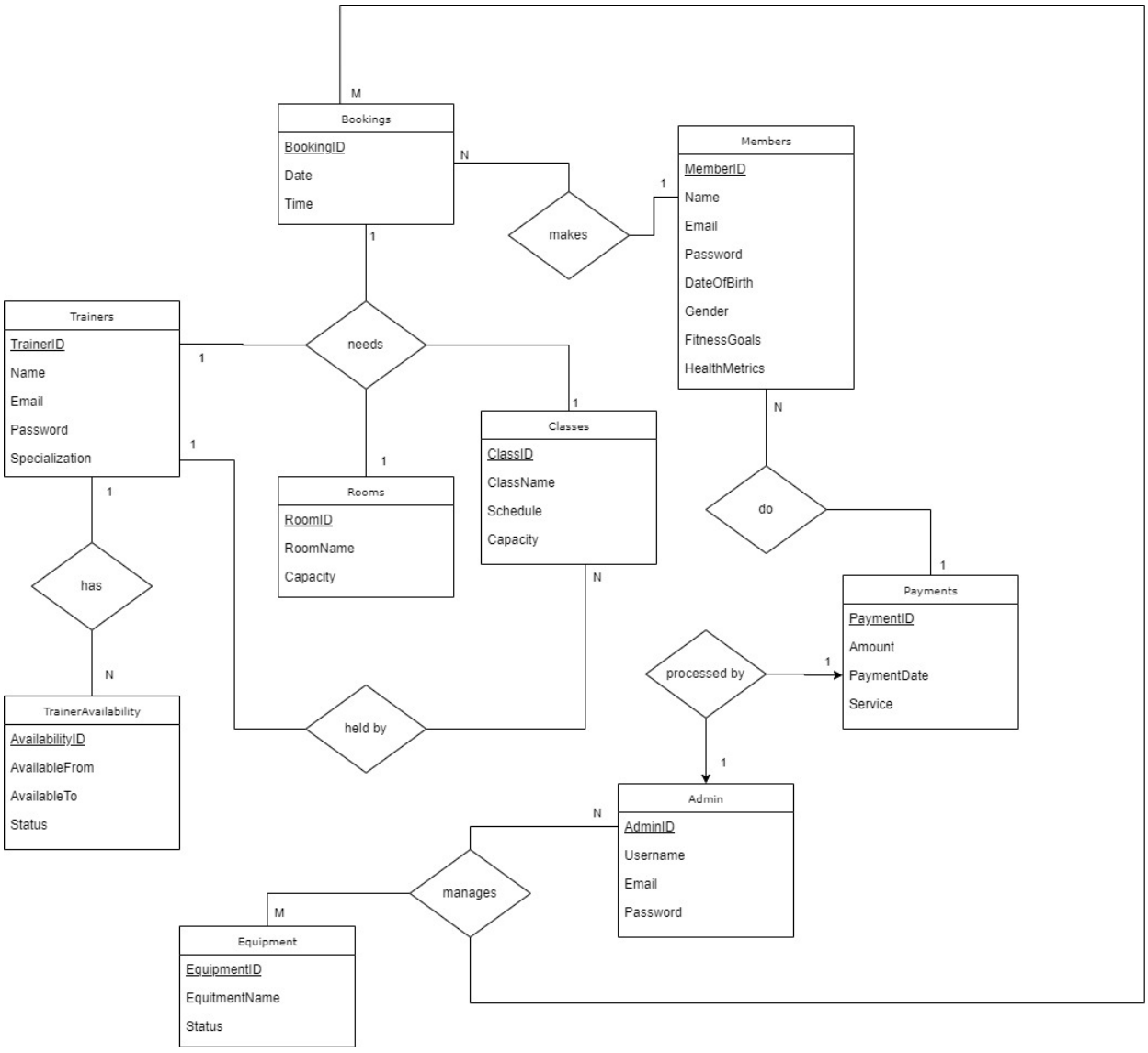
COMP3005 Final Project Report
Group Member1: Pak Yin, Kan, 101260592 (Preferred name: Knox Kan)
Group Member2: Muhammed Burhan, 101270831

Conceptual Design with ER Diagram:



The above ER diagram shows the database design for this Gym management program. This program is mainly used by three parties, namely, members, trainers as well as admins, and therefore, those are three main entities in this diagram.
For member, they have the ability to make bookings to various types of activities such as personal training and classes and have their responsibilities to make payment and as such this extends to another entities – “Payments”, “Bookings”,

“Classes” and “Room”. Those entities should store all necessary data input from the main entities – “Admin”, “Members” or “Trainer”.

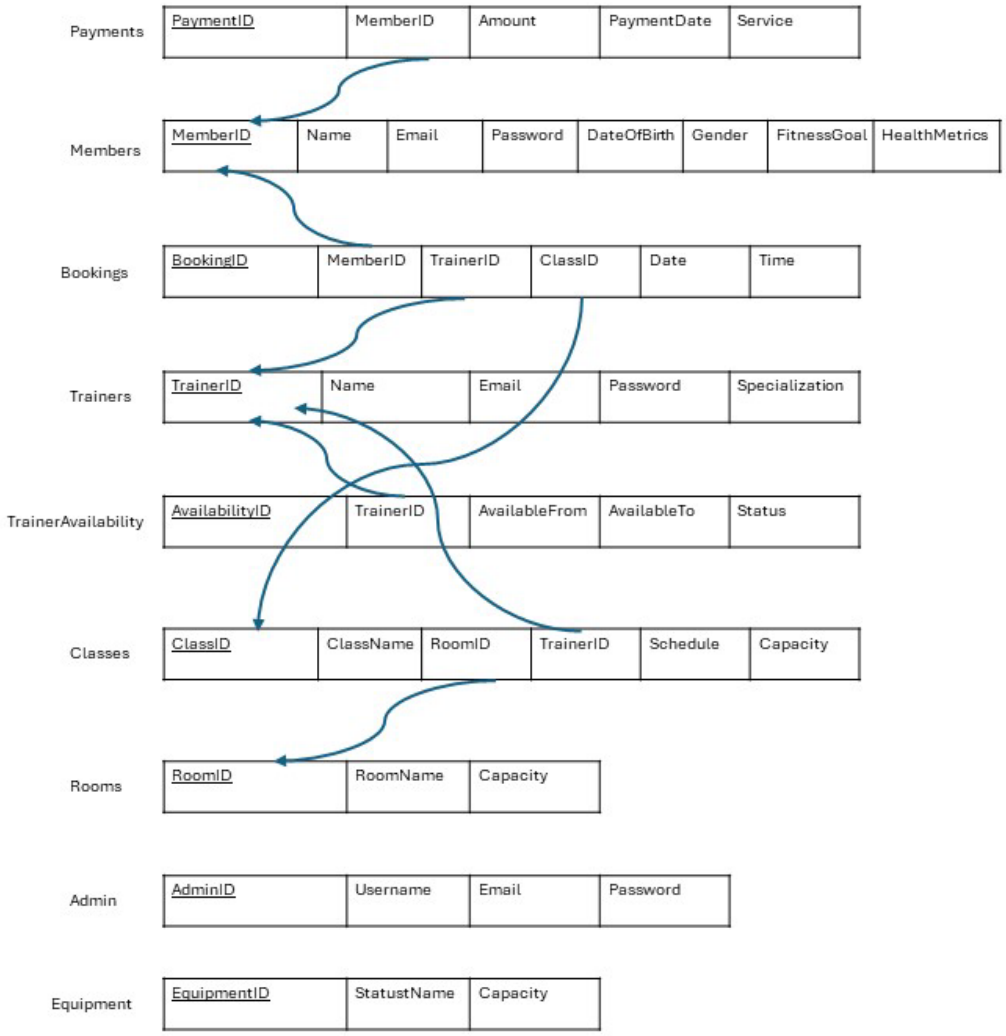
For the Trainer, since they should have the permission to enter their own availability for the personal training session, “TrainerAvailability” entity is created at this point for the trainer to input their availability as many as they can, with referencing to the TrainerID. In this approach, it prevents a ton of duplication record if we merged the availability to the Trainer table.

The “Admin” entity manages the current equipment of the Gym so it has access to the “Equipment” entity. The admin can also monitor and manage classes so they can also manipulate the “Classes” and “Rooms” tables.

Noted that, we acknowledge that when the program expands and especially when new requirements come, the schema needs to be adjusted, the above ER-diagram should cover the basic idea and requirements of this project.

Reduction to Relation Schemas:

With the ER diagram, we can come up with the following relation schemas:



The schema is pretty simple and can be easily understood by breaking down the ER diagram. Members, Trainers, Admin table represents three types of users, and the other tables serves for other functionalities.

DDL File:

Please check the “SQL” directory. In theory, you do not need to run this file as when starts running the program, it will create the tables for the users automatically, but it was provided in case if there is any issue.

Link to DDL file: https://github.com/knoxkpy/COMP3005-Project/blob/main/SQL/DDL.sql

DML File:

Please check the "SQL" directory. In theory, you do not need this file as well since if you follow the instructions of the program, the database should work properly. But this file serves as a "facilitator" for the demo.

Link to DML file: https://github.com/knoxkpy/COMP3005-Project/blob/main/SQL/DML.sql

Implementation:

The Health and Fitness Club Management System is designed to facilitate the management of club members, trainers, and administrative staff through a comprehensive platform. Developed using Python and interfaced with a PostgreSQL database via the psycopg library, this application efficiently handles the registration, scheduling, and tracking of fitness activities and member metrics.

1. Application:

- Presentation:
  - The user interface is primarily command-line-based, providing prompts and accepting inputs that allow users to interact with the system. This approach ensures simplicity and accessibility while maintaining the focus on functionality.
- Logic:
  - Core functionalities, including managing user profiles, scheduling classes, processing payments, and updating health metrics, are implemented in Python. This layer processes data, applies business rules, and handles decision-making processes.
- Data Access:
  - The psycopg library is employed to establish a connection between the Python application and the PostgreSQL database. It facilitates executing SQL queries, managing transactions, and handling database operations seamlessly.

2. Database Architecture:

- Database Schema:
  - The schema includes several key tables such as Members, Trainers, Classes, Admin, Bookings, and Payments etc. Relationships are defined with primary keys and foreign keys, ensuring relational integrity across the database. Please check the ER diagram and its

corresponding schema above for details

- Assumptions and Constraints
    - Health Metrics and JSON Data:
    - Health metrics are stored in a JSON format to allow flexibility in the data captured. Users are required to input JSON data correctly, and instructions of the format for the input is provided.
- Date and Time Handling:
    - Proper handling of date and time formats is crucial for the scheduling functionalities of the system. Users must adhere to specified formats to avoid errors in bookings and class schedules.
- Error Handling and Validation
    - Error Handling: The system is designed to gracefully handle errors such as database connectivity issues, transaction failures, and user input errors. This includes appropriate messaging to guide the user to correct actions or inputs.
- Input Validation:
    - Input validation is implemented to ensure that all user inputs are valid before they are processed or stored in the database. This prevents errors during database operations and enhances overall system robustness.
- Query Optimization:
    - SQL queries are optimized for performance, especially those involving joins and aggregations, to ensure the system remains responsive as the volume of data grows.

3. Future Enhancements:

- Potential enhancements include the development of a graphical user interface (GUI), integration of more complex health metrics analysis, and expansion of administrative functionalities.

The Health and Fitness Club Management System leverages a well-structured relational database and robust application architecture to provide a reliable and efficient platform for fitness management. While further enhancement can be made on other forms of GUI, current design would be sufficient to perform sufficient CURD operation.

Bonus Features:

Although we decided to use the command-line interface for our program, we added several other features.

- Registration (all members/admin staff/ trainers need to register for the system)
    - For registration, you can choose to register for a member, admin staff or as a trainer. But for trainers and admins, you need to have the invitation code in order to register. This prevents a random user registering the system as an admin or trainer.
- Authorization and authentication.
    - For different types of user (member, trainer, admin), they will have different privileges in this system. For example, the trainer can set their availability, but the member cannot. The member can join the classes but they cannot edit the details of the class and etc...
- Member checking their bills in the system.
    - This is not a required functionality in the specification, but now, once the member is logged in, they have the option to check their pending bills.
- Trainer can see the members detail who joined their classes and personal training sessions.
    - For Member Profile Viewing functionality in the trainer.py, it is only for viewing the member(s) who has joined the trainer's class or personal training session.
    - The original required "Search by Member's name" is moved to a new functionality called "Search Member Profile by Name" in the same trainer.py.

Collaboration Part:

Pak Yin, Kan (Knox Kan) (101260592):

- System design (overall, database)
- Login functionality (authentication. Authorization)
- ER diagram, ER mapping
- All member functions
- Trainer functions: Schedule Management (Trainer can set the time for which they are available.)
- Project report (this file)
- Demonstration

Muhammed Burhan (101270831):

- System design (Database)
- Trainer functions: Member Profile Viewing (Search by Member's name)
- All administrative Staff functions.
- DDL file preparation
- DML file preparation
- Demonstration

GitHub Repository:

https://github.com/knoxkpy/COMP3005-Project

Youtube Video Demo Link:

https://www.youtube.com/watch?v=aNKNAwOhG-s