SQL PROJECT

Grocery Store Management
Domain Knowledge
The Retail and Grocery domain involves the management of inventory, suppliers, customer orders, employees, and product categories. Effective data management in this domain allows businesses to:
Track sales and revenue

Monitor product availability

Analyze customer and employee activity

Improve operational efficiency
This project simulates a mini grocery store database where various entities such as products, suppliers, customers, and orders interact. Students will use SQL to perform data extraction, transformations, and derive business insights.

Data Description
Ø=ÜÌ Kindly click on the Dataset to check and download
This project includes 7 interrelated tables:
Table Name
Description
supplier
Stores information about the suppliers who provide products.
categories
Contains the different product categories (e.g., Beverages, Snacks).
employees
Records employees responsible for handling customer orders.
customers
Contains customer details who place orders.
products
Holds product information, including price, supplier, and category.
orders
Logs customer orders handled by employees.
j
Stores line items for each order, including product quantity and pricing.

Table Structures

```
-- 1. Supplier Table
CREATE TABLE IF NOT EXISTS supplier (
sup_id TINYINT PRIMARY KEY,
sup_name VARCHAR(255),
address TEXT
);

-- 2. Categories Table
CREATE TABLE IF NOT EXISTS categories (
cat_id TINYINT PRIMARY KEY,
cat_name VARCHAR(255)
);

-- 3. Employees Table
CREATE TABLE IF NOT EXISTS employees (
emp_id TINYINT PRIMARY KEY,
emp_name VARCHAR(255),
hire_date VARCHAR(255)
);

-- 4. Customers Table
CREATE TABLE IF NOT EXISTS customers (
cust_id SMALLINT PRIMARY KEY,
cust_name VARCHAR(255),
address TEXT
);

-- 5. Products Table
CREATE TABLE IF NOT EXISTS products (
prod_id TINYINT PRIMARY KEY,
prod_name VARCHAR(255),
sup_id TINYINT,
cat_id TINYINT,
price DECIMAL(10,2),
FOREIGN KEY (supid) REFERENCES supplier(supid)
ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY (catid) REFERENCES categories(catid)
ON UPDATE CASCADE ON DELETE CASCADE
```

```
);

-- 6. Orders Table
CREATE TABLE IF NOT EXISTS orders (
ord_id SMALLINT PRIMARY KEY,
cust_id SMALLINT,
emp_id TINYINT,
order_date VARCHAR(255),
FOREIGN KEY (custid) REFERENCES customers(custid)
ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY (empid) REFERENCES employees(empid)
ON UPDATE CASCADE ON DELETE CASCADE
);

-- 7. Order_Details Table
CREATE TABLE IF NOT EXISTS order_details (
orddetID SMALLINT AUTOINCREMENT PRIMARY KEY,
ord_id SMALLINT,
prod_id TINYINT,
quantity TINYINT,
each_price DECIMAL(10,2),
total_price DECIMAL(10,2),
FOREIGN KEY (ordid) REFERENCES orders(ordid)
ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY (prodid) REFERENCES products(prodid)
ON UPDATE CASCADE ON DELETE CASCADE
);
```

Use the above Query to create the entire schema of Grocery Store management
ER Diagram

One-to-Many from supplier to products

One-to-Many from categories to products

One-to-Many from products to order_details

One-to-Many from orders to order_details

One-to-Many from customers to orders

One-to-Many from employees to orders
Key things to consider while creating database:
Add auto increment constraint in all the table’s primary keys.
Make sure you are using cascade and while creating foreign keys in all the tables.
While importing the data carefully check whether the column names are matching to csv file column names or not.
If the data is too large, follow the steps in the document Click Here
Objectives
The main goals of this SQL project are:
To design and implement a relational database for a grocery store.

To retrieve and manipulate data using SQL queries.

To perform data analysis for business insights such as top customers, best-selling products, and revenue trends.

To practice using joins, aggregations, subqueries, and filtering techniques.

Analysis Questions
1þ Customer Insights
Gain an understanding of customer engagement and purchasing behavior.
How many unique customers have placed orders?

Which customers have placed the highest number of orders?

What is the total and average purchase value per customer?

Who are the top 5 customers by total purchase amount?
2. Product Performance

Evaluate how well products are performing in terms of sales and revenue.
How many products exist in each category?

What is the average price of products by category?

Which products have the highest total sales volume (by quantity)?

What is the total revenue generated by each product?

How do product sales vary by category and supplier?
3. Sales and Order Trends
Analyze business performance through orders and revenue over time.
How many orders have been placed in total?

What is the average value per order?

On which dates were the most orders placed?

What are the monthly trends in order volume and revenue?

How do order patterns vary across weekdays and weekends?
4þ Supplier Contribution
Identify the most active and profitable suppliers.
How many suppliers are there in the database?

Which supplier provides the most products?

What is the average price of products from each supplier?

Which suppliers contribute the most to total product sales (by revenue)?
5þ Employee Performance

Assess how employees are handling and influencing sales.
How many employees have processed orders?

Which employees have handled the most orders?

What is the total sales value processed by each employee?

What is the average order value handled per employee?
6þ Order Details Deep Dive
Explore item-level sales patterns and pricing behavior.
What is the relationship between quantity ordered and total price?

What is the average quantity ordered per product?

How does the unit price vary across products and orders?
Challenges You Might Face
Understanding table relationships and applying correct joins.
Ensuring data consistency with foreign key constraints.
Handling aggregation across joined tables.
Extracting time-based trends from date data (especially if in VARCHAR format).