🛒 E-COMMERCE DATABASE 🛍️

Project Purpose 🎯

This project aims to design and implement an efficient database system for an e-commerce platform. The database manages critical components such as customers, products, orders, and payments, ensuring smooth business operations. Key objectives include:

- **Data Integrity 🔒 :** Ensuring consistency through well-defined relationships using primary and foreign keys.
- **Efficiency ⚡ :** Facilitating fast data retrieval and transaction processing.
- **Scalability 📈 :** Supporting business growth by allowing easy integration of additional features.
- **Security 🔐 :** Protecting customer and transaction data with structured access controls.

1. Database Creation 🗃️

Query:

```
CREATE DATABASE ECOMMERCEDB;
USE ECOMMERCEDB;
```

Purpose 🔎 :

Creates a database named ECOMMERCEDB and selects it for use.

Explanation 👉 :

The CREATE DATABASE statement initializes a new database, while USE sets the active database for executing further queries.

---

2. Table Creation 📊

Customers Table 👥

```
CREATE Table Customers (
CustomerID INT PRIMARY KEY,
Customer_Name VARCHAR (50),
Email VARCHAR (50),
C_Password VARCHAR (50),
C_Address VARCHAR (50));
```

Purpose 🔎:

Stores customer details such as ID, name, email, password, and address.

Explanation 👉:

- CustomerID is the primary key, ensuring unique customer identification.
- Other columns store customer-related details.

Products Table 📦

```
CREATE TABLE Products (
ProductID INT PRIMARY KEY,
Product_Name VARCHAR (50),
Price INT,
StockQuantity INT);
```

Purpose 🔎:

Stores product details, including ID, name, price, and stock quantity.

Explanation 👉:

- ProductID uniquely identifies products.
- Price stores the product cost.
- StockQuantity tracks inventory.

Orders Table 🛒

```
CREATE TABLE ORDERS (
OrderID INT PRIMARY KEY,
CustomerID INT,
OrderDate DATE,
TotalCost INT,
FOREIGN KEY (CustomerID) REFERENCES Customers (CustomerID));
```

Purpose 🔎:

Stores order-related details such as order date and total cost.

Explanation 👉:

- OrderID is the primary key.
- CustomerID establishes a foreign key relationship with Customers.
- TotalCost stores the total value of an order.

Order Items Table 📝

```
CREATE TABLE ORDERITEM (
OrderItemID INT PRIMARY KEY,
```

```
OrderID INT,
ProductID INT,
Quantity INT,
FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID),
FOREIGN KEY (ProductID) REFERENCES Products (ProductID));
```

Purpose 🔎:

Links products to specific orders, defining the quantity of each item in an order.

Explanation 👉:

- OrderItemID uniquely identifies each order item.
- OrderID links to the Orders table.
- ProductID links to the Products table.

Payments Table 💳

```
CREATE TABLE PAYMENTS (
PaymentID INT PRIMARY KEY,
OrderId INT,
PaymentMethod VARCHAR (50),
FOREIGN KEY (OrderID) REFERENCES ORDERS (OrderID));
```

Purpose 🔎:

Stores payment information related to orders.

Explanation 👉:

- PaymentID uniquely identifies each transaction.
- OrderID links to the corresponding order.
- PaymentMethod stores the method used for payment.

---

3. Data Insertion 📥

Insert Data into Customers Table 👥

```
INSERT INTO customers (CustomerID, Customer_Name, Email, C_Password,
C_Address)
VALUES
(1, 'John Doe', 'john.doe@example.com', 'password123', '123 Main St'),
(2, 'Jane Smith', 'jane.smith@example.com', 'password123', '456 Elm St'),
(3, 'Bob Johnson', 'bob.johnson@example.com', 'password123', '789 Oak
St'),(4, 'Alice Williams', 'alice.williams@example.com', 'password123',
'1011 Maple St'),
(5, 'Mike Davis', 'mike.davis@example.com', 'password123', '1213 Pine
St'),
(6, 'Emily Miller', 'emily.miller@example.com', 'password123', '1415 Cedar
St'),
```

```
(7, 'David Wilson', 'david.wilson@example.com', 'password123', '1617
Spruce St'),
(8, 'Sarah Taylor', 'sarah.taylor@example.com', 'password123', '1819 Fir
St'),
(9, 'Kevin White', 'kevin.white@example.com', 'password123', '2021 Birch
St'),
(10, 'Lisa Harris', 'lisa.harris@example.com', 'password123', '2223 Beech
St'),
(11, 'Peter Martin', 'peter.martin@example.com', 'password123', '2425 Ash
St'),
(12, 'Helen Thompson', 'helen.thompson@example.com', 'password123', '2627
Cypress St'),
(13, 'Michael Jenkins', 'michael.jenkins@example.com', 'password123',
'2829 Dogwood St'),
(14, 'Laura Russell', 'laura.russell@example.com', 'password123', '3031
Elm St'),
(15, 'Daniel Hall', 'daniel.hall@example.com',
'password123', '3233 Oak St');
```

Purpose 🔎:

Adds a new customer to the database.

Explanation 👉:

Each column is populated with respective data.

Insert Data into Products Table 📦

```
INSERT INTO products (ProductID, Product_Name, Price, StockQuantity)
VALUES
(1, 'Apple iPhone 13', 999, 100),
(2, 'Samsung 4K TV', 1299, 50),
(3, 'Nike Air Max Shoes', 99, 200),
(4, 'Dell Inspiron Laptop', 699, 150),
(5, 'Sony PlayStation 5', 499, 100),
(6, 'Canon EOS Camera', 799, 50),
(7, 'LG Refrigerator', 1999, 20),
(8, 'Microsoft Xbox Series X', 599, 50),
(9, 'Asus ROG Gaming Laptop', 1499, 20),
(10, 'Bose SoundLink Speaker', 299, 100),
(11, 'HP Envy Laptop', 899, 50),
(12, 'JBL Flip Speaker', 99, 200),
(13, 'Lenovo ThinkPad Laptop', 1299, 20),
(14, 'Netgear Wi-Fi Router', 199, 100),
(15, 'Panasonic 4K Camera', 999, 20);
```

Purpose 🔎:

Adds a new product.

Explanation 👉:

Populates the Products table with product details.

4. Data Retrieval Queries 🔍

Retrieve All Customers 👥

```
SELECT * FROM Customers;
```

150 %

Results | Messages

	CustomerID	Customer_Name	Email	C_Password	C_Address
1	1	John Doe	john.doe@example.com	password123	123 Main St
2	2	Jane Smith	jane.smith@example.com	password123	456 Elm St
3	3	Bob Johnson	bob.johnson@example.com	password123	789 Oak St
4	4	Alice Williams	alice.williams@example.com	password123	1011 Maple St
5	5	Mike Davis	mike.davis@example.com	password123	1213 Pine St
6	6	Emily Miller	emily.miller@example.com	password123	1415 Cedar St
7	7	David Wilson	david.wilson@example.com	password123	1617 Spruce...
8	8	Sarah Taylor	sarah.taylor@example.com	password123	1819 Fir St
9	9	Kevin White	kevin.white@example.com	password123	2021 Birch St
10	10	Lisa Harris	lisa.harris@example.com	password123	2223 Beech St
11	11	Peter Martin	peter.martin@example.com	password123	2425 Ash St
12	12	Helen Thompson	helen.thompson@example....	password123	2627 Cypres...
13	13	Michael Jenkins	michael.jenkins@example....	password123	2829 Dogwo...
14	14	Laura Russell	laura.russell@example.com	password123	3031 Elm St
15	15	Daniel Hall	daniel.hall@example.com	password123	3233 Oak St

Purpose 🔎:

Fetches all customer data.

Retrieve All Products 📦

```
SELECT * FROM Products;
```

150 %

Results | Messages

	ProductID	Product_Name	Price	StockQuantity
1	1	Apple iPhone 13	999	100
2	2	Samsung 4K TV	1299	50
3	3	Nike Air Max Shoes	99	200
4	4	Dell Inspiron Laptop	699	150
5	5	Sony PlayStation 5	499	100
6	6	Canon EOS Camera	799	50
7	7	LG Refrigerator	1999	20
8	8	Microsoft Xbox Series X	599	50
9	9	Asus ROG Gaming Laptop	1499	20
10	10	Bose SoundLink Speaker	299	100
11	11	HP Envy Laptop	899	50
12	12	JBL Flip Speaker	99	200
13	13	Lenovo ThinkPad Laptop	1299	20
14	14	Netgear Wi-Fi Router	199	100
15	15	Panasonic 4K Camera	999	20

Purpose 🔎:

Fetches all product data.

Retrieve All Orders 🛒

```
Select * from ORDERS;
```

150 %

Results | Messages

	OrderID	CustomerID	OrderDate	TotalCost
1	1	1	2022-01-01	100
2	2	1	2022-01-15	200
3	3	2	2022-02-01	50
4	4	3	2022-03-01	150
5	5	1	2022-04-01	250
6	6	2	2022-05-01	75
7	7	3	2022-06-01	300
8	8	1	2022-07-01	400
9	9	2	2022-08-01	100
10	10	3	2022-09-01	200
11	11	1	2022-10-01	350
12	12	2	2022-11-01	125
13	13	3	2022-12-01	450
14	14	1	2023-01-01	500
15	15	2	2023-02-01	175

Purpose 🔎:

Fetches all order records.

Retrieve All OrderItem 📝

```
SELECT *FROM ORDERITEM;
```

150 %

Results | Messages

	OrderItemID	OrderID	ProductID	Quantity
1	1	1	1	1
2	2	1	2	1
3	3	2	3	2
4	4	3	1	1
5	5	3	4	1
6	6	4	5	1
7	7	5	2	1
8	8	5	6	1
9	9	6	7	1
10	10	7	8	1
11	11	7	9	1
12	12	8	10	2
13	13	9	11	1
14	14	10	12	1
15	15	10	13	1

Purpose 🔎:

Fetches all OrderItem records.

Retrieve All Payments 💳

```
SELECT *FROM PAYMENTS;
```

150 %

Results | Messages

	PaymentID	OrderId	PaymentMethod
1	1	1	Credit Card
2	2	2	PayPal
3	3	3	Bank Transfer
4	4	4	Credit Card
5	5	5	PayPal
6	6	6	Bank Transfer
7	7	7	Credit Card
8	8	8	PayPal
9	9	9	Bank Transfer
10	10	10	Credit Card
11	11	11	PayPal
12	12	12	Bank Transfer
13	13	13	Credit Card
14	14	14	PayPal
15	15	15	Bank Transfer

Purpose 🔎:

Fetches all OrderItem records.

Retrieve Customer Order History 🕒

```
SELECT c.Customer_Name ,o.OrderID, o.OrderDate, oi.ProductID, p.Product_Name
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID
JOIN OrderItem oi ON o.OrderID = oi.OrderID
JOIN Products p ON oi.ProductID = p.ProductID
WHERE c.CustomerID = 1;
```

150 %

Results | Messages

	Customer_Name	OrderID	OrderDate	ProductID	Product_Name
1	John Doe	1	2022-01-01	1	Apple iPhone 13
2	John Doe	1	2022-01-01	2	Samsung 4K TV
3	John Doe	2	2022-01-15	3	Nike Air Max Shoes
4	John Doe	5	2022-04-01	2	Samsung 4K TV
5	John Doe	5	2022-04-01	6	Canon EOS Camera
6	John Doe	8	2022-07-01	10	Bose SoundLink Speaker

Purpose 🔎:

Fetches order history of a specific customer.

Explanation 👉:

- Joins multiple tables (Customers, Orders, OrderItem, Products).
- Filters records for CustomerID = 1.

Retrieve Product Sales Report 📊

```sql
SELECT p.Product_Name, SUM(oi.Quantity) AS TotalQuantitySold
FROM Products p
JOIN OrderItem oi ON p.ProductID = oi.ProductID
GROUP BY p.Product_Name;
```

150 %

Results | Messages

	Product_Name	TotalQuantitySold
1	Apple iPhone 13	2
2	Asus ROG Gaming Laptop	1
3	Bose SoundLink Speaker	2
4	Canon EOS Camera	1
5	Dell Inspiron Laptop	1
6	HP Envy Laptop	1
7	JBL Flip Speaker	1
8	Lenovo ThinkPad Laptop	1
9	LG Refrigerator	1
10	Microsoft Xbox Series X	1
11	Nike Air Max Shoes	2
12	Samsung 4K TV	2
13	Sony PlayStation 5	1

Purpose 🔎:

Calculates total sales for each product.

Explanation 👉:

- Joins Products and OrderItem tables.
- Uses SUM(Quantity) to compute total units sold.
- Groups by Product_Name to get aggregated results.

---

5. Conclusion 🏁

The e-commerce database project has successfully designed and implemented a comprehensive database system to manage critical components of an online shopping platform. The database efficiently stores and retrieves data related to customers, products, orders, and payments, ensuring smooth business operations.

Key Achievements 🪄

- **Data Integrity** 🔒**:** Well-defined relationships between tables using primary and foreign keys ensure data consistency.
- **Efficiency** ⚡**:** Optimized queries facilitate fast data retrieval and transaction processing.
- **Scalability** 📈**:** The database design supports business growth by allowing easy integration of additional features.

The e-commerce database project demonstrates a solid foundation for managing the complexities of online shopping platforms. With ongoing maintenance and enhancements, this database system can support the growth and success of e-commerce businesses.