Universidade do Minho
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2021/2022

Escola de Futebol

Ana Beatriz Silva (A91678), Beatriz Oliveira (A91640), Francisco Teófilo (A93741), Henrique Faria (A91637)

Janeiro, 2022

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Escola de Futebol

Ana Beatriz Silva (A91678), Beatriz Oliveira (A91640), Francisco Teófilo (A93741), Henrique Faria (A91637)

Janeiro, 2022

Resumo

Este trabalho tem como objetivo aperfeiçoar o processo de criação e de desenvolvimento de procedimentos operacionais e, ainda, planear e executar um projeto de sistema de bases de dados não relacionais, *NoSQL*, através da migração de dados entre diferentes sistemas de dados.

Neste presente relatório descrevemos todo o processo de migração da base de dados relacional criada anteriormente, Escola de Futebol. Primeiramente, iremos apresentar o sistema relacional implementado. Posteriormente, iremos fazer uma avaliação relativamente à normalização da base de dados já criada e verificaremos se existe a possibilidade de melhorar a eficiência da mesma através da indexação.

No último capítulo, iremos abordar a conceção e a implementação de um sistema de base de dados em MongoDB, em que definimos a Base de Dados e apresentamos o esquema da coleção. Finalmente, apresentamos o processo de migração de migração de dados e estrutura de MySQL para MongoDB.

Área de Aplicação: Desenho, Arquitetura, Desenvolvimento e Implementação de Sistemas de Bases de Dados.

Palavras-Chave: Bases de Dados, Bases de Dados Não Relacionais, Normalização, Queries, MongoDB, MySQL Workbench, NoSQL.

Índice

Índice de Figuras

1. Sistema Relacional Implementado

1.1 Apresentação do sistema

O Sistema apresentado na primeira parte deste projeto foi criado para ajudar a gestão de dados e informação da Escola de Futebol Maximinense, cujos métodos de armazenação de dados estavam um pouco desatualizados. A Base de Dados elaborada, cujos objetivos envolvem uma melhor organização, manutenção e manipulação de dados, forneceu um modo mais simples de aceder à informação nela contida, e tornou mais fácil o trabalho da escola, trabalho este que envolve a criação de uma comunidade entre todos aqueles unidos pelo futebol.

A base de Dados em si é composta por seis entidades: Encarregado de Educação, Escalão, Treinador, Fisioterapeuta, Equipa e Jogador; cada uma destas com os seus devidos atributos. Existem diversas relações entre as diferentes entidades, como por exemplo, a relação Treinador – Equipa, que nos diz que treinador treina que equipa; a relação Escalão-Jogador, que nos fornece a informação sobre o escalão dos diversos jogadores; entre outras. Há também uma variedade de atributos em cada entidade, sendo talvez o melhor exemplo disto a entidade Jogador, onde podemos ver atributos do tipo INT, VARCHAR (45), e DATE.

A validação do modelo lógico da base de dados foi feita usando as Interrogações do Utilizador, ou seja, foram criadas diferentes queries, como por exemplo, a querie que conta o número de jogadores de cada localidade, uma que nos fornecia a lista de treinadores com o contacto diferente de ‘969876543’, e ainda uma que nos dá o fisioterapeuta da equipa com id igual a '57', entre múltiplas outras, para garantir que a base de dados funcionava corretamente e cumpria os requisitos iniciais.

Esta segunda parte do projeto vai incluir, novamente, a validação da base de dados, mas desta vez a partir da normalização. Vamos ainda explorar a indexação da Base de Dados e desenvolver um conjunto de procedimentos que iram realizar diversas tarefas, e também criar um processo de migração da base de dados relacional criada, para MongoDB, onde vão ser implementadas queries equivalentes às mencionada anteriormente.

1.2 Normalização de Dados

A normalização é, por palavras simples, o processo de organização de dados de uma base de dados. Tem como objetivo identificar, e estabelecer, as relações entre as diversas tabelas, de forma que a base de dados seja flexível, mas também de modo a eliminar redundâncias e alguma dependência de atributos que seja inconsistente.

Tendo em conta que a normalização está a ser explorada na segunda parte do projeto, quando iniciamos esta fase, a base de dados já estava construída. Visto isto, basta verificar se esta está normalizada segundo as diferentes formas de normalização.

Primeira Forma Normal (1FN): A primeira forma normalizada acontece quando todos os atributos são atómicos, ou seja, não há valores repetidos dentro das tabelas, nem atributos multivalorados. Podemos dizer que o modelo lógico está normalizado em 1FN porque estas condições se verificam.

Segunda Forma Normal (2FN): Uma relação está na segunda forma normalizada quando estava previamente na primeira forma normalizada, e quando todos os seus atributos, que não são chaves primárias, são totalmente dependentes da chave primária. Se tivermos em mente, por exemplo, a relação Equipa para Treinador, caso as informações destas tabelas estivessem agrupadas numa só, por exemplo, na tabela Equipa, que nos daria automaticamente o treinador, iria haver informações que não estariam relacionadas com o ID da equipa, como o código postal do treinador. Estas informações estarem em tabelas diferentes, contribui para a normalização da Base de Dados, logo, esta encontra-se normalizada em segunda forma.

Terceira Forma Normal (3FN): Uma relação que se encontra na terceira forma normalizada estava, anteriormente, em ambas a primeira e segunda formas normalizadas, com a adição de que, na 3FN, não deve haver atributos que são dependentes de outros, ou gerados a partir de outros. No caso da nossa base de dados, não há nenhum atributo que seja capaz de se obter a partir de outro, nem há dependências entre eles, logo está normalizada em 3FN.

Assim, podemos afirmar que a Base de Dados construída para a escola Maximinense se encontra normalizada até à Terceira Forma de Normalização.

1.3 Indexação do Sistema de Dados

A utilização de índices em SQL permite reduzir o tempo de procura de informação numa certa Base de Dados, no entanto, provoca um maior consumo do espaço disponível no armazenamento da mesma. Por predefinição, sempre que criamos um objeto, o SQL gera, automaticamente, índices para determinados atributos, como por exemplo, para *Primary Keys*.

Todavia, consideramos, necessário, a criação de outros índices de forma a aprimorar a execução da Base de Dados criada. Relativamente a tabelas como o Escalão e a Localidade, que não serão modificadas com frequência, julgamos que não é necessário a criação de índices, uma vez que os índices são mais eficientes em tabelas com elevado número de entradas.

Assim sendo, uma boa opção para introduzir índices será na tabela Jogador. Uma vez que se trata de uma tabela com elevado número de entradas, de atributos, e por isso com um grande número de pesquisas de uma determinada informação utilizando, preferencialmente, ou o ID ou o nome do Jogador. Pelo que iremos criar um índice para o atributo nome, fazendo:

```
CREATE INDEX NomeJogador ON jogador (nome);
```

Figura 1 - Comando utilizado para criar o índice NomeJogador na coluna nome da tabela jogador

Pensando no crescimento exponencial de inscrições que a Escola está a ter e no futuro da Escola, consideremos, ainda, ser necessário a criação de mais dois índices, um na tabela Encarregado de Educação, no atributo Nome, e outro na tabela Treinador, também no atributo Nome. Uma vez que, por cada Jogador inscrito, é necessário ter os dados do seu Encarregado de Educação, assim sendo, o crescimento exponencial do número de Jogadores, implica um crescimento exponencial de dados sobre os Encarregados de Educação. O aumento exponencial de Jogadores implica um aumento, mais discreto, do número de equipas e, indiretamenta, implica a necessidade de um maior número de Treinadores.

```
CREATE INDEX NomeEE ON encarregado_de_educacao (nome);
```

Figura 2 - Comando utilizado para criar o índice NomeEE na coluna nome da tabela encarregado_de_educacao

```
CREATE INDEX NomeT ON treinador (nome);
```

Figura 3 - Comando utilizado para criar índice NomeT na coluna nome da tabela treinador

1.4 Procedimentos Implementados

1. Obter os Jogadores da localidade dada como argumento

```
DELIMITER $$
CREATE PROCEDURE JogadoresdaLocalidade (In nlocalidade INT)
BEGIN
    SELECT * FROM Jogador
        WHERE Localidade = nlocalidade;
END $$

CALL JogadoresdaLocalidade('4');
```

Figura 4 - Procedimento que permite obter os jogadores de uma localidade fornecida como argumento e a chamada do procedimento

2. Calcula a diferença de anos entre o dia de hoje e a data de nascimento de um dado jogador

```
DELIMITER $$
CREATE PROCEDURE JogadorIdade (In n_jogador INT)
BEGIN
    SELECT year(date(now()))- year(Data_de_Nascimento) FROM Jogador
        WHERE Jogador.Id = n_jogador;
END $$

CALL JogadorIdade('4');
```

Figura 5 - Procedimento que calcula o número de anos que separa a data corrente da data de nascimento de um dado jogador e a respetiva chamada do procedimento

3. Contar o número de equipas que um dado treinador treina

```
DELIMITER $$
CREATE PROCEDURE EquipasTreinadasPor (In n_treinador INT)
BEGIN
    SELECT Count(*) FROM Treinador
        INNER JOIN Equipa as E on E.Treinador_ID = Treinador.ID
        WHERE Treinador.Id = n_treinador;
END $$

CALL EquipasTreinadorPor('5');
```

Figura 6 - Procedimento que conta o número de equipa treinadas pelo treinado cujo identificador é n_treinador e a respetiva chamada do procedimento

2. Conceção e Implementação de um Sistemas de Dados em MongoDB

2.1 Definição do Esquema da Base de Dados

O MongoDB é uma base de dados não relacional orientada por documentos que pertence aos modelos NoSQL. Estes são conhecidos por ter esquemas de dados flexíveis, pela sua velocidade e desempenho, por serem altamente escaláveis e, ainda, pela facilidade que fornece na manipulação de dados.

O MongoDB permite-nos criar relações de duas formas:

- **Referências**: podemos referenciar documentos de uma coleção noutras.
- **Documentos Embutidos**: onde temos todas as referências dentro de um único documento. Essa relação é vista como um subdocumento, pois os dados são embutidos em arrays ou campos do documento.

Devido a isto, foi feita uma migração da base de dados do MySQL para o MongoDB, onde todos os dados presentes no modelo relacional foram mantidos, apesar de terem ocorrido alterações à sua estrutura para garantir o melhor desempenho do modelo.

2.2 Criação da Base de Dados e das Coleções

Para implementarmos a Base de Dados, Escola de Futebol, em MongoDB, decidimos criar apenas três coleções: Equipa, Jogador e Treinador.

Apresentamos abaixo imagens meramente representativas das diferentes coleções, visto não ser possível mostrar todos os documentos presentes nas mesmas.

```
> db.Equipa.find().pretty()
{
        "_id" : ObjectId("61ec4103922e9304d2d1f42d"),
        "ID" : 57,
        "Classificacao" : 1,
        "Treinador_ID" : 5,
        "Fisioterapeuta" : {
                "ID" : 55,
                "Nome" : "Pedro Teixeira"
        }
}
```

Figura 7 - Exemplo da coleção Equipa

Figura 8 - Exemplo da coleção Jogador

Figura 7 - Exemplo da coleção Treinador

2.3 O Processo de Migração de Dados

Visto que a nossa Base de Dados contém tabelas como Localidade, Fisioterapeuta e Encarregado_de_Educacao, que são tabelas pouco povoadas, decidimos simplesmente fazer a transformação destas através do MySQL Workbench para ficheiros JSON, de forma a ser possível importá-las para MongoDB Compass.

Foi preciso, no entanto, fazer algumas alterações (“à mão”) de forma a não manter na sua totalidade o modelo relacional, e também de forma a tornar as queries mais eficientes.

2.4 Exploração de Dados em MongoDB

1. db.Jogador.find({"Encarregado_de_Educacao.Nome":{$regex:/^M/i}}).pretty()
2. db.Treinador.find({Contacto:"969876543"}).pretty()
3. db.Equipa.find({ID:57},{"Fisioterapeuta.Nome":1}).pretty()
4. db.Jogador.distinct("Data_de_Nascimento").length
5. db.Jogador.aggregate({$match:
   {"Encarregado_de_Educacao.Nome":
   {$ne:"Maria Joaquina Teixeira"}}}, {$group:{_id:null,sum:
   {$sum:"$Encarregado_de_Educacao .Mensalidade" } } })
6. db.Jogador.aggregate([{"$group" : {_id:"$Localidade.Descricao",
   count:{$sum:1}}},{$sort:{"count":1}}])

Nota: Estas queries são equivalentes às da parte um, e encontram-se pela mesma ordem.

3. Conclusões e Trabalho Futuro

Através da realização deste projeto, constatamos que os nossos níveis de compreensão da matéria lecionada aumentaram exponencialmente, apesar de o trabalho apresentado não estar perfeito, sentimos que durante o processo da sua realização, efetuaram-se melhorias relativamente ao que tínhamos previamente.

O maior desafio neste trabalho fomos nós próprios, uma vez que detetamos vários erros que foram cometidos durante a primeira parte do projeto, que com a sua correção, levaram a uma perda significativa do tempo disponível para a execução da segunda parte. Estas correções encontram-se no fim deste relatório.

Na mencionada segunda parte, foi pedida a abordagem do problema original, mas desta vez em NoSQL, em particular, utilizando o MongoDB.

Existem várias diferenças entre o SQL e o NoSQL, sendo possivelmente a maior destas o facto de o SQL ser um sistema de base de dados relacional que possui uma própria linguagem de consulta estruturada, enquanto o NoSQL trabalha com bases de dados não relacionais, ou distribuídas. Outra diferença relevante entre os dois, é que o SQL é baseado num sistema de tabelas, ao passo que o NoSQL é baseado em documentos.

Para alem destas diferenças através da nossa pequena experiência com ambos os modelos, consideramos que ambos tem as suas vantagens. Relativamente ao tratamento e correção de erros, achamos ser mais prática a utilização do MongoDB. Por outro lado, o SQL garante uma maior uniformização da Base de Dados.

Posto isto, e concluída a migração da Base de Dados do MySQL para o MongoDB, consideramos que foi desafiante perceber de que maneira deveríamos organizar o material presente de forma que a Base de Dados fosse o mais eficiente possível. Após algumas alterações, e um pouco de pesquisa, julgamos ter obtido o melhor modelo possível para a Base de Dados no MongoDB.

Por fim, tendo em consideração os desafios que enfrentamos, consideramos que o nosso trabalho como grupo melhorou, o que levou a uma melhor compreensão de todas as particularidades do trabalho.

Referências

Connolly, T., Begg, C., 2014. Database Systems: A Practical Approach to Design, Implementation, and Management. Addison-Wesley, Global Edition.

Belo, O., 2021. Bases de Dados Relacionais: Implementação com MySQL. FCA, Editora de Informática.

Gouveia, F., 2021. Bases de Dados - Fundamentos e Aplicações. 2ª Ed. FCA, Editora de Informática.

MongoDB, 2021. Reference Manual. [online] Available at:
What is MongoDB? — MongoDB Manual [Accessed 22 of January of 2022]

Lista de Siglas e Acrónimos

BD	Base de Dados
SQL	*Structured Query Language*
NoSQL	*Unstructured Query Language*