INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS

Ciência de Dados

Licenciatura Engenharia Informática

2º Semestre – 2021/2022

Ricardo Jesus Ferreira

ricardojesus.ferreira@my.istec.pt


mongoDB.

Vantagens

- Cada base de dados contém coleções que contêm sucessivamente documentos.
- Cada documento pode ser diferente com um número variado de campos.
- As dimensões e o conteúdo de cada documento podem ser diferentes uns dos outros

Vantagens

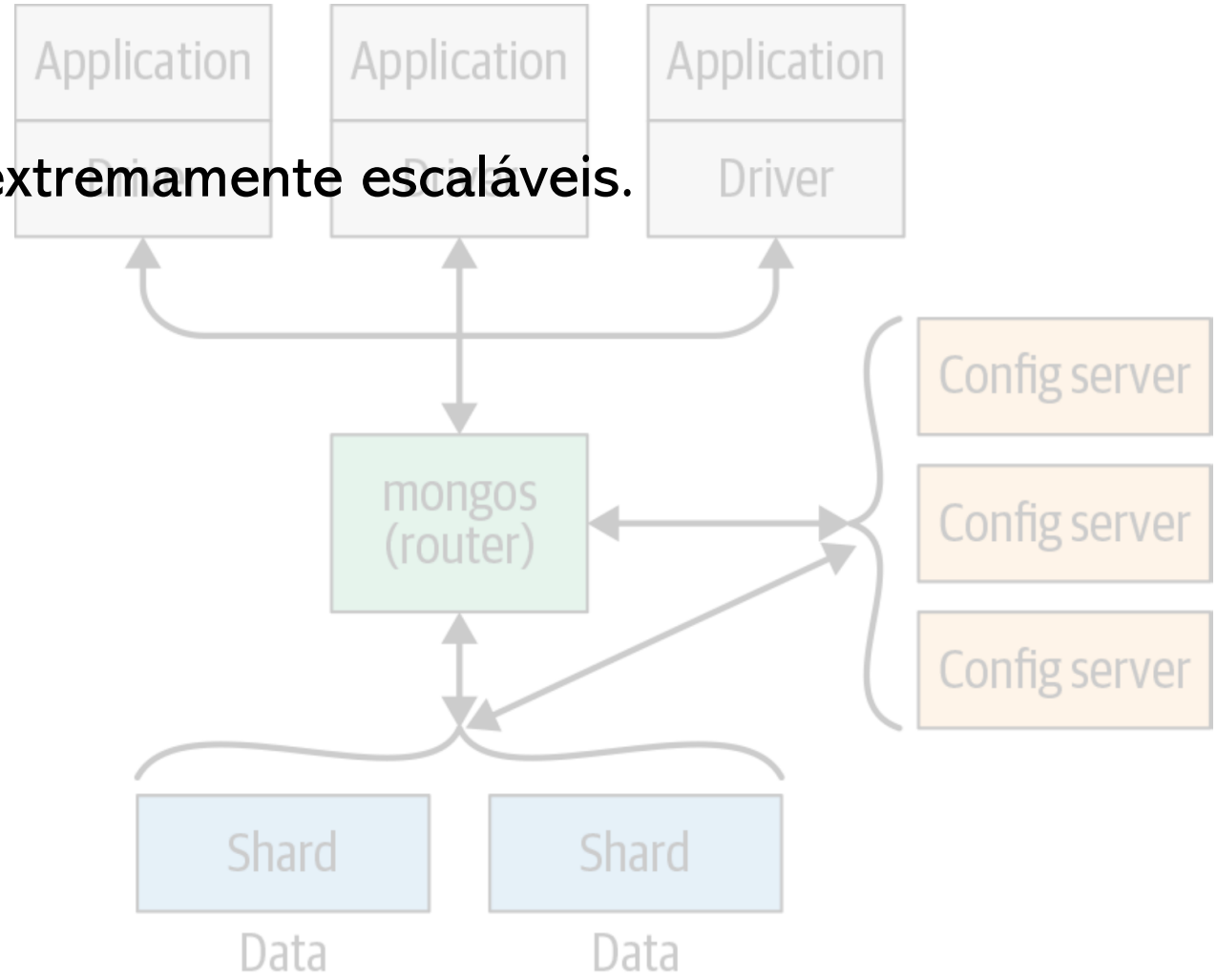
- A estrutura documental está mais em orientada à engenharia e estrutura de um software.

- Os documentos não têm de ter um esquema definido previamente.

- O modelo de dados disponível permite representar relações hierárquicas, armazenar matrizes e outras estruturas mais complexas.

Vantagens

- Escalabilidade
  - Os ambientes em mongoDB são extremamente escaláveis.
- Performance



Desvantagens

- O Mongo DB não é ACID (Atomic, Consistency, Isolation &
- Durability)

- As transações são complexas

- Não existe previsão de implementação de funções armazenadas (Procedures)

MongoDB vs RDBMS

RDBMS	MongoDB	Difference
Table	**Collection**	Em RDBMS, a tabela contém as colunas e linhas que são usadas para armazenar os dados enquanto, em MongoDB, esta mesma estrutura é conhecida como uma coleção. A coleção contém documentos que, por sua vez, contêm Fields, que por sua vez são pares de valores-chave.
Row	**Document**	Em RDBMS, uma linha representa um único item de dados implicitamente estruturado numa tabela. No MongoDB, os dados são armazenados em documentos.
Column	**Field**	Em RDBMS, a coluna denota um conjunto de valores de dados. Estes em MongoDB são conhecidos como Fields.
Joins	**Embedded documents**	Em RDBMS, os dados são por vezes distribuídos por várias tabelas e, de forma a mostrar uma visão completa de todos os dados, uma *join* é por vezes formada através das tabelas para obter os dados. No MongoDB, os dados são normalmente armazenados numa única recolha, mas separados utilizando documentos incorporados. Portanto, não há conceito de *join* em MongoDB.

JSON vs BSON

- O MongoDB utiliza uma variação do JSON como modelo de dados.

- Os documentos JSON são construídos a partir de um pequeno conjunto objetos:
  - Arrays, Objectos (dicionários), valores

- BSON foi concebido para ser uma representação mais compacta e eficiente dos dados JSON e utiliza codificação mais eficiente para números e outros tipos de dados.

Modelação de dados

- Modelo de dados incorporado
  - Neste modelo, incorpora-se todos os dados relacionados num único documento.

- Modelo de dados normalizado
  - Neste modelo, pode consultar os sub-documentos no documento original, utilizando referências

Modelo Dados Incorporado

```
{
    _id: ,
    Emp_ID: "10025AE336"
    Personal_details:{
        First_Name: "Radhika",
        Last_Name: "Sharma",
        Date_Of_Birth: "1995-09-26"
    }
}
```

Modelo Dados Normalizado

Employee

```
{
        _id: <ObjectId101>,
        Emp_ID: "10025AE336"
}
```

Personal_details

```
{
        _id: <ObjectId102>,
        empDocID: " ObjectId101",
        First_Name: "Radhika",
        Last_Name: "Sharma",
        Date_Of_Birth: "1995-09-26"
}
```

Comandos MongoDB

- **Comandos de consulta**, tais como find() e aggregate(), que devolvem informações das bases de dados
- **Comandos de manipulação de dados**, tais como insert(), update() e delete(), que modificam dados dentro da base de dados
- **Comandos de definição de dados**, tais como createCollection() e createIndex(), que definem a estrutura de dados na base de dados
- **Comandos de administração**, tais como createUser() e setParameter(), que controlam as operações da base de dados
-

Criar Base de Dados

- O comando criará uma nova base de dados se não existir, caso contrário, devolverá a base de dados existente.

- Para criar a base de dados, o motor deve conter pelo menos um documento.

```
use DATABASE_NAME
```

Eliminar Base de Dados

```
db.dropDatabase()
```

Eliminar Base de Dados

- O comando db.dropDatabase() é utilizado para apagar uma base de dados existente.

```
db.dropDatabase()
```

Comandos - Create Collection

- O comando db.createCollection (nome, opções) é usado para criar coleção.

```
db.createCollection(name, options)
```

Parâmetro	Tipo	Descrição
Name	String	Nome da coleção a criar
Options	Document	(Opcional) Especificar opções sobre o tamanho da memória e a indexação

Comandos - Drop Collection

- db.collection.drop() é usado para eliminar uma coleção da base de dados.

```
db.COLLECTION_NAME.drop()
```

Comandos - Insert Document

- Para inserir dados na collection do MongoDB, é necessário utilizar o método insert() ou de save().

```
db.COLLECTION_NAME.insert(document)
```

- Se precisar de inserir apenas um documento numa coleção, pode utilizar este método.

```
db.COLLECTION_NAME.insertOne(document)
```

- Pode inserir vários documentos utilizando o método de insertMany(). Para este método é necessário passar uma série de documentos.

```
db.COLLECTION_NAME.insertMany(Array)
```

Comandos - Query Document

- Para consultar os dados, é necessário utilizar o método de find().

- Além do método find() há o método findOne() que devolve apenas um documento.

```
db.COLLECTION_NAME.find()

db.COLLECTIONNAME.findOne()
```

Comandos - Update Document

- O método update() atualiza os valores do documento existente.

```
db.COLLECTION_NAME.update(SELECTION_CRITERIA, UPDATED_DATA)
```

- O método findOneAndUpdate() atualiza os valores do documento existente.

```
db.COLLECTION_NAME.findOneAndUpdate(SELECTIOIN_CRITERIA, UPDATED_DATA)
```

- Estes métodos atualizam um único documento que corresponde ao filtro dado.

```
db.COLLECTION_NAME.updateOne(<filter>, <update>)
db.COLLECTION_NAME.update(<filter>, <update>)
```

Comandos - Delete Document

- O método remove() é utilizado para remover um documento da coleção.

```
db.COLLECTION_NAME.remove(DELLETION_CRITTERIA)
```

Comandos - Projection

- O método find(), aceita um segundo parâmetro opcional que é a lista de campos que pretende obter.

```
db.COLLECTION_NAME.find({},{KEY:1})
```

Comandos - Outros

- Criar index, para optimizar pesquisas
- Obter resultados agregados (soma, media, etc..)

```
db.COLLECTION_NAME.find().limit(NUMBER)

db.COLLECTION_NAME.find().sort({KEY:1})

db.COLLECTION_NAME.createIndex({KEY:1})

db.COLLECTION_NAME.aggregate(AGGREGATE_
OPERATION)
```

Comandos - Relações

Modelo Dados Incorporado

```
{
      "_id":ObjectId("52ffc33cd85242f436000001"),
      ...
      "name": "Tom Benzamin",
      "address": [
            {
                  "building": "22 A, Indiana Apt",
                  ...
            },
            {
                  "building": "170 A, Acropolis Apt",
                  ...
            }
      ]
}
```

```
db.users.findOne(
      {"name":"Tom Benzamin"},
      {"address":1}
)
```

Modelo Dados Normalizado

```
{
    "_id":ObjectId("52ffc33cd85242f436000001"),
    "contact": "987654321",
    "dob": "01-01-1991",
    "name": "Tom Benzamin",
    "address_ids": [
        ObjectId("52ffc4a…000000"),
        ObjectId("52ffc4a…000001")
    ]
}
```

```
result = db.users.findOne(
    {"name":"Tom Benzamin"},
    {"address_ids":1}
    )

addresses = db.address.find(
    {"_id":
        {"$in":result["address_ids"]}
    }
)
```

Bibliografia

- B. Gomez,(2020) “Resolviendo problemas de Big Data”, Alfaomega.
- D. Insua, (2019)“Big data: Conceptos, tecnologías y aplicaciones”, CSIC.
- H. Jones, (2019)“Analítica de datos”, HJ,.
- J. Somed, (2020)“Big Data Analytics”, JLC.
- D. Petković (2020)“Microsoft® SQL Server® 2019 A Beginner’s Guide - Seventh Edition”, McGraw Hill.