Shape

Description automatically generated with low confidenceShape

Description automatically generated with low confidence **Entrega nº2 (alíneas A. a H.) – Grupo – Base de Dados 22/23**

Base de Dados da Liga dos Campeões edição 21/22

Diagrama UML

Chart, box and whisker chart

Description automatically generated

Contexto da Base de Dados

**Introdução:**

No âmbito da UC – Base de Dados da licenciatura em Engenharia Informática e Computação, desenvolvemos um projeto com base na Liga dos Campeões, edição 2021/22. No momento inicial do projeto, focamos a nossa análise na construção de uma base sólida com vista a evitar problemas futuros.

**Equipa:**

O projeto tem início na classe “Equipa”. Relativamente a esta, devemos reter o atributo do seu nome. A “Equipa” está associada a duas outras classes, “Jogador” e “Clube”. É importante realçar que a uma classe “Clube”, respeita apenas uma “Equipa”, sendo que a esta pode respeitar vários “Jogador”.

**Jogador:**

Como referido anteriormente, uma equipa contém muitos “Jogador” e desses é apenas relevante guardar informação relativamente ao seu nome, desse modo o nosso único atributo na classe “Jogador” não pode ser nulo, obviamente.

Associado a cada “Jogador” temos Generalizações (Substituição, Golo, Cartão) que derivam de uma outra classe denominada por “Evento”.

**Estádio:**

À classe “Estádio” respeita dois atributos: nome, cidade em que se localiza. Os atributos associados a esta classe respeitam informações cruciais à realização da nossa base de dados, daí não poderem ser nulos.

**Clube:**

A classe “Clube” deve conter os atributos nome, morada e ano de fundação (nota que o nome e a morada não poderão ser nulos). Associada a “Clube” encontramos uma outra classe - “Estádio” -, classe esta a que vários “Clubes” podem respeitar.

**Jornada:**

“Jornada” é uma classe sem qualquer atributo, no entanto, é igualmente necessária e crucial para localizar a fase de Grupos.

**Eliminatória:**

A classe “Eliminatória” contém o atributo tipo. Este está definido de forma a ser aceite apenas quando respeita a FG = fase de grupos; OF = oitavos de final; QF = quartos de final;

MF = meia-final; F = final;

**Jogo:**

No nosso ponto de vista, a classe “Jogo” é como uma *classe Mãe* nesta base de dados. Isto porque através dela, obtemos todas as informações necessárias à nossa análise. Na classe “Jogo” dispomos dos atributos nome e data de realização, sendo ambos não nulos. Associada a esta classe segue as classes "Finanças", “Jornada”, “Eliminatória”. Referimos também que existem duas ligações entre a classe “Equipa” e duas outras a que dizem respeito uma equipa visitante e uma equipa visitada.

**Substituição, Golo, Cartão e Evento:**

Três novas classes são: “Substituição”, “Golo” e “Cartão”. Estas derivam de uma ligação com uma outra classe “Evento”, através de uma junção de duas chaves estrangeiras – (idJogador->Jogador, idEvento->Evento), exceto a classe “Cartão” a que refere o atributo cor.

Relativamente á “Substituição”, de forma a guardar informações relativas a que “Jogador” entra, guardamos a partir da classe Jogador.

No que respeita à classe “Golo”, definimos também uma ligação com o nome do beneficiado de forma a conhecer que equipa beneficiou do golo marcado.

Na classe “Evento” temos um único atributo – minuto. Este não pode ser nulo e deve ser maior que 0, isto porque se trata de uma variável quantitativa positiva (tempo). Várias classes “Evento” associam a uma única classe “Jogo”.

Esquema relacional

Table

Description automatically generated

Esquema Relacional e Dependências funcionais

- **Equipa** (idEquipa, nome)

FDs: {idEquipa} -> {nome}

Formas: BCNF: Sim

3NF: Sim

- Evento (idEvento, minuto, idJogo->jogo)

FDs: {idEvento} -> {minuto, idJogo -> jogo}

Formas: BCNF: Sim

3NF: Sim

- Finanças (idFinanças, premio)

FDs: {idFinanças} -> {premio}

Formas: BCNF: Sim

3NF: Sim

- Jogador (idJogador, nome, idEquipa -> Equipa)

FDs: {idJogador} -> {nome, idEquipa -> Equipa}

Formas: BCNF: Sim

3NF: Sim

- Clube (idClube, nome, morada, anoFundação, idEstadio->Estadio, idEquipa->Equipa)

FDs: {idClube} -> {nome, morada, anoFundação, idEstadio, idEquipa}

Formas: BCNF: Sim

3NF: Sim

- Estadio (idEstadio, nome, cidade)

FDs: {idEstadio} -> {nome, cidade}

Formas: BCNF: Sim

3NF: Sim

- Jogo (idJogo, local, data, idEstadio->Estadio, idJornada->Jornada, idEliminatoria->Eliminatoria, idFinanças->Finanças)

FDs:{idJogo} -> {local, data, idEstadio, idJornada, idEliminatoria, idFinanças}

Formas: BCNF: Sim

3NF: Sim

- Visitada (idEquipa->equipa, idJogo->jogo)

FDs: {idEquipa} -> {idJogo}

Formas: BCNF: Sim

3NF: Sim

- Visitante (idEquipa->equipa, idJogo->jogo)

FDs: {idEquipa} -> {idJogo}

Formas: BCNF: Sim

3NF: Sim

- Jornada (idJornada)

- Eliminatoria (idEliminatoria, tipo)

FDs:

{idEliminatoria} -> {tipo}

Formas: BCNF: Sim

3NF: Sim

- Cartao (idEvento->Evento, idJogador -> Jogador, cor)

FDs: {idEvento} -> {idJogador, cor}

Formas: BCNF: Sim

3NF: Sim

- Golo (idEvento->Evento, idJogador -> Jogador)

FDs: {idEvento} -> {idJogador}

Formas: BCNF: Sim

3NF: Sim

- Substituiçao (idEvento->Evento, idJogador ->Jogador)

FDs: {idEvento} -> {IdJogador }

Formas: BCNF: Sim

3NF: Sim

- Contém (idJogador->nome, idEquipa->nome)

FDs: {idJogador} -> {idEquipa}

Formas: BCNF: Sim

3NF: Sim

- Beneficiado (idEquipa -> Equipa, idEvento -> Evento)

FDs: {idEvento} -> {idEquipa}

Formas: BCNF: Sim

3NF: Sim

Analise de Restrições

1. Equipa 🡪 Atributo nome não pode ser nulo. Todas as equipas que pertencem á Champions League possuem um nome deste modo este campo não pode ser NULL.
2. Visitante e Visitado 🡪 Utilizamos o Check como forma de garantir uma atribuição justa a cada Equipa. Valor por:
3. Campeão = 10000;
4. Vice-Campeão = 1500;
5. Vitoria = 1000;
6. Empate = 500;
7. Derrota = 0;
8. Estádio 🡪 Os atributos nome e cidade não podem ser nulos;
9. Clube 🡪 Os atributos nome e morada não podem ser nulos;
10. Eliminatória 🡪 Utilizamos o *Check* de forma a sabermos em que fase estamos;
11. FG = Fase de Grupos;
12. OF = Oitavos de Final;
13. QF = Quartos de Final;
14. MF = Meia-Final;
15. F = Final;
16. Minuto 🡪 Para garantir que não registamos eventos antes do minuto 0 introduzimos um *Check* (Minuto > 0);
17. Cartão 🡪 Utilizou-se o *Check* para garantirmos que todos os cartões que eram introduzidos na base de dados são iguais a V (Vermelho) ou A (Amarelo);
18. Jogador 🡪 O nome de um Jogador não pode ser nulo.

Querries SQL

Para verificarmos o correto funcionamento dos vários componentes e conexões na nossa base de dados definimos as seguintes interrogações:

1. Listagem dos estádios: verificação da existência da tabela Estádio constituída pelos seguintes atributos: idEstadio, nome, cidade;
2. Listagem de todos os eventos que ocorreram entre o minuto 45 e o minuto 90 (praticamente a segunda parte). Nesta interrogação verificamos que a variável é um inteiro (tal como definida) visto que nos permite fazer comparações com outros inteiros, dando uso ao parâmetro BETWEEN;
3. Listagem completa de todos os jogos que foram implementados na nossa Base de dados: Verificação da conexão entre o idjogo de Visitante e idjogo de Visitado.  
   Utilizou-se o INNER JOIN como método auxiliar da união.
4. Listagem das equipas que não tiveram nenhum jogo como visitante: Estabelecemos uma interligação entre Equipa e Visitante a partir do atributo que existe em comum entre ambas que é o idEquipa. Utilizou-se a função NOT IN para obtermos todas as equipas que não realizaram nenhum jogo fora de casa;
5. Listagem da média de Jogadores por equipa: Verificação da conexão entre equipa e jogador. Utilizou-se a função matemática AVG e COUNT como métodos auxiliares para calcular a media. Foram também atribuídos nomes de modo a facilitar o manuseamento entre os atributos.
6. Listagem dos melhores marcadores nesta Edição da Liga dos Campeões: Verificação da conexão entre Jogador e Golo. Recorreu-se ao Count(\*) como método auxiliar para calcularmos quais os jogadores com mais golos nesta edição. No final agrupamos a tabela pelo nome do jogador e ordenamos pelo número de golos.
7. Listagem dos jogadores que começam pela letra ‘r’ e que marcaram pelo menos um golo durante a sua campanha na Champions League: Estabelecemos uma relação entre Equipa,Jogador,Golo. Recorremos á função LIKE para obtermos todos os jogadores que começam pela letra ‘r’ de seguida estabelecemos ligação entre o Golo e o Jogador e de seguida Jogador e a Equipa a que pertence;
8. Listagem da Equipa que contem mais jogadores: Verificação da conexão entre Contem e Equipa a partir da chave idequipa. Utilizou-se o COUNT para contabilizarmos quantos jogadores cada equipa continha e no final o MAX para descobrirmos qual a equipa com mais jogadores.
9. Listagem dos Jogadores que marcaram e que, num momento diferente ou não, foram advertidos com um cartão: Verificação da conexão entre cartão e golo. Recorreu-se á função auxiliar INTERSECT para listar todos os jogadores envolvidos em ambos os eventos.
10. Listagem do Montante Total Arrecadado por cada Equipa: Criou-se duas views: uma para guardar os ganhos como visitante e outra para guardar os montantes como visitado. Para o Visitante estabeleceu-se relações entre Equipa,Visitante,Jogo. Deste modo a partir das chaves conseguiu-se estabelecer relação entre todos os intervenientes. Para o Visitado o método adotado foi o mesmo com a troca do Visitante pelo Visitado. No final para reunirmos ambos os valores recorreu-se aos seguintes métodos auxiliares UNION ALL e SUM para se atingir um dos objetivos deste trabalho.

Implementação dos Gatilhos:

1. Neste primeiro gatilho decidimos começar por um caso mais simples, mas que ao mesmo tempo fosse eficiente e útil. Desse modo decidimos implementar um gatilho que depois de adicionado um Jogador este é automaticamente adicionado a Contem (Associação que permite a ligação entre Jogador e Equipa).

Exemplo: Adicionamos um jogador com o idJogador = 62 (este valor pode também ser nulo visto que se não lhe dermos valor ele vai somar uma unidade ao último id da lista), nome = ‘Wilson Manafa’ e idequipa = 7. Automaticamente vai ser também adicionado á Tabela Contem com os seguintes valores (62,7);

1. O segundo gatilho foi estruturado de uma forma diferente. A partir da função auxiliar julianday() verificamos se a data do jogo é possível ou não, ou seja, ao atualizar-se uma data de um determinado jogo se esta for antes de dia 15-09-2021 será invalida porque este foi o primeiro dia da competição naquela época, no entanto se a data for posterior da anteriormente mencionada então a data será valida. Se a data atualizada for invalida então desencadeia-se um erro.

Exemplo:

O idjogo = 1 realizou-se no dia 15 de setembro de 2021;

Se se der update á data e modificá-la por uma data aleatória como por exemplo: 19 de março de 2019 então levantar-se-á um erro mostrando que a data de jogo modificada é invalida voltando assim á data inicial.

No entanto se se der update á mesma data para dia 15 de novembro de 2021 então a data introduzida é valida e será desta forma atualizada.

1. O terceiro gatilho é essencial para qualquer partida de futebol: Expulsão.

De forma a garantirmos que ambos os eventos fossem registados qualquer Jogador que no mesmo jogo acumule dois Amarelos ou um Vermelho é expulso não podendo receber mais nenhum cartão durante a partida toda. Caso se atribua mais algum cartão a esse jogador durante essa partida será levantada uma mensagem de erro dizendo que o Jogador foi expulso e não recebendo assim mais nenhum cartão. Atenção tanto o segundo amarelo como o primeiro vermelho são sempre registados!

Exemplo:

Jogador = Phil Foden;

Jogo = 1;

Eventos:

1. Phil Foden recebe um cartão Amarelo (regista-se na Tabela Cartão);
2. Phil Foden recebe o segundo cartão Amarelo (regista-se na Tabela Cartão);
3. A partir deste momento qualquer cartão atribuído ao Foden desencadeará numa mensagem de erro dizendo que o Jogador foi Expulso;

Contribuição por cada elemento do grupo num total de 100%:

1. Eduardo Machado – 25%
2. Luis Relvas – 50%
3. Tomás Silva – 25%