**Project Title : E-Commerce**

**Step** 1**. Entity-Relationship (ER) Diagram**:

Our E-Commerce ER diagram provided here is , it typically includes entities (tables), relationships between entities, and attributes associated with each entity.

**2. Creating Table Database Schema:**

.1. User Table:

Attributes:

* `UserID (Primary Key, INT)`
* `Username (VARCHAR(255))`
* `Email (VARCHAR(255))`
* `Password (VARCHAR(255))`

1. UserUpload Table:

Attributes:

* `UserID` (Primary Key, INT)
* `Image` (VARCHAR(255))

**Explanation:**

* This table is used to store user upload information, specifically the UserID and the path to the uploaded image.
* UserID is the primary key that uniquely identifies each user.

2. CustomerProducts Table:

Attributes:

* `ProductID` (Primary Key, INT)
* `Title` (VARCHAR(255))
* `Image` (VARCHAR(255))
* `Cost` (DECIMAL(10,2))
* `UserID` (INT)
* `CustomerID` (INT)

**Explanation:**

* This table represents the products listed by customers.
* ProductID is the primary key that uniquely identifies each product.Title, Image, and Cost columns store information about the product.
* UserID column references the UserUpload table, connecting the product to the user who uploaded it.
* CustomerID column references the Login table, linking the product to the customer who listed it.

3. Login Table:

Attributes:

* `UserID` (Primary Key, INT)
* `Username` (VARCHAR(255))
* `Email` (VARCHAR(255))
* `Contact` (VARCHAR(255))
* `Role` (VARCHAR(255))
* `Password` (VARCHAR(255))

**Explanation:**

* This table stores user login information.
* UserID is the primary key that uniquely identifies each user.
* Username, Email, Contact, Role, and Password columns store the user's login details.

4. OrderTable Table:

Attributes:

* `OrderID` (Primary Key, INT)
* `Name` (VARCHAR(255))
* `Address` (VARCHAR(255))
* `City` (VARCHAR(255))
* `State` (VARCHAR(255))
* `ZipCode` (VARCHAR(255))
* `Contact` (VARCHAR(255)

**Explanation:**

* This table stores order details.
* OrderID is the primary key that uniquely identifies each order.
* Name, Address, City, State, ZipCode, and Contact columns store information about the order.
* CustomerID column references the Login table, associating the order with the customer who placed it.

5. Category Table:

Attributes:

* `CatID` (Primary Key, INT)
* `CatName` (VARCHAR(255))
* `CatDescription` (VARCHAR(255))

**Explanation:**

* This table stores information about product categories.
* CatID is the primary key that uniquely identifies each category
* CatName and CatDescription columns store the name and description of the category.

6. Product Table:

Attributes:

* + `ProductID` (Primary Key, INT)
  + `Price` (DECIMAL(10,2))
  + `Color` (VARCHAR(255))
  + `Brand` (VARCHAR(255))
  + `CatID` (INT)
  + `ProductName` (VARCHAR(255))
  + `ProductDesc` (VARCHAR(255))
  + `Image1` (VARCHAR(255))
  + `Image2` (VARCHAR(255))
  + `Image3` (VARCHAR(255))

**Explanation:**

* This table represents individual products.
* ProductID is the primary key that uniquely identifies each product.
* Price, Color, Brand, CatID, ProductName, ProductDesc, Image1, Image2, and Image3 columns store information about the product.
* CatID column references the Category table, associating the product with its corresponding category.

7. Cart Table:

Attributes:

* `OrderID` (Primary Key, INT)
* `CustomerID` (INT)
* `Size` (VARCHAR(255))
* `Quantity` (INT)
* `ProductID` (INT)

- **Explanation:**

* This table stores the items added to the cart.
* OrderID is the primary key that uniquely identifies each item in the cart.
* CustomerID column references the Login table, associating the cart item with the customer who added it.
* Size and Quantity columns store information about the selected size and quantity of the product.
* ProductID column references the Product table, linking the item to the corresponding product.

Step 3. Ensuring Proper Data Types and Constraints:

We utilize appropriate data types: **VARCHAR** for textual data, **INT** for integers, **DECIMAL** for monetary values, **DATE** for dates, etc.

We apply constraints like **PRIMARY KEY**, **FOREIGN KEY, NOT NULL, UNIQUE**, etc., based on the nature of the data and relationships to ensure data integrity.

Step 4. Naming Conventions:

- We adopt a consistent naming convention throughout the schema, such as **camelCase** or **snake\_case.**

- We choose meaningful names for tables and attributes to enhance readability. Apologies for the confusion. Here is the revised explanation of the decision-making process for each table and attribute design choice based on the revised table structure:

Step 5. Namalization:

**Normalization** is a process used in database design to eliminate data redundancy and ensure data integrity. It involves organizing data into tables and defining relationships between tables based on functional dependencies. In the provided database schema, normalization can be improved in the following ways:

**Splitting UserUpload table:** Since UserUpload table only contains UserID and Image columns, it can be merged with the Login table. The Image column can be added to the Login table.

**Splitting CustomerProducts table**: The CustomerProducts table can be split into two separate tables: Customer and ProductListing. The Customer table will contain UserID, Username, Email, Contact, Role, and Password columns. The ProductListing table will include ProductID, Title, Image, Cost, and CustomerID columns. The CustomerID column in the ProductListing table will reference the Customer table

Step 6. explanation of your decision-making process for each table and attribute design choice.

**1. User Table:**

* The User table is designed to store information about users who interact with the e-commerce platform.
* The attributes chosen for this table include UserID, Username, Email, Contact, Role, and Password.
* UserID serves as the primary key to uniquely identify each user.
* Username, Email, and Contact are attributes commonly associated with user accounts in an e-commerce system, providing essential information for user identification and communication.
* Role represents the role of the user within the system, such as a customer or administrator.
* Password is included to securely store user passwords.

2**. Product Table:**

* The Product table is responsible for storing information about products available for purchase on the e-commerce platform.
* The attributes chosen for this table include ProductID, Title, Image, Cost, UserID, and CustomerID.
* ProductID serves as the primary key to uniquely identify each product.
* Title represents the name or title of the product.
* Image stores the image file name associated with the product.
* Cost indicates the price of the product.
* UserID and CustomerID are included to establish relationships with the UserUpload and CustomerProducts tables, respectively.

**3. UserUpload Table:**

* The UserUpload table is designed to store information about user-uploaded images.
* The attributes chosen for this table include UserID and Image.
* UserID serves as the primary key to uniquely identify each user.
* Image stores the image file name uploaded by the user.

4. **CustomerProducts Table:**

* The CustomerProducts table is responsible for storing information about products purchased by customers.
* The attributes chosen for this table include ProductID, Title, Image, Cost, UserID, and CustomerID.
* ProductID serves as the primary key to uniquely identify each product.
* Title represents the name or title of the product.
* Image stores the image file name associated with the product.
* Cost indicates the price of the product.
* UserID and CustomerID are included to establish relationships with the User and Product tables, respectively.

5**. Login Table:**

* The Login table is designed to store user login credentials.
* The attributes chosen for this table include UserID, Username, Email, Contact, Role, and Password.
* UserID serves as the primary key to uniquely identify each user.
* Username, Email, and Contact are attributes commonly associated with user accounts in an e-commerce system, providing essential information for user identification and communication.
* Role represents the role of the user within the system, such as a customer or administrator.
* Password is included to securely store user passwords.

The overall design choices and attribute selection aim to capture the necessary information for an e-commerce platform, including user details, product information, and the relationships between these entities. It is important to note that the design can be further refined based on specific project requirements and considerations.

Step 7. Document Assumptions:

**We Assummed that :**

**1. User Roles**: The "Role" attribute in the User and Login tables indicates the role of the user within the system. It is assumed that there are predefined roles such as "customer" or "administrator." Additional roles can be added as per the specific requirements of the e-commerce platform.

**2. Image Storage**: The "Image" attribute in the UserUpload, CustomerProducts, and Product tables stores the file name of the associated image. It is assumed that the actual image files will be stored in a separate storage system or a designated folder on the server, and the file paths will be used to retrieve and display the images.

**3. Data Validation**: It is assumed that appropriate data validation and error handling mechanisms will be implemented at the application layer to ensure the integrity and consistency of the data being stored in the tables.

**5. Foreign Key Constraints**: It is assumed that proper referential integrity will be maintained through foreign key constraints between related tables. This ensures that only valid and existing UserID, ProductID, and OrderID values can be inserted into the respective tables.

**6. User-Order Relationship**: The UserOrder table establishes a one-to-many relationship between users and orders. It is assumed that each order will be associated with a single user, and multiple orders can be placed by the same user.

Step 8. Test the Database Schema:

* Insert sample data into each table to verify that data can be successfully added.
* Run queries to check relationships between tables, ensuring that foreign keys are properly referencing primary keys.
* Confirm that the database behaves as expected in terms of data consistency and integrity.