UE21CS351A DBMS
Mini Project

# Retail Store Management System

by

Sadhika A Rao PES1UG21CS511
Roseline Jerry A PES1UG21CS500

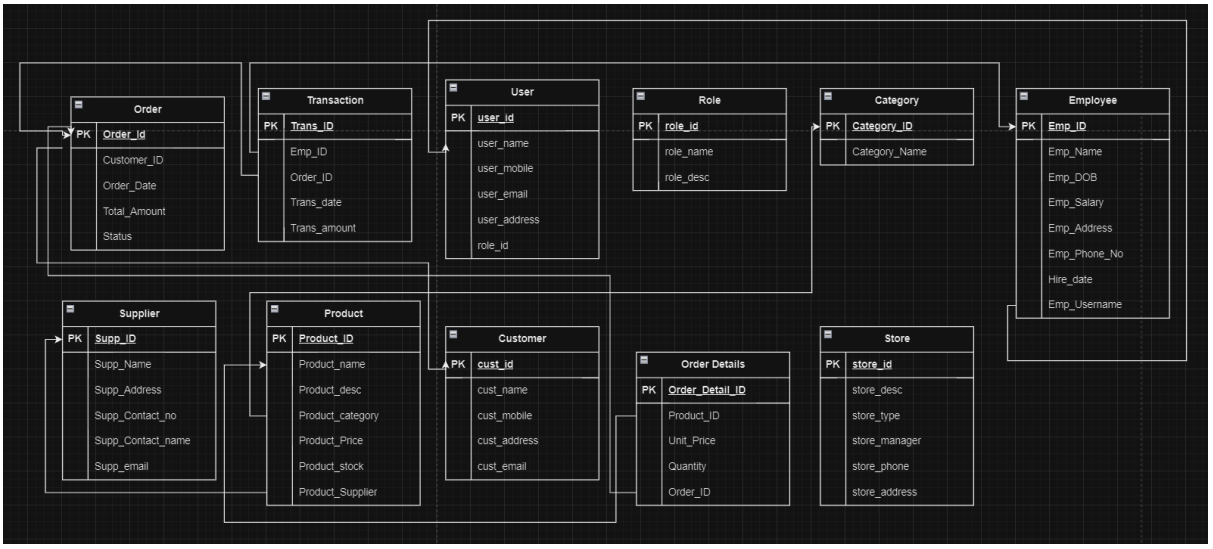
---

## Short Abstract

The Retail Store Management System is a robust software solution designed to enhance the efficiency of retail operations. With modules covering product, order, and employee management, the system offers real-time inventory tracking, order processing, and a user-friendly point-of-sale interface. Key features include secure customer registration, personalized shopping experiences, and detailed reporting for data-driven decision-making. Prioritizing security and scalability, the system provides a centralized platform for streamlined retail management, making it an invaluable tool for businesses aiming to optimize operations and enhance customer satisfaction.

## ER Diagram

# Relational Schema




# DDL SQL Commands

```
CREATE TABLE Role (
    RoleID INT PRIMARY KEY AUTO_INCREMENT,
    RoleName VARCHAR(50),
    RoleDescription VARCHAR(255)
);

CREATE TABLE User (
    UserID INT PRIMARY KEY AUTO_INCREMENT,
    Username VARCHAR(50),
    Mobile VARCHAR(20),
    Email VARCHAR(100),
    Address VARCHAR(255),
    RoleID INT,
    FOREIGN KEY (RoleID) REFERENCES Role(RoleID)
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY AUTO_INCREMENT,
    UserID INT,
    Name VARCHAR(100),
    DateOfBirth DATE,
    Salary DECIMAL(10, 2),
    Address VARCHAR(255),
    PhoneNumber VARCHAR(20),
    HireDate DATE,
    FOREIGN KEY (UserID) REFERENCES User(UserID)
);

CREATE TABLE Supplier (
    SupplierID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100),
    Address VARCHAR(255),
```

```
    PhoneNumber VARCHAR(20),
    ContactName VARCHAR(100),
    Email VARCHAR(100)
);

CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100),
    Email VARCHAR(100),
    PhoneNumber VARCHAR(20),
    Address VARCHAR(255)
);

CREATE TABLE Store (
    StoreID INT PRIMARY KEY AUTO_INCREMENT,
    Description VARCHAR(255),
    Type VARCHAR(50),
    Manager INT,
    PhoneNumber VARCHAR(20),
    Address VARCHAR(255),
    FOREIGN KEY (Manager) REFERENCES Employee(EmployeeID)
);

CREATE TABLE Category (
    CategoryID INT PRIMARY KEY AUTO_INCREMENT,
    CategoryName VARCHAR(100)
);

CREATE TABLE Product (
    ProductID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100),
    Description TEXT,
    CategoryID INT,
    SupplierID INT,
    Price DECIMAL(10, 2),
    QuantityInStock INT,
    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID),
    FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)
);

CREATE TABLE OrderTable (
    OrderID INT PRIMARY KEY AUTO_INCREMENT,
    CustomerID INT,
    OrderDate DATE,
    TotalPrice DECIMAL(10, 2),
    OrderStatus VARCHAR(50),
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
);

CREATE TABLE Transaction (
    TransactionID INT PRIMARY KEY AUTO_INCREMENT,
    OrderID INT,
```

```
    EmployeeID INT,
    TransactionDate DATE,
    TransactionAmount DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES OrderTable(OrderID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);

CREATE TABLE OrderDetails (
    OrderDetailsID INT PRIMARY KEY AUTO_INCREMENT,
    OrderID INT,
    ProductID INT,
    QuantityOrdered INT,
    Price DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES OrderTable(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);



-- Step 1: Add a new column for age
ALTER TABLE Employee
ADD age INT;

-- Update existing records with calculated age
UPDATE Employee
SET age = TIMESTAMPDIFF(YEAR,DateOfBirth,CURDATE());

ALTER TABLE Employee
    ADD FirstName VARCHAR(50),
    ADD LastName VARCHAR(50);

UPDATE Employee
SET FirstName = SUBSTRING_INDEX(Name, ' ', 1),
    LastName = SUBSTRING_INDEX(Name, ' ', -1);

ALTER TABLE Employee
    DROP COLUMN Name;
```

# CRUD operations Screenshots

Roles and Users

```
mysql> show grants for 'admin';
+-----------------------------------------------------+
| Grants for admin@%                                  |
+-----------------------------------------------------+
| GRANT USAGE ON *.* TO `admin`@`%`                   |
| GRANT ALL PRIVILEGES ON `store`.* TO `admin`@`%`    |
+-----------------------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

```
mysql> show grants for 'manager';
+-------------------------------------------------------------------------------+
| Grants for manager@%                                                          |
+-------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `manager`@`%`                                           |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `store`.`employee` TO `manager`@`%`   |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `store`.`orderdetails` TO `manager`@`%` |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `store`.`ordertable` TO `manager`@`%` |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `store`.`store` TO `manager`@`%`      |
+-------------------------------------------------------------------------------+
5 rows in set (0.00 sec)

mysql>
```

```
mysql> show grants for 'sales';
+-----------------------------------------------------+
| Grants for sales@%                                  |
+-----------------------------------------------------+
| GRANT USAGE ON *.* TO `sales`@`%`                   |
| GRANT SELECT ON `store`.`customer` TO `sales`@`%`   |
| GRANT SELECT ON `store`.`orderdetails` TO `sales`@`%` |
| GRANT SELECT ON `store`.`product` TO `sales`@`%`    |
| GRANT SELECT ON `store`.`transaction` TO `sales`@`%` |
+-----------------------------------------------------+
5 rows in set (0.00 sec)

mysql>
```

```
mysql> show grants for 'user1'@'localhost';
+-------------------------------------------+
| Grants for user1@localhost                |
+-------------------------------------------+
| GRANT USAGE ON *.* TO `user1`@`localhost` |
| GRANT `admin`@`%` TO `user1`@`localhost`  |
+-------------------------------------------+
2 rows in set (0.00 sec)
```

```
mysql> show grants for 'user2'@'localhost';
+-----------------------------------------+
| Grants for user2@localhost              |
+-----------------------------------------+
| GRANT USAGE ON *.* TO `user2`@`localhost`  |
| GRANT `manager`@`%` TO `user2`@`localhost` |
+-----------------------------------------+
2 rows in set (0.00 sec)
```

```
mysql> show grants for 'user3'@'localhost';
+-----------------------------------------+
| Grants for user3@localhost              |
+-----------------------------------------+
| GRANT USAGE ON *.* TO `user3`@`localhost` |
| GRANT `sales`@`%` TO `user3`@`localhost`  |
+-----------------------------------------+
2 rows in set (0.00 sec)

mysql> |
```

# Role Management

| | RoleID | RoleName | RoleDescription |
|---|---|---|---|
| 0 | 1 | Admin | Administrator role |
| 1 | 2 | Manager | Store manager |
| 2 | 3 | Sales | Sales representative |

Select Table

User

Select operation

Create

## Users

| | UserID | Username | Mobile | Email | Address | RoleID |
|---|---|---|---|---|---|---|
| 0 | 1 | user1 | 123-456-7890 | user1@example.com | 123 Main St, Anytown, US | 1 |
| 1 | 2 | user2 | 987-654-3210 | user2@example.com | 456 Elm St, Othertown, US | 2 |
| 2 | 3 | user3 | 555-555-5555 | user3@example.com | 789 Oak St, Anycity, US | 3 |

Employee

Select Table

Employee

Select operation

Create

## Employees

| | EmployeeID | UserID | FirstName | LastName | DateOfBirth | Salary | Address | Phon |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | John | Doe | 1985-05-10 | 60,000 | 123 Main St, Anytown, US | 111-1 |
| 1 | 2 | 2 | Jane | Smith | 1990-08-22 | 50,000 | 456 Elm St, Othertown, US | 222-2 |
| 2 | 3 | 3 | Mike | Johnson | 1988-12-05 | 70,000 | 789 Oak St, Anycity, US | 333-3 |
| 3 | 6 | 3 | Mary | Jane | 2000-11-19 | 300,000 | 891 avenue | 0976 |

Supplier:

# Suppliers

| | SupplierID | Name | Address | PhoneNumber | ContactName | Email |
|---|---|---|---|---|---|---|
| 0 | 1 | Supplier1 | 1 Supplier Ave, Town, US | 444-444-4444 | Supplier Manager | supplier1@exampl |
| 1 | 2 | Supplier2 | 2 Supplier St, City, US | 555-555-5555 | Supplier Contact | supplier2@exampl |

## Create New Supplier

Name

Supplier3

Address

3 supplier st,town,US

Phone Number

666-666-666

Contact Name

Supplier contact

Email

supplier3@gmail.com

Create Supplier

Supplier 'Supplier3' created successfully.

Select Table

Supplier

Select operation

Update

# Grocery Store Management

## Suppliers

| | SupplierID | Name | Address | PhoneNumber | ContactName | Email |
|---|---|---|---|---|---|---|
| 0 | 1 | Supplier1 | 1 Supplier Ave, Town, US | 444-444-4444 | Supplier Manager | supplier1@exampl |
| 1 | 2 | Supplier2 | 2 Supplier St, City, US | 555-555-5555 | Supplier Contact | supplier2@exampl |
| 2 | 3 | John | 4 supplier, town,US | 666-666-6666 | Supplier contact | supplier3@gmail.c |

### Update Supplier

Select Supplier to Update

Supplier1 (ID: 1)

Select Table

Supplier

Select operation

Delete

## Suppliers

| | SupplierID | Name | Address | PhoneNumber | ContactName | Email |
|---|---|---|---|---|---|---|
| 0 | 1 | Supplier1 | 1 Supplier Ave, Town, US | 444-444-4444 | Supplier Manager | supplier1@exampl |
| 1 | 2 | Supplier2 | 2 Supplier St, City, US | 555-555-5555 | Supplier Contact | supplier2@exampl |
| 2 | 3 | John | 4 supplier, town,US | 666-666-6666 | Supplier contact | supplier3@gmail.c |

### Delete Supplier

Enter Supplier ID to delete

3 − +

Delete Supplier

Supplier with ID 3 deleted.

# Grocery Store Management

## Suppliers

| | SupplierID | Name | Address | PhoneNumber | ContactName | Email |
|---|---|---|---|---|---|---|
| 0 | 1 | Supplier1 | 1 Supplier Ave, Town, US | 444-444-4444 | Supplier Manager | supplier1@exampl |
| 1 | 2 | Supplier2 | 2 Supplier St, City, US | 555-555-5555 | Supplier Contact | supplier2@exampl |

Customer:

Select Table

Customer

Select operation

Create

| | CustomerID | Name | Email | PhoneNumber | Address |
|---|---|---|---|---|---|
| 0 | 1 | Customer1 | customer1@example.com | 666-666-6666 | 321 Maple Rd, Village, US |
| 1 | 2 | Customer2 | customer2@example.com | 777-777-7777 | 654 Pine Ln, Hills, US |
| 2 | 3 | Customer3 | customer3@gmail.com | 888-888-8888 | 874 avenue |

## Create New Customer

Name

Customer4

Email

customer4@gmail.com

Phone Number

999-999-9999

Address

892 oak lane,US

Press Enter to apply

Create Customer

Select Table

Customer

Select operation

Create

## Create New Customer

Name

Customer4

Email

customer4@gmail.com

Phone Number

999-999-9999

Address

892 oak lane,US

Create Customer

Customer 'Customer4' created successfully.

Select Table

Customer

Select operation

Create

## Customers Table

| | CustomerID | Name | Email | PhoneNumber | Address |
|---|---|---|---|---|---|
| 0 | 1 | Customer1 | customer1@example.com | 666-666-6666 | 321 Maple Rd, Village, US |
| 1 | 2 | Customer2 | customer2@example.com | 777-777-7777 | 654 Pine Ln, Hills, US |
| 2 | 3 | Customer3 | customer3@gmail.com | 888-888-8888 | 874 avenue |
| 3 | 4 | Customer4 | customer4@gmail.com | 999-999-9999 | 892 oak lane,US |

## Update Customer

Select Customer to Update

Customer4 (ID: 4)

Updated Name

Jane Smith

Updated Email

customer4@gmail.com

Updated Phone Number

999-999-9999

Updated Address

892 oak lane,US

Update Customer

Customer with ID 4 updated successfully.

# Customers Table

| | CustomerID | Name | Email | PhoneNumber | Address |
|---|---|---|---|---|---|
| 0 | 1 | Customer1 | customer1@example.com | 666-666-6666 | 321 Maple Rd, Village, US |
| 1 | 2 | Customer2 | customer2@example.com | 777-777-7777 | 654 Pine Ln, Hills, US |
| 2 | 3 | Customer3 | customer3@gmail.com | 888-888-8888 | 874 avenue |
| 3 | 4 | Jane Smith | customer4@gmail.com | 999-999-9999 | 892 oak lane,US |

# Customers Table

| | CustomerID | Name | Email | PhoneNumber | Address |
|---|---|---|---|---|---|
| 0 | 1 | Customer1 | customer1@example.com | 666-666-6666 | 321 Maple Rd, Village, US |
| 1 | 2 | Customer2 | customer2@example.com | 777-777-7777 | 654 Pine Ln, Hills, US |
| 2 | 3 | Customer3 | customer3@gmail.com | 888-888-8888 | 874 avenue |
| 3 | 4 | Jane Smith | customer4@gmail.com | 999-999-9999 | 892 oak lane,US |

## Delete Customer

Enter Customer ID to delete

4 − +

Delete Customer

Customer with ID 4 deleted.

# Customers Table

| | CustomerID | Name | Email | PhoneNumber | Address |
|---|---|---|---|---|---|
| 0 | 1 | Customer1 | customer1@example.com | 666-666-6666 | 321 Maple Rd, Village, US |
| 1 | 2 | Customer2 | customer2@example.com | 777-777-7777 | 654 Pine Ln, Hills, US |
| 2 | 3 | Customer3 | customer3@gmail.com | 888-888-8888 | 874 avenue |

Store:

Select Table

Store

Select operation

Create

| | StoreID | Description | Type | Manager | PhoneNumber | Address |
|---|---|---|---|---|---|---|
| 0 | 1 | Store1 | Retail | 1 | 888-888-8888 | 789 Oak St, Anycity, US |
| 1 | 2 | Store2 | Outlet | 3 | 999-999-9999 | 456 Elm St, Othertown, US |

### Create New Store

Description

Store3

Type

Outlet2

Manager ID

1.97 − +

Phone Number

777-777-7777

Address

891 avenue, any city,US

Create Store

Store 'Store3' created successfully.

Select Table

Store

Select operation

Update

Select Store to Update

Store3 (ID: 3)

Updated Description

Store3

Updated Type

Outlet2

Updated Manager ID

2.98

Updated Phone Number

111-111-1111

Updated Address

891 avenue, oak avenue,US

Update Store

Store with ID 3 updated successfully.

# Store

| | StoreID | Description | Type | Manager | PhoneNumber | Address |
|---|---|---|---|---|---|---|
| 0 | 1 | Store1 | Retail | 1 | 888-888-8888 | 789 Oak St, Anycity, US |
| 1 | 2 | Store2 | Outlet | 3 | 999-999-9999 | 456 Elm St, Othertown, US |
| 2 | 3 | Store3 | Outlet2 | 3 | 111-111-1111 | 891 avenue, oak avenue,US |

Categories:

# Grocery Store Management

# Categories

| | CategoryID | CategoryName |
|---|---|---|
| 0 | 1 | Fruits |
| 1 | 2 | Vegetables |
| 2 | 3 | Dairy |
| 3 | 4 | Meat |

Enter category name

Frozen Desserts

Create

Category 'Frozen Desserts' created successfully.

# Categories

| | CategoryID | CategoryName |
|---|---|---|
| 0 | 1 | Fruits |
| 1 | 2 | Vegetables |
| 2 | 3 | Dairy |
| 3 | 4 | Meat |
| 4 | 5 | Frozen Desserts |

Select Table

Category

Select operation

Update

## Categories

| | CategoryID | CategoryName |
|---|---|---|
| 0 | 1 | Fruits |
| 1 | 2 | Vegetables |
| 2 | 3 | Dairy |
| 3 | 4 | Meat |
| 4 | 5 | Frozen Desserts |

Enter Category ID to update

5

Enter new name

Desserts

Update

Category with ID 5 updated successfully.

Select Table

Category

Select operation

Delete

## Categories

| | CategoryID | CategoryName |
|---|---|---|
| 0 | 1 | Fruits |
| 1 | 2 | Vegetables |
| 2 | 3 | Dairy |
| 3 | 4 | Meat |
| 4 | 5 | Desserts |

Enter Category ID to delete

5

Delete

Category with ID 5 deleted.

Select Table

Category

Select operation

Choose an option

# Grocery Store Management

## Categories

| | CategoryID | CategoryName |
|---|---|---|
| 0 | 1 | Fruits |
| 1 | 2 | Vegetables |
| 2 | 3 | Dairy |
| 3 | 4 | Meat |

Products:

Select Table

Product

Select operation

Create

## Products

| | ProductID | Name | Description | CategoryID | SupplierID | Price | QuantityInStock |
|---|---|---|---|---|---|---|---|
| 0 | 1 | Apples | ripe apples | 1 | 1 | 5.99 | 46 |
| 1 | 2 | Carrots | fresh carrots | 2 | 2 | 3.99 | 98 |
| 2 | 3 | Milk | fresh milk | 3 | 1 | 2 | 30 |

Select Table

Product

Select operation

Create

Name

Oranges

Description

fresh oranges

Category ID

1.00

Supplier ID

2.00

Price

1.99

Quantity in Stock

40.00

Create Product

Product 'Oranges' created successfully.

## Products

| | ProductID | Name | Description | CategoryID | SupplierID | Price | QuantityInStock |
|---|---|---|---|---|---|---|---|
| 0 | 1 | Apples | ripe apples | 1 | 1 | 5.99 | 46 |
| 1 | 2 | Carrots | fresh carrots | 2 | 2 | 3.99 | 98 |
| 2 | 3 | Milk | fresh milk | 3 | 1 | 2 | 30 |
| 3 | 4 | Oranges | fresh oranges | 1 | 2 | 1.99 | 40 |

Select Table

Product

Select operation

Update

Updated Description

fresh tomatoes

Updated Category ID

2.00

Updated Supplier ID

2.00

Updated Price

2.00

Updated Quantity in Stock

50.00

Update Product

Product with ID 4 updated successfully.

Select Table

Product

Select operation

Delete

## Products

| | ProductID | Name | Description | CategoryID | SupplierID | Price | QuantityInStock |
|---|---|---|---|---|---|---|---|
| 0 | 1 | Apples | ripe apples | 1 | 1 | 5.99 | 46 |
| 1 | 2 | Carrots | fresh carrots | 2 | 2 | 3.99 | 98 |
| 2 | 3 | Milk | fresh milk | 3 | 1 | 2 | 30 |
| 3 | 4 | Tomatoe | fresh tomatoe | 2 | 2 | 2 | 50 |

### Delete Product

Enter Product ID to delete

4

Delete Product

Product with ID 4 deleted.

# Products

| | ProductID | Name | Description | CategoryID | SupplierID | Price | QuantityInStock |
|---|---|---|---|---|---|---|---|
| 0 | 1 | Apples | ripe apples | 1 | 1 | 5.99 | 46 |
| 1 | 2 | Carrots | fresh carrots | 2 | 2 | 3.99 | 98 |
| 2 | 3 | Milk | fresh milk | 3 | 1 | 2 | 30 |

Order Table:

# Orders

| | OrderID | CustomerID | OrderDate | TotalPrice | OrderStatus |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2023-11-18 | 5.99 | Shipped |
| 1 | 2 | 2 | 2023-11-19 | 7.98 | Processing |
| 2 | 3 | 3 | 2023-11-19 | 11.98 | shipped |

Select Table

OrderTable

Select operation

Create

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 2 | 2023-11-19 | 7.98 | Processing |
| 2 | 3 | 3 | 2023-11-19 | 11.98 | shipped |

## Create New Order

Customer ID

3.00

Order Date

2023/11/20

Total Price

2.00

Order Status

processing

Create Order

Order created successfully.

# Orders

| | OrderID | CustomerID | OrderDate | TotalPrice | OrderStatus |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2023-11-18 | 5.99 | Shipped |
| 1 | 2 | 2 | 2023-11-19 | 7.98 | Processing |
| 2 | 3 | 3 | 2023-11-19 | 11.98 | shipped |
| 3 | 4 | 3 | 2023-11-20 | 2 | processing |

Select Table

OrderTable

Select operation

Update

## Update Order

Select Order to Update

Order ID: 4

Updated Customer ID

2.00

Updated Order Date

2023/11/19

Updated Total Price

2.00

Updated Order Status

shipped

Update Order

Order with ID 4 updated successfully.

# Orders

| | OrderID | CustomerID | OrderDate | TotalPrice | OrderStatus |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2023-11-18 | 5.99 | Shipped |
| 1 | 2 | 2 | 2023-11-19 | 7.98 | Processing |
| 2 | 3 | 3 | 2023-11-19 | 11.98 | shipped |
| 3 | 4 | 2 | 2023-11-19 | 2 | shipped |

Select Table

OrderTable

Select operation

Delete

## Orders

| | OrderID | CustomerID | OrderDate | TotalPrice | OrderStatus |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2023-11-18 | 5.99 | Shipped |
| 1 | 2 | 2 | 2023-11-19 | 7.98 | Processing |
| 2 | 3 | 3 | 2023-11-19 | 11.98 | shipped |
| 3 | 4 | 2 | 2023-11-19 | 2 | shipped |

## Delete Order

Enter Order ID to delete

4

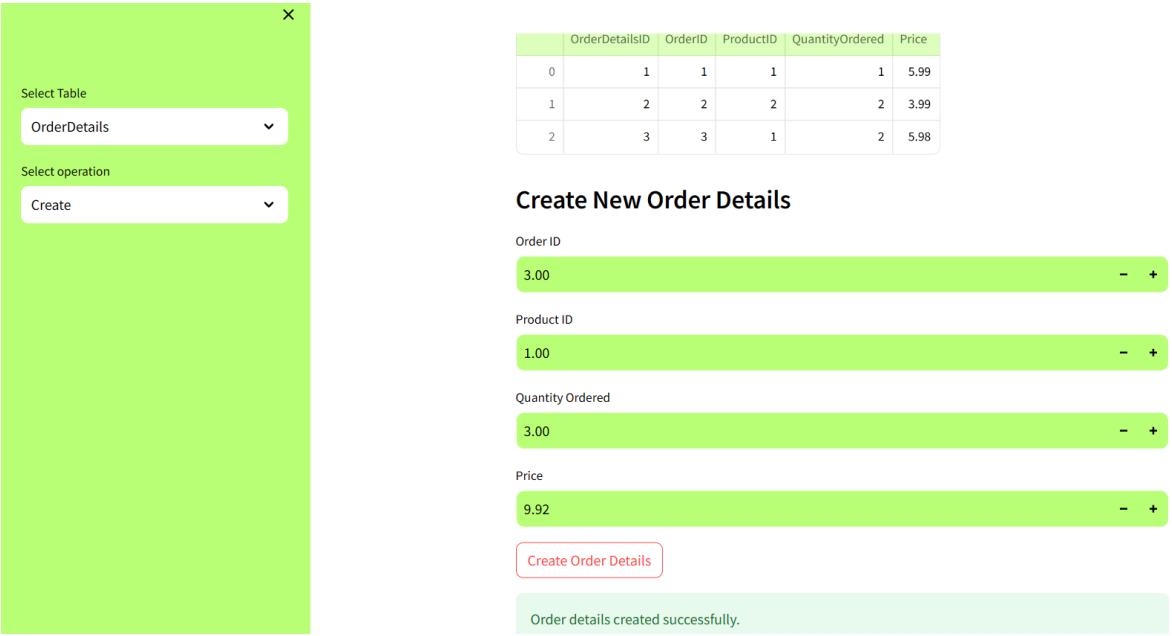
Delete Order

Order with ID 4 deleted.

# Orders

| | OrderID | CustomerID | OrderDate | TotalPrice | OrderStatus |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2023-11-18 | 5.99 | Shipped |
| 1 | 2 | 2 | 2023-11-19 | 7.98 | Processing |
| 2 | 3 | 3 | 2023-11-19 | 11.98 | shipped |

Order Details:




# Order Details

| | OrderDetailsID | OrderID | ProductID | QuantityOrdered | Price |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 5.99 |
| 1 | 2 | 2 | 2 | 2 | 3.99 |
| 2 | 3 | 3 | 1 | 2 | 5.98 |
| 3 | 4 | 3 | 1 | 3 | 9.92 |

## Update Order Details

Select Order Details to Update

Order Details ID: 4

Updated Order ID

2.00

Updated Product ID

2.00

Updated Quantity Ordered

3.00

Updated Price

9.92

Update Order Details

Order details with ID 4 updated successfully.

# Order Details

| | OrderDetailsID | OrderID | ProductID | QuantityOrdered | Price |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 5.99 |
| 1 | 2 | 2 | 2 | 2 | 3.99 |
| 2 | 3 | 3 | 1 | 2 | 5.98 |
| 3 | 4 | 2 | 2 | 3 | 9.92 |

# Order Details

| | OrderDetailsID | OrderID | ProductID | QuantityOrdered | Price |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 5.99 |
| 1 | 2 | 2 | 2 | 2 | 3.99 |
| 2 | 3 | 3 | 1 | 2 | 5.98 |
| 3 | 4 | 2 | 2 | 3 | 9.92 |

## Delete Order Details

Enter Order Details ID to delete

4 − +

Delete Order Details

Order details with ID 4 deleted.

# Order Details

| | OrderDetailsID | OrderID | ProductID | QuantityOrdered | Price |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 5.99 |
| 1 | 2 | 2 | 2 | 2 | 3.99 |
| 2 | 3 | 3 | 1 | 2 | 5.98 |

Transactions:

Select Table

Transaction

Select operation

Create

| | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 2023-11-19 | 5.99 |
| 1 | 2 | 2 | 3 | 2023-11-19 | 7.98 |
| 2 | 3 | 3 | 2 | 2023-11-19 | 11.98 |

## Create New Transaction

Order ID

3.00 − +

Employee ID

2.00 − +

Transaction Date

2023/11/20

Transaction Amount

2.00 − +

Create Transaction

Transaction created successfully.

Select Table

Transaction

Select operation

Create

# Grocery Store Management

## Transactions

| | TransactionID | OrderID | EmployeeID | TransactionDate | TransactionAmount |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 2023-11-19 | 5.99 |
| 1 | 2 | 2 | 3 | 2023-11-19 | 7.98 |
| 2 | 3 | 3 | 2 | 2023-11-19 | 11.98 |
| 3 | 4 | 3 | 2 | 2023-11-20 | 2 |

### Create New Transaction

Select Table

Transaction

Select operation

Update

### Update Transaction

Select Transaction to Update

Transaction ID: 4

Updated Order ID

3.00

Updated Employee ID

1.00

Updated Transaction Date

2023/11/18

Updated Transaction Amount

3.00

Update Transaction

Transaction with ID 4 updated successfully.

Select Table

Transaction

Select operation

Update

# Grocery Store Management

## Transactions

| | TransactionID | OrderID | EmployeeID | TransactionDate | TransactionAmount |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 2023-11-19 | 5.99 |
| 1 | 2 | 2 | 3 | 2023-11-19 | 7.98 |
| 2 | 3 | 3 | 2 | 2023-11-19 | 11.98 |
| 3 | 4 | 3 | 1 | 2023-11-18 | 3 |

Select Table

Transaction

Select operation

Delete

# Grocery Store Management

## Transactions

| | TransactionID | OrderID | EmployeeID | TransactionDate | TransactionAmount |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 2023-11-19 | 5.99 |
| 1 | 2 | 2 | 3 | 2023-11-19 | 7.98 |
| 2 | 3 | 3 | 2 | 2023-11-19 | 11.98 |
| 3 | 4 | 3 | 1 | 2023-11-18 | 3 |

### Delete Transaction

Enter Transaction ID to delete

4

Delete Transaction

Transaction with ID 4 deleted.

# Transactions

| | TransactionID | OrderID | EmployeeID | TransactionDate | TransactionAmount |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 2023-11-19 | 5.99 |
| 1 | 2 | 2 | 3 | 2023-11-19 | 7.98 |
| 2 | 3 | 3 | 2 | 2023-11-19 | 11.98 |

# List of Functionalities

1. CRUD Operations
   - **Store**, **Customer**, **Employee**, **Category**, **Product**, **Order**, **OrderDetails**, and **Transactions**: Comprehensive operations to Create, Read, Update, and Delete data within respective tables/entities.

2. Employee Operations
   Age Calculation: Deriving an employee's age from their date of birth.

   Fetching by Date of Birth: Retrieving employees based on their date of birth.
   Years of Experience: Calculating an employee's years of experience within the organization.

3. Customer-Related Operations
   - **Order Retrieval**: Fetching orders based on customer ID.
   - **Status-Based Order Fetch**: Retrieving orders based on their status.

4. Data Retrieval
   - **Employee Details by ID**
   - **Store Details by ID**
   - **Product Details by ID**

5. Inventory Management
   - **Low Stock Check**: Monitoring and detecting low stock levels for products.

6. Sales Analysis
   - **Top-Selling Products Identification**
   - **Total Price Computation**: Determining the total price from all orders.
   - **Average Product Price Calculation**

7. Financial Tracking
   - **Transactions by Employee ID**: Retrieving transactions and total amount handled by an employee.

8. Analytics

- Average Price by Category: Finding the average price within different product categories.

9. Error Handling
   - **Trigger for Missing Category ID**: Activation when a product lacks a category ID.
   - **Trigger for Missing Customer ID**: Activation when an order is missing a customer ID.

# Procedures/Functions/Triggers

1. CRUD Operations

   - **Store**, **Customer**, **Employee**, **Category**, **Product**, **Order**, **OrderDetails**, and **Transactions**: Comprehensive operations to Create, Read, Update, and Delete data within respective tables/entities.

2. Employee Operations

Age Calculation: Deriving an employee's age from their date of birth.

Fetching by Date of Birth: Retrieving employees based on their date of birth.

Procedure:

```
DELIMITER //

CREATE PROCEDURE GetEmployeesByDOB(IN date_of_birth DATE)
BEGIN
    CREATE TEMPORARY TABLE tempEmployees AS
    SELECT *
    FROM Employee
    WHERE DateOfBirth = date_of_birth;

    SELECT * FROM tempEmployees;

    DROP TEMPORARY TABLE IF EXISTS tempEmployees;
END //

DELIMITER ;
```

Output:

## Employee Lookup by Date of Birth

Select Date of Birth

2000/11/19

Get Employees

| | 0 |
|---|---|
| 0 | 6 |
| 1 | 3 |
| 2 | Mary |
| 3 | Jane |
| 4 | 2000-11-19 |
| 5 | 300000.00 |
| 6 | 891 avenue |
| 7 | 0976667736 |
| 8 | 2023-11-19 |
| 9 | 23 |

Years of Experience: Calculating an employee's years of experience within the organization.

Function:

```
DELIMITER //

CREATE FUNCTION CalculateExperienceLength(employee_id INT)
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE hire_date DATE;
    DECLARE experience_length INT;

    SELECT HireDate INTO hire_date FROM Employee WHERE EmployeeID =
employee_id;

    SET experience_length = TIMESTAMPDIFF(YEAR,hire_date,CURDATE());

    RETURN experience_length;
END//

DELIMITER ;
```

Output:

## Calculate Employee Experience Length

Enter Employee ID:

1

Calculate

Experience for Employee ID 1: 3 years

3. **Customer-Related Operations**
 - **Order Retrieval**: Fetching orders based on customer ID.

```
query = "SELECT * FROM OrderTable WHERE CustomerID = %s"
```

Output:

## Customer Order History

Customer ID

1.00

Fetch Order History

Order History:

Order ID: 1

Date: 2023-11-18

Total Price: 5.99

Status: Shipped

 - **Status-Based Order Fetch**: Retrieving orders based on their status.

Procedure:

```
DELIMITER //

CREATE PROCEDURE GetOrdersByStatus(IN order_status VARCHAR(50))
BEGIN
    SELECT *
    FROM OrderTable
    WHERE OrderStatus = order_status;
END//

DELIMITER ;
```

Output:

## Order Status Lookup

Enter Order Status (e.g.,shipped):

shipped

Fetch Orders

| | 0 |
|---|---|
| 0 | 1 |
| 1 | 1 |
| 2 | 2023-11-18 |
| 3 | 5.99 |
| 4 | Shipped |

| | 0 |
|---|---|
| 0 | 3 |
| 1 | 3 |
| 2 | 2023-11-19 |
| 3 | 11.98 |
| 4 | shipped |

4. **Data Retrieval**
 - **Employee Details by ID**

```
query = f"SELECT * FROM Employee WHERE EmployeeID = {employee_id}"
```

Output:

## Fetch Employee Details

Enter Employee ID:

1

Fetch Employee Details

Employee Details:

| | 0 |
|---|---|
| 0 | 1 |
| 1 | 1 |
| 2 | John |
| 3 | Doe |
| 4 | 1985-05-10 |
| 5 | 60000.00 |
| 6 | 123 Main St, Anytown, US |
| 7 | 111-111-1111 |
| 8 | 2020-01-15 |
| 9 | 38 |

- **Store Details by ID**

```
query = f"SELECT * FROM Product WHERE ProductID = {product_id}"
```

Output:

## Fetch Store Details

Enter Store ID:

1

Fetch Store Details

Store Details:

| | 0 |
|---|---|
| 0 | 1 |
| 1 | Store1 |
| 2 | Retail |
| 3 | 1 |
| 4 | 888-888-8888 |
| 5 | 789 Oak St, Anycity, US |

- **Product Details by ID**

```
query = f"SELECT * FROM Store WHERE StoreID = {store_id}"
```

Output:

## Fetch Product Details

Enter Product ID:

1

Fetch Product Details

Product Details:

| | 0 |
|---|---|
| 0 | 1 |
| 1 | Apples |
| 2 | ripe apples |
| 3 | 1 |
| 4 | 1 |
| 5 | 5.99 |
| 6 | 46 |

5. **Inventory Management**

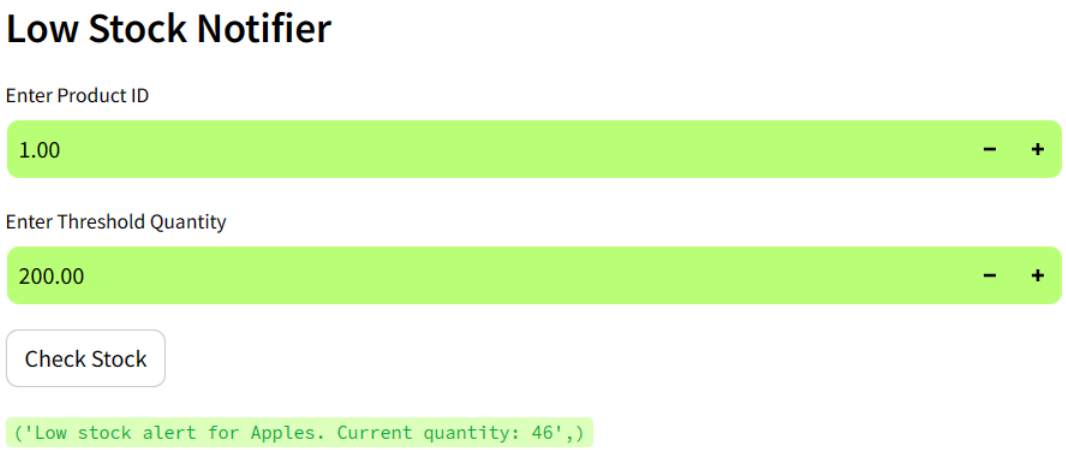
- **Low Stock Check**: Monitoring and detecting low stock levels for products.

Procedure:

```
DELIMITER //

CREATE PROCEDURE NotifyLowStock(IN product_id INT, IN threshold_quantity
INT)
BEGIN
    DECLARE current_quantity INT;
    DECLARE product_name VARCHAR(100);

    -- Get the current quantity of the specified product
    SELECT QuantityInStock, Name INTO current_quantity, product_name
    FROM Product
    WHERE ProductID = product_id;

    -- Check if the current quantity is below the threshold
    IF current_quantity < threshold_quantity THEN
        -- Use any notification mechanism you have (e.g., printing to
console, sending an email)
        SELECT CONCAT('Low stock alert for ', product_name, '. Current
quantity: ', current_quantity) AS Notification;
    ELSE
        SELECT 'Stock level is satisfactory' AS Notification;
    END IF;
END//
```

DELIMITER ;

Output:



6. **Sales Analysis**

- **Top-Selling Products Identification**

```
query = """
    SELECT p.Name, SUM(od.QuantityOrdered) as TotalQuantitySold
    FROM Product p
    JOIN OrderDetails od ON p.ProductID = od.ProductID
    GROUP BY p.ProductID
    ORDER BY TotalQuantitySold DESC
    LIMIT %s
    """
```

Output:

# Top Selling Products

Apples - Total Quantity Sold: 3

Carrots - Total Quantity Sold: 2

- **Total Price Computation**: Determining the total price from all orders.

```
query = "SELECT SUM(TransactionAmount) FROM Transaction"
```

# Store Analytics

Average Price of Products: $3.99

Total Transaction Amount: $25.95

- **Average Product Price Calculation**

```
query = "SELECT AVG(Price) FROM Product"
```

Output:

# Store Analytics

Average Price of Products: $3.99

Total Transaction Amount: $25.95

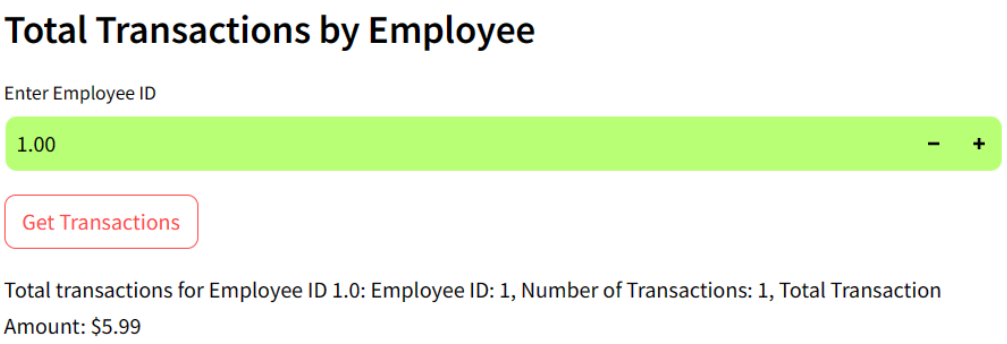
7. **Financial Tracking**

- **Transactions by Employee ID**: Retrieving transactions and total amount handled by an employee.

Function:

```
DELIMITER //

CREATE FUNCTION GetEmployeeTransactionInfo(employee_id INT)
RETURNS VARCHAR(255)
DETERMINISTIC
BEGIN
    DECLARE num_transactions INT;
    DECLARE total_amount DECIMAL(10, 2);

    SELECT COUNT(TransactionID), SUM(TransactionAmount)
    INTO num_transactions, total_amount
    FROM `Transaction`
    WHERE EmployeeID = employee_id;

    RETURN CONCAT('Employee ID: ', employee_id,
                  ', Number of Transactions: ', num_transactions,
                  ', Total Transaction Amount: $', total_amount);
END//


DELIMITER ;
```

Output:



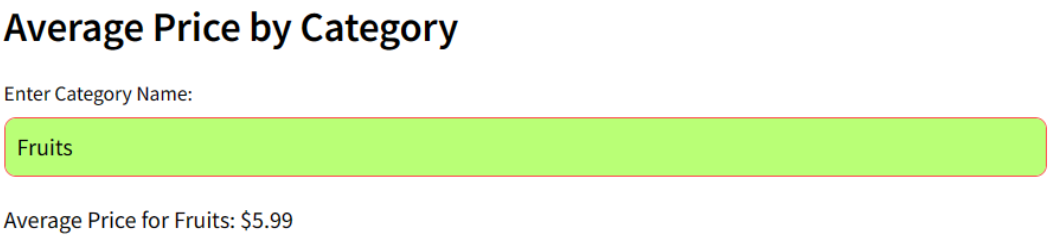

8. **Analytics**

- **Average Price by Category**: Finding the average price within different product categories.

Nested Query:

```
# Nested query to calculate average price of products in a specific category
    query = f"""
    SELECT AVG(Price)
    FROM Product
    WHERE CategoryID = (
        SELECT CategoryID
        FROM Category
        WHERE CategoryName = %s
    )
    """
```

Output:




9. **Error Handling**

- **Trigger for Missing Category ID**: Activation when a product lacks a category ID.

Trigger:

```
DELIMITER //

CREATE TRIGGER Before_Insert_Product
```

```
BEFORE INSERT ON Product
FOR EACH ROW
BEGIN
    DECLARE categoryCount INT;

    SELECT COUNT(*)
    INTO categoryCount
    FROM Category
    WHERE CategoryID = NEW.CategoryID;

    IF categoryCount = 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Category does not exist in the Category
table';
    END IF;
END //

DELIMITER ;
```

Output:


### Trigger for Missing Customer ID:

Activation when an order is missing a customer ID.

Trigger:

```
DELIMITER //

CREATE TRIGGER Before_Insert_OrderTable
BEFORE INSERT ON OrderTable
FOR EACH ROW
```

```
BEGIN
    DECLARE customerCount INT;

    SELECT COUNT(*)
    INTO customerCount
    FROM Customer
    WHERE CustomerID = NEW.CustomerID;

    IF customerCount = 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Customer does not exist in the Customer
table';
    END IF;
END //

DELIMITER ;
```

Output: