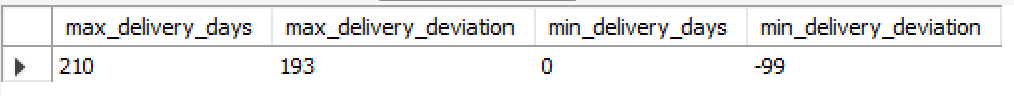
max(datediff(order\_estimated\_delivery\_date, order\_delivered\_carrier\_date)) as max\_delivery\_deviation,

min(datediff(order\_delivered\_customer\_date, order\_purchase\_timestamp)) as min\_delivery\_days,

min(datediff(order\_estimated\_delivery\_date, order\_delivered\_carrier\_date)) as min\_delivery\_deviation

from orders

where order\_status = "delivered"



1. 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

select

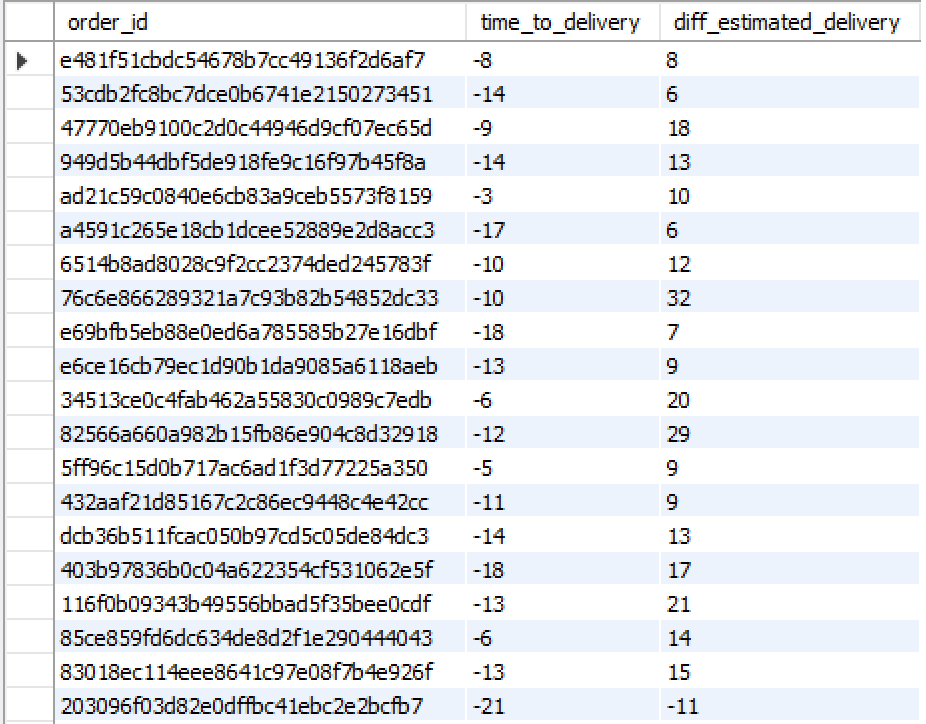
order\_id,

datediff( order\_purchase\_timestamp, order\_delivered\_customer\_date) as time\_to\_delivery,

datediff(order\_estimated\_delivery\_date, order\_delivered\_customer\_date) as diff\_estimated\_delivery

from orders

where order\_status = "delivered"



1. Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery

select cst.customer\_state,

avg(oit.freight\_value) as mean\_freight\_value,

avg(datediff( order\_purchase\_timestamp, order\_delivered\_customer\_date)) as mean\_time\_to\_delivery,

avg(datediff(order\_estimated\_delivery\_date, order\_delivered\_customer\_date)) as meandiff\_estimated\_delivery

from

orders as od

join

customers as cst

on od.customer\_id = cst.customer\_id

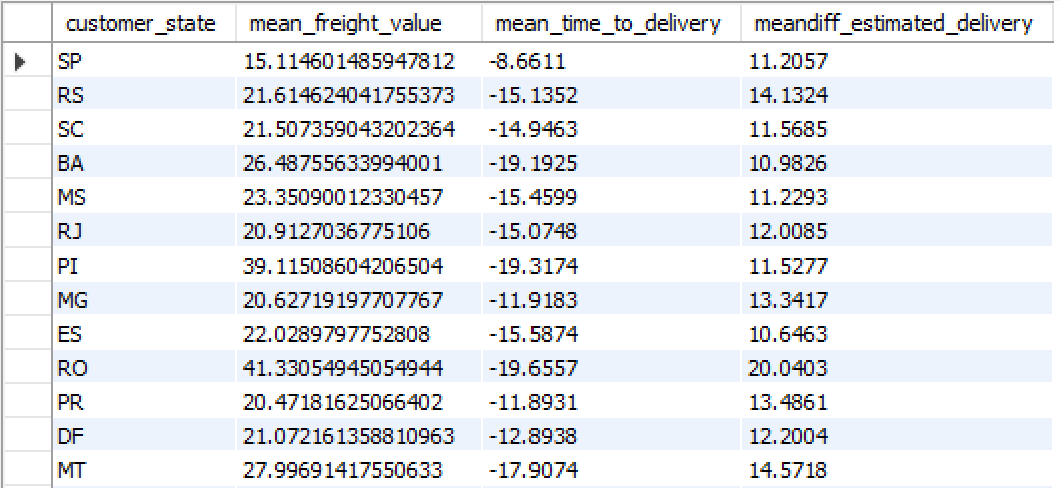
join

order\_items as oit

on od.order\_id = oit.order\_id

where od.order\_status = "delivered"

group by cst.customer\_state



1. Sort the data to get the following:
2. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

Top 5 highest avg freight value:

select

cst.customer\_state,

avg(oit.freight\_value) as avg\_freight

from

orders as od

join

customers as cst

on od.customer\_id = cst.customer\_id

join

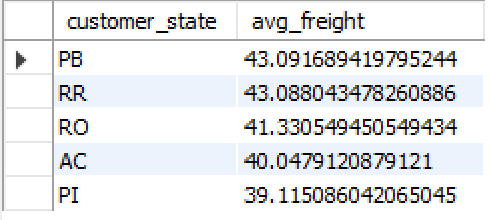
order\_items as oit

on od.order\_id = oit.order\_id

group by cst.customer\_state

order by avg\_freight desc

limit 5



Top 5 lowest avg freight value:

select

cst.customer\_state,

avg(oit.freight\_value) as avg\_freight

from

orders as od

join

customers as cst

on od.customer\_id = cst.customer\_id

join

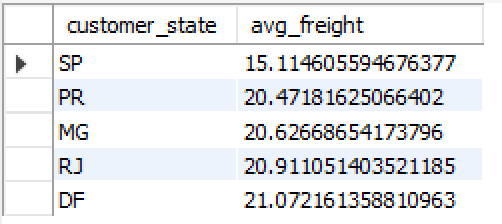
order\_items as oit

on od.order\_id = oit.order\_id

group by cst.customer\_state

order by avg\_freight

limit 5



1. Top 5 states with highest/lowest average time to delivery

Top 5 highest average time to delivery:

select

cst.customer\_state,

avg(datediff( od.order\_purchase\_timestamp, od.order\_delivered\_customer\_date)) as mean\_time\_to\_delivery

from

orders as od

join

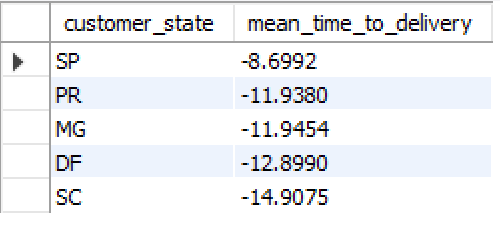
customers as cst

on od.customer\_id = cst.customer\_id

group by cst.customer\_state

order by mean\_time\_to\_delivery desc

limit 5



Top 5 lowest average time to delivery:

select

cst.customer\_state,

avg(datediff( od.order\_purchase\_timestamp, od.order\_delivered\_customer\_date)) as mean\_time\_to\_delivery

from

orders as od

join

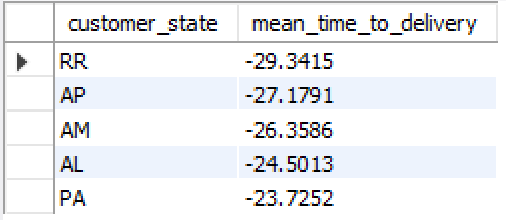
customers as cst

on od.customer\_id = cst.customer\_id

group by cst.customer\_state

order by mean\_time\_to\_delivery

limit 5



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

Top 5 slowest delivery speed:

select

cst.customer\_state,

avg(datediff( order\_estimated\_delivery\_date, order\_delivered\_customer\_date)) as delivery\_speed

from

orders as od

join

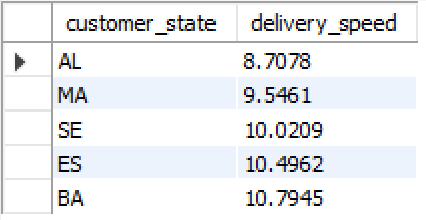
customers as cst

on od.customer\_id = cst.customer\_id

group by cst.customer\_state

order by delivery\_speed

limit 5



Top 5 fastest delivery speed:

select

cst.customer\_state,

avg(datediff( order\_estimated\_delivery\_date, order\_delivered\_customer\_date)) as delivery\_speed

from

orders as od

join

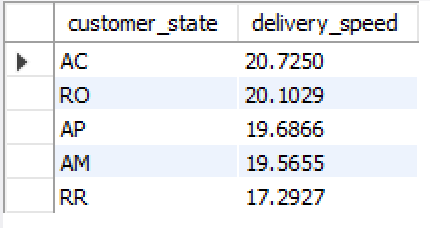
customers as cst

on od.customer\_id = cst.customer\_id

group by cst.customer\_state

order by delivery\_speed desc

limit 5



1. Payment type analysis:
   1. Month over Month count of orders for different payment types

select

year(od.order\_purchase\_timestamp) as yr,

month(od.order\_purchase\_timestamp) as mth,

py.payment\_type as py\_type,

count(od.order\_id) as order\_count

from

orders as od

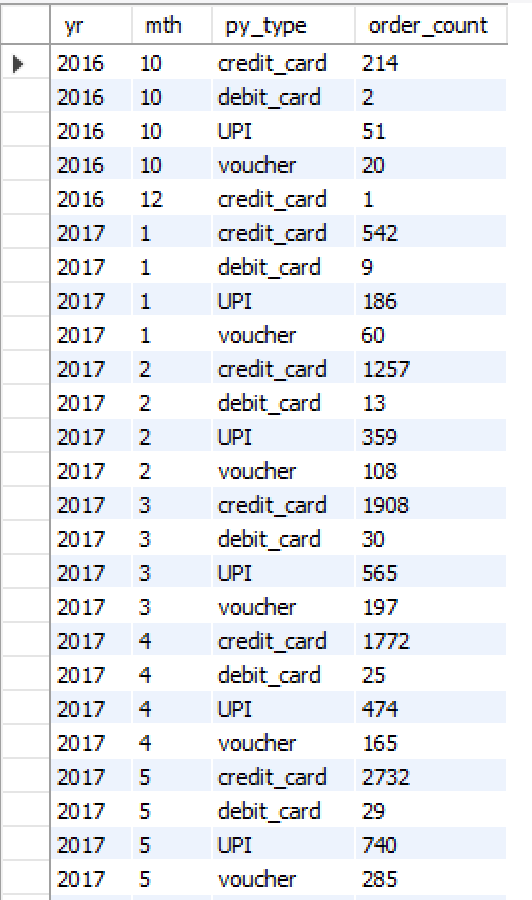
join

payments as py

on od.order\_id = py.order\_id

group by yr,mth,py\_type

order by yr,mth



**Insight**: as can be seen from above result each month credit card is the most popular mode of payment. Over the months it can be seen that the popularity of UPI as a payment mode is increasing.

**Recommendation**: debit card usage is very low each month. So special offers or cashbacks can be given to boost that mode of payment

* 1. Count of orders based on the no. of payment installments

select

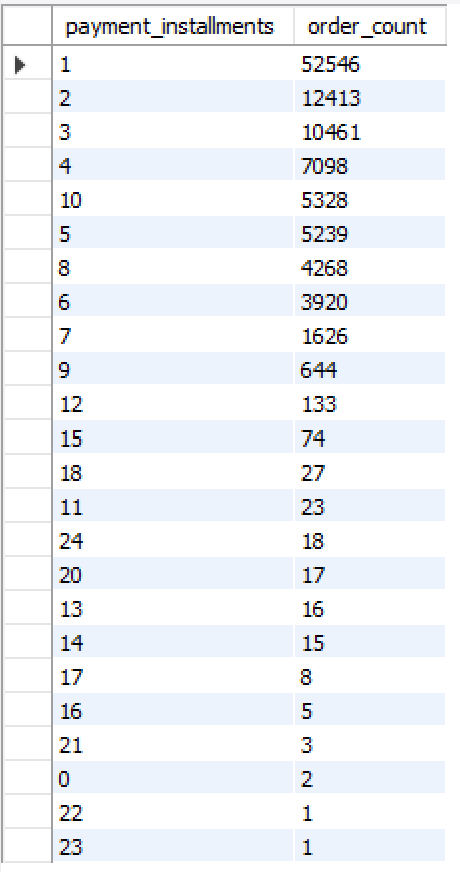
payment\_installments,

count(order\_id) as order\_count

from payments

group by payment\_installments

order by order\_count desc



**Insight**: as can be seen from the out put majority of the orders gets paid within 10 instalments