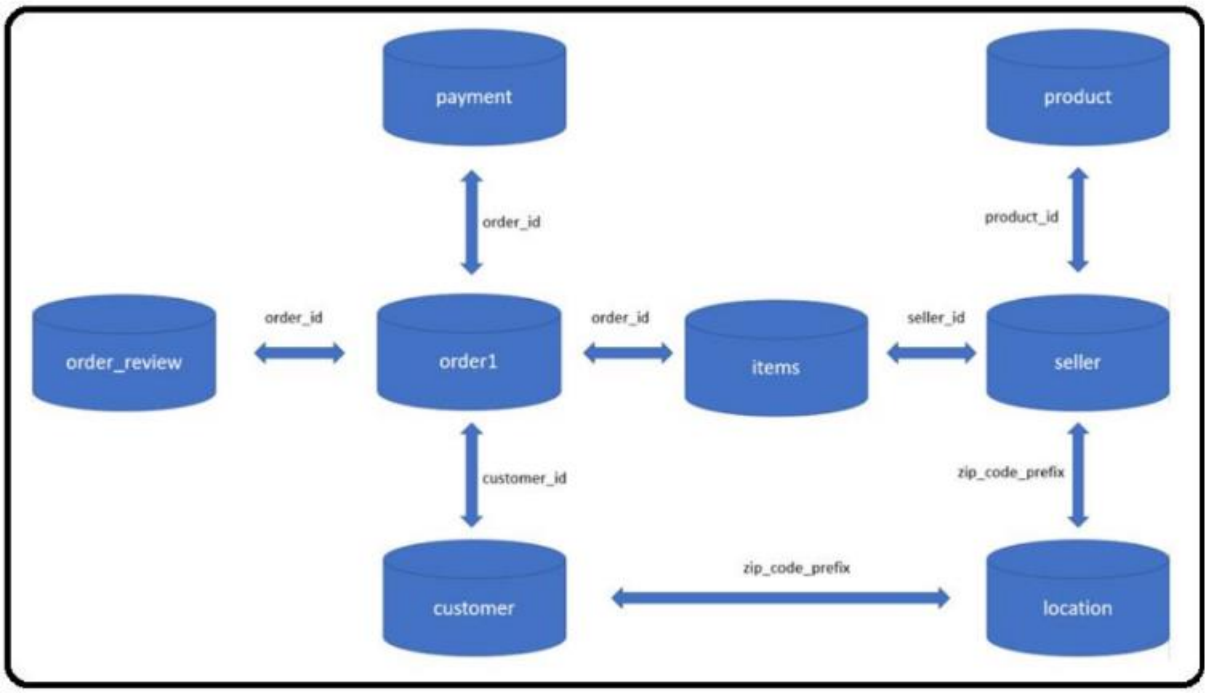
Problem Statement:

There is a Brazilian ecommerce public dataset of orders made at Olist Store. The dataset has information of multiple marketplaces in Brazil. Its features allow viewing an order from multiple dimensions: order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers, it is a geolocation dataset that relates Brazilian zip codes to latitudes /longitudes coordinates.

Below is the schema diagram and description of tables to help answer the questions:

Consider the below schema diagram/description to answer the questions:



Tables:

Customer → This table has information about the customer and its location. Use it to identify unique customers in the orders dataset and to find the orders delivery location.

Each order is assigned to a unique customer_id.

Location → This table has information Brazilian zip codes and its latitudes /longitudes coordinates.

Items → This table includes data about the items purchased within each order.

Payment → This table includes data about the orders payment options.

Order_review →This table includes data about the reviews made by the customers.

Order1→ This is the core table. From each order you might find all other information.

Product→ This table includes data about the products sold by Olist.

Seller→ This table includes data about the sellers that fulfilled orders made at Olist.

Use it to find the seller location and to identify which seller fulfilled each product

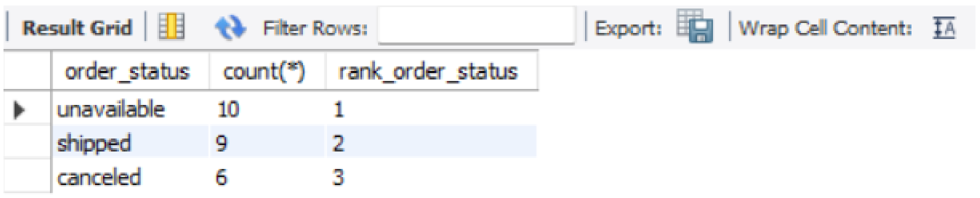
Note:

1. An order might have multiple items.
2. Each item might be fulfilled by a distinct seller.

QUESTIONS:

1) Write a SQL query to display all order statuses with their customer ids and rank each status based upon the descending order of counts in a new column as "Rank_Order_Status" for all the order statuses which are anything but delivered. Find out the top3 ranked statuses from the new column created. Comment if the orders shipped are more than the orders unavailable/lost during shipping – (6 marks)

Answer- select order_status , count(*) , rank() over (order by count(*) desc) as rank_order_status from order1 where order_status !='delivered' group by order_status order by count(*) desc limit 3;



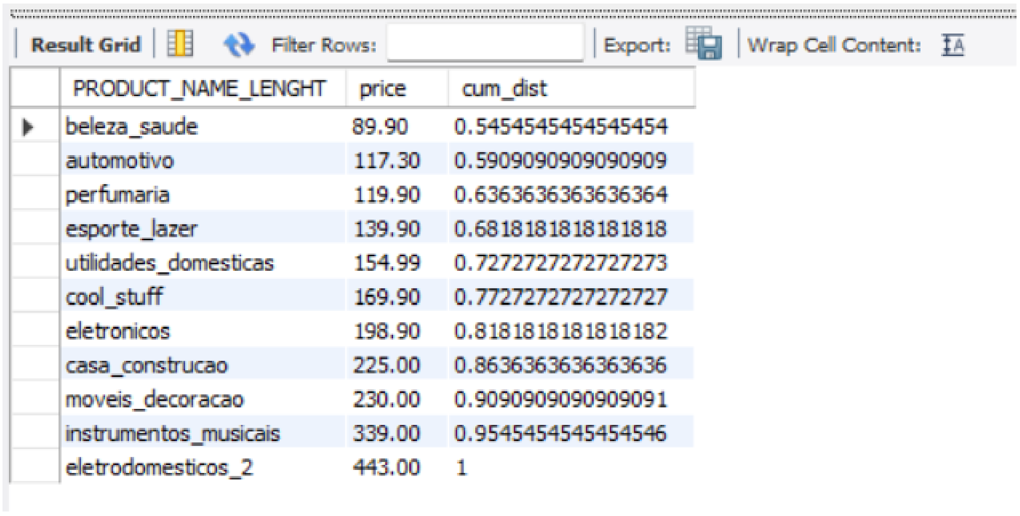
order_status	count(*)	rank_order_status
unavailable	10	1
shipped	9	2
canceled	6	3

Hence we can see orders shipped are not more than the orders unavailable/lost during shipping. So the statement written in question is not True.

2) Write a SQL query to display all product names with their respective price and the cumulative percentile for price which is greater than 0.5

select * from (Select PRODUCT_NAME_LENGHT, price , cume_dist() over (order by price) cum_dist from product p join items i using(product_id) group by PRODUCT_NAME_LENGHT) t

where cum_dist > 0.5;



PRODUCT_NAME_LENGHT	price	cum_dist
beleza_saude	89.90	0.5454545454545454
automotivo	117.30	0.5909090909090909
perfumaria	119.90	0.6363636363636364
esporte_lazer	139.90	0.6818181818181818
utilidades_domesticas	154.99	0.7272727272727273
cool_stuff	169.90	0.7727272727272727
eletronicos	198.90	0.8181818181818182
casa_construcao	225.00	0.8636363636363636
moveis_decoracao	230.00	0.9090909090909091
instrumentos_musicais	339.00	0.9545454545454546
eletrodomesticos_2	443.00	1

Or we can use **with t as** and save as a table

with t as (Select PRODUCT_NAME_LENGHT, price , cume_dist() over (order by price) cum_dist

from product p join items i using(product_id) group by PRODUCT_NAME_LENGHT)

select * from t where cum_dist>0.5;

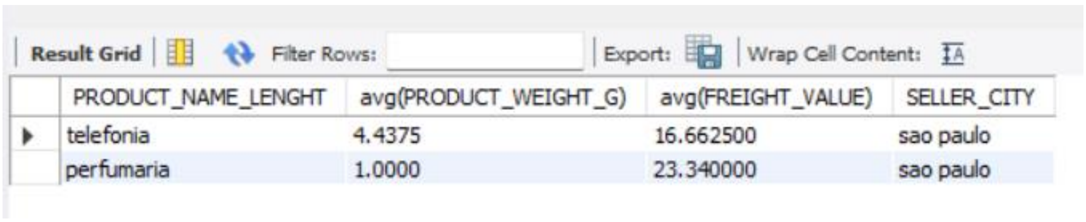
3) Write a SQL query to display average weight of products, average freight value for the products whose 2nd letter of the name is 'e' and the last letter is 'a' and for those products the seller is shipping the same from 'sao paulo'

select PRODUCT_NAME_LENGHT, avg(PRODUCT_WEIGHT_G), avg(FREIGHT_VALUE), SELLER_CITY

from product p join items i using(PRODUCT_ID) join seller s using(seller_id)

where product_name_lenght like '_e%a' and seller_city = 'sao paulo'

group by PRODUCT_NAME_LENGHT;



PRODUCT_NAME_LENGHT	avg(PRODUCT_WEIGHT_G)	avg(FREIGHT_VALUE)	SELLER_CITY
telefonia	4.4375	16.662500	sao paulo
perfumaria	1.0000	23.340000	sao paulo

4) Write a SQL query to display product length, product name, "Modified Product

Name" which is defined as:

If product length < 500 then modify the product name to all Uppercase

If 500<=product length<1500 then reverse the product name

If 1500<=product length<2500 then add “000” at the end of each product name

If 2500<=product length<3500 then replace all ‘a’ with ‘A’ in each of the product

name

If 3500<=product length<5000 then duplicate the product name 2 times without

any space

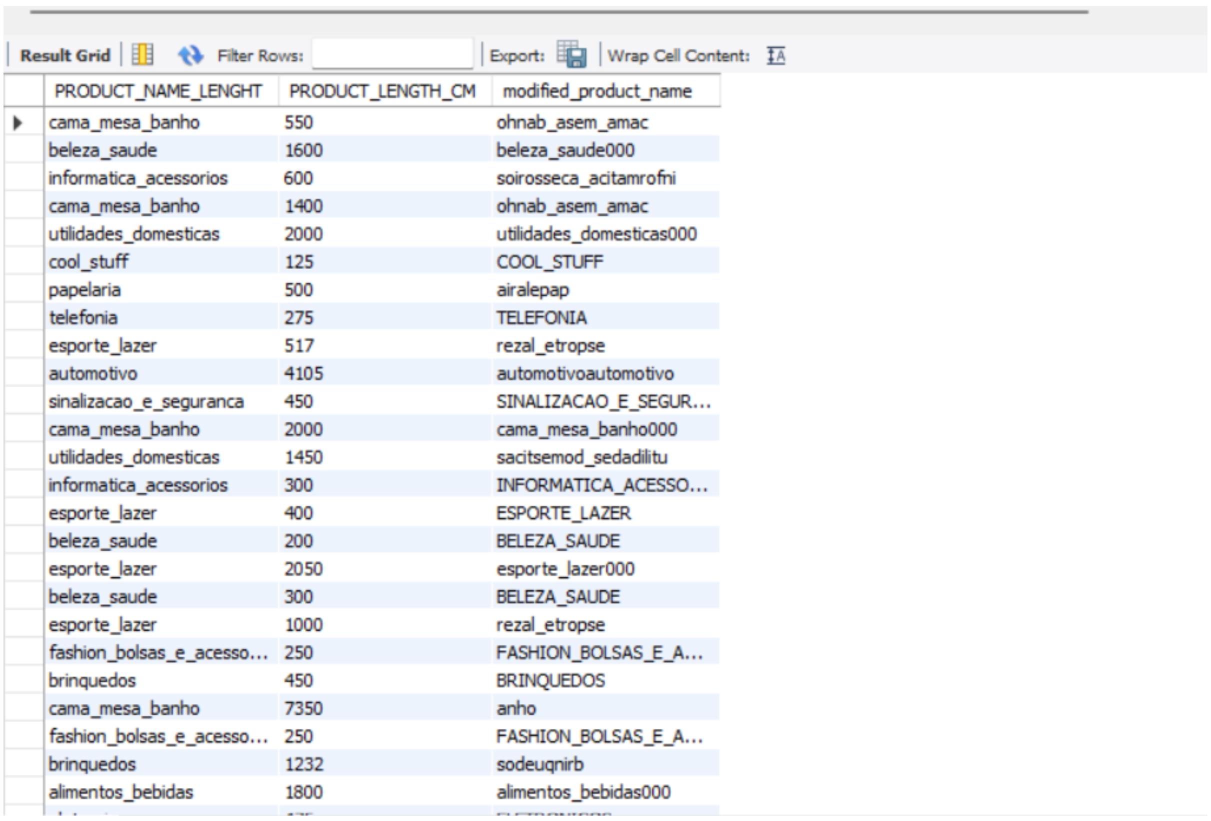
If product length >= 5000 then modify the product name to extract

last 4 characters from the product name – (6 marks)

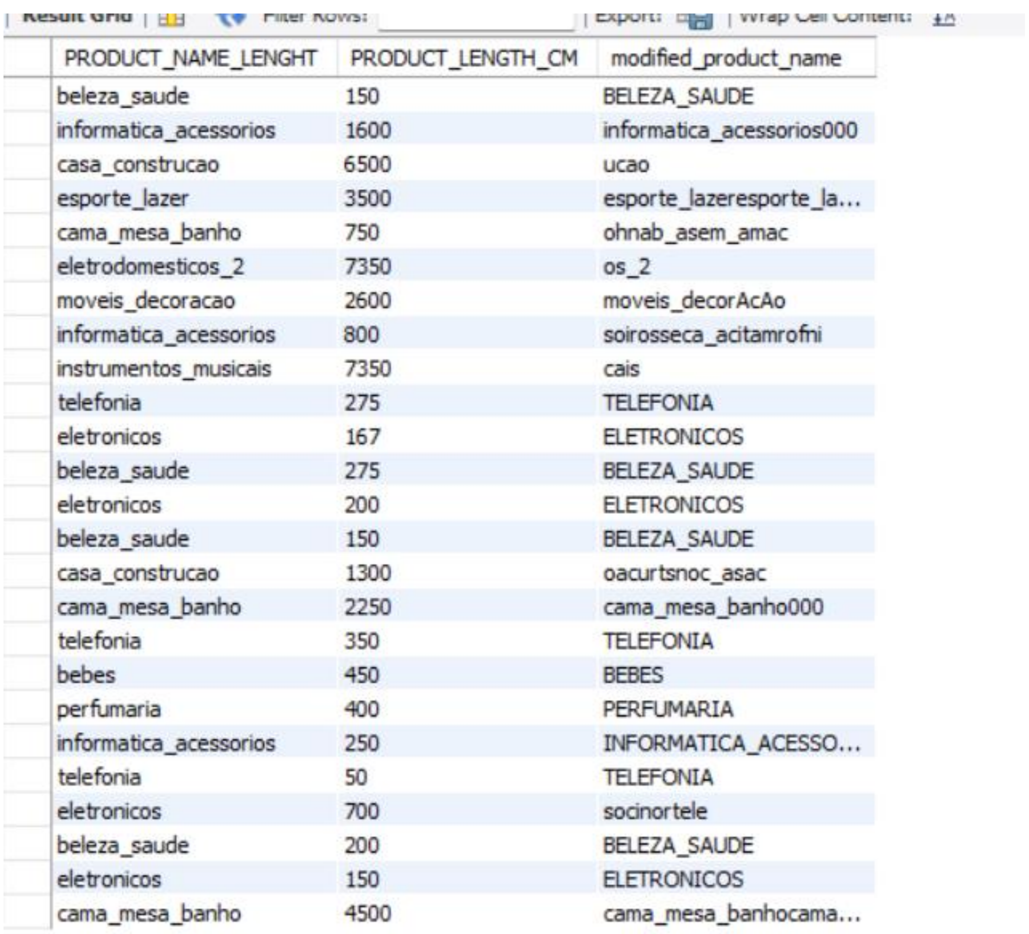
(*USE Case statement*)

ANSWER 4

```
select PRODUCT_NAME_LENGHT, PRODUCT_LENGTH_CM,
        case
                when PRODUCT_LENGTH_CM <500 then upper(PRODUCT_NAME_LENGHT)
                when PRODUCT_LENGTH_CM >=500 and PRODUCT_LENGTH_CM <1500 then
reverse(PRODUCT_NAME_LENGHT)
                when PRODUCT_LENGTH_CM>=1500 and PRODUCT_LENGTH_CM<2500 then
concat(PRODUCT_NAME_LENGHT, '000')
                when PRODUCT_LENGTH_CM>=2500 and PRODUCT_LENGTH_CM<3500 then
replace (PRODUCT_NAME_LENGHT, 'a', 'A')
                when PRODUCT_LENGTH_CM>=3500 and PRODUCT_LENGTH_CM <5000 then
repeat(PRODUCT_NAME_LENGHT,2)  #repeat - no spacing
                when PRODUCT_LENGTH_CM>=5000 then right(PRODUCT_NAME_LENGHT,4)  #or
we can use - substr(PRODUCT_NAME_LENGHT,-4)
                else 'others'  # else is optional
        end modified_product_name
from product;
```


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PRODUCT_NAME_LENGHT	PRODUCT_LENGTH_CM	modified_product_name
cama_mesa_banho	550	ohnab_asem_amac
beleza_saude	1600	beleza_saude000
informatica_acessorios	600	soirosseca_acitamrofni
cama_mesa_banho	1400	ohnab_asem_amac
utilidades_domesticas	2000	utilidades_domesticas000
cool_stuff	125	COOL_STUFF
papelaria	500	airalepap
telefonia	275	TELEFONIA
esporte_lazer	517	rezal_etropse
automotivo	4105	automotivoautomotivo
sinalizacao_e_seguranca	450	SINALIZACAO_E_SEGUR...
cama_mesa_banho	2000	cama_mesa_banho000
utilidades_domesticas	1450	sacitsemod_sedadilitu
informatica_acessorios	300	INFORMATICA_ACESSO...
esporte_lazer	400	ESPORTE_LAZER
beleza_saude	200	BELEZA_SAUDE
esporte_lazer	2050	esporte_lazer000
beleza_saude	300	BELEZA_SAUDE
esporte_lazer	1000	rezal_etropse
fashion_bolsas_e_acesso...	250	FASHION_BOLSAS_E_A...
brinquedos	450	BRINQUEDOS
cama_mesa_banho	7350	anho
fashion_bolsas_e_acesso...	250	FASHION_BOLSAS_E_A...
brinquedos	1232	sodeuqnirb
alimentos_bebidas	1800	alimentos_bebidas000


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PRODUCT_NAME_LENGHT	PRODUCT_LENGTH_CM	modified_product_name
beleza_saude	150	BELEZA_SAUDE
informatica_acessorios	1600	informatica_acessorios000
casa_construcao	6500	ucao
esporte_lazer	3500	esporte_lazeresporte_la...
cama_mesa_banho	750	ohnab_asem_amac
eletrodomesticos_2	7350	os_2
moveis_decoracao	2600	moveis_decorAcAo
informatica_acessorios	800	soirosseca_acitamrofni
instrumentos_musicais	7350	cais
telefonia	275	TELEFONIA
eletronicos	167	ELETRONICOS
beleza_saude	275	BELEZA_SAUDE
eletronicos	200	ELETRONICOS
beleza_saude	150	BELEZA_SAUDE
casa_construcao	1300	oacurtsnoc_asac
cama_mesa_banho	2250	cama_mesa_banho000
telefonia	350	TELEFONIA
bebes	450	BEBES
perfumaria	400	PERFUMARIA
informatica_acessorios	250	INFORMATICA_ACESSO...
telefonia	50	TELEFONIA
eletronicos	700	socinortele
beleza_saude	200	BELEZA_SAUDE
eletronicos	150	ELETRONICOS
cama_mesa_banho	4500	cama_mesa_banhocama...

5) Write a SQL query to display all the customers, products, and their review scores

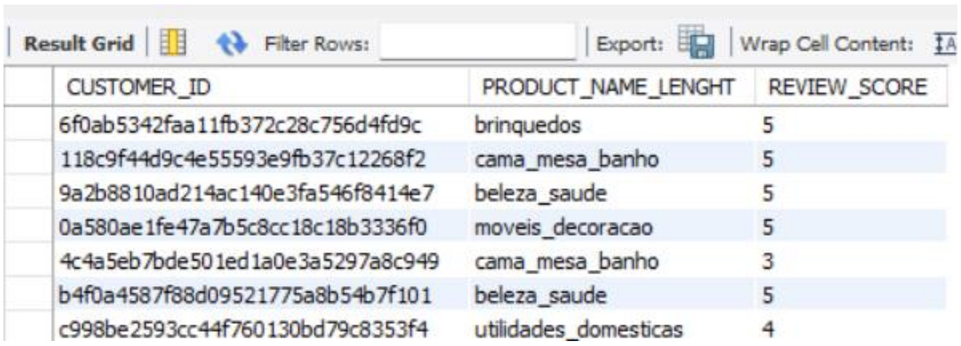
which are greater than the minimum review score

select CUSTOMER_ID , PRODUCT_NAME_LENGHT , REVIEW_SCORE from customer c join order1 o using (customer_id)

join order_review orr using(order_id) join items i using(order_id)

join product p using(product_id)

where review_score> (select min(review_score) from order_review);



CUSTOMER_ID	PRODUCT_NAME_LENGHT	REVIEW_SCORE
6f0ab5342faa11fb372c28c756d4fd9c	brinquedos	5
118c9f44d9c4e55593e9fb37c12268f2	cama_mesa_banho	5
9a2b8810ad214ac140e3fa546f8414e7	beleza_saude	5
0a580ae1fe47a7b5c8cc18c18b3336f0	moveis_decoracao	5
4c4a5eb7bde501ed1a0e3a5297a8c949	cama_mesa_banho	3
b4f0a4587f88d09521775a8b54b7f101	beleza_saude	5
c998be2593cc44f760130bd79c8353f4	utilidades_domesticas	4

And many more…..

6) Write a SQL query to display how many days does it take for the customer to get

the ordered products whose seller resides in the same city, also display the seller

and the customer city with the product and customer details

-- ANSWER 6 -----------

select customer_id, product_NAME_LENGHT,order_status,

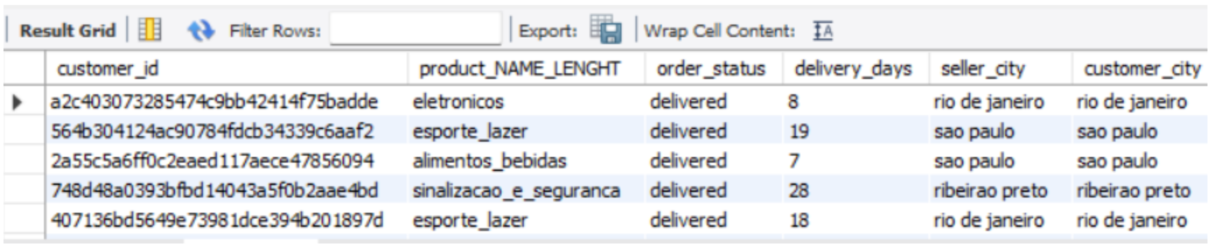
datediff(ORDER_DELIVERED_CUSTOMER_DATE, ORDER_PURCHASE_TIMESTAMP) delivery_days , seller_city, customer_city

from customer c join order1 o using (CUSTOMER_ID) join items i using (ORDER_ID) join product p using (product_id)

join seller s using(seller_id) where customer_city= seller_city;

-- # Null in output means one of the two value(_time stamp) in date diff is not present means order delivered but not yet delivered

-- # see order status shows in 'processing'

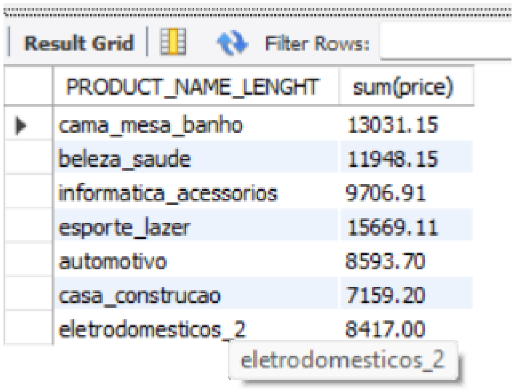


customer_id	product_NAME_LENGHT	order_status	delivery_days	seller_city	customer_city
a2c403073285474c9bb42414f75badde	eletronicos	delivered	8	rio de janeiro	rio de janeiro
564b304124ac90784fdcb34339c6aaf2	esporte_lazer	delivered	19	sao paulo	sao paulo
2a55c5a6ff0c2eaed117aece47856094	alimentos_bebidas	delivered	7	sao paulo	sao paulo
748d48a0393bfbd14043a5f0b2aae4bd	sinalizacao_e_seguranca	delivered	28	ribeirao preto	ribeirao preto
407136bd5649e73981dce394b201897d	esporte_lazer	delivered	18	rio de janeiro	rio de janeiro

And many more….

7) Write a SQL query to display all the products names with their total prices for which the total price for each product is greater than the total price for product 'eletronicos'

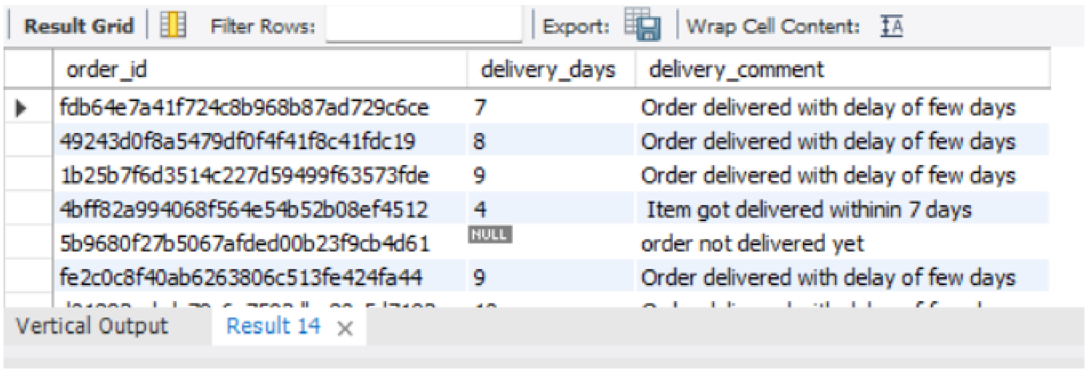
```
select PRODUCT_NAME_LENGHT, sum(price)  from product p join items i using( product_id)
 group by PRODUCT_NAME_LENGHT having sum(price)
>(select sum(price)  from product p join items i using( product_id)
where PRODUCT_NAME_LENGHT='eletronicos') ;
```



PRODUCT_NAME_LENGHT	sum(price)
cama_mesa_banho	13031.15
beleza_saude	11948.15
informatica_acessorios	9706.91
esporte_lazer	15669.11
automotivo	8593.70
casa_construcao	7159.20
eletrodomesticos_2	8417.00

8) Write a SQL query to display all the customer id's and order statuses also compute the delivery days that an item took to get delivered in a separate column as "reached_in_days" and if the computed values are null the replace with 'NA' and also create a new column as "delivery_comments" which should have the comments for the similar comparisons (a) if the item got delivered within 7 days put the same as comment (b) if the item got delivered between 7 to 30 days put the comment as "Order delivered with Delay of few days " (c) if the item got delivered after 30 days put the comment as "Order delivered with Delay of a month " (d) "Order not delivered yet"

```
with delivery_details as(
select order_id, datediff(order_delivered_customer_date, order_purchase_timestamp) delivery_days
from order1)

select * , case

                                        when delivery_days<7 then ' Item got delivered withinin 7 days '

           when delivery_days between 7 and 30 then 'Order delivered with delay of few days '

           when delivery_days > 30 then 'Order delivered with delay of a month'

           else "order not delivered yet"

        End delivery_comment

from delivery_details;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_id	delivery_days	delivery_comment
fdb64e7a41f724c8b968b87ad729c6ce	7	Order delivered with delay of few days
49243d0f8a5479df0f4f41f8c41fdc19	8	Order delivered with delay of few days
1b25b7f6d3514c227d59499f63573fde	9	Order delivered with delay of few days
4bff82a994068f564e54b52b08ef4512	4	Item got delivered withinin 7 days
5b9680f27b5067afded00b23f9cb4d61	NULL	order not delivered yet
fe2c0c8f40ab6263806c513fe424fa44	9	Order delivered with delay of few days

Vertical Output | Result 14

Many more….

We can save as a view as well by writing

Create view delivery_det as (ENTIRE ABOVE QUERY with delivery_details as …………….);

After running this we do not get any output because this query save as a view , so whenever we want that output which is coming after running the code got that without writing whole code , I simply write “” select * from delivery_det “” got same result as above

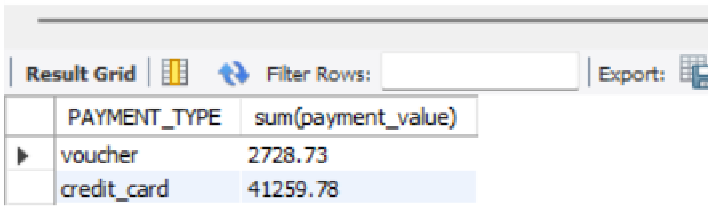
This is the benefit of view , just write one line and got same result

9) Write a SQL query to display the total payment_value for the payments done by

‘voucher’ or ‘credit card’ for all the payment_value which are less than the average

payment_value.

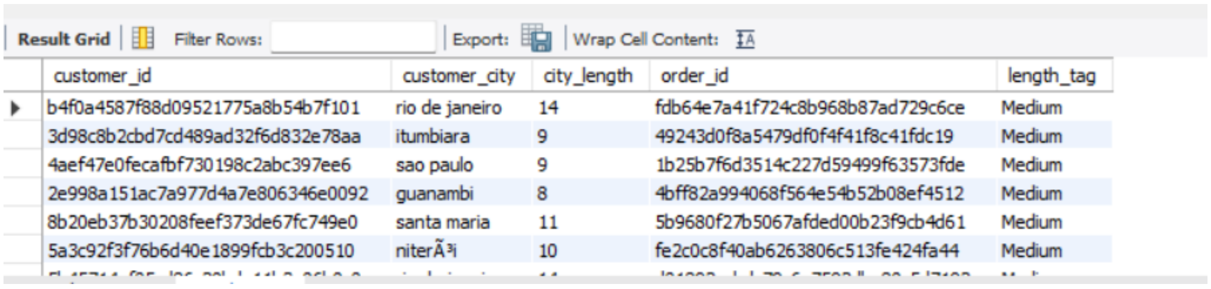
```
select PAYMENT_TYPE , sum(payment_value) from payment where PAYMENT_TYPE in
('voucher','credit_card') and

payment_value < (select avg(PAYMENT_value) from payment where  PAYMENT_TYPE in
('voucher','credit_card')) group by payment_type;
```



PAYMENT_TYPE	sum(payment_value)
voucher	2728.73
credit_card	41259.78

10) Write a SQL query To display the cutomer_ID, Order_ID, Customer_city and define a new column "city name length" where if name of the city i) has < 8 characters then it is 'small' ii) more than 8 and less than 15 it is 'medium' iii) for any other large values "large" for all the matching values in the 2 tables – (5 marks)(*Use Cross Join*)

```
with city_detail as (select customer_id, customer_city,length(customer_city) city_length, order_id
from customer c join order1 o using (customer_id))
select *, case
                                        when city_length< 8 then 'Small'
           when city_length< 15 then 'Medium'
           else 'Large'
                        end length_tag
from city_detail;
```



customer_id	customer_city	city_length	order_id	length_tag
b4f0a4587f88d09521775a8b54b7f101	rio de janeiro	14	fdb64e7a41f724c8b968b87ad729c6ce	Medium
3d98c8b2cbd7cd489ad32f6d832e78aa	itumbiara	9	49243d0f8a5479df0f4f41f8c41fdc19	Medium
4aef47e0fecafbf730198c2abc397ee6	sao paulo	9	1b25b7f6d3514c227d59499f63573fde	Medium
2e998a151ac7a977d4a7e806346e0092	guanambi	8	4bff82a994068f564e54b52b08ef4512	Medium
8b20eb37b30208feef373de67fc749e0	santa maria	11	5b9680f27b5067afded00b23f9cb4d61	Medium
5a3c92f3f76b6d40e1899fcb3c200510	niterÃ³i	10	fe2c0c8f40ab6263806c513fe424fa44	Medium

And so on ....