**CIS 411 Database System II**

**Spring 2025**

**Project Title: QuickMart: Retail Store Management Database**

**Group Members: La’Ron Hampton, Yash Patel, Gracy Patel, Amaree Ryans, Micah Shepherd**

**Instructor: Qi, Guanqui**

**Submission Date: May 11, 2025**

**1. Introduction**

QuickMart is a retail convenience store that serves the local community with various products, including groceries, beverages, snacks, and essential household goods. As the business has grown, QuickMart faces challenges in managing its daily operations, especially around customer orders, inventory control, and employee task management. The current system is manual, paper-based, and lacks real-time integration.

The goal of this project is to design and implement a robust relational database management system that digitizes the core operations of QuickMart. The database will facilitate faster decision-making, reduce human errors, and support scalable store expansion by ensuring accurate, timely data.

**2. Purpose of the System**

The QuickMart Retail Store Management Database is developed to centralize the store’s data. The primary purposes include:

* Maintain accurate customer records and their purchase histories
* Manage and monitor product inventory in real time
* Track supplier relationships and product sourcing
* Record and manage employee tasks and their involvement in orders
* Generate business-critical reports for analytics and restocking
* Improve operational efficiency through streamlined data entry forms and navigation

This system empowers store staff to perform their duties efficiently while allowing management to make data-driven decisions.

**3. Main Entities and Relationships**

Entities:

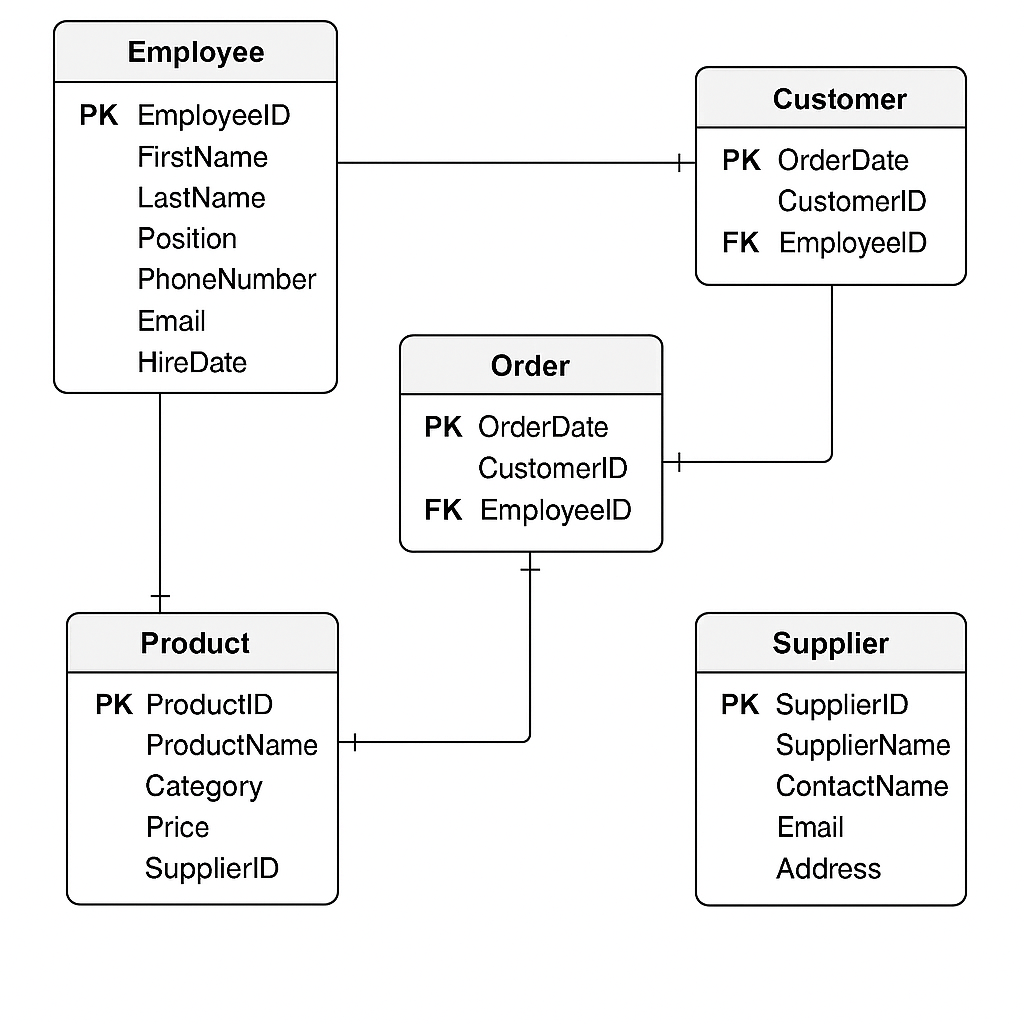
* Customers: Individuals who purchase products from QuickMart
* Orders: Transactions initiated by customers
* OrderItems: Items listed within a single order
* Products: Goods available for sale
* Suppliers: Companies providing products
* Employees: Staff members handling store operations

Relationships:

* A Customer can place many Orders
* Each Order can contain multiple OrderItems
* Each OrderItem references one Product
* A Product is supplied by one Supplier
* An Employee may handle many Orders

These relationships were implemented using foreign keys and one-to-many/many-to-one cardinalities.

**4. Entity Relationship Diagram**



The ER diagram illustrates the relationships between six key entities:

* A Customer places one or more Orders
* Each Order consists of multiple OrderItems
* Each OrderItem is linked to one Product
* A Product is supplied by one Supplier
* An Employee manages one or more Orders

**Functional Dependencies**

* Customers: CustomerID → Name, Email, Phone
* Employees: EmployeeID → Name, Role, Phone
* Suppliers: SupplierID → Name, Phone
* Products: ProductID → Name, Price, Stock, SupplierID
* Orders: OrderID → CustomerID, EmployeeID, OrderDate
* OrderItems:
  + OrderItemID → OrderID, ProductID, Quantity, Price
  + (OrderID, ProductID) → Quantity, Price

**Normalized Relations (3NF)**

CUSTOMERS  
CustomerID (PK), Name, Email, Phone

EMPLOYEES  
EmployeeID (PK), Name, Role, Phone

SUPPLIERS  
SupplierID (PK), Name, Phone

PRODUCTS  
ProductID (PK), Name, Price, Stock, SupplierID (FK)

ORDERS  
OrderID (PK), CustomerID (FK), EmployeeID (FK), OrderDate

ORDERITEMS  
OrderItemID (PK), OrderID (FK), ProductID (FK), Quantity, Price

**Why Normalization Matters**

Normalization helps minimize redundancy and improve data integrity. By organizing data into 3NF, the QuickMart system ensures efficient querying and updating of data without anomalies or inconsistencies.

**Denormalization**

No denormalization was needed, as all entities and their relationships were cleanly represented in normalized form.

**SQL Data Definition Language (DDL)**

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(100),

Email VARCHAR(100),

Phone VARCHAR(15)

);

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

Name VARCHAR(100),

Role VARCHAR(50),

Phone VARCHAR(15)

);

CREATE TABLE Suppliers (

SupplierID INT PRIMARY KEY,

Name VARCHAR(100),

Phone VARCHAR(15)

);

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

Name VARCHAR(100),

Price DECIMAL(10,2),

Stock INT,

SupplierID INT,

FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID)

);

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

EmployeeID INT,

OrderDate DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),

FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID)

);

CREATE TABLE OrderItems (

OrderItemID INT PRIMARY KEY,

OrderID INT,

ProductID INT,

Quantity INT,

Price DECIMAL(10,2),

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

**Queries and SQL Usage**

* 1. **Query 1 (2 tables): List all orders with customer name**

SELECT Orders.OrderID, Customers.Name FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

* 1. **Query 2 (3 tables):** Show order details including product info

SELECT Orders.OrderID, Products.Name, OrderItems.Quantity FROM Orders

JOIN OrderItems ON Orders.OrderID = OrderItems.OrderID

JOIN Products ON OrderItems.ProductID = Products.ProductID;

* 1. **Query 3 (4 tables):** List employee, order, customer, and product info

SELECT Employees.Name AS Employee, Customers.Name AS Customer, Products.Name AS Product

FROM Employees

JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID

JOIN Customers ON Orders.CustomerID = Customers.CustomerID

JOIN OrderItems ON Orders.OrderID = OrderItems.OrderID

JOIN Products ON OrderItems.ProductID = Products.ProductID;

**Forms**

* Customer Entry Form
* Order Entry Form with subform for OrderItems
* Product Form with Supplier lookup

**Reports**

* Sales by Date Report
* Inventory Status Report

**Navigation Form**

A screenshot of a data

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Conclusion**

The QuickMart system fulfills the goals of managing a small-to-medium retail store's core operations efficiently using a relational database approach. All forms, reports, and queries function as intended, supporting business decisions with structured and normalized data

Each team member contributed to the successful development of the QuickMart database system. Here's a breakdown of responsibilities:

* **Gracy Patel**: Interface design, database creation in Access, helped finalize forms and navigation forms, documentation lead
* **La’Ron Hampton**: Led Access database development, SQL scripting, query integration, and co-developed forms with Gracy
* **Yash Patel**: Developed and tested forms, implemented and validated queries,
* **Amaree Ryans**: Conducted application testing, gathered screenshots, and supported report formatting
* **Micah Shepherd**: Report design, grammar checking, structure formatting, worked on presentation layout