

Figura 1 – Diagrama de Classes UML submetido na primeira entrega do projeto

- **ChampSelect** (idPartida->Partida, idJogador->Jogador, pick, ban, runas);
- **Objetivo** (idObjetivo, minuto, tipo, localizacao, idPartida->Partida, destroi->Equipa);
- **Eliminacao** (idEliminacao, minuto, idPartida->Partida, jogElimina->Jogador, jogEliminado->Jogador);
- **Assistencia** (idEliminacao->Eliminacao, jogador->Jogador)

De seguida, decidimos usar o ChatGPT como a ferramenta de Inteligência Artificial. O ChatGPT é um *chatbot*, isto é, uma aplicação, normalmente *online*, que simula um ser humano durante uma conversa com uma pessoa. Este *chatbot* baseia-se num grande modelo de linguagem desenvolvido pela empresa OpenAI, isto é, um modelo de linguagem natural que analisa probabilidades sobre um conjunto de palavras, de modo a prever a palavra seguinte e formar uma frase coerente, sendo sujeito a um processo de treino em que analisa enormes quantidades de dados através de métodos de aprendizagem computacional (*machine learning*). O grande modelo de linguagem disponibilizado pelo ChatGPT é o GPT-3.5 (*Generative Pre-trained Transformer 3.5*).

Após realizar o modelo relacional apresentado acima sem integração da ferramenta de Inteligência Artificial, abordamos a integração com esta ferramenta de dois métodos diferentes:

1º método: Entregando-lhe uma descrição completa e correta do UML (apresentado no final da 1ª entrega do projeto), e pedindo-lhe um modelo relacional adequado, obtivemos o seguinte modelo:

(PK significa *primary key* ou chave primária e FK significa *foreign key* ou chave externa)

Tabela Organização:

Atributos: ID (PK), Nome, Região, Localização

Tabela Equipe:

Atributos: ID (PK), Nome, ID_Organizacao (FK para Organização)

Tabela Jogador:

Atributos: ID (PK), Nome, Nickname, Idade, Nacionalidade, Residência, Posição, ID_Equipe (FK para Equipe)

Tabela Competição:

Atributos: ID (PK), Nome_Competição, Tipo de competição, Num_Jogos

Tabela Jogo:

Atributos: ID_Jogo (PK), Tipo de jogo, Data, Resultado Final, ID_Competicao (FK para Competição), ID_Equipe1 (FK para Equipe), ID_Equipe2 (FK para Equipe), ID_Arena (FK para Arena)

Tabela Partida:

Atributos: ID_Partida (PK), Resultado, ID_Jogo (FK para Jogo)

Tabela ChampSelect (Classe Associativa entre Jogo e Jogador):

Atributos: ID_Partida (PK, FK para Partida), ID_Jogador (PK, FK para Jogador), Pick, Ban, Runa

Tabela Evento:

Atributos: ID_Evento (PK), Tipo de evento

Tabela Eliminação (Especialização de Evento):

Atributos: *ID_Evento* (PK, FK para *Evento*), *Jogador_Eliminado*, *Jogador_Elimina*

Relações: *ID_Jogadores_Assistencia* (FK para *Jogador*)

Tabela Objetivo (Especialização de Evento):

Atributos: *ID_Evento* (PK, FK para *Evento*), *Nome*

Tabela Arena:

Atributos: *Nome* (PK), *Capacidade*, *Localização*

Tabela Época:

Atributos: *Ano* (PK)

Tabela Período de Contratação:

Atributos: *ID_Periodo* (PK), *Data de Início*, *Data de Fim*

Tabela Contrato (Classe Associativa entre Equipe, Jogador e Período de Contratação):

Atributos: *ID_Equipe* (PK, FK para *Equipe*), *ID_Jogador* (PK, FK para *Jogador*), *ID_Periodo* (PK, FK para *Período de Contratação*)

Este modelo relacional foi de encontro ao estabelecido por nós, com exceção da:

- Tabela *Evento*, que pelo facto de ter apenas um atributo, esse atributo foi adicionado à Tabela *Eliminação e Objetivo*;
- Relação *Contrato*, que considera o *ID_Equipe* uma chave primária, sendo que isso não vai de encontro à definição teórica de uma relação ternária com multiplicidade de 1 para muitos para muitos

2º método: Entregando o modelo relacional elaborado por nós, pedimos ao ChatGPT para reescrevê-lo usando a lógica necessária para elaborar uma base de dados de uma competição de *esports* (competição baseada num videojogo), ao que o ChatGPT não realizou alterações relevantes, por vezes sugerindo alterações que já se encontravam no modelo relacional (p. ex. índice 9), outras que não cumpriam o UML submetido (p. ex. índice 12.) ou que não se adequassem à realidade do domínio do projeto (p. ex. índice 4), como mostramos abaixo:

1. **Organizacao** (*idOrganizacao*, *nomeOrg*, *regiao*, *localizacao*);
 - *Keep the organization information as is.*
2. **Equipa** (*idEquipa*, *nomeEquipa*, *organizacao->Organizacao*);
 - *Add attributes relevant to teams, such as creation date, team captain, etc.*
3. **Jogador** (*idJogador*, *nomeJogador*, *position*, *nick*, *dataNascimento*, *nacionalidade*, *residencia*, *equipa->Equipa*);
 - *Include attributes like player salary, years of experience, preferred champions/characters, etc.*
4. **Contrato** (*idContrato*, *jogador->Jogador*, *inicio*, *fim*);
 - *Remove the team reference from the contract, as a player can have contracts with different teams.*
5. **Epoca** (*idEpoca*, *ano*, *dataInicio*, *dataFim*);
 - *Combine period information into the season table for clarity.*
6. **Competicao** (*idCompeticao*, *nomeComp*, *tipo*, *numJogos*, *epoca->Epoca*);
 - *Connect the competition to a specific season.*
7. **Arena** (*idArena*, *nomeArena*, *localizacao*, *capacidade*);
 - *Keep arena information as is.*

8. **Jogo** (*idJogo, dataRealizacao, tipo, resultadoFinal, competicao->Competicao, arena->Arena, blueTeam->Equipa, redTeam->Equipa*);
 - Clearly define the participating teams in a game.
9. **Partida** (*idPartida, resultado, jogo->Jogo*);
 - Keep information about individual matches.
10. **ChampSelect** (*idChampSelect, partida->Partida, jogador->Jogador, pick, ban, runas*);
 - Add an identifier for champ select to link it with the specific game.
11. **Objetivo** (*idObjetivo, minuto, tipo, localizacao, partida->Partida, equipa->Equipa*);
 - Connect objectives to a specific team within a match.
12. **Eliminacao** (*idEliminacao, minuto, partida->Partida, jogadorElimina->Jogador, jogadorEliminado->Jogador*);
 - Clearly define the players involved in eliminations.
13. **Assistencia** (*idAssistencia, eliminacao->Eliminacao, jogadorAssistente->Jogador*);
 - Define assists with a reference to eliminations and the assisting player.

Dependências Funcionais e Análise das Formas Normais

De seguida, apresentaremos as dependências funcionais não triviais de cada uma das relações descritas anteriormente:

1. **Organizacao:**
 - idOrganizacao -> nomeOrg, regiao, localizacao
2. **Equipa:**
 - idEquipa -> nomeEquipa, organizacao
3. **Jogador:**
 - idJogador -> nomeJogador, position, nick, dataNascimento, nacionalidade, residencia, equipa
4. **PeriodoContratacao:**
 - idPeriodoContratacao -> dataInicio, dataFim
5. **Contrato:**
 - jogador, periodoContratacao -> equipa, inicio, fim
6. **Epoca:**
 - idEpoca -> ano, periodoDeContratacao
7. **Competicao:**
 - idCompeticao -> nome, tipo, numJogos
8. **EquipaEpocaCompeticao:**
 - Dependência funcional é trivial (equipa, epoca, competição são a chave primária desta relação e são os únicos atributos desta relação)
9. **Arena:**
 - idArena -> nome, localizacao, capacidade
10. **Jogo:**
 - idJogo -> dataRealizacao, tipo, resultadoFinal, competicao, arena, blue, red
11. **Partida:**

- idPartida -> resultado, jogo

12. ChampSelect:

- partida, jogador -> pick, ban, runas

13. Objetivo:

- idObjetivo -> minuto, tipo, localizacao, partida, destroi

14. Eliminacao:

- idEliminacao -> minuto, partida, jogElimina, jogEliminado

15. Assistencia:

- Dependência funcional é trivial (eliminacao, jogador são a chave primária desta relação e são os únicos atributos desta relação)

Todas as dependências funcionais respeitam a forma normal de Boyce-Codd (isto é, uma relação está na BCNF se, para cada dependência funcional não trivial $X \rightarrow Y$ da relação, X é uma super-chave), pois os elementos à esquerda de cada dependência funcional são super-chaves.

Isto também inclui a 3ª Forma Normal, que indica que para cada dependência funcional não trivial $X \rightarrow Y$, ou X é uma super-chave, ou Y consiste em atributos pertencentes a uma chave. Logo, como verifica a primeira condição, não há violações da 3ª Forma Normal.

Após esta análise, submetemos o nosso modelo relacional à ferramenta de AI e pedimos que elaborasse as dependências funcionais e que além disso concluísse quais das dependências respeitavam a forma normal de Boyce-Codd. Comparando a resposta obtida através deste pedido concluímos que a análise das dependências estava maioritariamente idêntica.

- *In BCNF, for every non-trivial functional dependency $X \rightarrow Y$, X must be a superkey.*

1. Organizacao:

- idOrganizacao -> nome, regioao, localizacao (BCNF: Yes, idOrganizacao is a superkey.)

2. Equipa:

- idEquipa -> nome, organizacao (BCNF: Yes, idEquipa is a superkey.)

3. Jogador:

- idJogador -> nome, position, nick, dataNascimento, nacionalidade, residencia, equipa (BCNF: Yes, idJogador is a superkey.)

4. PeriodoContratacao:

- idPeriodoContratacao -> dataInicio, dataFim (BCNF: Yes, idPeriodoContratacao is a superkey.)

5. Contrato:

- jogador, periodoContratacao, equipa -> inicio, fim (BCNF: Yes, jogador, periodoContratacao, equipa is a superkey.)

6. Epoca:

- idEpoca -> ano, periodoDeContratacao (BCNF: Yes, idEpoca is a superkey.)

7. Competicao:

- idCompeticao -> nome, tipo, numJogos (BCNF: Yes, idCompeticao is a superkey.)

8. EquipaEpocaCompeticao:

- equipa, epoca, competicao -> (no specific functional dependencies provided)

9. Arena:

- *idArena -> nome, localizacao, capacidade (BCNF: Yes, idArena is a superkey.)*

10. Jogo:

- *idJogo -> dataRealizacao, tipo, resultadoFinal, competicao, arena, blue, red (BCNF: Yes, idJogo is a superkey.)*

11. Partida:

- *idPartida -> resultado, jogo (BCNF: Yes, idPartida is a superkey.)*

12. ChampSelect:

- *partida, jogador -> pick, ban, runas (BCNF: Yes, partida, jogador is a superkey.)*

13. Objetivo:

- *idObjetivo -> minuto, tipo, localizacao, partida, destroi (BCNF: Yes, idObjetivo is a superkey.)*

14. Eliminacao:

- *idEliminacao -> minuto, partida, jogElimina, jogEliminado (BCNF: Yes, idEliminacao is a superkey.)*

15. Assistencia:

- *eliminacao, jogador -> (no specific functional dependencies provided)*

Criação da Base de Dados em SQLite

Após a criação da base de dados observada no ficheiro create1, o nosso grupo usou o ChatGPT para o melhorar, ao que a ferramenta respondeu para usarmos tipos de dados mais adequados, dando o exemplo de usar *DATE* em vez de *VARCHAR* para datas, para além de recomendar o uso de mais restrições como por exemplo:

```
-- Restraints for Organizacao Table
```

```
ALTER TABLE Organizacao
```

```
ADD CONSTRAINT check_nomeOrg_length CHECK (LENGTH(nomeOrg) > 0);
```

```
-- Restraints for PeriodoContratacao Table
```

```
ALTER TABLE PeriodoContratacao
```

```
ADD CONSTRAINT check_dataFim_after_dataInicio CHECK (julianday(dataFim) - julianday(dataInicio) > 0);
```

Após adaptarmos as recomendações do ChatGPT, disponibilizamos a nossa base de dados atualizada no ficheiro create2.

Preenchimento da Base de Dados

Elaboramos todos os comandos de *INSERT* do ficheiro populate1 com base no [website https://lol.fandom.com/wiki/2022_Season_World_Championship](https://lol.fandom.com/wiki/2022_Season_World_Championship). O conteúdo deste ficheiro representa apenas uma parte da informação acerca do evento Worlds 2022 de *League of*

Legends, de forma a popular todas as tabelas em número suficiente para a deteção de eventuais erros.

De seguida, usamos a ferramenta de Inteligência Artificial para formular alguns exemplos para cada classe, de forma a aperfeiçoar o preenchimento da base de dados. Após a análise desta resposta concluímos que apesar de a maioria dos exemplos estarem corretos, havia erros, como por exemplo:

```
INSERT INTO Jogo VALUES (1, '2022-10-01', '1', '3-2', 1, 1, 1, 2):
```

- Em que apesar do tipo do jogo ser “à melhor de 1”, o ChatGPT apresenta o resultado como 3-2, o que conceptualmente é ilógico e impossível.

```
INSERT INTO Partida VALUES (1, '3-2', 1);
```

- De forma análoga, o resultado da partida é igual ao resultado do jogo e deveria ser, por exemplo, ‘Blue Wins’ ou ‘Red Wins’.

Tendo em consideração os exemplos e as imperfeições, corrigimos os comandos de *INSERT* no ficheiro *populate2*.

Durante a segunda parte deste projeto, mantivemos uma interação forte com a ferramenta de Inteligência Artificial. Os resultados que obtivemos com esta ferramenta, após a sua integração nos diversos temas, foram úteis. No entanto, é necessário avaliar as respostas fornecidas com conhecimentos prévios e realizar várias tentativas de modo a aperfeiçoar cada resposta à situação descrita. Assim, de forma contrastante com a conclusão da primeira submissão, verificamos que o ChatGPT se mostrou uma ferramenta benéfica na criação e preenchimento da base de dados, oferecendo sugestões indispensáveis ao seu desenvolvimento.