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

Unidade Curricular de Bases de Dados

Ano Lectivo de 2016/2017

Reserva de Viagens: Implementação em MongoDB

João Reis, João Gomes, Tiago Fraga

Janeiro, 2017

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Reserva de Viagens: Implementação em MongoDB

João Reis, João Gomes, Tiago Fraga

Janeiro, 2017

Resumo

Na segunda e última parte do projeto, proceder-se-á à migração de um tipo de base de dados para outro, seguindo procedimentos rigorosos e estudados.
Trabalhando sobre a mesma temática, trata-se duma continuação à primeira parte do trabalho em que a migração terá de obedecer e ser coerente com que já foi elaborado. No final, teremos um produto em duas bases de dados distintas, cada uma com as suas vantagens e funções específicas.

Área de Aplicação: Sistemas de Bases de Dados
Palavras-Chave: MongoDB, Migração, Java, MySQL

Índice

Índice de Figuras

1. Introdução

1.1. Contextualização

De modo a obter o seu bilhete em formato físico, um cliente assim que chegasse à estação de comboios colocava em frente a um sensor o código de barras recebido por correio eletrónico. Inicialmente era algo que era feito instantaneamente, mas com o passar do tempo e com o crescimento da base de dados, os clientes começaram a queixar-se da lentidão do sistema e atrasos como resultado.

Assim que foi comunicada à equipa esta situação, rapidamente se aperceberam qual o seu problema - provêm das queries lentas devido ao elevado volume de dados. A equipa partiu então para discussão de uma solução, tendo portanto um novo desafio à sua frente.

1.2. Apresentação do Caso de Estudo

De modo a solucionar o problema, a equipa decidiu migrar os dados relativos aos bilhetes para uma nova base de dados orientada a documentos, MongoDB. Para tal, criou-se um script em Java que conectado à base de dados MySQL, colocava de forma organizada na base de dados MongoDB.

O pedido da administração da empresa, foi também migrada os dados comboios existentes, de modo poder imprimi-los e coloca-los na estação. Posteriormente, também os utilizadores tiveram o mesmo tratamento em caso de necessidade futura.

O problema agora trata-se apenas da estruturação dos documentos que o script colocaria na base de dados MongoDB.

1.3. Motivação e Objetivos

Tratando-se da última etapa da avaliação prática da Unidade Curricular, este trabalho foi elaborado com o objetivo de aprender bases de dados NoSQL, nomeadamente MongoDB, que se trata de uma linguagem orientada a documentos.

Ao longo da sua criação será possível aperceber de que maneira se distingue e em que é mais útil relativamente a uma base de dados relacional.

1.4. Estrutura do Relatório

Na primeira parte do relatório estabelecer-se-á uma estrutura dos documentos a ser colocados na base de dados Mongo. De seguida, explica-se o método de migração que se irá ser usado de modo obter dados de MySQL e colocar em MongoDB. Procedendo para terceira parte do trabalho, será demonstrada a sintaxe e queries elaboradas em MongoDB. Finalmente, na quarta e última parte do relatório, tirar-se-á conclusões entre o que foi elaborado.

2. Estruturação

Antes de começar a inserção de documentos numa coleção de dados, ter-se-á de definir uma estrutura que os molde. Portanto, nesta secção averigua-se de que modo vai ser armazenada informação em cada uma das coleções existentes – reservas, utilizadores e comboios.

2.1 Reservas

No que toca à estrutura das reservas, optou-se por juntar a informação de várias tabelas da base de dados MySQL.

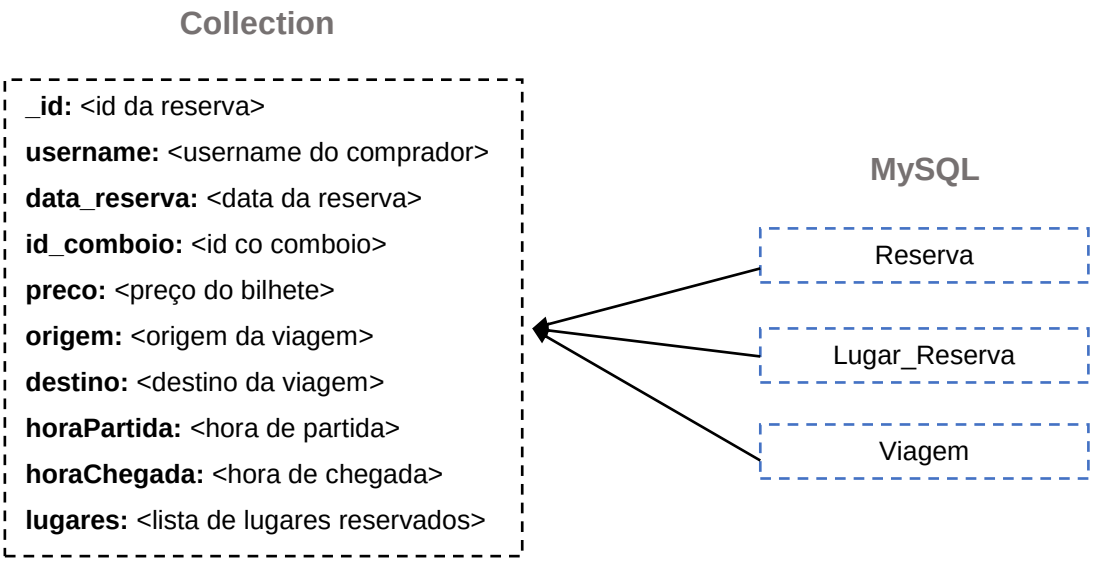

Figura 1 - Estrutura Reserva

Como é possível observar na figura, concilia-se em cada documento da coleção reservas três tabelas existentes na BDMySQL:

- Reserva (_id, username, data_reserva, preco)
- Lugar_Reserva (lugares)
- Viagem: (id_comboio, origem, destino, horaPartida, horaChegada)

2.2 Utilizadores

Relativamente à coleção utilizadores, trata-se apenas duma migração direta dos dados de MySQL, isto é, todos elementos de uma tabela irão fazer parte de um documento.

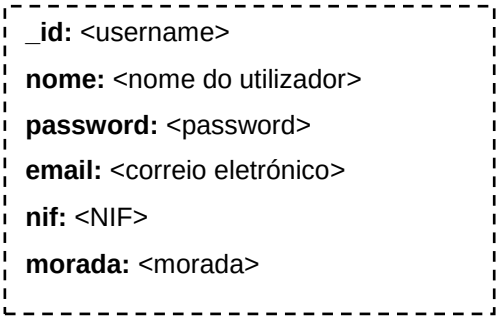

Figura 2 - Estrutura Utilizador

2.3 Comboios

Finalmente, relativamente à coleção Comboios, trata-se de duas combinações de tabelas MySQL, Comboio e Lugar_Comboio.

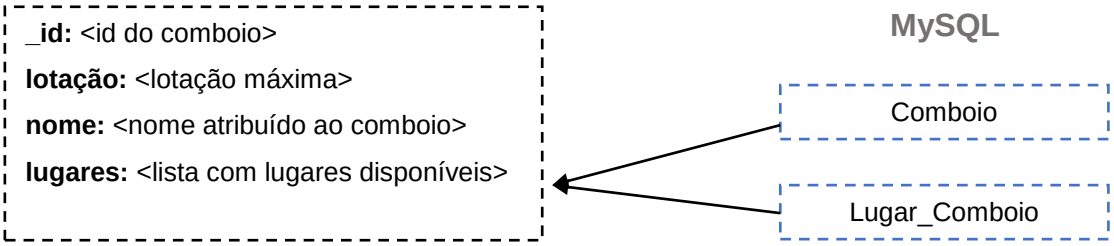

Figura 3 - Estrutura Comboios

3. Migração

De modo a proceder à migração de dados, criou-se um script com a simples função de recolher os dados presentes na base de dados MySQL e coloca-la numa MongoDB.

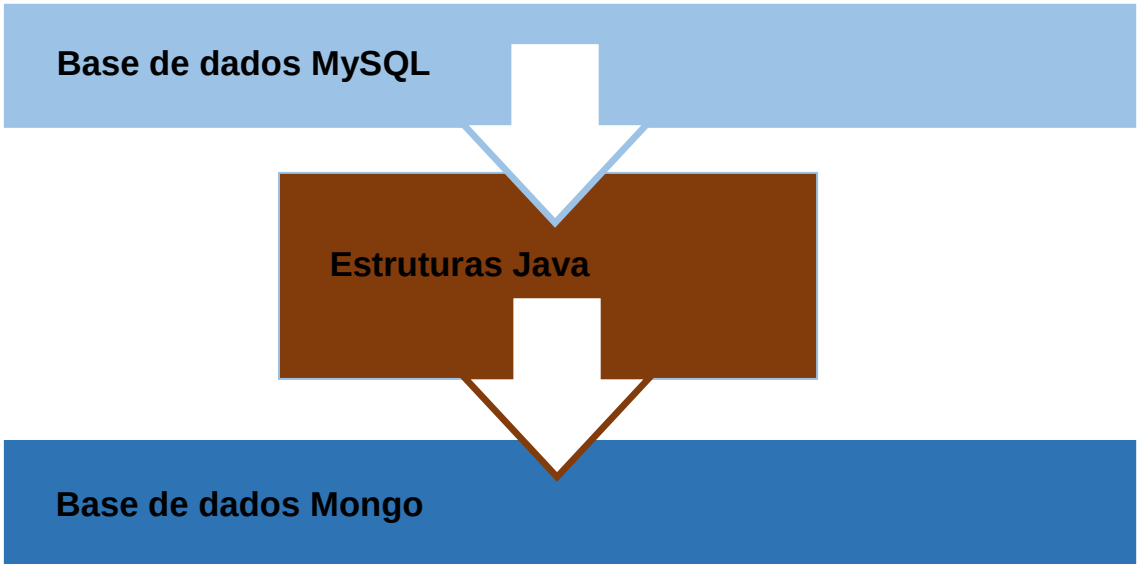

Figura 4 - Funcionamento do script

A tabela acima representada dá uma noção superficial do algoritmo do script criado. Partindo para uma forma mais detalhada do seu funcionamento, divide-se em três etapas:

1. **Recolher dados da base de dados MySQL**
   Inicialmente estabelece-se um conceção com a base de dados MySQL. Caso esta seja feita com sucesso, a aplicação executa *queries* de modo a obter os dados necessários para as estruturas.
2. **Armazenar dados em estruturas Java**
   Criou-se classes em java de acordo os esquemas criados para coleções de MongoDB, após a recolha de dados de MySQL, os mesmos serão armazenados nessas estruturas.
3. **Inserir dados na MongoDB**
   Assim que completa a recolha e armazenamento de dados, estes serão inseridos na base de dados conforme os esquemas anteriormente definidos.

O script é bastante iterativo e fácil de usar: basta executar e tudo é feito automaticamente. Ao longo da sua execução é explicado em tempo real o que está a ser feito e caso ocorra, a origem dos erros. Para o seu funcionamento, é necessário que estejam ligados ambos os servidores Mongo e a base de dados MySQL.

Todos os dados foram migrados usando esta ferramenta, no entanto, caso haja necessidade de migrar algo mais específico, ter-se-á de alterar o código para a necessidade em questão.

4. Queries

Devido às diferentes funções entres as base de dados MySQL e Mongo, esta ultima não será capaz de produzir os mesmos resultados que as *queries* criadas anteriormente. Devido ao seu funcionamento não relacional, não seremos capazes de comparar dados entre diferentes coleções. Como exemplo, não poderemos ver os lugares disponíveis numa viagem. Para tal ser possível seria necessário um *software* que operasse sobre a base de dados. Como as queries serão executadas no cliente Mongo, tal não será possível, e daí algumas queries tiveram de ser adaptadas ou substituídas.

Apresentar a lista dos lugares reservados e das datas das reservas efetuadas por um determinado cliente.

Usando como exemplo, `usr7`:

```
db.reservas.find({username:"usr7"},{data_reserva:1,lugares:1}).p
retty()
```

Resultado:

```
{
  "_id" : "res4",
  "data_reserva" : "2016-01-04 22:14:09.0",
  "lugares" : [
          {
                  "numero" : 2,
                  "carruagem" : 1,
                  "classe" : 1
          }
  ]
}
```

Identificar o preço de uma reserva efetuada por um determinado cliente.

Usando como exemplo, `res3`:

```
db.reservas.find({_id:"res3"},{username:1,preco:1}).pretty()
```

Resultado:

```
{ "_id" : "res3", "username" : "usr6", "preco" : 10 }
```

Fazer uma reserva para uma viagem, para uma determinada data para um lugar específico de uma classe do comboio.

```
db.reservas.insert(
{_id:”res5”,
 username:”user3”,
 data_reserva:”2017-01-02 17:03:54.0”,
 id_comboio:”FASTX1”,
 preco:13,
 origem: “Porto”,
 "horaPartida" : "2016-03-28 11:00:00.0",
 "horaChegada" : "2016-03-28 13:00:00.0",
 "lugares" : [
        {
                "numero" : 4,
                "carruagem" : 1,
                "classe" : 1
        },
        {
                "numero" : 5,
                "carruagem" : 1,
                "classe" : 1
        }
]
}
)
```

Resultado:

```
WriteResult({ "nInserted" : 1 })
```

Apresentar nome do comboio dado um ID

Usando como exemplo, comboio `FASTX1`:

```
db.comboios.find({_id:"FASTX1"},{nome:1}).pretty()
```

Resultado:

```
{ "_id" : "FASTX1", "nome" : "Supersónico" }
```

Insere um cliente na base de dados.

Exemplo:

```
db.utilizadores.insert({
      id:"usr17",
      nome:"João Reis",
      password:"passmongo",
      email:"joao@reis.me",
      nif:119,morada:"Marco da Giesta"
})
```

Resultado:

```
WriteResult({ "nInserted" : 1 })
```

Dado um cliente, devolver e-mail.

Como exemplo, `user17`:

```
db.utilizadores.find({_id:”usr17”},{email:1}).pretty()
```

Resultado:

```
{ "_id" : "usr17", "email" : "joao@reis.me" }
```

5. Conclusões

A parte prática da unidade curricular Base de Dados dividiu-se em duas partes, sendo elas a criação de uma base de dados relacional (em MySQL) e outra orientada a documentos (em MongoDB). Ao longo da elaboração do trabalho foi possível aperceber que os diferentes modelos possuem diferentes características que lhe conferem melhores funcionamentos em diferentes situações.

No que diz respeito a criar uma estrutura para a base de dados, o uso MongoDB torna-a numa tarefa muito mais simples, uma vez que possuem uma estrutura flexível, isto é, as suas coleções não forçam o uso da estruturação de documentos. Esta flexibilidade confere a disponibilidade de igualar as necessidades de uma aplicação, assim como os seus requerimentos de performance. Por outro lado, em MySQL, tivemos de definir previamente a estrutura de modo a que a base de dados iria recolher os dados – o que a torna muito mais trabalhosa, devido à sua rigorosa validação.

Relativamente à construção de queries, a sintaxe da linguagem MongoDB foi muito mais intuitiva porque é semelhante às linguagens que temos vindo a aprender e daí é mais fácil obter resultados pretendidos.

Embora o MongoDB ofereça algumas vantagens, a nível global e conforme os requisitos de um projeto de como a reserva de bilhetes, uma base de dados relacional adequa-se muito melhor devido à possibilidade de efetuar transações complexas. MongoDB não consegue oferecer tantas transações como MySQL, uma vez que precisaria de um *software* a operar por cima. Por outro lado, as partes que se relacionam com interceção de utilizadores, MongoDB já se trataria se uma solução mais adequada.

Referencias

- *MySQL vs MongoDB* [online] Disponível em: https://www.upguard.com/articles/mysql-vs-mongodb [Acedido a 19 de Janeiro]
- *MongoDB and MySQL Compared* [online] Disponível em: https://www.mongodb.com/compare/mongodb-mysql [Acedido a 19 de Janeiro]
- *Query Documents* [online] Disponível em: https://docs.mongodb.com/manual/tutorial/query-documents/ [Acedido a 19 de Janeiro]
- *Connect Java to a MySQL database* [online] Disponível em: https://stackoverflow.com/questions/2839321/connect-java-to-a-mysql-database [Acedido a 19 de Janeiro]
- *MongoDB – Java* [online] Disponível em: https://www.tutorialspoint.com/mongodb/mongodb_java.htm [Acedido a 19 de Janeiro]