**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**

This project includes 7 interrelated tables:

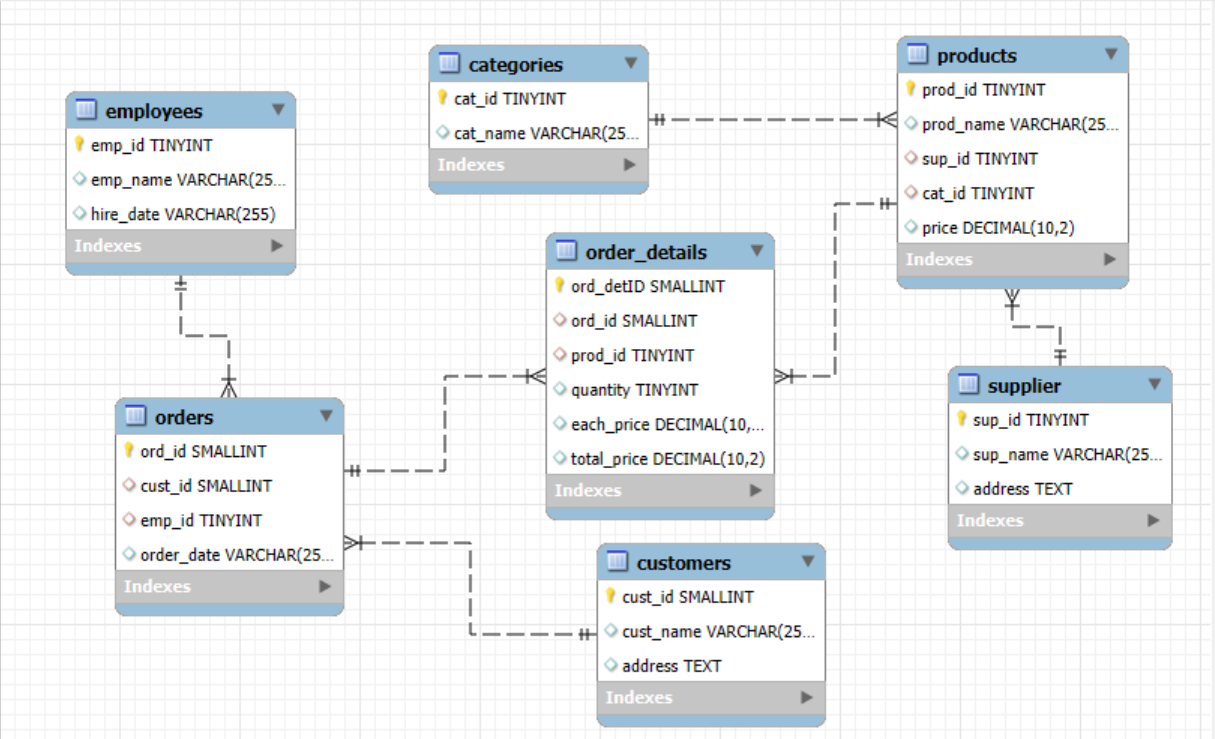
|  |  |
| --- | --- |
| **Table Name** | **Description** |
| **supplier** | Stores information about the suppliers who provide products. |
| **categories** | Contains the different product categories (e.g., Beverages, Snacks). |
| **employees** | Records employees responsible for handling customer orders. |
| **customers** | Contains customer details who place orders. |
| **products** | Holds product information, including price, supplier, and category. |
| **orders** | Logs customer orders handled by employees. |
| **order\_details** | Stores line items for each order, including product quantity and pricing. |

**Table Structure**

|  |
| --- |
| -- 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 (sup\_id) REFERENCES supplier(sup\_id)         ON UPDATE CASCADE ON DELETE CASCADE,     FOREIGN KEY (cat\_id) REFERENCES categories(cat\_id)         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 (cust\_id) REFERENCES customers(cust\_id)         ON UPDATE CASCADE ON DELETE CASCADE,     FOREIGN KEY (emp\_id) REFERENCES employees(emp\_id)         ON UPDATE CASCADE ON DELETE CASCADE );  -- 7. Order\_Details Table CREATE TABLE IF NOT EXISTS order\_details (     ord\_detID SMALLINT AUTO\_INCREMENT PRIMARY KEY,     ord\_id SMALLINT,     prod\_id TINYINT,     quantity TINYINT,     each\_price DECIMAL(10,2),     total\_price DECIMAL(10,2),     FOREIGN KEY (ord\_id) REFERENCES orders(ord\_id)         ON UPDATE CASCADE ON DELETE CASCADE,     FOREIGN KEY (prod\_id) REFERENCES products(prod\_id)         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**](https://docs.google.com/document/d/1im9uM7al21vucJuHiH1QZUA8IAg_5zyxdaxBcNPbD-Q/edit?usp=sharing)

**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**

**1️. 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 months?

**4️. 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)?

**5️. 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?

**6️. 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).