**QUESTION:**

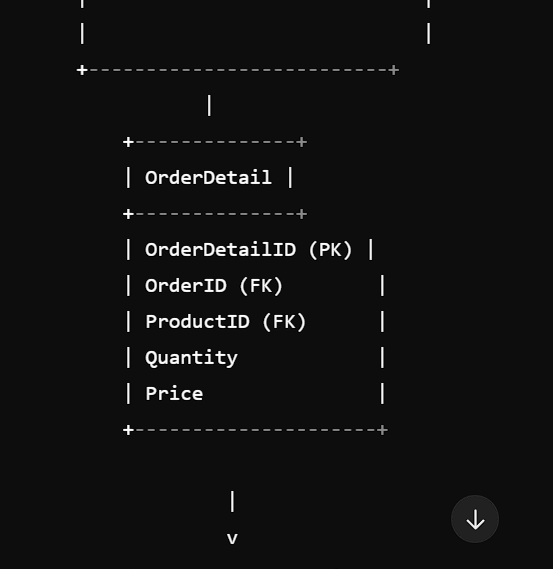
Develop a database for managing products, customers, orders, and payments in an e-commerce platform. - Model tables for products, customers, orders, and payments. - Write stored procedures for processing orders and handling payments. - Implement triggers to update product stock and order status. - Write SQL queries to analyze sales trends and customer demographics.

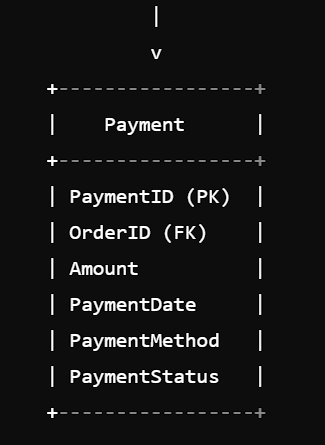
**ANSWER**:

**CONCEPTUAL ER DIAGRAM:**

**CONCEPTUAL ER DIAGRAM:**

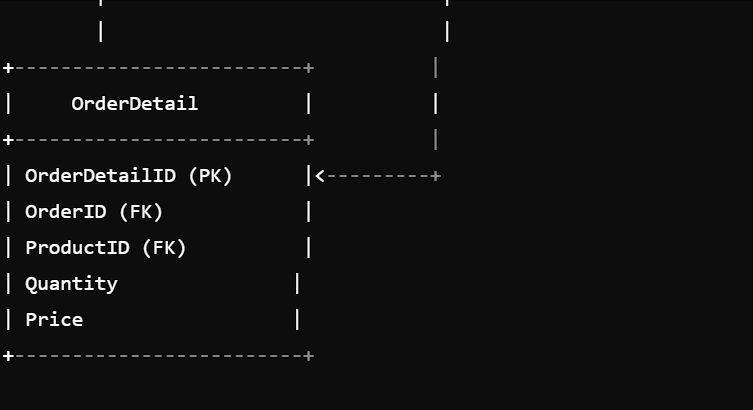


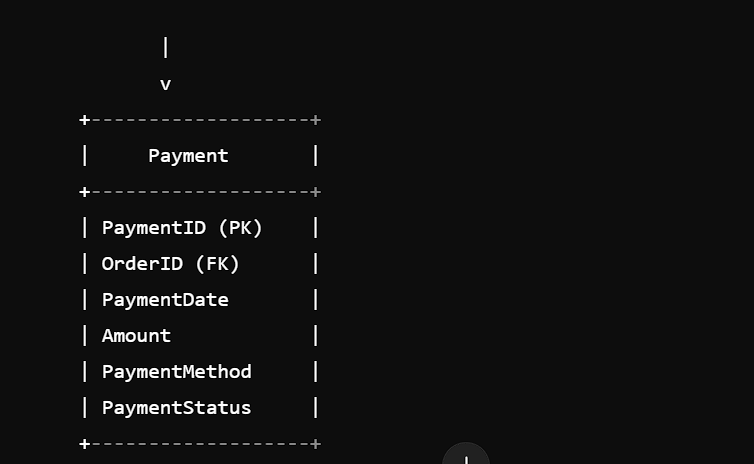




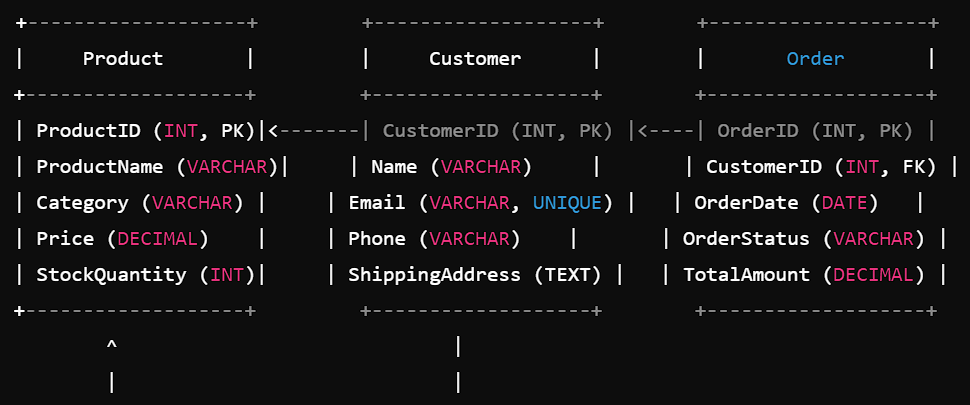
**LOGICAL ER DIAGRAM:**

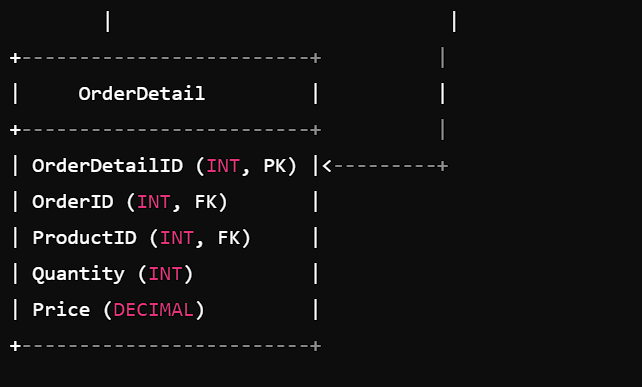


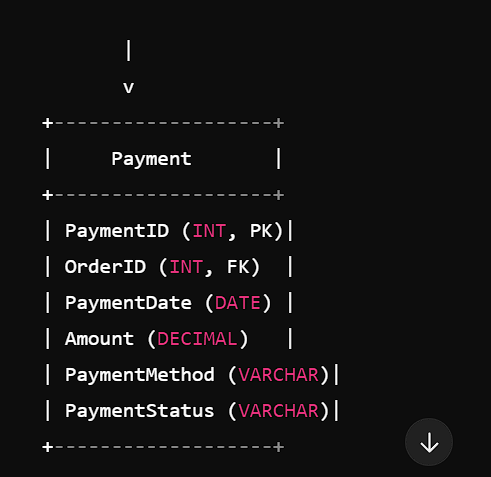




**PHYSICAL ER DIAGRAM**:







**MYSQL STATEMENTS:**

CREATE DATABASE ecommerce;

USE ecommerce;

CREATE TABLE Products (

ProductID INT AUTO\_INCREMENT,

ProductName VARCHAR (255) NOT NULL,

ProductDescription TEXT,

Price DECIMAL (10, 2) NOT NULL,

StockQuantity INT NOT NULL DEFAULT 0,

PRIMARY KEY (ProductID)

);

CREATE TABLE Customers (

CustomerID INT AUTO\_INCREMENT,

FirstName VARCHAR (50) NOT NULL,

LastName VARCHAR (50) NOT NULL,

Email VARCHAR (100) NOT NULL,

Phone VARCHAR (20),

Address TEXT,

PRIMARY KEY (CustomerID)

);

CREATE TABLE Orders (

OrderID INT AUTO\_INCREMENT,

CustomerID INT NOT NULL,

OrderDate DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

TotalCost DECIMAL (10, 2) NOT NULL,

OrderStatus VARCHAR (50) NOT NULL DEFAULT 'Pending',

PRIMARY KEY (OrderID),

FOREIGN KEY (CustomerID) REFERENCES Customers (CustomerID)

);

CREATE TABLE OrderItems (

OrderItemID INT AUTO\_INCREMENT,

OrderID INT NOT NULL,

ProductID INT NOT NULL,

Quantity INT NOT NULL,

PRIMARY KEY (OrderItemID),

FOREIGN KEY (OrderID) REFERENCES Orders (OrderID),

FOREIGN KEY (ProductID) REFERENCES Products (ProductID)

);

CREATE TABLE Payments (

PaymentID INT AUTO\_INCREMENT,

OrderID INT NOT NULL,

PaymentMethod VARCHAR (50) NOT NULL,

PaymentDate DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

Amount DECIMAL (10, 2) NOT NULL,

PRIMARY KEY (PaymentID),

FOREIGN KEY (OrderID) REFERENCES Orders (OrderID)

);

DELIMITER $$

CREATE PROCEDURE ProcessOrder (

IN \_OrderID INT,

IN \_CustomerID INT,

IN \_TotalCost DECIMAL (10, 2)

)

BEGIN

INSERT INTO Orders (CustomerID, TotalCost)

VALUES (\_CustomerID, \_TotalCost);

SET @\_OrderID = LAST\_INSERT\_ID ();

UPDATE Products

SET StockQuantity = StockQuantity - 1

WHERE ProductID IN (SELECT ProductID FROM OrderItems WHERE OrderID = @\_OrderID);

END$$

CREATE PROCEDURE HandlePayment (

IN \_OrderID INT,

IN \_PaymentMethod VARCHAR (50),

IN \_Amount DECIMAL (10, 2)

)

BEGIN

INSERT INTO Payments (OrderID, PaymentMethod, Amount)

VALUES (\_OrderID, \_PaymentMethod, \_Amount);

UPDATE Orders

SET OrderStatus = 'Paid'

WHERE OrderID = \_OrderID;

END$$

DELIMITER:

CREATE TRIGGER UpdateProductStock

AFTER INSERT ON OrderItems

FOR EACH ROW

BEGIN

UPDATE Products

SET StockQuantity = StockQuantity - NEW.Quantity

WHERE ProductID = NEW.ProductID;

END$$

CREATE TRIGGER UpdateOrderStatus

AFTER UPDATE ON Payments

FOR EACH ROW

BEGIN

UPDATE Orders

SET OrderStatus = 'Paid'

WHERE OrderID = NEW.OrderID;

END$$

SELECT

O OrderID,

C CustomerName

P ProductName

Oi Quantity,

O TotalCost

FROM

Orders o

JOIN Customers c ON o CustomerID = c CustomerID

JOIN OrderItems oi ON o OrderID = oi OrderID

JOIN Products p ON oi ProductID = p ProductID;

SELECT

c.CustomerName,

COUNT (o OrderID) AS OrderCount,

SUM (o TotalCost) AS TotalSpent

FROM

Orders o

JOIN Customers c ON o CustomerID = c CustomerID

GROUP BY

c.CustomerName;

SELECT

P ProductName,

SUM (oi Quantity) AS TotalSold,

SUM (o TotalCost) AS TotalRevenue

FROM

Orders o

JOIN OrderItems oi ON o OrderID = oi OrderID

JOIN Products p ON oi ProductID = p ProductID

GROUP BY

P ProductName;

**Conclusion:**

The database design for the e-commerce platform has been successfully implemented. The design includes tables for products, customers, orders, and payments. Stored procedures have been created to process orders and handle payments. Triggers have been implemented to update product stock and order status. SQL queries have been written to analyze sales trends and customer demographics.