**Project**

**on**

**Online Retail Application**

**DatabaseDocumentation**

**1. Introduction**

**1.1 Project Overview**

The Online Retail Application Database project is designed to manage customer registrations, product purchases, and transaction processes for an online retail platform. The database supports functionalities such as customer management, product catalog management, order processing, and payment handling.

**1.2 Objectives**

* To create a robust database schema for managing an online retail application.
* To ensure data integrity and efficient data retrieval through relational database design.
* To provide a scalable solution for handling customer orders and transactions.

**2. Database Schema**

**2.1 Entities and Attributes**

**2.1.1 Customers**

* **CustomerID** (INT, Primary Key, Auto Increment): Unique identifier for the customer.
* **Name** (VARCHAR(100)): Customer’s full name.
* **Address** (TEXT): Customer’s address.
* **ContactInformation** (VARCHAR(50)): Contact phone number or similar.
* **Email** (VARCHAR(100), Unique): Customer’s email address.
* **PasswordHash** (VARCHAR(255)): Hashed password for authentication.

**2.1.2 Products**

* **ProductID** (INT, Primary Key, Auto Increment): Unique identifier for the product.
* **Name** (VARCHAR(100)): Name of the product.
* **Description** (TEXT): Detailed description of the product.
* **Price** (DECIMAL(10, 2)): Price of the product.
* **StockQuantity** (INT): Number of units available in stock.

**2.1.3 Orders**

* **OrderID** (INT, Primary Key, Auto Increment): Unique identifier for the order.
* **CustomerID** (INT, Foreign Key): Reference to the customer who placed the order.
* **OrderDate** (DATETIME): Date and time when the order was placed.
* **TotalAmount** (DECIMAL(10, 2)): Total amount of the order.

**2.1.4 OrderItems**

* **OrderItemID** (INT, Primary Key, Auto Increment): Unique identifier for the order item.
* **OrderID** (INT, Foreign Key): Reference to the order.
* **ProductID** (INT, Foreign Key): Reference to the product.
* **Quantity** (INT): Quantity of the product ordered.
* **UnitPrice** (DECIMAL(10, 2)): Price of the product at the time of order.

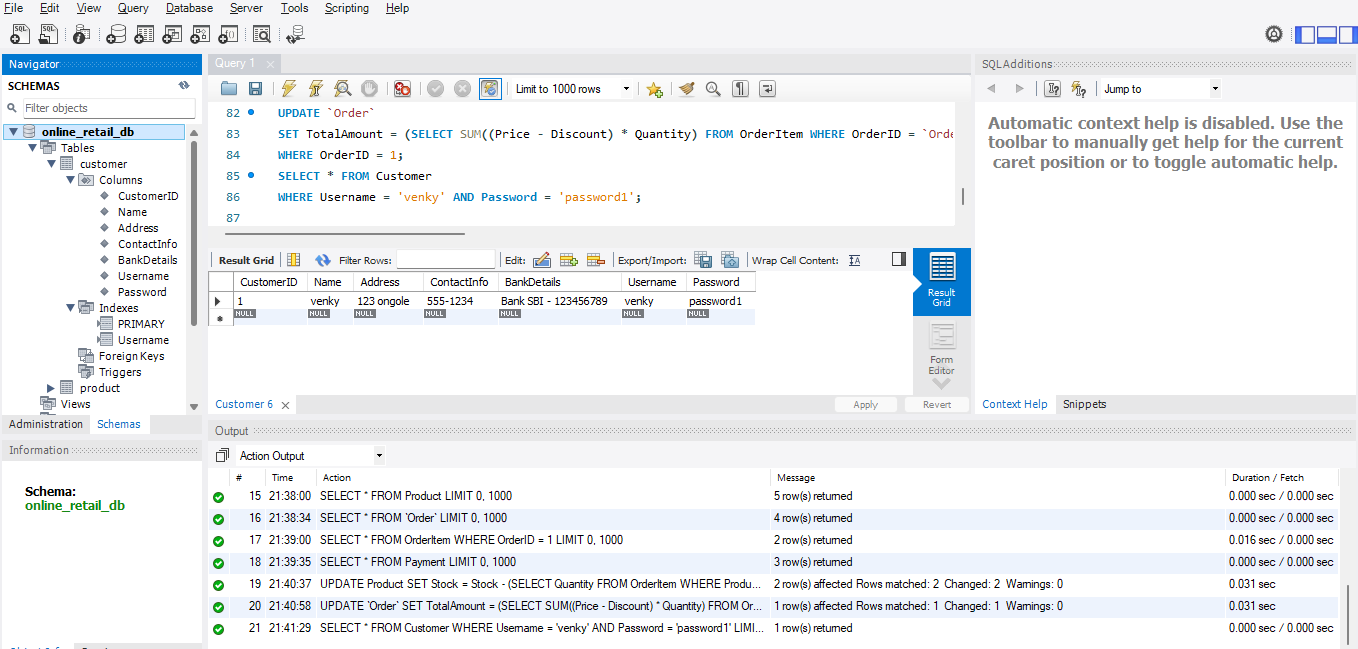
**2.1.5 Payments**

* **PaymentID** (INT, Primary Key, Auto Increment): Unique identifier for the payment.
* **OrderID** (INT, Foreign Key): Reference to the order.
* **PaymentDate** (DATETIME): Date and time when the payment was made.
* **PaymentMethod** (VARCHAR(50)): Method of payment (e.g., credit card, PayPal).
* **Amount** (DECIMAL(10, 2)): Amount paid.

**2.2 Relationships**

* **Customer to Orders**: One-to-Many (One customer can have multiple orders).
* **Order to OrderItems**: One-to-Many (One order can have multiple items).
* **Product to OrderItems**: One-to-Many (One product can appear in multiple order items).
* **Order to Payments**: One-to-One (Each order has one payment record).

**3. Database Schema Diagram**

****

**4. SQL Scripts**

**4.1 Create Database**

CREATE DATABASE OnlineRetailDB;

USE OnlineRetailDB;

**4.2 Create Tables**

🡪 Create Customers Table

CREATE TABLE Customers (

CustomerID INT AUTO\_INCREMENT PRIMARY KEY,

Name VARCHAR(100),

Address TEXT,

ContactInformation VARCHAR(50),

Email VARCHAR(100) UNIQUE,

PasswordHash VARCHAR(255)

);

🡪Create Products Table

CREATE TABLE Products (

ProductID INT AUTO\_INCREMENT PRIMARY KEY,

Name VARCHAR(100),

Description TEXT,

Price DECIMAL(10, 2),

StockQuantity INT

);

🡪 Create Orders Table

CREATE TABLE Orders (

OrderID INT AUTO\_INCREMENT PRIMARY KEY,

CustomerID INT,

OrderDate DATETIME,

TotalAmount DECIMAL(10, 2),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

🡪Create OrderItems Table

CREATE TABLE OrderItems (

OrderItemID INT AUTO\_INCREMENT PRIMARY KEY,

OrderID INT,

ProductID INT,

Quantity INT,

UnitPrice DECIMAL(10, 2),

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

🡪Create Payments Table

CREATE TABLE Payments (

PaymentID INT AUTO\_INCREMENT PRIMARY KEY,

OrderID INT,

PaymentDate DATETIME,

PaymentMethod VARCHAR(50),

Amount DECIMAL(10, 2),

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)

);

**5. Sample Queries**

**5.1 Insert Sample Data**

🡪 Insert Sample Customers

INSERT INTO Customers (Name, Address, ContactInformation, Email, PasswordHash)

VALUES ('John Doe', '123 Elm Street', '555-1234', 'john@example.com', 'hashed\_password\_1');

🡪 Insert Sample Products

INSERT INTO Products (Name, Description, Price, StockQuantity)

VALUES ('Laptop', 'High-performance laptop', 1200.00, 50);

🡪 Insert Sample Orders

INSERT INTO Orders (CustomerID, OrderDate, TotalAmount)

VALUES (1, NOW(), 1200.00);

🡪 Insert Sample Order Items

INSERT INTO OrderItems (OrderID, ProductID, Quantity, UnitPrice)

VALUES (1, 1, 1, 1200.00);

🡪 Insert Sample Payment

INSERT INTO Payments (OrderID, PaymentDate, PaymentMethod, Amount)

VALUES (1, NOW(), 'Credit Card', 1200.00);

**5.2 Retrieve Data**

🡪 Retrieve all customers

SELECT \* FROM Customers;

🡪 Retrieve all orders for a specific customer

SELECT \* FROM Orders WHERE CustomerID = 1;

🡪 Retrieve all items in a specific order

SELECT \* FROM OrderItems WHERE OrderID = 1;

🡪 Retrieve payment details for a specific order

SELECT \* FROM Payments WHERE OrderID = 1;

**6. Additional Notes**

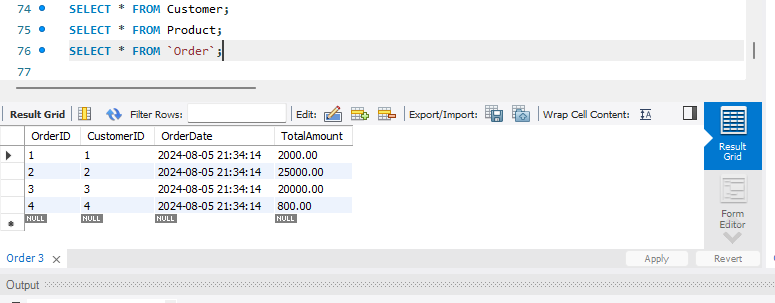
* **Data Integrity**: Foreign keys are used to enforce referential integrity between related tables.
* **Normalization**: The schema is designed to minimize redundancy and ensure efficient data storage.
* **Indexing**: Consider indexing frequently queried columns to improve performance.

**7. Conclusion**

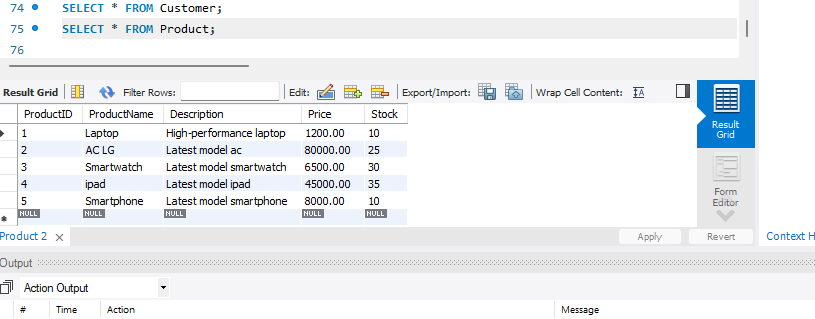
The Online Retail Application Database is designed to manage core functionalities of an online retail platform effectively. This documentation provides a comprehensive overview of the database schema, including tables, relationships, and sample queries.

**Output**

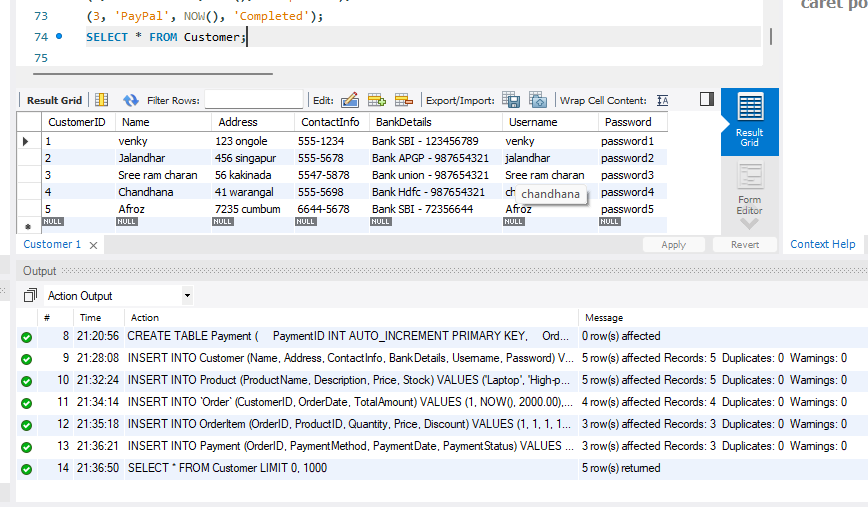
**Order Table**

****

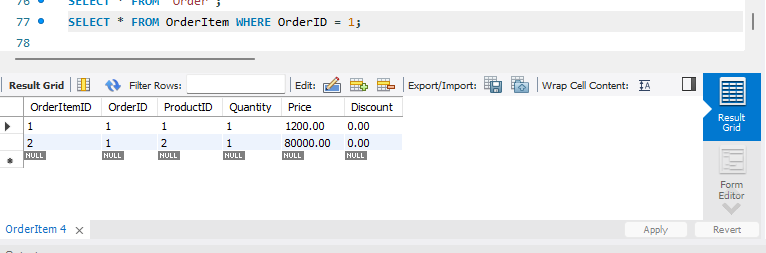
**Product Table**

****

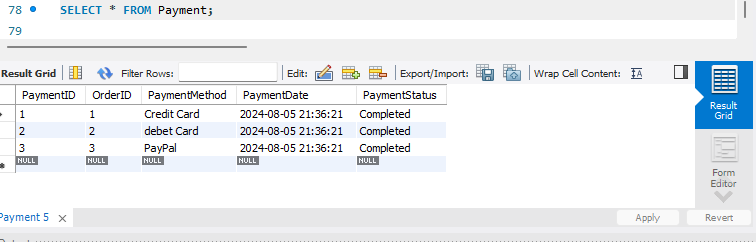
**Customer Table**

****

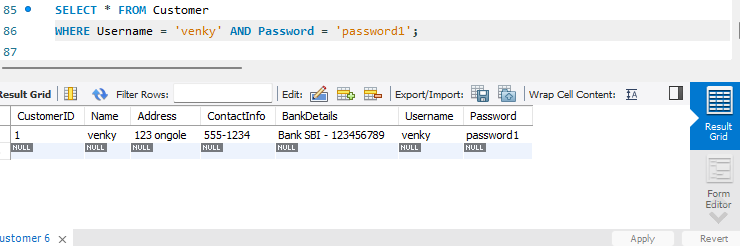
**Operations**

****

**Payment Table**

****

**Getting Customer details**

****