Fitness Studio Database

**Group Members**

Arushi Vyas, Hamidullah Sakhi, Jannik Haas, Raymond Magambo, Ruyun Chen

# Problem Description

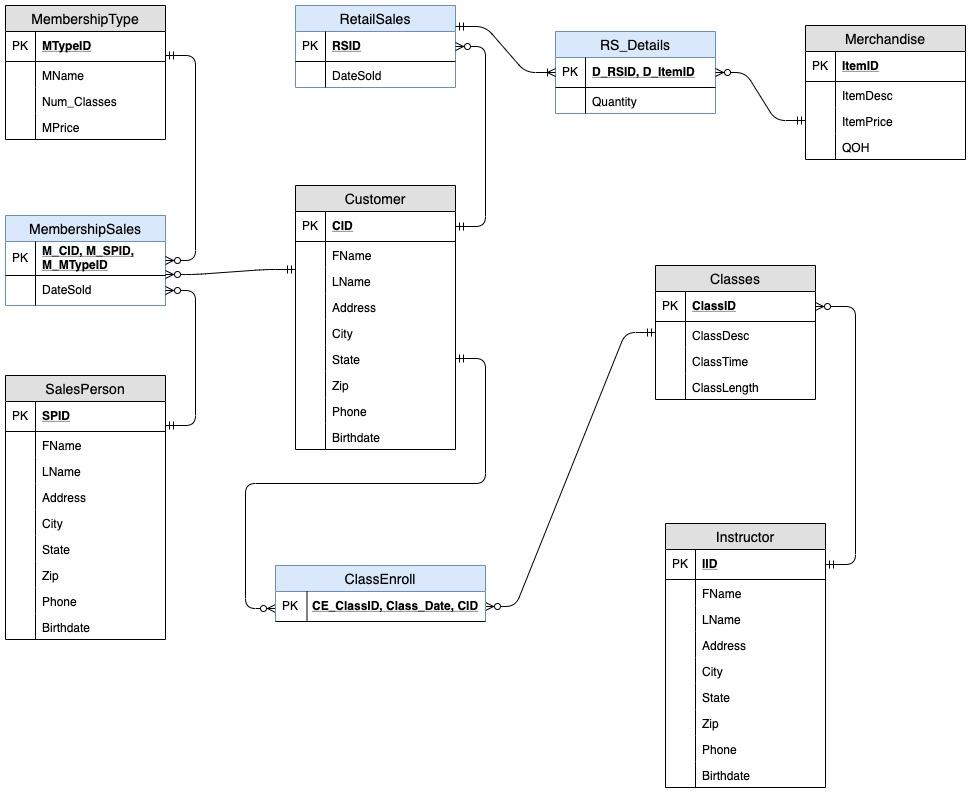
Our goal is to design a database for our fitness studio to keep a better overview of our accounts and memberships. The fitness studio offers a variety of daily classes and offers different membership options with varying number of classes that members are allowed to take. The studio also sells retail merchandise such as T-shirts, water bottles, sweatshirts, etc. We will also need to keep track of our employees which include salespeople and instructors.

What our database will include and generate reports on:

* **Revenue:** The income that our business takes in from normal operations and activities. We will be able to track and compare the different sources of income such as merchandise and memberships. This will also help us be able to understand what items sell the best and be able to focus on advertising these items.
* **Class enrollment:** Our database will also track class enrollment so we can see which members are signed up for which classes. This will also give us a better understanding of what type of class members enjoy and adjust our schedule to accommodate customers' needs.
* **Membership sales:** The Fitness studio offers various membership options with varying numbers of classes that customers are allowed to take per month. Tracking how many customers hold each type of membership will give us a better understanding of our customers’ needs.
* **Retail sales:** Tracking retails sales allows us to order more merchandise when we are running low as well as seeing what type of items sell best and creating more similar items.
* **Employee performance:** The database will keep track of the number of memberships sold by each employee which will give the manager concrete statistics to reward outstanding employees and to focus on improving the performance of underperforming employees.

# 

# Conceptual Model

****

# Third Normal Form Relations

**Customer (CID,** FName, LName, Birthdate, Address, City, State, Zip, Phon**e)**

**MembershipSales (M\_CID, M\_SPID, M\_MTypeID,**Date\_Sold**)**

**SalesPerson (SPID,** FName, LName, Address, City, State, Zip, Phone, Birthdate**)**

**MembershipType (MTypeID,** Num\_Classes, MPrice, MName**)**

**RetailSales (RSID,** *RS\_CID***,** DateSold**)**

**RS\_Details (D\_RSID, D\_ItemID,** Quantity**)**

**Merchandise (ItemID,** ItemDesc, ItemPrice, QOH**)**

**ClassEnroll (CE\_ClassID, Class\_Date,*CE\_CID*)**

**Classes (ClassID,** ClassDesc, ClassLength, ClassTime,*IID***)**

**Instructor (IID,** FName, LName, Address, City, State, Zip, Phone, Birthdate**)**

# Data Dictionary

**Customer Table:** Records general customer information for mailing, billing and general record keeping.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description (PK/FK)** |
| CID | CHAR (3) | NOT NULL | Unique customer number used as a primary key (PK) |
| FName | VARCHAR (10) | NOT NULL | Customer’s First name |
| LName | VARCHAR (10) | NOT NULL | Customer’s Last name |
| Address | VARCHAR (20) |  | Customer address |
| City | VARCHAR (15) |  | Customer city |
| State | CHAR (2) |  | Customer state |
| Zip | CHAR (5) |  | Customer zip |
| Phone | CHAR (10) |  | Customer phone |
| Birthdate | DATE | NOT NULL | Customer birthdate |

**SalesPerson Table:** Records information on front desk/salespeople staff who are responsible for selling memberships and merchandise. Used for tax purposes as well as general record keeping

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| SPID | CHAR (3) | NOT NULL | Unique salesperson number used as a primary key (PK) |
| FName | VARCHAR (10) | NOT NULL | Salesperson first name |
| LName | VARCHAR (10) | NOT NULL | Salesperson last name |
| Address | VARCHAR (20) |  | Salesperson address |
| City | VARCHAR (15) |  | Salesperson city |
| State | CHAR (2) |  | Salesperson state |
| Zip | CHAR (5) |  | Salesperson zip |
| Phone | CHAR (10) |  | Salesperson phone number |
| Birthdate | DATE | NOT NULL | Salesperson birthdate |

**Instructor Table:** Records information of all the trainers in the studio

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| IID | CHAR (3) | NOT NULL | Unique instructor number used as a primary key (PK) |
| FName | VARCHAR (10) | NOT NULL | Instructor first name |
| LName | VARCHAR (10) | NOT NULL | Instructor Last name |
| Address | VARCHAR (20) |  | Instructor address |
| City | VARCHAR (15) |  | Instructor city |
| State | CHAR (2) |  | Instructor state |
| Zip | CHAR (5) |  | Instructor state |
| Phone | CHAR (10) |  | Instructor phone |
| Birthdate | DATE | NOT NULL | Instructor birthdate |

**MembershipType Table:** Records the details of the different types of monthly memberships the customers can enroll in

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| MTypeID | CHAR (2) | NOT NULL | Unique membership type number (PK) |
| MName | VARCHAR (10) | NOT NULL | Membership name |
| Num\_Classes | INTEGER | NOT NULL | Number of classes per month allowed |
| MPrice | DECIMAL (5,2) | NOT NULL | Price of the membership |

**MembershipSales Table:** Records the membership sales from salespeople to customers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| M\_CID | CHAR (3) | NOT NULL | Customer ID number of the customer buying the membership (FK and PK) |
| M\_SPID | CHAR (3) | NOT NULL | Membership salesperson ID of the salesperson who sold the membership (FK and PK) |
| M\_TypeID | CHAR (2) | NOT NULL | The ID of the membership type the customer purchased (FK and PK) |
| DateSold | DATE |  | Date on which membership was sold |

The membership type ID is included in the compound primary key so that a customer could potentially later upgrade or downgrade to a different membership type without removing the record of the past sale.

**Merchandise Table:** Keeps inventory of the merchandise available to customers for purchase

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| ItemID | CHAR (2) | NOT NULL | Unique item ID number used as a primary key (PK) |
| ItemDesc | VARCHAR (20) | NOT NULL | Item description |
| ItemPrice | DECIMAL (3,2) | NOT NULL | Price for an item |
| QOH | INTEGER |  | Quantity on hand of that item |

**Classes Table:** Records all available classes and their details

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| ClassID | CHAR (3) | NOT NULL | Unique class ID number used as a primary key (PK) |
| ClassDesc | VARCHAR (20) | NOT NULL | The description/ name of the class |
| ClassTime | INTEGER | NOT NULL | What time of day the class takes place |
| ClassLength | INTEGER | NOT NULL | Class length in minutes |
| IID | CHAR (3) | NOT NULL | The instructor ID of the instructor that teaches the class (FK) |

**ClassEnroll Table:** Records all class enrollments - which customers are signed up for which classes and on what date

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| CE\_ClassID | CHAR (3) | NOT NULL | The ID number of the class that will be taken (FK and PK) |
| Class\_Date | DATE | NOT NULL | The date the class will take place (PK) |
| CE\_CID | CHAR (3) | NOT NULL | Customer ID of the customer taking the class (PK) |

**Retail Sales Table:** Records each retail/merchandise sale to customers

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| RSID | CHAR (2) | NOT NULL | Unique retail sales number used as a primary key (PK) |
| RS\_CID | CHAR (3) | NOT NULL | Customer ID of the customer purchasing the items (FK) |
| DateSold | DATE | NOT NULL | The date of the sale |

**RS\_Details (Retail Sales Details):** Records the details of each sale so that for sales involving multiple items we will not have unnecessary data storage

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Data Type** | **Nullable** | **Description** |
| D\_RSID | CHAR (2) | NOT NULL | The Retail Sales ID from the Retail Sales table (FK and PK) |
| D\_ItemID | CHAR (3) | NOT NULL | Item ID number of the item purchased (FK and PK) |
| Quantity | INTEGER | NOT NULL | The quantity sold of the item |

# Relationships

The relationships captured in our database are as follows: Each customer buys a membership from one salesperson. The relationship is 1:N so that a customer can change their membership later on to a different membership without the database losing information from the first membership sales. Customers can also enroll in many classes and each class can have many customers in it, but customers do not have to be enrolled in classes so that we can store customer information before they have signed up for a class. Similarly, classes do not have to have enrollment so that we can add classes before members sign up for them. Each class is taught by one instructor, and every instructor can teach many classes. On the membership sales aspect of the business, every salesperson can sell many memberships and each type of membership can be sold to many members. Our merchandise is purchased by customers and since customers may buy more than one item at once we have included a detailed table to keep track of the items sold.

## 

# Query Documentation

Business/Employee Queries:

1. List all customers from a specific City (Worcester)

SELECT \*

FROM Customer

WHERE Customer.City = 'Worcester';



1. List all customers with a specific membership type (Elite Plus)

SELECT Customer.CID, Customer.FName, Customer.LName, MembershipType.MName

FROM Customer, MembershipSales, MembershipType

WHERE Customer.CID=MembershipSales.M\_CID AND MembershipSales.M\_MTypeID = MembershipType.MTypeID AND MembershipType.MName = 'Elite Plus';



1. List all customers who are enrolled in a specific class (Bodybuilding)

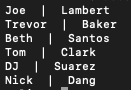
SELECT Customer.FName, Customer.LName

FROM Customer, Classes, ClassEnroll

WHERE Customer.CID = ClassEnroll.CE\_CID

AND ClassEnroll.CE\_ClassID = Classes.ClassID

AND Classes.ClassDesc = 'Bodybuilding';



1. List all customers and their membership type

SELECT Customer.Cid, Customer.FName, Customer.LName, MembershipType.MName

FROM Customer, MembershipSales, MembershipType

WHERE Customer.CID = MembershipSales.M\_CID AND MembershipType.MTypeID = MembershipSales.M\_MTypeID;



1. List customers and membership price that he/she paid, but only list those customers whose membership costs more than $120

SELECT Customer.cid, Customer.FName, Customer.LName, MembershipType.MPrice

FROM Customer, MembershipSales, MembershipType

WHERE Customer.CID=MembershipSales.M\_CID

AND MembershipType.MTypeID= MembershipSales.M\_MTypeID

AND MembershipType.MPrice > 120

ORDER BY MembershipType.MPrice;



1. List the sales person and the number of each membership type that he/she sold

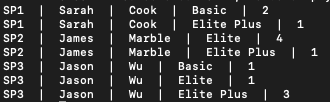
SELECT SalesPerson.SPID,SalesPerson.FName, SalesPerson.LName, MembershipType.MName, COUNT(MembershipType.MName)

FROM SalesPerson, MembershipSales, MembershipType

WHERE SalesPerson.SPID=MembershipSales.M\_SPID AND

MembershipSales.M\_MTypeID=MembershipType.MTypeID

GROUP BY SalesPerson.SPID, MembershipType.MName;



1. Show the overall revenue from the membership sales of a specific salesperson (Jason Wu)

SELECT SUM(MembershipType.MPrice) AS SALES\_BY\_JASON

FROM SalesPerson, MembershipSales, MembershipType

WHERE SalesPerson.SPID=MembershipSales.M\_SPID AND

MembershipSales.M\_MTypeID=MembershipType.MTypeID

AND SalesPerson.FName='Jason' AND SalesPerson.LName='Wu';



1. List all sales persons who haven’t sold any memberships

SELECT SalesPerson.SPID,SalesPerson.FName,SalesPerson.LName

FROM SalesPerson where SalesPerson.SPID NOT IN

(SELECT SalesPerson.SPID

FROM SalesPerson, MembershipSales

WHERE SalesPerson.SPID=MembershipSales.M\_SPID);



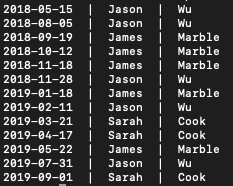
1. List all the membership sales in order by date

SELECT MembershipSales.DateSold, SalesPerson.FName, SalesPerson.LName

FROM MembershipSales, SalesPerson

WHERE MembershipSales.M\_SPID= SalesPerson.SPID

ORDER BY MembershipSales.DateSold;



1. List all retail sales, the customer that bought it and the total price of the sale

SELECT RetailSales.RSID, Customer.FName, Customer.LName, SUM(ItemPrice) AS Total\_Price

FROM Customer, RetailSales, RS\_Details, Merchandise

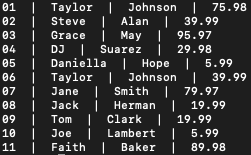
WHERE Customer.CID = RetailSales.RS\_CID

AND RetailSales.RSID = RS\_Details.D\_RSID

AND RS\_Details.D\_ItemID = Merchandise.ItemID

GROUP BY Customer.CID, RetailSales.RSID

ORDER BY RetailSales.RSID;



1. Show the total number of each item sold

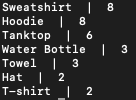
SELECT Merchandise.ItemDesc, SUM(RS\_Details.Quantity) AS Tot\_Quant

FROM RS\_Details, Merchandise

WHERE Merchandise.ItemID = RS\_Details.D\_ItemID

GROUP BY ItemID

ORDER BY Tot\_Quant DESC;



1. List the total number of customers that hold each membership type

SELECT MembershipType.MName, COUNT(MembershipType.MPrice) AS TOTAL\_SALES

FROM MembershipType, MembershipSales

WHERE MembershipSales.M\_MTypeID = MembershipType.MTypeID

GROUP BY MembershipType.MName;

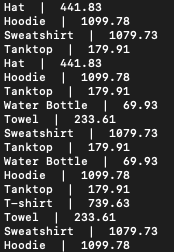


1. Calculate the total value of the inventory of each merchandise item

SELECT Merchandise.ItemDesc, (Merchandise.ItemPrice \* Merchandise.QOH) as Inventory\_Value

FROM Merchandise, RS\_Details

WHERE Merchandise.ItemID = RS\_Details.D\_ItemID;



1. Show the number of people enrolled in each class

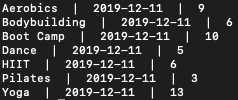
SELECT Classes.ClassDesc, ClassEnroll.Class\_Date, Count(CE\_ClassID)

FROM Classes, ClassEnroll

WHERE Classes.ClassID = ClassEnroll.CE\_ClassID

GROUP BY ClassID, Class\_Date

ORDER BY ClassDesc;



Customer Queries

1. Show all the membership options and the number of classes allowed

SELECT MName, MPrice, Num\_Classes

FROM MembershipType;

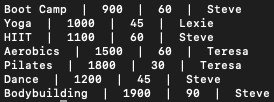


1. Show all classes available and the instructor that coaches it

SELECT Classes.ClassDesc, Classes.ClassTime, Classes.ClassLength, Instructor.FName

FROM Instructor, Classes

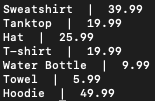
WHERE Instructor.IID = Classes.IID;



1. Show all the merchandise available for purchase

SELECT ItemDesc, ItemPrice

FROM Merchandise;



1. Show all the instructors at the studio

SELECT FName, LName

FROM Instructor;



# App Navigation

As shown in the hierarchy chart of the app. The main homepage of the app consists of two buttons: Employee and Customer. Then based on the user selection, it will be directed to either employee or customer subpages. In Employee, the login page will be shown and after user authentication main employee page will appear where the employee can add classes, update customer information, and generate reports. On the other hand, if in the main page user selects the Customer button, it will be directed to the Customer main page where new customer can sign up, if already signed up, he can enroll in classes or get more information about the Membership Options,

Classes, Merchandise Items, and Instructors.



The followings are some examples of screenshots of the app:

A screenshot of a cell phone

Description automatically generated A screenshot of a cell phone

Description automatically generated A screenshot of a cell phone

Description automatically generated

# A screenshot of a cell phone Description automatically generated A screenshot of a cell phone Description automatically generated A screenshot of a cell phone Description automatically generated

# Technical Documentation

Total Space Estimates

|  |  |
| --- | --- |
| **Current** | 31,046 bytes |
| **Estimated in 5 years** | 195,681 bytes |

**Customer Table:**

**Realistic application: No. of rows = 200**

**5 years growth: No of rows = 1000**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| CID | CHAR (3) | 3 | 600 | 3000 |
| FName | VARCHAR (10) | 10 | 2000 | 10000 |
| LName | VARCHAR (10) | 10 | 2000 | 10000 |
| Address | VARCHAR (20) | 20 | 4000 | 20000 |
| City | VARCHAR (15) | 15 | 3000 | 15000 |
| State | CHAR (2) | 2 | 400 | 2000 |
| Zip | CHAR (5) | 5 | 1000 | 5000 |
| Phone | CHAR (10) | 10 | 2000 | 10000 |
| Birthdate | DATE | 4 | 800 | 4000 |
| Sub total | | | 15800 | 79000 |

**SalesPerson Table:**

**Realistic application: No. of staff = 3**

**5 years growth: No of rows = 6**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| SPID | CHAR (3) | 3 | 9 | 18 |
| FName | VARCHAR (10) | 10 | 30 | 60 |
| LName | VARCHAR (10) | 10 | 30 | 60 |
| Address | VARCHAR (20) | 20 | 60 | 120 |
| City | VARCHAR (15) | 15 | 45 | 90 |
| State | CHAR (2) | 2 | 6 | 12 |
| Zip | CHAR (5) | 5 | 15 | 30 |
| Phone | CHAR (10) | 10 | 30 | 60 |
| Birthdate | DATE | 4 | 12 | 24 |
| **Sub total** | | | 237 | 474 |

**Instructor Table:**

**Realistic application: No. of rows = 7** (one for each class)

**5 years growth: No of rows = 14**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| IID | CHAR (3) | 3 | 21 | 42 |
| FName | VARCHAR (10) | 10 | 70 | 140 |
| LName | VARCHAR (10) | 10 | 70 | 140 |
| Address | VARCHAR (20) | 20 | 140 | 280 |
| City | VARCHAR (15) | 15 | 105 | 210 |
| State | CHAR (2) | 2 | 14 | 28 |
| Zip | CHAR (5) | 5 | 35 | 70 |
| Phone | CHAR (10) | 10 | 70 | 140 |
| Birthdate | DATE | 4 | 28 | 56 |
| **Sub total** | | | 553 | 1106 |

**MembershipType Table:**

No. of memberships = 3 (Elite Plus, Elite, Basic)

**Realistic application: No. of rows = 3**

**5 years growth: No of rows = 6**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| MTypeID | CHAR (2) | 2 | 6 | 12 |
| MName | VARCHAR (10) | 10 | 30 | 60 |
| Num\_Classes | INTEGER | 2 | 6 | 12 |
| MPrice | DECIMAL (5,2) | 5 | 15 | 30 |
| **Sub total** | |  | 57 | 114 |

**MembershipSales Table:**

No. of members =200, average number of memberships per member= 1

**Realistic application: No. of rows = 200**

Assuming a member changes membership type 5 times over the course of 5 years and the number of members increase to 1000 members

**5 years growth: No of rows = 1000 \* 5 = 5000**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| M\_CID | CHAR (3) | 3 | 600 | 15,000 |
| M\_SPID | CHAR (3) | 3 | 600 | 15,000 |
| M\_TypeID | CHAR (2) | 2 | 400 | 10,000 |
| DateSold | DATE | 4 | 800 | 20,000 |
| **Sub total** | | | 2400 | 60,000 |

**Merchandise Table:**

No. of items = 7 (Sweatshirt, Tank top, Hat, T-shirt, Water Bottle, Towel, Hoodie)

**Realistic application: No. of rows = 7**

**5 years growth: No of rows = 21**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| ItemID | CHAR (2) | 2 | 14 | 42 |
| ItemDesc | VARCHAR (20) | 20 | 140 | 420 |
| ItemPrice | DECIMAL (3,2) | 3 | 21 | 63 |
| QOH | INTEGER | 2 | 14 | 42 |
| **Sub total** | | | 189 | 567 |

**Classes Table:**

No. of Classes = 7 (Boot Camp, Yoga, HIIT, Aerobics, Pilates, Dance, Bodybuilding)

**Realistic application: No. of rows = 7**

**5 years growth: No of rows = 14**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| ClassID | CHAR (3) | 3 | 21 | 42 |
| ClassDesc | VARCHAR (20) | 20 | 140 | 280 |
| ClassTime | INTEGER | 2 | 14 | 28 |
| ClassLength | INTEGER | 2 | 14 | 28 |
| IID | CHAR (3) | 3 | 21 | 42 |
| **Sub total** | | | 210 | 420 |

**ClassEnroll Table:**

No. of customers = 200, Average no. of enrollments per customer = 2

**Realistic application: No. of rows = 400**

No. of customers = 1000, Average no. of enrollments per customer = 3

**5 years growth: No of rows = 3000**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| CE\_ClassID | CHAR (3) | 3 | 1200 | 9,000 |
| Class\_Date | DATE | 4 | 1600 | 12,000 |
| CE\_CID | CHAR (3) | 3 | 1200 | 9,000 |
| **Sub total** | | | 4000 | 30,000 |

**Retail Sales Table:**

No. of customers = 200, average no. of retail sales per customer = 2

**Realistic application: No. of rows = 400**

No. of customers = 1000, average no. of retail sales per customer = 2

**5 years growth: No of rows = 2000**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| RSID | CHAR (2) | 2 | 800 | 4,000 |
| RS\_CID | CHAR (3) | 3 | 1200 | 6,000 |
| **Sub total** | | | 2000 | 10000 |

**RS\_Details (Retail Sales Details):** A lookup table between retail sales and merchandize tables which give more details for every retail sale made.

No. of retail sales = 400, average no. of items in each retail sales = 2

**Realistic application: No. of rows = 800**

No. of customers = 1000, average no. of items in each retail sales = 2

**5 years growth: No of rows = 2000**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Length** | **Current Size** | **5-year Size** |
| D\_RSID | CHAR (2) | 2 | 1600 | 4000 |
| D\_ItemID | CHAR (3) | 3 | 2400 | 6000 |
| Quantity | INTEGER | 2 | 1600 | 4000 |
| **Sub total** | | | 5600 | 14000 |

# Backup and Security Features

### We were not able to implement tight backup and security measures since the focus of the project was on the database design of the system rather than the application side To maintain the basic security solution and protect the information of both the Fitness Studio and the customers, we have limited most of the features of the app to only registered employees. Employees with the login username and password are allowed to create, modify, and delete customers, and classes.

### At this point, our app cannot support online database synchronization and sync backup. The simple copying as a backup is used for backup.

### For future improvements on the application, we suggest that it should be connected to an online, centralized server which will also create a backup of the database so critical information is not lost. To ensure data quality we have implemented a few safety options in our app design. Customers only have permission to add information for new customers into the database, however, to ensure that all of the data entered is correct, it gets sent to an employee who checks it to detect any obvious mistakes. To update a customer’s information, such as address or last name, we have implemented a screen in our app that is only accessible to employees such that any change in information must be verified before being entered. These steps will help to ensure there is no spam data in our database and that all data entered is correct.

# Potential Data Quality Problems and How to Reduce Them

Without the use of other advance or medium levels of relational database management systems, data quality and data insurance is hard to implement. In this project, using the features of relational databases and SQlite, we have implemented constraints and limitations to ensure duplicated data is not entered. Also, using the NOT NULL constraints, we have made sure that all required information is entered while filling out the forms.

However, there are still some risks identified in this system. Right now, Fitness Studio cannot verify the information that the users provide. Besides, right now the customer can fill as many forms as he wants and this will create data discrepancy and data reliability problem.

# Data Definition Language (DDL)

.echo ON

.mode list

.separator " | "

PRAGMA foreign\_keys = ON;

DROP TABLE IF EXISTS Customer;

DROP TABLE IF EXISTS SalesPerson;

DROP TABLE IF EXISTS Instructor;

DROP TABLE IF EXISTS MembershipType;

DROP TABLE IF EXISTS MembershipSales;

DROP TABLE IF EXISTS Merchandise;

DROP TABLE IF EXISTS Classes;

DROP TABLE IF EXISTS ClassEnroll;

DROP TABLE IF EXISTS RetailSales;

DROP TABLE IF EXISTS RS\_Details;

CREATE TABLE Customer (

CID CHAR(3) CONSTRAINT Customer\_CID\_pk PRIMARY KEY,

FName VARCHAR(10) NOT NULL ,

LName VARCHAR(10) NOT NULL,

Address VARCHAR(20),

City VARCHAR(15),

State CHAR(2),

Zip CHAR(5),

Phone CHAR(10),

Birthdate DATE NOT NULL

);

CREATE TABLE SalesPerson (

SPID CHAR(3) CONSTRAINT SalesPerson\_SPID\_pk PRIMARY KEY,

FName VARCHAR(10) NOT NULL,

LName VARCHAR(10) NOT NULL,

Address VARCHAR(20),

City VARCHAR(15),

State CHAR(2),

Zip CHAR(5),

Phone CHAR(10),

Birthdate DATE NOT NULL

);

CREATE TABLE Instructor (

IID CHAR(3) CONSTRAINT Instructor\_IID\_pk PRIMARY KEY,

FName VARCHAR(10) NOT NULL,

LName VARCHAR(10) NOT NULL,

Address VARCHAR(20),

City VARCHAR(15),

State CHAR(2),

Zip CHAR(5),

Phone CHAR(10),

Birthdate DATE NOT NULL

);

CREATE TABLE MembershipType (

MTypeID CHAR(2) CONSTRAINT MembershipType\_MTypeID\_pk PRIMARY KEY,

MName VARCHAR(10) NOT NULL,

Num\_Classes INTEGER CONSTRAINT MemType\_cc CHECK((Num\_Classes = 20) OR (Num\_Classes = 15) OR (Num\_Classes = 10)),

MPrice DECIMAL(5,2)

);

CREATE TABLE MembershipSales (

M\_CID CHAR(3) CONSTRAINT MembershipSales\_CID\_fk REFERENCES Customer(CID),

M\_SPID CHAR(3) CONSTRAINT MembershipSales\_SPID\_fk REFERENCES SalesPerson(SPID),

M\_MTypeID CHAR(2) CONSTRAINT MembershipSales\_MTypeID\_fk REFERENCES MembershipType(MTypeID),

DateSold DATE NOT NULL,

CONSTRAINT MembershipSales\_CID\_SPID\_pk PRIMARY KEY(M\_CID, M\_SPID, M\_MTypeID)

);

CREATE TABLE Merchandise(

ItemID CHAR(2) CONSTRAINT Merchandise\_ItemID\_pk PRIMARY KEY,

ItemDesc VARCHAR(20) NOT NULL,

ItemPrice DECIMAL(3,2) NOT NULL,

QOH INTEGER CONSTRAINT Merchandise\_QOH\_cc CHECK (QOH < 200)

);

CREATE TABLE Classes(

ClassID CHAR(3) CONSTRAINT Classes\_ClassID\_pk PRIMARY KEY,

ClassDesc VARCHAR(20) NOT NULL,

ClassTime INTEGER CONSTRAINT Classes\_ClassTime\_cc CHECK (ClassTime >= 0 AND ClassTime <2400),

ClassLength INTEGER CONSTRAINT Classes\_ClassLength\_cc CHECK (ClassLength >= 0 AND ClassLength < 120),

IID CHAR(3) CONSTRAINT Classes\_IID\_fk REFERENCES Instructor(IID)

);

CREATE TABLE ClassEnroll(

CE\_ClassID CHAR(3) CONSTRAINT ClassEnroll\_CE\_ClassID\_fk REFERENCES Classes(ClassID),

CE\_CID CHAR(3) CONSTRAINT ClassEnroll\_CE\_CID\_fk REFERENCES Customer(CID),

Class\_Date DATE NOT NULL,

CONSTRAINT ClassEnroll\_ClassID\_CID\_Date\_pk PRIMARY KEY(CE\_ClassID, Class\_Date, CE\_CID)

);

CREATE TABLE RetailSales(

RSID CHAR(2) CONSTRAINT RetailSales\_RSID\_pk PRIMARY KEY,

RS\_CID CONSTRAINT RetailSales\_RS\_CID\_fk REFERENCES Customer(CID),

DateSold DATE

);

CREATE TABLE RS\_Details (

D\_RSID CHAR(2) CONSTRAINT RS\_Details\_RSID\_fk REFERENCES RetailSales(RSID),

D\_ItemID CHAR(3) CONSTRAINT RS\_Details\_ItemID\_fk REFERENCES Merchandise(ItemID),

Quantity INTEGER,

CONSTRAINT RS\_Details\_RSID\_ItemID\_pk PRIMARY KEY(D\_RSID, D\_ItemID)

);

.save Fitness\_Club.DB

.output stdout

.echo OFF