COMP3005 Final Project Report

Health and Fitness Club Management System

Student Name: Hoa Nguyen
Student Number: 101268337

YouTube Link:

Conceptual Design:

Requirements:

Members:

1. Register their profiles

   Dashboard:

   → Tracks exercise routines
   → Tracks fitness achievements
   → Tracks health statistics.
2. Manage their profiles
3. Set personal fitness goals and input health metrics
4. Schedule, reschedule, or cancel personal training sessions with certified trainers
5. Register for group fitness classes, workshops, and other events
6. Receive timely reminders for their sessions.

Trainers:

1. Manage their schedules
2. View member profiles
3. Input progress notes after each training session.

Administrative staff:

1. Features to oversee club resources effectively:

   → Managing room bookings
   → Monitoring fitness equipment maintenance
   → Updating class schedules

   → Oversee billing, process payments for membership fees and personal training sessions and other services,..
   → Monitor club activities for quality assurance

Loyalty program: (unique selling point)

Transaction:

→ Earns members loyalty points
→ Redeemed for future services.

Conceptual design explanation of the database:

Entities: 10 tables

Member, Trainer, and Administrative Staff
Training Session
Activity
Fitness Goal
Billing
Loyalty Program
Equipment
Room

Assumptions: (Cardinalities & Participation)
Below is the description of some important (not all) relationships based on my assumption.

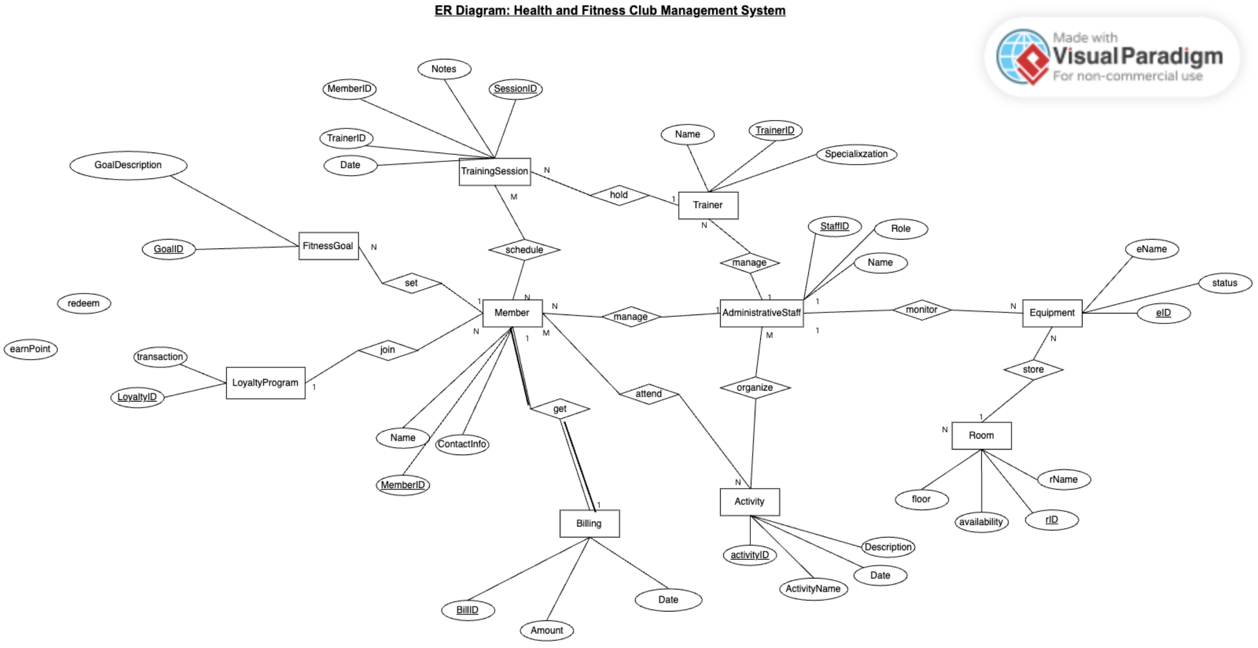
Explanations:
I have chosen to design my database system as above to increase the clarity of the design and maintain the simplicity of the system with many relations. The 3 main users of the system would have relations with their corresponding needs. Staff manages the profiles of the trainers and the members.

Members can have relationships with Personal Training Sessions, Activities, Personal Fitness Goals, Billings, and Loyalty Programs. There are multiple loyalty programs available but members are only able to choose 1 and stick with it.
Personal Training Sessions are managed by Trainers and can have multiple Members attending.
Activities can be managed by Administrative Staff and can have multiple Members attending.
Billing is associated with a Member. Loyalty Programs can have multiple Members enrolled.
Administrative Staff manages Equipment, Rooms, Activities, and Billings.

Relationships:
Member: have many Training Sessions, Fitness Goals, and Bills (if applicable for services).
Trainer: conduct many Training Sessions.
Administrative Staff: oversee Equipment, Rooms, Events, and Billings.
Training Session: Connects a Member with a Trainer.
Activities: Might involve Members and/or Administrative Staff.
Fitness Goal: Belongs to a Member.
Billing: Relates to Members for payments.
→ Total Participation of 1-to-1 relationship.
→ As each member must have a billing payment for the membership and each billing is specific to 1 customer.

Loyalty Program: Connects with Members for earning/redeeming points.
Equipment:Could be present in Rooms.
Room: Can contain Equipment.

ER-diagram:

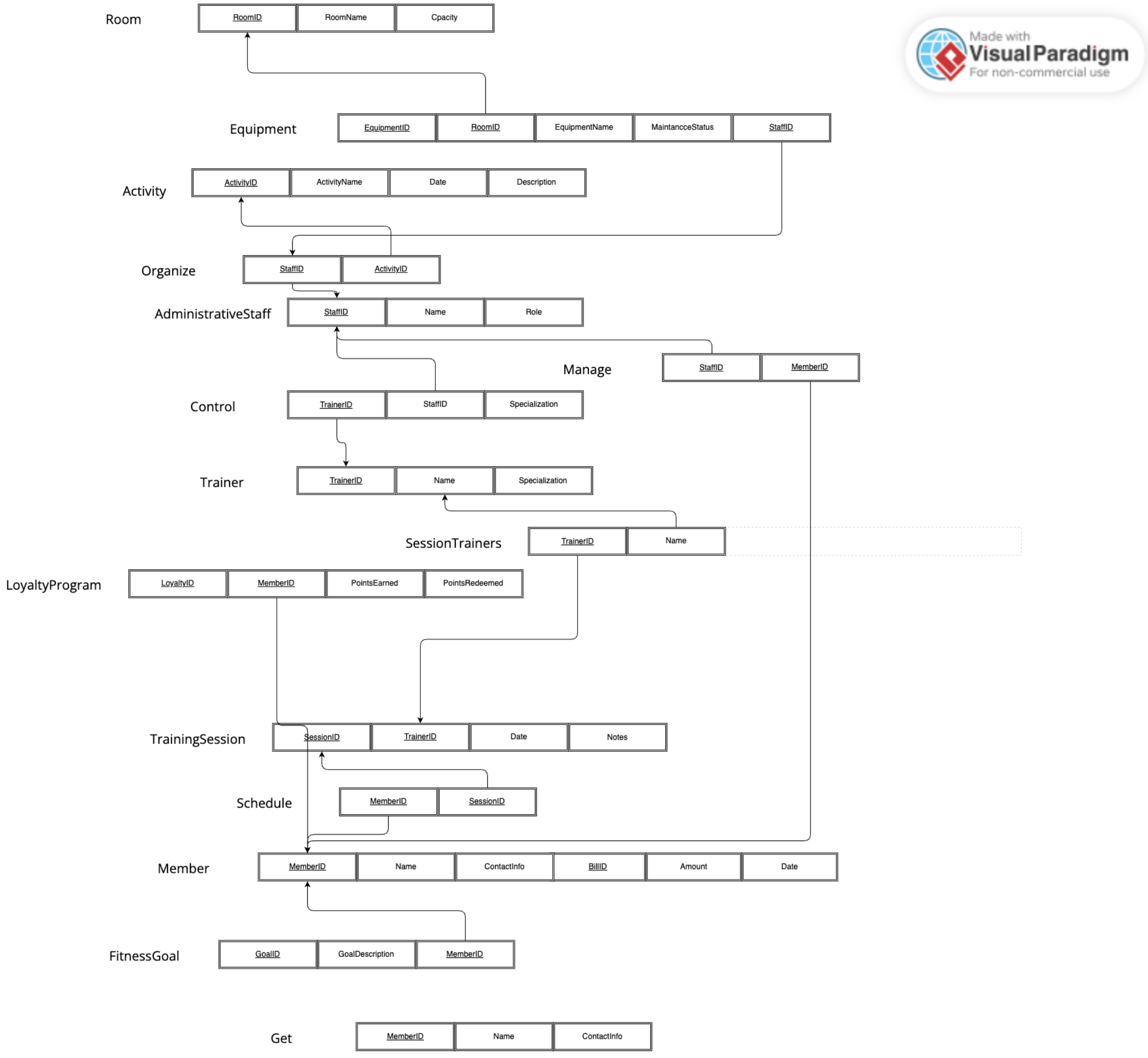


Reduction to Relation Schemas:

Database Structure:

Member(MemberID PK, Name, ContactInfo)
Trainer(TrainerID PK, Name, Specialization)
AdministrativeStaff(StaffID PK, Name, Role)
TrainingSession(SessionID PK, TrainerName, TrainerID, Date, Notes)
Activity(ActivityID PK, ActivityName, Date, Description)
FitnessGoal(GoalID PK, MemberID, GoalDescription)
Billing(BillID PK, MemberID, Amount, Date)
LoyaltyProgram(LoyaltyID PK, MemberID, PointsEarned, PointsRedeemed)
Room(RoomID PK, RoomName, Capacity)
Equipment(EquipmentID PK, EquipmentName, RoomID, MaintenanceStatus)

Normalization of Relation Schemas:



Current relational database schema.

Currently, there are partial dependencies such that part of a composite primary key determines a non-prime attribute.

Hence, I would decompose into 2NF to get rid of the dependencies.

Right now, the table TrainingSession currently has 2 keys, with 1 foreign key TrainerID and PK SessionID forming a composite key. While Trainer can get TrainerName from TrainerID directly in TrainingSession, hence, this creates a partial dependency of the relationship.
As {SessionID, TrainerID} → TrainerName. However, TrainerID → TrainerName.

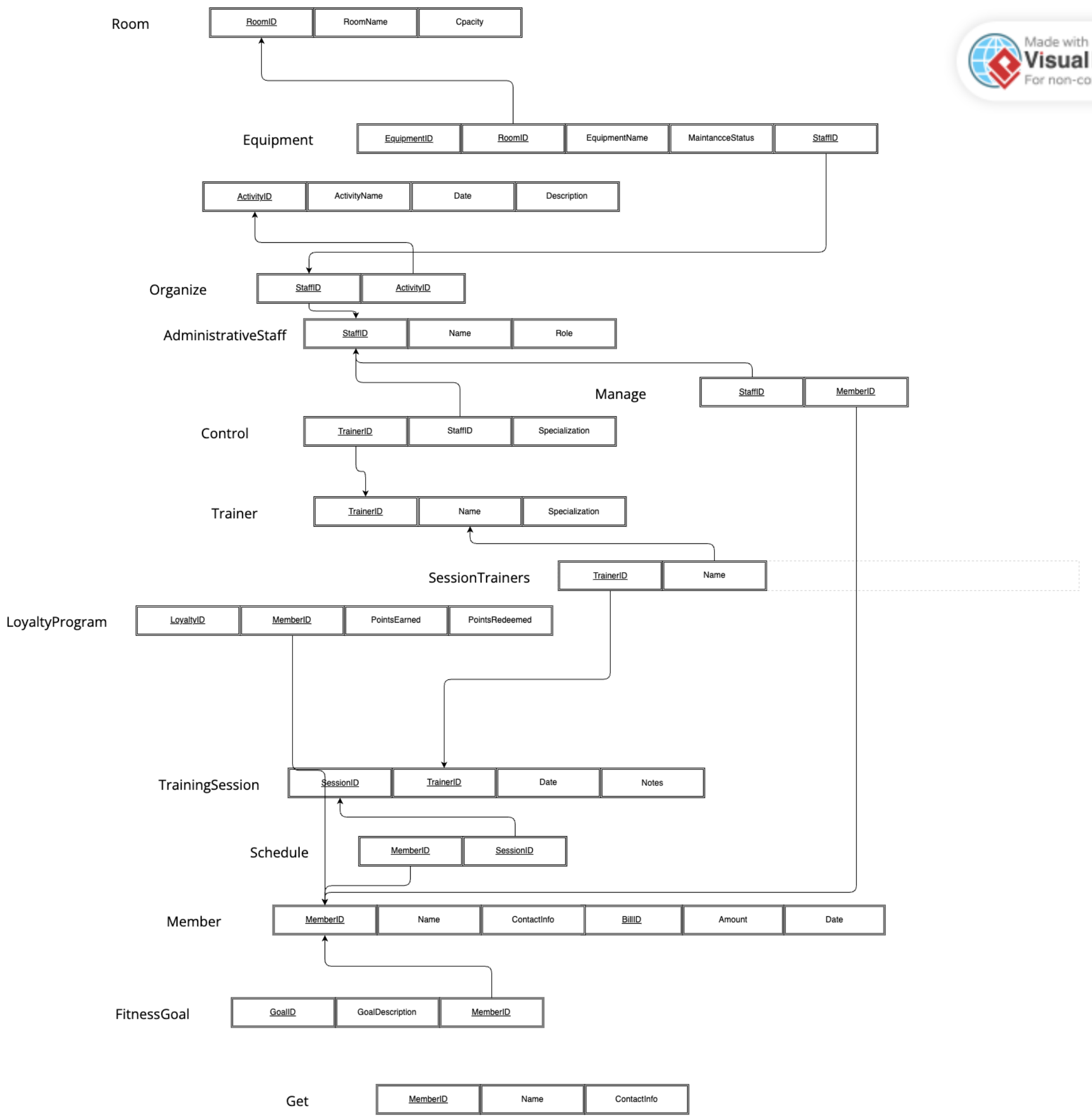
To solve this, I have to create a new table, SessionTrainers, to hold information about Trainers here.

TrainingSession(SessionID PK, TrainerID, TrainerName, Date, Notes) now becomes:
→ TrainingSession(SessionID PK, TrainerID, Date, Notes)
→ SessionTrainers(TrainerID, TrainerName)

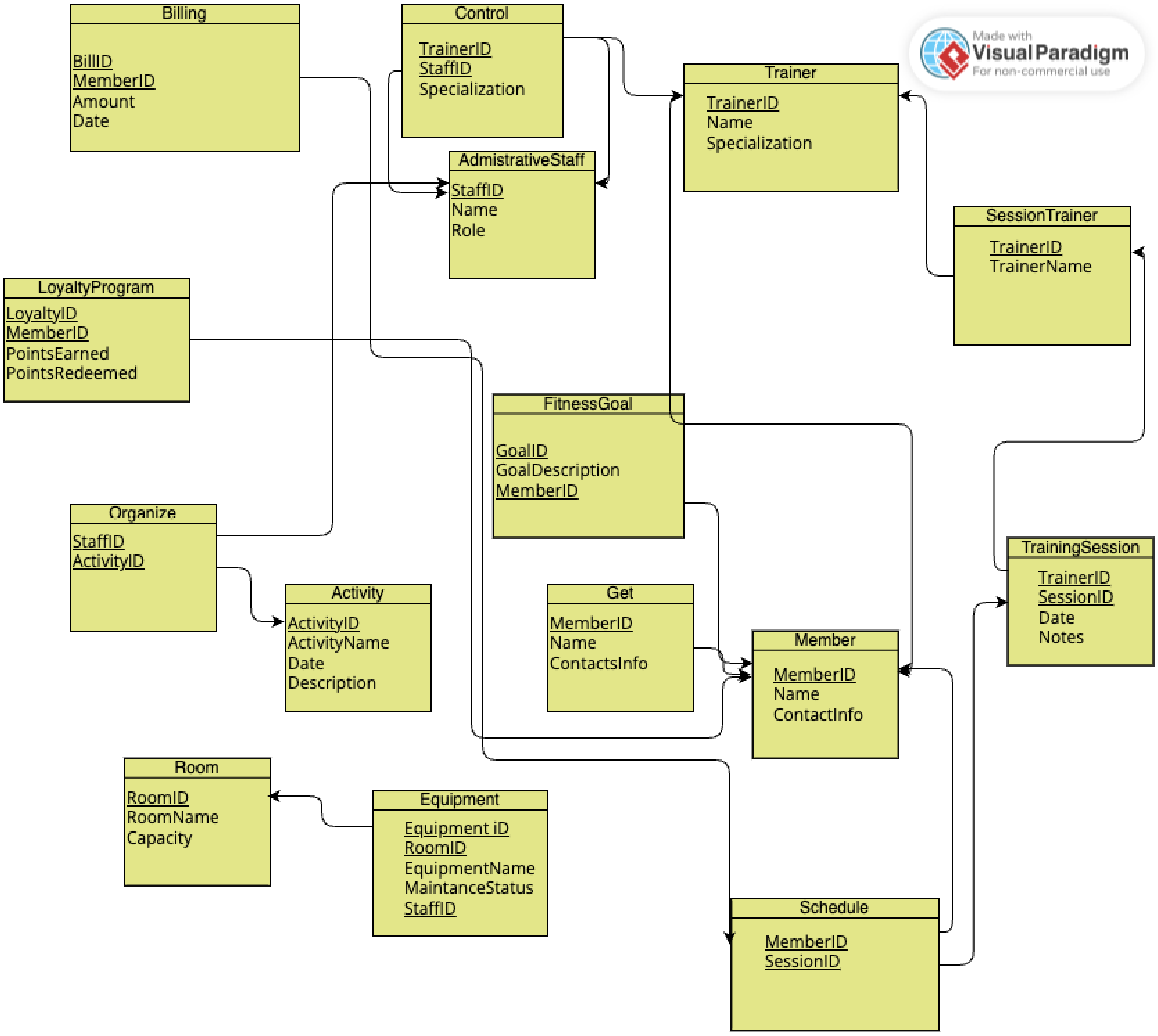
→ **Decomposition into 2NF.**

After looking at partial dependency, now we look at transitive dependencies.
So if
X → Z and Z → Y hold, where Z is non prime attributes, then relationship is transitive dependencies.
From the proposed tables, it seems like they're in 3NF as well, as each attribute appears to be dependent on the primary keys or candidate keys.

After normalization, the schema look like this:



Database Schema Diagram:



Implementation: Desktop, console based with user-model interaction

Workflow:
User run program → user read options → user choose options → user input changed data or program display result → data is stored and changed.

Bonus Features: Bonus features are in the GitHub link. I did 2 ways
displayUI.py shows the GUI interface for the adding and displaying of user data.
Server.py gets users to interact with the query by adding, deleting, and changing the query data through the console.

GitHub Repository: https://github.com/HoaKatie/HealthAndFitnessSystem