Projecto Base de Dados

Modulo de Fisiologista – Acompanhamento

João Tiago Machado Barroso

ISEP

Licenciatura em Engenharia Biomédica

Abstract

O objetivo desta base de dados é apenas servir de plataforma de armazenamento de dados para uma web app de agendamento de avaliações físicas e como tal funções de procura serão realizadas pelo projeto de "*back-end*" ainda por realizar. Para este efeito foi usado *MySQL* para criar esta estrutura de vital importância ao projeto global.

Estrutura da Base de Dados

A base de dados foi projetada com algumas redundâncias de forma a permitir adição de futuras funcionalidades que venham a ser necessárias (i.e., área do cliente na qual possa ver as suas avaliações, resultados e marcações), e dar margem de manobra durante a implementação de funções durante o desenvolvimento do "*back-end*".

A base de dados foi estruturada de forma que qualquer individuo com o devido acesso possa de forma rápida e eficaz obter a informação relevante ao pedido, sem informações desnecessárias. O melhor exemplo disto encontra-se na tabela "*m4assessment*", que permite ao utilizador obter somente os resultados de uma certa avaliação com uma "query":

```
#Query exemplar de uma procura especifica, para o cliente 1 e data 27 de Abril de 2024
SELECT * FROM m4assessment WHERE ((client_id = 1) AND (assessment_date = STR_TO_DATE('27.04.2024', '%d.%c.%Y')));
```

E caso o utilizador requeira os exercícios específicos realizados numa certa sessão poderá obtê-los com

```
SELECT * FROM ExerciseDetails WHERE assessment_id = 1;
```

Como referido anteriormente, de forma a fazer as tabelas mais legíveis certas funcionalidades foram relegadas para outras partes do projecto, uma das quais, a contagem de sessões com um cliente uma vez que tal função pode ser alcançada com o comando:

```
SELECT COUNT(*) FROM m4assessment WHERE client_id=1;
```

Funcionalidade e Descrições

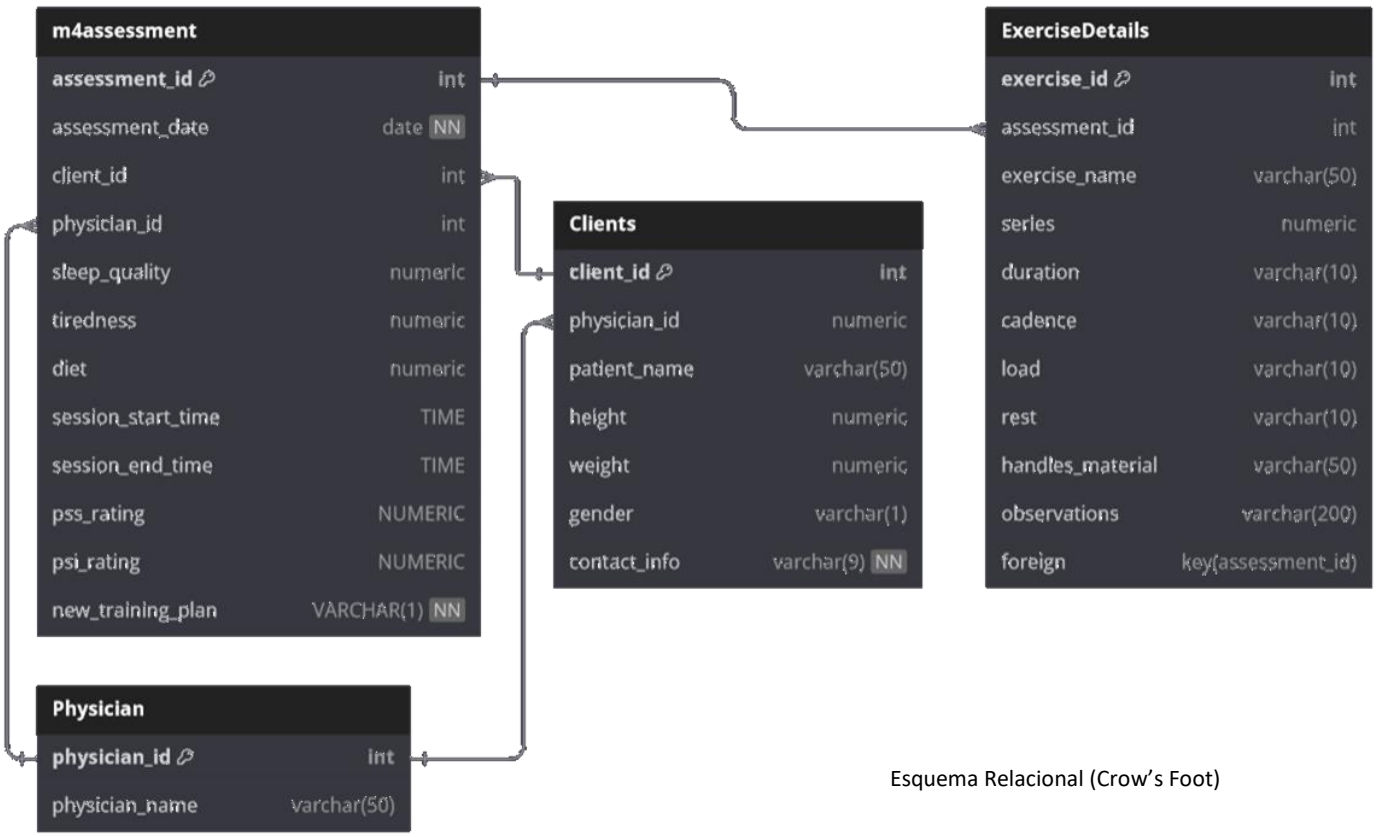
O esquema da base de dados foi criado de forma que qualquer ponto de partida consiga chegar a qualquer tipo de informação, como tal as relações estado definidas de forma que qualquer utilizador tenha acesso a toda a informação que lhe corresponde.

A escolha de estruturar a base de dados à volta da tabela Fisiologistas ("Physician") permite uma maior proteção dos dados dos clientes (relação one-to-many com as tabelas clientes e m4assessment), uma vez que apenas o profissional atribuído ao cliente terá acesso à sua informação. Ainda na tabela de fisiologistas, é relevante realçar que isto permite uma fácil e rápida integração com outras bases de dados (nomeadamente no projecto paralelo da página de login), apesar de tais medidas ainda não estarem implementadas.

Relativamente à tabela cliente, apesar de não ser pedido decidi adicionar algumas colunas que possam ser úteis para a monitorização do estado de saúde do cliente. Esta tabela tem uma relação one-to-many com a tabela m4assessment, dada natureza recorrente da última (um cliente irá realizar várias avaliações ao longo da sua monitorização).

A tabela m4assessment corresponde à avaliação M4 em si (como o nome indica), estruturada de forma a requerer tanto um fisiologista como um cliente, apresenta uma redundância ao nível da "*foreign key*" physician_id, uma vez que a última já serve de chave externa para a tabela cliente. Isto foi feito deliberadamente de forma a minimizar perdas de dados em caso de corrupção da informação, permitindo então que as avaliações possam ser recriadas através de uma composição das tabelas "Clients" e "Physician". Esta tabela armazena o grosso da informação relevante a uma avaliação (as perguntas pré-treino, tempo de início e fim, as avaliações PSS e PSI e se o cliente precisa de um novo plano). Apresenta uma relação one-to-many com a tabela "ExerciseDetails" de forma que a englobar todos os exercícios realizados durante uma avaliação.

Finalmente, a tabela "ExerciseDetails" corresponde a cada exercício de uma avaliação e contém a informação sobre esse exercício individual.



Esquema Relacional (Crow's Foot)

Conclusão

Apesar de cumprir o objetivo requerido, ainda há áreas que podem ser expandidas e ou melhoradas, dependendo da evolução do projecto.

Analisando a evolução do projecto nesta primeira parte, verifiquei que em certas partes tomei decisões demasiado ambiciosas (como tentar implementar “triggers” de forma a minimizar possíveis erros humanos) que resultaram em setbacks e perdas de tempo desnecessárias, e que acabaram por ser desnecessárias com uma melhor divisão de funções entre os vários níveis de full-stack development.