ONLINE SHOPPING SYSTEM

CS 331- 003

Group 10

Yash Shah (yns5@njit.edu)

Hrishikesh Sakunala(hs732@njit.edu)

Arban Haxhiu (ah742@njit.edu)

2. **System Requirements:** Online Shopping System/Portal

● Shopping portal has a different category to shop things, for example clothes or shoes. So there should be category id and category name.

● Shopping portal can be identified by its Name and has Url_address

● The different categories have brands. So it has a brand id and a brand name.

● The different brands have different products. Products have product price, product id, product name.

● Then there are customers, which have first name, last name, username, email id, password, customer id, address.

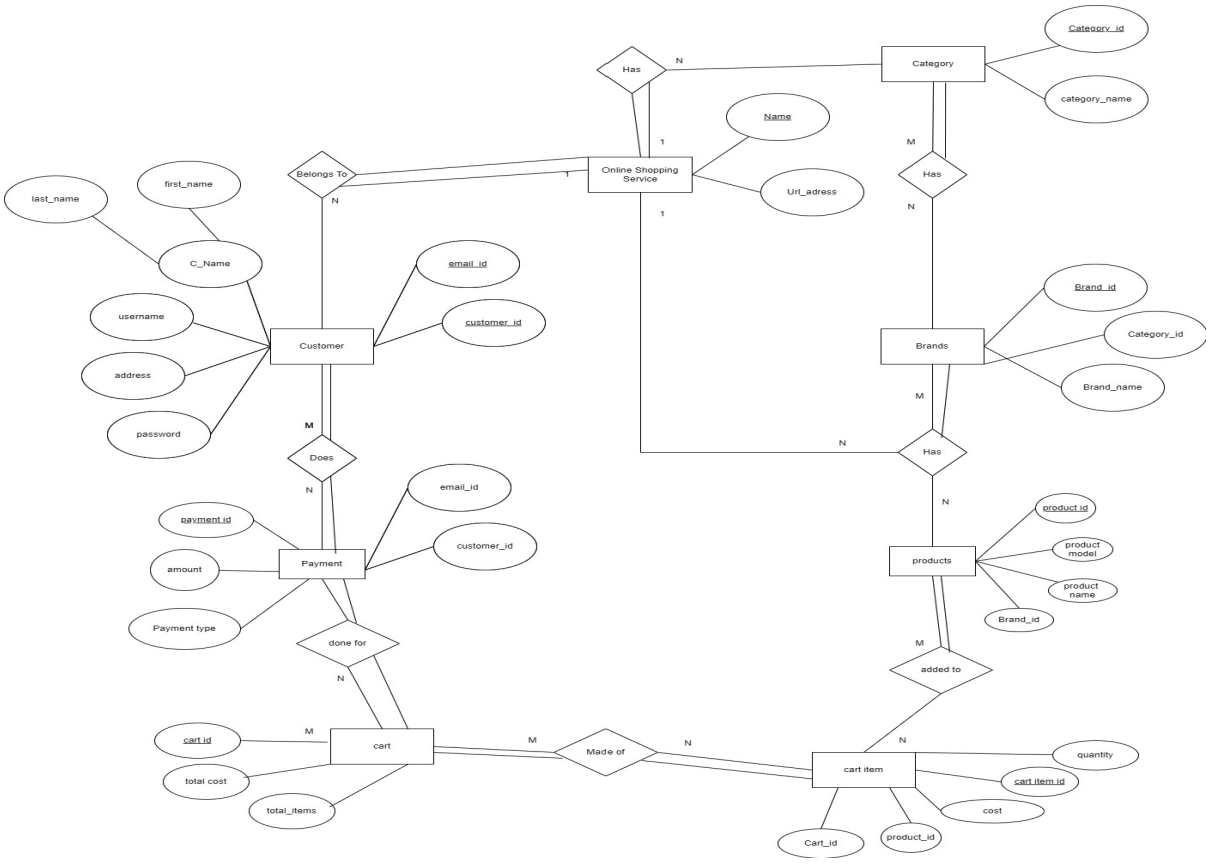
● Customer adds products to cart which have cart id and total cost.

● Customer does payment which has payment id, payment type, payment amount.

● Payment is done in the cart.

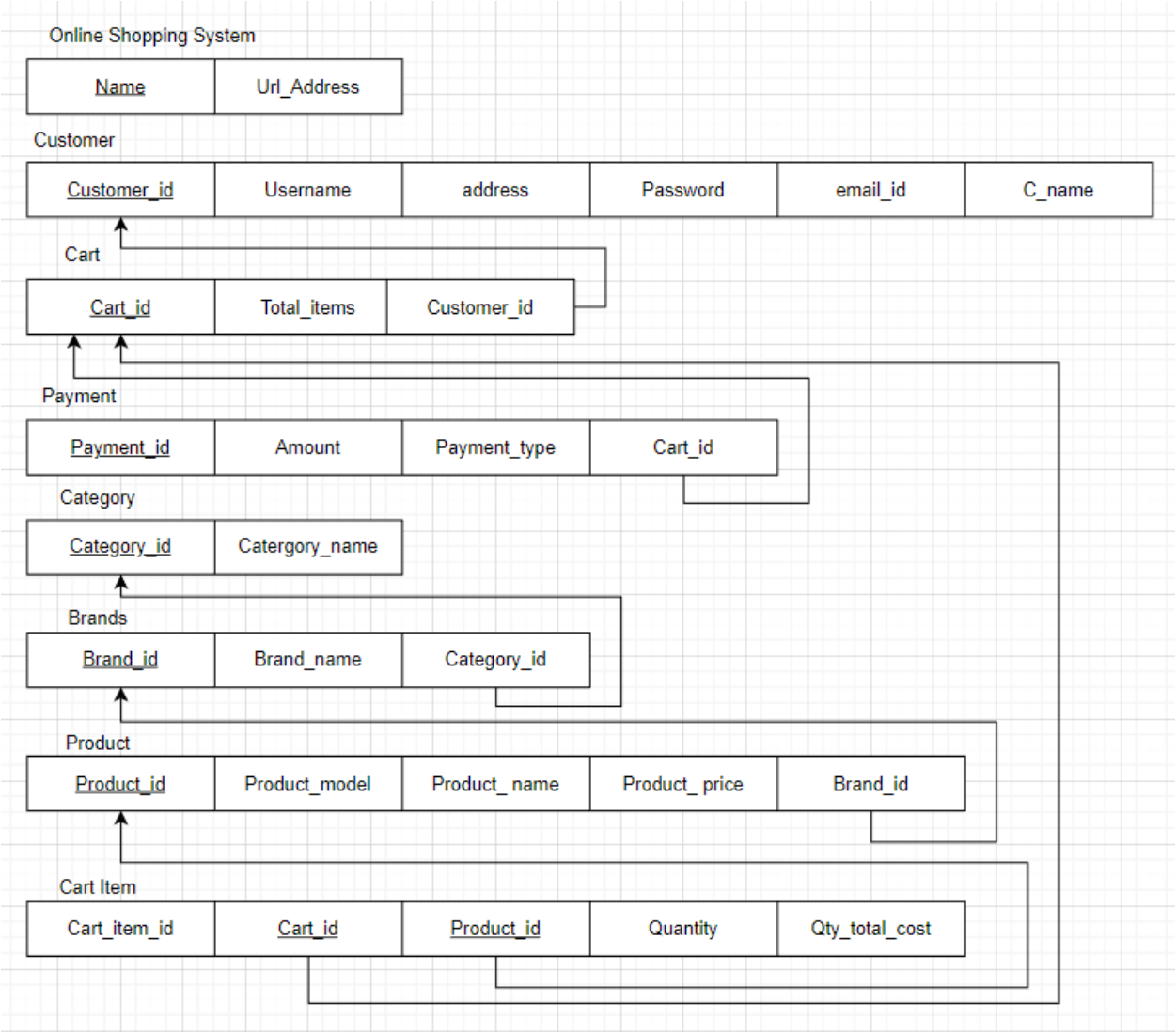
● Cart has cart item which has attributes quantity, cart item id, total cost.

ER Diagram: There were no changes in the ER diagram because it includes all the attributes and the primary key for the database schema.



Relational Schema:

The changes we made in the schema are that we added customer_id as a foreign key to Cart, so that it knows what customer has which cart. Another change is that we added cart_id as a foreign key in Payment so that it knows what is the total amount the customer needs to pay.



3.

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign keys.

Online_Shopping_System

O (Name, Url_address)

Primary key: Name

Customer

C (Customer_id, Username, address, Password,email_id, C_name)

Primary key: Customer_id

Cart

D (Cart_id, Total_items, Customer_id)

Primary key: Cart_id

Foreign key: Customer_id of Customer

Payment

P (Payment_id, Amount, Payement_type, Cart_id)

Primary key: Payment_id

Foreign key: Cart_id of Cart

Category

E (Category_id, Category_name)

Primary key: Category_id

Brands

B (Brand_id, Brand_name, Category_id)

Primary key: Brand_id

Foreign key: Category_id of Category

Product

Z (Product_id, Product_model, Product_name, Product_price, Brand_id)

Primary key: Product_id

Foreign key: Brand_id of Brands

Cart Item

Y (Cart_item_id, Cart_id, Product_id, Quantity, Qty_total_cost)

Primary key: Cart_id, Product_id

Foreign key: Cart_id of Cart, Product_id of Product

b) Provide some sample data for the relation (5 rows).

ONLINE_SHOPPING_SYSTEM

Name	Url_address
'Amazon'	'www.amazon.com'

CUSTOMER

Customer_id	Username	Address	Password	Email_id	C_name
'C1'	'yash123'	'1 mlk blvd'	'abcdef'	'abc@gmail.com'	'Yash Shah'
'C2'	'arban456'	'2 jfk blvd'	'Nkay00'	'def@gmail.com'	'Arban Haxhiu'
'C3'	'rishi900'	'10 irving st'	'Purr45'	'pur@gmail.com'	'Rishi Saku'
'C4'	'justinnn'	'4 newark ave'	'whyWe'	'just99@gmail.com'	'Justin Lam'
'C5'	'dan66'	'55 broad blvd'	'byebye'	'dan205@gmail.com'	'Dan Shaw'
'C6'	'brooly'	'265 Park st'	'brosforlife'	'brobro@gmail.com'	'Broly Brooger'

CART

Cart_id	Total_Items	Customer_id
'11111'	1	'C1'
'22222'	2	'C2'
'33333'	3	'C3'
'44444'	4	'C4'
'66666'	6	'C5'
'77777'	7	'C5'

PAYMENT

Payment_id	Amount	Payment_Type	Cart_id
'111'	300	'creditcard'	'11111'
'222'	200	'debitcard'	'22222'
'333'	300	'promocode'	'33333'
'444'	400	'paypal'	'44444'
'666'	600	'applepay'	'66666'

CATEGORY

Category_id	Category_name
'aa1'	'Clothes'
'ab2'	'Technology'
'ac3'	'Games'
'ad4'	'Cooking'
'ae5'	'Books'
'af6'	'Furniture'

BRANDS

Brand_id	Brand_name	Category_id
'5555555'	'Gucci'	'aa1'
'6666666'	'Louis Vuitton'	'ab2'
'7777777'	'Nike'	'ac3'
'8888888'	'Adidas'	'ad4'
'9999999'	'Calvin Klein'	'ae5'
'4444444'	'Givenchy'	'aa1'

PRODUCT

Product_id	Product_model	Product_name	Product_price	Brand_id
'12A'	'Slim'	'Pants'	20	'5555555'
'13B'	'Large'	'Tshirt'	20	'6666666'
'14C'	'One'	'Cap'	8	'7777777'
'15D'	'Small'	'Hoodie'	24	'8888888'
'16E'	'Medium'	'Pants'	18	'9999999'
'16.1E'	'Large'	'LongTee'	22	'9999999'
'16.2E'	'Small'	'Pants'	16	'9999999'
'16.3E'	'ExtraLarge'	'Gloves'	12	'9999999'

CART_ITEM

Cart_id	Product_id	Cart_item_id	Quantity	Qty_total_cost
'11111'	'12A'	'A1B1'	2	300
'22222'	'13B'	'C1D1'	1	100
'33333'	'14C'	'E1F1'	3	300
'44444'	'15D'	'G1H1'	2	400
'66666'	'16E'	'K1J1'	4	1000
'66666'	'16.1E'	'K1J1'	2	650
'66666'	'16.2E'	'K1J1'	1	850
'66666'	'16.3E'	'K1J1'	3	200

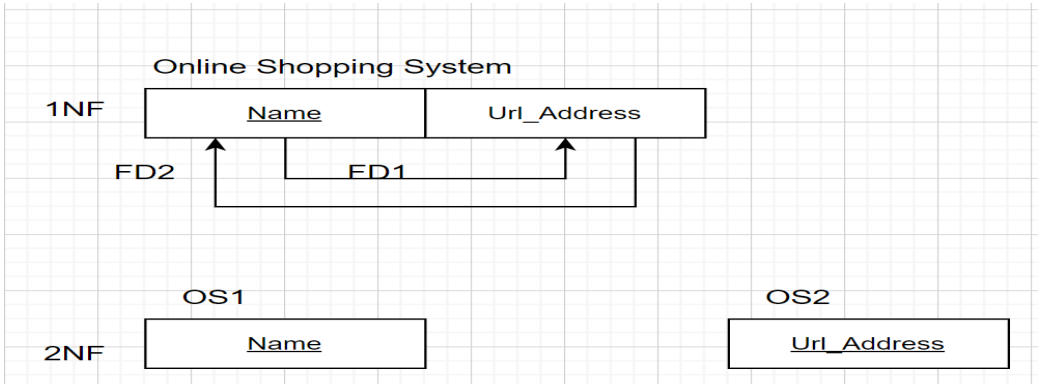
c) State the Key for the relation and write down Functional Dependencies

1. Online_Shopping_System
    a. Key: Name
    b. FDs:
        i. Name → Url_Address
        ii. Url_Address → Name
2. Customer
    a. Key: Customer_id
    b. FDs:
        i. Customer_id → {Username, address, Password, email_id, C_name}
        ii. Email_id → Password
        iii. {Email_id, Password} → {Username, Address, Password, C_name,Customer_id }
3. Cart
    a. Key: Cart_id
    b. FDs:
        i. Cart_id → {Total_items, Customer_id}
        ii. {Cart_id, Customer_id} → Total_items

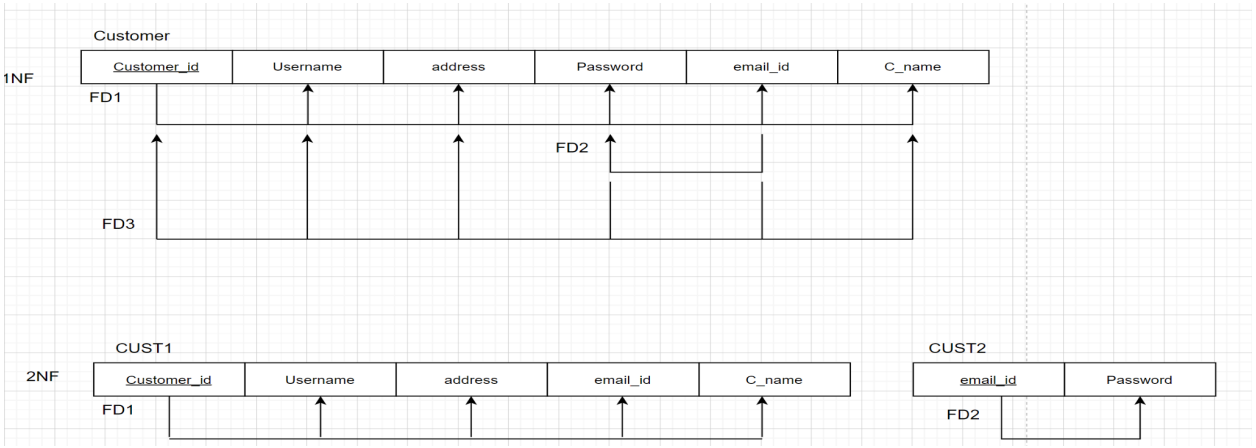
4. Payment
    a. Key: Payment_id
    b. FDs:
        i. Payment_id → {Amount, Payment_type, Cart_id}
        ii. {Payment_id, Cart_id} → {Amount, Payment_type}
5. Category
    a. Key: Category_id
    b. FDs:
        i. Category_id → Category_name
6. Brands
    a. Key: Brand_id
    b. FDs:
        i. Brand_id → {Brand_name, Category_id}
        ii. {Brand_id, Category_id}→ Brand_name
7. Product
    a. Key: Product_id
    b. FDs:
        i. Product_id → {Product_model, Product_name, Product_price, Brand_id}
        ii. {Product_id, Brand_id} → {Product_model, Product_name, Product_price}
        iii. Product_name → Product_price
8. Cart Item
    a. Key: {Cart_id, Product_id}
    b. FDs:
        i. {Cart_id, Product_id} → {Cart_item_id, Quantity, Qty_total_cost}
        ii. Cart_item_id → Qty_total_cost

d) State that this relation is in 3NF.

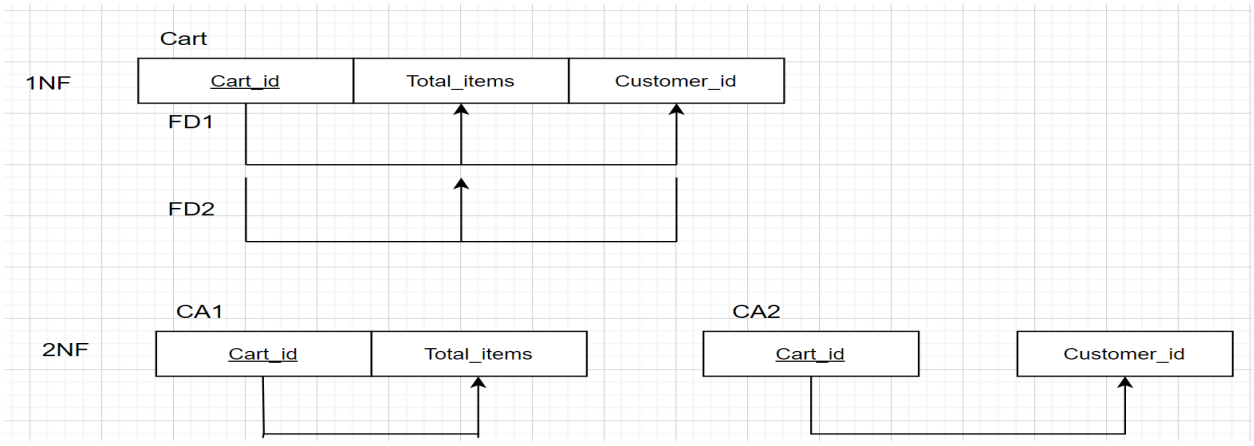
This relation is in 3NF because no non-prime attribute is transitively dependent on the primary key.



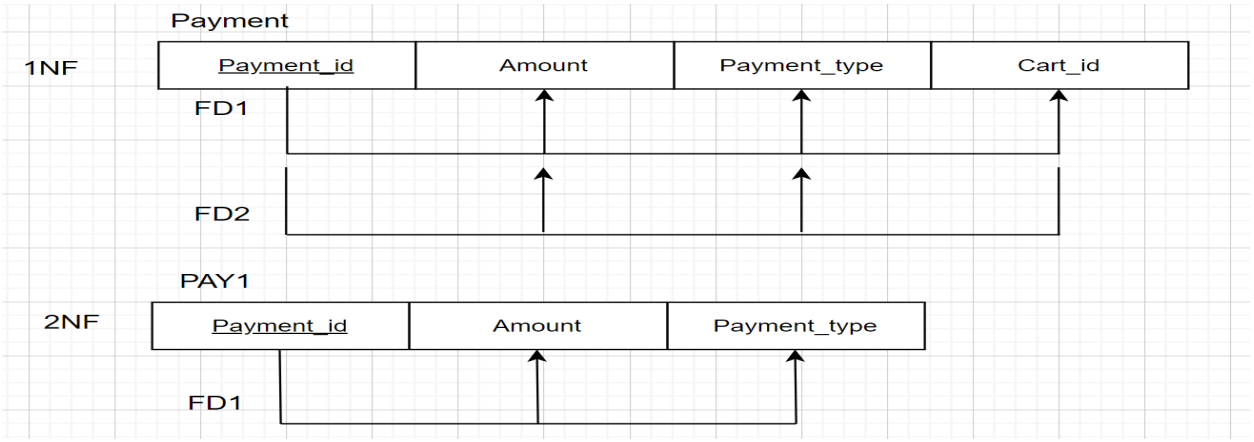
This relation wasn't in 3NF so it is converted into it. Now it doesn't contain a partial- key dependency or doesn't contain a transitive dependency.



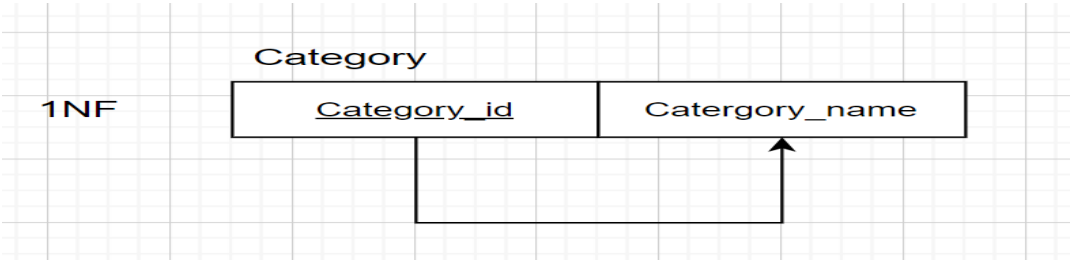
This relation is in third normal form (3NF) because it is in 2NF and no non-prime attribute is transitively dependent on the primary key.



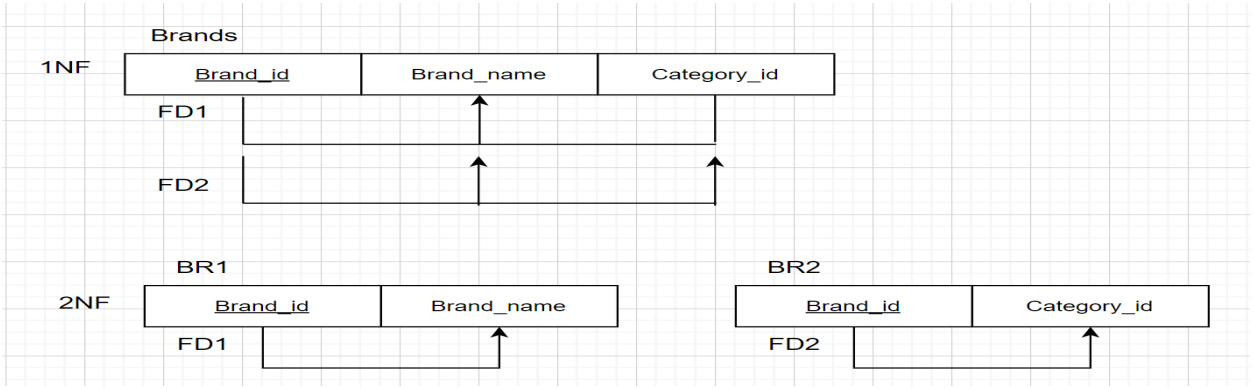
This relation is in third normal form (3NF) because it doesn't contain a partial- key dependency or doesn't contain a transitive dependency.



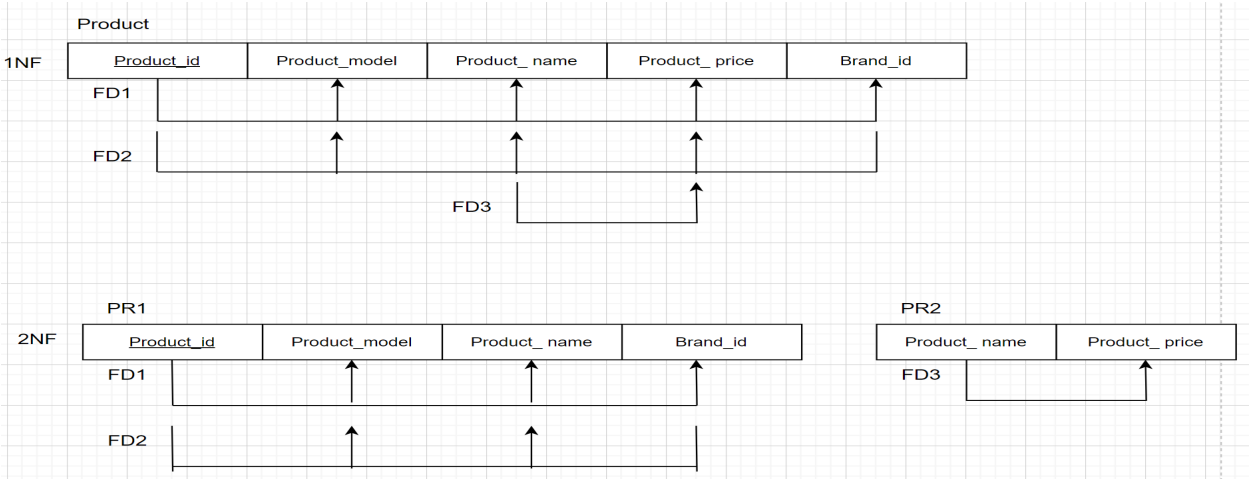
Even though this relation is in 1NF it is also in third normal form (3NF) because it doesn't contain a partial- key dependency or doesn't contain a transitive dependency.



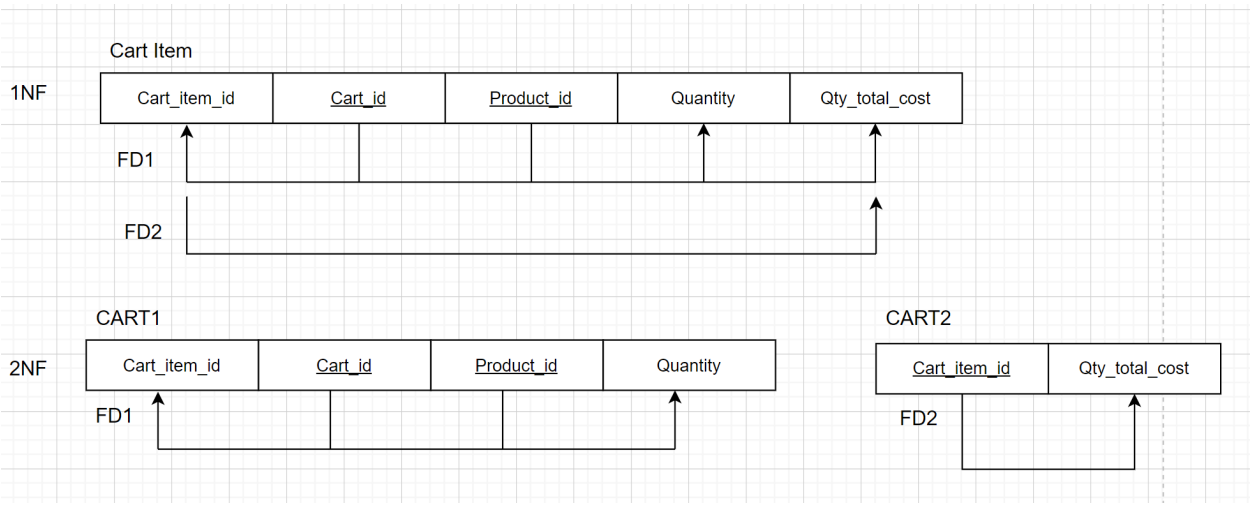
This relation is in third normal form (3NF) because it doesn't contain a partial- key dependency or doesn't contain a transitive dependency.



This relation is in third normal form (3NF) because it is in 2NF and no non-prime attribute is transitively dependent on the primary key.



This relation is in third normal form (3NF) because it is in 2NF and no non-prime attribute is transitively dependent on the primary key.



4. Write four SQL Queries;

- 1 contains GROUP BY

This query retrieves the amount of products a brand has by counting the product ids and printing out the amount of products next to the brand id.

```
SELECT COUNT(Product_id), Brand_id
FROM PRODUCT
GROUP BY Brand_id;
```

```
COUNT(PRODUCT_ID) BRAND_ID
----------------- ------------------------------
                1 7777777
                1 8888888
                4 9999999
                1 5555555
                1 6666666
```

- 1 contains GROUP BY and HAVING

This query retrieves the amount of products a brand has by counting the product ids and printing out the amount of products next to the brand id which have a count greater than 2.

```
SELECT      COUNT(Product_id), Brand_id
FROM        PRODUCT
GROUP BY    Brand_id
HAVING      COUNT(Product_id)>2;
```

```
COUNT(PRODUCT_ID) BRAND_ID
----------------- ------------------------------
                4 9999999
```

- 1 contains nested query with ALL

This query retrieves the Cart item id, the quantity total cost, and the cart id of the tuple that ahs the lowest Qty_total_cost compared to the Qty_total_cost of all Cart items with a cart id of 66666.

```
SELECT      Cart_item_id, Qty_total_cost, Cart_id
FROM        CART_ITEM
WHERE       Qty_total_cost < ALL
                    (SELECT Qty_total_cost
                    FROM CART_ITEM
                    WHERE Cart_id= '66666');
```

```
CART_ITEM_ID                 QTY_TOTAL_COST CART_ID
---------------------------- -------------- ------------------------------
C1D1                                    100 22222
```

- 1 contains nested query with IN

This query retrieves all of the attributes of tuples in the relation CART_ITEM where the Cart_id is the same as the Cart_id in the CART relation where the Total_items is greater than 3.

```
SELECT *
FROM CART_ITEM
WHERE Cart_id IN (SELECT Cart_id
        FROM Cart
        WHERE Total_items > 3);
```

CART_ID	PRODUCT_ID	CART_ITEM_ID	QUANTITY	QTY_TOTAL_COST
44444	15D	G1H1	2	400
66666	16E	K1J1	4	1000
66666	16.1E	K1J1	2	650
66666	16.2E	K1J1	1	850
66666	16.3E	K1J1	3	200

5. a) the group's experience with the project (which steps were the most difficult? Which were the easiest? what did you learn that you did not imagine you would have? if you had to do it all over again, what would you have done differently?)

It was a good experience learning how a database works and the fundamentals that are required to create a database. The easiest thing in the project was to write the SQL statements because if the syntax was correct the output would be shown. The most difficult part of the project was creating the relational schema because it had to be created based on logic rather than just copy and paste like the SQL statements, everything had to be connected and determining the foreign keys was difficult because it didn't have to look redundant. Few things that we learned from the project were that there needs to be a connection between different tables so that the nested SQL statements work. One thing we would have done differently from the start of the project was to spend more time on creating the relational schema because everything is dependent on that or choose a database schema that has less tables than we had because more tables create confusion.

b) any final comments and conclusions.

It was great to learn about databases and implement one itself built by a team so that we could have some experience about the real world even though it was very less data. It would have been more fun and interesting if we could code something with the databases as we did at the end of the course.