BD Projeto: Club Manager Pro

Grupo: P10G8

- Ricardo Martins, MEC: 112876 – 60%
- Diogo Martins, MEC: 108548 – 40%

Conteúdo

Introdução

Neste trabalho abordamos o desafio de tentar projetar e mais tarde de implementar um sistema de gestão de clubes e atletas, um AMS. que para os treinadores e administradores iria permitir fazer uma melhor gestão dos membros de um clube, incluindo atletas, treinadores, administradores e sócios, e das informações de cada um, bem como permitir aos treinadores introduzir, guardar e visualizar as estatísticas de todos os seus atletas muito mais facilmente, podendo também fazer comparações entre os vários atletas que tem no presente e até comparar com os atletas que já treinou no passado, para conseguir perceber melhor a evolução de cada atleta e se os planos de treinos estão a ser ou não eficazes para o atleta em questão.

Já para os atletas seria também uma oportunidade para também conseguirem mais facilmente observar as suas estatísticas ao longo do tempo e mesmo comparar com as outras pessoas do clube de forma a conseguirem ter uma perceção da sua evolução ao longo do tempo e se os seus resultados têm sido bons ou não e representativos do esforço que coloca nos treinos.

A escolha para o desenvolvimento deste projeto foi motivada pelo meu interesse e participação no mundo do atletismo, e a necessidade de um sistema de gestão de clubes e atletas que permitisse uma melhor organização e análise dos dados dos atletas e das competições em que participam.

Análise de Requisitos

1. Utilizadores

- Gestão de utilizadores: Permitir o registo e gestão de contas de utilizadores, incluindo atletas, treinadores, administradores, e sócios.

- Registo de Utilizadores: Captura de dados pessoais e criação de contas com verificação por email.

- Autenticação de Utilizadores: Login utilizando email e senha.

- Atualização de dados pessoais: Permitir aos utilizadores atualizar informações pessoais e acrescentar novos dados como imagem, salário, altura, peso, e cartão de cidadão.

- Gestão de Clubes: Funcionalidade para criar clubes ou entrar noutros existentes, com gestão de papéis (atleta, treinador, etc.) e status (ativo, inativo, pendente).

- Restrições Específicas: Prevenção de criação de contas duplicadas ou fraudulentas através de confirmação por email. E colocar um limite bem definido sobre o número de clubes que um utilizador pode criar para evitar abusos na criação e eliminação dos mesmos.

2. Clube

Facilitar a gestão de clubes de atletismo ao, gerir os membros, estatísticas dos atletas, competições a participar, fazendo uso das seguintes funcionalidades:

- **Gestão de Membros:** Aprovar ou recusar solicitações de entrada, gerir papéis e status dos membros.

- **Definição dos papeis e permissões**: Definir quais as permissões dos admins e treinadores de cada clube para que fique definido quais as mudanças e capacidades que cada um vai ter dentro do clube em relação a quais tabelas podem modificar, ao por exemplo introduzir dados sobre uma competição de um atleta que lhes pertença

- **Sistema de Convites:** Envio e gestão de convites para novos membros com definição de papéis.

- **Controles de Segurança:** Implementar restrições e validações para proteger contra ações mal-intencionadas ou erros.

3. Estatísticas de Atletas

- **Alteração dos dados:** Apenas permitir que o atleta faça modificações nos valares das tabelas que lhe correspondem, mas permitir aos treinadores correspondentes e aos admins alterar os valores de qualquer das tabelas dos atletas que treinam dentro do clube

- **Comparação entre Atletas:** Permitir comparações de desempenho entre atletas do mesmo clube ou com o seu próprio histórico.

4. Requisitos Transversais:

- **Alteração dos dados:** Permitir que em todas as tabelas o utilizador seja capaz de realizar as operações de inserção, remoção, alteração e visualização dos dados aos quais tem permissão de aceder.

- **Validação de Dados:** Garantir a integridade e precisão dos dados inseridos com campos obrigatórios, formatos específicos para dados numéricos, etc.

- **Segurança e Privacidade:** Proteger os dados dos utilizadores e do clube através de práticas robustas de segurança.

Estrutura do diretório e Indicações para o Projeto

O nosso diretório APFT_112876_108548 está dividido em 4 subdiretórios:

1. Diagrams_Relatório: Com os PNGs dos 3 diagramas que elaboramos para o nosso projeto junto com o ficheiro docx e pdf do nosso relatório.

2. Video_PP_Apresentacao: com o PowerPoint e vídeo que apresentámos no dia 31 maio.

3. ClubManagerPro: Contém os ficheiros para correr o programa no Visual Studio.

4. Ficheiros_SQL: Contém todos os ficheiros SQL utilizados no desenvolvimento do projeto separados em ficheiros individuais e ainda num ficheiro completo que engloba todas as operações dos outros ficheiros para criar a base de dados de raiz com todas a informação.

5. Pasta ClubManagerPro: Contém os ficheiros para correr o programa, incluindo todos os ficheiros de código e de design de Visual Studio.

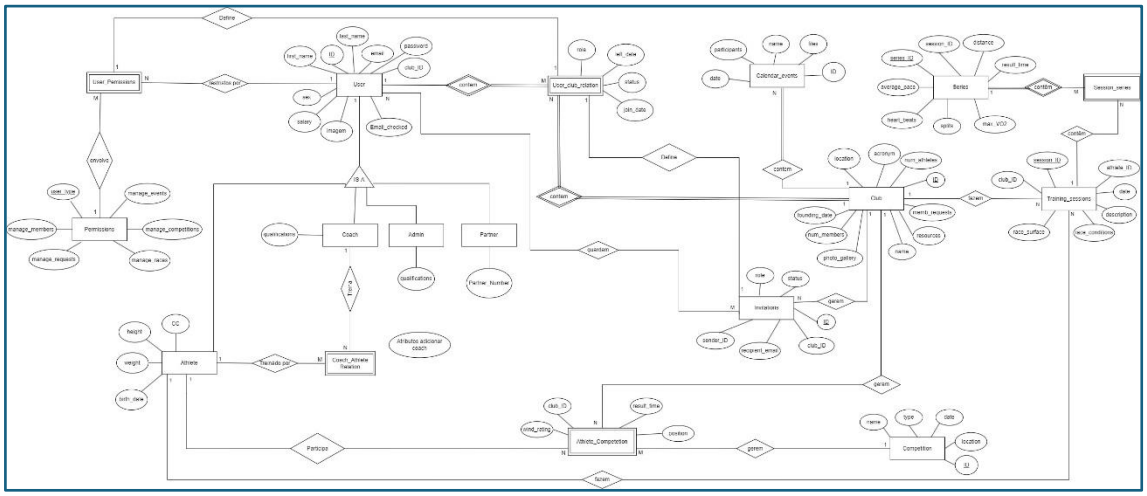
COMO CORRER O CÓDIGO

Para correr o código na base de dados global deve-se utilizar a primeira SQLConnection com data source para o SQL Server. Se decidir utilizar a base de dados local deve-se colocar “data source = ProfDesktop“ com o seu valor do desktop e no catalog igual ao nome da base de dados local, sendo esta Project se for criada usando os nossos ficheiros SQL. Deve-se mudar esta SQLConnection nos 5 ficheiros .cs.

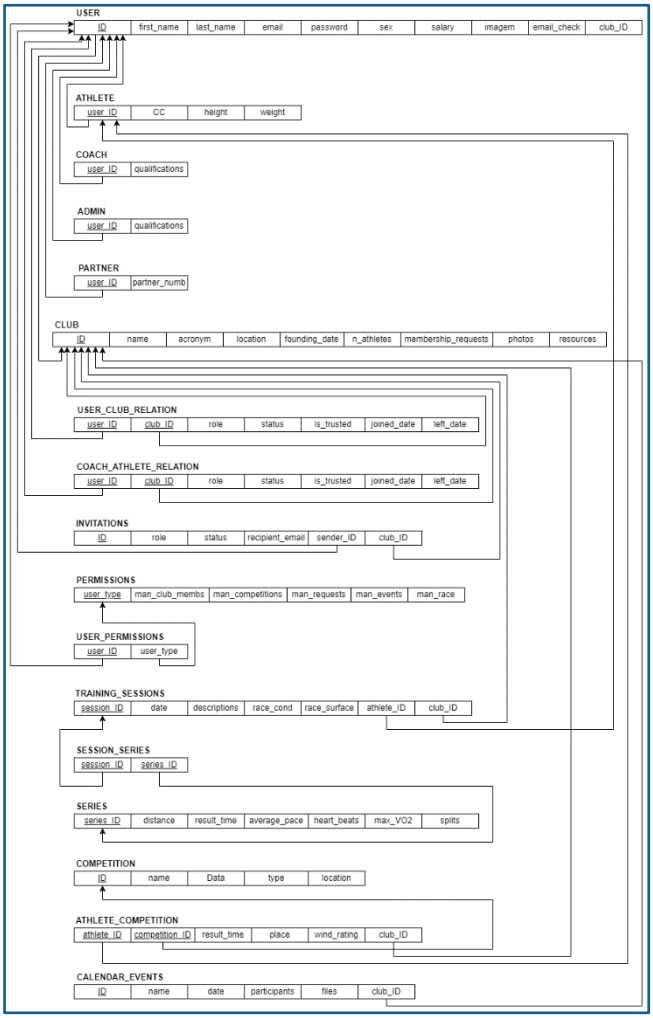
Entidades do projeto

Diagramas

DER



ER



SQL Code

Para a criação da base de dados, tabelas, índices, procedimentos armazenados, triggers e funções, foi utilizado o SQL Server Management Studio (SSMS) para executar os scripts SQL. Primeiro os scripts foram testados localmente e depois foram implementados no servidor de base de dados das aulas para a entrega final.

Os ficheiros são os seguintes:

• **RecreateEntireDatabase** – ficheiro SQL que recria completamente a base de dados localmente;

• **Creates_ProjectDatabase** – ficheiro SQL que cria a base de dados localmente com o schema AMS;

• **SQL_DDL** – ficheiro SQL para fazer a criação e a destruição de tabelas, bem como a definição das suas Primary keys, Foreign Keys e checks básicos dos valores nelas introduzidos;

• **ValuesInsert** – ficheiro SQL para inserir valores para dentro das tabelas;

• **UDF** – ficheiro SQL para criar as User Defined Functions usadas no projeto;

• **Views** – ficheiro SQL criar Views usadas no projeto;

• **SP** – ficheiro SQL para criar as Stored Procedures usadas no projeto de forma a garantir a consistência dos dados quando se faz operações nas tabelas;

• **TRIGGERS** – ficheiro SQL para criar os triggers usados no projeto;

• **IndexesCreate** – ficheiro SQL para criar os índices usados no projeto;

SQL_DDL - Data Definition Language

Seguindo os diagramas anteriormente estabelecidos foram criadas 17 tabelas SQL para armazenar os dados do sistema, justo com as respetivas chaves primárias, estrangeiras e restrições de integridade referencial. Foram utilizados os comandos habituais de SQL *CREATE TABLE* e *ALTER TABLE* para definir as tabelas e as relações entre elas, bem como *DROP TABLE* para eliminar tabelas existentes.

1. AMS.Club: Tabela principal para armazenar os dados dos clubes, onde os atletas e treinadores estão associados.

2. AMS.[User]: Tabela principal para armazenar as informações dos utilizadores.

3. AMS.Athlete: Tabela para armazenar informações de atletas, um tipo de conta para utilizadores.

4. AMS.Coach: Tabela reservada às informações de treinadores, outro tipo de conta para utilizadores.

5. AMS.Admin: Tabela para armazenar os dados de administradores, um tipo de conta reservado aos gestores de clube.

6. AMS.Partner: Tabela para armazenar informações de parceiros, para os utilizadores que são parceiros do clube.

7. AMS.Coach_Athlete: Tabela de relação entre treinadores e atletas, associa os treinadores aos atletas que treinam.

8. AMS.User_club_relation: Tabela de relação entre utilizadores e clubes, define se o utilizador é confiável e qual o seu papel.

9. AMS.Invitations: Tabela para armazenar convites enviados para novos membros, indicando o papel e o estado do convite.

10. AMS.Permissions: Tabela para armazenar as permissões de todo o tipo de utilizadores.

11. AMS.User_permissions: Tabela de relação entre utilizadores e permissões, relaciona as permissões a cada utilizador.

12. AMS.Training_sessions: Tabela para armazenar informações sobre as sessões de treino, que podem englobar várias séries.

13. AMS.Series: Tabela para armazenar informações sobre as séries de treino, que podem ser realizados pelos atletas.

14. AMS.Session_series: Tabela de relação entre sessões de treino e séries, associa as séries a cada sessão.

15. AMS.Competition: Tabela para armazenar informações sobre competições, onde os atletas podem participar.

16. AMS.Athlete_Competition: Tabela de relação entre atletas e competições, onde o posicionamento dos atletas nas competições é armazenado.

17. AMS.Calendar_events: Tabela para armazenar eventos no calendário, como competições e sessões de treino.

InsertValues - Data Manipulation Language (DML)

Para inserir os dados em cada tabela, foram utilizados os comandos SQL *INSERT INTO* para adicionar registos às tabelas. Os dados foram inseridos manualmente para testar a funcionalidade das tabelas e garantir que as suas relações funcionavam corretamente.

Importante mencionar alguns casos especiais, na tabela *AMS.[User]* as palavras-passe foram encriptadas para garantir a segurança dos dados dos utilizadores.

Stored Procedures

- AMS.DeleteUserAndRelatedTables

O procedimento armazenado AMS.UserAndRelatedTables é uma função genérica criada para eliminar registos de um utilizador e tabela relacionadas. Este procedimento é particularmente útil para manter a integridade referencial dos dados ao eliminar registos de uma base de dados relacional.

1. Parâmetros de Entrada:

- @UserID: ID do utilizador a ser eliminado.

2. Processo de Eliminação:

O procedimento começa por eliminar todas as tabelas baseadas em associações do utilizador (Session_series). De seguida parte para as tabelas específicas ao tipo do utilizador e finalmente, caso não tenha ocorrido nenhum erro, o utilizador é eliminado da sua tabela e a função termina.

3. Tratamento de Erros:

O procedimento inclui um bloco TRY...CATCH para capturar qualquer exceção que ocorra durante a execução dos comandos SQL. No caso de uma exceção, a função é quebrada e uma mensagem de erro detalhada é gerada e levantada.

- AMS.DeleteClubAndRelatedTables

O procedimento armazenado AMS.DeleteClubAndRelatedTables é utilizado para eliminar um clube da base de dados, assim como todos os registos relacionados. Abaixo está uma descrição concisa do procedimento:

1. Parâmetros de Entrada:

- @ClubID: ID do clube a ser eliminado.

2. Processo de Eliminação:

O procedimento começa por eliminar todas as tabelas às quais o clube está dependente, e caso suceda parte para a tabela dos Users onde coloca o seu clube como NULL. Finalmente, se tudo for sucedido apaga o clube da sua própria tabela.

3. Tratamento de Erros:

Semelhante ao anterior, utiliza um bloco TRY...CATCH para capturar exceções. Em caso de erro, a transação é anulada (ROLLBACK). Levanta uma mensagem de erro personalizada com RAISERROR se ocorrer um erro. Este procedimento assegura a eliminação segura e consistente de utilizadores e dos respetivos dados relacionados na base de dados.

Triggers

AMS.trg_UpdateNumOfAthletes_Insert

Este trigger é utilizado para atualizar o número de atletas num clube quando um novo atleta é inserido na tabela AMS.Athlete. Abaixo está uma explicação concisa para inclusão no relatório:

Nome: AMS.trg_UpdateNumOfAthletes_Insert

Objetivo: Incrementar o número de atletas no clube ao adicionar um novo atleta.

Ação: Após a inserção de um novo registo na tabela AMS.Athlete, este trigger incrementa a coluna num_of_athletes na tabela AMS.Club para refletir a adição de um novo atleta ao clube correspondente.

AMS.trg_UpdateNumOfAthletes_Delete

Este trigger é utilizado para atualizar o número de atletas num clube quando um atleta é eliminado da tabela AMS.Athlete. Abaixo está uma explicação concisa para inclusão no relatório:

Nome: AMS.trg_UpdateNumOfAthletes_Delete

Objetivo: Decrementar o número de atletas no clube ao eliminar um atleta.

Ação: Após a eliminação de um registo na tabela AMS.Athlete, este trigger decrementa a coluna num_of_athletes na tabela AMS.Club para refletir a remoção de um atleta do clube correspondente.

UDF's

Foram criadas 2 funções para auxiliar na validação de dados e na obtenção de informações específicas da base de dados.

AMS.fnGetUserType

Esta função devolve o tipo de utilizador com base no ID do utilizador fornecido. A função aceita o parâmetro de entrada @UserID e devolve o tipo de utilizador (Athlete, Coach, Admin, Partner) associado ao utilizador. A função é útil para determinar o tipo de utilizador ao executar operações específicas para cada tipo.

Todo o procedimento é feito com declarações IF...ELSE para verificar o tipo de utilizador. Se o ID do utilizador for igual ao ID do:

- Atleta da tabela AMS.Athlete, a função devolve *'athlete'*;
- Treinador da tabela AMS.Coach, a função devolve *'coach'*;
- Administrador da tabela AMS.Admin, a função devolve *'admin'*;
- Parceiro da tabela AMS.Partner, a função devolve *'partner'*.

Se o ID do utilizador não corresponder a nenhum dos tipos, a função devolve *'unknown'*.

isUserAdmin

Esta função verifica se um utilizador é um administrador com base no ID do utilizador fornecido. A função aceita o parâmetro de entrada @UserID e devolve um valor booleano (1 ou 0) para indicar se o utilizador é admin. A função é útil para restringir operações específicas apenas aos admins.

O procedimento é feito com duas declarações IF...ELSE para verificar se o ID do utilizador corresponde a um admin na tabela AMS.Admin. Se o ID do utilizador for igual ao ID da tabela:

- AMS.User_club_relation, a função devolve 1;
- AMS.User_permissions, a função devolve 1;

Se o ID do utilizador não corresponder a nenhum, a função devolve 0.

Views

Foram criadas 5 views para facilitar a visualização dos dados armazenados nas tabelas, permitindo aos utilizadores executar consultas complexas de forma mais simples e eficiente. Outra vantagem do uso de views no nosso projeto foi a capacidade de separar informação que certos utilizadores não devem ver e mostrar apenas a informação que lhes é relevante.

AMS.vwUsersAndClubs

Esta view combina os dados das tabelas AMS.[User] e AMS.Club para mostrar informações detalhadas sobre os utilizadores e os clubes a que pertencem. A view aparece em formato de tabela na página principal da aplicação, permitindo ao utilizador ver rapidamente os detalhes dos utilizadores e dos clubes.

AMS.vwTrainingSessionsAndSeries

Esta view combina os dados das tabelas AMS.Session_series, AMS.Series, AMS.[User] e AMS.Club para mostrar informações sobre as sessões de treino e suas séries associadas de cada utilizador. A view é utilizada na página principal da aplicação.

AMS.vwCompetitionsAndAthletes

Esta view combina os dados das tabelas AMS.Competition e AMS.Athlete_Competition para mostrar os dados sobre as competições e os atletas que participam nelas. A view é também utilizada na página principal da aplicação, permitindo ao utilizador ver imediatamente os detalhes das competições.

AMS.vwClubsAndEvents

Esta view combina os dados das tabelas AMS.Club e AMS.Calendar_events para mostrar informações sobre os eventos de cada clube irá ter. A view é utilizada na página principal da aplicação, permitindo ao utilizador ver onde os seus clubes irão participar.

AMS.vwUsersAndPermissions

Esta view combina os dados das tabelas AMS.User_permissions e AMS.Permissions para mostrar informações sobre os utilizadores e as suas permissões. A view é utilizada na página de gestão de utilizadores, permitindo apenas ao administrador ver e editar as permissões de cada utilizador.

Visual Studio - Interface and Code

Página 1, registo de conta ou login:

1. Ao entrar na aplicação o utilizador terá de se escolher entre fazer login ou registar-se na página:

2. Na página de registo terá de colocar o primeiro nome, último nome, data de nascimento, sexo, email e password. Assim ao usar email já se pode diferenciar entre cada pessoa mesmo que tenham o mesmo nome e já previne que criem contas ao acaso porque é necessário aceitar a confirmação enviada para o email para que a conta seja ativada com sucesso.

3. Na página de login pede-se o email e a password para entrar.

Página 2: Perfil

1. Página onde o utilizador pode alterar os seus dados pessoais e acrescentar dados para serem identificados mais facilmente, salário, altura, peso, cartão de cidadão

2. É possível adicionar uma imagem de perfil para o utilizador e é também possível aqui apagar a sua conta junto com todos os seus dados.

Página 3: Main page

1. Se o utilizador ainda não pertencer a clubes a maior parte da página estará vazia, mas tem as opções para aceder às várias funcionalidades e caso não pertença a um clube tem as opções de criar clube ou de entrar para um clube existente ao introduzir a SIGLA.

2. No caso de pertencer a um clube irá aparecer o calendário do clube onde apenas administradores e treinadores podem colocar quais os atletas que vão participar na prova e treinador(es) para vigiar, desses dias.

4. Sendo que quando envia o convite para entrar no clube ou o clube envia o convite para o mail da pessoa, um dos admins do clube depois decide se a pessoa entra como atleta, treinador, sócio, espectador, ou outro admin.

5. O primeiro a criar e depois a entrar no clube entra como admin e depois já pode editar o que quiser.

Página 3: Clube

1. O utilizador pode procurar por todos os grupos existentes, ver os seus dados tais como email, número de atletas, localização, entre outros e escolher pertencer a qualquer um deles.

2. Caso o utilizador seja admin também pode na página criar um clube e adicionar dados relevantes a ele. Também consegue procurar por um clube e alterar os seus dados existentes ou eliminá-lo por completo.

Conclusão

Para concluir com este projeto fomos capazes de adquirir um conhecimento aprofundado sobre bases de dados relacionais, Linguagem SQL, C#, Microsoft SQL Server, e como fazer a ligação e interação entre Visual Studio e o Microsoft SQL Server.

Embora não tenhamos conseguido alcançar todos os objetivos definidos nos inícios do projeto, devido à enorme complexidade e tempo necessário que se mostrou ser necessário para conseguir desenvolver o projeto.

Acreditamos mesmo assim que conseguimos implementar a maioria das funcionalidades que achámos mais relevantes para conseguir demonstrar de forma clara a nossa aplicação a funcionar, demonstrando tudo aquilo que fomos capazes de aprender e desenvolver para esta cadeira de base de dados.