Health and Fitness Club Management System - Project Report

COMP3005A - Dr. Abdelghny Orogot

Shuva Gautam - 101223315

December 10, 2023

Table of Contents

Project Requirements

The Health and Fitness Club Management System is a database-driven application that manages the operations of a fitness club. It is designed to help the club manage its members, trainers, administrative staff, training sessions, fitness classes, billing, and loyalty points. More details about them are provided below.

Members

- Members will have a unique member ID and register by providing their first name, last name, date of birth, email, and a hashed password to sign in. They also have a loyalty points balance.
- Members can display their exercise routines, fitness goals, and health statistics.
    - **Exercise Routines**
        - They can log routines by providing the date, duration, and a routine type
    - **Fitness Goals**
        - Members can write fitness goals by giving a goal name, date to accomplish it, and description of the goal, and can mark the status as complete or not
    - **Health Metrics**
        - They can provide health metrics tracking their recording date, weight, height, caloric intake and body fat
- They can take training sessions and join in group fitness classes

Trainers

- Trainers are registered with a unique trainer ID, providing first name, last name, email and hashed password to sign in, and certification details.
- They can teach personal training sessions and instruct fitness classes

Administrative Staff

- Administrative staff members are registered with a unique staff ID, providing first name, last name, position, email, and hashed password.
- Administrative staff maintain club resources and equipment
- Handle processing bills

Personal Training Sessions

- Members can take personal training sessions with a trainer. They may have more than one trainer.

- A trainer can also instruct/train many club members
- Sessions have details like date, time, duration, room location and training activity

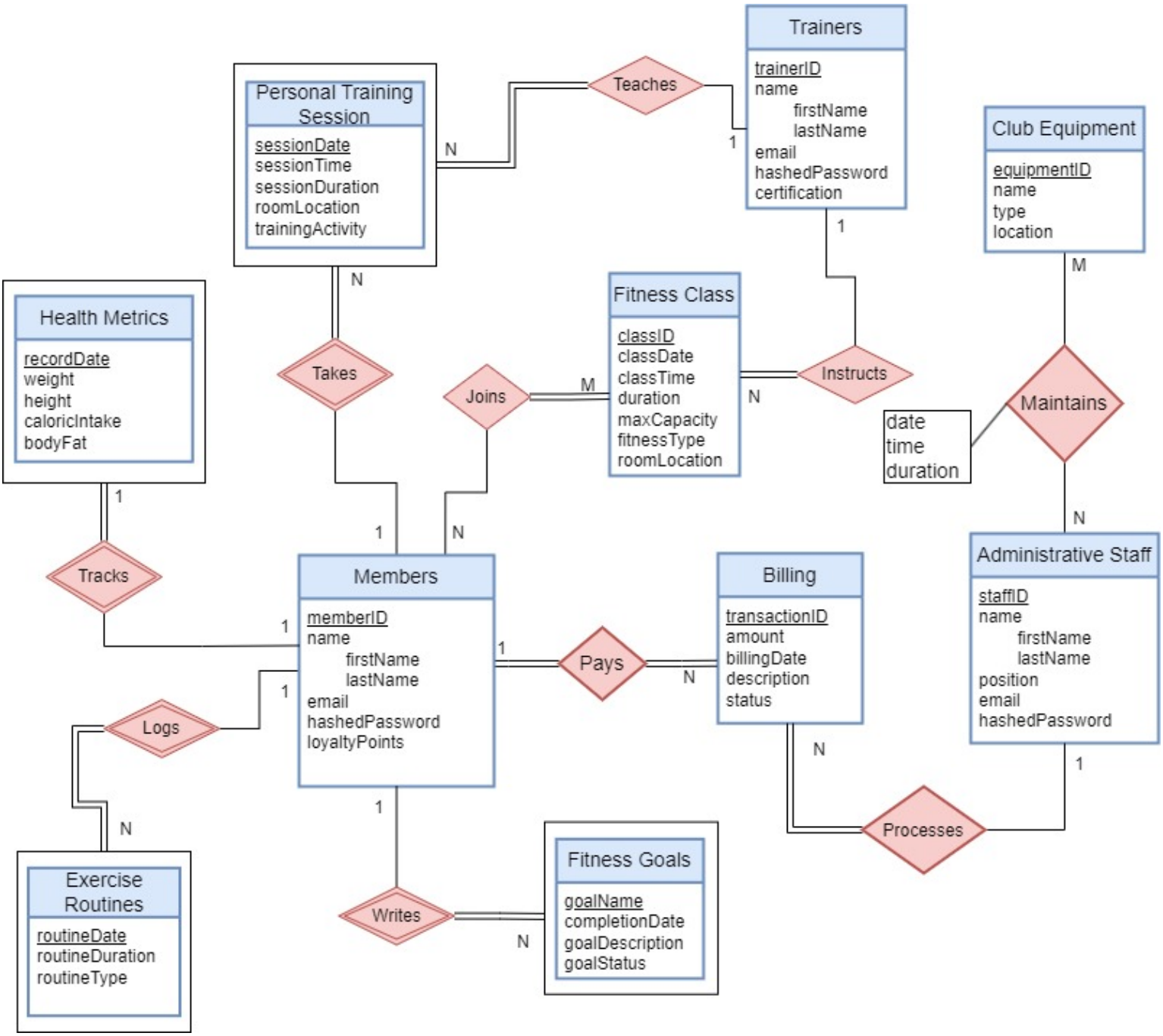
Fitness Classes

- Each fitness class is assigned a unique class ID, with details like class name, date, time, location, max capacity, fitness class type
- Members can enroll in multiple fitness classes.

Billing

- Billing transactions are recorded with a unique transaction ID, associated with a member ID, amount, date, description, and status (Paid, Unpaid).

ER Model Conceptual Design



Assumptions and Explanations of ER Model

The member entity in our ER model has many relations as it can perform and partake in many activities at the club.

Member tracks Health Metrics relation:

- A One-to-One Relation between a Member and Health Metrics
    - One member can track their health metrics, and each health metric is associated with one member
- The health metric is a weak entity as it can't be fully identified by its attributes, so it is total participation
- A member does not have to track health metrics, so it is partial participation

Member logs Exercise Routines relation:

- A One-to-Many Relation between a Member and Routines
    - A member can log many workout routines and each routine is logged by one member
- The Routine entity is a weak entity as it can't be fully identified by its attributes, so it is total participation
- A member does not have to log workout routines, so it is partial participation

Member writes Fitness Goals relation:

- A One-to-Many Relation between a Member and Goals
    - A member can write many fitness goals but each goal is written by one member
- The Fitness Goals relation is a weak entity, so it is has total participation
- A member does not need to write fitness goals so it is partial participation

Member takes Training Session relation:

- A One-to-Many relation between a Member and a Training Session
    - One member can take many training sessions, and each session is taken by one member
- Training Sessions are a weak entity so their is total participation between it and Members
- A member does not need to take a training session so their is partial participation

Member joins Fitness Class relation:

- A Many-to-Many relation between a Member and a Fitness Class
    - A member can join many fitness classes and each fitness class can have many members

- A member does not have to join a fitness class so it is partial participation
- Each fitness class is joined by at least one member, so their is total participation
-

Member pays Billings relation:

- A One-to-Many relation between a Member and a Billing
    - A member can pay many bills, and each bill is paid by one member
- Every member must pay their bills so it is total participation
- Every bill is paid for by someone, so it is total participation

Now a Trainer can instruct Training Sessions as well as Fitness Classes

Trainer teaches Training Session relation:

- A One-to-Many relation between a Trainer and Training Sessions
    - A trainer can teach many Training Sessions and each training session is taught by one trainer
- A trainer may not teach any training sessions so this is partial participation
- Every training session is taught by a trainer, and it is a weak entity so this is total participation

Trainer instructs Fitness Class relation:

- A One-to-Many relation between a Trainer and Fitness Class
    - A trainer can instruct many Fitness Classes and each fitness class is instructed by one trainer
- A trainer may not teach any fitness classes so this is partial participation
- Every fitness class is taught by a trainer, so it is total participation

Lastly, the Administrative Staff can process Billings as well as maintain the Club Resources

Administrative Staff processes Billing relation:

- A one-to-many relation between an administrative staff and a billing transaction
    - An admin staff can process many billing transactions, and each bill is processed by one admin staff
- An admin staff may not process any bills so it is partial participation
- A bill is processed by someone, so it is total participation

Administrative Staff maintains Club Equipment relation:

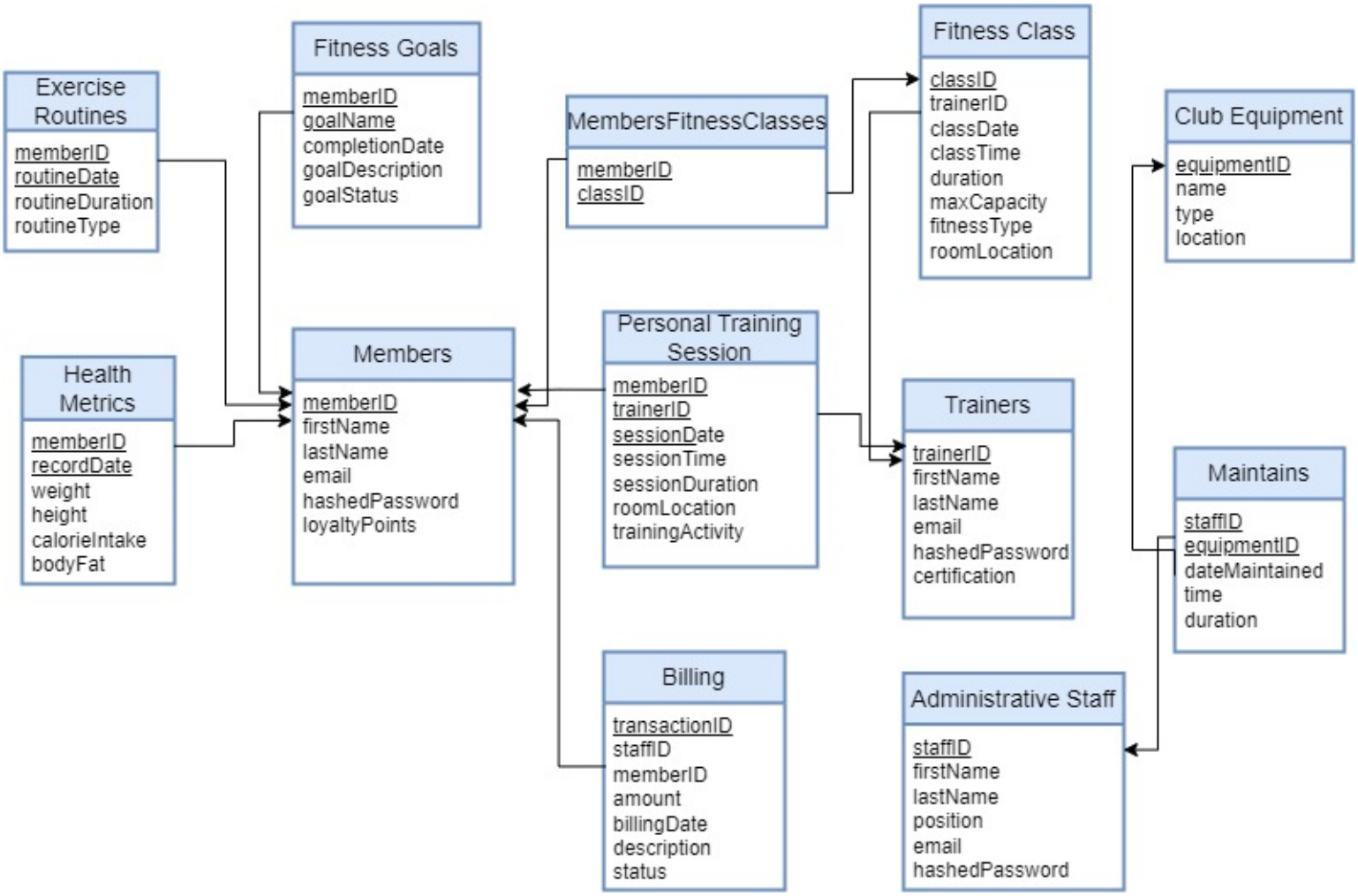
- A many-to-many relation between an administrative staff and the club resources
    - An admin staff can maintain many club resources and equipment and the club resources can be maintained by different staff on different days
- An admin staff may not maintain any club resources so this is partial participation
- Every club resource is maintained by at least one staff member, so this is total participation

Reduction to Relational Schema

To map the ER diagram to a Relational Schema, we go through the 7 steps of mapping:

- Mapping of Regular entities, weak entities, 1:1 relationships, 1:N relationships, M:N relationships, multivalued attributes, and finally N-ary relationships

After going through the 7 steps of ER to Relational Schema Mapping, we get the following schema.



Normalization and Functional Dependencies

We need to ensure that all the relations in our database are in 2NF and 3NF.

The first thing we do is check that they are all in 1NF. This means there are no multivalued or composite attributes, only single atomic values. All the attributes in the relational schema above are atomic values, so they meet the requirements for first normal form.

For a relation to be in 2NF, it must be in 1NF (verified above) and there should not be any partial dependencies on the primary key, all non-primary attributes should be fully dependent on the primary key.

For a relation to be in 3NF, it must be in 2NF and not have any transitive dependencies, meaning a non-prime attribute should depend on another non-prime attribute.

Member Relation

Using memberID as the primary key, our functional dependencies (FD) are:

- FD1: memberID → {firstName, lastName, DoB, email, hashedPassword}

Since all attributes depend on the primary key and there are no partial dependencies, it is in 2NF.
Since they all depend directly on the primary key, there are no transitive dependencies so it is in 3NF.

Trainer Relation

Using trainerID as the primary key, our functional dependencies are:

- FD1: trainerID → {firstName, lastName, email, hashedPassword, certification}

Since all attributes depend on the primary key and there are no partial dependencies, it is in 2NF.
Since they all depend directly on the primary key, there are no transitive dependencies so it is in 3NF.

Administrative Staff relation

Using staffID as the primary key, our functional dependencies are:

- FD1: staffID → {firstName, lastName, email, position, hashedPassword}

Since all attributes depend on the primary key and there are no partial dependencies, it is in 2NF.
Since they all depend directly on the primary key, there are no transitive dependencies so it is in 3NF.

Exercise Routines Relation

Using memberID and the routineDate as the primary keys, our functional dependencies are:

- FD1: {memberID, routineDate} → {routineDuration, routineType}

Since all attributes depend on the entire primary key and there are no partial dependencies, it is in 2NF.

Since they all depend directly on the primary keys, there are no transitive dependencies so it is in 3NF.

Fitness Goals Relation

Using memberID and the fitness goal name as the primary keys, our functional dependencies are:

- FD1: {memberID, goalName} → {completionDate, goalDescription, goalStatus}
- The goal name itself is not enough to identify the non-key attributes, so they depend on the entire composite key of memberID and goalName

Since all attributes depend on the entire primary key and there are no partial dependencies, it is in 2NF.

Since they all depend directly on the primary keys, there are no transitive dependencies so it is in 3NF.

Health Metrics Relation

Using memberID and the recordDate as the primary keys, our functional dependencies are:

- FD1: {memberID, recordDate} → {weight, height, caloricIntake, bodyFat}
- The non-key attributes depend on both the day it was recorded and the memberID to be uniquely identified, therefore depend on the entire composite key.

Since all attributes depend on the entire primary key and there are no partial dependencies, it is in 2NF.

Since they all depend directly on the primary keys, there are no transitive dependencies so it is in 3NF.

Personal Training Session Relation

Using memberID, trainerID and the session date as the primary keys, our functional dependencies are:

- FD1: {memberID, trainerID, sessionDate} → {time, duration, roomLocation, trainingActivity}
- The session date is not enough to identify a personal training sessions non-prime attributes, we also need to know the trainer and the member involved in the training

Since all attributes depend on the entire primary key and there are no partial dependencies, it is in 2NF.

Since they all depend directly on the primary keys, there are no transitive dependencies so it is in 3NF.

Fitness Class Relation

Using classID as the primary key, our functional dependencies are:

- FD1: classID → {trainerID, classDate, time, duration, maxCapacity, fitnessType, roomLocation}

Since all attributes depend on the primary key and there are no partial dependencies, it is in 2NF.

There is, however, a transitive dependency between the room location and max class size. The max class size attribute depends on the room location which is also a non-prime attribute.

- Transitive Dependency: {classID → roomLocation → maxCapacity }

So to make this in 3NF, we can separate the relation into two new ones where:
FitnessClass (classID, trainerID, classDate, time, duration, fitnessType, roomLocation)
RoomCapacity (roomLocation, maxCapacity)

MembersFitnessClasses Relation

Using memberID and the recordDate as the primary keys, and no other attributes, there are no functional dependencies making the relation in 2NF as well as 3NF.

Billing Relation

Using transactionID as the primary key, our functional dependencies are:

- FD1: transactionID → {staffID, memberID, amount, billingDate, description, status}

Since all attributes depend on the primary key and there are no partial dependencies, it is in 2NF.
Since they all depend directly on the primary key, there are no transitive dependencies so it is in 3NF.

Maintains Relation

Using staffID and itemID as the primary keys, our functional dependencies are:

- FD1: {staffID, itemID} → {dateMaintained, timeMaintained, duration}
- An assumption we make is that an item does not get maintained more than once on a given day

Since the attributes depend on the entire primary key and there are no partial dependencies, it's in 2NF.
Since it depends directly on the primary keys, there are no transitive dependencies so it is in 3NF.

Club Resources Relation

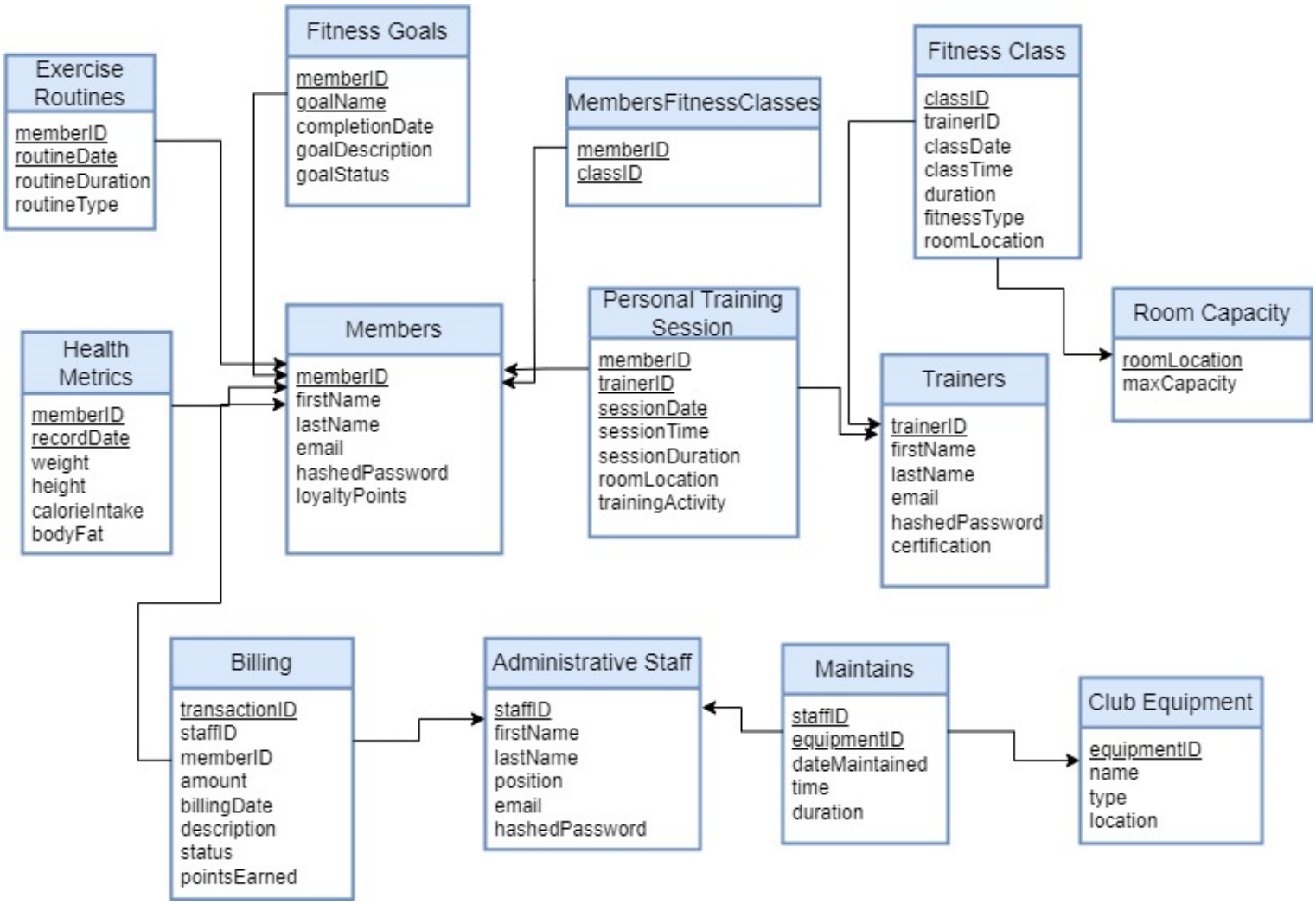
Using itemID as the primary key, our functional dependencies are:

- FD1: itemID → {name, quantity, location}

Since all attributes depend on the primary key and there are no partial dependencies, it is in 2NF.
Since they all depend directly on the primary key, there are no transitive dependencies so it is in 3NF.

Database Diagram

After going through our normalization process, and making sure that our diagram meets the requirements for 1NF (single atomic values), 2NF (no partial dependencies) and 3NF (no transitive dependencies) our final database schema diagram looks like:



DDL File and SQL command Execution

The link to the Github repository with the DDL file and SQL queries file is:
https://github.com/NarayanGautam/COMP3005A---Final-Project

The YouTube Video Demonstration of the project is also on the Github where I go over the project, set up the databases and execute the commands.