Project 1 Part 2

Kunpeng Liu (kl3070) Zichen Pan (zp2197)
Used UNI: zp2197

1. Queries

(1) select the customer with the most products

```
SELECT customers.user_id
FROM products, customers, carts
WHERE customers.use_id = carts.customer_id and
        carts.cart_id = products.cart_id
GROUP BY cutomers.user_id
ORDER BY count(*) DESC
LIMIT 1;
```

(2) select the best seller in term of the number of sold products

```
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;
```

(3) a complex one: select the customer with most expensive products after using coupons

```
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)
);


SELECT customers.user_id
FROM products, customers, carts, final_price
WHERE customers.user_id = carts.customer_id and
          carts.cart_id = products.cart_id and
          products.product_id = final_price.product_id
GROUP BY customers.user_id
ORDER BY sum(final_price.price) DESC
LIMIT 1;
```

2. Changes to schema design

(1) Drop column total_price in table Carts, which can be calculated and selected from other base data in database.

(2) Drop column discounted_price in table Products for the same reason.

(3) Create a trigger for table Coupon_applied to ensure the owners of a row of coupon and product are the same, which enforces that a coupon delivered by a seller can only be applied to his products.

```
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();
```

(4) Delete table Customers_Carts. The relationship between Customers and Carts are complicated. Carts are regarded as a weak entity to Customers and the relationship is one-to-one. Instead of creating one more table Customers_Carts to enforce the one-to-one relationship, which adds redundancy, we add UNIQUE for column customer_id in table Carts which ensures that one customer can only have one cart. However, we still cannot ensure full participation of customers. We made a tradeoff to reduce redundancy.