Universidade do Minho
Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2017/2018

Base de Dados do Agrupamento de Escuteiros 424-Nogueira

Frederico Pinto a73639,

Ricardo Canela a74568,

Ricardo Leal a75411,

Lucas Ribeiro a68547

Janeiro, 2018

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Base de Dados do Agrupamento de Escuteiros 424-Nogueira

Frederico Pinto a73639,

Ricardo Canela a74568,

Ricardo Leal a75411,

Lucas Ribeiro a68547

Janeiro, 2018

Resumo

Neste relatório é descrito todos os passos da migração da base de dados relacional, existente no Agrupamento de Escuteiros 424 – Nogueira. Numa primeira parte é apresentada a contextualização do nosso projeto e os seus principais objetivos. Posteriormente é feita uma introdução ao novo paradigma dos Sistemas de Base de Dados, NoSQL e no final é apresentada uma breve descrição sobre o MongoDB. No último capítulo é apresentado o processo de migração, apresentamos o esquema do modelo lógico da base de dados já existente e posteriormente apresentamos o esquema da coleção existente. Por fim é explicado como foi feita a migração de dados e estrutura de MySQL para MongoDB.

Em anexo está presente um lista de queries efetuadas sobre a base de dados em MongoDB e o script utilizado para efetuar a migração de dados.

Palavras-Chave: Bases de Dados Relacionais, Sistemas de Gestão de Base de Dados, SQL, MongoDB, NoSQL;

Índice

Índice de Figuras

1. Introdução

1.1. Contextualização

O agrupamento de escuteiros 424, que conta já com vários anos de existência, tem visto um aumento exponencial do número de elementos de ano para ano. O aumento da popularidade dos escuteiros tem feito com que novos jovens curiosos, exploradores e dinâmicos ingressem nas diferentes secções do agrupamento. Com o passar dos anos, estes jovens agora mais crescidos mantêm a sua lealdade ao agrupamento, com os valores que nos escuteiros lhe são transmitidos na maioria dos casos tomam a decisão de se manterem ativos no Agrupamento. Com a população mais velha do Agrupamento a manter-se e um enorme número de novos escuteiros a surgir, este agrupamento de escuteiros sofreu de um "BOOM" no número de elementos e consequentemente equipas.

Dado este acontecimento diversos problemas surgiram na base de dados que armazena informações do agrupamento, tendo o chefe do agrupamento de escuteiros reunido com a equipa responsável pela gestão da Base de Dados. Após a o Chefe do Agrupamento reportar diversos problemas como instabilidade de dados e lentidão de acesso à base de dados, a equipa responsável pela gestão da Base de Dados realizou uma análise meticulosa. Após algum tempo a debruçar-se sobre o assunto a equipa informou o chefe que iria mudar a estrutura da Base De Dados, sugerindo a BD NoSQL (MongoDB), que permitiria resolver os problemas descritos e suportar o crescente número de elementos do Agrupamento.

1.2. Fundamentação da Implementação da base de dados

Devido ao número crescente de escuteiros a inscrever-se, os chefes decidiram investir um pouco mais de dinheiro do Agrupamento ao mudar a sua base de dados relacional para uma base de dados NoSQL para conseguirem lidar com a informação de maneira

mais rápida. Apesar desta mudança, o objetivo da nova base de dados é exatamente igual ao da primeira, guardar toda a informação relevante ao Agrupamento.

1.3. Análise de Viabilidade do Processo

Tendo em conta o pedido do agrupamento face ao aumento de adesões de elementos a opção de transitarmos para uma base de dados NoSQL, tem um potencial de crescimento muito maior do que uma base de dados relacional. Uma base de dados que não é tanto rígida, que tem um melhor desempenho na inserção de dados, que garante também um melhor desempenho na consulta e transação de informação, entre outros foi o que nos levou a optar por este tipo de base de dados e que vai de encontro ao pedido que nos foi realizado. Comprovamos assim que o processo é viável e está bem justificado.

2. Introdução ao novo paradigma NoSQL

2.1. Opção por Sistemas NoSQL no Agrupamento 424

Como o problema abordado apresentou um elevado número de elementos a inserir na base de dados, com um SGBDR , o armazenamento de dados e a manutenção das relações, entre os mesmos, torna-se ineficiente e a alteração da estrutura demorada. NoSQL pode oferecer vantagens em problemas flexíveis, estruturados de maneira dinâmica. Com uma estrutura dinâmica e sem qualquer dependências entre elementos, a facilidade de armazenamento, o acesso e as transações, de um elevado número de informação, é melhorada. Este dinamismo permite fáceis alterações nessa mesma estrutura. Utilizando uma ferramenta apropriada de NoSQL podemos obter a capacidade de simplificar a base de dados onde toda a informação relacionada com cada elemento, está organizada numa única identidade.

Apesar da perda de algumas características como atomicidade e consistência de um esquema SGBDR é solucionado um problema, mais significativo, de performance com o aumento de unidades.

2.2. MongoDB

Um Sistema de Gestão de Base de Dados (NoSQL) pode tirar partido de algumas características fulcrais de acordo com o problema proposto. O MongoDB guarda toda a informação num unico documento, livrando-se de esquemas, atribuir identificadores únicos que permitem métodos mais eficientes de agrupamento e filtragem de informação. Cada documento pode tomar uma estrutura diferente de todos os outros, atribuindo toda a responsabilidade da manutenção e consistência da base de dados para

o gestor da mesma que insere e armazena dados em linguagem JSON. Este novo paradigma relativamente aos tradicionais SGBDR's, como já realçado, importa algumas consequências benéficas e prioritárias, tais como:

- Independência entre documentos:

  O facto de os documentos se comportarem como unidades independentes, permite que estes sejam distribuidos por vários clusters permitindo uma maior escalabilidade e um aumento de desempenho.

- Facilidade de armazenamento:

  Como MongoDB não tem esquemas, não existem condições a serem verificadas o que representa um desempenho muito maior nas inserções comparativamente com um SGBD.

- Eficiência em larga escala: Consulta e Transações;

Como não existem relacionamentos, um documento tem todas as informações necessárias, ou seja, as consultas/transações são mais simples e fáceis.

Apesar de tudo tem algumas desvantagens como:

- Inconsistência:

  O MongoDB não garante a consistência dos dados. Cabe ao gestor da base de dados garantir que todas as inserções ou alterações nos dados não perdem a sua consistência.

- Redundância de dados:

  O facto de não existirem relacionamentos entre documentos implica que hajam muitos dados repetidos.

3. Processo de Migração

3.1. Diagrama do Modelo Lógico

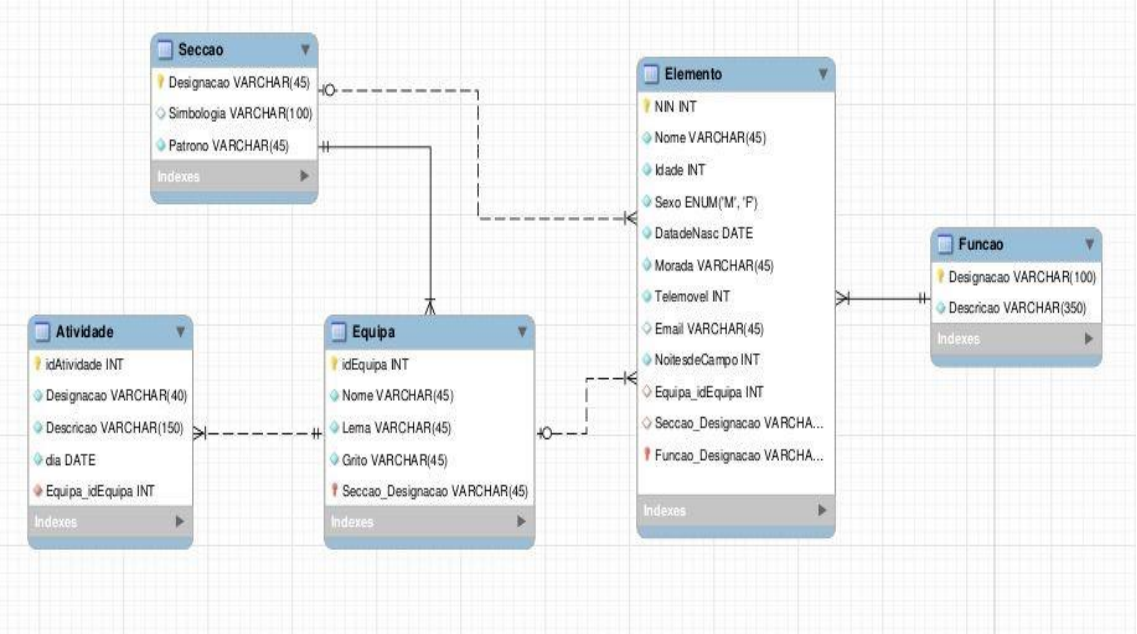

Fig.1 - Modelo Lógico

3.2. Estrutura dos Documentos

Para implementar a Base de Dados em MongoDB, decidimos criar apenas uma coleção: Elemento. Esta coleção contém todos os atributos do elemento incluindo os dados da secção, equipa e atividades efetuadas por ele:

{

“Nin”: Número de Identificação do Elemento;

“Nome”: Nome do Elemento;

“Idade”: Idade do Elemento;

“Sexo”: Sexo do Elemento;

“Data de Nascimento”: Data de nascimento do Elemento;

“Morada”: Morada do Elemento;

“Telemovel”: Contacto de telemóvel do Elemento;

“Email”: Email do Elemento;

“Noites de Campo”: Noites de Campo que o Elemento tem;

“Função”: Função do Elemento no Agrupamento; **“Secção”:**
{

“Designação”: Nome da Secção a que o Elemento pertence;

“Simbologia”: Simbologia da Secção a que o Elemento pertence;

“Patrono”: Patrono da Secção a que o Elemento pertence;

},

“Equipa”: {

“Nome”: Nome da Equipa a que o Elemento pertence;

“Lema”: Lema da sua Equipa

“Grito”: Grito da sua Equipa **“Atividade”:**
[{

“Designação”: Nome da Atividade;

“Descrição”: Descrição da Atividade;

“Dia”: Dia da Atividade;

}] }

}

3.3. Migração dos dados de MySQL para MongoDB

O processo de migração de uma base de dados relacional para uma base de dados NoSQL, pode ser efetuado de várias maneiras. Optámos por fazer um script na linguagem Ruby, que recolhe os dados da base de dados relacional, utilizando queries em SQL, guardando os documentos num ficheiro .json.

Na nossa base de dados relacional, os chefes do agrupamento diferenciavam-se dos restantes elementos não só pela sua função, mas também por não terem equipa nem secção. Posto isto, para retirar a informação dos elementos que possuem equipa e secção utilizados duas queries:

```
elementos = client.query("SELECT E.NIN, E.Nome, E.Idade, E.Sexo, E.DatadeNasc, E.Morada, E.Telemovel, E.Email, E.NoitesdeCampo,
                                E.Funcao_Designacao AS Funcao,
                                E.Seccao_Designacao AS NomeSeccao,
                                Seccao.Simbologia AS Simbologia,
                                Seccao.Patrono AS Patrono,
                                E.Equipa_idEquipa AS IdEquipa,
                                Equipa.Nome AS EquipaNome,
                                Equipa.Lema AS Lema,
                                Equipa.Grito AS Grito
                                FROM Elemento AS E
                                INNER JOIN Seccao ON Seccao.Designacao = E.Seccao_Designacao
                                INNER JOIN Equipa ON Equipa.idEquipa = E.Equipa_idEquipa;")

elementos.each do |elemento|
    result = client.query("SELECT Atividade.Designacao AS Designação, Atividade.Descricao AS Descrição, Atividade.dia AS Dia
                           FROM Atividade
                           WHERE Atividade.Equipa_idEquipa = #{elemento['IdEquipa']}")

    res << {
        'Nin'   => elemento['NIN'],
        'Nome'  => elemento['Nome'],
        'Idade' => elemento['Idade'],
        'Sexo'  => elemento['Sexo'],
        'Data de Nascimento'    => elemento['DatadeNasc'],
        'Morada'    => elemento['Morada'],
        'Telemovel' => elemento['Telemovel'],
        'Email' => elemento['Email'],
        'Noites de Campo' => elemento['NoitesdeCampo'],
        'Função' => elemento['Funcao'],
        'Secção' => {'Designação' => elemento['NomeSeccao'], 'Simbologia' => elemento['Simbologia'], 'Patrono' => elemento['Patrono']},
        'Equipa' => {'Nome' => elemento['EquipaNome'], 'Lema' => elemento['Lema'], 'Grito' => elemento['Grito'],
                     'Atividade' => result.to_a}

    }
end
```

Fig.2 – Queries utilizadas para migrar os Elementos

A primeira query seleciona toda a informação da necessária para completar os dados do elemento no respetivo documento e a segunda seleciona todas as atividades efetuadas pela equipa a que o elemento pertence.

Para obter a informação dos chefes:

```
chefes = client.query("SELECT E.NIN, E.Nome, E.Idade, E.Sexo, E.DatadeNasc, E.Morada, E.Telemovel, E.Email, E.NoitesdeCampo,
                              E.Funcao_Designacao AS Funcao FROM Elemento AS E
                              WHERE E.Seccao_Designacao IS NULL AND E.Equipa_idEquipa IS NULL")

chefes.each do |chefe|

    che << {
        'Nin'   => chefe['NIN'],
        'Nome'  => chefe['Nome'],
        'Idade' => chefe['Idade'],
        'Sexo'  => chefe['Sexo'],
        'Data de Nascimento'    => chefe['DatadeNasc'],
        'Morada'    => chefe['Morada'],
        'Telemovel' => chefe['Telemovel'],
        'Email' => chefe['Email'],
        'Noites de Campo' => chefe['NoitesdeCampo'],
        'Função' => chefe['Funcao'],
        'Secção' => {'Designação' => nil, 'Simbologia' => nil, 'Patrono' => nil},
        'Equipa' => {'Nome' => nil, 'Lema' => nil, 'Grito' => nil,
                     'Atividade' => nil}

    }
end
```

Fig.3 – Query utilizadas para migrar os Elementos Chefes

Com esta query conseguimos obter os dados dos elementos chefes e colocar no documento a sua informação.

Posto isto, importou-se o ficheiro resultante de correr a script para o MongoDB criando assim a coleção projetada.

3.4. Migração do esquema de MySQL para MongoDB

A principal característica da migração de um esquema de mySQL para MongoDB é a eliminação das relações entre tabelas. Apesar de não haver relações em MongoDB, os dados têm que ser armazenados de maneira a que a base de dados responda às necessidades impostas pelo cliente.

Decidimos optar apenas por uma collection pois é possível armazenar informação estruturadamente e que permite que a base de dados mantenha um bom desempenho. Portanto, o documento Elemento, guarda toda a informação do elemento do agrupamento, guardando também toda a informação da sua secção e equipa.

4. Conclusões

Depois de efetuada a migração para MongoDB com o intuito de melhorar a base de dados podemos concluir que nem tudo são melhorias. Num SGBD em MySQL existem relacionamentos que permitem combinar vários dados entre várias tabelas, sendo necessário o uso de JOINs no entanto em MongoDB, isso não existe, tornando necessário garantir a combinação de dados certa no momento da inserção, o que provoca redundância de dados. Para além disso, uma base de dados em MySQL apresenta um esquema mais rígido que uma base de dados em MongoDB, em que para ser construída é preciso definir um esquema e estrutura. Para além disso possui as restrições de integridade, de entidade, de domínio e ainda as gerais que tornam o sistema ainda mais rígido. Pelo contrário o MongoDB não apresenta restrições, o que faz com que possua um esquema extremamente flexível, o que pode provocar inconsistência de dados.

Apesar dos problemas acima definidos, esta migração trouxe também vantagens que neste caso, acabam por compensar a parte negativa.

A base de dados em MongoDB têm muito melhor performance na inserção que a base de dados não relacional, visto que em MongoDB não são verificadas inúmeras restrições de integridade.

Após a migração ficamos numa situação em que possuímos uma base de dados no schema, portanto sem características rígidas. O facto de ser *no schema* implica que caso seja necessário alterar a estrutura, isso implica um gasto de tempo da equipa que desenvolveu a BD muito menor ao comparar com uma base de dados relacional.

A base de dados MongoDB apresenta uma tolerância ao particionamento e uma consistência forte, ou seja, se for feita uma escrita e em seguida a leitura do que foi escrito, é sempre possível ler o resultado.

Apesar de tudo, a disponibilidade não existe em MongoDB, o que faz com que alguns dados estejam inacessíveis.

Referências

- Thomas M. Connolly, Carolyn E. Begg - Database Systems: A Practical Approach to Design, Implementation and Management - 4th Edition. ● https://www.mongodb.com/

Lista de Siglas e Acrónimos

NoSQL - Not only SQL

BD - Base de Dados

SGBD - Sistema de Gestão de Base de Dados

SBDR - Sistema de Base de Dados Relacional

Anexos

I - Exemplos de Queries em MongoDB

II - Script utilizado na migração feito em Ruby.

Anexo 1

Exemplos de Queries em MongoDB

Neste anexo apresentamos a resolução de algumas queries em MongoDB.

1. Query que devolve o nome e designação da secção dos elementos que fizeram a atividade “Debates de Personalidades”;

```
db.elementos.find({"Equipa.Atividade.Designação":"Debates de Personalidades"},{Nome:1,"Secção.Designação":1,_id:0})
```

Fig.4 – "Resolução MongoDB da query 1"

2. Query que devolve o Nin e a Função de todos os elementos que têm mais de 17 anos;

```
db.elementos.find({Idade:{$gt:17}},{Nin:1, Função:1, _id=0})
```

Fig.5 – "Resolução MongoDB da query 2"

3. Query em que se insere uma atividade no elemento com o Nin 8;

```
db.elementos.update({Nin:8},{$push:{"Equipa.Atividade":{"Equipa.Atividade.Designação":"Conversas
Dinâmicas","Equipa.Atividade.Descrição":"Conversas sobre a vida depois dos 10
anos","Equipa.Atividade.Dia":"2017-03-05"}}})
```

Fig.6 – "Resolução MongoDB da query 3"

4. Query em que se insere um elemento na base de dados;

```
db.elementos.insert({
    "Nin":40,
    "Nome":Tiaguinho Batista
    "Idade":8,
    "Sexo":"M",
    "Data de Nascimento":"2005-04-24",
    "Morada":"Peões",
    "Telemovel":255648785
    "Noites de Campo":8,
    "Função":"Cozinheiro",
    "Secção":{"Designação":"Lobitos",
                "Simbologia":"Grande Uivo, Círculo de Conselho, Círculo de Parada",
                "Patrono":"S. Francisco de Assis"},
    "Equipa": {
    "Nome": null,
    "Lema": null,
    "Grito": null,
    "Atividade": null
    }
}
```

Fig.7 – "Resolução MongoDB da query 4"

5. Query em que se elimina um elemento da base de dados;

```
db.elementos.deleteOne({Nin:39})
```

Fig.8 – "Resolução MongoDB da query 5"

Anexo 2

Script

```
require 'json'
require 'mysql2'

DB = 'Agrup424'
HOST = 'localhost'
USERNAME = 'root'
PASSWORD = 'root'

client = Mysql2::Client.new(host: HOST, username: USERNAME, password: PASSWORD,
database: DB) res = [] che = []

elementos = client.query("SELECT E.NIN, E.Nome, E.Idade, E.Sexo, E.DatadeNasc, E.Morada,
E.Telemovel, E.Email, E.NoitesdeCampo, E.Funcao_Designacao AS Funcao,
E.Seccao_Designacao AS NomeSeccao, Seccao.Simbologia AS Simbologia, Seccao.Patrono
AS Patrono, E.Equipa_idEquipa AS IdEquipa, Equipa.Nome AS EquipaNome, Equipa.Lema AS
Lema, Equipa.Grito AS Grito FROM Elemento AS E
                    INNER JOIN Seccao ON Seccao.Designacao = E.Seccao_Designacao
                    INNER JOIN Equipa ON Equipa.idEquipa = E.Equipa_idEquipa;")

elementos.each do |elemento|
   result = client.query("SELECT Atividade.Designacao AS Designação, Atividade.Descricao AS
Descrição, Atividade.dia AS Dia FROM Atividade
                                    WHERE Atividade.Equipa_idEquipa = #{elemento['IdEquipa']}")

   res << {
          'Nin'    => elemento['NIN'],
          'Nome'    => elemento['Nome'],
          'Idade' => elemento['Idade'],
          'Sexo'    => elemento['Sexo'],
          'Data de Nascimento'    => elemento['DatadeNasc'],
          'Morada'    => elemento['Morada'],
          'Telemovel' => elemento['Telemovel'],
           'Email'    => elemento['Email'],
          'Noites de Campo' => elemento['NoitesdeCampo'],
```

```
        'Função' => elemento['Funcao'],
        'Secção' => {'Designação' => elemento['NomeSeccao'], 'Simbologia' => elemento['Simbologia'],
'Patrono' => elemento['Patrono']},
        'Equipa' => {'Nome' => elemento['EquipaNome'], 'Lema' => elemento['Lema'], 'Grito' =>
elemento['Grito'], 'Atividade' => result.to_a}

  }
end

chefes = client.query("SELECT E.NIN, E.Nome, E.Idade, E.Sexo, E.DatadeNasc, E.Morada,
E.Telemovel, E.Email, E.NoitesdeCampo, E.Funcao_Designacao AS Funcao  FROM
                        Elemento AS E
                        WHERE E.Seccao_Designacao IS NULL AND E.Equipa_idEquipa IS
                        NULL;")

chefes.each do |chefe|

  che << {
        'Nin'    => chefe['NIN'],
        'Nome'    => chefe['Nome'],
        'Idade' => chefe['Idade'],
        'Sexo'    => chefe['Sexo'],
        'Data de Nascimento'    => chefe['DatadeNasc'],
        'Morada'    => chefe['Morada'],
        'Telemovel' => chefe['Telemovel'],
        'Email'    => chefe['Email'],
        'Noites de Campo' => chefe['NoitesdeCampo'],
        'Função' => chefe['Funcao'],
        'Secção' => {'Designação' => nil, 'Simbologia' => nil, 'Patrono' => nil},
        'Equipa' => {'Nome' => nil, 'Lema' => nil, 'Grito' => nil,
                                'Atividade' => nil}
}
end res.each do |re| puts
JSON.pretty_generate(re) end
che.each do |ch| puts
JSON.pretty_generate(ch)
end
```