revise: a customer can have more than one carts.

CREATE TABLE Users(

user\_id int,

name text NOT NULL,

gender text,

PRIMARY KEY (user\_id)

)

CREATE TABLE Customers(

user\_id int,

address text,

PRIMARY KEY (user\_id),

FOREIGN KEY (user\_id) REFERENCES Users(user\_id),

)

CREATE TABLE Sellers(

user\_id int,

rate float,

category text,

PRIMARY KEY (user\_id),

FOREIGN KEY (user\_id) REFERENCES Users(user\_id),

CHECK(

rate >= 0 and rate <= 5

)

)

CREATE TABLE Orders(

order\_id int,

customer\_id int NOT NULL,

PRIMARY KEY (order\_id),

FOREIGN KEY (customer\_id) REFERENCES Customers(user\_id) ON DELETE CASCADE

);

CREATE TABLE Products(

product\_id int,

name text,

description text,

seller\_id int NOT NULL,

cart\_id int,

price float,

discounted\_price float,

PRIMARY KEY (product\_id),

FOREIGN KEY (seller\_id) REFERENCES Sellers(user\_id) ON DELETE CASCADE,

FOREIGN KEY (cart\_id) REFERENCES Carts(cart\_id)

)

CREATE TABLE Coupons(

coupon\_id int,

discount float,

seller\_id int NOT NULL,

customer\_id int,

PRIMARY KEY (coupon\_id),

FOREIGN KEY (seller\_id) REFERENCES Sellers(user\_id) ON DELETE CASCADE,

FOREIGN KEY (customer\_id) REFERENCES Customers(user\_id)

)

CREATE TABLE Coupon\_applied(

coupon\_id int NOT NULL,

product\_id int NOT NULL,

PRIMARY KEY(coupon\_id,product\_id),

FOREIGN KEY(coupon\_id) REFERENCES Coupons(coupon\_id),

FOREIGN KEY(product\_id) REFERENCES Products(product\_id)

);

CREATE TABLE orders\_products(

order\_id int NOT NULL,

product\_id int NOT NULL,

quantity int NOT NULL,

PRIMARY KEY(order\_id, product\_id),

FOREIGN KEY(order\_id) REFERENCES Orders(order\_id),

FOREIGN KEY(product\_id) REFERENCES Products(product\_id)

);

CREATE FUNCTION checkcoupon() RETURNS trigger

AS $$

BEGIN

IF (NEW.product\_id in (

SELECT P.product\_id

FROM Coupons C, Products P

WHERE New.coupon\_id = C.coupon\_id and

C.seller\_id = P.seller\_id)) THEN

RETURN NEW;

ELSE

RETURN null;

END IF;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER t\_checkcoupon BEFORE INSERT ON coupon\_applied

FOR EACH ROW

EXECUTE PROCEDURE checkcoupon();

CREATE TABLE Customers\_Carts(

customer\_id int UNIQUE NOT NULL,

cart\_id int UNIQUE NOT NULL,

PRIMARY KEY (customer\_id, cart\_id),

FOREIGN KEY (customer\_id) REFERENCES Customers(user\_id),

FOREIGN KEY (cart\_id) REFERENCES Carts(cart\_id)

)

select customers.user\_id

from products, customers, carts

where customers.user\_id = carts.customer\_id and

carts.cart\_id = products.cart\_id

group by customers.user\_id

order by count(\*) DESC

limit 1;

select sellers.user\_id from sellers, products where sellers.user\_id = products.seller\_id and cart\_id in (select cart\_id from carts) group by user\_id order by count(\*) DESC limit 1;

CREATE VIEW discounted\_products as(

select products.product\_id, products.price \* (1 -(coupons.discount))

from products, coupons, coupon\_applied

where coupons.coupon\_id = coupon\_applied.coupon\_id and

coupon\_applied.product\_id = products.product\_id);

CREATE VIEW final\_price as(

(select product\_id, price

from products

where product\_id not in (select product\_id from discounted\_products))

UNION

(select \* from discounted\_products)

);