#project olist\_products\_ecommerce

use ecommerce;

show tables;

#1. Select customers from São Paulo

select \* from olist\_customers\_dataset where customer\_city='Sao Paulo';

#2. Select products with more than 3 photos

select \* from olist\_products\_dataset where product\_photos\_qty>3;

#3. Select orders with status ‘delivered’

select \* from olist\_orders\_dataset where order\_status='delivered';

#4 .Find customers who placed more than 5 orders

select c.customer\_id,c.customer\_city from olist\_customers\_dataset c where c.customer\_id in (select

customer\_id from olist\_orders\_dataset o group by o.customer\_id having count(o.order\_id)>5) limit 30;

select c.customer\_city, COUNT(o.order\_id) AS total\_orders FROM olist\_customers\_dataset c

inner join olist\_orders\_dataset o ON c.customer\_id = o.customer\_id

group by c.customer\_city

order by total\_orders DESC

limit 10;

#5. Find customers who placed more than 5 orders

SELECT c.customer\_id

FROM olist\_customers\_dataset c

WHERE c.customer\_id IN (

SELECT o.customer\_id

FROM olist\_orders\_dataset o

GROUP BY o.customer\_id

HAVING COUNT(o.order\_id) > 5);

#6. Select orders with payment value above average

select order\_id, payment\_value from olist\_order\_payments\_dataset

where payment\_value > (select AVG(payment\_value) from olist\_order\_payments\_dataset);

#7 .Select orders with payment value above average (with join for order details)

select o.order\_id, o.customer\_id, o.order\_status, p.payment\_value

from olist\_orders\_dataset o

inner join olist\_order\_payments\_dataset p ON o.order\_id = p.order\_id

where p.payment\_value > (select AVG(payment\_value) from olist\_order\_payments\_dataset);

#8. Select product\_id and max price per product

select product\_id, MAX(price) AS max\_price from olist\_order\_items\_dataset group by product\_id;

#9. Select most expensive product(s)

select product\_id, price from olist\_order\_items\_dataset where price = (select MAX(price) from olist\_order\_items\_dataset);

#10. Select most expensive product(s) with category name

select p.product\_id,p.product\_category\_name,oi.price FROM olist\_order\_items\_dataset oi

inner join olist\_products\_dataset p ON oi.product\_id = p.product\_id

where oi.price = (SELECT MAX(price) FROM olist\_order\_items\_dataset);

#11. Retrieve order details with customer information and payment type

select o.order\_id,o.customer\_id,c.customer\_city,c.customer\_state,p.payment\_type,p.payment\_value

from olist\_orders\_dataset o

inner join olist\_customers\_dataset c ON o.customer\_id = c.customer\_id

inner join olist\_order\_payments\_dataset p ON o.order\_id = p.order\_id;

#12. List all products ordered along with the seller and shipping cost

select oi.order\_id,oi.product\_id,oi.seller\_id,s.seller\_city,s.seller\_state,oi.freight\_value

from olist\_order\_items\_dataset oi

inner join olist\_sellers\_dataset s on oi.seller\_id = s.seller\_id;

#13. Show orders with product categories translated into English

select o.order\_id,oi.product\_id,p.product\_category\_name,pct.product\_category\_name\_english

from olist\_orders\_dataset o

inner join olist\_order\_items\_dataset oi on o.order\_id = oi.order\_id

inner join olist\_products\_dataset p on oi.product\_id = p.product\_id

left join product\_category\_name\_translation pct on p.product\_category\_name = pct.product\_category\_name;

#14. Get customer reviews and ratings along with order and product details

select r.review\_id,r.review\_score,r.review\_comment\_title,r.review\_comment\_message,r.review\_creation\_date,o.order\_id,

c.customer\_id,p.product\_id,p.product\_category\_name

from olist\_order\_reviews\_dataset r

inner join olist\_orders\_dataset o on r.order\_id = o.order\_id

inner join olist\_customers\_dataset c on o.customer\_id = c.customer\_id

inner join olist\_order\_items\_dataset oi on o.order\_id = oi.order\_id

inner join olist\_products\_dataset p on oi.product\_id = p.product\_id;

#15. max and min date

select min(shipping\_limit\_date)as min\_Date,max(shipping\_limit\_date)as max\_Date,datediff(max(shipping\_limit\_date),

min(shipping\_limit\_date)) as difference from olist\_order\_items\_dataset;

select\* from olist\_orders\_dataset;

#16 views

create view v\_order\_summary as

select o.order\_id,c.customer\_state,i.product\_id,p.payment\_type,p.payment\_value,o.order\_purchase\_timestamp

from olist\_orders\_dataset o

inner join olist\_customers\_dataset c ON o.customer\_id = c.customer\_id

inner join olist\_order\_items\_dataset i ON o.order\_id = i.order\_id

inner join olist\_order\_payments\_dataset p ON o.order\_id = p.order\_id;

select \* from v\_order\_summary where payment\_type ='credit\_card';

#17 .stored procedures. Procedure to show total sales by a seller.

DELIMITER //

create procedure GetsellerSSS(in input\_seller\_id varchar(200))

begin

select s.seller\_id,sum(i.price) as total\_sales,count(distinct i.order\_id)as total\_orders

from olist\_order\_items\_dataset i inner join olist\_sellers\_dataset s

on i.seller\_id=s.seller\_id

where s.seller\_id=input\_seller\_id

group by s.seller\_id;

end//

DELIMITER //

CALL GetsellerSSS('3442f8959a84dea7ee197c632cb2df15');

show procedure status where Db='ecommerce';

drop procedure if exists GetsellerSSS;

select \* from olist\_sellers\_dataset;

#18 functions

DELIMITER //

create function ReviewRatingLabel(score INT)

returns varchar(10)

begin

declare label varchar(10);

if score >= 4 then

set label = 'Good';

elseif score = 3 then

set label = 'Neutral';

else

set label = 'Bad';

end if;

return label;

end //

DELIMITER ;

select

review\_id,

review\_score,

ReviewRatingLabel(review\_score) AS review\_label

from olist\_order\_reviews\_dataset;

#end of project