Entrega número 1

Projeto de Base de Dados

Campeonato Português de Futebol

Grupo 401
Turma 2LEIC04

Daniel dos Santos Ferreira - up202108771
Francisco Cardoso - up202108793
Maria Eduarda Rabelo - up202000130

Docente: António Sá Pinto

Novembro de 2022

Index

Descrição da base de dados

Este projeto baseia-se numa base de dados para a gerência dos resultados do campeonato português de futebol da primeira divisão.

Em primeiro lugar, esta base de dados considera o armazenamento dos jogos do campeonato, guardando assim a data a que eles ocorrem. É também importante associar as equipas a cada jogo em que participaram, distinguindo-as em equipa visitante e equipa que joga em casa, guardando também a equipa que venceu, podendo esta ser nula quando o jogo resulta num empate.

Para cada jogo, importa saber a sua data. Além disso, um jogo está associado a todos os seus eventos, sendo estes: os golos marcados a favor de uma certa equipa por um certo jogador, os cartões que os jogadores recebem, sendo importante saber a cor do cartão, as substituições que ocorrem e qualquer outro evento que possa ocorrer (lesões de jogadores, defesas por parte dos guarda-redes, etc.). Nesta base de dados, estes eventos são organizados por uma hierarquia completa (qualquer evento que não pertença a uma das subclasses golo, substituição ou cartão terá de pertencer à subclasse outro) e disjunta (cada evento só poderá pertencer a uma e só uma subclasse), sendo importante saber o minuto a que qualquer dos eventos ocorre.

Cada um dos jogos terá de fazer parte de uma jornada do campeonato, sendo armazenado o número de cada jornada. Além disso, cada jornada está associada a todas as equipas do campeonato, armazenando-se o número de pontos e respetiva classificação para cada equipa nessa jornada.

Para a representação das equipas, são armazenados o nome, a localidade, as condições da equipa para irem à Champions e à liga Europa, e se estão em risco de despromoção. São armazenados também os estádios onde se realizam os jogos do campeonato e a capacidade destes, sendo os estádios associados à equipa a que lhe pertence.

Por último, são armazenados todos os jogadores que participam no campeonato, sendo guardados o nome, nacionalidade, a idade (apesar de normalmente guardar-se a data de nascimento como atributo e ter-se a idade como atributo derivado como nós fomos buscar os dados ao site zerozero e no site não tinha a data de nascimento, tivemos de optar por usar a idade, sendo que a idade na base de dados é a idade que o jogador tinha no fim da época)e o número da camisola de cada jogador. Cada jogador está associado à equipa a que pertence e também a qualquer evento que ocorra em qualquer jogo em relação a tal jogador.

Atributos derivados

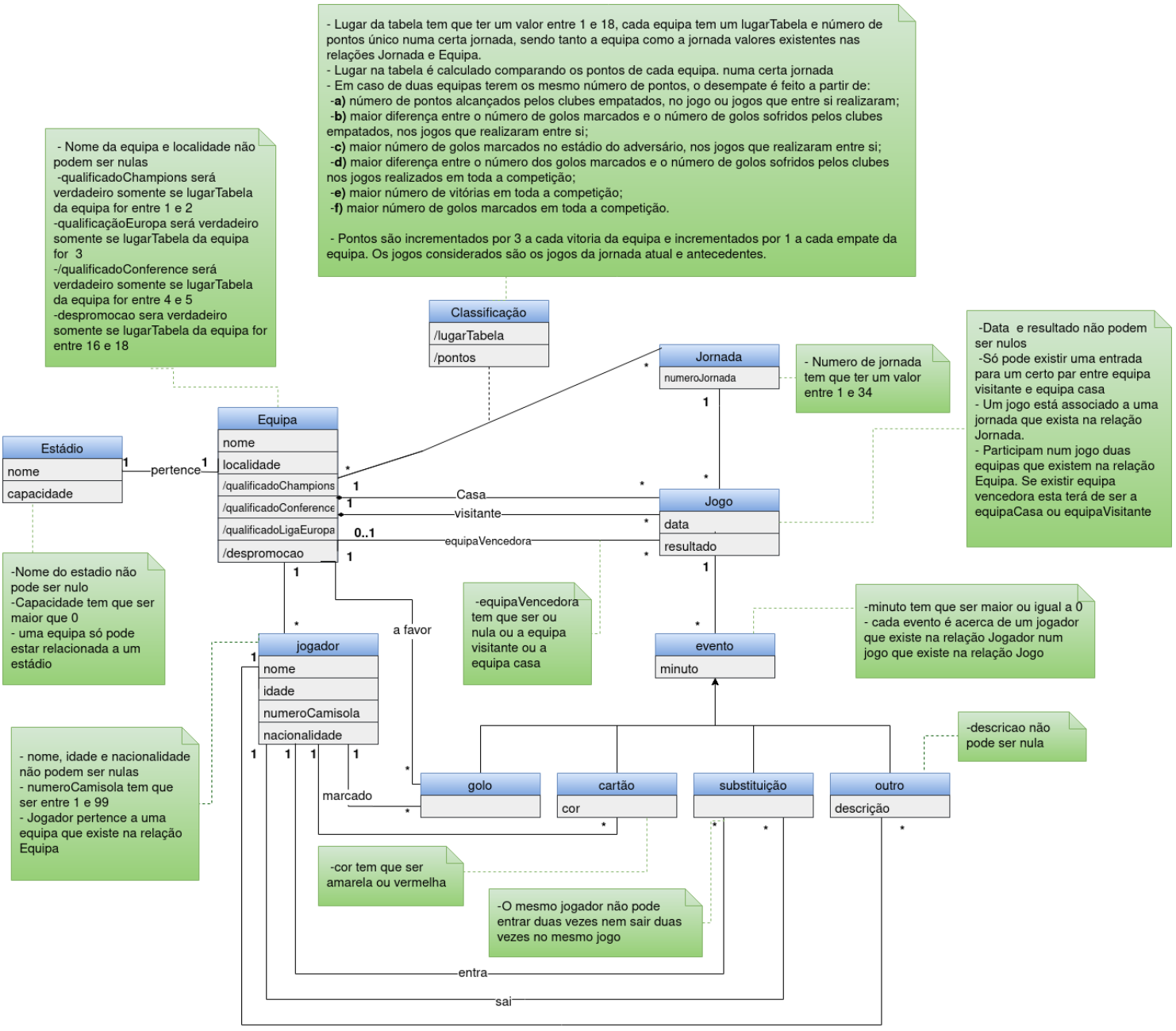
Nesta base de dados, existem alguns atributos derivados, ou seja, atributos cujo valor é calculado e atualizado à medida que se adicionam dados noutras relações na base de dados. Estes atributos serão implementados na base de dados através de triggers. Os atributos derivados em questão são: os pontos e lugar na tabela, de cada equipa, em cada jornada, além das condições de apuramento à Champions, à Liga Europa e o risco de despromoção de cada equipa.

Na relação Classificação, os atributos derivados são os pontos e o lugar na tabela. Para a atribuição do valor dos pontos, terá de se adicionar ao número de pontos atual, mais 3 pontos a cada vitória ou mais um ponto a cada empate. O número de pontos de uma certa jornada está associada aos jogos desta jornada e de todas as jornadas antecessoras . Para a atribuição do lugar na tabela numa certa jornada, ter-se-á de recolher todos os pontos de cada equipa nessa jornada, e obter o respetivo lugar, tendo em conta as regras de desempate presentes no esquema UML.

Na relação Equipa, os atributos derivados relativos à qualificação à Champions, à Conference League, à Liga Europa e o risco de despromoção de cada equipa são booleanos, pelo que se deve verificar certas condições. Para ser qualificado à Champions, o atributo será verdadeiro quando o lugar na tabela da respetiva equipa for 1 ou 2. Para ser qualificado à Liga Europa, o atributo será verdadeiro quando o lugar na tabela da respetiva equipa for 3. Para ser qualificado à Conference League, o atributo será verdadeiro quando o lugar na tabela for 4º ou 5º. Por último, para uma equipa estar em risco de despromoção, o atributo será verdadeiro se o lugar na tabela da mesma estiver entre 16 e 18.

A implementação dos atributos derivados em SQL será realizado somente para a segunda entrega do projeto.

Diagrama de classes UML



Definição do Esquema Relacional

Jogador(idJogador, nome, idade, numeroCamisola, nacionalidade, idEquipa->Equipa)

Equipa(idEquipa, nome, localidade, /qualificadoChampions, /qualificadoLigaEuropa, /qualificadoConference, /despromocao)

Estadio(idEstadio, nome, capacidade, idEquipa->Equipa)

Jogo(idJogo, data, resultado, numeroJornada->Jornada, idEquipaCasa->Equipa, idEquipaVisitante->Equipa, idEquipaVencedora->Equipa)

Classificacao(idEquipa -> Equipa, numeroJornada -> Jornada, /lugarTabela, /pontos)

Jornada(numeroJornada)

Golo(idEvento, minuto, idJogo->Jogo, idEquipa->Equipa, idJogador->Jogador)

Cartão(idEvento, minuto, idJogo->Jogo, idJogador->Jogador, cor)

Substituição(idEvento, minuto, idJogo->Jogo, idJogadorEntra -> Jogador, idJogadorSai -> Jogador)

Outros(idEvento, minuto, idJogo->Jogo, idJogador->Jogador, descricao)

Análise de Dependências Funcionais e Formas Normais

Análise de Dependências Funcionais

Tabela Jogador:

Jogador(idJogador, nome, idade, numeroCamisola, nacionalidade, idEquipa -> Equipa)

FD: {idJogador} -> {nome, idade, numeroCamisola, nacionalidade, idEquipa->Equipa}

Existe apenas um id por cada jogador (*idJogador*), de forma que é possível determinar qual jogador está a ser referido e, consequentemente, todos os seus atributos. Ou seja, se existirem dois jogadores com o mesmo idJogador terão de concordar em todos os outros atributos.

Tabela Equipa:

Equipa(idEquipa, nome, localidade, /qualificadoChampions, /qualificadoLigaEuropa, /qualificadoConference, /despromocao)

FD: {idEquipa} -> {nome, localidade, /qualificadoChampions, /qualificadoLigaEuropa, /qualificadoConference, /despromocao}

Existe apenas um id por cada equipa (idEquipa), de forma que é possível determinar qual equipa está a ser referida e, consequentemente, todos os seus atributos. Ou seja, se existirem duas equipas com o mesmo idEquipa terão de concordar em todos os outros atributos.

Tabela Estádio:

Estádio(idEstadio, nome, capacidade, idEquipa->Equipa)

FDs: {idEstadio} -> { nome, capacidade, idEquipa ->Equipa}
{idEquipa ->Equipa} -> {idEstadio, nome, capacidade}

Existe apenas um id para cada estádio (idEstadio), de forma que é possível determinar qual estádio está a ser referido e, consequentemente, todos os seus atributos. Assim, duas entradas nesta relação com o mesmo idEstadio têm de concordar nos restantes atributos.

Além disso, cada equipa tem um e um só estádio, pelo que se existirem duas entradas na relação Estádio com o mesmo idEquipa, os restantes atributos das duas entradas terão de ser iguais.

Tabela Jogo:

Jogo (idJogo, data, resultado, numeroJornada->Jornada, idEquipaCasa -> Equipa, idEquipaVisitante->Equipa, idEquipaVencedora->Equipa)

FD: {idJogo} -> {data, resultado, numeroJornada->Jornada, idEquipaCasa ->Equipa, idEquipaVisitante ->Equipa, idEquipaVencedora ->Equipa}
{IdEquipaCasa->Equipa, idEquipaVisitante->Equipa} -> {idJogo, data, resultado, #númeroJornada->Jornada, idEquipaVencedora ->Equipa}

Existe apenas um id para cada jogo (*idJogo*), de forma que é possível determinar qual jogo está a ser referido e, consequentemente, todos os seus atributos. Ou seja, se existirem duas entradas nesta relação com o mesmo idJogo, então os restantes atributos terão de concordar.

Existe apenas um jogo com cada equipa como casa e outra equipa como visitante, pelo que com o idEquipaCasa e o idEquipaVisitante, sabe-se a qual jogo se refere e, consequentemente, todos os seus demais atributos. Assim, se existirem duas entradas nesta relação com o mesmo idEquipaCasa e idEquipaVisitante, então os restantes atributos terão de concordar.

Tabela Classificação:

Classificação(idEquipa -> Equipa, numeroJornada -> Jornada, /lugarTabela, /pontos)

FDs: {idEquipa -> Equipa, numeroJornada -> Jornada}-> {/lugarTabela, /pontos}
{numeroJornada -> Jornada, /lugarTabela} -> {idEquipa -> Equipa, /pontos}

A relação da classe associada “Classificação” pode ser determinada pelas chaves estrangeiras idEquipa e númeroJornada, o que permite saber qual o lugar na tabela e o número de pontos dessa equipa nessa determinada jornada (atributo restantes).

Em cada jornada, cada lugar na tabela aponta para apenas uma equipa e número de pontos, pelo que os dois primeiros são suficientes para se chegar aos atributos restantes. Ou seja, se existirem duas entradas nesta relação com o mesmo lugarTabela e idJornada, então os restantes atributos terão de concordar.

Tabela Jornada:

Jornada (numeroJornada)

Na relação Jornada, temos apenas um atributo (numeroJornada), pelo que não é possível obter-se dependências não triviais. Assim, também não é possível a decomposição para a Forma Normal Boyce-Codd ou para a 3ª Forma Normal.

Tabela Golo:

Golo(idEvento, minuto, idJogo->Jogo, idEquipa->Equipa, idJogador->Jogador)

FD: {idEvento} -> {minuto, idJogo->Jogo, idEquipa->Equipa, idJogador -> Jogador}

Existe apenas um id de evento (*idEvento*) para cada golo, de forma que, com isso, é possível determinar qual golo está a ser referido e, consequentemente, todos os seus atributos. Ou seja, se existirem duas entradas nesta relação com o mesmo idEvento, então os restantes atributos terão de concordar.

Tabela Cartão:

Cartão (idEvento, minuto, idJogo->Jogo, idJogador->Jogador, cor)

FD: {idEvento} -> {minuto, #idJogo->Jogo, #idJogador->Jogador, cor}

Os atributos da relação Cartão podem ser determinados pelo idEvento, o que permite saber todos os demais atributos: minuto, idJogo, idJogador e cor; pelo que o lado esquerdo da FD é uma super chave.

Tabela Substituição:

Substituição (idEvento, minuto, idJogo->Jogo, idJogadorEntra -> Jogador, idJogadorSai -> Jogador)

FDS: {idEvento} -> { minuto, idJogo->Jogo, #idJogadorEntra -> Jogador, idJogadorSai -> Jogador}
{idJogo->Jogo, idJogadorEntra->Jogador} ->{idEvento, minuto, idJogadorSai -> Jogador}
{idJogo->Jogo, idJogadorSai->Jogador} ->{idEvento, minuto, idJogadorEntra -> Jogador}

Os atributos da relação Substituição podem ser determinadoa pelo idEvento, o que permite saber todos os demais atributos: minuto, idJogo->Jogo, idJogadorEntra -> Jogador e idJogadorSai -> Jogador.

Uma vez que um mesmo jogador não pode entrar mais de uma vez no mesmo jogo e, da mesma forma, um mesmo jogador não pode sair mais de uma vez do mesmo jogo, é possível construir as duas outras dependências funcionais representadas: a partir de idJogo->Jogo e idJogadorEntra->Jogador chega-se aos atributos restantes, assim como a partir de idJogo->Jogo e idJogadorSai->Jogador, ou seja se existirem duas entradas nesta relação com o mesmo idJogo e idJogadorEntra ou com o mesmo idJogo e idJogadorSai, então os restantes atributos terão de concordar.

Tabela Outros

Outros(idEvento, minuto, idJogo->Jogo, idJogador->Jogador, descricao)

FD: {idEvento} -> {minuto, idJogo->Jogo, idJogador->Jogador, descricao}

Um idEvento só pode estar relacionado a um evento “Outros”, pelo que, a partir de idEvento, sabe-se todos os demais atributos: minuto, idJogo, idJogador e descricao.

Análise das Formas Normais

Uma relação atende à Forma Normal de Boyce-Codd se, para toda dependência funcional A -> B não trivial, A for uma superchave. Já para que atenda à 3a Forma Normal, é preciso que, para toda A - > não trivial, A seja uma super chave, ou B consista apenas em atributos primos (atributos que são membros de pelo menos uma chave da relação).

Neste projeto, apenas para a relação Jornada não é possível construir dependências funcionais não triviais, pelo que a relação não poderia atender a nenhuma das Formas Normais. Todas as dependências funcionais construídas são não-triviais, ou seja, os atributos que aparecem na parte esquerda não se repetem na parte direita. Além disso, a partir de A em cada uma delas, é possível chegar aos demais atributos da relação, pelo que é considerada uma super chave. Assim, afirma-se que todas as demais relações seguem a Forma Normal de Boyce-Codd, e portanto também a 3ª Forma Normal.

Adição de restrições à base de dados

Estádio:

- Dois estádios não devem ter o mesmo idEstadio
    - idEstadio PRIMARY KEY

- Nome do estádio não pode ser nulo
    - nome NOT NULL

- Capacidade do estádio tem de ser superior a 0
    - CHECK(capacidade > 0)

- idEquipa tem de ser uma equipa existente na relação Equipa
    - equipa REFERENCES Equipa(idEquipa)

- Não podem existir duas entradas cujo idEquipa seja o mesmo e os restantes atributos não o sejam (pois cada equipa só pode ter um e um só estádio)
    - UNIQUE(idEquipa)

- idEquipa não pode ser nula
    - idEquipa NOT NULL

Equipa

- Não podem existir duas equipas com o mesmo idEquipa
    - idEquipa PRIMARY KEY

- Nome da equipa não pode ser nulo
    - nome NOT NULL

- Localidade não pode ser nula
    - localidade NOT NULL

Jogo

- Não podem existir dois jogos com o mesmo idJogo
    - idJogo PRIMARY KEY

- Data não pode ser nula
    - data_ NOT NULL

- resultado não pode ser nulo
    - resultado NOT NULL

- Número da jornada terá de ser um número existente na relação Jornada
    - numeroJornada REFERENCES Jornada(numeroJornada)

- equipaCasa terá de ser uma equipa que exista na relação Equipa: uso de chave estrangeira
    - equipaCasa REFERENCES Equipa(idEquipa)

- equipaCasa não pode ser nula
    - equipaCasa NOT NULL

- equipaVisitante terá de ser uma equipa que exista na relação Equipa: uso de chave estrangeira
    - equipaVisitante REFERENCES Equipa(idEquipa)

- equipaVisitante não pode ser nula
    - equipaVisitante NOT NULL

- equipaVencedora terá de ser uma equipa que exista na relação Equipa: uso de chave estrangeira
    - equipaVencedora REFERENCES Equipa(idEquipa)

- equipaVencedora terá de ser ou a equipaCasa, ou a equipaVisitante, ou nula (em caso de empate)
    - CHECK(equipaVencedora = NULL OR equipaVencedora = equipaCasa OR equipaVencedora = equipaVisitante)

- não deverão existir duas entradas na relação jogo com um par (“equipaCasa”, “equipaVisitante”) igual, em que algum dos restantes atributos seja diferente
    - UNIQUE(equipaCasa, equipaVisitante)

Jornada

- numeroJornada é chave primária
    - numeroJornada PRIMARY KEY

- numeroJornada tem de ser um valor entre 1 e 24, inclusive
    - CHECK(numeroJornada >= 1 AND numeroJornada <= 34)

Golo

- idEvento é chave primária
    - idEvento PRIMARY KEY

- minuto terá de ser maior ou igual a 0
    - CHECK(minuto >= 0)

- idJogo terá de ser um jogo existente na relação Jogo
    - idJogo REFERENCES Jogo(idJogo)

- idEquipa terá de ser um jogo existente na relação Jogo
    - idEquipa REFERENCES Equipa(idEquipa)

- idJogador terá de ser um jogo existente na relação Jogo
    - idJogador REFERENCES Jogador(idJogador)

Jogador

- idJogador é chave primária
    - idJogador PRIMARY KEY

- nome não pode ser nulo
    - nome NOT NULL

- idade não pode ser nula
    - idade NOT NULL

- nacionalidade não pode ser nula
    - nacionalidade NOT NULL

- numeroCamisola tem de ser um número inteiro entre 1 e 99, inclusive
    - CHECK(numeroCamisola > 0 AND numeroCamisola <100)

- idEquipa terá de ser uma equipa existente na relação Equipa: uso de chave estrangeira
    - idEquipa REFERENCES Equipa(idEquipa)

Classificação

- Não poderão haver duas classificações diferentes para a mesma equipa numa certa jornada
    - PRIMARY KEY (idEquipa, numeroJornada)

- idEquipa terá de ser uma equipa existente na relação Equipa: uso de chave estrangeira
    - idEquipa REFERENCES Equipa(idEquipa)

- numeroJornada terá de ser uma jornada existente na relação Jornada: uso de chave estrangeira
    - numeroJornada REFERENCES Jornada(numeroJornada)

Cartão

- Não podem existir dois eventos de cartão com o mesmo idEvento
    - idEvento PRIMARY KEY

- minuto tem de ser maior ou igual que 0
    - CHECK(minuto >= 0)

- cor terá de ser amarelo ou vermelho
    - CHECK(cor = “amarelo” OR cor = “vermelho”)

- idJogo terá de ser um jogo existente na relação Jogo: uso de chave estrangeira)
    - idJogo REFERENCES Jogo(idJogo)

- idJogador terá de ser uma jogador existente na relação Jogador: uso de chave estrangeira
    - idJogador REFERENCES Jogador(idJogador)

Substituição

- Não podem existir dois eventos de substituição com o mesmo idEvento
    - idEvento PRIMARY KEY

- minuto tem de ser maior ou igual que 0
    - CHECK(minuto >= 0)

- idJogo deve ser um jogo existente na relação Jogo: uso de chave estrangeira
    - idJogo REFERENCES Jogo(idJogo)

- idJogadorEntra terá de ser um jogador existente na relação Jogador: uso de chave estrangeira
    - idJogadorEntra REFERENCES Jogador(idJogador)

- idJogadorSai terá de ser um jogador existente na relação Jogador: uso de chave estrangeira
    - idJogadorSai REFERENCES Jogador(idJogador)

- não se pode ter o mesmo jogador a sair mais de uma vez, no mesmo jogo
    - UNIQUE(idJogo, idJogadorSai)

- não se pode ter o mesmo jogador a entrar mais de uma vez, no mesmo jogo
    - UNIQUE(idJogo, idJogadorEntra)

Outros

- Não podem existir dois eventos de substituição com o mesmo idEvento
    - idEvento PRIMARY KEY

- minuto tem de ser maior ou igual que 0
    - CHECK(minuto >= 0)

- idJogo terá de ser um jogo existente na relação Jogo: uso de chave estrangeira
    - idJogo REFERENCES Jogo(idJogo)

- idJogador terá de ser uma jogador existente na relação Jogador: uso de chave estrangeira
    - idJogador REFERENCES Jogador(idJogador)

- descricao não pode ser nula
    - descricao NOT NULL

Integridade da base de dados

Para além disso, em todas as chaves estrangeiras foi implementado o ON DELETE CASCADE e o ON UPDATE CASCADE, de forma a manter a integridade da base de dados.

Avaliação da participação dos elementos do grupo

A divisão de tarefas do projeto foi feita de forma equilibrada entre os elementos do grupo. O diagrama UML foi realizado em conjunto, em que todos participaram da sua construção, assim como a definição do esquema relacional, das dependências funcionais, da análise sobre as formas normais (BCNF e 3NF), e a criação da base de dados. A definição das restrições existentes no projeto foi feita em equilíbrio pelos integrantes do grupo.

O carregamento de dados à base de dados foi feito pelos participantes Daniel Ferreira e Francisco Cardoso, e foi realizado através de um web scraper usado para buscar informação necessária no site zerozero a passá-la para um ficheiro csv, após isso foi usado um scraper em c++ para reutilizar a informação e transforma-la num ficheiro. As informações usadas para o campeonato português de futebol foram encontradas no link abaixo:

https://www.zerozero.pt/edition_matches.php?id_edicao=156405&fase_in&equipa=0&estado=1&filtro=&op=calendario&page=7