Database Management Systems - Final Project Report

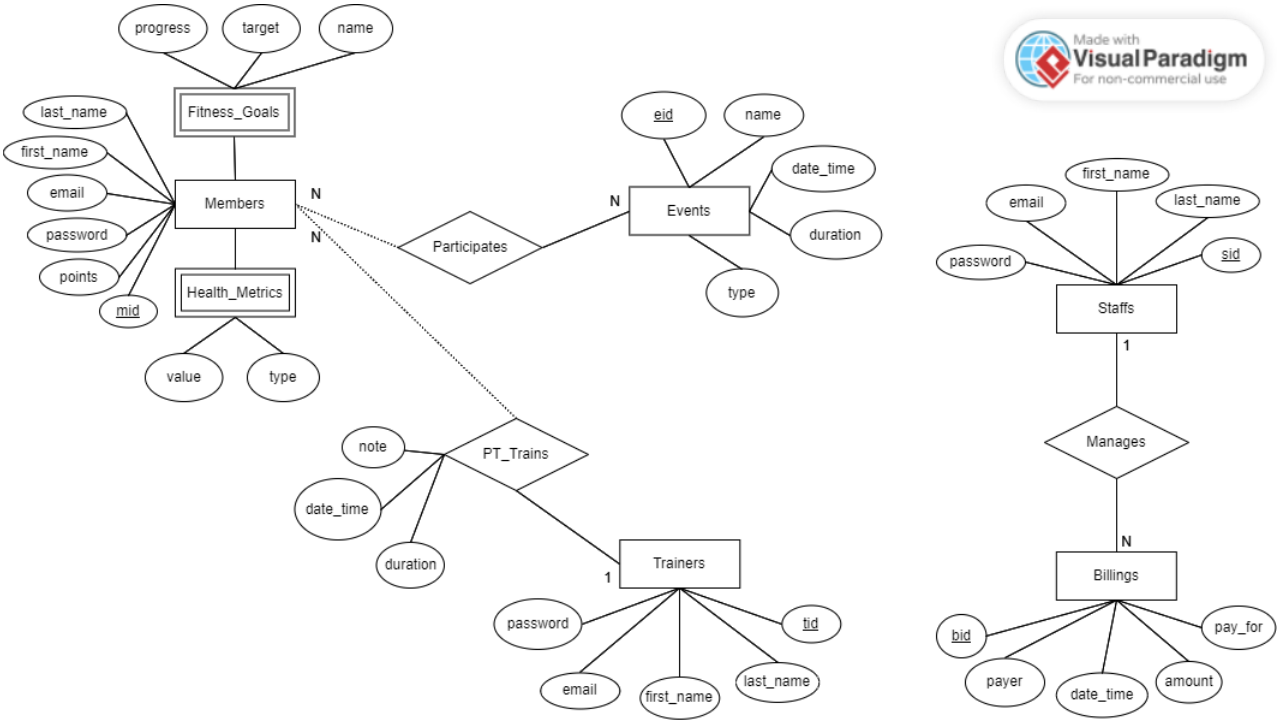
Charles Nguyen - `101184017`

December 10th, 2023

Conceptual Design

The following requirements are chosen from the given problem statement and are satisfied by the project:

- Members
  - register and manage their profile, including fitness goals and health metrics in addition to typical profile information (name, email, etc...)
  - register and participate in events, such as offered gym classes, hosted workshops, competitions and challenges etc...
  - schedule for personal training (PT) sessions with a trainer
- Trainers
  - schedule for personal training (PT) sessions with the members
  - view the members' profiles, including their health metrics and fitness goals
- Staffs
  - manage and process the billings such as membership fees, PT fees, operational costs, salaries, etc...

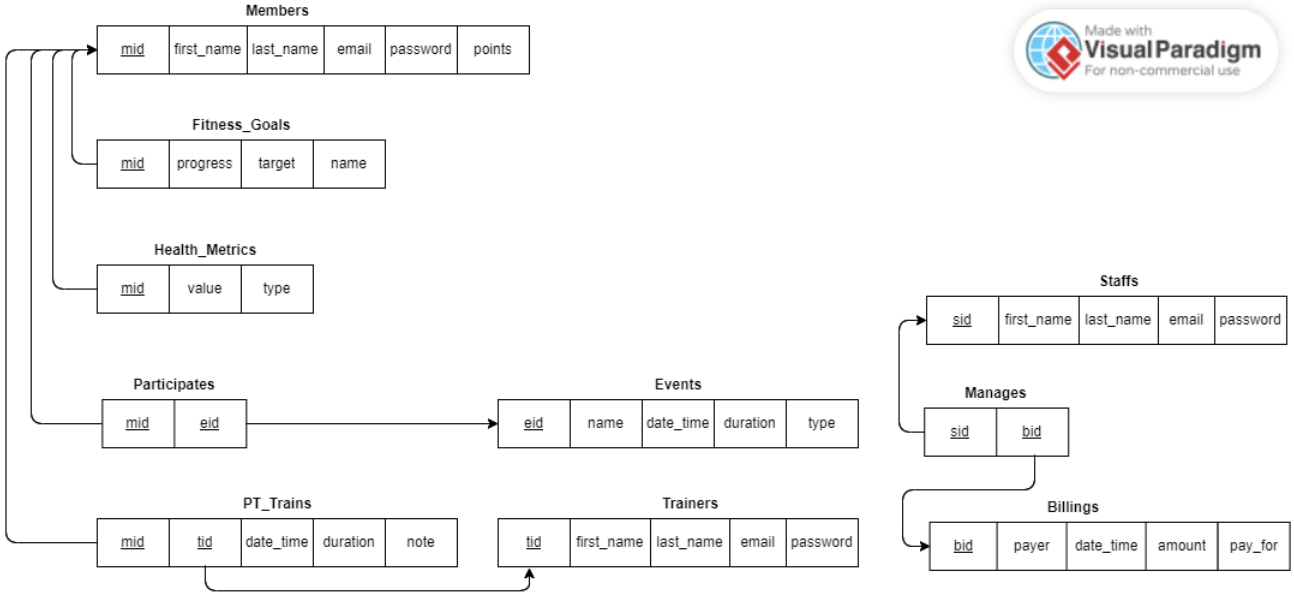


For members, they can choose to participate in the available events and opt in to schedule a PT session. Thus the relationship is optional for the members regarding the events and PT sessions. Note that a member can choose to participate in multiple events and an events can be hosted for multiple members. A member can only have a PT session scheduled with 1 trainer, but a trainer can have multiple PT sessions scheduled with multiple members as their customers.

For trainers however, if a member request to schedule a PT session then the trainer must be available to accept the request, providing that an agreements is made between the two parties regarding schedule. Thus, the relationship is mandatory for the trainers regarding the PT sessions. Similarly, for the Events entity, once a member registers for an event, the event must be available for the member to participate in. Thus, the relationship is mandatory for the Events entity regarding the members.

For staffs, they are responsible for managing the billings and thus the relationship is mandatory for the Staffs entity. Additionally, a staff can be tasked to handle multiple bills, but it makes sense that a bill can only be handled by 1 staff.

Reduction to Relational Schemas



Normalization of Relational Schemas

- Members
  1NF: The primary key is `mid` and the attributes are atomic.
  `mid` $\rightarrow$ `first_name, last_name, email, password, points`
  2NF: The primary key is `mid` and the attributes are fully functional dependent on primary key.
  `mid` $\rightarrow$ `first_name, last_name, email, password, points`

- Fitness Goals
  1NF: The primary/foreign key is `mid` and the attributes are atomic.
  `mid` $\rightarrow$ `progress, target, name`
  2NF: The primary/foreign key is `mid` and the attributes are fully functional dependent on primary key.
  `mid` $\rightarrow$ `progress, target, name`

- Health Metrics
  1NF: The primary/foreign key is `mid` and the attributes are atomic.
  `mid` $\rightarrow$ `value, type`
  2NF: The primary/foreign key is `mid` and the attributes are fully functional dependent on primary key.
  `mid` $\rightarrow$ `value, type`

- Participates
  The relation is introduced to handle many-to-many relationship between Members and Events with the least duplicate and NULL data possible, thus achieving **1NF** and **2NF**.

- Events
  1NF: The primary key is `eid` and the attributes are atomic.
  `eid` → `name, date_time, duration, type`
  2NF: The primary key is `eid` and the attributes are fully functional dependent on primary key.
  `eid` → `name, date_time, duration, type`

- PT Trains
  The relation is introduced to handle many-to-one relationship between Members and Trainers with the least duplicate and NULL data possible, thus achieving **1NF** and **2NF**.

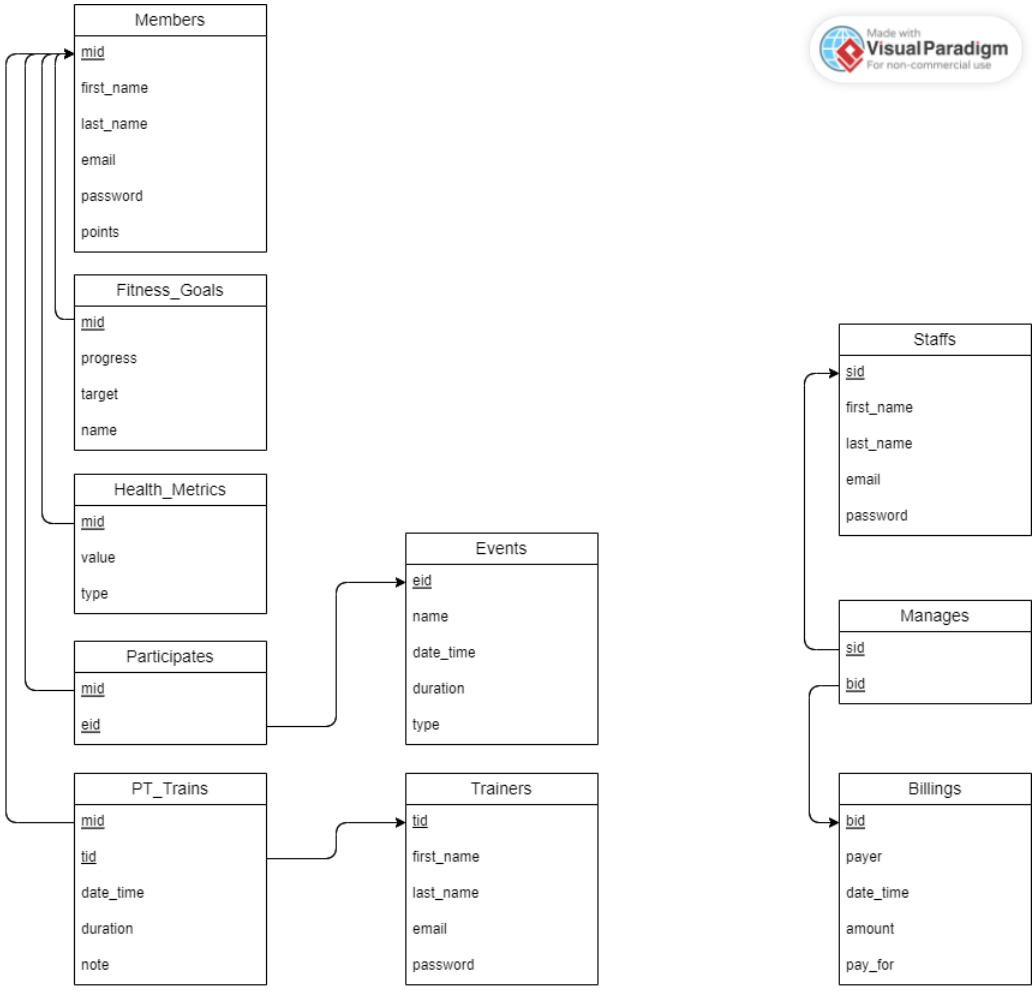
- Trainers
  1NF: The primary key is `tid` and the attributes are atomic.
  `tid` → `first_name, last_name, email, password`
  2NF: The primary key is `tid` and the attributes are fully functional dependent on primary key.
  `tid` → `first_name, last_name, email, password`

- Staffs
  1NF: The primary key is `sid` and the attributes are atomic.
  `sid` → `first_name, last_name, email, password`
  2NF: The primary key is `sid` and the attributes are fully functional dependent on primary key.
  `sid` → `first_name, last_name, email, password`

- Manages
  The relation is introduced to handle many-to-one relationship between Staffs and Billings with the least duplicate and NULL data possible, thus achieving **1NF** and **2NF**.

- Billings
  1NF: The primary key is `bid` and the attributes are atomic.
  `bid` → `payer, date_time, amount, pay_for`
  2NF: The primary key is `bid` and the attributes are fully functional dependent on primary key.
  `bid` → `payer, date_time, amount, pay_for`

Database Schema Diagram



Implementation Considerations (Bonus)

Even though the project is not implemented, SQL queries are provided (in GitHub repo) to demonstrate the use of the database with the above design. Note that some attribute names have been modified so they do not conflict with the reserved SQL keywords. The logic, however, remains exactly the same.

In addition, when implementing the database one should note the security aspects of the system. In particular, the `password` attribute should be hashed and salted before storing in the database.

The Members table is accessible by members, trainers and staffs, i.e. trainers should be given the privilege to read the table (in order to implement viewing their customer profiles) whereas both members and staffs should be given the privilege to read and write to the

table (in order to update their profile/goals/metrics and suspend the account if necessary, repectively).

Similarly, read and write access privilege is given to trainers and staffs for the Trainers as well as Events table, and read and write access privilege is strictly given to staffs only for the Staffs as well as Billings table.

GitHub Repository

`https://github.com/charlespnh/hfcms`