****

**Database Management**

**GYM Project**

**Dr. Anas Samara**

**Kareen Sead**

**Alaa Abuzer**

**Rami Abuayyash**

**Welcome to our database project:**

At the start of our project, we set out to design and build a comprehensive database system for a local gym. To ensure that our solution met the needs of the gym's staff and members, we first needed to gather and analyze their requirements.

To do this, we conducted a series of interviews with key stakeholders, including gym managers, trainers, and members. During these interviews, we asked questions about their current processes and pain points, as well as their ideal workflows and features for the new system.

We then used the insights we gained from these interviews to create a detailed requirements document, which we used as the foundation for our design and development work. This document included a range of functional and non-functional requirements, such as the ability to track member attendance, manage equipment inventory, and generate reports on key metrics like revenue and membership growth.

Once we had a clear understanding of the gym's requirements, we began the process of designing and building the database system. We started with a conceptual design phase, in which we created an overall schema for the database and identified the key entities and relationships that needed to be captured.

We then moved on to the physical design phase, in which we selected specific database technologies and tools to implement our schema. We used a combination of open-source and proprietary software, including MySQL for our relational database management system, and Miro for ERD and UML diagram.

Throughout the development process, we stayed in close communication with the gym's stakeholders, soliciting feedback and making adjustments as needed to ensure that the system met their needs. We also conducted extensive testing and quality assurance to ensure that the system was reliable, scalable, and easy to use.

**The gym components' relationships are :**

* The staff may be a nutrition specialist, a coach, an accountant, etc.
* The staff may register multiple payments.
* The staff may manage multiple facilities, and the facility may contain multiple staff.
* A nutritionist may have multiple appointments with members.
* A coach may train multiple numbers.
* A coach may write multiple workout plans.
* A coach may give multiple classes, and a class may contain multiple coaches.
* A member may pay multiple payments.
* A member may attend multiple classes, and a class may contain multiple members.
* A facility may contain multiple pieces of equipment.
* A member may train in multiple facilities, and a facility may contain multiple members.
* A member may have multiple workout plans.

**The SQL commands to create tables:**

CREATE TABLE Staff (

Staff\_ID INTEGER PRIMARY KEY,

Staff\_Name TEXT NOT NULL,

Staff\_Phone\_Num TEXT NOT NULL,

Staff\_Work\_period INTEGER NOT NULL,

Staff\_Salary INTEGER NOT NULL

);

CREATE TABLE Nutrition\_Specialist (

Nutrition\_ID INTEGER PRIMARY KEY,

Members\_ID INTEGER NOT NULL,

Food\_Plan TEXT NOT NULL,

FOREIGN KEY (Nutrition\_ID) REFERENCES Staff(Staff\_ID)

FOREIGN KEY (Members\_ID) REFERENCES Members(M\_ID)

);

CREATE TABLE Coach (

Coach\_ID INTEGER PRIMARY KEY,

Coach\_Work\_period INTEGER NOT NULL

FOREIGN KEY (Coach\_ID) REFERENCES Staff(Staff\_ID)

);

CREATE TABLE Members (

M\_ID INTEGER PRIMARY KEY,

M\_Name TEXT NOT NULL,

M\_Contact TEXT NOT NULL,

M\_Address TEXT NOT NULL,

M\_EndOfMembershipDate DATE NOT NULL,

M\_IsVIP VARCHAR NOT NULL

);

CREATE TABLE Payment (

P\_ID INTEGER PRIMARY KEY,

M\_ID INTEGER NOT NULL,

P\_date DATE NOT NULL,

P\_Amount INTEGER NOT NULL,

Staff\_ID INTEGER NOT NULL,

FOREIGN KEY (M\_ID) REFERENCES Members(M\_ID),

FOREIGN KEY (Staff\_ID) REFERENCES Staff(Staff\_ID)

);

CREATE TABLE WorkOutPlan (

P\_ID INTEGER PRIMARY KEY,

M\_ID INTEGER PRIMARY KEY NOT NULL,

Coach\_ID INTEGER PRIMARY KEY NOT NULL,

P\_TimeNeeded INTEGER NOT NULL,

FOREIGN KEY (M\_ID) REFERENCES Members(M\_ID),

FOREIGN KEY (Coach\_ID) REFERENCES Coach(Coach\_ID)

);

CREATE TABLE Classes (

Class\_ID INTEGER PRIMARY KEY,

Class\_Name TEXT NOT NULL,

Class\_Period DATE NOT NULL,

Class\_Duration INTEGER NOT NULL,

);

CREATE TABLE Facilities (

F\_ID INTEGER PRIMARY KEY,

F\_Name TEXT NOT NULL,

F\_Period INTEGER NOT NULL,

);

CREATE TABLE Equipment (

E\_ID INTEGER PRIMARY KEY,

E\_Name TEXT NOT NULL,

E\_Description TEXT NOT NULL,

F\_ID INTEGER NOT NULL,

FOREIGN KEY (F\_ID) REFERENCES Facilities(F\_ID)

);

CREATE TABLE Member\_Facilities (

M\_ID INTEGER PRIMARY KEY,

F\_ID INTEGER PRIMARY KEY,

FOREIGN KEY (M\_ID) REFERENCES Members(M\_ID),

FOREIGN KEY (F\_ID) REFERENCES Facilities(F\_ID)

);

CREATE TABLE Facilities-Staff(

Staff\_ID INTEGER PRIMARY KEY,

F\_ID INTEGER PRIMARY KEY,

FOREIGN KEY (Staff\_ID ) REFERENCES Staff(Staff\_ID ),

FOREIGN KEY (F\_ID) REFERENCES Facilities(F\_ID)

);

CREATE TABLE Classes-Members(

Class\_ID INTEGER PRIMARY KEY,

M\_ID INTEGER PRIMARY KEY,

FOREIGN KEY (Class\_ID ) REFERENCES Classes(Staff\_ID ),

FOREIGN KEY (M\_ID) REFERENCES Members(M\_ID),

);

CREATE TABLE Classes-Coach(

Class\_ID INTEGER PRIMARY KEY,

Coach\_ID INTEGER PRIMARY KEY,

FOREIGN KEY (Class\_ID ) REFERENCES Classes(Staff\_ID ),

FOREIGN KEY (Coach\_ID ) REFERENCES Coach(M\_ID),

);

