Group id: #90

User Manual

Our final project includes three interfaces:

- The **first interface** is the customer and employee log-in interface.

Customers and employees can both login using this interface.

- The **second interface** is used by customer. There are 4 main functions.
    - Customers can search certain product by its p_name or p_id
    - Customers can add items they want to purchase into the shopping cart.
    - Customers can checkout using this interface.
    - Customers can check the inventory of certain product.

- The **third interface** is used by employees. There are 3 main functions.
    - Employees can add or delete customers or members to the database.
    - Employees can search certain product by its p_name or p_id
    - Employees can add or delete products into the database.

How our final schema differ from the schema we turned in:

- We delete the **manufacturer** table and the relationship between product and manufacturer which is **produce**. We then put two attributes of manufacturer which are “m_id” and “m_price” to the product table and also put the purchase price to the product table.

  Our original schema looks like
  Our new schema is Product(p_id, category, sale Price, Inventory, Unit, p_name, rating, color, PurchasePrice, p_size, BrandName, thumbnail)
    - Represents: entity set Product
    - Primary key: p_id
    - Domain:
        - p_id: has to be in the form of digits and size of char[10]
        - Category: a string
        - sale Price: must be a float > 0
        - Inventory: must be an integer > 0
        - Unit: a quantity unit in the form of string
        - P_name: has to be a size of char[10]
        - Rating: a string
        - Color: a string
        - Purchase Price: must be a float > 0
        - P_size: a string
    - Constraints: must have a corresponding manufacturer

Functional Dependency

Transaction_DealWith_Pay (TransactionNum, dateTime, Payment Method, **email, e_id,** amount)

TransactionNum -> dateTime, payment Method, email, e_id, amount

// A transactionNum can identify a transaction, and implies other attributes of a transaction such as transaction dateTime, the payment method.

// This transaction will associate with a certain employee, a customer identified by his/her email, and the products purchased in this transaction.

Include(**Transaction#**, **p_id**)

Null

Customer (**email**, c_name, address, password)

Email -> address, c_name,password

// An email can identify a customer, and implies other attributes of a customer including customer name and address. password that corresponds to the employee's e_id

Member (**email,** points, rewardRate)

Email -> points, rewardRate

// An email can identify a member, and implies other attributes of a member including
// customer points and rewardRate.

Product (**p_id**, category, salePrice, purchasePrice, Inventory, p_name, p_size, color, rating, BrandName, thumbnail)

P_id -> category, sale Price, Inventory, p_name

// A p_id can identify a product, and implies other attributes of a product including
// its category, inventory, salePrice, purchasePrice, Inventory, p_name, p_size, color, rating, BrandName, thumbnail.

Employee(**e_id,** e_name, salary, startDate,e_type,password)

e_id -> salary, startDate, e_type, e_name,password

// the e_id is the employee id, which can identify an employee, and it can imply
// the name, his/her salary, the startDate of this employee, and the job type, password
// corresponds to the employee's e_id

SQL used:

Selection and projection query:

```
select p_id, p_name, saleprice from product where category like 'Pets'
```

Join query:

```
select p_id from PRODUCT p where not exists ((select distinct c.EMAIL from
CUSTOMER c) minus (select distinct EMAIL from TRANSACTION_DEALWITH_PAY
tdp natural join INCLUDE i where p.P_ID like i.P_ID))
```

Division query:

```
select p_id from PRODUCT p where not exists ((select distinct c.EMAIL from
CUSTOMER c) minus (select distinct EMAIL from TRANSACTION_DEALWITH_PAY
tdp natural join INCLUDE i where p.P_ID like i.P_ID))
```

Aggregation query:

```
select P_ID, total from (select P_ID, sum(QUANTITY) total from INCLUDE group by
P_ID order by total desc) where rownum=1
```

Nested aggregation with group-by:

```
select CATEGORY, avg
from (select CATEGORY, avg(QUANTITY) avg
    from INCLUDE i natural join PRODUCT p
    group by CATEGORY
    order by avg desc) where ROWNUM=1
```

Delete operation:

```
delete from Customer where EMAIL like 'Jonathan.Ramirez1952@gmail.com'
```

Update operation:

```
update customer set c_name = 'Luis Montgomery', address = '34509 Miller Tunnel
Apt. 711, Ericksonville, NJ 27119-4080', password = '3596218a' where email like
'Luis.Montgomery1990@gmail.com'
```