COMP 3005 – Final Project Report

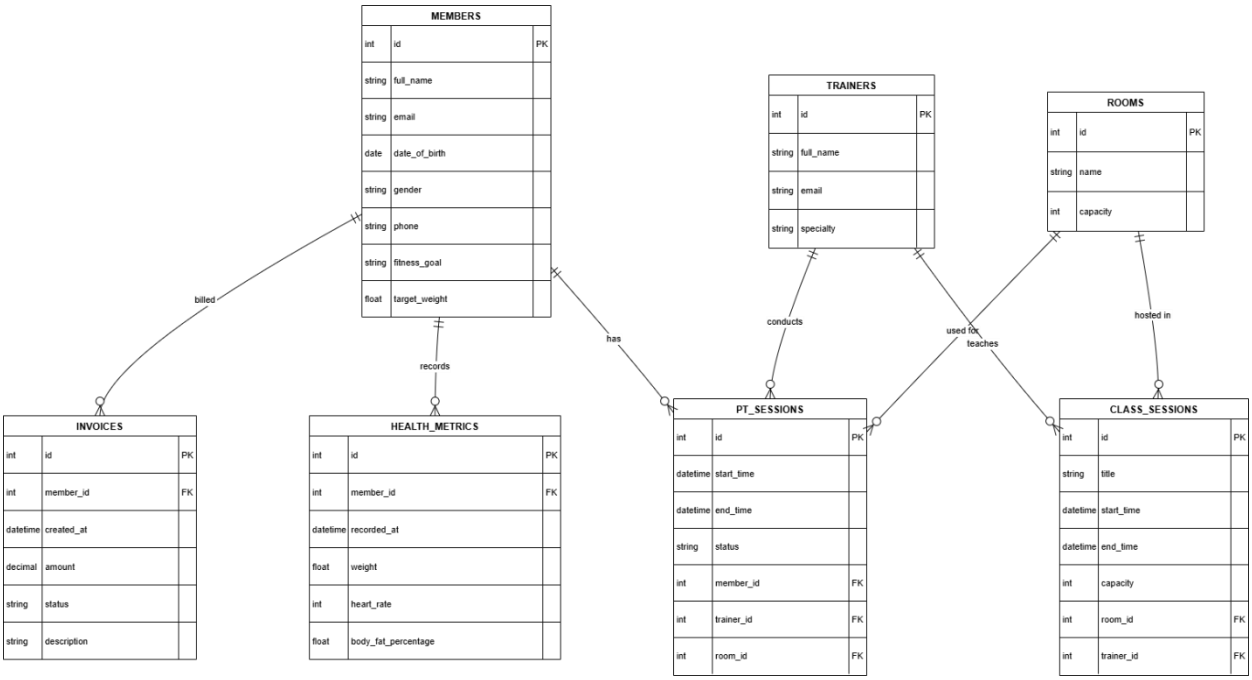
Health Club Database System
Abdulrahman Al-Dousari 101289389

1. Introduction

For this project, I built a small Health Club database system using PostgreSQL and Python. The idea was to create something realistic that a gym could use tracking members, trainers, rooms, sessions, health metrics, and invoices. I used SQLAlchemy for the Python side, so the code talks directly to the database.

Everything works together:
the database, the relationships, the Python app, and the advanced SQL parts (view, trigger, index).
This report explains the design and how everything works.

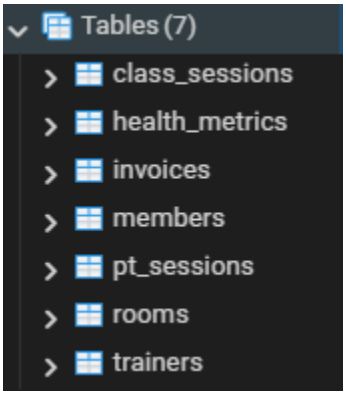
2. ER Diagram



The diagram shows all the main tables and how they connect.
Members link to PT sessions, health metrics, and invoices.
Trainer's link to classes and PT sessions.
Rooms host both classes and training sessions.

Overall, the diagram keeps the structure clean and easy to understand.

3. Tables and Schema



Here's a quick explanation of each table and what it stores:

Members

Basic information about every member.

- id (PK)
- full_name
- email
- date_of_birth
- gender
- phone
- fitness_goal
- target_weight

Trainers

Details about gym trainers.

- id (PK)
- full_name

- email
- specialty

Rooms

Every room in the club.

- id (PK)
- name
- capacity

PT Sessions

Personal training appointments.

- id (PK)
- member_id (FK)
- trainer_id (FK)
- room_id (FK)
- start_time
- end_time
- status

Class Sessions

Group classes.

- id (PK)
- title
- start_time
- end_time
- capacity
- room_id (FK)
- trainer_id (FK)

Health Metrics

Member progress tracking.

- id (PK)
- member_id (FK)
- recorded_at
- weight
- heart_rate
- body_fat_percentage

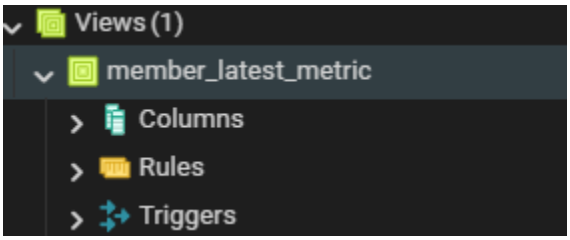
Invoices

Billing information.

- id (PK)
- member_id (FK)
- created_at
- amount
- status
- description

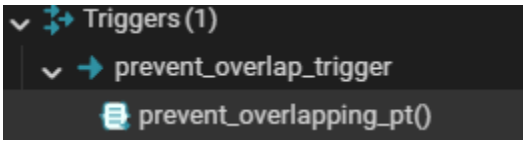
4. Advanced SQL Features

View



I made a view called member_latest_metric.
It shows each member with only their most recent health metric entry.
This makes checking someone's current weight or body fat way easier.

Trigger



I created a trigger called prevent_overlap_trigger.
It blocks any PT session that overlaps with another session for the same member.
If someone tries to double-book, the insert fails.
This protects the schedule.

Index

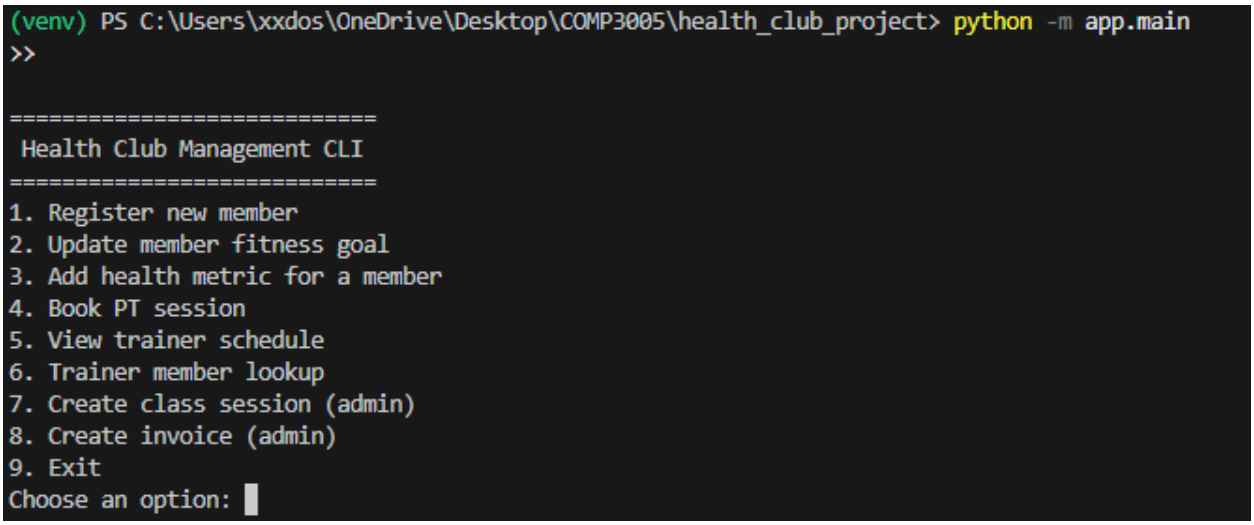
I added an index on pt_sessions(start_time) to speed up queries that sort or filter by session start time.
This is useful since staff often check upcoming sessions.

5. Python Application

The Python app uses:

- PostgreSQL
- SQLAlchemy
- A simple command-line menu

The app lets you:



- Add members
- List members
- Update a member's fitness goal / target weight
- Schedule PT sessions (the trigger prevents overlaps)
- Record health metrics
- View invoices
- Check the most recent health metrics via the view
- Seed the database with sample data

The project structure is simple and organized, and everything was tested directly in VS Code.

6. Testing

I tested the main features to make sure everything works:

- Adding members and trainers
- Scheduling PT sessions
- Trying to double-book (trigger worked)

- Adding health metrics and checking the view
- Confirming tables in pgAdmin
- Checking the index and trigger in the UI
- Running the full app through the terminal

Everything ran without issues.

7. Conclusion

This project helped me practice designing a full database system and connecting it with a working Python application. I also learned how to implement more advanced SQL features like views, triggers, and indexes. The system is complete, functional, and easy to extend if needed.