

PROJECT FOR SQL MODULE

MILK DAIRY MANAGEMENT SYSTEM

Institute Name:- Itvedant Education Pvt .Ltd

Name:- HARSHAL SANDIP SHIRSEKAR

Email Address:- [harryshirsekar@gmail.com](mailto:harryshirsekar@gmail.com)

Date of submission:-26 May 2024

# Project Aims:

# Database Design and Schema Creation:

# Create an optimized and normalized database schema that accurately represents the relationships between different entities such as Dairy Products, Factory Workers, distributors, and Customers.

# Data Population and Integrity

Populate the database with sample data to simulate real-world scenarios in a Milk Dairy Management.

Ensure data integrity through the use of constraints, such as primary keys, foreign keys, and unique constraints.

# User Authentication and Authorization:

Implement a user authentication system to secure the database, allowing only authorized users to access and modify data.

# Define and enforce role-based access control to restrict users' actions based on their roles (e.g., Factory Team Leader, Administrator).

# Transaction Management:

Implement a transaction management system to handle the distribution and product requirement tracking, etc...

Track transaction details, including dates, amount of distribution customers, and requirements of distribution customers require in the future.

# Customer Requirement Tracking

Develop a mechanism for customers requiring issues on real real-time basis. (e.g., If customers require them to increase the amount of distribution they currently have daily, they can upgrade the amount of distribution as they need.)

# Customer Information Management:

Create a system for managing customer information, including details such as names, contact information, and requirements of product types.

Implement features for adding, updating, and deactivating member records.

# Reporting and Analysis:

# Develop SQL queries for generating various reports, such as a list of currently distributed product details, pending amounts of customers, most products buy customers, etc

# Implement analysis queries to gather insights into library usage patterns.

# User-Friendly Interface:

# Consider the development of a user-friendly interface or integrate the database with an existing milk dairy management system for seamless interaction.

# Scalability and Performance:

# Design the database with scalability in mind to accommodate potential future expansions of the Milk Dairy collection and user base.

# Optimize SQL queries and database indexing for improved performance.

# By achieving these aims, the project aims to create a well-organized, secure, and efficient database system that enhances the overall management of the library's resources and services.

# Project Objective:

# The objective of this SQL project is to design and implement a comprehensive database system for a milk dairy management system. The primary focus is on creating a robust and efficient database schema to manage various aspects of dairy management operations, including product storage, customer information, worker information, customer requirements, and tracking monthly distribution. The project aims to enhance the overall efficiency and organization of the dairy by providing a centralized and well-structured database.

# ER diagram:

# 

# Each rectangle represents an entity. Lines between the entities represent the relationships between them. The lines between the entities represent primary keys and foreign keys connecting the tables. The ER diagram showcases the relationships between the different tables, such as vehicle\_details, accident\_details, road\_details, insurance\_details, contractor\_details as described in the project description.

# Table description:

# 1: Product Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| Product\_Id | int | NO | PRI | NULL | auto\_increment |
| product\_name | varchar(100) | YES |  | NULL |  |
| product\_type | varchar(55) | YES |  | NULL |  |
| product\_price | decimal(4,2) | YES |  | NULL |  |

# 2: Worker Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| Worker\_id | int | NO | PRI | NULL | auto\_increment |
| FirstName | varchar(55) | YES |  | NULL |  |
| middlename | varchar(55) | YES |  | NULL |  |
| lastname | varchar(55) | YES |  | NULL |  |
| worker\_post | varchar(15) | YES |  | NULL |  |
| salary | float | NO |  | NULL |  |

# 3: Customer Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| Cust\_id | int | NO | PRI | NULL | auto\_increment |
| cust\_fname | varchar(55) | YES |  | NULL |  |
| cust\_mname | varchar(55) | YES |  | NULL |  |
| cust\_lname | varchar(55) | YES |  | NULL |  |
| Cust\_addline1 | varchar(55) | YES |  | NULL |  |
| Cust\_addline2 | varchar(55) | YES |  | NULL |  |
| Cust\_addline3 | varchar(55) | YES |  | NULL |  |
| Cust\_division | varchar(55) | YES |  | NULL |  |
| Cust\_order | int | NO |  | NULL |  |
| product\_id | int | YES | MUL | NULL |  |
| Distributor\_id | int | YES | MUL | NULL |  |

# Commands:

# 1: Create Database-

# CREATE DATABASE DAIRY;

# 2: Using or Entering Database -

# USE DAIRY;

# 3: Creating Tables -

# CREATE TABLE PRODUCTS(

# PRODUCT\_ID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

# PRODUCT\_NAME VARCHAR(100),

# PRODUCT\_TYPE VARCHAR (55),

# PRODUCT\_PRICE DECIMAL (4,2));

# 

# CREATE TABLE WORKER(

# WORKER\_ID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

# FIRSTNAME VARCHAR (55),

# MIDDLENAME VARCHAR (55),

# LASTNAME VARCHAR (55),

# WORKER\_POST VARCHAR(15),

# SALARY FLOAT NOT NULL);

# CREATE TABLE CUSTOMER(

# CUST\_ID INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

# CUST\_FNAME VARCHAR (55),

# CUST\_MNAME VARCHAR (55),

# CUST\_LNAME VARCHAR (55),

# CUST\_ADDLINE1 VARCHAR (55),

# CUST\_ADDLINE2 VARCHAR (55),

# CUST\_ADDLINE3 VARCHAR (55),

# CUST\_DIVISION VARCHAR (55),

# CUST\_ORDER INT NOT NULL,

# PRODUCT\_ID INT,

# DISTRIBUTOR\_ID INT,

# FOREIGN KEY (PRODUCT\_ID) REFERENCES PRODUCTS (PRODUCT\_ID),

# FOREIGN KEY (DISTRIBUTOR\_ID) REFERENCES WORKER (WORKER\_ID));

# 4: Show Tables & Database-

# SHOW DATABASES;

# SHOW TABLES;

# 5: INSERTING DATA –

# Product Table Entry:

# INSERT INTO Products (Product\_Id, product\_name, product\_type, product\_price) VALUES

# (001, "Cow Milk","1 Liter", 25),

# (002, "Buffalo Milk","1 Liter", 35);

# Worker Table Entry:

# INSERT INTO WORKER (WORKER\_ID, FIRSTNAME, MIDDLENAME, LASTNAME, WORKER\_POST, SALARY) VALUES

# (1001,"JAYRAM","PRAKASH","YADAV","MANAGER", 25000),

# (1002,"RAJA","RAOJI","YADAV","HELPER", 20000),

# (1003,"PRAKASH","SADASHIV","MORE","DISTRIBUTOR", 20000),

# (1004,"KIRAN","SADASHIV","MORE","DISTRIBUTOR", 20000);

# Customer Table Entry:

# INSERT INTO CUSTOMER (CUST\_ID, CUST\_FNAME, CUST\_MNAME, CUST\_LNAME, CUST\_ADDLINE1, CUST\_ADDLINE2, CUST\_ADDLINE3, CUST\_DIVISION, CUST\_ORDER, PRODUCT\_ID, DISTRIBUTOR\_ID) VALUES

# (1,"GANGU"," ","RANE","ROOM NO 01","GANESH APT B WING","DEVENDRA INDUSTRY","SECTOR 1","2", 001,1003),

# (2,"RAGHU","G","MORE","ROOM NO 02","GANESH APT B WING","DEVENDRA INDUSTRY","SECTOR 1","1", 002,1003),

# (3,"JAYA","H","GHARE","ROOM NO 12","GANESH APT A WING","DEVENDRA INDUSTRY","SECTOR 1","3", 001, 1003),

# (4,"MANU","M","GHOLE","ROOM NO 03","GANESH APT A WING","DEVENDRA INDUSTRY","SECTOR 1","3", 001, 1003),

# (5,"RAJU","J","R","ROOM NO 01","SAI DARSHAN APT A WING","SURESH MEDICAL","SECTOR 3","2", 002, 1004),

# (6,"JYOTI","S","JAGDALE","ROOM NO 02","SAI DARSHAN APT B WING","SURESH MEDICAL","SECTOR 3","2", 001,1004),

# (7,"KIRAN","J","MHATRE","ROOM NO 14","SAI DARSHAN APT B WING","SURESH MEDICAL","SECTOR 3","2", 001,1004),

# (8,"BAJI","SHAMBHU","BHOSALE","ROOM NO 22","SAI DARSHAN APT B WING","SURESH MEDICAL","SECTOR 3","3", 001,1004);

# 6: DELETE TABLE –

# DROP TABLE PRODUCT;

# DROP TABLE WORKER;

# DROP TABLE CUSTOMER;

# 7: TRUNCATE TABLE –

# TRUNCATE PRODUCT;

# TRUNCATE CUSTOMER;

# TRUNCATE WORKER;

# 8: ALTER QUERY –

# Add city in worker table.

# ALTER TABLE WORKER ADD COLUMN CITY;

# Remove city from worker table

# ALTER TABLE WORKER DROP COLUMN CITY;

# Rename the column name “Worker\_post to Post”.

# ALTER TABLE WORKER CHANGE COLUMN WORKER\_POST POST VARCHAR (55);

# Add column after Addline3 the pincode column in customer table.

# ALTER TABLE CUSTOMER ADD COLUMN AFTER CUST\_ADDLINE3 PINCODE INT;

# Restrict only 6 digit number for Pincode.

# ALTER TABLE CUSTOMER ADD CONSTRAINT RISTRICT\_PINCODE\_CONSTRAINT CHECK (PINCODE >= 100000 AND PINCODE <=999999);

# 9: UPDATE QUERY –

# Update salary 22000 for worker id 1002.

# UPDATE WORKER SET SALARY = 22000 WHERE WORKER\_ID = 1002;

# 10: DELETE QUERY –

# Delete customer details where order is 1.

# DELETE FROM CUSTOMER WHERE CUST\_ORDER = 1;

# Delete all customer details where order is greater than 2.

# DELETE FROM CUSTOMER WHERE CUST\_ORDER >2;

# 10: TABLE DESCRIPTION –

# DESC WORKER;

# 11: SELECT QUERY –

# SELECT \* FROM PRODUCTS;

# SELECT \* FROM WORKER;

# SELECT \* FROM CUSTOMER;

# 12: SELECT QUERY WITH CONDITION –

# Details of all worker working as distributor.

# SELECT \* FROM WORKER WHERE POST = "DISTRIBUTOR";

# Details of all customer who living in sector 3 and ordering cow milk.

# SELECT \* FROM CUSTOMER WHERE CUST\_DIVISION = "SECTOR 3" AND PRODUCT\_ID = 1;

# Details of all customer who living in sector 3 and ordering Buffalo milk.

# SELECT \* FROM CUSTOMER WHERE CUST\_DIVISION = "SECTOR 3" AND PRODUCT\_ID = 2;

# 13: SUB QUERY –

# Second highest salary from worker.SELECT MAX(SALARY) AS SECONDHIGHESTSALARY FROM WORKER WHERE SALARY < (SELECT MAX(SALARY) FROM WORKER);

# Second minimum salary from worker.

# SELECT MIN(SALARY) AS SECONDMINSALARY FROM WORKER WHERE SALARY > (SELECT MIN(SALARY) FROM WORKER);

# Second minimum order from customer.

# SELECT MIN(CUST\_ORDER) FROM CUSTOMER WHERE CUST\_ORDER > (SELECT MIN(CUST\_ORDER) FROM CUSTOMER);

# 14: JOINT QUERY –

# LEFT JOIN

# SELECT WORKER.WORKER\_ID, CUSTOMER.DISTRIBUTOR\_ID FROM WORKER LEFT JOIN CUSTOMER ON WORKER.WORKER\_ID = CUSTOMER.DISTRIBUTOR\_ID;

# RIGHT JOIN

# SELECT WORKER.WORKER\_ID, CUSTOMER.DISTRIBUTOR\_ID FROM WORKER RIGHT JOIN CUSTOMER ON WORKER.WORKER\_ID = CUSTOMER.DISTRIBUTOR\_ID;

# UNION ALL

# SELECT PRODUCTS.PRODUCT\_ID AS PRD\_ID, CUSTOMER.PRODUCT\_ID FROM PRODUCTS RIGHT JOIN CUSTOMER ON PRODUCTS.PRODUCT\_ID = CUSTOMER.PRODUCT\_ID;

# Conclusion:

The “**Milk Management System”** developed using SQL offers a robust solution for efficiently managing various aspects of worker details, distribution, and sales. By leveraging the power of SQL databases, the system ensures data integrity, scalability, and ease of access for users at different levels within the organization.

**The system encompasses several key features, including:**

* **Worker Management:**

Recording worker names, post and salary details ect.

* **Supplier Management :**

Recording supplier details, delivery schedules, and maintaining a database of reliable suppliers.

* **Customer Management :**

Managing customer accounts, orders, and delivery preferences to ensure timely and accurate deliveries.

* **Sales and Billing Management :**

Generating invoices, tracking sales transactions, and managing billing information for both wholesale and retail customers.

By utilizing SQL queries and database operations, the system efficiently handles complex data manipulations, joins, and aggregations required for seamless operations. The use of indexes, constraints, and normalization techniques ensures optimal database performance and data consistency.

In summary, the milk management system implemented in SQL not only streamlines day-to-day operations but also lays the foundation for innovation and growth in the dairy industry, empowering businesses to thrive in a competitive market landscape.

