

# **Database Structure of Online Shopping Website**

Database Management System

Final Project

By

**Soodabeh Ramezani**

Instructor

**Dr. Ling Xu**

# Part1

## Abstract

The proposed ER diagram is used as a tool to facilitate a web-based attire shopping. The structure provides the sellers with opportunity to register themselves to the website and sell their products online. The sellers can advertise their products and include discounts and promotions to their products. The database provides the sellers with opportunity to target a specific group of buyers and send them relevant advertisements. The buyers can also register themselves and search for a product and buy it through the online platform. Finally, there is an option for the buyers to return their purchased items if they are not satisfied.

This database can manage up to 10000 sellers, and each seller is allowed to register up to 100 items. Returning of unlimited purchased items is also acceptable.

## Mission statement

The purpose of the online shopping database system is to maintain the sellers/buyers data to be used in online shopping and facilitate business for customers and sellers.

## Mission objectives

To maintain (enter, update, and delete) data on Sellers.

To maintain (enter, update, and delete) data on Customers.

To maintain (enter, update, and delete) data on Items.

To maintain (enter, update, and delete) data on Shopping\_cart.

To maintain (enter, update, and delete) data on Purchases.

To maintain (enter, update, and delete) data on Discounts.

To maintain (enter, update, and delete) data on Item\_Review.

To maintain (enter, update, and delete) data on Card\_Info.

To maintain (enter, update, and delete) data on Websie\_Owner.

To maintain (enter, update, and delete) data on Return\_Items.

To perform searches on Sellers.

To perform searches on Items.

To perform searches on Customers.

To perform searches on Shopping\_Cart.

To perform searches on Purchases.

To perform searches on Discounts.

To perform searches on Card\_Info.

To perform searches on Return\_Items.

To track the status of Purchase.

To track the status of Return\_Items.

To report on Sellers.

To report on Customers.

To report on Items.

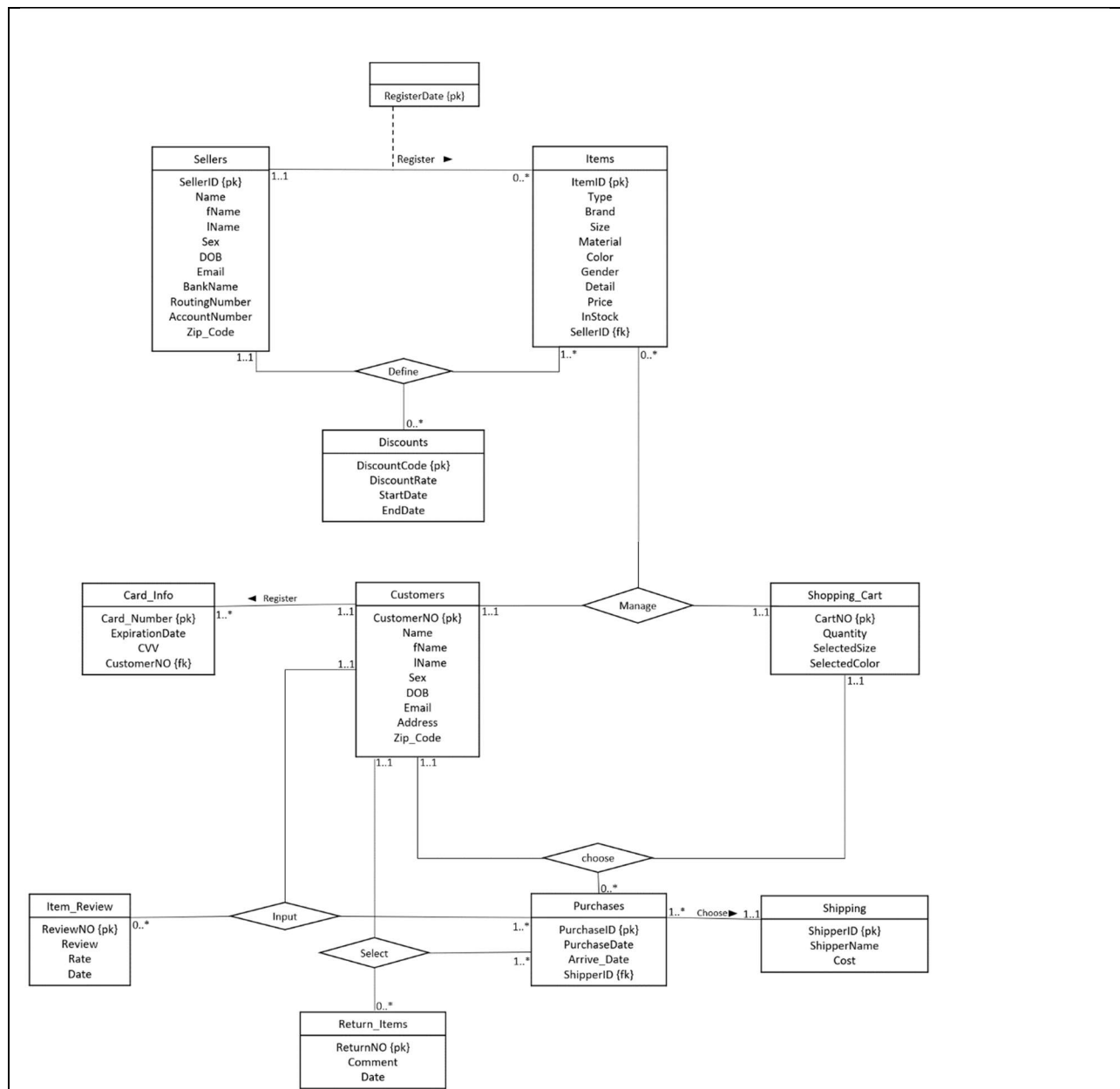
To report on Purchases.

To report on Discounts.

To report on Item\_Review.

To report on Return\_Items.

## An Entity\_Relationship (ER) Diagram of Online Shopping Website



## Relational Model

**Sellers** (SellerID, fName, IName, Sex, DOB, Email, BankName, RoutingNO, AccountNO, Zip\_Code)

**Primary Key** SellerID

**Items** (ItemID, Type, Brand, Size, Material, Color, Gender, Detail, Price, SellerID, RegisterDate)

**Primary Key** ItemID

**Foreign Key** SellerID references Sellers (SellerID)

**ItemColor** (Color, ItemID)

**Primary Key** Color

**Foreign Key** ItemID references Item (ItemID)

**ItemSize** (Size, ItemID)

**Primary Key** Size

**Foreign Key** ItemID references Item (ItemID)

**Discounts** (DiscountCode, DiscountRate, StartDate, EndDate)

**Primary Key** DiscountCode

**Define** (DiscountCode, SellerID, ItemID)

**Primary Key** DiscountCode, SellerID, ItemID

**Foreign Key** DiscountCode references Discount (Discount\_code)

**Foreign Key** SellerID references Seller (SellerID)

**Foreign Key** ItemID references Item (ItemID)

**Customers** (CustomerNO, fName, IName, Sex, DOB, Email, Address, Zip\_Code)

**Primary Key** CustomerNO

**Card\_Info** (Card\_Number, ExpirationDate, CVV, CustomerNO)

**Primary Key** Card\_Number

**Foreign Key** CustomerNO references Customer (CustomerNO)

**Shopping\_Cart** (CartNO, Quantity, SelectedSize, SelectedColor)

**Primary Key** CartNO

**Manage** (ItemID, CustomerNO, CartNO)

**Primary Key** ItemID, CustomerNO, CartNO

**Foreign Key** ItemID references Item (ItemID)

**Foreign Key** CustomerNO references Customer (CustomerNO)

**Foreign Key** CartNO references Shopping\_Cart (CartNO)

**Purchases** (PurchaseID, PurchaseDate, Arrive\_Date, ShipperID)

**Primary Key** PurchaseID

**Foreign Key** ShipperID references Shipping (ShipperID)

**Shipping** (ShipperID, ShipperName, Cost)

**Primary Key** ShipperID

**Choose** (PurchaseID, CustomerNO, CartNO)

**Primary Key** PurchaseID, CustomerNO, CartNO

**Foreign Key** PurchaseID references Purchases (PurchaseID)

**Foreign Key** CustomerNO references Customer (CustomerNO)

**Foreign Key** CartNO references Shopping\_Cart (CartNO)

**Item\_Review** (ReviewNO, Review, Rate, Date)

**Primary Key** ReviewNO

**Input** (PurchaseID, CustomerNO, ReviewNO)

**Primary Key** PurchaseID, CustomerNO, ReviewNO

**Foreign Key** PurchaseID references Purchases (PurchaseID)

**Foreign Key** CustomerNO references Customer (CustomerNO)

**Foreign Key** ReviewNO references Item\_Review (ReviewNO)

**Return\_Items** (ReturnNO, Date)

**Primary Key** Returned\_NO

**Select** (ReturnNO, PurchaseID, CustomerNO)

**Primary Key** ReturnNO, PurchaseID, CustomerNO

**Foreign Key** PurchaseID references Purchases (PurchaseID)

**Foreign Key** Customer\_NO references Customer (Customer\_NO)

## list of all actors and use cases

### 1. Use Case name: Register a new seller

**Actor:** Seller

**Steps:**

1. Actor clicks on "New Seller" button;
2. A new seller ID is shown;
3. Prompt to enter Name, Sex, DOB, Email, BankName, RoutingNO, AccountNO, Zip\_Code;
4. All information is displayed; ask for confirmation;
5. Actor clicks on "Confirm" button.

### 2. Use Case name: Register a new customer

**Actor:** Customer

**Steps:**

1. Actor clicks on "New Customer" button;
2. A new customer number is shown;
3. Prompt to enter Name, Sex, DOB, Email, Address, Zip\_Code;
4. All information is displayed; ask for confirmation;
5. Actor clicks on "Confirm" button.

### 3. Use Case name: Register a new item

**Actor:** Seller

**Steps:**

1. Actor clicks on "New Item for Sell" button;
2. A new item ID and register date are shown;
3. Prompt to enter ItemID, Type, Brand, Size, Material, Color, Gender, Detail, Price, SellerID;
4. All information is displayed; ask for confirmation;
5. Actor clicks on "Confirm" button.

#### **4.Use Case name:** Define a new discount

**Actor:** Seller

##### **Steps:**

1. Actor clicks on “Enter Discount” button;
2. A new discount code is shown;
3. Actor enter item ID; item information is shown;
4. Prompt to enter DiscountRate, StartDate, EndDate;
5. All information is displayed; ask for confirmation;
6. Actor clicks on “Confirm” button.

#### **5.Use Case name:** Adding a new item to shopping cart

**Actor:** Customer

##### **Steps:**

1. Actor selects an existing item;
2. Actor selects size and color;
3. Actor enters quantity;
4. Actor clicks on “Add to Cart” button;
5. A new cart number with all the item information are added in shopping cart.

#### **6.Use Case name:** Purchase an item

**Actor:** Customer

##### **Steps:**

1. Actor selects an item in shopping cart and click “Checkout” button;
2. An existing customer’s address is shown; ask for review or edit;
3. Actor click on “Confirm” button.
4. Different shipping options are displayed;
5. Actor select a desirable shipping method;
6. Actor clicks on “Proceed to Checkout” button;
7. An existing customer’s bank information is shown, ask for review or edit;
8. Actor click on “Confirm” button;
9. All information is displayed; ask for review or edit order;



10. Actor clicks on “Place Your Order” button;
11. Order details including date, order number, and total price are displayed.

**7.Use Case name:** Adding a review on purchased item

**Actor:** Customer

**Steps:**

1. Actor selects an item in purchased list and click on “Add Review” button;
2. A new review number and date are shown;
3. Prompt to enter Review, Rate;
4. All information is displayed; ask for confirmation;
5. Actor clicks on “Submit” button.

**8.Use Case name:** Return a purchased item

**Actor:** Customer

**Steps:**

1. Actor selects an item in purchased list and click on “Return” button;
2. A new return number is shown;
3. Prompt to enter Comment, Date;
4. All information is displayed; ask for confirmation;
5. Actor clicks on “Confirm” button.

**9.Use Case name:** Adding a new bank info

**Actor:** Customer

**Steps:**

1. Actor clicks on “Add a bank info” button;
2. Prompt to enter Card\_Number, ExpirationDate, CVV;
3. All information is displayed; ask for confirmation;
4. Actor clicks on “Confirm” button.

**10.Use Case name:** Delete a customer

**Actor:** Customer

**Steps:**

1. Actor clicks on "Unsubscribe"
2. Confirm that all associated information to be deleted as well.
3. Final reviewing to the action is shown;
4. Actor "Confirm" the unsubscribe action.

**11.Use Case name:** Delete a Seller

**Actor:** Seller

**Steps:**

1. Actor clicks on "Unsubscribe"
2. Confirm that all associated information is to be deleted as well.
3. Final reviewing to the action is shown;
4. Actor "Confirm" the unsubscribe action.

**12.Use Case name:** Delete an item

**Actor:** Seller

**Steps:**

1. Actor selects an existing item;
2. Clicks on "Delete" button
3. Confirm that all associated information is to be deleted as well.
4. Actor "Confirm" the delete action.

**13.Use Case name:** Delete a discount

**Actor:** Seller

**Steps:**

1. Actor selects an existing discount;

2. Clicks on “Delete” button
3. Final reviewing to the action is shown;
4. Actor “Confirm” the delete action.

**14.Use Case name:** Remove an item from shopping cart

**Actor:** Customer

**Steps:**

1. Actor selects an existing item in shopping cart;
2. Actor clicks on “Remove”
3. Final reviewing to the action is shown;
4. Actor “Confirm” the remove action.

**15.Use Case name:** Remove a review

**Actor:** Customer

**Steps:**

1. Actor selects an existing review;
2. Actor clicks on “Remove”
3. Final reviewing to the action is shown;
4. Actor “Confirm” the remove action.

**16.Use Case name:** Delete a bank info

**Actor:** Customer

**Steps:**

1. Actor selects an existing bank info;
2. Actor clicks on “Remove”
3. Final reviewing to the action is shown;
4. Actor “Confirm” the remove action.

**17.Use Case name:** Update seller

**Actor:** Seller

**Steps:**

1. Actor clicks on “Edit Seller” button;
2. Prompt to edit Name, Sex, DOB, Email, BankName, RoutingNO, AccountNO, Zip\_Code;
3. Final reviewing to the action is shown;
4. Actor “Confirm” the edit action.

**18.Use Case name:** Update customer

**Actor:** Customer

**Steps:**

1. Actor clicks on “Edit Customer” button;
2. Prompt to edit Name, Sex, DOB, Email, Address, Zip\_Code;
3. Final reviewing to the action is shown;
4. Actor “Confirm” the edit action.

**19.Use Case name:** Update an item

**Actor:** Seller

**Steps:**

1. Actor selects an existing item
2. Actor clicks on “Edit” button;
6. Prompt to edit ItemID, Type, Brand, Size, Material, Color, Gender, Detail, Price, SellerID, RegisterDate;
3. Final reviewing to the action is shown;
4. Actor “Confirm” the edit action.

**20.Use Case name:** Update a discount

**Actor:** Seller

**Steps:**

1. Actor selects an existing discount
2. Actor clicks on “Edit” button;
3. Prompt to edit DiscountRate, StartDate, EndDate;
4. Final reviewing to the action is shown;
5. Actor “Confirm” the edit action.

## **21.Use Case name:** Update shopping cart

**Actor:** Customer

### **Steps:**

1. Actor selects an existing item in shopping cart
2. Actor clicks on “Edit” button;
3. Prompt to edit Quantity, SelectedSize, SelectedColor;
4. Final reviewing to the action is shown;
5. Actor “Confirm” the edit action.

## **22.Use Case name:** Update review

**Actor:** Customer

### **Steps:**

1. Actor selects a review
2. Actor clicks on “Edit” button;
5. Prompt to edit Review, Rate;
3. Final reviewing to the action is shown;
4. Actor “Confirm” the edit action.

## **23.Use Case name:** Update bank info

**Actor:** Customer

### **Steps:**

5. Actor selects an existing bank info
6. Actor clicks on “Edit” button;
6. Prompt to edit Card\_Number, ExpirationDate, CVV;
7. Final reviewing to the action is shown;

8. Actor "Confirm" the edit action.

**24.Use Case name:** Count distinct items posted by a seller

**Actor:** Seller

**Steps:**

1. Actor clicks on "Distinct Items Posted" button;
2. Information is shown, indicating how many distinct items posted by the seller.

**25.Use Case name:** Sum of discount added by a seller

**Actor:** Seller

**Steps:**

1. Actor clicks on "Total discounts" button;
2. Information is shown, indicating how much discount posted by the seller.

**26.Use Case name:** Number and list of items in shopping cart

**Actor:** Customer

**Steps:**

1. Actor clicks on "Shopping Cart" button;
2. Number of items with list of items are shown.

**27.Use Case name:** Total price of items in shopping cart

**Actor:** Customer

**Steps:**

1. Actor clicks on "Total price" button;
2. Total price is shown.

**28.Use Case name:** List number of orders for each item posted by a seller

**Actor:** Seller

**Steps:**

1. Actor clicks on “Order list” button;
2. A table with ID and name of each item and number of their order is shown.

**29.Use Case name:** List of all discount

**Actor:** Seller

**Steps:**

1. Actor clicks on “Discount list” button;
2. List of IDs and names and discount for all items which contain discount is shown.

**30.Use Case name:** List items by type

**Actor:** Customer

**Steps:**

1. Actor inserts a type;
2. Actor clicks on “Search” button;
3. All attributes for selected items are shown.

## Use Case Realization

### Entity: Sellers

- `INSERT INTO sellers VALUES (SellerID, fName, lName, Sex, DOB, Email, BankName, RoutingNumber, AccountNumber, Zip_Code);`
- `DELETE FROM sellers WHERE SellerID = dataValue;`
- `UPDATE sellers SET BankName=dataValue1, RoutingNumber= dataValue2, AccountNumber= dataValue3, Zip_Code= dataValue4 WHERE SellerID = dataValue;`

### Entity: Customers

- `INSERT INTO customers VALUES (CustomerNO, fName, lName, Sex, DOB, Email, Address, Zip_Code);`
- `DELETE FROM Customers WHERE CustomerNO = dataValue;`
- `UPDATE customers SET Email=dataValue1, Address= dataValue2, Zip_Code= dataValue3 WHERE CustomerNO = dataValue;`

### Entity: Items

- `INSERT INTO items VALUES (ItemID, Type, Brand, Size, Material, Color, Gender, Detail, Price, SellerID, RegisterDate);`
- `DELETE FROM items WHERE ItemID = dataValue;`
- `UPDATE items SET Price=dataValue1 WHERE ItemID = dataValue;`
- `SELECT COUNT (ItemID) AS NumberOfItems FROM items WHERE SellerID = dataValue;`
- `SELECT * FROM items WHERE Type= dataValue;`

### Entity: Discounts

- `INSERT INTO discounts VALUES (DiscountCode, DiscountRate, StartDate, EndDate);`
- `DELETE FROM discounts WHERE DiscountCode = dataValue;`
- `UPDATE discounts SET DiscountRate =dataValue1 WHERE DiscountCode = dataValue;`

### Entity: Shopping\_Cart

- `INSERT INTO shopping_Cart VALUES (CartNO, Quantity, SelectedSize, SelectedColor);`
- `DELETE FROM shopping_Cart WHERE CartNO = dataValue;`



- `UPDATE shopping_Cart SET Quantity=dataValue1, SelectedSize =dataValue2, SelectedColor =dataValue3 WHERE CartNO = dataValue;`

#### Entity: Purchases

- `INSERT INTO purchases VALUES (PurchaseID, PurchaseDate, Arrive_Date, ShipperID);`

#### Entity: Item\_Review

- `INSERT INTO item_Review VALUES (ReviewNO, Review, Rate, Date);`

#### Entity: Return\_Items

- `INSERT INTO return_Items VALUES (ReturnNO, Date);`

#### Entity: Card\_Info

- `INSERT INTO card_Info VALUES (Card_Number, ExpirationDate, CVV, CustomerNO);`
- `DELETE FROM card_Info WHERE Card_Number = dataValue;`
- `UPDATE card_Info SET Card_Number =dataValue1, ExpirationDate =dataValue2, CVV =dataValue3 WHERE CustomerNO = dataValue;`

## Test plan and record

I have successfully tested 30 use cases; here I am showing a few examples,

```
2 • INSERT INTO sellers
3 VALUES ('S2', 'Amir', 'Frooqnia', 'M', '1980-02-01', 'a.frooqnia@gmail.com', 'Chase', '2222222', '0011155', '77079');
```

[illegible]

```
4 • DELETE FROM sellers WHERE SellerID = 'S2' ;
```

[illegible]

```
7 • UPDATE sellers
8 SET BankName='America', RoutingNumber= 678999980, AccountNumber= 654789322, Zip Code= '77090' WHERE SellerID = 'S2';
```

[illegible]

```
2 • INSERT INTO online_shopping.items
3 VALUES ('I3','Pants', 'SUB70', 'Large', '100% Polyester', 'Black','M', 'Wicks and dries fast ', '40','8','S3', default);
```

[illegible]

4 • `DELETE FROM online_shopping.items WHERE ItemID = 'I3';`

Result Grid

| ItemID | Type    | Brand    | Size  | Material       | Color  | Gender | Detail              | Price | InStock | Sellers_SellerID | RegisterDate        |
|--------|---------|----------|-------|----------------|--------|--------|---------------------|-------|---------|------------------|---------------------|
| I1     | Shirt   | Goodthrs | 1     | plaid          | Navy   | M      | Herringbone         | 30    | 5       | S1               | 2019-05-05 16:23:50 |
| I2     | T-Shirt | SUB70    | Large | 100% Polyester | Yellow | Unisex | Wicks and drys fast | 40    | 8       | S2               | 2019-05-05 16:11:30 |
|        |         |          |       |                |        |        |                     |       |         |                  |                     |

12 • `UPDATE items SET Price='70' WHERE ItemID = 'I3';`

13

Result Grid

| ItemID | Type    | Brand    | Size  | Material       | Color  | Gender | Detail              | Price | InStock | Sellers_SellerID | RegisterDate        |
|--------|---------|----------|-------|----------------|--------|--------|---------------------|-------|---------|------------------|---------------------|
| I1     | Shirt   | Goodthrs | 1     | plaid          | Navy   | M      | Herringbone         | 30    | 5       | S1               | 2019-05-05 16:23:50 |
| I2     | T-Shirt | SUB70    | Large | 100% Polyester | Yellow | Unisex | Wicks and drys fast | 40    | 8       | S2               | 2019-05-05 16:11:30 |
| I3     | Pants   | SUB70    | Large | 100% Polyester | Black  | M      | Wicks and drys fast | 70    | 8       | S3               | 2019-05-05 16:54:09 |
|        |         |          |       |                |        |        |                     |       |         |                  |                     |

9 • `SELECT COUNT(ItemID) AS NumberOfItems FROM online_shopping.items WHERE Sellers_SellerID = 'S3';`

Result Grid

| NumberOfItems |
|---------------|
| 1             |

## Conclusion

A thorough investigation was initially carried out to layout the framework required for building the database. The database framework helped to avoid process redundancies and ensured that all the entities and their associations are taken into account. The database was then successfully implemented in the MySQL workbench.

## References

Textbook: "Database Systems—A Practical Approach to Design, Implementation, and Management" by Thomas Connolly and Carolyn Begg, Addison Wesley, Latest Edition.

## Part2-Demo

For each table show all data

### Entity: Items

Show all data

```
1 • SELECT * FROM online_shopping.items;
```

| ItemID | Type    | Brand             | Size   | Material        | Color  | Gender | Detail                               | Price | InStock | Sellers_SellerID | RegisterDate      |
|--------|---------|-------------------|--------|-----------------|--------|--------|--------------------------------------|-------|---------|------------------|-------------------|
| I1     | Shirt   | Goodthrs          | 1      | plaid           | Navy   | M      | Herringbone                          | 30    | 5       | S1               | 2019-05-05 16:23: |
| I2     | T-Shirt | SUB70             | Large  | 100% Polyester  | Yellow | Unisex | Wicks and drys fast                  | 40    | 8       | S2               | 2019-05-05 16:11: |
| I3     | Pants   | SUB70             | Large  | 100% Polyester  | Black  | M      | Wicks and drys fast                  | 40    | 8       | S3               | 2019-05-05 17:46: |
| I4     | Shorts  | Conceited Premium | Small  | PREMIUM STRETCH | Red    | F      | Ultra Soft 40 Trending Prints        | 17    | 10      | S1               | 2019-05-05 17:38: |
| I5     | Dress   | MiaoYi            | Small  | Cotton          | Blue   | F      | Off Shoulder Asymmetrical Hem ...    | 100   | 8       | S2               | 2019-05-05 17:47: |
| I6     | Dress   | Dressystar        | Small  | Lylon           | Blue   | F      | Lace Off Shoulder Hi-Lo Short Sle... | 180   | 2       | S2               | 2019-05-05 17:52: |
| I7     | Dress   | Dymaisei          | 4      | Silk            | Red    | F      | Gold Appliques Mermaid Evening ...   | 1000  | 2       | S2               | 2019-05-05 18:04: |
| I8     | Pants   | Haggar            | XLarge | Polyester       | Black  | M      | Slim Fit Flat Front Casual Pant      | 67    | 20      | S3               | 2019-05-05 18:04: |

Using aggregate function: Total price of items








```
2 • SELECT sum(Price) AS MySun From online_shopping.items;
```

| MySun |
|-------|
| 1407  |

## Entity: Customers

Show all data

```
1 • SELECT * FROM online_shopping.customers;
```

| < | Result Grid |  |  Filter Rows: <input type="text"/> | Edit:  |  |  | Export/Import:  |  | Wrap Cell Content |
|---|-------------|---|---|---|---|---|--|---|-------------------|
|   | CustomerNO  | fName   | lName   | Sex   | DOB   | Email   | Address  | Zip_Code  |                   |
|   | C1          | Najme   | Vaez  | F   | 1989-02-01  | N.vaez@gmail.com  | 1151 GLENWOO...  | 99099   |                   |
|   | C2          | Rick  | Nelson  | M   | 1956-03-12  | r.nelsn@hotmail....   | 1204 Fry Rd  | 76034   |                   |
|   | C3          | Alex  | Besov   | M   | 1992-06-08  | a.besv@gmail.com  | 342 West Montr...  | 73134   |                   |
|   | C4          | Mayam   | Mousavi   | F   | 1982-05-01  | mmsav@yahoo....   | 127 S Gessner Dr   | 43198   |                   |
|   | C5          | Behrooz   | Raessi  | M   | 1981-05-93  | bras@gmail.com  | 435 Katy Dr  | 75395   |                   |
|   | C6          | Martin  | Albertini   | M   | 1963-02-06  | maltin@gmail.com  | 3428 Westheime...  | 78305   |                   |
|   | C7          | Wilson  | Pineda  | M   | 1968-02-09  | wil_pin@yahoo.c...  | 453 N Post Oak ...   | 76296   |                   |

Using aggregate: Number of male customers

```
5 • SELECT COUNT(CustomerNO) FROM customers WHERE Sex='M';
```

|   |                   |              |         |                    |
|---|-------------------|--------------|---------|--------------------|
| < | Result Grid       | Filter Rows: | Export: | Wrap Cell Content: |
|   | COUNT(CustomerNO) |              |         |                    |
| ▶ | 5                 |              |         |                    |

## Entity: Sellers

Show all data

```
1 • SELECT * FROM online_shopping.sellers;
```

|   |             |       |              |      |            |                       |                |               |                    |          |
|---|-------------|-------|--------------|------|------------|-----------------------|----------------|---------------|--------------------|----------|
|   | Result Grid |       | Filter Rows: |      | Edit:      |                       | Export/Import: |               | Wrap Cell Content: |          |
|   | SellerID    | fname | lname        | Sex  | DOB        | Email                 | BankName       | RoutingNumber | AccountNumber      | Zip_Code |
| ▶ | S1          | Emili | Ramez        | F    | 2000-04-29 | emili.ramez@gmail.com | WellsFargo     | 123456        | 65432              | 77077    |
|   | S2          | Amir  | Frooqnia     | M    | 1980-02-01 | a.frooqnia@gmail.com  | America        | 678999900     | 654789322          | 77090    |
|   | S3          | Mahdi | Haghshenas   | M    | 1984-02-01 | mahdi1234@gmail.com   | Chase          | 27865321      | 1999333            | 77079    |
| * | NULL        | NULL  | NULL         | NULL | NULL       | NULL                  | NULL           | NULL          | NULL               | NULL     |

Using aggregate function: Number of sellers

```
2 • SELECT count(SellerID) FROM online_shopping.sellers;
```

|   |                 |  |              |  |         |  |                    |  |
|---|-----------------|--|--------------|--|---------|--|--------------------|--|
|   | Result Grid     |  | Filter Rows: |  | Export: |  | Wrap Cell Content: |  |
|   | count(SellerID) |  |              |  |         |  |                    |  |
| ▶ | 3               |  |              |  |         |  |                    |  |

## Entity: Discounts

Show all data

```
1 • SELECT * FROM online_shopping.discounts;
```

| DiscountCode | DiscountRate | StartDate  | EndDate    |
|--------------|--------------|------------|------------|
| D1           | 30           | 2019-01-01 | 2019-05-01 |
| D2           | 20           | 2019-01-01 | 2019-05-01 |
| D3           | 10           | 2019-01-01 | 2019-05-01 |

Using aggregate function: Maximum discount rate

```
2 • SELECT max(DiscountRate) From online_shopping.discounts;
```

| max(DiscountRate) |
|-------------------|
| 30.00             |



## Entity: Shopping\_Cart

Show all data

```
1 • SELECT * FROM online_shopping.shopping_cart;
```

| CartNO | Quantity | SelectedSize | SelectedColor |
|--------|----------|--------------|---------------|
| 1      | 2        | Small        | Blue          |
| 2      | 1        | 4            | Red           |
| 3      | 1        | XLarge       | Black         |
| 4      | 7        | Lage         | Black         |
| 5      | 1        | 1            | Navy          |
| 6      | 2        | Large        | Black         |
| 7      | 3        | Small        | Blue          |

Using aggregate function: Minimum quantity in shopping\_Cart list

```
2 • SELECT MIN(Quantity) FROM online_shopping.shopping_cart;
```






| MIN(Quantity) |
|---------------|
| 1             |



## Entity: Purchases





Show all data

```
1 • SELECT * FROM online_shopping.purchases;
```

| <b>Result Grid</b>   Filter Rows: <input type="text"/> Edit:    Export/Imp |            |              |             |           |
|---|------------|--------------|-------------|-----------|
|   | PurchaseID | PurchaseDate | Arrive_Date | ShipperID |
|   | P1         | 2019-01-05   | 2019-01-07  | SH1       |
|   | P2         | 2019-02-08   | 2019-02-09  | SH2       |
| ▶   | P3         | 2019-04-02   | 2019-04-09  | SH3       |

Using aggregate function: Number of purchases

```
2 • SELECT COUNT(PurchaseID) from online_shopping.purchases;
```

| <b>Result Grid</b>   Filter Rows: <input type="text"/> Export:  Wrap Cell Content:  |                   |
|---|-------------------|
|   | COUNT(PurchaseID) |
| ▶   | 3                 |

## Entity: Items, Sellers

Relationships: List name of the sellers and their items

```
4 • SELECT S.fname, S.lname , I.ItemID
5 FROM online_shopping.sellers S, online_shopping.items I
6 Where I.Sellers_SellerID=S.SellerID;
7
```

|   | fname | lname      | ItemID |
|---|-------|------------|--------|
| ▶ | Emili | Ramez      | I1     |
|   | Emili | Ramez      | I4     |
|   | Amir  | Frooqnia   | I2     |
|   | Amir  | Frooqnia   | I5     |
|   | Amir  | Frooqnia   | I6     |
|   | Amir  | Frooqnia   | I7     |
|   | Mahdi | Haghshenas | I3     |
|   | Mahdi | Haghshenas | I8     |

## Entity: Items, Customers, Shopping Cart

```
1 • SELECT * FROM online_shopping.manage;
2
```

|   | CartNO | CustomerNO | ItemID |
|---|--------|------------|--------|
| ▶ | 5      | C5         | I1     |
|   | 4      | C4         | I3     |
|   | 6      | C6         | I3     |
|   | 1      | C1         | I5     |
|   | 2      | C2         | I7     |
|   | 3      | C3         | I8     |
| • | NULL   | NULL       | NULL   |

Relationships: List of items which are in customer C5 shopping cart

```
2 • SELECT m.CustomerNO,C.fName,C.lName, I.ItemID
3 FROM online_shopping.customers C, online_shopping.items I,online_shopping.manage m, online_shopping.shopping_cart sh
4 Where m.CustomerNo='C5' and m.ItemID=I.ItemID and m.CartNo=sh.CartNo and m.CustomerNO=C.CustomerNO;
5
```

<

Result Grid  Filter Rows:  Export:  Wrap Cell Content: 

|   | CustomerNO | fName   | lName  | ItemID |
|---|------------|---------|--------|--------|
| ▶ | C5         | Behrooz | Raessi | I1     |

## Some Sample Codes

```
1 CREATE TABLE Discounts( DiscountCode VARCHAR(10) NOT NULL,  
2     DiscountRate Decimal(6,2),  
3     StartDate DATE,  
4     EndDate DATE ,  
5     PRIMARY KEY (DiscountCode));  
6  
7 CREATE TABLE Define(  
8     SellerID varchar(10),  
9     DiscountCode Varchar(10),  
10    ItemID Varchar(10),  
11    PRIMARY KEY (SellerID,DiscountCode,ItemID),  
12    FOREIGN KEY (SellerID)  
13    REFERENCES online_shopping.sellers(SellerID)  
14    ON DELETE CASCADE  
15    ON UPDATE CASCADE,  
16    FOREIGN KEY (ItemID)  
17    REFERENCES online_shopping.items(ItemID)  
18    ON DELETE CASCADE  
19    ON UPDATE CASCADE,  
20    FOREIGN KEY (DiscountCode)  
21    REFERENCES online_shopping.discounts(DiscountCode)  
22    ON DELETE CASCADE  
23    ON UPDATE CASCADE);
```

```
34
35 • CREATE TABLE Shopping_Cart( CartNO VARCHAR(10) NOT NULL,
36   Quantity int(10),
37   SelectedSize Varchar(10),
38   SelectedColor Varchar(10),
39   PRIMARY KEY (CartNO));
40
41 • CREATE TABLE Manage(
42   CartNO varchar(10),
43   CustomerNO Varchar(10),
44   ItemID Varchar(10),
45   PRIMARY KEY (CartNO,CustomerNO,ItemID),
46   FOREIGN KEY (CartNO)
47   REFERENCES online_shopping.Shopping_Cart(CartNO)
48   ON DELETE NO ACTION
49   ON UPDATE NO ACTION,
50   FOREIGN KEY (ItemID)
51   REFERENCES online_shopping.items(ItemID)
52   ON DELETE NO ACTION
53   ON UPDATE NO ACTION,
54   FOREIGN KEY (CustomerNO)
55   REFERENCES online_shopping.Customers(CustomerNO)
56   ON DELETE NO ACTION
57   ON UPDATE NO ACTION);
58
```