Advanced Database Management Systems- Project 1

1. You need to design a database for an online shopping system. This system has the following requirements:

   - A user has id, username, password, full name, address, phone number, email

   - A user can log in to the system (assuming that he/she has already registered).

   - A user can list a product for sale. A product has an id, name, description, price, and quantity. It is assumed that each product is listed by only a user. However, products can have similar names and descriptions. For example, two users can sell the same Apple iPhone 13. However, those products will have different id numbers and potentially different prices or quantities.

   - A user can buy a listed product. After the order is made, the product quantity will be deducted accordingly. Note: a user can sell multiple products at the same time. And a user can buy multiple products (from different sellers) in the same order.

Entities:

a) Users.
b) Products.
c) Order.
d) Order Items.

Attributes:

Users:

a. UserID.
b. UserName
c. Password.
d. FullName.
e. Address.
f. PhoneNumber.
g. Email.

Products:

I. ProductID.
II. UserID.
III. Name.
IV. Description.
V. Quantity.
VI. Price.

Orders:

I. OrderID.

II. UserID.
III. Date.
IV. TotalCost.

Order Items:

I. UserID.
II. OrderID.
III. ProductID.
IV. Quantity.
V. Cost.

Relationships:

a) User (1)- (N) Product: One user can have multiple products for sale.
b) User (1)- (N) Order: One user can place multiple orders.
c) Product (1)- (N) Order Items: One product can be included in multiple order items.
d) Order (1)- (N) Order Items: One order can have multiple order items.

Domains for Attributes:

Users:

UserID - INTEGER
UserName - VarChar(30)
Password - VarChar(30)
FullName - VarChar(30)
Address - VarChar(100)
PhoneNumber - Numeric
Email - VarChar(100)

Products:

ProductID - INTEGER
UserID - INTEGER
Name - VarChar(30)
Description - VarChar(100)
Quantity- FLOAT
Price- FLOAT

Orders:

OrderID - INTEGER

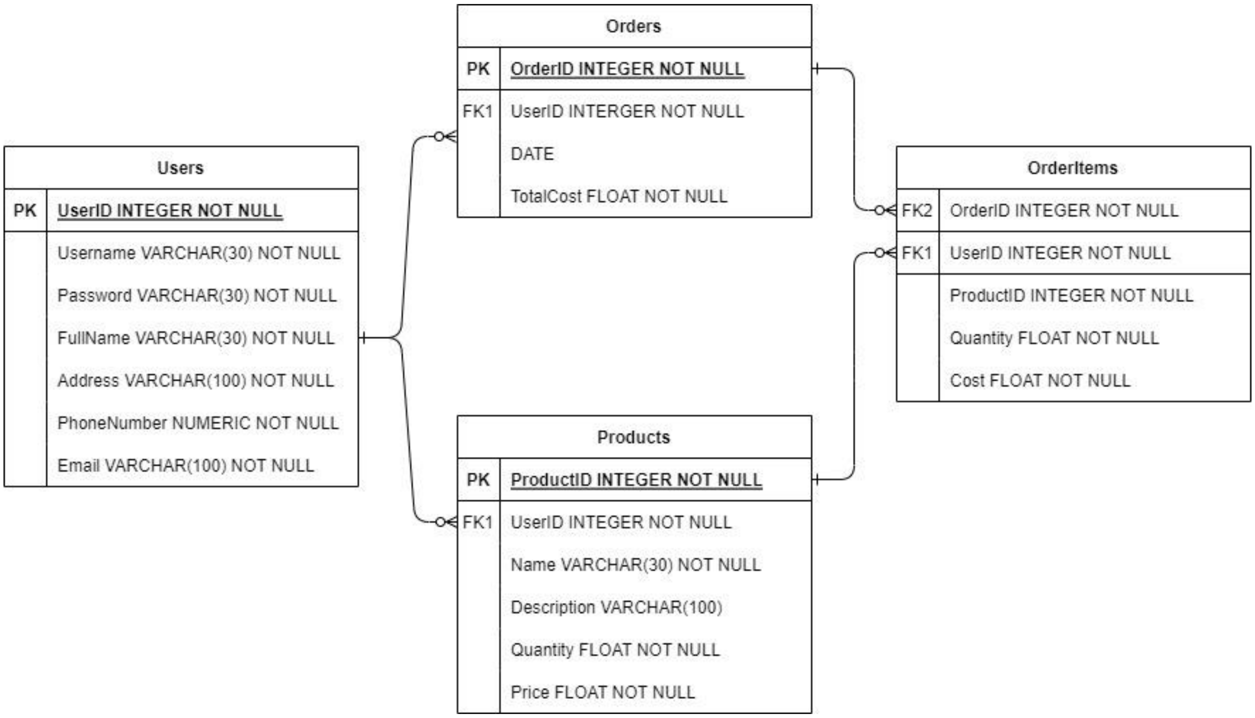
UserID - INTEGER
Date
TotalCost - FLOAT

Order Items:

UserID - INTEGER
OrderID - INTEGER
ProductID - INTEGER

Quantity - FLOAT
Cost - FLOAT

Entity-Relationship Diagram:


Orders
PK OrderID INTEGER NOT NULL
FK1 UserID INTERGER NOT NULL
DATE
TotalCost FLOAT NOT NULL
Users
PK UserID INTEGER NOT NULL
Username VARCHAR(30) NOT NULL
Password VARCHAR(30) NOT NULL
FullName VARCHAR(30) NOT NULL
Address VARCHAR(100) NOT NULL
PhoneNumber NUMERIC NOT NULL
Email VARCHAR(100) NOT NULL
OrderItems
FK2 OrderID INTEGER NOT NULL
FK1 UserID INTEGER NOT NULL
ProductID INTEGER NOT NULL
Quantity FLOAT NOT NULL
Cost FLOAT NOT NULL
Products
PK ProductID INTEGER NOT NULL
FK1 UserID INTEGER NOT NULL
Name VARCHAR(30) NOT NULL
Description VARCHAR(100)
Quantity FLOAT NOT NULL
Price FLOAT NOT NULL

SQL Codes for the Database:

Users:

```
 CREATE TABLE "Users" (
	"UserID"	INTEGER NOT NULL,
	"UserName"	VARCHAR(30) NOT NULL,
	"Password"	VARCHAR(30) NOT NULL,
	"FullName"	VARCHAR(30) NOT NULL,
	"Address"	VARCHAR(100) NOT NULL,
	"PhoneNumber"		NUMERIC NOT NULL,
	"Email"VARCHAR(100),
	PRIMARY KEY("UserID")
);
```

Products:

```
CREATE TABLE "Products" (
        "ProductID"     INTEGER NOT NULL,
        "UserID"        INTEGER NOT NULL,
        "Name"VARCHAR(30) NOT NULL,
        "Description"   VARCHAR(100),
        "Price" INTEGER NOT NULL,
        "Quantity"      FLOAT NOT NULL,
        PRIMARY KEY("ProductID")
);
```

Orders:

```
CREATE TABLE "Orders" (
        "OrderID"       INTEGER NOT NULL,
        "UserID"        INTEGER NOT NULL,
        "Date"  VARCHAR NOT NULL,
        "TotalCost"     FLOAT NOT NULL,
        PRIMARY KEY("OrderID")
);
```

OrderItems:

```
CREATE TABLE "OrderItems" (
        "OrderID"       INTEGER NOT NULL,
        "UserID"        INTEGER NOT NULL,
        "ProductID"     INTEGER NOT NULL,
        "Quantity"      FLOAT NOT NULL,
        "Cost"  FLOAT NOT NULL,
        FOREIGN KEY("OrderID") REFERENCES "Orders",
        FOREIGN KEY("UserID") REFERENCES "Users"
);
```