PROJECT FINAL REPORT

Health and Fitness database

Author: Lucía Caldado Rein

Table of contents

1. README

For our application , it is necessary to have these items:

a. MySQL server with 'health_and_fitness_databaseSchema' installed.
b. pymysql library for Python to connect to MySQL, use this command:
   pip install pymysql

The steps for running the application will be:

1) Open the terminal and go to directory counting python file with application code.
2) Use this command to run application: python applicationcode_project_caldadol.py
3) Enter username and password
4) If the credentials are correct, you will have to choose between
   4.1. Login with user id
   4.2. Create a new account, it will return user id after creating
5) After login, you will have to choose one option from the menu
   a) Option 1. Add new goal
   b) Option 2. Track health test
   c) Option 3. Track meal
   d) Option 4. Add workout
   e) Option 5. Edit workout
6) The menu will be displayed again until you decide to close the session answering the question with 'No' .

2. Technical Specifications

The database is SQL-based, created using MySQL Workbench.
The database is formed by the following entities: user, goal , health_test, meal, workout and exercise. Exercise has subclasses cardio and strength.
There exists these relationships :

- health_test evaluate user (health_test-user)
- user wants to achieve goal (user-goal)
- meal helps user to achieve goal (meal-user-goal)
- user practices workout (user-workout)
- workout divided in exercises (workout-exercise)

Also the database has the following stored procedures:

- add_goal, add new goal to table 'goal'
- add_exercise , add new exercise to table 'exercise'
- add_workout, add new workout to table 'workout'
- create_user , create a new user if it doesn't exist already
- edit_workout , update table workout
- new_exercise_in_workout , add new relationship between an exercise and a workout
- track_health_test , add new health test into health_test table
- track_meal, add new meal into meal table

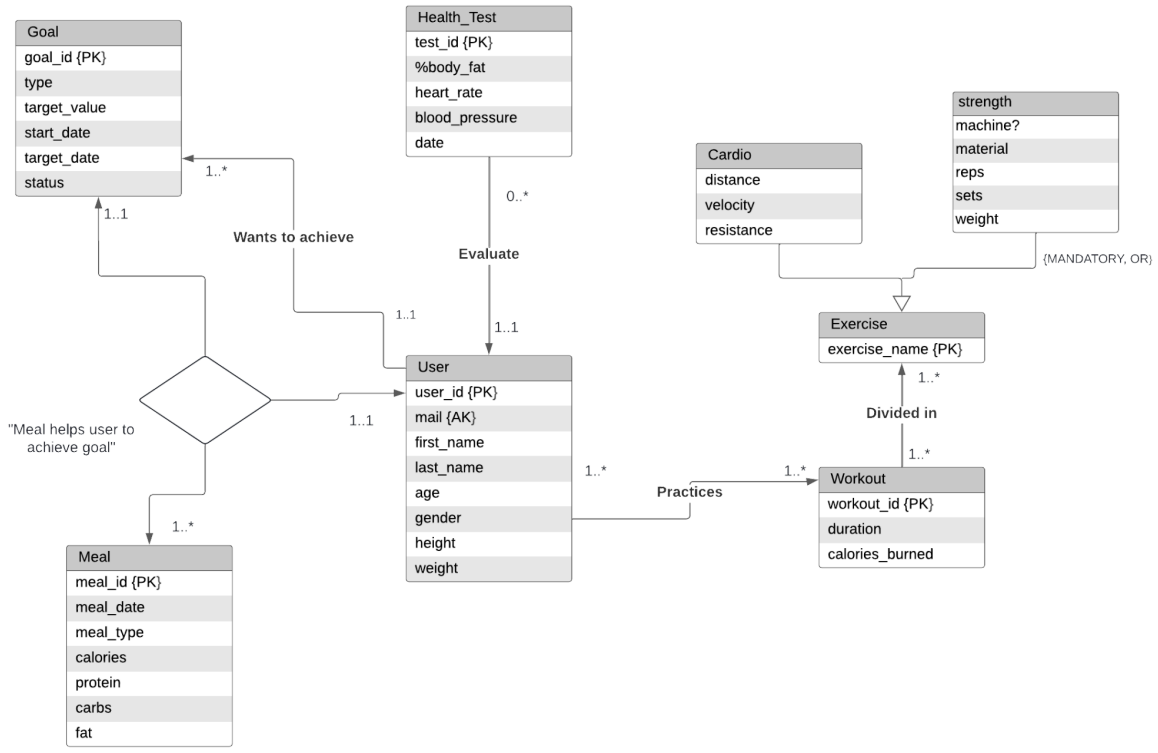
The following function: check_valid_id() that returns True if id is stored in the user table and False otherwise.
And the following triggers:

- before_update_workout , trigger to no update workout if new values for duration and calories burned are the same as the old ones
- before_insert_exercise, it checks if exercise is already related to workout before adding to relationship table 'workout_exercise'

I use Python to facilitate communication between the user and the database, acting as an intermediary on their behalf. The interface is terminal-based, meaning users can interact with the system by entering commands directly into the terminal. Python will process these inputs, convert them into SQL queries, and send them to the database for execution.

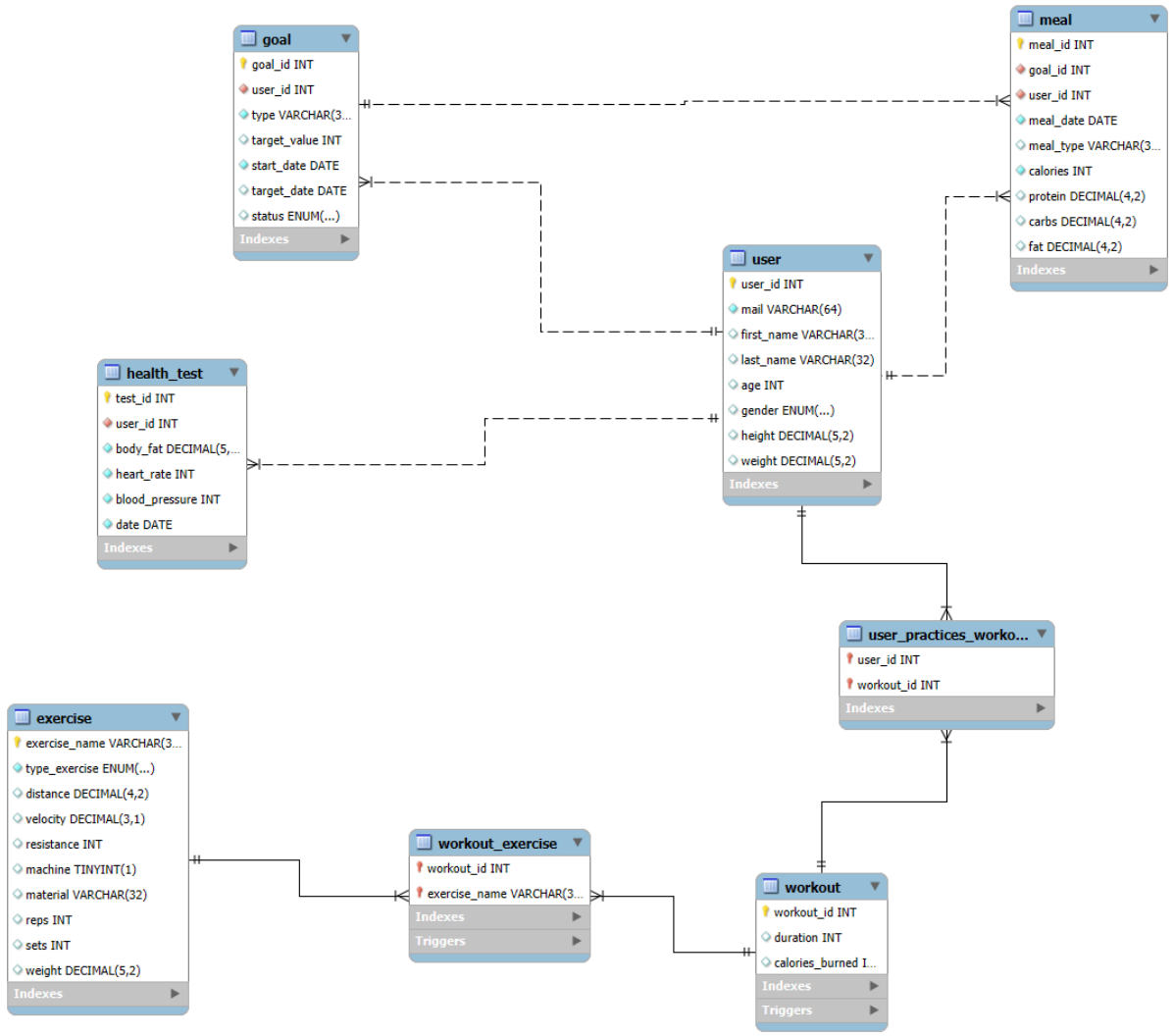
3. Conceptual Design

In this Health and Fitness database, we track the users that want to get a healthier life. The attributes for each user are their first and last name, their age, gender, height and weight. Each user is identified by its user_id. Users have 1 to many goals they want to achieve . These goals will be identified with an id . Each goal has a type (e.g.: weight loss, endurance, muscle gain), a target value , the start and target date, and its status (e.g.: in progress, finished).
An user will have a set of workouts, each workout can be done for many users. For the workouts, we track its workout_id , its duration and the calories that we burned doing it. Each workout has a list of exercises, each exercise can be done in different workouts. We look for the exercise's name. Exercises can be classified in cardio or strength. A cardio exercise has these attributes : the distance covered, velocity and resistance. A strength exercise tracks if it is done in a machine or not, if it is used some kind of material (e.g.: dumbbells, weights...) , the number of repetitions , the number of sets and the total weight used .
Also a user can have different meal plans where each meal has a meal id , a date , type of meal (e.g.: dinner, lunch, breakfast) , the number of calories, the grams of protein , carbs and fats. Each meal will be associated with one specific user depending on its goals.
To track their improvements, users can take health tests that track their body fat percentage , heart rate and blood pressure. Each test has an id that identifies it and the date when it was made.

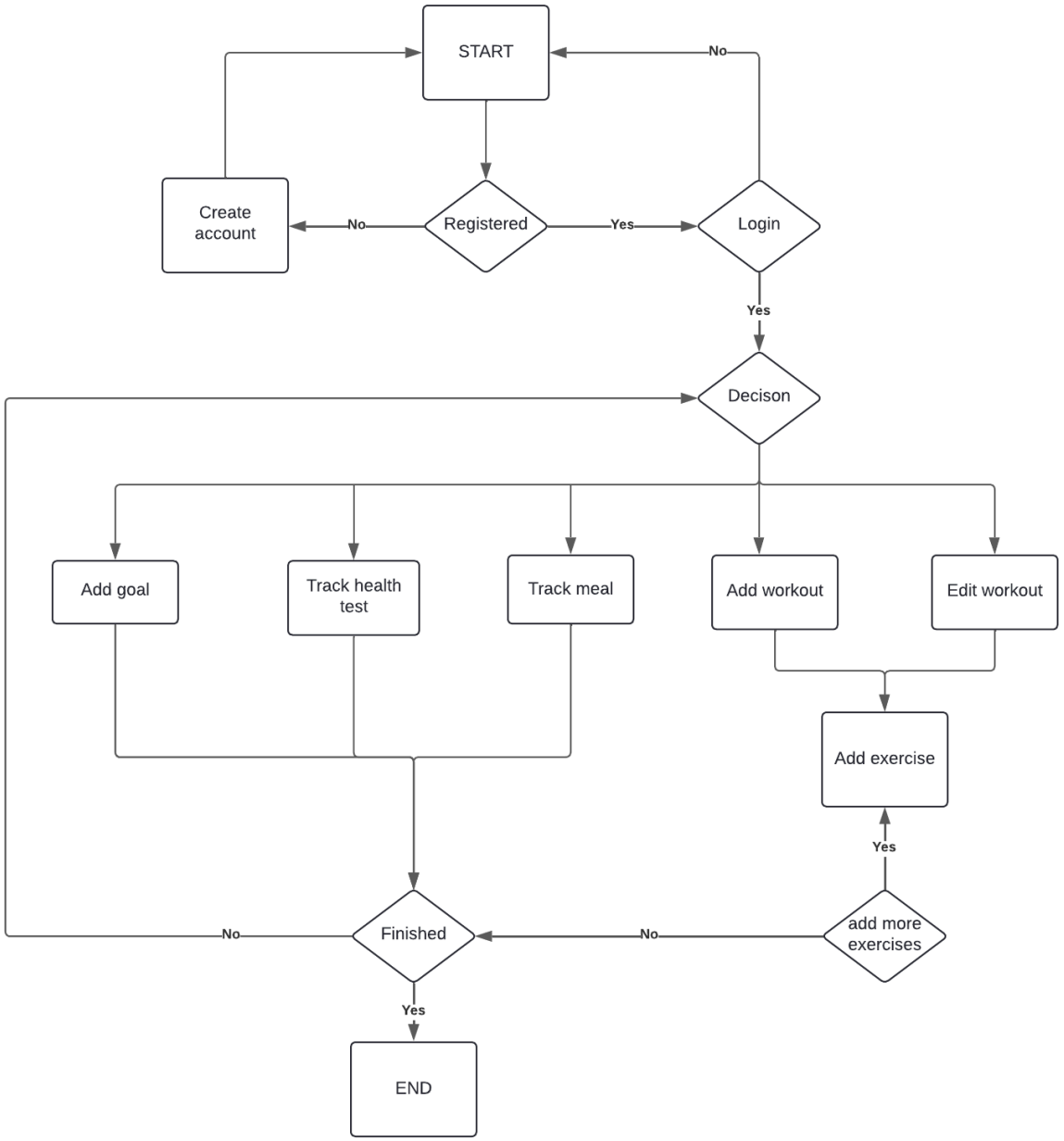


4. Logical Design

Using reverse Engineering:



5. User Flow of the system



An user can perform the following actions to interact with database:

- Create an account, it will use stored procedure ‘create_user’
- Add goal , it will use stored procedure ‘add_goal’
- Track health test using stored procedure ‘track_health_test’
- Track meal using stored procedure ‘track_meal’
- Add workout using stored procedure ‘add_workout’
- Edit workout using stored procedure ‘edit_workout’
- Add new exercises to workout using stored procedures ‘add_exercise’ and ‘new_exercise_in_workout’.

6. Lessons Learned

- Understand how to properly open, use, and close database connections in Python
- Learn how to enforce relationships between tables using foreign keys to ensure data integrity
- Understand how to use constraints as AUTO_INCREMENT
- Learn to write triggers , stored procedures and functions to automate database tasks
- Learn to call sql-statements or procedures from database application in Python
- Understand how to handle errors using try and except to manage MySQL exceptions or errors

7. Future Work

- Track personal health metrics
- Track and visualize progress toward fitness goals
- Track meal plans to achieve goals
- Create personalized workouts
- Help personal trainers to track, manage, and analyze client data efficiently, providing personalized guidance and progress monitoring.
- It can be integrated to another third-party applications and platforms