Tema 05
Gerenciamento de Coleções no MongoDB

Banco de Dados NoSQL

UNICARIOCA
Centro Universitário

RDBMS x MongoDB

- Em BDs relacionais (RDBMS - *Relational Database Management System*), a linguagem SQL é uma só
- Porém ela é dividida em tipos (sublinguagens), de acordo com a funcionalidade dos comandos

UNICARIOCA
Centro Universitário

Sublinguagem	Descrição	Comandos
DDL - Data Definition Language (Linguagem de Definição de Dados)	Comandos que interagem com os objetos do banco	**CREATE**, **ALTER** e **DROP**
DML - Data Manipulation Language (Linguagem de Manipulação de Dados)	Comandos que interagem com os dados dentro das tabelas	**INSERT**, **DELETE** e **UPDATE**
DQL - Data Query Language (Linguagem de Consulta de dados)	Comandos de consulta	**SELECT** Obs.: Alguns autores consideram o **SELECT** como parte da DML.
DTL - Data Transaction Language (Linguagem de Transação de Dados)	Comandos para controle de transação	**BEGIN TRANSACTION**, **COMMIT**, **ROLLBACK**
DCL - Data Control Language (Linguagem de Controle de Dados)	Comandos para controlar a parte de segurança do banco de dados	**GRANT**, **REVOKE**, **DENY**.

RDBMS x MongoDB

- Inferimos a possibilidade de existência de algumas equivalências entre RDBMS e MongoDB, quanto aos:
  - Termos e conceitos fundamentais
  - Conceitos DDL (*Data Definition Language*) e DML (*Data Manipulation Language*)

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by MongoDB itself)
Database Server and Client	
mysqld/Oracle	mongod
mysql/sqlplus	mongo

RDBMS x MongoDB

Collection Management *(Gerenciamento de Coleções)*

- *Equivalentes aos comandos DDL no SQL*
- Operações como criação de coleções, definição de índices e exclusão de coleções

CRUD Operations *(Operações CRUD – Create, Read, Update, Delete)*, *ou* ***Document Operations*** *(Operações com Documentos)*

- *Equivalentes aos comandos DML em SQL*
- Operações como inserções, atualizações e exclusões de documentos

Query Operations *(Operações de Consulta)*

- *Equivalentes ao DQL no SQL*
- Comandos para consultar os dados

UNICARIOCA Centro Universitário

SQL DDL	Equivalência no MongoDB
CREATE	`db.createCollection()`
ALTER	`db.collection.createIndex()`, `db.collection.dropIndex()` Obs.: Não existe um comando direto de alteração de esquema, mas pode modificar indiretamente ao modificar documentos, ou adicionar/remover índices
DROP	`db.collection.drop()`
SQL DML	**Equivalência no MongoDB**
INSERT	`db.collection.insertOne()`, `db.collection.insertMany()`
DELETE	`db.collection.deleteOne()`, `db.collection.deleteMany()`
UPDATE	`db.collection.updateOne()`, `db.collection.updateMany()`, `db.collection.replaceOne()`
SQL DQL	**Equivalência no MongoDB**
SELECT	`db.collection.find()`, `db.collection.findOne()` Obs.: `find()` retorna um conjunto de documentos que atendem aos critérios da consulta.

UNICARIOCA Centro Universitário

Mongo Playground

- ***Mongo Playground***
  - < *https://mongoplayground.net/* >
  - Excelente para aprender, testar e compartilhar consultas MongoDB sem necessidade de instalar um ambiente local
  - Seções principais:
    - 1. Área de Codificação (Editor)
    - 2. Resultados
    - 3. Documentação e Exemplos
    - 4. Botões de Controle

UNICARIOCA Centro Universitário

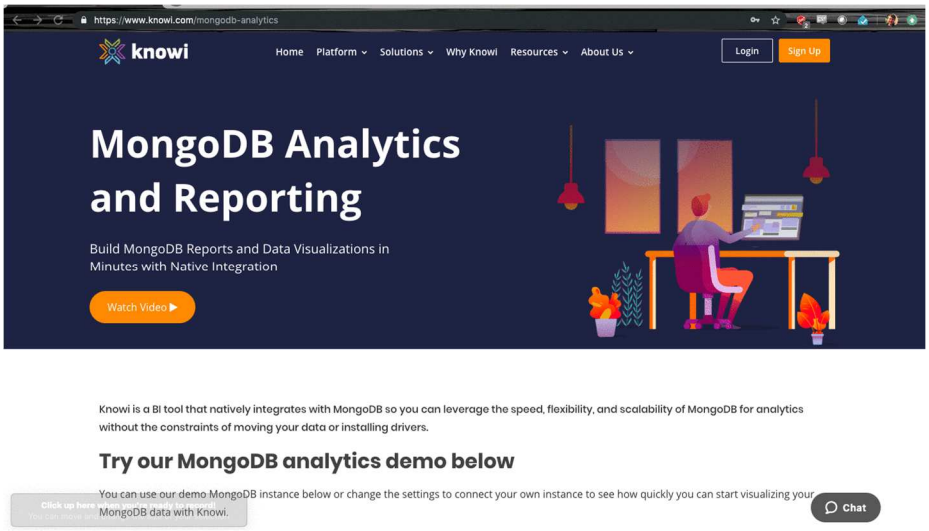
Caixas de seleção no Mongo Playground		
Template	*Single Collections (Padrão)*	Voltado para cenários onde você trabalha com uma única coleção
	Multiple Collections	Para definir e consultar várias coleções simultaneamente
	Mgodatagen	Gera automaticamente um grande número de documentos de acordo com um esquema definido
	Update	Ajustado especificamente para lidar com cenários onde as atualizações de documentos são o foco
	Index	Configura um ambiente voltado para a criação e teste de índices em MongoDB
	Explain	Testa e analisa o desempenho das consultas com o comando `explain()`
Database	*BSON (Padrão)*	Representação padrão dos dados em MongoDB. No MongoPlayground, permite visualizar e insirir documentos no formato JSON comum
	Mgodatagen	Gera dados automaticamente para testar as consultas em grandes volumes de dados sintéticos

UNICARIOCA Centro Universitário

Mongo Playground

- Mongo Playground permite:
  - Adicionar documentos diretamente em coleções fictícias para testar suas consultas
  - Executar consultas
  - Testar comandos de atualização
  - Executar pipelines de agregação
  - Compartilhar consultas criadas por meio de links gerados

Mongo Playground (https://mongoplayground.net/)



Uma alternativa online:
knowi (https://www.knowi.com/mongodb-analytics)

UNICARIOCA
Centro Universitário

Exercício de aula 2

- Desenvolver um sistema para uma pequena livraria online praticando comandos básicos de MongoDB no Mongo Playground, utilizando as funcionalidades suportadas: `find()`, `update()`, `aggregate()` e `explain()`

UNICARIOCA
Centro Universitário

ObjectId

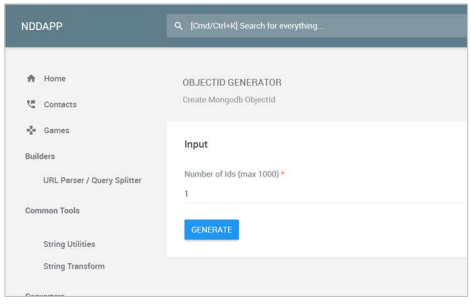
- ***ObjectId*** : Cada documento armazenado em uma coleção requer um campo **`_id`** exclusivo que atua como ***chave primária***
- São pequenos, provavelmente únicos, rápidos de gerar e encomendados
- Possuem 12 bytes de comprimento, consistindo de:
  - Um carimbo de data/hora (*timestamp*)
  - Um valor aleatório de 5 bytes gerado uma vez por processo, exclusivo da máquina e do processo
  - Um contador incrementador de 3 bytes, inicializado para um valor aleatório

Mais em: https://www.mongodb.com/pt-br/docs/manual/reference/method/ObjectId/

ObjectId

- "*ObjectId Generator*"
  - Ex. de utilidade: Treino no Mongo Playground

https://observablehq.com/@hugodf/mongodb-objectid-generator



https://nddapp.com/object-id-generator.html

Data e Hora

- Para expressar ***data e hora***, podemos usar p. ex.:
  - **`"2024-08-26T00:00:00Z"`**
    - String no formato ISO 8601
    - Não é tratada como objeto de data
  - **`ISODate("2024-08-26T00:00:00Z")`**
    - Objeto de data, usado para operações e comparações com datas
  - **`2024-08-26`**
    - Data (ano, mês, dia).
  - **`T`**
    - Delimitador entre a data e a hora
  - **`00:00:00`**
    - Hora, minutos e segundos.
  - **`Z`**
    - Indicador de fuso horário, onde "Z" representa o UTC (Tempo Universal Coordenado)

Parâmetros

- Os ***parâmetros dos comandos*** no MongoDB podem suportar uma ampla gama de operadores e combinações lógicas

`find()`	
`$gt`	Maior que
`$lt`	Menor que
`$in`	Valores contidos em um array
`$and, $or`	Combinações lógicas
`$regex`	Busca usando expressões regulares
`$exists`	Verifica se um campo existe

`update()`	
`$set`	Define (ou atualiza) o valor de um campo específico
`$unset`	Remove um campo do documento
`$inc`	Incrementa um valor numérico
`$push`	Adiciona um valor a um array
`$pull`	Remove um valor de um array

```
// Encontrar pedidos feitos após "2024-08-25"
db.collection.find({ "order_date": { $gt: "2024-08-25T00:00:00Z" } })

// Encontrar pedidos feitos antes de "2024-08-26"
db.collection.find({ "order_date": { $lt: "2024-08-26T00:00:00Z" } })

// Encontrar pedidos de clientes com o nome "Alice" ou "Charlie"
db.collection.find({ "customer.name": { $in: ["Alice", "Charlie"] } })

// Encontrar pedidos do cliente "Alice" onde o preço do primeiro livro seja maior que 30
db.collection.find({ $and: [ { "customer.name": "Alice" }, { "books.price": { $gt: 30 } } ] })

// Encontrar pedidos do cliente "Bob" ou onde o livro "Node.js: The Complete Guide" foi comprado
// Os operadores lógicos como $and, $or, $nor, e $not devem ser seguidos por um array de condições
db.collection.find({ $or: [ { "customer.name": "Bob" }, { "books.title": "Node.js: The Complete Guide" } ] })

// Consultar a autora "Jane Doe" apenas pelo primeiro nome "Jane"
db.collection.find({ "books.authors.name": { $regex: "^Jane" } })

// Buscar pedidos onde o e-mail do cliente termina com "@example.com"
db.collection.find({ "customer.email": { $regex: "@example.com$" } })

// Buscar pedidos onde o campo email não está presente
db.collection.find({ "customer.email": { $exists: false } })
```

Consultas com find()

```
// Atualizar o campo email de "Alice"
db.collection.update( { "customer.name": "Alice" }, { $set: { "customer.email":
"alice_new@example.com" } } )

// Remover o campo email de "Bob"
db.collection.update( { "customer.name": "Bob" }, { $unset: { "customer.email": "" } } )

// Incrementar o preço do livro "MongoDB: A Beginner's Guide"
db.collection.update( { "books.title": "MongoDB: A Beginner's Guide" }, { $inc: { "books.$.price": 5 }
} )

// Adicionar um novo autor ao livro "Advanced MongoDB Techniques"
db.collection.update( { "books.title": "Advanced MongoDB Techniques" }, { $push: { "books.$.authors":
{ "name": "Jane Doe" } } } )

// Remover o autor "John Smith" do livro "Advanced MongoDB Techniques"
db.collection.update( { "books.title": "Advanced MongoDB Techniques" }, { $pull: { "books.$.authors":
{ "name": "John Smith" } } } )
```

Atualizações com update()

Exercício de aula 3

- Cenário:
  - Você está desenvolvendo um sistema para um centro de atendimento médico.
  - Esse sistema precisa armazenar informações sobre os pacientes e os tratamentos que esses pacientes recebem.
  - Cada paciente pode receber um ou mais tratamentos.

UNICARIOCA Centro Universitário

- Tarefas:
  - 1. Elaborar o Banco de Dados em Modelo Incorporado
    - a) Fazer uma consulta simples para buscar um paciente pelo nome
    - b) Fazer uma consulta para buscar todos os pacientes que receberam um tratamento
    - c) Atualizar a idade de um paciente
    - d) Adicionar um novo tratamento para um paciente
  - 2. Elaborar o Banco de Dados em Modelo Normalizado
    - a) Usar o operador **`$lookup`** para combinar as coleções pacientes e tratamentos para obter uma lista de pacientes junto com seus respectivos tratamentos

UNICARIOCA Centro Universitário

UNICARIOCA
Centro Universitário

Melhor Centro Universitário privado do Rio de Janeiro | Melhor EAD do Brasil Conceito 5 do MEC