COMP 3005: Project "V2"

Matthew Czarnowski 101151676

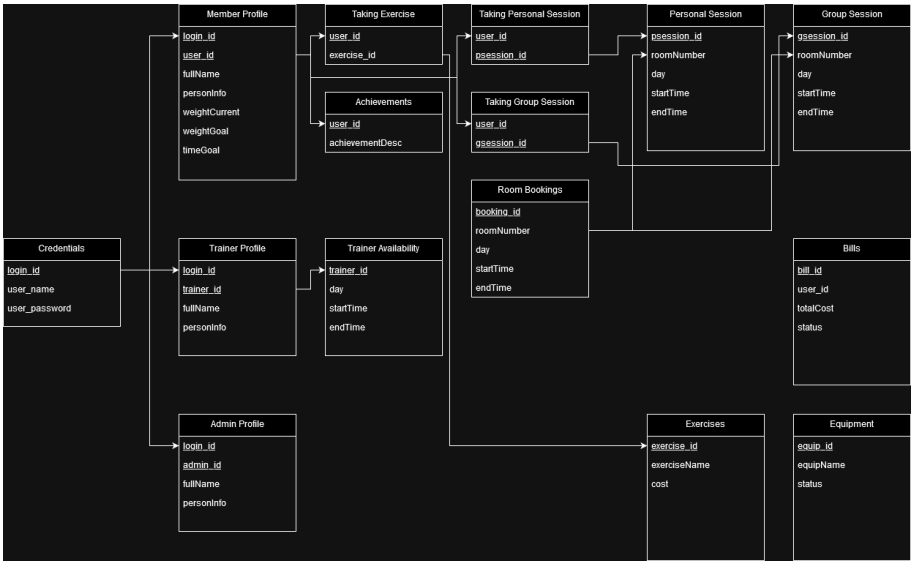
April 10, 2024

```
Important links:
Github repository: https://github.com/MattCzar13/Comp3005-ProjectV2
Video demonstration: https://youtu.be/bEkFzww-A84

If something isn't explained here, it's explained better either in the video
or in the repository readme.
```

1 Conceptual Design (ER Diagram)

The full image file can be found inside the public repository, in the "Diagrams" folder.

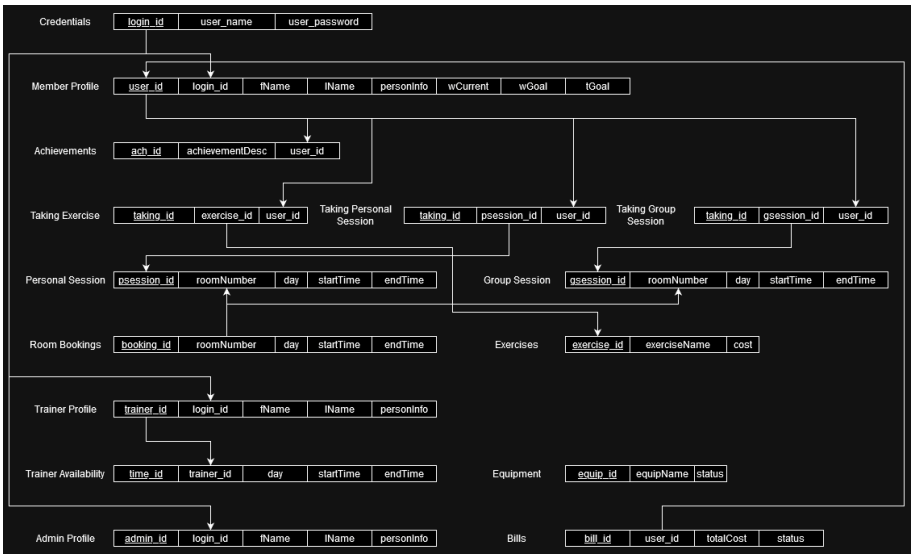


The ER diagram starts by separating the members, trainers, and admins into different user types with their own profiles. However, all of these users are connected to one credentials table, where each user has an ID connecting to their login username and password. The idea is to have all users in the club to login through the same portal pulling from the same table, and depending on what type of user has logged in, the interface will be different.
Regular members have their fitness goals tied to their profile, so it can be viewed easily by trainers. Their personal achievements are also kept in another table, tied to members by their ID. Members are tied to three "taking" tables, which keep track of the exercises, personal sessions, and group sessions they are a part of. These "taking" tables are also tied to the sessions themselves.
The sessions are their own tables, also connected to a "room bookings" table that keeps track of which rooms are booked, and at what time.
Trainers are a separate user from the regular members. They are connected to an "availability" table that shows when trainers are available. Trainers will be able to modify their own availability table entry. The

trainers will also be able to search the "member profile" table, which shows every member in the club. Since the member health and goal information is stored directly in their profile, trainers will easily be able to look at a member's information.
Admins are their own user as well. They will be able to look at and modify the "room bookings" table, the "equipment" table, and "group session" table. Admins will also be able to modify the "bills" table, which can connect to members.

2 Reduction to Relational Schemas (RDS Diagram)

The full image file can be found inside the public repository, in the "Diagrams" folder.



The relational schemas stay fairly close to the structure of the ER diagram. All of the proper connections between primary keys and foreign keys are much more visible here.

3 DDL file

```
CREATE TABLE Credentials (
login_id SERIAL PRIMARY KEY,
user_name VARCHAR(255) NOT NULL,
user_password VARCHAR(255) NOT NULL
);

CREATE TABLE MemberProfile (
user_id SERIAL PRIMARY KEY,
login_id INT,
FOREIGN KEY (login_id) REFERENCES Credentials (login_id),
first_name VARCHAR(255) NOT NULL,
last_name VARCHAR(255) NOT NULL,
personal_info TEXT,
weight_current INT,
weight_goal INT,
time_goal DATE
);

CREATE TABLE Achievements (
achievement_id SERIAL PRIMARY KEY,
achievement_desc TEXT,
user_id INT,
FOREIGN KEY (user_id) REFERENCES MemberProfile (user_id)
);

CREATE TABLE Exercises (
exercise_id SERIAL PRIMARY KEY,
exercise_name VARCHAR(255) NOT NULL,
cost INT
);

CREATE TABLE TakesExercise (
taking_id SERIAL PRIMARY KEY,
exercise_id INT,
FOREIGN KEY (exercise_id) REFERENCES Exercises (exercise_id),
user_id INT,
FOREIGN KEY (user_id) REFERENCES MemberProfile (user_id)
);

CREATE TABLE RoomBookings (
booking_id SERIAL PRIMARY KEY,
room_number INT UNIQUE NOT NULL,
time_day DATE,
time_start TIME,
time_end TIME
);

CREATE TABLE PersonalSession (
psession_id SERIAL PRIMARY KEY,
room_number INT,
FOREIGN KEY (room_number) REFERENCES RoomBookings (room_number) ON DELETE CASCADE,
time_day DATE,
```

```
time_start TIME,
time_end TIME
);

CREATE TABLE TakesPersonalSession (
taking_id SERIAL PRIMARY KEY,
psession_id INT,
FOREIGN KEY (psession_id) REFERENCES PersonalSession (psession_id) ON DELETE CASCADE,
user_id INT,
FOREIGN KEY (user_id) REFERENCES MemberProfile (user_id)
);

CREATE TABLE GroupSession (
gsession_id SERIAL PRIMARY KEY,
room_number INT,
FOREIGN KEY (room_number) REFERENCES RoomBookings (room_number) ON DELETE CASCADE,
time_day DATE,
time_start TIME,
time_end TIME
);

CREATE TABLE TakesGroupSession (
taking_id SERIAL PRIMARY KEY,
gsession_id INT,
FOREIGN KEY (gsession_id) REFERENCES GroupSession (gsession_id) ON DELETE CASCADE,
user_id INT,
FOREIGN KEY (user_id) REFERENCES MemberProfile (user_id)
);

CREATE TABLE TrainerProfile (
trainer_id SERIAL PRIMARY KEY,
login_id INT,
FOREIGN KEY (login_id) REFERENCES Credentials (login_id),
first_name VARCHAR(255) NOT NULL,
last_name VARCHAR(255) NOT NULL,
personal_info TEXT
);

CREATE TABLE TrainerAvailability (
time_id SERIAL PRIMARY KEY,
trainer_id INT,
FOREIGN KEY (trainer_id) REFERENCES TrainerProfile (trainer_id),
time_day DATE,
time_start TIME,
time_end TIME
);

CREATE TABLE AdminProfile (
admin_id SERIAL PRIMARY KEY,
login_id INT,
FOREIGN KEY (login_id) REFERENCES Credentials (login_id),
first_name VARCHAR(255) NOT NULL,
last_name VARCHAR(255) NOT NULL,
personal_info TEXT
```

```
);

CREATE TABLE Equipment (
equip_id SERIAL PRIMARY KEY,
equip_name VARCHAR(255) NOT NULL,
status TEXT
);

CREATE TABLE Bills (
bill_id SERIAL PRIMARY KEY,
user_id INT,
FOREIGN KEY (user_id) REFERENCES MemberProfile (user_id),
total_cost INT NOT NULL,
status TEXT
);
```

4 DML file

```
-- Login information (username, password)
INSERT INTO Credentials (user_name, user_password)
VALUES
('member', '1'),
('trainer', '1'),
('admin', '1'),
('bobman45', '1234'),
('johnworkout', '123abc'),
('jeremye', '985'),
('steveguy7', 'abcde'),
('trainer25', 'stuff');

INSERT INTO MemberProfile (login_id, first_name, last_name, personal_info, weight_current, weight_goal,
VALUES
(1, 'Member', 'Member', 'Basic member login', 160, 159, '2024-08-10'),
(4, 'Bob', 'Man', 'the bob man', 170, 160, '2024-07-02'),
(5, 'John', 'Mustard', 'John mustard', 240, 220, '2025-01-01'),
(6, 'Jeremy', 'Clifford', 'q', 180, 150, '2024-11-06'),
(7, 'Steve', 'Guy', 'Steve', 190, 180, '2024-01-12');

INSERT INTO Achievements (achievement_desc, user_id)
VALUES
('Did a backflip', 1),
('100 reps in a row', 2),
('Championship winner', 3),
('Team player', 4);

INSERT INTO Exercises (exercise_name, cost)
VALUES
('Large Machine reps (10)', 10),
('Professional squats (20)', 5),
('Running (20)', 2);

INSERT INTO TakesExercise (exercise_id, user_id)
VALUES
(1, 1),
(3, 5),
(2, 2),
(3, 3);

INSERT INTO RoomBookings (room_number, time_day, time_start, time_end)
VALUES
(101, '2024-05-05', '14:30:00', '16:30:00'),
(102, '2024-05-06', '12:30:00', '13:30:00'),
(104, '2024-05-07', '18:30:00', '19:00:00');

INSERT INTO PersonalSession (room_number, time_day, time_start, time_end)
VALUES
(101, '2024-05-05', '14:30:00', '16:30:00');

INSERT INTO TakesPersonalSession (psession_id, user_id)
VALUES
```

```
(1, 5);

INSERT INTO GroupSession (room_number, time_day, time_start, time_end)
VALUES
(102, '2024-05-06', '12:30:00', '13:30:00'),
(104, '2024-05-07', '18:30:00', '19:00:00');

INSERT INTO TakesGroupSession (gsession_id, user_id)
VALUES
(1, 1),
(1, 3),
(1, 4),
(2, 4);

INSERT INTO TrainerProfile (login_id, first_name, last_name, personal_info)
VALUES
(2, 'Trainer', 'Trainer', 'Basic trainer login'),
(8, 'Burt', 'Trainerman', 'i train');

INSERT INTO TrainerAvailability (trainer_id, time_day, time_start, time_end)
VALUES
(1, '2024-01-01', '10:00:00', '18:00:00');

INSERT INTO AdminProfile (login_id, first_name, last_name, personal_info)
VALUES
(3, 'Admin', 'Admin', 'Basic admin login');

INSERT INTO Equipment (equip_name, status)
VALUES
('bike', 'operational'),
('weights', 'operational'),
('the machine', 'broken');

INSERT INTO Bills (user_id, total_cost, status)
VALUES
(5, 100, 'unpaid');
```

5 Implementation

To create my interface, I used Python, as I am most familiar with that language. The driver I used to connect with my PostgreSQL database is psycopg. This is a Python based PostgreSQL driver.
The version of Python that I used for development was Python 3.12 (64-bit), and I installed the newest version of psycopg by following the instructions on their website.
The interface is a CLI, so to use the program, you can run either CMD or PowerShell in the folder containing the main.py file, and running the file with Python.
The video goes into much more detail on how the program works. The video is linked in the public repository.