Christopher Rusu #101189532
3005 V2 Project Report

https://github.com/ChrisRusu1/3005FinalProjectV2

https://www.youtube.com/watch?v=kwhiI3ACRyc

Conceptual Design

The Health and Fitness Club Management System is designed to make running a fitness club smoother by keeping track of members, classes, and staff all in one place. At the heart of the system is a database that connects everything together. For example, there's a Members table that holds all the member details and links to their health stats, fitness goals, what they've achieved, and their payments. This setup lets members keep an eye on their health journey and stay on top of their fees.

In addition to the member-focused features, there's a spot in the database for Employees, which separates trainers from admin staff. This ties into the PersonalTrainingSessions for scheduling and the PayrollPayments for managing money matters for the staff. The database also looks after the physical stuff like rooms and equipment, so the club's spaces and gear are always ready to go.

In short, this system is all about making sure that whether you're a member working out, a trainer organizing classes, or running the club's back office, everything you need is easy to get to and use. This meets the main goal of making a fitness club easy to manage and ensuring everyone has a good experience.

DDL FILE

```
-- Drop existing tables to ensure the script can run multiple times without errors
DROP TABLE IF EXISTS FitnessAchievements CASCADE;
DROP TABLE IF EXISTS FitnessGoals CASCADE;
DROP TABLE IF EXISTS HealthMetrics CASCADE;
DROP TABLE IF EXISTS PayrollPayments CASCADE;
DROP TABLE IF EXISTS Payments CASCADE;
DROP TABLE IF EXISTS PersonalTrainingSessions CASCADE;
DROP TABLE IF EXISTS ClassBookings CASCADE;
DROP TABLE IF EXISTS FitnessClasses CASCADE;
DROP TABLE IF EXISTS Members CASCADE;
DROP TABLE IF EXISTS Employees CASCADE;
DROP TABLE IF EXISTS ExerciseRoutines CASCADE;
DROP TABLE IF EXISTS Equipment CASCADE;
DROP TABLE IF EXISTS Rooms CASCADE;

-- Create Employees table (renamed from Trainers for consistency, can adjust as needed)
CREATE TABLE Employees (
    EmployeeID SERIAL PRIMARY KEY,
    Name VARCHAR(255),
    Type VARCHAR(50),  -- 'Trainer' or 'Admin'
    Specialization VARCHAR(255) NULL,  -- NULL for Admin
    Role VARCHAR(255) NULL,  -- NULL for Trainers
    Salary DECIMAL(10, 2),
    Password VARCHAR(255)
);

-- Create Members table
CREATE TABLE Members (
    MemberID SERIAL PRIMARY KEY,
    Name VARCHAR(255),
    Email VARCHAR(255) UNIQUE,
    Address VARCHAR(255),
    BirthDate DATE,
    MembershipStartDate DATE,
    MembershipType VARCHAR(50),
    Password VARCHAR(255)
);

-- Create Rooms table
CREATE TABLE Rooms (
    RoomID SERIAL PRIMARY KEY,
    RoomName VARCHAR(255),
    Capacity INT
);

-- Create Fitness Classes table
CREATE TABLE FitnessClasses (
    ClassID SERIAL PRIMARY KEY,
    ClassName VARCHAR(255),
    TrainerID INT REFERENCES Employees(EmployeeID),
    RoomID INT REFERENCES Rooms(RoomID),  -- Linking classes to rooms
    Cost DECIMAL(10, 2),
    StartTime TIMESTAMP,
    EndTime TIMESTAMP
```

```sql
);

-- Create ClassBookings table for managing class members (many-to-many relationship)
CREATE TABLE ClassBookings (
    BookingID SERIAL PRIMARY KEY,
    ClassID INT REFERENCES FitnessClasses(ClassID),
    MemberID INT REFERENCES Members(MemberID),
    BookingDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Create Personal Training Sessions table
CREATE TABLE PersonalTrainingSessions (
    SessionID SERIAL PRIMARY KEY,
    TrainerID INT REFERENCES Employees(EmployeeID),
    RoomID INT REFERENCES Rooms(RoomID),  -- Linking sessions to rooms
    StartTime TIMESTAMP,
    EndTime TIMESTAMP,
    MemberID INT REFERENCES Members(MemberID) NULL,  -- Initially NULL, set when a
member books the session
    SessionStatus VARCHAR(20) DEFAULT 'Available'
);

-- Create Equipment table
CREATE TABLE Equipment (
    EquipmentID SERIAL PRIMARY KEY,
    EquipmentType VARCHAR(255),
    PurchaseDate DATE,
    LastMaintenanceDate DATE,
    RoomID INT REFERENCES Rooms(RoomID)  -- Equipment linked to specific rooms
);

-- Create Payments table for members
CREATE TABLE Payments (
    PaymentID SERIAL PRIMARY KEY,
    MemberID INT REFERENCES Members(MemberID),
    Date DATE,
    Amount DECIMAL(10, 2),
    Status VARCHAR(50)
);

-- Create Payroll Payments table for employees
CREATE TABLE PayrollPayments (
    PayrollID SERIAL PRIMARY KEY,
    EmployeeID INT REFERENCES Employees(EmployeeID),
    PaymentDate DATE,
    Amount DECIMAL(10, 2),
    Status VARCHAR(50)
);

-- Health metrics table to store various health data points for members
CREATE TABLE HealthMetrics (
    MetricID SERIAL PRIMARY KEY,
    MemberID INT REFERENCES Members(MemberID),
    Date DATE,
    Weight DECIMAL(5, 2),
    HeartRate INT,  -- Measured in beats per minute
    BloodPressure VARCHAR(50)  -- Stored as a string, e.g., "120/80"
);

-- Fitness goals table for tracking personal fitness objectives
CREATE TABLE FitnessGoals (
    GoalID SERIAL PRIMARY KEY,
    MemberID INT REFERENCES Members(MemberID),
    GoalDescription TEXT,
    TargetDate DATE,
    Status VARCHAR(50)  -- For example, "Pending", "Achieved", "Cancelled"
);

-- Fitness achievements table to record notable achievements
CREATE TABLE FitnessAchievements (
    AchievementID SERIAL PRIMARY KEY,
    MemberID INT REFERENCES Members(MemberID),
    AchievementDescription TEXT,
    AchievementDate DATE
);

-- Create ExerciseRoutines table
CREATE TABLE ExerciseRoutines (
    RoutineID SERIAL PRIMARY KEY,
    MemberID INT REFERENCES Members(MemberID),
    RoutineName VARCHAR(255),
    Description TEXT,
    DateCreated DATE
);
```

DML FILE

```
-- Insert Employees (Trainers and Admins)
INSERT INTO Employees (Name, Type, Specialization, Role, Salary, Password) VALUES
('John Doe', 'Trainer', 'Cardio Specialist', NULL, 50000, 'fqiimehtuvw'),
('Jane Smith', 'Trainer', 'Strength Training', NULL, 52000, 'fqiimehtuvw'),
('Alice Johnson', 'Admin', NULL, 'Operations Manager', 55000, 'fqiimehtuvw');

-- Insert Members
INSERT INTO Members (Name, Email, Address, BirthDate, MembershipStartDate,
MembershipType, Password) VALUES
('Tom Brown', 'tombrown@example.com', '1234 Maple St', '1985-08-15', '2023-01-01',
'Premium', 'fqiimehtuvw'),
('Sara Connor', 'saraconnor@example.com', '5678 Oak Ave', '1990-09-25', '2023-01-10', 'Basic',
'fqiimehtuvw'),
('Mike Tyson', 'miketyson@example.com', '91011 Pine Rd', '1980-07-30', '2023-01-20',
'Premium', 'fqiimehtuvw'),
('Chris Redfield', 'chrisredfield@example.com', '1012 Umbrella St', '1986-11-29', '2023-02-01',
'None', 'fqiimehtuvw');

-- Inserting data into Rooms
INSERT INTO Rooms (RoomName, Capacity) VALUES
('Cardio Room', 10),
('Weight Room', 15),
('Yoga Studio', 12);

-- Inserting data into Equipment
INSERT INTO Equipment (EquipmentType, PurchaseDate, LastMaintenanceDate, RoomID)
VALUES
('Treadmill', '2022-01-01', '2024-04-01', 1),
('Dumbbells', '2022-02-15', '2024-04-10', 2),
('Yoga Mats', '2023-01-01', '2024-04-15', 3);

-- Inserting data into Fitness Classes with assigned rooms and times
INSERT INTO FitnessClasses (ClassName, TrainerID, RoomID, Cost, StartTime, EndTime)
VALUES
('Morning Yoga', 1, 3, 15.00, '2024-05-01 08:00:00', '2024-05-01 09:00:00'),
('Power Lifting', 2, 2, 20.00, '2024-05-01 10:00:00', '2024-05-01 11:00:00'),
('High-Intensity Cardio', 3, 1, 18.00, '2024-05-01 12:00:00', '2024-05-01 13:00:00');

-- Inserting data into Personal Training Sessions with assigned rooms
INSERT INTO PersonalTrainingSessions (TrainerID, RoomID, StartTime, EndTime,
SessionStatus) VALUES
```

```sql
(1, 1, '2024-05-01 08:00:00', '2024-05-01 09:00:00', 'Available'),
(2, 2, '2024-05-01 10:00:00', '2024-05-01 11:00:00', 'Available'),
(3, 3, '2024-05-01 12:00:00', '2024-05-01 13:00:00', 'Available');

-- Insert Health Metrics for a member
INSERT INTO HealthMetrics (MemberID, Date, Weight, HeartRate, BloodPressure) VALUES
(1, '2023-12-01', 78.5, 72, '120/80'),
(2, '2023-12-05', 65.0, 75, '118/79');

-- Insert Fitness Goals
INSERT INTO FitnessGoals (MemberID, GoalDescription, TargetDate, Status) VALUES
(1, 'Lose 5 kg by June', '2024-06-01', 'Pending'),
(2, 'Run a 5k without stopping', '2024-05-15', 'Pending');

-- Insert Fitness Achievements
INSERT INTO FitnessAchievements (MemberID, AchievementDescription, AchievementDate)
VALUES
(1, 'Completed first marathon', '2023-11-10'),
(2, 'Lifted personal best in bench press', '2023-11-20');

-- Insert example exercise routines for members
INSERT INTO ExerciseRoutines (MemberID, RoutineName, Description, DateCreated) VALUES
(1, 'Cardio Blast', 'A high-intensity routine focused on improving cardiovascular health. Includes
30 minutes of running, 15 minutes of cycling, and 15 minutes of HIIT.', '2023-01-01'),
(1, 'Strength Training', 'Comprehensive weight training session targeting all major muscle
groups. Includes 3 sets of bench presses, squats, deadlifts, and overhead presses.',
'2023-01-08'),
(2, 'Beginner Yoga', 'Gentle yoga routine for beginners, focusing on flexibility and mindfulness.
Includes basic poses like downward dog, warrior, and tree pose.', '2023-01-15'),
(3, 'Marathon Prep', 'A progressive running plan designed to prepare for a half marathon.
Includes long-distance runs, sprints, and rest days.', '2023-01-20');

-- Insert ClassBookings for existing members
INSERT INTO ClassBookings (ClassID, MemberID, BookingDate) VALUES
(1, 1, '2024-04-12 09:00:00'),
(2, 1, '2024-04-13 10:00:00'),
(3, 2, '2024-04-14 11:00:00'),
(1, 3, '2024-04-15 12:00:00');

-- Insert booked PersonalTrainingSessions for existing members
INSERT INTO PersonalTrainingSessions (TrainerID, StartTime, EndTime, MemberID,
SessionStatus) VALUES
(1, '2024-04-12 09:00:00', '2024-04-12 10:00:00', 1, 'Booked'),
(2, '2024-04-12 11:00:00', '2024-04-12 12:00:00', 2, 'Booked'),
(1, '2024-04-13 09:00:00', '2024-04-13 10:00:00', 3, 'Booked'),
(2, '2024-04-14 14:00:00', '2024-04-14 15:00:00', 1, 'Booked');

-- Insert initial payments for members based on their membership type
INSERT INTO Payments (MemberID, Date, Amount, Status)
SELECT MemberID, CURRENT_DATE,
    CASE WHEN MembershipType = 'Premium' THEN 150.00
        WHEN MembershipType = 'Basic' THEN 100.00
        ELSE 120.00 -- default price if not Premium or Basic
    END,
    'Unpaid'
FROM Members;

-- Inserting payroll data for staff members
INSERT INTO PayrollPayments (EmployeeID, PaymentDate, Amount, Status) VALUES
-- Assuming EmployeeIDs from 1 to 3 are valid and correspond to existing staff members in the
Employees table
(1, '2024-04-01', 3000.00, 'Paid'),
(1, '2024-05-01', 3000.00, 'Unpaid'),
(2, '2024-04-01', 2500.00, 'Paid'),
(2, '2024-05-01', 2500.00, 'Unpaid'),
(3, '2024-04-01', 3500.00, 'Paid'),
(3, '2024-05-01', 3500.00, 'Unpaid');
```

Implementation

For the implementation phase, I rolled out a comprehensive solution that streamlines the interaction between the club's patrons and its services. I developed a set of Python functions that serve as the front-end operations members, trainers, and admins interact with. These functions take care of everything from signing up new members to booking classes and managing schedules. They're easy to navigate, making the process of managing your fitness journey or handling administrative tasks straightforward.

Underneath these user-friendly functions lies a robust PostgreSQL database. This database houses all the data tables we've designed, structured to capture the multifaceted relationships between different entities like members, staff, classes, and equipment. I paid special attention to the database's integrity by ensuring that related data is updated or deleted appropriately, minimizing errors or orphan data. The integration of the Python code with the database was meticulously tested to guarantee that users would have a seamless experience whether they're checking in for a workout or updating the club's maintenance log.

Bonus Features

In addition to the core features of our system, I introduced several bonus functionalities to enhance the management experience and security. Notably, the payroll management feature automates and simplifies the process of handling payments for the club's staff, allowing administrators to view outstanding salaries and process payments efficiently. Security is bolstered through the implementation of the Caesar cipher for password encryption, ensuring that all user credentials are stored securely. These additions not only streamline administrative tasks but also safeguard sensitive information, offering both convenience and protection to enhance the operational efficiency of the fitness club.