# Base de Dados Campeonato Português de Andebol

---

**Grupo 6**

UP202108737 - João Pedro Cruz Moreira de Oliveira
UP202108751 - Samuel Alexandre Cruz Moreira de Oliveira
UP202006950 - Vicente Salvador Martinez Lora

---

## Estrutura

**1. Contexto** 1
**2. Diagrama UML** 2
**3. Definição do Esquema Relacional** 3
**4. Análise Dependências Funcionais** 4
**5. Adição de restrições à base de dados** 6
**6. Reflexão** 8
**7. Anexos** 9

---

### Enunciado

Desenvolver uma base de dados para gerir os **resultados do campeonato** português de andebol, da primeira divisão. Esta base de dados deve responder ao lançamento de **resultados jornada a jornada**, **marcadores dos golos**, **equipas** que jogam em **casa e jogam como visitante**. Pretende-se saber o estado do campeonato em qualquer jornada, as equipas em condições de ir às competições europeias e em risco de despromoção.

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

# 1. Contexto

Pretende-se implementar uma base de dados que permita gerir os resultados do campeonato de andebol português da primeira divisão Seniores Masculinos, atualmente designado, Placard Andebol 1.

A base de dados deve permitir o lançamento dos resultados jornada a jornada, quem foram os marcadores dos golos, as equipas que jogaram em casa e que jogaram como visitante e, daí, poder concluir as equipas que estão em condição de ir às competições europeias e em risco de despromoção, isto é, as 3 equipas com mais pontos, e as 3 equipas com menos pontos, respetivamente.

Um campeonato é identificado pelo seu nome, pelo ano em que decorre e possui um patrocinador que é atualizado todos os anos. Esta é constituída no máximo por 30 jornadas, tendo cada uma, no máximo, 8 jogos, podendo haver assim um total de 240 jogos por campeonato.

Cada jogo tem uma data, uma hora e um local associado, e participam sempre 2 equipas, uma visitante e uma visitada, como também uma lista dos marcadores do jogo e o seu resultado que é possível concluir pelo número de golos visitantes e visitados. Cada jogo pode se concluir num empate ou numa equipa vitoriosa e numa derrotada. Com base nesse resultado, uma equipa pode obter: 3 pontos por vitória, 2 pontos por empate e 1 por derrota.

Os atletas são distinguidos pelo nome, idade, número de camisola, posição e país de nascimento podendo ser um ponta esquerda(PE), ponta direita(PD), lateral esquerdo(LE), lateral direito(LD), central(C), pivot(PV) ou guarda-redes(GR). Assumimos que todos jogadores de uma equipa se mantêm nela até a abertura da janela de transferências no fim de cada Campeonato.

# 2. Diagrama UML

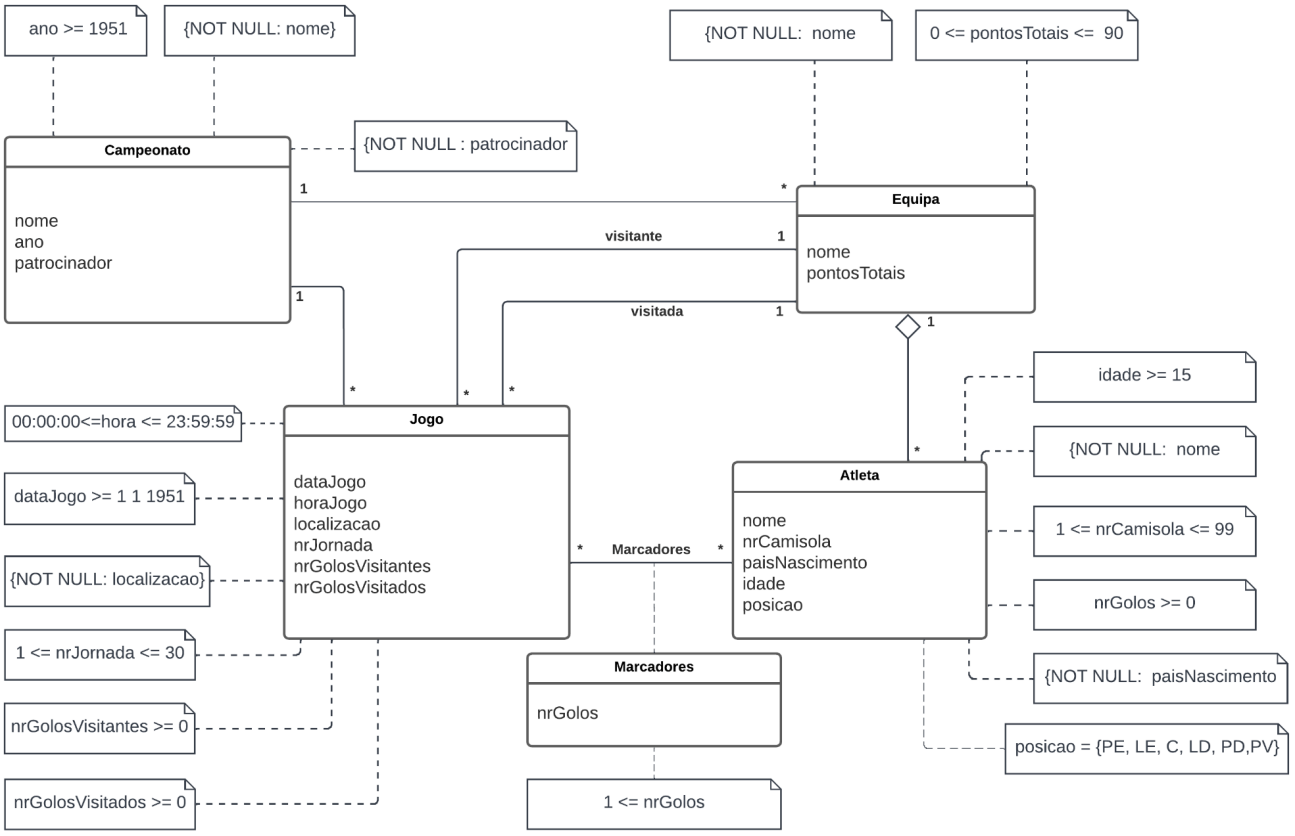



Fig 1: UML Revisto

O UML foi pensado no intuito de construir a base de dados mais simples mas capaz de responder a todas as questões pedidas no enunciado.

Decidimos associar a Equipa ao campeonato, pois cada ano, o nome de uma equipa e o seu plantel pode mudar, permitindo assim um melhor arquivamento da informação relativa às equipas para cada ano. Além disso, é possível também separar a informação relativa a cada jogador para cada época, pois estão associados a uma equipa e, consequentemente, a um campeonato.

Escolhemos representar os marcadores dos golos como uma classe de associação, pois assim temos acesso direto aos marcadores dos golos, podendo assim calcular o número de golos em cada jogo e comparar os marcadores entre si (pelo número de golos).

# 3. Definição do Esquema Relacional

Campeonato (idCampeonato, nome, ano, patrocinador)

Jogo (idJogo, dataJogo, horaJogo, localizacao, nrJornada, nrGolosVisitantes, nrGolosVisitados, visitante -> Equipa, visitada -> Equipa, idCampeonato -> Campeonato)

Equipa (idEquipa, nome, pontosTotais, idCampeonato -> Campeonato)

Atleta (idAtleta, nome, nrCamisola, posicao, paisNascimento, idade, idEquipa -> Equipa)

Marcadores (idJogo -> Jogo, idAtleta -> Atleta, nrGolos)

# 4. Análise Dependências Funcionais

Baseado no esquema relacional, destacamos as seguintes dependências funcionais para cada relação. De seguida, calculamos as relações que não cumprem a 3ª Forma Normal e a Forma Normal de Boyce-Codd. Colocando as conclusões nas duas colunas à direita.

A justificação encontra-se na página seguinte.

| Relação | Dependências Funcionais | 3FN | FNBC |
|---|---|---|---|
| **Campeonato** | **idCampeonato** -> nome, ano, patrocinador<br><br>**ano** -> idCampeonato, nome, patrocinador | ✅ | ✅ |
| **Jogo** | **idJogo** -> dataJogo, localizacao, nrJornada, nrGolosVisitantes, nrGolosVisitados, visitante, visitada<br><br>**nrJornada, localizacao** -> idJogo, dataJogo, nrGolosVisitantes, nrGolosVisitadas, visitante, visitada<br><br>**nrJornada, visitante** -> idJogo, dataJogo, localizacao, nrGolosVisitantes, nrGolosVisitadas, visitada<br><br>**nrJornada, visitada** -> idJogo, dataJogo, localizacao, nrGolosVisitantes, nrGolosVisitadas, visitante<br><br>**dataJogo, visitante** -> idJogo, localizacao, nrJornada, nrGolosVisitantes, nrGolosVisitadas, visitada<br><br>**dataJogo, visitada** -> idJogo, localizacao, nrJornada, nrGolosVisitantes, nrGolosVisitadas, visitante<br><br>**dataJogo, localizacao** -> idJogo, nrJornada, nrGolosVisitantes, nrGolosVisitadas, visitante, visitada | ✅ | ✅ |
| **Equipa** | **idEquipa** -> nome, pontosTotais, idCampeonato<br><br>**nome, idCampeonato** -> idEquipa, pontosTotais | ✅ | ✅ |
| **Atleta** | **idAtleta** -> nome, nrCamisola, posicao, paisNascimento, idade, idEquipa<br><br>**idEquipa, nrCamisola** -> idAtleta, nome, posicao, paisNascimento, idade | ✅ | ✅ |
| **Marcadores** | **idJogo, idAtleta** -> nrGolos | ✅ | ✅ |

**3FN - 3ª Forma Normal** **FNBC - Forma Normal de Boyce Codd**

Todas as relações cumprem a **primeira forma normal**, pois todas as linhas, de cada uma delas, possuem um registo, como também, todas as colunas possuem apenas um valor.

Além disso, em todas as dependências, todos os atributos que não pertencem às chaves, dependem de toda a chave e não de um subconjunto da chave, cumprindo assim a **segunda forma normal**.

Sabendo então que todas as dependências cumprem a 1ª e 2ª forma normal, podemos agora verificar se cumprem a **3ª forma normal**. Para tal, todos os atributos que não pertencem à chave, não podem depender de nenhum atributo que, também, não pertença à chave, o qual **se cumpre em todas as situações**.

Finalmente, para uma relação estar na forma normal de Boyce Codd, é necessário que todo o determinante da relação (numa dependência funcional A -> B, o determinante é o A) seja uma chave candidata, o que, mais uma vez, se verifica em todos os casos.

No entanto é importante notar que, num jogo, normalmente, sabendo a equipa visitada, é possível saber a localização do jogo. Porém, por motivos de manutenção do estádio, segurança ou outras razões, um jogo poderá decorrer numa **localização diferente à expectável**.

Nesse caso, **hipotético**, em que não se considere essa possível mudança de localização, ocorreria uma violação à 3ª forma normal e, consequentemente, à forma normal de Boyce Codd, pois um atributo fora da chave dependeria de outro, também, fora da chave.

**idJogo** -> dataJogo, localizacao, nrJornada, nrGolosVisitantes, nrGolosVisitados, visitante, visitada

visitada -> localizacao

No primeiro anexo (na página 9) incluímos uma decomposição assumindo esse caso, apenas por razões de demonstração de conhecimentos.

# 5. Adição de restrições à base de dados

Para garantir a integridade da base de dados definimos restrições para cada relação. As restrições usadas resumem-se nas seguintes: NOT NULL, CHECK, UNIQUE, PRIMARY KEY, FOREIGN KEYS.

**NOT NULL**: Garante que não se podem atribuir valores nulos
**CHECK**: Garante que os valores atribuídos devem cumprir uma certa condição.
**UNIQUE**: Garante que não existe valores iguais em tuplos distintos
**PRIMARY KEY**: Seleciona a coluna responsável por identificar cada fila da relação, sendo todos os seus valores distintos.
**FOREIGN KEYS**: Relaciona colunas entre tabelas.

Seguem-se as restrições aplicadas a cada relação da nossa base de dados:

## Campeonato:

- **PK_Campeonato** : identificador da tabela Campeonato, não podendo haver dois campeonatos com o mesmo ID; restrição chave (primary key).
- **UNIQUE_Campeonato_ano** : não pode haver dois Campeonatos com o mesmo ano; restrição chave (unique).
- **NOTNULL_Campeonato_ano** : O ano do Campeonato não pode ser nulo; restrição chave (NOT NULL)
- **NOT_NULL_Campeonato_nome** : O nome do Campeonato não pode ser nulo; restrição chave (NOT NULL)
- **NOT_NULL_Campeonato_patrocinador** : O patrocinador não pode ser nulo; restrição chave (NOT NULL)

## Jogo:

- **PK_Jogo** : identificador da tabela Jogo, não podendo haver dois jogos com o mesmo ID; restrição chave (primary key).
- **FOREIGNKEY_Jogo_visitante** : chave estrangeira da tabela Equipa referente ao visitante; integridade referencial (chave estrangeira)
- **FOREIGNKEY_Jogo_visitada** : chave estrangeira da tabela Equipa referente ao visitada; integridade referencial (chave estrangeira)
- **FOREIGNKEY_Jogo_idCampeonato** : chave estrangeira da tabela Campeonato referente ao idCampeonato; integridade referencial (chave estrangeira)
- **NOT_NULL_Jogo_localizacao** : A localizacao do Jogo não pode ser nula; restrição chave (NOT NULL)
- **CHECK_Jogo_dataJogo** : a data do jogo deverá ser pós a criação da liga de andebol portuguesa. Restrição CHECK (dataJogo >= 1951-01-01)
- **CHECK_Jogo_nrJornada** : restringe os dados inseridos no atributo jornada. Restrição CHECK(nrJornada >= 0 and nrJornada <= 30)

- **CHECK_Jogo_horaJogo** : a hora do jogo deverá ser dentro do horário possível dentro de um dia. Restrição CHECK (00:00:00 <= horaJogo AND horaJogo <= 23:59:59)
- **CHECK_Jogo_NrGolosVisitantes** : restringe os dados inseridos no atributo NrGolosVisitantes. Restrição CHECK(nrGolosVisitantes >= 0)
- **CHECK_Jogo_NrGolosVisitados** : restringe os dados inseridos no atributo NrGolosVisitados. Restrição CHECK(nrGolosVisitados >= 0)

## Equipa:

- **PK_Equipa** : identificador da tabela Equipa, não podendo haver duas equipas com o mesmo ID; restrição chave (primary key).
- **FOREIGNKEY_Equipa_idCampeonato** : chave estrangeira da tabela Campeonato referente ao idCampeonato; integridade referencial (chave estrangeira)
- **NOT_NULL_Equipa_nome** : O nome da Equipa não pode ser nulo; restrição chave (NOT NULL)
- **CHECK_Equipa_pontosTotais** : os pontos totais da equipa, deverão ser entre os valores possíveis num campeonato. Restrição CHECK (0 <= pontosTotais <= 90)

## Atleta:

- **PK_Atleta** : identificador da tabela Atleta, não podendo haver dois atletas com o mesmo ID; restrição chave (primary key).
- **FOREIGNKEY_Atleta_idJogo** : chave estrangeira da tabela Jogo referente ao idJogo; integridade referencial (chave estrangeira)
- **FOREIGNKEY_Atleta_idEquipa** : chave estrangeira da tabela Equipa referente ao idEquipa; integridade referencial (chave estrangeira)
- **CHECK_Atleta_nrCamisola** : O número da camisola do atleta deve estar entre os valores aceites pelo campeonato. Restrição CHECK (1 <= nrCamisola <= 99)
- **CHECK_Atleta_nrGolos** : O número de golos do atleta deve ser igual ou superior a zero. Restrição CHECK (0 <= nrGolos)
- **CHECK_Atleta_Posicao** : a posição do atleta deve ser uma das posições possíveis dentro de uma equipa de andebol. Restrição CHECK (posicao = ‘PE’ (Ponta Esquerda) OR posicao = ‘PD’ (Ponta Direita) OR posicao = ‘LE’ (Lateral Esquerdo) OR posicao = ‘LD’ (Lateral Direito) OR posicao = ‘C’ (Central) OR posicao = ‘PV’ (Pivot) OR posicao = ‘GR’ (Guarda Redes) )
- **NOT_NULL_Atleta_nome** : O nome do Atleta não pode ser nulo; restrição chave (NOT NULL)
- **NOT_NULL_Atleta_paisNascimento** : O país de nascimento do Atleta não pode ser nulo; restrição chave (NOT NULL)
- **CHECK_Atleta_idade** : a idade do atleta deve ser superior à idade mínima aceite pelo campeonato. Restrição CHECK (15 <= idade)

## Marcadores:

- **PK_Marcadores :** identificador da tabela Marcadores; restrição chave PRIMARY KEY (idJogo, idAtleta)
- **CHECK_Marcadores_nrGolos** : o número de golos deve ser um número natural igual ou superior a 1. Restrição CHECK (1 <= nrGolos)

# 6. Reflexão

Sentimos muitas dificuldades inicialmente ao começar o trabalho. Tivemos muita tendência para complicar o que nos era pedido. Especificamente, demorou até percebermos como funciona a criação de um UML, tendo perdido muito tempo na sua criação. É possível ver nos anexos (a partir da página 10) as várias fases por que a UML passou. Resumidamente, começámos por complicar demasiado até encontrármos um modelo muito mais simples, mas capaz de realizar tudo o que nos é requisitado pelo enunciado.

Se pudéssemos recomeçar, iriamos nos focar em, primeiro, compreender o que nos é exatamente pedido e entender a matéria, mal nos é lecionada. Deste modo poderíamos evitar perder tempo num objetivo que facilmente seria resolvido com um pouco de pesquisa e estudo.

Todos investimos bastante tempo na realização do trabalho e estamos confiantes que alcançámos um nível expectável de qualidade em conjunto. Esforçámo-nos por completar os objetivos pedidos e tentar ir um pouco além na sua execução. Na povoação do SQL fizemos por povoar mais do que achávamos necessário para permitir, no futuro, colocar questões, à nossa base de dados, com respostas realistas e precisas.

Concluindo, todos os constituintes participaram de igual forma na realização do trabalho pelo que classificamos o esforço de cada membro 33.3%.

# 7. Anexos

## Decomposição

Para o caso hipotético, em que se **assume** a **impossibilidade de mudar a localização de um jogo**, haveria uma violação à terceira forma normal e à forma normal de Boyce Codd. Realizámos então uma **decomposição** para essa situação:

**idJogo** -> dataJogo, localizacao, nrJornada, nrGolosVisitantes, nrGolosVisitados, visitante, visitada

visitada -> localizacao

| Determinante | | | | | | | |
|---|---|---|---|---|---|---|---|
| idJogo | dataJogo | localizacao | nrJornada | nrGolosVisitantes | nrGolosVisitados | visitante | visitada |
| visitada | | localizacao | | | | | |

### Decomposto em:

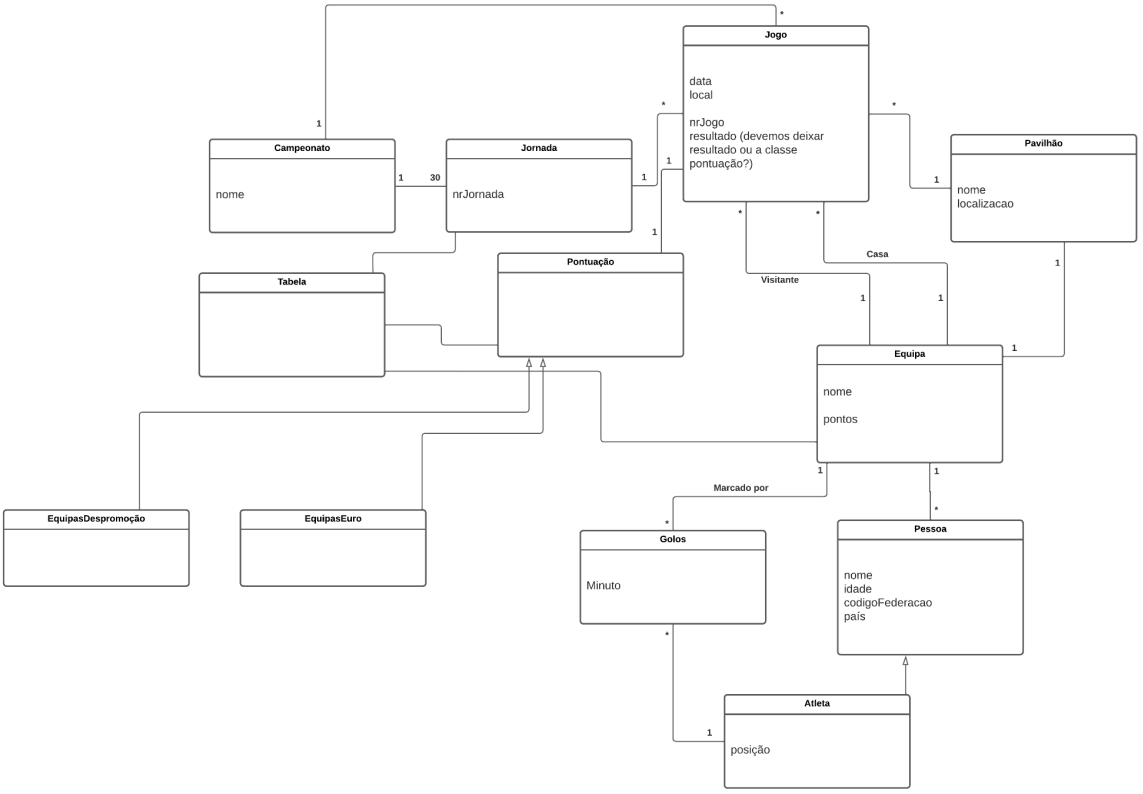
| Determinante | | | | | | |
|---|---|---|---|---|---|---|
| idJogo | dataJogo | nrJornada | nrGolosVisitante | nrGolosVisitados | visitante | visitada |

**E**

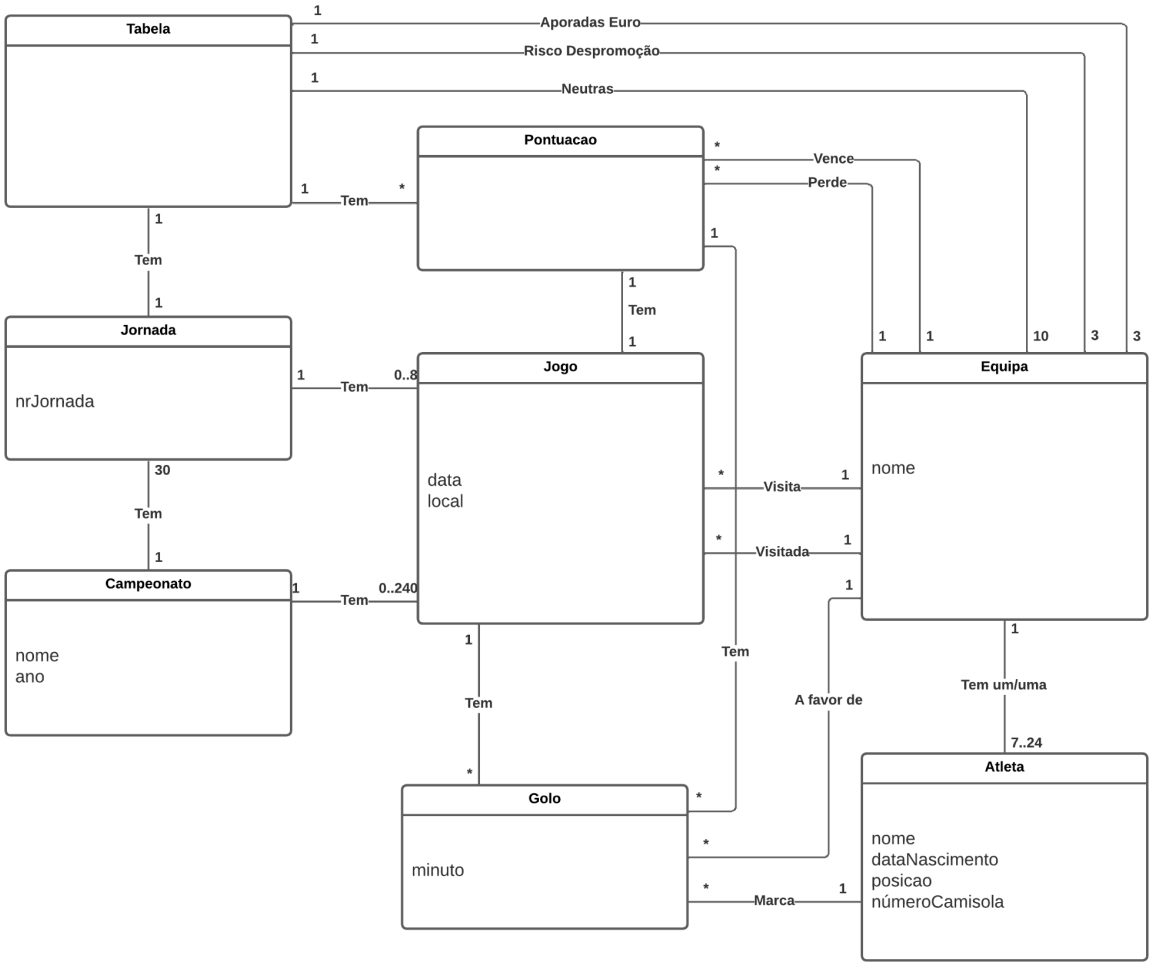
| Determinante | |
|---|---|
| visitada | localizacao |

# Evolução do UML

## 1ª Tentativa



## 2ª Tentativa



# 3ª Tentativa