Ciências ULisboa

Bases de Dados Avançadas
2023/2024
Grupo 04

Relatório do Projeto

Realizado por:
Ana Teixeira - fc56336
Daniel Lopes - fc56357
Miguel Ramos - fc56377
Tomás Piteira - fc56303

Descrição do Dataset

O dataset é composto por três ficheiros csv principais: results.csv, shootouts.csv e goalscorers.csv.

O ficheiro results.csv fornece informações sobre os resultados dos jogos de cada torneio e tem as seguintes colunas:

- date - data do jogo
- home_team - nome da equipa da casa
- away_team - nome da equipa visitante
- home_score - a pontuação total da equipa da casa, excluindo pênaltis
- away_score - a pontuação total da equipa visitante, excluindo pênaltis
- tournament - nome do torneio
- city - nome da cidade ou vila onde foi realizado o jogo
- country - nome do país onde foi realizado o jogo
- neutral - TRUE/FALSE indicando se o jogo foi disputado num local neutro

O ficheiro shootouts.csv fornece informações sobre os resultados dos jogos que foram a penaltis e tem as seguintes colunas:

- date: data do jogo
- home_team - nome da equipa da casa
- away_team - nome da equipa visitante
- winner - nome da equipa que venceu a disputa de penaltis
- first_shooter - nome da equipa que começou a disputa de pênaltis

O ficheiro goalscorers.csv fornece informações sobre os jogadores que marcaram golos no jogo e tem as seguintes colunas:

- date: data do jogo
- home_team - nome da equipa da casa
- away_team - nome da equipa visitante
- team - nome da equipa que marcou o golo
- scorer - nome do jogador que marcou o golo
- own_goal - TRUE caso tenha sido autogolo, FALSE caso contrário
- penalty - TRUE caso o golo tenha sido marcado num pênalti, FALSE caso contrário

dataset_link:https://www.kaggle.com/datasets/martj42/international-football-results-from-1872-to-2017/data

Esquema para as duas databases antes da otimização

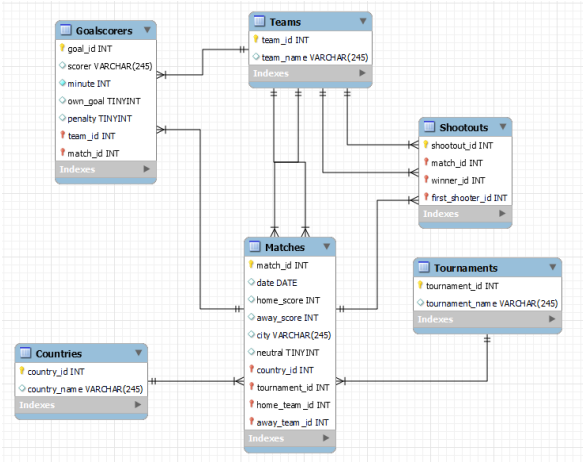

Figura 1: Esquema da base de dados relacional MySQL, antes da otimização

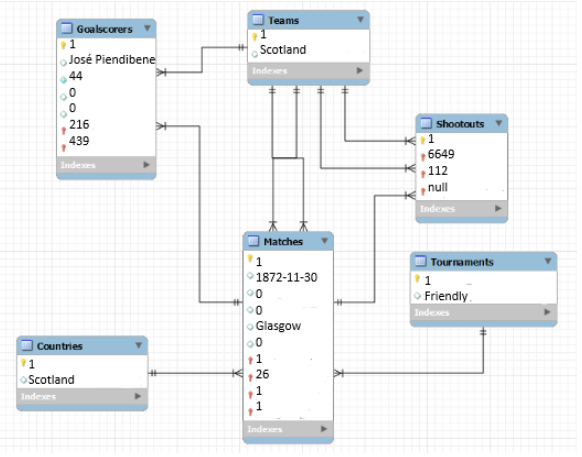

Figura 2: Um exemplo do esquema dos dados antes da otimização, em MongoDB

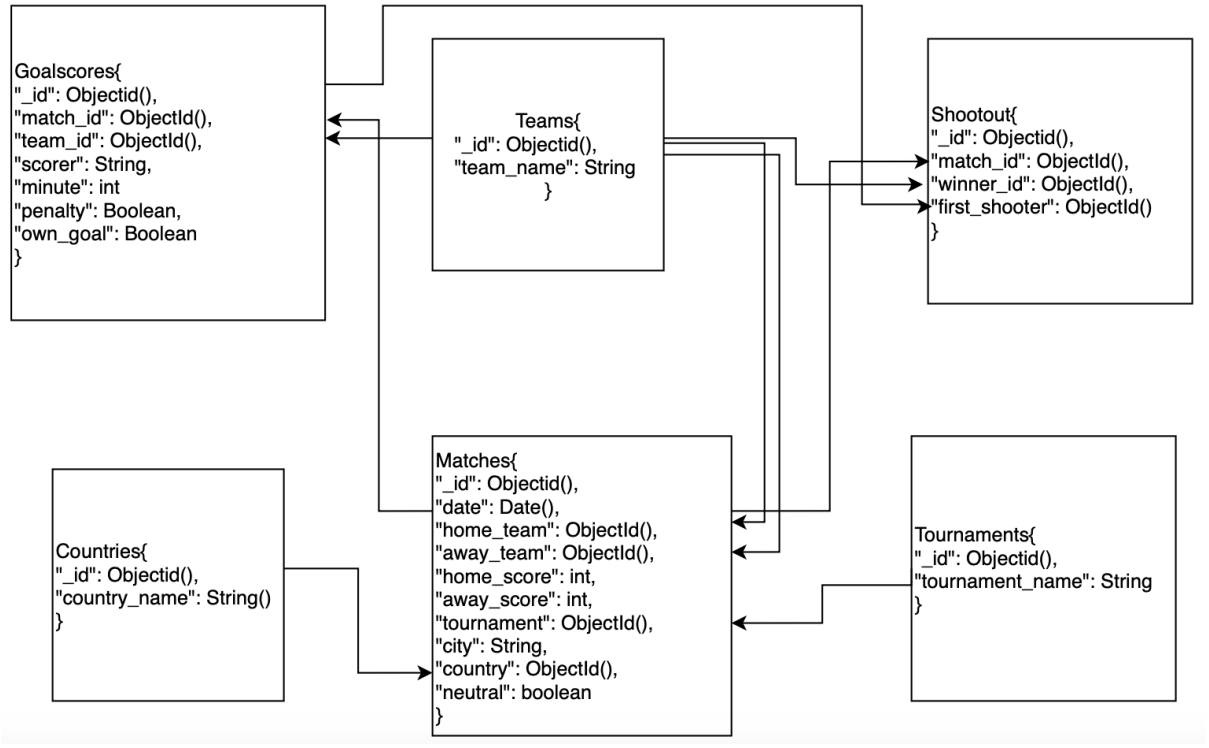

Figura 3: Esquema da base MongoDB, antes da otimização

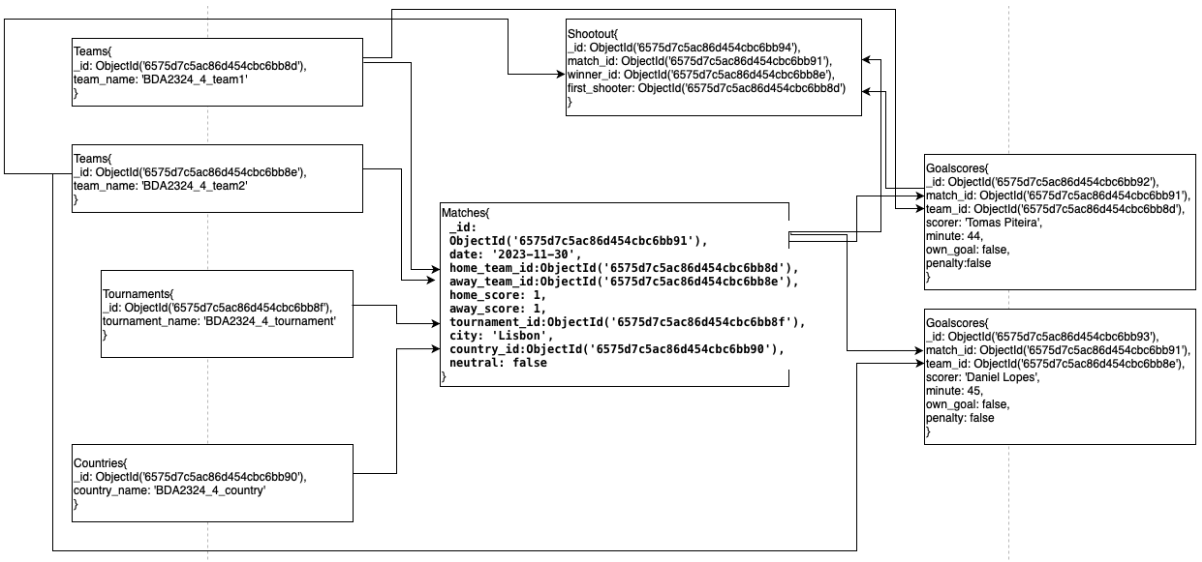

Figura 4: Exemplo do esquema dos dados inseridos no ponto 3 do enunciado, antes da otimização em, MongoDB

Queries Implementadas

- Query simples - irá obter todos os jogos que foram realizados em Lisbon, no ano de 2023
- Query simples - irá obter todos os golos marcados pelo Cristiano Ronaldo antes dos primeiros 5 minutos de jogo.
- Query Complexa - irá obter os primeiros 5 jogadores de Portugal, por ordem decrescente com mais golos no Torneio: FIFA World Cup Qualification, cujo os jogos foram realizados em Portugal e que esses jogos podem ter ou não ter sido disputados em pênaltis.
- Query Complexa -irá obter todos os golos marcados por Cristiano Ronaldo antes dos primeiros 5 minutos de jogo
- Update - irá obter da base de dados todos os jogos que foram realizados na data 1882-02-18 e atualiza o resultado de home_score, away_score e home_team.
- Insert - insere novas informações relativas a um novo jogo realizado entre duas novas equipas “BDA2324_4_team1” e “BDA2324_4_team2”. Como houve golos no jogo, é preciso também inserir na tabela goalscores as informações sobre cada golo marcado. Como o resultado foi um empate, o jogo teve de ir aos pênaltis, pelo que foi preciso inserir a informação relativa aos pênaltis na tabela shootouts.

Indexing e Otimização

1. **Reestruturação dos dados:**

Relativamente à otimização, a versão inicial apresentava ser uma representação mais estruturada e organizada da base de dados, mas devido ao facto de a performance das queries não ser a melhor uma vez que é preciso fazer um maior número de Joins/Aggregates para receber os dados necessários das outras tabelas. Como algumas destas tabelas tinham apenas um conteúdo do tipo *string* como “teams” , “countries” e

“tournaments”, optamos por não criar tabelas à parte para esses dados, inserindo-os diretamente na tabela principal para otimizar o programa. Esse foi um grande ponto de melhoria no nosso programa, tal como é possível ver através dos resultados dos boxplots anexados dentro da pasta *bd_diagrams*, em que é possível reparar que a média de tempo de execução de cada query é bastante inferior após a otimização e reestruturação do projeto.

2. Indexes:

Foram implementados indexes de modo a otimizar as queries e fazer uma procura mais rápida e seletiva dos dados. Os indexes utilizados em MongoDB e em MySQL são os mesmos e foram testados diferentes combinações de indexes de forma a verificar quais os que melhoram a performance das queries.

Esquema para as duas databases depois da otimização

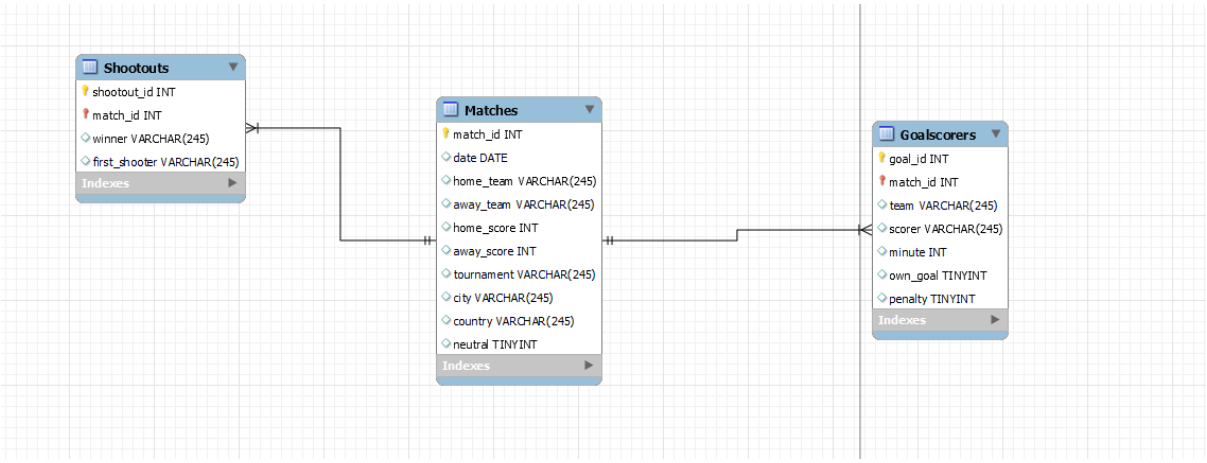

Figura 5: Esquema da base de dados relacional MySQL, depois da otimização

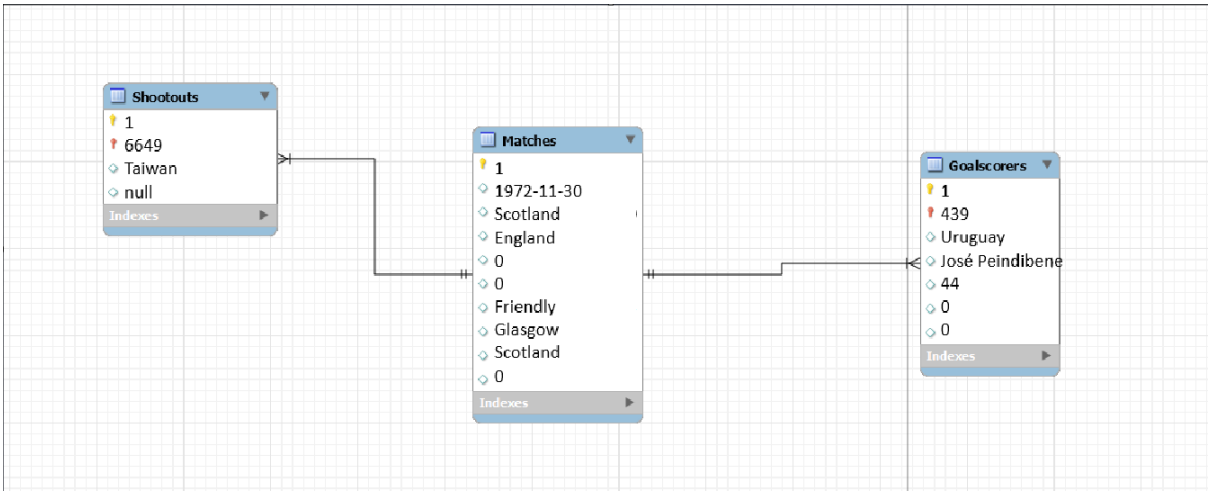

Figura 6: Exemplo do esquema dos dados, depois da otimização em MySQL

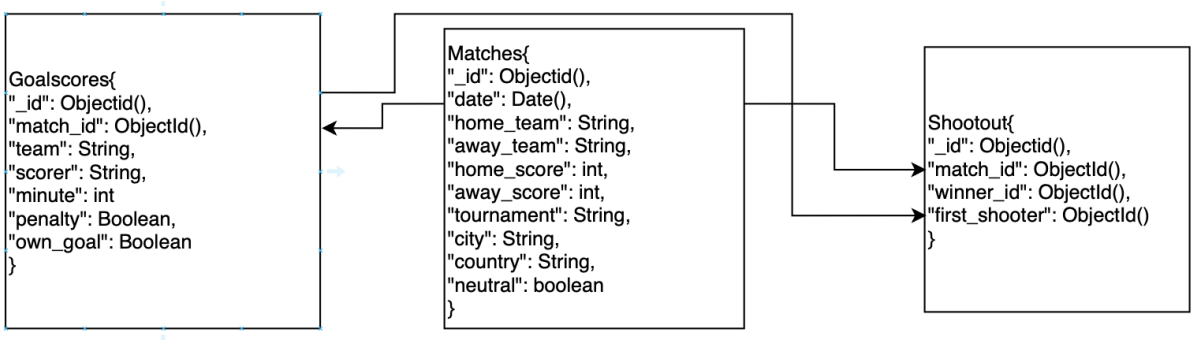

Figura 7: Esquema da base de dados MongoDB, depois da otimização

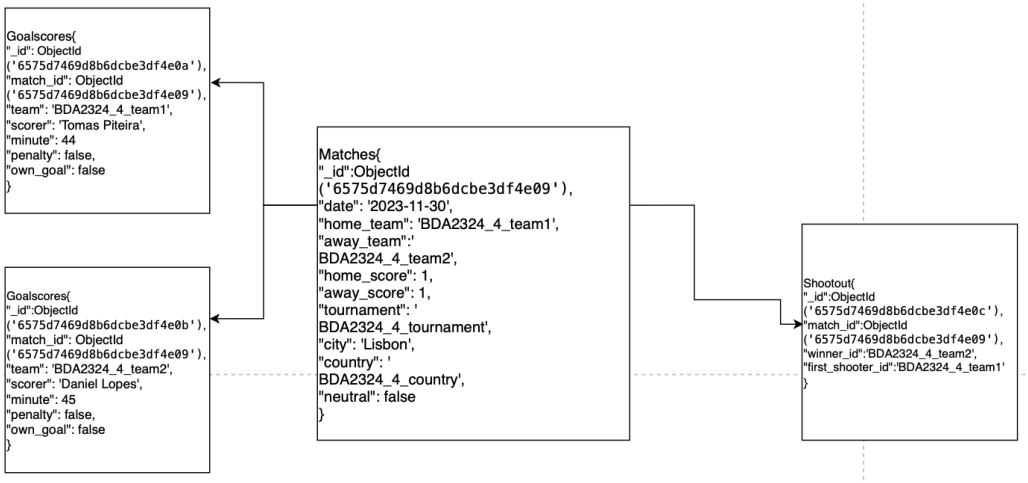

Figura 8: Exemplo do esquema dos dados inseridos no ponto 3 do enunciado, depois da otimização em MongoDB

Discussão dos pontos feitos/ não feitos no projeto

Todos os objetivos do enunciado foram concluídos. O projeto apresenta duas versões, uma antes da otimização (versão inicial) e outra após a sua otimização. Foram feitos ficheiros de benchmarking de modo a executar o programa 30 vezes para cada uma das versões e verificar os tempos de execução de cada query antes e depois da aplicação de índices.

Descrição da estrutura do projeto

De modo a ter uma melhor organização do projeto, foram separados os objetivos do projeto em três ficheiros diferentes para cada base de dados: um para a inserção dos dados nas bases de dados, um para a realização das queries e outro para a realização das queries utilizando indexes.

De modo a entender melhor o funcionamento do código antes e depois da otimização, foram criadas duas pastas denominadas por *after_optimization* e *before_optimization*. Cada uma tem 3 ficheiros que correspondem à execução do projeto utilizando MongoDB, outros 3 ficheiros que correspondem à execução do projeto utilizando MySQL e 1 ficheiro que permite aceder aos dados dos CSVs. Foram criados os ficheiros *performance_mongo.csv* e *performance_mysql.csv* que contêm os tempos de execução de

cada query, sendo que as primeiras quatro colunas correspondem aos tempos de execução das queries sem qualquer tipo de otimização/estruturação da base de dados e sem o uso de indexes e as últimas 4 colunas correspondem às mesmas queries após a reestruturação da base de dados e com a aplicação dos indexes. Assim, os boxplots são formados através destes dois ficheiros.

Descrição de como replicar o projeto

1) Estando no diretório BDA2324_G04, executar o comando *python .\x_optimization\mydb_creation.py* no terminal ou se for utilizado o VisualStudioCode clicar em “Run Python File” para inserir os dados na base de dados.
2) Executar o comando *python .\x_optimization\mydb_queries.py* no terminal ou se for utilizado o VisualStudioCode clicar em “Run Python File” para obter os resultados das queries realizadas.
3) Executar o comando *python .\x_optimization\mydb_indexes.py* no terminal ou se for utilizado o VisualStudioCode clicar em “Run Python File” para obter os resultados das queries realizadas utilizando índices.

Notas:
1) Mudar *x* para *before* ou para *after* consoante o tipo de ficheiro que se está a executar (estrutura da base de dados não otimizada ou otimizada)
2) Mudar *mydb* para *mongo* ou para *mysql* dependendo da base de dados a utilizar.
3) Exemplo: Executar o ficheiro mongo_queries.py antes da otimização: executar o comando *python .\before_optimization\mongo_queries.py* no terminal
4) Os comandos mencionados neste relatório podem variar consoante a instalação do python e do sistema operativo utilizado.